Hybrid numerical methods based on the lattice Boltzmann approach with application to non-uniform grids

Tobias Horstmann

N° d'ordre NNT: 2018LYSEC027

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de l'École Centrale de Lyon

École Doctorale N° 162
Mécanique Énergétique Génie Civil Acoustique

Spécialité de doctorat : Mécanique des fluides

Soutenue publiquement le 12/10/2018, par
Jan Tobias HORSTMANN

Hybrid numerical method based on the lattice Boltzmann approach with application to non-uniform grids

---

Devant le jury composé de:

Corre, Christophe	Professeur	École Centrale de Lyon	Président du Jury
Ginzburg, Irina	Ingénieure de Recherche	IRSTEA	Rapporteuse
Lamballais, Eric	Professeur	Université de Poitiers	Examinateur
Le Garrec, Thomas	Ingénieur de Recherche	ONERA	Encadrant
Lévêque, Emmanuel	Directeur de Recherche	École Centrale de Lyon	Directeur de thèse
Massot, Marc	Professeur	École Polytechnique	Rapporteur
Mincu, Daniel-Ciprian	Ingénieur	Safran Aircraft Engines	Invité
Ricot, Denis	Ingénieur	Renault	Examinateur

Acknowledgement

This manuscript documents the scientific work that I have conducted over the past three years in the SN2A unit of the *Département Aéroacoustique* [1] at the *Office National d'Études et de Recherches Aérospatiales* (ONERA).

While a PhD thesis mostly relies on working independently, a number of other factors are critical to success. Getting accepted for a project is a good start. Receiving guidance along the way is key. The importance of finding a group of experts from your field that are willing to read your manuscript, discuss your work and, ideally, give you the permission to add a *two-letter prefix* to the name on your door, cannot be understated. With this in mind, I would like to take this opportunity to thank those who have provided me with invaluable support.

Starting with the jury, I would like to thank Irina Ginzburg and Marc Massot for having accepted to review this manuscript, for offering constructive feedback and for recommending future avenues of research within the field. Furthermore, I am also grateful to Christophe Corre for assuming the position of head of jury and to Eric Lambaillais for being part of the committee and sharing his warm words. Denis Ricot is also owed special thanks for having accepted our invitation and for his involvement in interesting discussions throughout the project.

It goes without saying that this thesis wouldn't have been possible without the indispensable guidance of my thesis director, Emmanuel Lévêque, as well as my two thesis supervisors, Thomas Le Garrec and Daniel-Ciprian Mincu. I have very much appreciated your varied inputs to this thesis.

Emmanuel, I'm thankful for the great deal of time that you have dedicated to the supervision of this PhD thesis, despite our offices being some 458 km apart. Your pedagogic approach to teaching me the lattice Boltzmann method, our countless engaging discussions and your open mind for new and different ideas have all been ever so helpful to me.

Thomas, the "walking fluid dynamics encyclopedia", I have tried to absorb as much as possible from your interesting and lengthy(!) lectures, which were often held in my office doorway I might add. Your close guidance and support over the final stretch, including all your efforts to help prepare me for the big day, has been crucial.

Cip, your pragmatic nature and motivational talks certainly kept me going. I am truly grateful to you for having chosen me for this project and, what's more, for having

[1] later merged with other departments to form the *Département Aérodynamique, Aéroélasticité, Acoustique*

given me the freedom to explore different ideas that may have sometimes gone beyond the scope of the initial topic. In this regard, I would also like to thank Eric Manoha for giving me the opportunity to conduct my thesis in his unit.

Last, but by no means least, I would like to thank Lydie Bretaud. Though less scientifically-orientated, you were as vital as anyone to the success of this thesis. The heart and soul of the department, I thank you for your attentive nature, helpfulness and reactivity regarding any administrative or organisational issues.

Mens sana in copore sano and a "healthy" PhD student needs a "healthy" department: I would like to thank everyone in the SN2A and MAXE unit for the positive atmosphere and the memorable times spent together at the countless "pots". Moreover, I would also like to thank the members of the running group for the much needed sporting relief, as well as the Friday beers group for letting me take my mind off the LBM at the end of the week.

This acknowledgement wouldn't be complete without a special mention to my daughter. I was very appreciative of how she learnt to sleep through much of the night at just six weeks of age, which came in very handy when I was writing up the manuscript. Finally, I would like to thank my family for agreeing to the long journey from Münster/Stuttgart/Bonn/Marseille to attend my doctoral viva and to my wife for the support along the way.

This research work has been financially supported by the Bpifrance in the framework of the"Développement de l'Economie Numérique, Programme d'Investissement d'Avenir : Calcul Intensif et Simulation Numérique" (FUI16, Grant N. P3543-189834).

Jan Tobias Horstmann
Sceaux, December 2018

Contents

Nomenclature

Roman characters

$\boldsymbol{M}^{(n)}$	moment of order n
$\boldsymbol{S}$	strain rate tensor
$\boldsymbol{u}$	flow velocity
$\boldsymbol{v}$	peculiar velocity
$\mathcal{R}$	spatial discretization operator
c	characteristic particle propagation speed
c_0	physical speed of sound
c_s	lattice speed of sound
f_α	genuine probability distribution functions
g_α	mesh-specific probability distribution functions
m	refinement factor
m_α	entries of the vector of moments of g_α
q	number of discrete velocities
w_α	weight coefficients
M	Mach number

Greek characters

α	velocity space index
$\boldsymbol{\sigma}^{'}$	deviatoric stress tensor
$\boldsymbol{\xi}$	velocity vector in continuous velocity space

$\boldsymbol{\xi}_\alpha$ velocity vector in discrete velocity space

δ_{ij} Kronecker delta

ϵ Knudsen number

$\mu^{'}$ bulk viscosity

μ dynamic viscosity

ν kinematic viscosity

Ω collision operator

ω relaxation frequency

ψ collision invariant

ρ density

τ relaxation time associated with f_α

τ_g relaxation time associated with g_α

ζ scaling parameter

Indices and exponents

$\square^c$ coarse

$\square^f$ fine

$\square_l$ lattice units

$\tilde{\square}$ non-dimensional

$\widehat{\square}$ post-collision state

Acronyms

BE Boltzmann Equation

BGK Bhatnagar Gross Krook

CFL Courant Friedrichs Lewy

DVBE Discrete Velocity Boltzmann Equations

FV-LBE Finite-Volume Lattice Boltzmann Equations

FV-NSE Finite-Volume Navier-Stokes Equations

HLBM Hybrid Lattice Boltzmann Method

LBE Lattice Boltzmann Equations

LBM Lattice Boltzmann Method

LGA Lattice Gas Automata

LHS Left-Hand Side

MME Mass Momentum Energy

MRT Multi-Relaxation Time

NSE Navier-Stokes Equations

QUICK Quadratic Upstream Interpolation for Convective Kinematics

RHS Right-Hand Side

RK Runge-Kutta

SPL Sound Pressure Level

Abbreviations

ac acoustic

cont continuous

cst constant

eq equilibrium

neq non-equilibrium

num numerical

phy physical

Chapter 1

Introduction

Context

During more than seventy years of dedicated research in the field of aerospace science, the Office National d'Etudes et de Recherches Aérospatiales (ONERA) has acquired an international reputation for its competences in the field of computational fluid dynamics (CFD). Over the years, a portfolio of sophisticated CFD solvers, including *elsA*, *CEDRE* and *FUNk*, has been developed that serve engineers as high-fidelity tools for a variety of aerodynamic, aeroelastic and aeroacoustic applications. Based on the macroscopic conservation of mass, momentum and energy, these solvers belong to the class of conventional, or rather continuum-based solvers. A few decades ago, an alternative to the classical approach was postulated, known as the lattice Boltzmann method (LBM). It breaks with the assumption of a continuum by modelling the molecular dynamics in a simplified fashion. Ten years ago, in order to engage in this new technology, ONERA decided to collaborate with a french consortium built around the automotive and aerospace heavyweights, Renault and Airbus, the engineering company CS and two CNRS academic laboratories. The objective of this consortium was the development of a **La**ttice **B**oltzmann **S**olver (LaBS). In 2015, the partners launched the funded project CLIMB (**C**omputationa**L** methods with **I**ntensive **M**ultiphysics **B**oltzmann solver; FUI16, Grant N. P3549-189834) with the intention to push the development of the LaBS software towards a mature and competitive product. The present PhD thesis was funded within this project.

Intuitively, one would think that a kinetic, discrete particle approach is computationally more expensive due to the increased number of degrees of freedom. For the lattice Boltzmann method, this is, nevertheless, not true. First of all, discrete particles are represented by probability distribution functions, which reduces the number of unknowns to a manageable set of mesoscopic variables. Secondly, the distribution functions stream in discrete phase space in such a way that during one time step, the

distance between two defined points in phase space is covered. From a computational point of view, this type of Lagrangian advection is expressed as a simple index shift in memory. Mathematically, this type of streaming constitutes an exact solution to the Lagrangian derivative along constant velocity vectors that define a lattice. Advection in the lattice Boltzmann method is therefore extremely efficient and non-dissipative [1]. In addition, the compact streaming and the local collision at the nodes make the lattice Boltzmann approach particularly suited for massively parallel computations. As a kinetic scheme, the lattice Boltzmann method is naturally adapted to simulate multiphase/multi-component flows as well as small-scale flows outside the limit of the continuum assumption. Of course, there is also a "dark side of the LB moon" [Succi, 2015]. The streaming along the lattice vectors confines the discrete space points to the lattice nodes (i.e. discrete points in velocity space). The spatial discretization is therefore imposed by the lattice, which leads to uniform Cartesian grids. Moreover, the discreteness of velocity-space introduces an error term that scales with the Mach number. The lattice Boltzmann method is therefore best suited for quasi-incompressible flows. And last but not least, the method is inherently transient, so it is not the best choice for steady-state computations.

Among various fields of potential applications [Succi, 2015], the lattice Boltzmann method has particularly found its way into the field of aeroacoutics. We identify two main reasons. First, the discrete presentation of the advection term in an exact and efficient manner allows the propagation of an acoustic signal over long distances at reasonable costs. Second, the biggest flaw of this method, the limitation to the low Mach number regime, does not actually jeopardize many aeroacoustic application. A car usually travels at a maximum speed of 0.1 M and even aircrafts, during landing and take-off, when the environmental noise impact is the highest, do not exceed 0.3 M. This value lies still within the acceptable Mach number range (< 0.4 M). Under the assumption that noise generating structures are often complex geometries (landing gear, high lift devices, etc.), one may add the straightforward meshing in this approach, as a further reason for the attractiveness of this method for aeroacoustic applications.

There is, however, one drawback that affects aeroacoustic simulations in particular: the limitation to uniform Cartesian grids. The grid size is determined by the smallest scales to resolve. For a given frequency f, one may therefore introduce the length scale $\lambda = \boldsymbol{u}/f$ as a measure for these scales, where $\boldsymbol{u}$ denotes the flow velocity. In analogy, the wavelength of the respective acoustic waves may be given as $\lambda_{ac} = c_0/f$, where c_0 denotes the speed of sound. Simulating the aeroacoustics of a car, these two length scales differ by one order of magnitude. A strong interest therefore exists to lower the resolution of the spacial mesh outside the regions of the small-scale turbulent flow. Due to the limitation to uniform Cartesian grids, mesh refinement in the LB

[1] Other sources of numerical dissipation may be introduced into the scheme due to the discretization of velocity space and the collision term.

method is usually achieved with multi-scale lattice Boltzmann schemes. The domain is decomposed into multiple subdomains, and in each subdomain, the lattice Boltzmann equation (LBE) is solved at a different resolution in space and time. For the sake of connectivity, it is necessary that the resolution between two neighbouring subdomains differs by the integer factor 2^n, where n is usually one. The classical multi-scale lattice Boltzmann method is therefore confined to quadtree grids in 2D and octree grids in 3D. Although it adds a rather complex element to the otherwise very simple, two-step, *stream and collide* algorithm, many commercial and non-commercial LB solver rely on such multi-scale scheme, which has been successfully applied to aerodynamic simulations (cf. Fig 1.1).

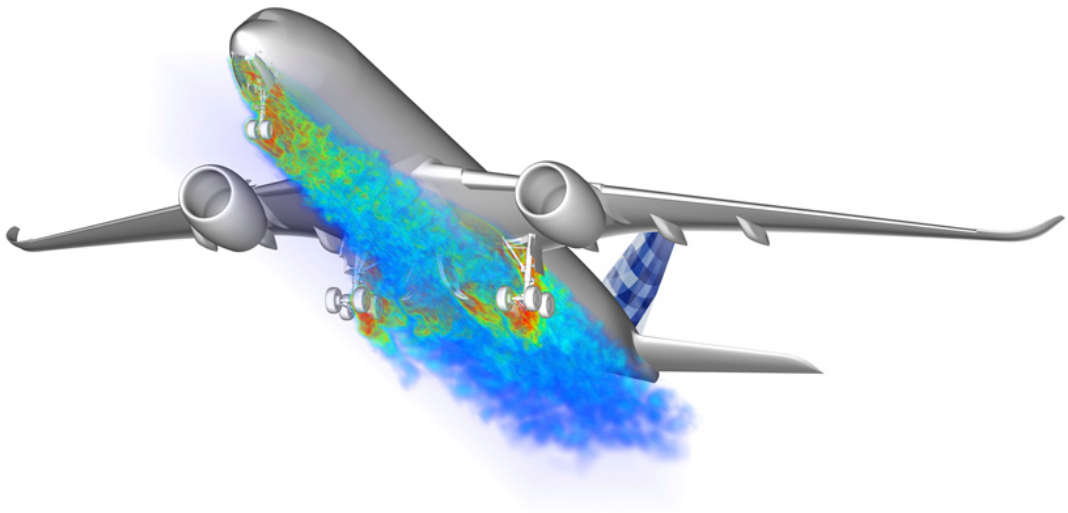


Figure 1.1: Vorticity field around the landing gear of an Airbus A320neo computed with ProLB (former LaBS) using multiple resolution domains.

The same can not be said for aeroacoustic simulations. The body of literature that presents validated multi-scale, or rather multi-resolution LB algorithms for aeroacoustic simulations is very limited [Gendre et al., 2017]. The acoustic waves are more sensitive to approximation errors that are inevitably committed at the refinement interface. The pressure level usually varies in a range of only a few pascal (1 Pa $\approx$ 94dB SPL). At this scale, anomalies become visible, such as the emission of spurious noise from the refinement interface (cf. Fig. 1.2). Identifying the cause for this emission constitutes a non-trivial task. It is common knowledge that an abrupt coarsening of the computational mesh leads to the reflection of unsupported structures back into the fine

domain. The non-physical signal shown in Fig. 1.2, however, radiates in all direction. Alternative explanations may be a violation of mass conservation or the excitation of non-physical modes. The quadtree, or rather octree grid refinement algorithm contains various stages, e.g. interpolation in space and time, filtering, rescaling, reconstruction, etc., where either of these phenomena may be triggered.

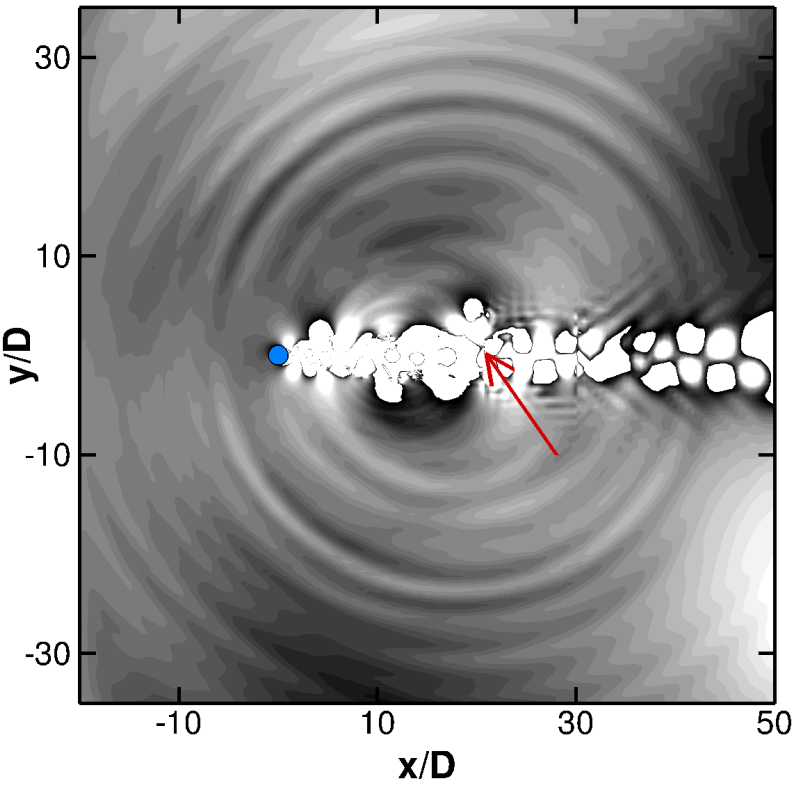


Figure 1.2: Pressure fluctuations -5 Pa $< p' < 5$ Pa around a 2D cylinder. The origin of spurious noise generation at a refinement interface is indicated by the arrow.

Objectives

The present PhD thesis has been defined in the light of such anomalies that have been observed with the LaBS solver. Solving this problem is a high priority for the LaBS consortium, since it weakens the performance of the software in aeroacoustic applications. Consequently, a veritable task force was convened to tackle the issue of mesh refinement. In this context, a few month earlier another PhD project had started in the aeroacoustic department at Renault with the same objective. Moreover, the work group "mesh transition" was initiated to discuss about this issue on a regular basis.

Most attempts to improve the mesh refinement algorithm stayed within the numerical framework of LaBS. That is to say, an octree grid with a cell-vertex data structure; in general, nodal values can either be stored in the cell-vertices or at the cell-centres. The reasons are clear. Even though the development and testing of new mesh refinement algorithms was mostly carried out on simplified testing platforms, the ultimate objective was the implementation into LaBS. Some testing was also done in the LaBS source code, directly.

During this thesis, an in-house LB prototype was developed that presents a two-dimensional, simplified version of LaBS with the ability of mesh refinement. The motivation for the development of such demonstration solver was manifold. First of all, it is considered valuable from a pedagogical point of view, because it helps to obtain a good understanding of the method. Secondly, LaBS is an advanced, three-dimensional lattice Boltzmann solver that is tuned by computer scientist for optimal performance on parallel multi-processor clusters. This results in an intricate structure in which the implementation and validation of new algorithms is a time-consuming endeavour. Ideas for new mesh refinement algorithms, however, were abundant and the testing and primary validation of those ideas are evidently more straightforward in a simple two-dimensional solver allowing for a change in resolution. Another reason was the fact that a demonstration code is more flexible. It leaves more space for the development of new ideas, because the architecture of the code can easily be changed. The practical work of the present thesis therefore started out with the development of a two-dimensional testing platform for grid refinement techniques in the lattice Boltzmann method.

Initial attempts to improve the refinement algorithms and reduce the spurious noise emission were based on the multi-scale lattice Boltzmann method in a cell-vertex formulation (LaBS framework). In this context, a partial reconstruction strategy was tested, which did not lead to an improvement of the mesh refinement algorithm. However, a PhD thesis is also an opportunity to explore novel and disruptive ideas that may even leave the predefined context.

Drawing on the in-house LB solver that was not confined to the numerical framework of LaBS, the issue of mesh refinement was approached in a very global and unbiased manner. It was pointed out that even though the exact mechanisms for the generation of non-physical noise are unknown, it is certainly linked to quadtree (2D) or rather octree (3D) grids, introducing an abrupt transition and a solution miss-match due to different discretization errors. The following question therefore arose: **is it possible to avoid octree grids without (entirely) sacrificing the efficiency and low-dissipation of the LB method?**

The only way to avoid an octree data structure is to discretize physical space and velocity-space independently. Otherwise, the spatial mesh is imposed by the lattice. This is achieved by discretizing the advection term in the Eulerian framework, such that the space and time derivatives are treated separately. In that case, physical space

may be discretized in any arbitrary way (e.g. unstructured, gradually refined, body-fitted grids). Of course, certain advantages of the classical LBM are compromised: first, the advection term is no longer solved in an exact manner, which raises the level of numerical dissipation. Secondly, the efficiency of the streaming that translates into a simple index shift in memory is lost, due to the evaluation of an additional flux term.

The Eulerian approach therefore solves one problem (abrupt mesh refinement), but creates another (dissipation, costs). An interesting question is: **do these problems coincide *geographically* in an aeroacoustic simulation?** The answer is no. Let us decompose a computational domain into a smaller near field, aerodynamic region around a geometry and an larger, far-field aeroacoustic region. Dissipation and low efficiency become a particular problem in the far field region, due to the weakness of the aeroacoustic signal and the size of the domain. Mesh refinement, on the other hand is not required ($\lambda_{ac} = \text{cst}$). In the near field region, the situation is reversed. Turbulent, energetic structures are more resistant to numerical dissipation. Also a certain amount of dissipation is desirable to stabilize the computation. On the other hand, the required change in resolution to adapt from the turbulent scales λ to the aeroacoustic scales λ_{ac} is entirely confined to this region.

These considerations gave rise to the question of whether it is possible to combine the classical lattice Boltzmann equation with a discretization in the Eulerian sense. This work gives some answers to this question by presenting the first, second-order accurate, hybrid lattice Boltzmann method (HLBM) combing these approaches. It is demonstrated that the HLBM constitutes a valuable alternative to the classical multi-scale Boltzmann scheme. This work led to a publication in the November 2017 issue of the Journal of Computational Physics (Horstmann et al., 2017).

Structure of the manuscript

The first chapter of this thesis, **"Fundamentals of LBM"** may be considered as a theoretical handbook to a classical LB solver based on the *stream and collide* algorithm. It lays emphasis on those aspects that were considered important for the understanding of the classical LB method, also in the context of continuum mechanics, and its practical implementation. A LB example code follows the presentation of the theory. The reader will realize that the actual *stream and collide* kernel, the heart piece of nearly any commercial and non-commercial LB solver, can be coded within a few lines. This illustrates, one more time, the efficiency of this approach.

When introducing a numerical method, it is usually common to depart from a continuous equivalent. This was intentionally avoided in the first chapter to underline the fact that the lattice Boltzmann method is a prominent descendent of another discrete, kinetic scheme: the lattice gas automata. It can thus be derived under a discrete consideration of time, space and velocity-space. As the name suggests the

LB method has inherited the concept of a lattice and the streaming of distribution functions between lattice nodes, expressed as a perfect shift. Along with this comes the limitation to uniform Cartesian grids and the obligation to use a quadtree (2D) and octree (3D) data structure when multiple resolutions are desired.

Nevertheless, any discrete method has a continuous counterpart. The time, space and velocity-space continuous equivalent to the lattice Boltzmann equation (LBE) is the Boltzmann equation (BE). If only the velocity-space is discrete one obtains the discrete velocity Boltzmann equation (DVBE). Obviously, the LBE can be obtained by discretizing the DVBE in space and time. Presenting a linear transport equation, classical approaches such as finite-difference, finite-volume and finite-element schemes (Eulerian sense) are, *a priori*, just as much a discretization method of choice, as using the method of characteristics (Lagrangian sense). Nevertheless, only the method of characteristics yields the classical second-order lattice Boltzmann equation (LBE) [2]. The other approach yields the finite-different lattice Boltzmann equation (FD-LBE), the finite-volume lattice Boltzmann equation (FV-LBE), and the finite-element lattice Boltzmann equation (FE-LBE), respectively.

Of course, the lattice Boltzmann method based on the streaming between lattice nodes is a very elegant way to describe the advection of the particle distribution functions. It is not without reason that the *stream and collide* algorithm is implemented in many solvers. However, it also comes with some inconveniences, as previously discussed. The lattice Boltzmann approach is often reduced to the well-known *stream and collide* scheme, a common belief that is typically attributed to its origin in the lattice gas automata. The concept of streaming constitutes its principal heritage. Here, we would like to provide the reader with a global view on different numerical schemes. The second chapter, entitled **"Space-time discretization of the discrete velocity Boltzmann equation"** sheds some light on the different possibilities to discretize the DVBE. Of course, the classical LBE (*stream and collide* algorithm) will be derived, but other models, such as the FV-LBM will be presented, too. It will become clear that the mesh refinement by a factor two is only a result of a very particular space-time discretization of the DVBE.

The last chapter, **"HLBM– A new mesh refinement algorithm"** presents the algorithms that were developed during the present thesis as an alternative to conventional (quadtree/octree) mesh refinement in the classical LB approach. The single stages of the algorithm are elaborated. Results are presented for two-dimensional, periodic test cases.

[2]Using a finite-difference approach, only the first-order LBE can be obatined, which will be demonstrated in the manuscript.

Related articles

J T Horstmann, T Le Garrec, D-C Mincu, and E Lévêque. **Hybrid simulation combining two space–time discretization of the discrete–velocity Boltzmann equation.** *Journal of Computational Physics* 349:399-414, 2017

26th International Conference on Discrete Simulation of Fluid Dynamics, *oral presentation:* **Hybrid simulation combining two space–time discretization of the discrete–velocity Boltzmann equation.** 10-14 July 2017, Erlangen, Germany.

Chapter 2

Fundamentals of LBM

2.1 Origin

The lattice Boltzmann method (LBM) is a descendant of the lattice gas automata (LGA). As the names suggest, LBM has inherited the concept of a lattice, which shall be elucidated in the following. LGA rely on the microscopic approach, in which the fluid system is represented by point particles instead of continuous state variables. These particles are object to propagation between neighbouring nodes of a regular lattice and collision at the nodes. In the real world, the particle speed between two collisions constitutes an independent variable that is *a priori* continuous. Here, the lattice enforces a discrete representation of the velocity space. In a particular class of LGA, the FHP models [Frisch et al., 1986], named after their creators Frisch, Hasslacher and Pomeau, the lattice, for example, may be hexagonal as seen in Fig. 2.1.

Common to lattice methods is the assumption that during a time interval Δt, a particle covers exactly the distance Δx between two adjacent lattice sites. For a hexagonla lattice, we therefore obtain the relation

$$\boldsymbol{\xi}_\alpha = \frac{\Delta x}{\Delta t}\hat{\boldsymbol{e}}_\alpha,$$

where $\hat{\boldsymbol{e}}_\alpha = \boldsymbol{e}_\alpha/|\boldsymbol{e}_\alpha|$ is the unit vector of the direction of motion $\alpha = \{1, 2, 3, 4, 5, 6\}$ and $\boldsymbol{\xi}_\alpha$ is the discrete velocity attributed to a particle state $\boldsymbol{n}_\alpha(\boldsymbol{\xi}_\alpha, \boldsymbol{x}, t)$ that is either 0 or 1. Neglecting collision, the motion of particles in discretized form may be written as

$$n_\alpha(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t) = n_\alpha(\boldsymbol{x}, t). \tag{2.1}$$

Eq. (2.1) constitutes the well-know streaming step that is characteristic to LGA and was passed on to LBM. Many LBM enthusiasts see a great advantage in this straightfor-

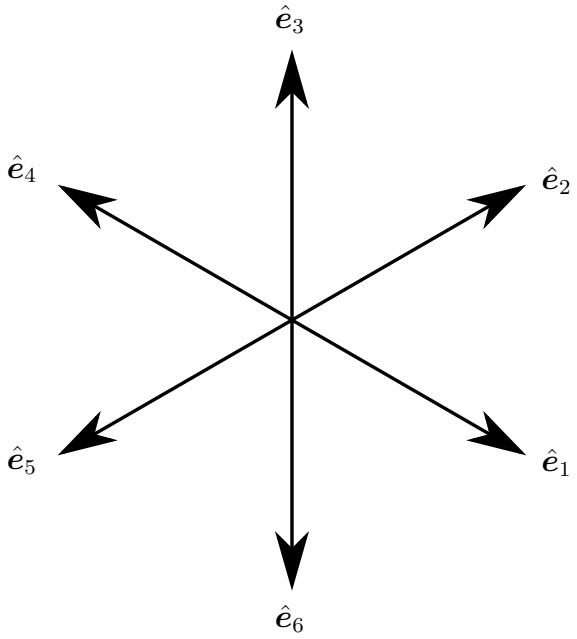


Figure 2.1: Hexagonial lattice of the FHP model with six discrete velocity directions in two-dimensional velocity space.

ward dependence of the physical space and velocity-space discretization, as it makes the generation of the spatial mesh unnecessary. This would explain the sometimes misleading claim that LBM does not require a mesh. The lattice dictating the discretization in space, however, imposes severe constraints on the mesh geometry and limits the method to uniform hexagonal, or more commonly, Cartesian grids. In chapter 3, we explicitly address this limitation and explain how physical space can, nevertheless, be discretized independently.

The point particles in the LGA were represented by Boolean variables, where *true* indicates the presence of a particle with a certain (discrete) velocity at a particular site and *false* represents its absence, accordingly. This clearly demonstrates the interest of the method to represent a fluid by its dynamics at the molecular level [Wolf-Gladrow, 2000]. Obviously, a true representation of common fluid dynamics at the molecular level turns out to be quite unrealistic, given that 1 cm^3 of air already contains about 2.7×10^{19} particles. The dramatic underrepresentation of molecules made the LGA method intrinsically noisy and required averaging over large subdomains and time intervals to obtain reasonably resolved densities. Remedy to this problem was provided by [McNamara and Zanetti, 1988], who suggested to replace the point particles by distribution functions representing a density of particles – an abstraction that forms the basis of Boltzmann's kinetic equation. The transported quantities are thus no longer Boolean variables but real values that indicate a spatial density of probability to find a particle of speed $\boldsymbol{\xi}$, at a certain time t around position x. The streaming step

in discrete velocity space without collision therefore becomes

$$f_\alpha(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t) = f_\alpha(\boldsymbol{x}, t), \tag{2.2}$$

where f_α is referred to as probability distribution function or simply population. In literature, this is considered as the birth of LBM. Due to the statistical view on the microscopic processes, this method is usually referred to as mesoscopic approach. In Eqs. (2.1) and (2.2), the interaction between particles was neglected for the sake of simplicity. Taking the interaction of particles into account, a single equation representation of the molecular processes is given by

$$f_\alpha(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t) = f_\alpha(\boldsymbol{x}, t) + \Omega\left(f_\alpha(\boldsymbol{x}, t)\right), \tag{2.3}$$

where Ω is the collision operator. A separate consideration of propagation and collision, however, makes sense as the collision occurs locally at the lattice nodes. For this reason, following two step *stream and collide* representation is often found

$$\begin{aligned} &\textbf{collision:} && \widehat{f}_\alpha(\boldsymbol{x}, t) = f_\alpha(\boldsymbol{x}, t) + \Omega\left(f_\alpha(\boldsymbol{x}, t)\right) \\ &\textbf{streaming:} && f_\alpha(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t) = \widehat{f}_\alpha(\boldsymbol{x}, t), \end{aligned}$$

where $\widehat{\square}$ denotes a post-collision state. The interaction of particles in the LGA models were governed by some rudimentary collision rules that obey the conservation of mass and momentum. For the FHP model, only two and three-particle collision were authorized. Moreover, the direction to enter a lattice site was eliminated for exit. This results in two possible outcomes for the two particle collision and a single post-collisional configuration for the three particle interaction (Fig. 2.2).

Even though the first LB models were based on these collision rules, it was soon realized that for more sophisticated lattices models (FCHC with 24 particles) or the addition of a third dimension in space, the resulting scattering matrix becomes impractical. A first improvement presented an enhanced collision, where the nonlinear collision operator is simplified by linearization about a reference equilibrium state f_α^{eq} [Higuera and Jiménez, 1989, Higuera et al., 1989]; i.e. $\Omega\left(f_\alpha(\boldsymbol{x}, t)\right) = \boldsymbol{A}_{\alpha\beta}\left(f_\alpha\left(\boldsymbol{x}, t\right) - f_\alpha^{eq}\left(\boldsymbol{x}, t\right)\right)$. Here $\boldsymbol{A}$ is referred to a scattering matrix. A further simplification of the method that led to the nowadays most common LB model was achieved by Qian et al. [1992] with their suggestion to model the collision between particles simply as a relaxation towards the equilibrium state: $\boldsymbol{A} = -\delta_{\alpha\beta}\tilde{\omega}$. The scattering matrix is thus replaced by a single relaxation parameter $\tilde{\omega}$, which is defined as the inverse of the relaxation time τ normalized by the time step Δt, i.e. $\tilde{\omega}^{-1} = \tilde{\tau} = \dfrac{\tau}{\Delta t}$. Eq. (2.3) thus becomes

$$f_\alpha(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t) = f_\alpha(\boldsymbol{x}, t) - \tilde{\omega}\left(f_\alpha(\boldsymbol{x}, t) - f_\alpha^{eq}(\boldsymbol{x}, t)\right), \tag{2.5}$$

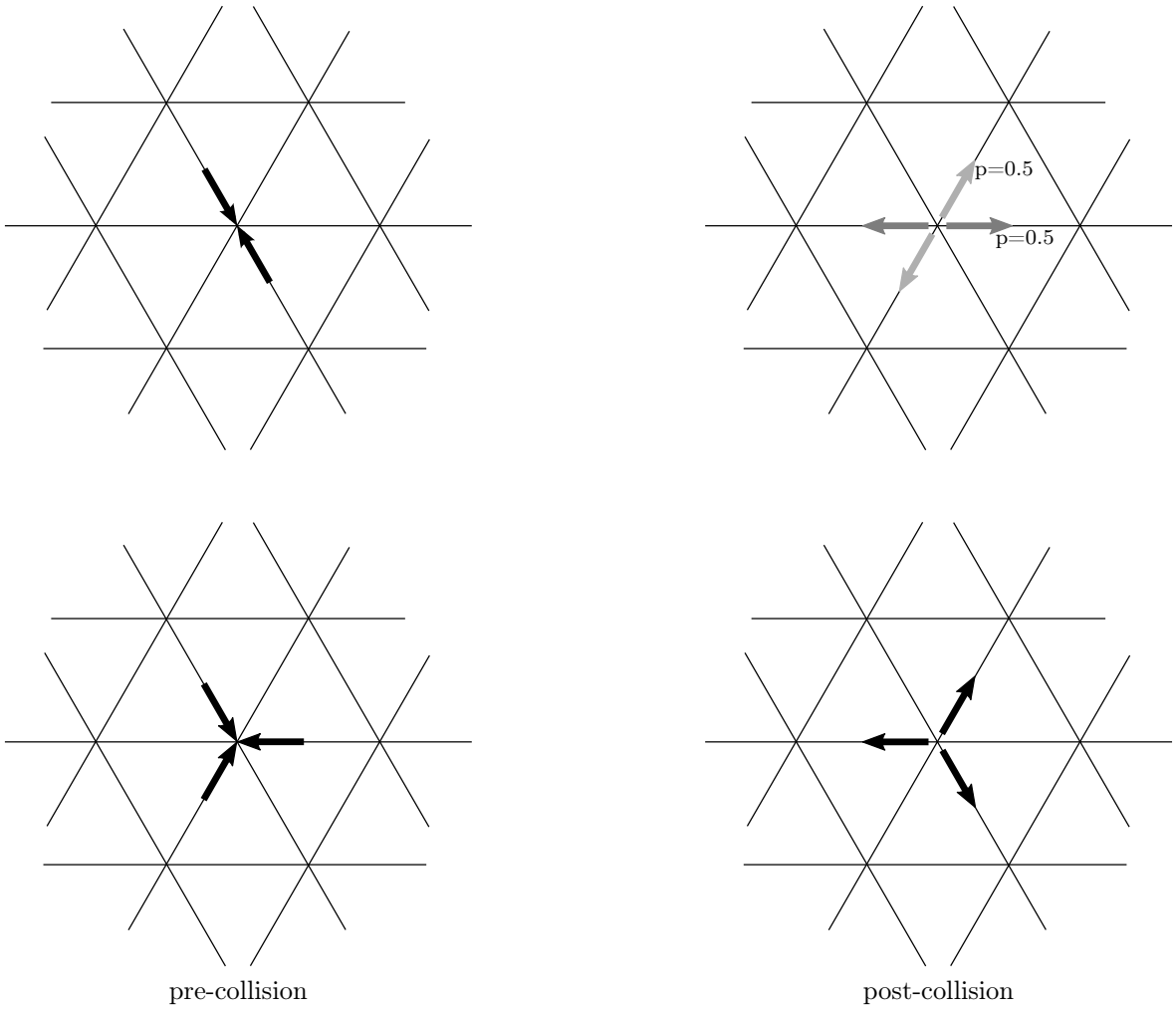


Figure 2.2: Two- and three-particle collision in the FHP model. p denotes the probablility of a particular post-collision state.

where f_α^{eq} denotes the discrete Maxwell-Boltzmann equilibrium function. Obtaining the entries of the original scattering matrix required some reasoning (e.g. [Succi, 2001, pp. 51-55]) so that latter manipulation increased the simplicity and efficiency of the lattice Boltzmann approach. Nevertheless, the complete set of distribution functions is now relaxed with the same constant parameter, which excludes the distinct treatment of different physical and non-physical modes. Remedy was provided by D'Humières [1992], who showed that the scattering matrix $\boldsymbol{A}$ can directly be obtained from the eigenvector basis, which laid the foundation for the development of multi-relaxation time (MRT) models that are discussed in more detail in § 4.2.2.

The idea to model the effect of collision rather than the molecular process itself, was first suggested by Bhatnagar, Gross and Krook in order to avoid solving the intricate collision integral of the continuous Boltzmann equation [Bhatnagar et al., 1954]. The rationale behind this approach is the fact that most details hidden in the collision operator do not play a role at the macroscopic level. It is therefore replaced by a much handier expression retaining only the basic features of fluid dynamics. Besides the conservation of mass, momentum and energy (MME), the BGK collision operator respects the $\mathcal{H}$-theorem, which states that the distribution functions always evolve towards equilibrium. The lattice Boltzmann models that rely on this approximation are referred to as BGK-LBM, or rather LBGK, and constitute the most basic and wildly applied class of LB schemes. In chapter 4, we will learn about a more advanced collision models that encompasses a spectrum of different relaxation time-scales.

2.2 LBM in the context of continuum mechanics

For the vast majority of aerodynamic applications, it is adequate to describe the state of a fluid by a set of macroscopic variables like pressure and velocity that arise from statistical averaging of the microscopic particle dynamics. In this case, the fluid may be considered as a continuum and the governing equations – called the Navier-Stokes equations (NSE) – rely on some global conservation laws. It is, however, well-known that the continuum hypothesis constitutes the highest level of abstraction for a fluid by completely ignoring the underlying microscopic world. A reproduction of the detailed dynamics at the microscopic level, on the other hand, is prohibitively expensive to achieve computationally. The lattice Boltzmann method resides somewhere between these extremes (Fig. 2.3). It may be considered as a more fundamental approach since it accounts for underlying particle dynamics and interactions, even if this is achieved in a very simplified manner. The reason, the LBM is computational competitive or even faster than a Navier-Stokes (NS) solver, lies in these simplifications of the microscopic dynamics. In order to show that these abstraction do not compromise its validity at the macroscopic level, it is necessary to relate these two approaches. The purpose of this section is thus to establish the link between the mesoscopic and the macroscopic

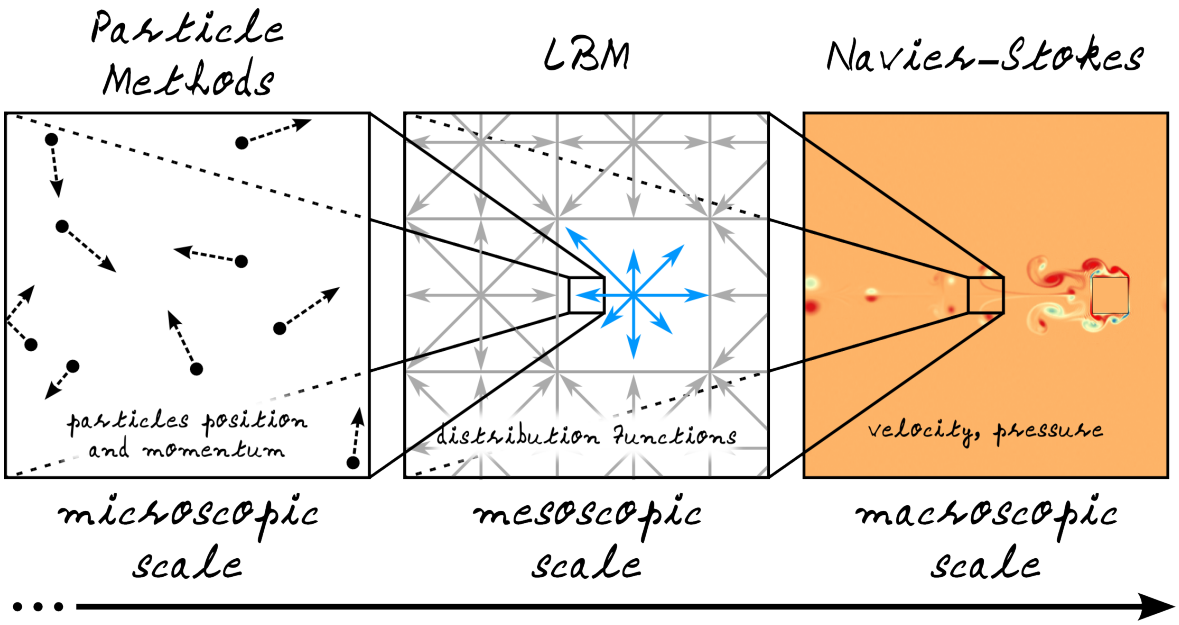


Figure 2.3: Different abstraction levels of the fluid dynamics.

world.

Evolved from lattice gas automata, the lattice Boltzmann model is based on a discrete velocity space, which – by construction – discretizes physical space and time. *A priori* the density functions are, however, continuous solutions of the Boltzmann equation, so that we begin our consideration in continuous velocity space. Subsequently, we will proceed to a discrete velocity model and point out the constraints that are to respect. In a final step, we will retrieve the Navier-Stokes equations from our discrete LBGK model (2.5) using a multi-timescale analysis.

2.2.1 From mesoscopic to macroscopic quantites

A general observation is that in a given control volume, a mesoscopic quantity comprises the totality of particles that possess a velocity in the interval $\boldsymbol{\xi} + \mathrm{d}\boldsymbol{\xi}$ and represents thus a fraction of its macroscopic equivalent. The link from the mesoscopic to the macroscopic scale is thus established by integrating the distribution functions over velocity space $\boldsymbol{\xi}$, which yields the velocity moment tensor of order n

$$\boldsymbol{M}^{(n)} = \int_{\mathbb{R}^{\mathrm{D}}} \underbrace{\boldsymbol{\xi}\boldsymbol{\xi}\ldots\boldsymbol{\xi}}_{\text{n-times}} f(\boldsymbol{x}, \boldsymbol{\xi}, t)\, \mathrm{d}\boldsymbol{\xi}.$$

Here and in the following, D denotes the dimension in space. The indication of the integration domain $\mathbb{R}^{\mathrm{D}}$ will be omitted, hereafter.

While at the macroscopic level, the conservation of mass, momentum and energy

is guaranteed by the nature of the equations themselves, in the mesoscopic world, MME conservation is governed by the collision operator and translates to the following equation

$$\int (\Omega(f) \cdot \psi_k) \, \mathrm{d}\boldsymbol{\xi} = 0, \tag{2.6}$$

where ψ_k denotes the collision invariants, i.e. those variables for which Eq. (2.6) holds. It can be shown that $2 + \mathrm{D}$ collision invariants exist, namely $\psi_0 = 1$, $\psi_{1-\mathrm{D}} = \boldsymbol{\xi}$ and $\psi_{(\mathrm{D}+1)} = |\boldsymbol{\xi}|^2$. If we now integrate the distribution functions with respect to the collision invariants, we naturally define the related macroscopic quantities in continuous velocity space

$$\begin{cases} \rho(\boldsymbol{x}, t) = \displaystyle\int f(\boldsymbol{x}, \boldsymbol{\xi}, t) \, \mathrm{d}\boldsymbol{\xi} \\ \rho \boldsymbol{u}(\boldsymbol{x}, t) = \displaystyle\int \boldsymbol{\xi} f(\boldsymbol{x}, \boldsymbol{\xi}, t) \, \mathrm{d}\boldsymbol{\xi} \\ \rho E(\boldsymbol{x}, t) = \dfrac{1}{2} \displaystyle\int |\boldsymbol{\xi}|^2 f(\boldsymbol{x}, \boldsymbol{\xi}, t) \, \mathrm{d}\boldsymbol{\xi}, \end{cases} \tag{2.7}$$

where E denotes the specific energy. For a monatomic gas, in which collisions are assumed elastic, the specific energy is composed of the internal and the kinetic energy, i.e.

$$\rho E = \rho \left(e + \frac{1}{2} |\boldsymbol{u}|^2 \right).$$

It can be shown that in this case, the conservation of energy can equally be represented by the integration with respect to the peculiar velocity $\boldsymbol{v} = \boldsymbol{\xi} - \boldsymbol{u}$, that is the particle velocity in the reference frame of the mean flow $\boldsymbol{u}$

$$\rho e(\boldsymbol{x}, t) = \frac{1}{2} \int |\boldsymbol{v}|^2 f(\boldsymbol{x}, \boldsymbol{\xi}, t) \, \mathrm{d}\boldsymbol{\xi}. \tag{2.8}$$

In continuous fluid mechanics, the conservation equation of the internal energy e is required to link the pressure to the density via an equation of state $p = p(\rho, e)$, namely

$$p = \frac{1}{\mathrm{D}} \int |\boldsymbol{v}|^2 f(\boldsymbol{x}, \boldsymbol{\xi}, t) \, \mathrm{d}\boldsymbol{\xi} = \frac{2}{\mathrm{D}} \rho e. \tag{2.9}$$

In the following, we will restrict our demonstration to an isothermal fluid with $T = \mathrm{cst}$. The motivation for this restriction will emerge during the course of this chapter. Using

the perfect gas law in the above expression, we thus obtain

$$e = \frac{\mathrm{D}}{2}\frac{p}{\rho} = \frac{\mathrm{D}}{2}RT_0 = \frac{\mathrm{D}}{2}\frac{k_B T_0}{m} = \frac{\mathrm{D}}{2}c_0^2, \tag{2.10}$$

where T_0 and c_0 denote constant temperature and speed of sound, respectively. We may therefore rewrite Eq. (2.9) under the following form

$$p = \rho c_0^2, \tag{2.11}$$

which is the isentropic equation of state with an adiabatic coefficient of $\gamma = 1$ (e.g. [Kundu et al., 2012]).

2.2.2 The thermodynamic equilibrium

If a fluid is in a state free of stress, it can be described by the equilibrium distribution function. Such a scenario only occurs when density and velocity are constant in the entire fluid domain. In this case the equilibrium is described by the Maxwellian distribution

$$f^{eq} = \frac{\rho}{(2\pi c_0^2)^{\mathrm{D}/2}} exp\left(\frac{-(\boldsymbol{\xi}-\boldsymbol{u})^2}{2c_0^2}\right), \tag{2.12}$$

where ρ and $\boldsymbol{u}$ denote the macroscopic density and velocity, respectively. We identify the quadratic term in the exponent as the peculiar velocity $\boldsymbol{v}$. The Maxwellian distribution f^{eq} can thus be interpreted as the probability to find a particle about a certain velocity $\boldsymbol{\xi}$ in a system, where all directions of the peculiar velocity $\boldsymbol{v}$ are equally probable. Likewise, f^{eq} may be seen as the density function that cancels the collision operator. It is therefore possible to derive $ln(f^{eq})$ from a linear combination of the collision invariants (for more details see [Marié, 2008]). This also implies that

$$\begin{cases} \rho(\boldsymbol{x},t) = \displaystyle\int f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} \\ \rho\boldsymbol{u}(\boldsymbol{x},t) = \displaystyle\int \boldsymbol{\xi} f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} \\ \rho E = \displaystyle\int |\boldsymbol{\xi}|^2 f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi}. \end{cases} \tag{2.13}$$

In conjunction with the definitions (2.7), it logically follows that

$$\begin{cases} \int f^{neq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} = 0 \\ \int \boldsymbol{\xi} f^{neq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} = 0 \\ \int |\boldsymbol{\xi}|^2 f^{neq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} = 0, \end{cases}$$

where $f^{neq} = f - f^{eq}$ denotes the non-equilibrium part of the distribution functions. It becomes clear that the BGK collision operator $\Omega(f) = \tilde{\omega}\,(f - f^{eq})$ respects these conservation constraints.

Apart from the conserved quantities, the multi-timescale analysis in § 2.2.6 will lead to the appearance of higher-order velocity moments that can be split into an equilibrium and a non-equilibrium part. The second-order velocity moment is the momentum flux tensor

$$P(\boldsymbol{x},t) = \underbrace{\int \boldsymbol{\xi}\boldsymbol{\xi} f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi}}_{P^{eq}(\boldsymbol{x},t)} + \underbrace{\int \boldsymbol{\xi}\boldsymbol{\xi} f_{\alpha}^{neq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi}}_{P^{neq}(\boldsymbol{x},t)},$$

If the particle velocity is expressed as the sum of the fluid velocity $\boldsymbol{u}$ and peculiar velocity $\boldsymbol{v}$, we may obtain a simple expression for P^{eq} . Since the odd velocity moments of the peculiar velocity are zero, e.g.

$$\int \boldsymbol{v} f(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} = \underbrace{\int \boldsymbol{\xi} f(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi}}_{\rho\boldsymbol{u}} - \boldsymbol{u}\underbrace{\int f(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi}}_{\rho} = 0,$$

the second moment reads

$$\langle \xi_i \xi_j \rangle = \langle (u_i + v_i)(u_j + v_j) \rangle = u_i u_j + \langle v_i v_j \rangle,$$

with $\langle \square \rangle = \int \square f(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi}$. Introducing the integration rule for a spherically symmetric integrand [Viggen, 2014]

$$\int x_i x_j f(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \frac{\delta_{ij}}{3}\int x_i^2 f(\boldsymbol{x})\mathrm{d}\boldsymbol{x},$$

in addition to Eq. (2.8), (2.10) and (2.13), finally yields

$$\begin{aligned} P^{eq}(\boldsymbol{x},t) &= u_i u_j \int f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} + \frac{\delta_{ij}}{3}|\boldsymbol{v}|^2 f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t)\,\mathrm{d}\boldsymbol{\xi} \\ &= \rho u_i u_j + \rho c_0^2 \delta_{ij}. \end{aligned} \tag{2.14}$$

Scrutinizing this result, we recognize that P^{eq} represents the inviscid momentum flux tensor. At this point, we anticipate that P^{neq} takes the role of the deviatoric stress tensor $\boldsymbol{\sigma'}$, so that $P = P^{eq} + P^{neq}$ represents the total momentum flux tensor. We can thus assume that the isothermal Navier-Stokes dynamics are contained within the first three velocity moments of f.

2.2.3 From macroscopic to mesoscopic quantities

In the previous section, we have learnt that the macroscopic variables are the velocity moments of $f(\boldsymbol{x},\boldsymbol{\xi},t)$. So, if the distribution f related to the infinitesimal small velocity space $\boldsymbol{\xi} + \mathrm{d}\boldsymbol{\xi}$ has its share in any moment up to arbitrary order, then there should exist a polynomial representation for f with respect to these moments. Indeed, the distribution function can be expressed as an infinite series of dimensionless Hermite polynomials in $\tilde{\boldsymbol{\xi}}$, which is the normalized velocity space with respect to the speed of sound, i.e. $\tilde{\boldsymbol{\xi}} = \boldsymbol{\xi}/c_0$.

$$f(\boldsymbol{x},\boldsymbol{\xi},t) = \frac{1}{c_0^{\mathrm{D}}} w(\tilde{\boldsymbol{\xi}}) \sum_{n=0}^{\infty} \frac{1}{n!} \boldsymbol{a}^{(n)}(\boldsymbol{x},t) \mathscr{H}^{(n)}(\tilde{\boldsymbol{\xi}}), \tag{2.15}$$

where $\mathscr{H}^{(n)}$ denotes a polynomial of degree n. A general definition of $\mathscr{H}^{(n)}$ and expressions for $n \leq 3$ are given in appendix C. The peculiarity about this class of polynomials is the fact that the expansion coefficients $\boldsymbol{a}^{(n)}$ are the velocity moments of f itself [Shan et al., 2006], i.e.

$$\boldsymbol{a}^{(n)} = c_0^{\mathrm{D}} \int f(\boldsymbol{x},\boldsymbol{\xi},t) \mathscr{H}^{(n)}(\tilde{\boldsymbol{\xi}})\,\mathrm{d}\tilde{\boldsymbol{\xi}}.$$

The weighing function w associated to the Hermite polynomials is given as

$$w(\tilde{\boldsymbol{\xi}}) = \frac{1}{(2\pi)^{\mathrm{D}/2}} exp(-\tilde{\boldsymbol{\xi}}^2/2). \tag{2.16}$$

In the same way, it is possible to express the equilibrium distribution function f^{eq} as a projection on a basis of Hermite polynomials, with the slight difference that only the

equilibrium part of the moments $\boldsymbol{a}^{eq,(n)}$ is taken into account

$$f^{eq}(\boldsymbol{x},\boldsymbol{\xi},t) = \frac{1}{c_0^{\mathrm{D}}} w(\tilde{\boldsymbol{\xi}}) \sum_{n=0}^{\infty} \frac{1}{n!} \boldsymbol{a}^{eq,(n)}(\boldsymbol{x},t) \mathscr{H}^{(n)}(\tilde{\boldsymbol{\xi}}). \tag{2.17}$$

Comparing the Maxwellian distribution in Eq. (2.12) with the weight function Eq. (2.16), we recognize a similar structure. In fact, the distribution function can be expressed as

$$f^{eq} = \frac{\rho}{c_0^{\mathrm{D}}} w(\tilde{\boldsymbol{v}}),$$

where $\tilde{\boldsymbol{v}} = (\boldsymbol{\xi} - \boldsymbol{u})/c_0$ can be interpreted as the peculiar Mach number. The Hermite coefficients of the Maxwellian can therefore be calculated as

$$\boldsymbol{a}^{eq,(n)} = \rho \int w(\tilde{\boldsymbol{v}}) \mathscr{H}^{(n)}(\tilde{\boldsymbol{v}} + \frac{\boldsymbol{u}}{c_0}) \, \mathrm{d}\tilde{\boldsymbol{v}}. \tag{2.18}$$

2.2.4 Discretization of the velocity space

Truncating the infinite series of Hermite polynomials in Eq. (2.15) corresponds to a discrete consideration of velocity space $\boldsymbol{\xi}$, while continuous and discrete moments remain identical up to the truncated order. In the previous section, we came to the conclusion that the variables relevant to the Navier-Stokes dynamics are contained within the first three velocity moments, so that a truncation at second-order seems reasonable. An important constraint on the truncation arises from the approximation of the moment integral in Eq. (2.18) using Gaussian quadrature

$$\int w(\boldsymbol{x}) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \approx \sum w_\alpha f(\boldsymbol{x}_\alpha) \quad \text{with} \quad \alpha = 0, 1, ..., q-1,$$

where $f(\boldsymbol{x})$ denotes any function and w_α are a set of constant weights. The discrete velocities $\tilde{\boldsymbol{\xi}}_\alpha$ represent thus the abscissae of the truncated Hermite series of degree N. Higher-order truncations lead to an increased number of discrete velocity that require intricate lattice topologies. The quadrature formula E is defined by three numbers

$$E^q_{\mathrm{D},n},$$

where D denotes the dimension in space, n is the algebraic degree of precision and q represents the number of points employed by the quadrature. Moreover $n > 2N$ and also $n = 2q - 1$, in one dimension [Shan et al., 2006]. It would now be interesting to relate the order of the highest moment that determines the hydrodynamic equations to the number of abscissa (i.e. discrete velocities q) that are required. Assuming that

this order is N, we obtain

$$q = N + 1$$

in 1D with $n = 2N + 1$. In higher dimensions no general Gauss quadrature theory is known. One may, of course, say

$$q = (N + 1)^{\mathrm{D}}. \tag{2.19}$$

Considering a dicretization of velocity space in two dimensions, a truncation at $N = 2$ thus requires 9 discrete velocities, whereas a truncation at $N = 3$ already demands 16 abscissae. The actual number of lattice vectors may, nevertheless, be slightly different. Due to the symmetry of certain moments in two and three dimensions (rank 2 tensor and higher), the number of abscissae can be reduced while maintaining the same order N. The geometrical requirement to design a space-filling, isotropic lattice may, on the other hand, increase the number of lattice vectors above the value obtained with Eq. (2.19). Adopting the naming convention DdQq introduced by Qian et al. [1992], where d stand for the dimension in space and q denotes the number of lattice vectors, the only one-dimensional lattice that allows to maintain the velocity moments up to second-order is the D1Q3 lattice. In two dimensions, according to relation (2.19), this would be the D2Q9 lattice. It can, however, be shown that a D2Q7 topology also produces the exact velocity moments up to second-order. A truncation in 2D at $N = 3$ requires 16 velocity moments, which does not produce a very convenient lattice, due to the zero velocity vector. In this case, the number of lattice vectors is increased by one, i.e. D2Q17.

Using Eq. (2.18), the following Hermite coefficients are obtained for a second-order expansion of f^{eq}

$$\begin{cases} \boldsymbol{a}^{eq,0} = \rho \\ \boldsymbol{a}^{eq,1} = \rho \dfrac{\boldsymbol{u}}{c_0} \\ \boldsymbol{a}^{eq,2} = \rho \dfrac{u_i u_j}{c_0^2}. \end{cases}$$

Substituting the above expressions in the polynomial representation of f^{eq} (Eq.(2.17)) truncated at second order, yields

$$f_\alpha^{eq}(\boldsymbol{x}, t) = w_\alpha \rho \left(1 + \frac{\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u}}{c_0^2} + \frac{(\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u})^2}{2c_0^4} - \frac{u^2}{2c_0^2} \right), \tag{2.20}$$

with $f_\alpha^{eq}(\boldsymbol{x}, t) = f^{eq}(\boldsymbol{x}, \boldsymbol{\xi}_\alpha, t)$ and $w_\alpha = w(\frac{\xi_\alpha}{c_0})/c_0^{\mathrm{D}}$. While the above derivation presents the rigorous and systematic way to obtain the distribution functions and discretize

velocity space, it certainly constitutes one of the less easily digestible sections of this manuscript. The theory was restricted to a minimum and probably requires some prior knowledge in this field. The reader may refer to Shan et al. [2006], Malaspinas [2009] for a more thorough consideration of this subject.

An alternative representation constitutes the development of f^{eq} (Eq. (2.12)) in a Taylor series. This approach, however, is only valid for a truncation up to second-order. A truncation at higher-order diverges from the preceding Hermite expansion. Using the weight function (Eq. (2.16)), we can express Eq. (2.12) as

$$f^{eq}\left(\boldsymbol{x},\boldsymbol{\xi},t\right)=\frac{\rho}{c_0^{\mathrm{D}}}w(\frac{\boldsymbol{\xi}}{c_0})exp\left(\frac{2\boldsymbol{\xi}\cdot\boldsymbol{u}-u^2}{2c_0^2}\right).$$

Applying the expansion $e^x=\sum_{n=0}^{\infty}\frac{x^n}{n!}$ to the exponential with truncation at $\mathcal{O}(u^2)$, one obtains

$$f_\alpha^{eq}\left(\boldsymbol{x},t\right)=\rho w_\alpha\left(1+\frac{\boldsymbol{\xi}_\alpha\cdot\boldsymbol{u}}{c_0^2}+\frac{(\boldsymbol{\xi}_\alpha\cdot\boldsymbol{u})^2}{2c_0^4}-\frac{u^2}{2c_0^2}\right), \tag{2.21}$$

where the speed of sound c_0^{D} is absorbed in the discrete weighting function w_α according to the previous definition: $w_\alpha=w(\frac{\boldsymbol{\xi}}{c_0})/c_0^{\mathrm{D}}$.

For a second-order truncation, which corresponds to the most classical lattices, i.e. D2Q9 in two dimensions and D3Q19 or rather D3Q27 in three dimension, both approaches are equally valid (cf. Eq. (2.20) and Eq. (2.21)). For a higher-order truncations the projection on Hermite polynomials is the only option.

2.2.5 Discrete moments and constraints

The discretization of velocity space results in a discrete set of distribution functions f_α in each node of the lattice that propagate according to an assigned velocities $\boldsymbol{\xi}_\alpha$. The integration of f in continuous moment space thus reduces to a weighted summation of the distribution function defined on the discrete velocity points $\boldsymbol{\xi}_\alpha$. The macroscopic moments are then expressed as

$$\boldsymbol{M}^{(n)}=\sum_{\alpha=0}^{q-1}\underbrace{\boldsymbol{\xi}_\alpha\boldsymbol{\xi}_\alpha\ldots\boldsymbol{\xi}_\alpha}_{\text{n-times}}f_\alpha(\boldsymbol{x},t),$$

where $f_\alpha\left(\boldsymbol{x},t\right)=f\left(\boldsymbol{x},\boldsymbol{\xi}_\alpha,t\right)$. It was previously stated that the discrete moments are exact up to the order at which the Hermite polynomial are truncated. Using a second-order truncation, following relations thus have to hold that allow us to derive

expressions for the lattice weights w_α and the speed of sound c_0

$$\rho(\boldsymbol{x},t) = \sum_\alpha f_\alpha^{eq}(\boldsymbol{x},t) \tag{2.22}$$

$$\rho\boldsymbol{u}(\boldsymbol{x},t) = \sum_\alpha \boldsymbol{\xi}_\alpha f_\alpha^{eq}(\boldsymbol{x},t), \tag{2.23}$$

$$P^{eq}(\boldsymbol{x},t) = \rho u_i u_j + \rho c_0^2 \delta_{ij} = \sum_\alpha \boldsymbol{\xi}_\alpha \boldsymbol{\xi}_\alpha f_\alpha^{eq}(\boldsymbol{x},t). \tag{2.24}$$

Substituting for f^{eq} defined by Eq. (2.21), we obtain following set of constraints

$$\text{from (2.22):} \begin{cases} \sum_\alpha w_\alpha = 1, \\ \sum_\alpha w_\alpha \boldsymbol{\xi}_\alpha = 0, \\ \sum_\alpha w_\alpha \xi_{\alpha,i} \xi_{\alpha,j} = c_0^2 \delta_{ij}. \end{cases}$$

$$\text{from (2.23):} \quad \sum_\alpha w_\alpha \xi_{\alpha i} \xi_{\alpha j} \xi_{\alpha k} = 0.$$

$$\text{from (2.24):} \quad \sum_\alpha w_\alpha \xi_{\alpha i} \xi_{\alpha j} \xi_{\alpha k} \xi_{\alpha l} = c_0^4 \left(\delta_{ij}\delta_{kl} + \delta_{ij}\delta_{kl} + \delta_{ij}\delta_{kl}\right).$$

Together with the requirement that a lattice should be isotropic, we will demonstrate in § 2.3, how these constraints will yield the weights and the speed of sound for the classical D2Q9 lattice.

2.2.6 From LBM to Navier-Stokes

Having introduced the velocity moments as the intermediary between the mesoscopic and macroscopic world, we are now going to retrieve the Navier-Stokes equation from the lattice Boltzmann equation via a multi-timescale analysis. Many aspects of fluid dynamic behaviour at the macroscopic level manifest in the deviation of the distribution function from its state of equilibrium f^{eq}. If in an underlying mesoscopic system the distribution functions are, without exception, at equilibrium, one can easily imagine that there is not much going on at the macroscopic level. The idea of the approach is thus to expand the distribution function f around the equilibrium f^{eq}, then denoted $f^{(0)}$ and characterize the non-equilibrium terms according to different timescales

$$f = \underbrace{f^{(0)}}_{f^{eq}} + \underbrace{\epsilon f^{(1)} + \epsilon^2 f^{(2)}}_{f^{neq}} \ldots. \tag{2.25}$$

The multi-timescales expansion relies on a small parameter ϵ, which accounts for the ratio between the collision time scale $\tau = \nu/c_0^2$ and the lattice time scale Δt. The advection timescale is directly related to $\Delta t_1 = \tau/\epsilon$, while the viscous diffusion time scale is given by

$$\Delta t_2 = \frac{\Delta x^2}{\nu} \sim \frac{c_0^2 \Delta t^2}{\nu} = \frac{\tau}{\epsilon^2}.$$

A development up to second-order in ϵ thus seems reasonable in order to retrieve the Navier-Stokes equations. In fact, it can be shown that a first-order development only yields the Euler equations [Huang, 1987]. The time derivative is developed such that

$$\frac{\partial}{\partial t} = \epsilon \frac{\partial}{\partial t_1} + \epsilon^2 \frac{\partial}{\partial t_2}, \tag{2.26}$$

whereas the space derivative only consists of the first-order term

$$\frac{\partial}{\partial \boldsymbol{x}} = \epsilon \frac{\partial}{\partial \boldsymbol{x}_1}. \tag{2.27}$$

With respect to the decomposition of the non-equilibrium part of the distribution function, the conservation constraints (2.6) translate to the following solvability conditions

$$\sum_\alpha f_\alpha^{(n)} = \sum_\alpha \boldsymbol{\xi}_\alpha f_\alpha^{(n)} = 0, \quad \text{for } n = 1, 2, ... \tag{2.28}$$

We recall that the interest of this section is to manipulate the lattice Boltzmann equation in such a way that it yields the Navier-Stokes equations. As a first step, we expand the LHS term of Eq. (2.5) in a Taylor series up to second-order

$$\begin{aligned} f_\alpha(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t) \approx \\ f_\alpha(\boldsymbol{x}, t) + \left(\xi_{\alpha i} \Delta t \frac{\partial}{\partial x_i} + \Delta t \frac{\partial}{\partial t} \right) f_\alpha(\boldsymbol{x}, t) + \\ \frac{1}{2} \left(\xi_{\alpha i} \xi_{\alpha j} \Delta t^2 \frac{\partial^2}{\partial x_i \partial x_j} + 2 \xi_{\alpha i} \Delta t^2 \frac{\partial}{\partial x_i} \frac{\partial}{\partial t} + \Delta t^2 \frac{\partial^2}{\partial t^2} \right) f_\alpha(\boldsymbol{x}, t) + \mathcal{O}(\Delta t^3). \end{aligned}$$

If we substitute in Eq. (2.5) for the above expression and replace the non-dimensional relaxation frequency $\tilde{\omega}$ by the dimensional relaxation time τ, we obtain

$$\Delta t \left(\xi_{\alpha i} \frac{\partial}{\partial x_i} + \frac{\partial}{\partial t} \right) f_\alpha + \frac{\Delta t^2}{2} \left(\xi_{\alpha i} \xi_{\alpha j} \frac{\partial^2}{\partial x_i \partial x_j} + 2 \xi_{\alpha i} \frac{\partial}{\partial x_i} \frac{\partial}{\partial t} + \frac{\partial^2}{\partial t^2} \right) f_\alpha = -\frac{\Delta t}{\tau} \left(f_\alpha - f_\alpha^{eq} \right).$$

Following, we substitute for the respective multi-timescale expressions and divide by Δt

$$\left[\xi_{\alpha i}\left(\epsilon\frac{\partial}{\partial x_{1i}}\right)+\left(\epsilon\frac{\partial}{\partial t_1}+\epsilon^2\frac{\partial}{\partial t_2}\right)\right]\left(f_\alpha^{(0)}+\epsilon f_\alpha^{(1)}+\epsilon^2 f_\alpha^{(2)}\right)+$$
$$\frac{\Delta t}{2}\left[\xi_{\alpha i}\xi_{\alpha j}\left(\epsilon^2\frac{\partial^2}{\partial x_{1i}\partial x_{1j}}\right)+\left(2\xi_{\alpha i}\epsilon^2\frac{\partial}{\partial x_{1i}}\frac{\partial}{\partial t_1}\right)+\left(\epsilon^2\frac{\partial^2}{\partial t_1^2}\right)\right]\left(f_\alpha^{(0)}+\epsilon f_\alpha^{(1)}+\epsilon^2 f_\alpha^{(2)}\right)=$$
$$-\frac{1}{\tau}\left(f_\alpha^{(0)}+\epsilon f_\alpha^{(1)}+\epsilon^2 f_\alpha^{(2)}-f_\alpha^{eq}\right).$$

Now, the terms are regrouped according to their ascending order in ϵ

$\mathcal{O}(\epsilon^0)$**:**

$$f_\alpha^{(0)}=f_\alpha^{eq}$$

$\mathcal{O}(\epsilon^1)$**:**

$$\epsilon\left(\xi_{\alpha i}\frac{\partial}{\partial x_{1i}}+\frac{\partial}{\partial t_1}\right)f_\alpha^{(0)}=-\frac{1}{\tau}\epsilon f_\alpha^{(1)} \tag{2.29}$$

$\mathcal{O}(\epsilon^2)$**:**

$$\epsilon^2\frac{\partial}{\partial t_2}f_\alpha^{(0)}+\left(\xi_{\alpha i}\epsilon\frac{\partial}{\partial x_{1i}}+\epsilon\frac{\partial}{\partial t_1}\right)\epsilon f_\alpha^{(1)}+\frac{\Delta t}{2}\left(\xi_{\alpha i}\epsilon\frac{\partial}{\partial x_{1i}}+\epsilon\frac{\partial}{\partial t_1}\right)^2 f_\alpha^{(0)}=-\frac{1}{\tau}\epsilon^2 f_\alpha^{(2)}$$

Applying Eq.(2.29) to the above equation we get

$$\epsilon^2\frac{\partial}{\partial t_2}f_\alpha^{(0)}+\left(\xi_{\alpha i}\epsilon\frac{\partial}{\partial x_{1i}}+\epsilon\frac{\partial}{\partial t_1}\right)\left(1-\frac{\Delta t}{2\tau}\right)\epsilon f_\alpha^{(1)}=-\frac{1}{\tau}\epsilon^2 f_\alpha^{(2)}. \tag{2.30}$$

The procedure to obtain the macroscopic conservation equations now requires the integration of Eq.(2.29) and Eq. (2.30) with respect to the collision invariants. The moment equations at $\mathcal{O}(\epsilon)$ are then

$$\epsilon\frac{\partial\rho}{\partial t_1}+\epsilon\frac{\partial\rho u_i}{\partial x_{1i}}=0, \tag{2.31a}$$
$$\epsilon\frac{\partial\rho u_i}{\partial t_1}+\epsilon\frac{\partial P_{ij}^{(0)}}{\partial x_{1j}}=0, \tag{2.31b}$$

which are the inviscid Euler equations. In the same way, we obtain the respective equations at $\mathcal{O}(\epsilon^2)$

$$\epsilon^2 \frac{\partial \rho}{\partial t_2} = 0$$
$$\epsilon^2 \frac{\partial \rho u_i}{\partial t_2} + \left(1 - \frac{\Delta t}{2\tau}\right) \epsilon^2 \frac{P_{ij}^{(1)}}{\partial x_{1j}} = 0.$$

As a final step of the analysis, the different scales of the time and space derivative are assembled for each moment (cf. Eq. (2.26) and Eq. (2.27)), such that the zeroth-moment equation writes

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} = 0.$$

The above equation can easily be identified as the continuity equation. Gathering the time and space scales of the first moment equation, one obtains

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j} \left[P_{ij}^{eq} + \left(1 - \frac{\Delta t}{2\tau}\right) P_{ij}^{neq} \right] = 0, \tag{2.33}$$

with $P^{neq} = \epsilon P^{(1)}$. As it was stated previously, the non-equilibrium part of the non-conserved moments is *a priori* unknown. In order to calculate $P_{ij}^{(1)}$, Eq.(2.29) is transformed into moment space with respect to $\boldsymbol{\xi}_\alpha \boldsymbol{\xi}_\alpha$

$$\epsilon \frac{\partial P_{ij}^{(0)}}{\partial t_1} + \epsilon \frac{\partial Q_{ijk}^{(0)}}{\partial x_{1k}} = -\frac{1}{\tau} \epsilon P_{ij}^{(1)}. \tag{2.34}$$

We observe the appearance of a third-order moment Q_{ijk}, which is obtained from the truncated equilibrium distribution function (2.21)

$$Q_{ijk}^{(0)} = Q_{ijk}^{eq} = \sum_\alpha f_\alpha^{eq} \xi_{\alpha,i} \xi_{\alpha,j} \xi_{\alpha,k} = \rho c_0^2 \left(u_i \delta_{jk} + u_j \delta_{ik} + u_k \delta_{ij} \right).$$

As a next step, we introduce following derivatives based on the product rule

$$\epsilon \frac{\partial \rho u_i u_j}{\partial t_1} = u_i \epsilon \frac{\rho u_j}{\partial t_1} + u_j \epsilon \frac{\rho u_i}{\partial t_1} - u_i u_j \epsilon \frac{\partial \rho}{\partial t_1},$$
$$\epsilon \frac{\partial \rho u_i u_j u_k}{\partial x_{1k}} = u_i \epsilon \frac{\rho u_j u_k}{\partial x_{1k}} + u_j \epsilon \frac{\rho u_i u_k}{\partial x_{1k}} - u_i u_j \epsilon \frac{\partial \rho u_k}{\partial x_{1k}}.$$

Using the above corollaries, the first term on the LHS of Eq. (2.34) can be expressed as

$$\begin{aligned}
\epsilon\frac{\partial P_{ij}^{(0)}}{\partial t_1} &= \epsilon\frac{\partial \rho u_i u_j}{\partial t_1} + c_0^2\delta_{ij}\epsilon\frac{\partial \rho}{\partial t_1} \\
&= u_i\epsilon\frac{\partial \rho u_j}{\partial t_1} + u_j\epsilon\frac{\partial \rho u_i}{\partial t_1} - u_i u_j\epsilon\frac{\partial \rho}{\partial t_1} + c_0^2\delta_{ij}\epsilon\frac{\partial \rho}{\partial t_1} \\
&= -u_i\epsilon\frac{\partial P_{jk}^{(0)}}{\partial x_{1k}} - u_j\epsilon\frac{\partial P_{ij}^{(0)}}{\partial x_{1k}} + u_i u_j\epsilon\frac{\partial \rho u_k}{\partial x_{1k}} - c_0^2\delta_{ij}\epsilon\frac{\partial \rho u_k}{\partial x_{1k}} \quad \text{using Eqs. (2.31)} \\
&= -u_i\epsilon\frac{\partial}{\partial x_{1k}}\left(\rho u_j u_k + \rho c_0^2\delta_{jk}\right) - u_j\epsilon\frac{\partial}{\partial x_{1k}}\left(\rho u_i u_k + \rho c_0^2\delta_{ik}\right) \\
&\quad + u_i u_j\epsilon\frac{\partial \rho u_k}{\partial x_{1k}} - c_0^2\delta_{ij}\frac{\partial \rho u_k}{\partial x_{1k}} \\
&= -\epsilon\frac{\partial \rho u_i u_j u_k}{\partial x_{1k}} - c_0^2\left(u_i\epsilon\frac{\partial \rho}{\partial x_{1j}} + u_j\epsilon\frac{\partial \rho}{\partial x_{1i}}\right) - c_0^2\delta_{ij}\epsilon\frac{\partial \rho u_k}{\partial x_{1k}}.
\end{aligned} \tag{2.35}$$

The spatial derivative of the third moment can be rearranged to

$$\begin{aligned}
\epsilon\frac{\partial Q_{ijk}^{(0)}}{\partial x_{1k}} &= \epsilon\frac{\partial}{\partial x_{1k}}\rho c_0^2\left(u_i\delta_{jk} + u_j\delta_{ik} + u_k\delta_{ij}\right) \\
&= c_0^2\left(\epsilon\frac{\partial \rho u_i}{\partial x_{1j}} + \epsilon\frac{\partial \rho u_j}{\partial x_{1i}}\right) + c_0^2\delta_{ij}\epsilon\frac{\partial \rho u_k}{\partial x_{1k}},
\end{aligned}$$

so that the summation of the two derivatives yields the expression for $P_{ij}^{(1)}$

$$\epsilon P_{ij}^{(1)} = -\rho c_0^2\tau\left(\epsilon\frac{\partial u_i}{\partial x_{1j}} + \epsilon\frac{\partial u_j}{\partial x_{1i}}\right) + \tau\epsilon\frac{\partial \rho u_i u_j u_k}{\partial x_{1k}}.$$

Substituting for $P^{neq} = \epsilon P^{(1)}$ and $\partial/\partial\boldsymbol{x} = \epsilon\partial/\partial\boldsymbol{x}_1$ one finally obtains

$$P_{ij}^{neq} = \underbrace{-\rho c_0^2\tau\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)}_{\boldsymbol{\sigma'}} + \underbrace{\tau\frac{\partial \rho u_i u_j u_k}{\partial x_k}}_{\text{error}}.$$

According to fluid mechanics, the general form of the deviatoric stress tensor $\boldsymbol{\sigma'}$ in Newtonian fluids is

$$\sigma'_{ij} = \mu\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\delta_{ij}\boldsymbol{\nabla}\boldsymbol{u}\right) + \mu'\delta_{ij}\boldsymbol{\nabla}\boldsymbol{u} \tag{2.36}$$

[Dellar, 2001], where μ denotes the shear viscosity and μ' stands for the bulk viscosity. We can thus identify the first term on the RHS as the deviatoric stress $\boldsymbol{\sigma'}$ under

the assumption that $\mu' = \frac{2}{3}\mu$. Given that the equilibrium part of P_{ij} contains the momentum flux $\rho \boldsymbol{u}\boldsymbol{u}$ and the isothermal pressure $\rho c_0^2 \boldsymbol{I}$ (see Eq. (2.14)), the second term of the RHS can be identified as error term. It arises from the fact that the third-order moment Q_{ijk}^{eq} was calculated with the discrete Maxwellian, which was truncated under the assumption that for recovering the Navier-Stokes equations, only the velocity moments up the second-order are required (cf. § 2.2.4). Higher-order moments obtained from the weighted summation of the discrete equilibrium function therefore deviate from their continuous counterpart. In this particular case, it can be shown that the integration over continuous velocity space yields

$$Q_{ijk}^{eq} = \int_{\boldsymbol{\xi}=-\infty}^{\infty} \xi_i \xi_j \xi_k f^{eq}(\boldsymbol{x}, \boldsymbol{\xi}, t)\, \mathrm{d}\boldsymbol{\xi} = \partial \rho u_i u_j u_k + \rho c_0^2 \left(u_i \delta_{jk} + u_j \delta_{ik} + u_k \delta_{ij}\right).$$

The discrete moment thus lacks a $\mathcal{O}(u^3)$ term, which would cancel out its negative homologue in the expression for $\epsilon \dfrac{\partial P_{ij}^{(0)}}{\partial t_1}$ (cf. Eq.(2.35)). The error term is negligible compared to the first term of P^{neq} if $u^2 << c_0^2$. It is thus discarded under the low Mach number assumption. Substituting for P_{ij}^{eq} and P_{ij}^{neq} in Eq (2.33), we finally obtain the macroscopic equation for the conservation of momentum as

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \mu \frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{2.37}$$

with $\mu = \left(1 - \dfrac{\Delta t}{2\tau}\right) \rho c_0^2 \tau$. The relation between the kinetic viscosity $\nu = \mu/\rho$ and the relaxation time τ is then given as

$$\nu = c_0^2 \left(\tau - \frac{\Delta t}{2}\right) \quad \text{or rather} \quad \tau = \frac{\nu}{c_0^2} + \frac{\Delta t}{2}. \tag{2.38}$$

Given that the starting point for the above discretization was the Taylor expansion of the lattice Boltzmann equation (Eq. (2.5)) up to second-order, we can confidently say, the this model recovers the Navier Stokes equation with a second-order accuracy in space and time. A more detailed consideration of this matter will follow in chapter 3. Moreover, the preceding analysis enabled us to shed some light on one of the most severe constraint of classical LBM, which is the limitation to the low Mach number regime, usually < 0.4 M. While this limitation can theoretically be repealed by increasing the number of discrete velocities, it is generally traded for a simple lattice topology, which preserves the local character of the algorithm and facilitates the parallelisation of the code.

2.3 Numerical implementation: How to construct a LBM solver

We would like to close the present chapter with the development of a simple LBM solver written in C that is based on the preceding theory. We are going to limit this demonstration to two-dimensional space. As it was already emphasised in § 2.1, by nature of the streaming process, the lattice dictates the spatial mesh and time step. While hexagonal lattices have the advantage of a unique velocity magnitude, the most commonly chosen lattice topology in two dimensions is nowadays the D2Q9 lattice. It produces a regular Cartesian grid as seen in Fig. 2.4.

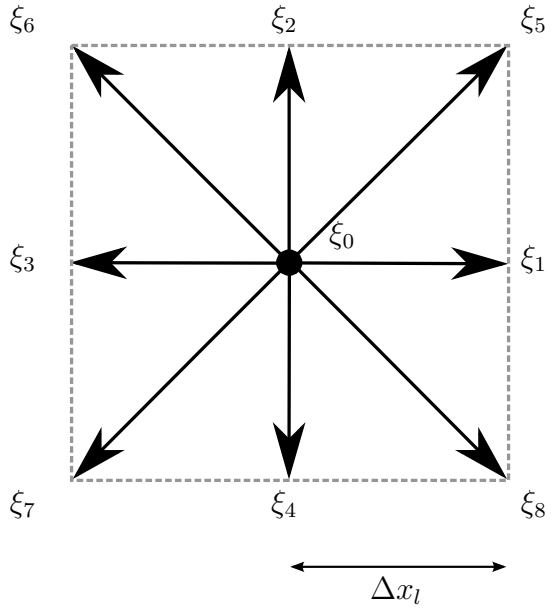


Figure 2.4: D2Q9 lattice topology. The grey line defines four spatial grid cells, whose intererior borders are the vertical and horizontal velocity vectors.

The respective x- and y-components of $\boldsymbol{\xi}_\alpha$ that correspond to the lattice in Fig. 2.4 are therefore

$$\begin{aligned}
\alpha &= \{0, \quad 1, \quad 2, \quad 3, \quad 4, \quad 5, \quad 6, \quad 7, \quad 8\} \\
\xi_{\alpha x} &= \{0, \quad c, \quad 0, \; -c, \quad 0, \quad c, \; -c, \; -c, \quad c\} \\
\xi_{\alpha y} &= \{0, \quad 0, \quad c, \quad 0, \; -c, \quad c, \quad c, \; -c, \; -c\},
\end{aligned}$$

where $c = \Delta x/\Delta t$ is the characteristic particle speed, or rather lattice speed. We can thus distinguish three classes of velocity vectors. The zero vector $\boldsymbol{\xi}_0$, the vectors $\boldsymbol{\xi}_{1-4}$ of length c that point to the nearest neighbours and the group $\boldsymbol{\xi}_{5-8}$ of magnitude $\sqrt{2}c$ that connects with the second nearest neighbours. Accordingly, we may separate the weights w_α into three groups, denoting w_0 the weight associated to $\boldsymbol{\xi}_0$, w_+ corresponding to the second class of vectors and $w_\times$ representing the weights linked to the diagonal vectors.

The lattice constraints defined in § 2.2.5 allow us now to attribute values to the weights and the speed of sound. Notably, the constraints yield following set of equations for the D2Q9 lattice

$$\begin{cases} \sum w_\alpha = & w_0 + 4w_+ + 4w_\times = 1 & \\ \sum w_\alpha \xi_{\alpha,i}\xi_{\alpha,j} = & c^2\,(2w_+ + 4w_\times) = c_0^2 & \text{for } i = j \\ \sum w_\alpha \xi_{\alpha,i}\xi_{\alpha,j}\xi_{\alpha,k}\xi_{\alpha,l} = & c^4 4w_\times = c_0^4 & \text{for } i = j \neq k = l \\ \sum w_\alpha \xi_{\alpha,i}\xi_{\alpha,j}\xi_{\alpha,k}\xi_{\alpha,l} = & c^4\,(2w_+ + 4w_\times) = 3c_0^4 & \text{for } i = j = k = l. \end{cases}$$

The above system is solved by

$$c_0 = \frac{c}{\sqrt{3}}, \qquad w_0 = \frac{4}{9}, \qquad w_+ = \frac{1}{9}, \qquad w_\times = \frac{1}{36}.$$

We have now established a relation between the lattice speed c and the speed of sound c_0. It might seem somewhat counter-intuitive that the latter is smaller than the lattice speed. The physical explanation is simple. The speed of sound describes the macroscopic velocity, by which the underlying microscopic system propagates a pressure signal through particle dynamics. In contrast, the lattice speed represents the microscopic free mean speed of particles between two collisions. Given that the collision process has a decelerating impact, it becomes obvious that a pressure signal can not be transported faster than the mean speed of the particles. Of course certain particles move slower and others move faster, which is described by the Maxwell speed distribution. Logically, a probability exists to encounter particle of speed $\sqrt{2}c$. This probability, however, is small, which is accounted for by the respective weight coefficients $w_\times$.

The physical time step Δt is proportional to the mesh size Δx and amounts to

$$\Delta t = \frac{\Delta x}{\sqrt{3}c_0}$$

A technique to accelerate the convergence in aerodynamic simulations that neglect acoustics is thus to chose a small value for the speed of sound c_0.

The speed of sound in air at room temperature is about 343 m/s. The characteristic particle speed c in the D2Q9 model then amounts to $\sqrt{3}c_0 = 594$ m/s [1]. From a computational point of view, it is more convenient to introduce the lattice units $\Delta x_l = \Delta t_l = 1$, so that c equals one, as well. The lattice speed of sound c_s then amounts to $1/\sqrt{3}$. For a given physical speed of sound c_0, velocities in lattice units are therefore obtained by multiplication with c_s/c_0.

[1] In comparison, the root-mean-square velocity of nitrogen molecules, the main element of air, is 515 m/s.

2.3.1 Initialization

The easiest way to initialize a simulation is under the hypothesis that the fluid is in a state of equilibrium. This way, the distribution functions are the Maxwell distributions for the discrete velocities $\boldsymbol{\xi}_\alpha$ and can be calculated as a function of the initial macroscopic velocity field $\boldsymbol{u}$ and pressure field p or rather the density ρ. Also, the speed of sound c_0 and the viscosity ν are given. The macroscopic values require pretreatment before fed into the truncated Maxwell distribution. In particular, the macroscopic velocity in lattice units is given as

$$\boldsymbol{u}_l = \boldsymbol{u}\frac{c_s}{c_0} = \frac{\boldsymbol{u}}{\sqrt{3}c_0}$$

The density does not require rescaling in lattice units, however for a fluid with an adiabatic coefficient deviating from $\gamma = 1$, the density value is calculated as

$$\rho_{\text{solver}} = \frac{p_0\gamma}{c_0^2}$$

where p_0 denotes the initial pressure field.

2.3.2 Stream and Collide Algorithm

The initialization with the f_α^{eq} corresponds to a post-collision state, in which the populations have been relaxed in such a manner to attain the Maxwell distribution. Consequently, the functions are ready to propagate to a neighbour site according to their discrete velocities. The simplicity of implementation shall be illustrated by the following example code, where N_x and N_y denote the number of nodes in x- and y-direction respectively. q represents the number of discrete lattice vectors and is nine in the present case. The distribution functions of two subsequent time steps are stored in a single array and IDX is the index that runs through it.

```
1 #include "header.h" // contains variables that are not initialized below,
     e.g. Nx, Ny, i, j, l, as well as global variables, e.g. offset, etc.
2 #define IDX(i, j, l) (q * (j + Ny * i) + l); // index
3
4 int main(){
5 double *f = (double *) malloc (2 * Nx * Ny * q * sizeof(double)); //
     distribution function
6 double *feq = (double *) malloc (2 * Nx * Ny * q * sizeof(double)); //
     equilibrium distribution function
7
8 int offset = Nx * Ny * q;
9                                  // Nx -> number of nodes in x-direction
```

```
                                    // Ny -> number of nodes in y-direction
                                    // q  -> number of populations per node
void streaming(double *f){

   for(int i=1; i<=Nx; i++)
      for(int j=1; j<=Ny; j++)
         for(int l=0; l<q; l++){
            int idx0 = IDX(i, j, l); // post-collision index
            int idx1 = offset + IDX(i+ex[alpha],j+ey[alpha],l); //
                pre-collision index
            f[idx1] = f[idx0];  // streaming as index shift in memory; very
                efficient if it fits in cache memory
      }
   }
```

Leaving the initialization step aside, we realize that the streaming step itself is a single line of code and simply corresponds to an index shift in memory. After the streaming operation, the set of equilibrium distribution functions is recalculated in every lattice site and the populations f are relaxed according to relation (2.38). τ is normalized with the physical time step Δt

$$\tilde{\tau} = \frac{\tau}{\Delta t} = 0.5 + \frac{\nu}{c_0^2 \Delta t} = 0.5 + \frac{\nu\sqrt{3}\Delta x}{c_0},$$

so that a possible collision routine could be coded into the main functions as followed.

```

double nu = 1.5e-05;
double c0 = 343.2;

void collision(double *f, double *feq){

   double tau = 0.5 + nu*sqrt(3)*dx/c0;
   double omega = 1./tau;

   for(int i=1; i<=Nx; i++)
      for(int j=1; j<=Ny; j++)
         for(int l=0; l<q; l++){
            int idx = offset + IDX(i, j, l);

            f[idx] = f[idx] - omega * (f[idx] - feq[idx]); // BGK collision
      }
   }
```

```
18    return 0;
19 }
```

Here f^{eq} has been calculated in a different routine that sums over the populations f to obtain ρ and $\boldsymbol{u}$ and calculates f^{eq} according to (2.21). A respective function in C could look like the following.

```
1
2  void Maxwellian(double *f, double *feq){
3
4  double feq_;
5  double Ux, Uy;
6  double Rho = 0.0, RhoUx = 0.0, RhoUy = 0.0;
7
8
9     for(int i=1; i<=Nx; i++)
10       for(int j=1; j<=Ny; j++){
11          for(int l=0; l<q; l++){
12             int idx = offset + IDX(i, j, l);
13
14             Rho += f[idx]
15             RhoUx +=f[idx]*ex; // ex=[0,1,0,-1,0,1,-1,-1,1]
16             RhoUy +=f[idx]*ey; // ey=[0,0,1,0,-1,1,1,-1,-1]
17          }
18          Ux = RhoUx/Rho;
19          Uy = RhoUy/Rho;
20
21          for(l=0; l<q; l++){
22             idx = offset + IDX(i,j,l);
23
24                feq_ = w[l]*Rho*(1.0 -
25                        1.5*(Ux*Ux+Uy*Uy) +
26                        3.0*(ex[l]*Ux+ey[l]*Uy) +
27                        4.5*(ex[l]*Ux+ey[l]*Uy)*(ex[l]*Ux+ey[l]*Uy)); //w[l]
                              are the weights
28
29                feq[idx] = feq_;
30          }
31       }
32    }
```

This terminates the most simple iteration cycle of the *stream and collide* algorithm. The main intention here was to illustrate the strength of LBM compared to conventional

CFD methods. Based on the preceding multi-timescale analysis, the reader should bear in mind that these few lines of code are a numerical description of weakly-compressible isothermal Navier-Stokes dynamics. This offers an enormous simplification compared to the hassle of treating a non-linear advection term in conventional continuum-based methods.

2.3.3 Boundary Conditions

Boundary conditions are usually defined in the context of the macroscopic variables. A no-slip condition for example is imposed by a zero velocity at the wall, i.e.

$$\boldsymbol{u} = 0.$$

A slip condition, on the other hand, implies the absence of a wall-normal flow, which translates to following expression

$$\boldsymbol{n} \cdot \boldsymbol{u} = 0,$$

where $\boldsymbol{n}$ denotes the wall normal vector. Other boundary condition types, such as in- and outflow, may be obtained with the same reasoning. In order to adapt the corresponding boundary conditions to the discrete mesoscopic variables, we first consider a lattice boundary node on a straight wall (Fig. 2.5). The populations that originate from nodes lying outside the fluid domain (grey area) are unknown. In the example shown in Fig. 2.5, these are the density functions f_2, f_5 and f_6. Suppose, a no-slip condition is desired. In this case, the pressure, or rather the density, can not be fixed unless an exact solution is available. The unknown populations need to be calculated such that $u_x = 0$ and $u_y = 0$, which does not constitute a uniquely defined problem, i.e.

$$\begin{cases} \rho - (f_2 + f_5 + f_6) = f_0 + f_1 + f_3 + f_4 + f_7 + f_8 & (2.39) \\ -f_5 + f_6 = f_1 - f_3 - f_7 + f_8 & (2.40) \\ -(f_2 + f_5 + f_6) = -f_4 - f_7 - f_8, & (2.41) \end{cases}$$

with the unknown variables on the LHS. The same problem arises for the slip-condition, as well as for the in- and outlet conditions, where either velocity or pressure are given. This issue is usually solved with physical reasoning. What does a particular macroscopic boundary condition correspond to at the mesoscopic level? In a real fluid, the velocity at a solid wall is zero (no-slip). At the microscopic level, it can be argued that the molecules bounce of the solid wall be reversing their momentum. At the discrete mesoscopic level, this translates to the so-called bounce-back condition, in which the unknown density functions are replace by those moving in the exact opposite direction.

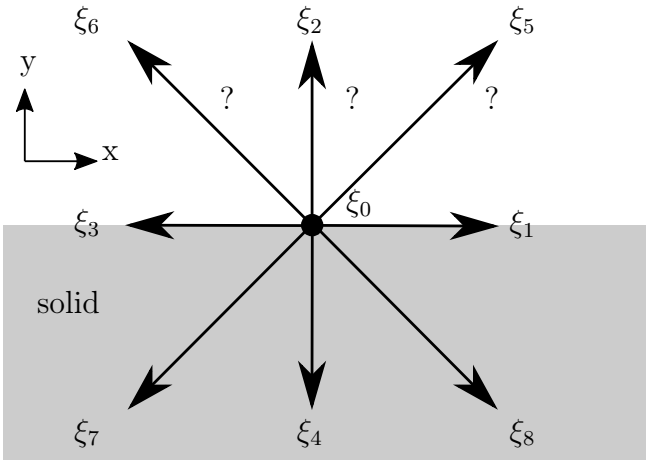


Figure 2.5: Boundary node of the D2Q9 lattice. The grey area represents a solid. The distribution functions f_2 f_5 f_6 originate from nodes that lie outside the computational domain and are therefore unknown.

In the particular case presented in Fig. 2.5, one obtains

$$f_2 = f_4 \quad f_5 = f_7 \quad f_6 = f_8.$$

It should be noted that this reasoning intrinsically contains the collision towards the thermodynamic equilibrium, so that the collision operator does not have to be applied explicitly to the functions with non-zero velocity in y-direction.

A slip condition can be seen as a symmetry condition, in which the populations are mirrored with respect to the wall. As a consequence, no mass is exchanged across the wall boundary, while the macroscopic velocity remains unchanged in the boundary node. Mathematically, this is expressed as

$$f_2 = f_4 \quad f_5 = f_8 \quad f_6 = f_7.$$

Since these are first-order approximations obtained without taking into account the macroscopic variables, it is not guaranteed that at the macroscopic level, the boundary conditions are exactly matched. In case of the bounce-back condition, the order can be increased if the interaction with the wall occurs at mid-way between two immediate neighbours, i.e. at $\Delta x/2$ [Ricot, 2002]. This can only be achieved in a volumetric or rather cell-centred formulation, when the solid boundary exactly coincides with the boarder of the computation cell.

Another approach exist to exactly match the macroscopic boundary conditions by including the raw moments. Choosing the example of a slip condition with $u_y = 0$ in case of the configuration shown in Fig. 2.5, the density and the momentum in y-direction are equally given by Eqs. (2.39) and (2.41). Combining these equations, ρ

can be defined as a function of the velocity u_y, whose value at the wall is zero. The definition of the density then simplifies to

$$\rho = f_0 + f_1 + f_3 + 2\left(f_4 + f_7 + f_8\right). \tag{2.42}$$

With ρ and $\boldsymbol{u}$ given, it is possible to calculate the Maxwellian and compute the populations in the wall node under the hypothesis

$$f_\alpha \approx f_\alpha^{eq}. \tag{2.43}$$

While in this case it is guaranteed that the flow velocity is zero at the wall, this approximation amounts to $\tilde{\omega} = 1$ (cf. Eq. (2.5)). Consequently, it changes the viscosity at the wall compared to the rest of the domain. This technique is therefore only suited for boundary conditions that contain little to no velocity gradients, so that viscous effects are negligible. This is true for the slip or homogeneous inflow condition. In order to be used for a real fluid wall boundary condition (no-slip), a corrective non-equilibrium term is required that can be obtained, *inter alia*, by extrapolation of f^{neq} from the surrounding fluid nodes [Guo et al., 2002]. Another possibility is the *Zou-He method*, which applies the bounce-back rule to the non-equilibrium function. This approach has the advantage that it retains the local character of LBM. Moreover, it captures the $f^{(2)}$ terms (c.f. (2.25)), which makes it exact for parabolic solutions. The *Zou-He method* is thus formally third-order accurate on straight boundaries. A more general boundary condition that has the same order of accuracy irrespective of the orientation and or position of the boundary was proposed by Ginzburg and D'Humières [2003]. Here the populations are reconstructed from a second-order Chapman-Enskog analysis together with a Taylor expansion of the results at the boundary.

Throughout this study we used doubly periodic boundary conditions, which apply to the macroscopic and mesoscopic variables in the same way. An additional row of nodes was added around the actual fluid domain, i.e.

$$\begin{aligned} x &= \{0, \underbrace{1, \ldots, N_x}_{\text{fluid domain}}, N_x + 1\} \\ y &= \{0, \underbrace{1, \ldots, N_y}_{\text{fluid domain}}, N_y + 1\}. \end{aligned}$$

Periodicity is then achieved by

$$\begin{cases} g_\alpha(0, y) = g_\alpha(N_x, y), \\ g_\alpha(N_x + 1, y) = g_\alpha(1, y) \end{cases}$$

and

$$\begin{cases} g_\alpha(x,0) = g_\alpha(x,N_y), \\ g_\alpha(x,N_y+1) = g_\alpha(x,1) \end{cases}$$

in x- and y-direction, respectively.

2.4 Summary

The intention of this chapter was to give the reader a general idea about the construction of LBM with hopefully more emphasis on the physical insight than on the analytical development. The *stream and collide* algorithm was derived based on a chronological consideration of the events that led from the lattice gas automata (LGA) to the first lattice Boltzmann (LB) model with BGK collision operator. Intrinsically discrete (i.e. particle approach), we have further demonstrated how to establish a link to the continuous macroscopic world via integration over velocity space. The same integration over a second-order expansion of the LBE, together with a multi-timescale analysis yields the Navier-Stokes equations. The chapter was concluded with a practical section on numerical implementation, which was illustrated with a little example code, solving the 2D LBE with periodic boundary conditions in all directions.

Chapter 3

Space-time discretization of the discrete velocity Boltzmann equation

The previous chapter has introduced the lattice Boltzmann method as a descendent of the lattice gas automata. Probability distributions instead of point particles populate physical space in discrete lattice nodes and migrate along defined trajectories that span a structured, space-filling grid. The lattice Boltzmann equation, however, can also be seen as a particular discretization of the continuous Boltzmann equation (cf. Fig. 3.1) [Sterling and Chen, 1996, Abe, 1997, He and Luo, 1997]

$$\frac{\partial f}{\partial t} + (\boldsymbol{\xi} \cdot \nabla) f = \Omega(f),$$

which did not occur to the first LBM pioneers. The BGK simplification of the collision term can be equally applied to the Boltzmann equation[1], as well as the discretization of the velocity space within the framework presented in § 2.2.4. We thus obtain the discrete velocity Boltzmann equation (DVBE)

$$\frac{\partial f_\alpha}{\partial t} + (\boldsymbol{\xi}_\alpha \cdot \nabla) f_\alpha = -\frac{1}{\tau} \left(f_\alpha - f_\alpha^{eq} \right), \tag{3.1}$$

which forms the departure point for the subsequent discussion on different space-time discretization schemes. The above expression represents a set ($\alpha = 0, 1, \ldots, q-1$) of linear partial differential equations, where f_α denotes the probability to find a particle with discrete velocity $\boldsymbol{\xi}_\alpha$. Performing a Chapman-Enskog analysis on Eq. (3.1) recovers the Navier-Stokes equations by establishing the following $\nu - \tau$ relation for the BGK

[1] In fact, it is the continuous Boltzmann equation, the BGK simplification was initially proposed for, by Bhatnagar, Gross and Kook in 1954. So well before the era of LBM.

relaxation time

$$\nu = c_0^2 \tau \quad \text{or rather} \quad \tau = \frac{\nu}{c_0^2}$$

(e.g. [Mei and Shyy, 1998]). Compared to the space and time continuous case, we note that the relaxation time obtained for the LBE differs by an additional term: $\tau = \nu/c_0^2 + \Delta t/2$ (cf. Eq. (2.38)). Consequently, the populations that are solutions to the above (space and time continuous) DVBE are not the same as the populations f of the (space and time discrete) LBE, since they are relaxed with different time scales. It is therefore necessary to unambiguously distinguish between these two solutions. Concerning this matter, it is actually not so much a question of a continuous or discrete consideration of space and time, but rather of the associated relaxation time. In the following, the duo (f, τ) will denote the populations that are associated with a relaxation time $\tau = \nu/c_0^2$. Any pair, in which the distribution function are relaxed with a different parameter, will be marked explicitly[2].

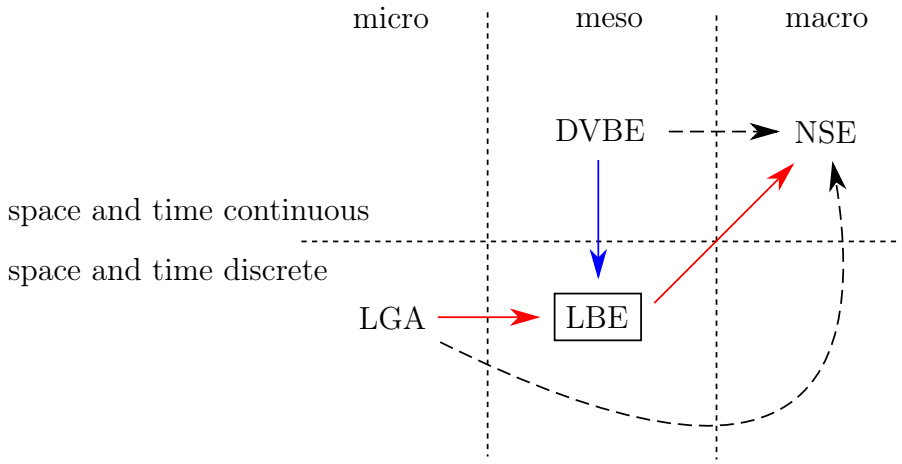


Figure 3.1: Global view on LBM. The red arrows summarize the content of chapter 2. This chapter will deal with the discretization of the DVBE leading to the LBE (blue arrow). Black dashed arrows indicate relations that are not treated explicitly in the manuscript. NSE: Navier-Stokes equations; DVBE: discrete velocity Boltzmann equations; LBE: lattice Boltzmann equations; LGA: lattice gas automata.

A discretization in the Eulerian sense (fixed frame of reference) usually constitutes the preferred method of choice in continuum fluid mechanics. Eq. (3.1), however, literally calls for a discretization in the Lagrangian sense (moving frame of reference), as the advection term on the LHS constitutes a Lagrangian derivative along the lattice vectors. Each approach has advantages over the other, which shall be illustrated in

[2]It is not our intention to deliberately confuse the reader by choosing the same notation for different variables. It rather reflects the inconsistent use of f to denote both, the solution of DVBE and LBE. Our motivation is therefore to demonstrate the nature of each solution and establish distinct notations.

§ 3.1 and § 3.2 of this chapter. The *stream and collide* algorithm presented in the previous chapter can be obtained from both approaches; however, not necessarily with the same order of accuracy. While the streaming step is only a particular case of finite differences in the Eulerian sense [Cao et al., 1997], it is the exact solution of the material derivative in the Lagrangian sense [He et al., 1998]. Nevertheless, the interest of a discretization in the Eulerian sense is actually to approximate the advection term independently of the lattice. Streaming the populations along the lattice vectors is simple and efficient, but imposes severe constraints on the mesh geometry. It makes mesh refinement an intricate matter, which will be discussed in § 3.3. The said section will also contain some original research, which sheds some light on the complexity of this issue and provides some perspectives.

3.1 Space-time discretization in the Lagrangian sense

The philosophy of the Lagrangian approach is to observe a fluid particle while following its path. In the reference frame of the moving particle, changes as a function of space only appear as a function of time. Eq. (3.1) can thus be simplified to

$$\frac{\mathrm{D}}{\mathrm{D}t} f_\alpha \left(\tilde{\boldsymbol{x}}(t), t\right) = -\frac{1}{\tau} \left(f_\alpha \left(\tilde{\boldsymbol{x}}(t), t\right) - f_\alpha^{eq} \left(\tilde{\boldsymbol{x}}(t), t\right)\right), \tag{3.2}$$

where $\mathrm{D}/\mathrm{D}t = (\partial/\partial t + \boldsymbol{\xi}_\alpha \cdot \nabla)$ denotes the Lagrangian derivative along the trajectory $\tilde{\boldsymbol{x}}(\boldsymbol{x}_{\mathrm{ref}}, t_{\mathrm{ref}}; t)$ of the particle that, at time $t = t_{\mathrm{ref}}$, was situated in point $\boldsymbol{x}_{\mathrm{ref}}$. This approach becomes particularly interesting, when the particle is advected with constant speed, which is the case for the DVBE. In this event, the trajectory, or rather characteristic, is solution of the problem

$$\frac{d}{dt} \tilde{\boldsymbol{x}}(t) = \boldsymbol{\xi}_\alpha \tag{3.3}$$

[Zienkiewicz and Codina, 1995]. Let $\boldsymbol{x} = \tilde{\boldsymbol{x}}(t)$ denote the position of the particle at time t, then at time $t+\Delta t$ its new position is $\boldsymbol{x}_{\mathrm{new}} = \tilde{\boldsymbol{x}}(t+\Delta t) = \tilde{\boldsymbol{x}}(t)+\boldsymbol{\xi}_\alpha \Delta t$, according to the above relation. Integrating Eq. (3.2) over the time interval Δt, we obtain

$$f_\alpha \left(\tilde{\boldsymbol{x}}(t+\Delta t), t+\Delta t\right) - f_\alpha \left(\tilde{\boldsymbol{x}}(t), t\right) = -\frac{1}{\tau} \int_0^{\Delta t} \left(f_\alpha \left(\tilde{\boldsymbol{x}}(t), t\right) - f_\alpha^{eq} \left(\tilde{\boldsymbol{x}}(t), t\right)\right) \mathrm{d}t. \tag{3.4}$$

It should be noted that this particular treatment of the advection term is exact and the order of the method is controlled by the approximation of the integral on the RHS

of Eq. (3.4). Choosing a first-order rectangular method, we obtain

$$f_\alpha\left(\boldsymbol{x}+\boldsymbol{\xi}_\alpha\Delta t, t+\Delta t\right) = f_\alpha\left(\boldsymbol{x},t\right) - \frac{\Delta t}{\tau}\left(f_\alpha\left(\boldsymbol{x},t\right) - f_\alpha^{eq}\left(\boldsymbol{x},t\right)\right). \tag{3.5}$$

While the above equation is seemingly identical to the original lattice Boltzmann equation presented in § 2.1, it bears a different relaxation time. The naive use of the above equation with $\tau = \nu/c_0^2$ introduces a severe $\mathcal{O}(\Delta t)$-error in the diffusion term of the Navier-Stokes equations. In particular, the corresponding momentum equation reads

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \mu_{\text{phy}}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \underbrace{- \mu_{\text{num}}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)}_{\mathcal{O}(\Delta t)\text{ - error term}},$$

with $\mu_{\text{phy}} = \rho c_0^2 \tau$ and $\mu_{\text{num}} = \rho c_0^2(\Delta t/2)$ (cf. Eq. (2.33) and (2.37)). In practice, Δt lies within the range of 10^{-5} s to 10^{-8} s, whereas τ is of the order 10^{-10} s. It means that using the first-order rectangular method, introduces a numerical diffusion that is at least three orders of magnitude higher than its physical counterpart. This method should therefore not be used, except for maybe extremely viscous fluids, or unpractically small time steps. It can thus be speculated that a higher-order discretization in time modifies the relaxation time in such a way that it cancels out the $\mathcal{O}(\Delta t)$-error term, i.e. $\mu = \rho c_0^2\left(\tau + \Delta t/2\right)$ [3]. In the following, it will be shown that this is exactly the case, if we use a second-order Crank-Nicolson scheme in conjunction with a change of variable. Approximating the RHS of Eq. (3.4) with the trapezoidal rule, one obtains the semi-implicit equation

$$f_\alpha^{t+\Delta t} = f_\alpha^t - \frac{\Delta t}{2\tau}[f_\alpha^{t+\Delta t} - f_\alpha^{eq,t+\Delta t} + f_\alpha^t - f_\alpha^{eq,t}] + \mathcal{O}(\Delta t^3), \tag{3.6}$$

with $\square^{t+\Delta t} = \square\left(\boldsymbol{x}+\boldsymbol{\xi}_\alpha\Delta t, t+\Delta t\right)$ and $\square^t = \square\left(\boldsymbol{x},t\right)$. In order to render Eq. (3.6) fully explicit, all terms that are evaluated at time $t+\Delta t$ are moved to the LHS

$$f_\alpha^{t+\Delta t} + \frac{\Delta t}{2\tau}\left[f_\alpha^{t+\Delta t} - f_\alpha^{eq,\,t+\Delta t}\right] = f_\alpha^t - \frac{\Delta t}{2\tau}\left[f_\alpha^t - f_\alpha^{eq,\,t}\right],$$

and replaced by a single variable g

$$g = f + \frac{\Delta t}{2\tau}\left(f - f^{eq}\right). \tag{3.7}$$

[3] Not to be confused with relation (2.38), where the identically named relaxation parameter actually amounts to $\tau + \Delta t/2$

The explicit equation then reads

$$g_\alpha^{t+\Delta t} = g_\alpha^t - \frac{\Delta t}{\tau}\left[f_\alpha^t - f_\alpha^{eq,\,t}\right]. \tag{3.8}$$

Eq. (3.7) implies that

$$g^{eq} = f^{eq} \quad \text{and} \quad f = g - \frac{\Delta t}{2\tau + \Delta t}\left(g - g^{eq}\right) \tag{3.9}$$

Substituting for f in Eq. (3.8) using Eq. (3.9), one obtains

$$\begin{aligned} g_\alpha^{t+\Delta t} &= g_\alpha^t - \frac{\Delta t}{\tau}\left(g_\alpha^t - \frac{\Delta t}{2\tau + \Delta t}\left(g_\alpha^t - g_\alpha^{eq,\,t}\right) - g_\alpha^{eq,\,t}\right) \\ &= g_\alpha^t - \frac{\Delta t}{\tau}\left(\frac{2\tau}{2\tau + \Delta t} g_\alpha^t - \frac{2\tau}{2\tau + \Delta t} g_\alpha^{eq,\,t}\right). \end{aligned}$$

Replacing $\tau + \Delta t/2$ by τ_g, the mesh-specific lattice parameter, finally yields the second-order lattice Boltzmann equation

$$g_\alpha^{t+\Delta t} = g_\alpha^t - \frac{\Delta t}{\tau_g}\left(g_\alpha^t - g_\alpha^{eq,\,t}\right), \tag{3.10}$$

usually expressed in the two step *stream and collide* algorithm

$$\textbf{collision:} \qquad \widehat{g}_\alpha^t = g_\alpha^t - \frac{\Delta t}{\tau_g}\left(g_\alpha^t - g_\alpha^{eq,\,t}\right) \tag{3.11a}$$

$$\textbf{streaming:} \qquad g_\alpha^{t+\Delta t} = \widehat{g}_\alpha^t. \tag{3.11b}$$

The solution to the original lattice Boltzmann equation (Eq.(2.5)) is therefore the variable pair (g, τ_g) and not the genuine populations (f, τ). Before proceeding to the next section, a few concluding remarks shall be given on the matter of the distribution functions and their associated relaxation times. Within the LBM community, there is actually quite some confusion about the notation of f and g. In literature, it is commonly accepted that second-order accuracy of the lattice Boltzmann equation can be achieved by simply adjusting the $\nu - \tau$ relation [Chen et al., 1992, Mei and Shyy, 1998, Lee and Lin, 2003]. Rarely mentioned in this context, is the fact that a modification of the relaxation time of the DVBE also changes the definition of the population with respect to f, according to the above demonstration. When remaining in a fully-discrete framework, the relation of the discrete populations to their velocity-space continuous counterpart might actually be irrelevant. This, however, changes as soon as a simulation features different local time steps. In this case, the $f - g$ relation can be used to derive a scaling function to transfer the mesh specific functions g between different resolution domains (will be discussed in § 3.3.1.1). The more severe problem, however,

is that numerous studies exist, in which the relaxation time was modified but the populations were assumed to be the genuine solutions of the DVBE [Lee and Lin, 2003, Rao and Schaefer, 2015]. Using the θ-method as a generalized formula to approximate the RHS of Eq. (3.4)

$$f_\alpha^{t+\Delta t} = f_\alpha^t - \frac{\Delta t}{\lambda}\left[(1-\theta)\left(f_\alpha^t - f_\alpha^{eq,t}\right) + \theta\left(f_\alpha^{t+\Delta t} - f_\alpha^{eq,t+\Delta t}\right)\right] \tag{3.12}$$

[Hirsch, 2007], where $\theta = \{0, 0.5, 1\}$ denotes a fully explicit, Crank-Nicolson, or fully implicit treatment of the integral, it was argued that second-order accuracy is generally maintained as long as the relaxation parameter λ is adjusted according to

$$\underbrace{\lambda + \theta\Delta t - \frac{\Delta t}{2}}_{\tau} = \frac{\nu}{c_0^2}\,, \quad \text{so that} \quad \lambda = \tau + \frac{\Delta t}{2} - \theta\Delta t \tag{3.13}$$

[Lee and Lin, 2001]. It is very important to realize that the populations in Eq. (3.12) are not necessarily the genuine populations and will hereafter be explicitly denoted as f^λ. It can easily be shown that using the θ-method with $\theta = 0$ modifies the relaxation parameter to $\lambda = \tau + \Delta t/2$. Given that in this case $\lambda = \tau_g$, Eq. (3.10) and Eq. (3.12) are identical and therefore $f^\lambda = g$. By choosing $\theta = 1/2$, one obtains $\lambda = \tau$ and Eq. (3.12) is equal to Eq. (3.6). Consequently, $f^\lambda = f$. It becomes clear that f^λ changes its relation to f as a function of θ. As a result, the θ-method in conjunction with an adjustable relaxation parameter always comes down to a semi-implicit treatment of the collision term. This lack of transparency was correctly pointed out by Bardow et al. [2006]. The θ-method with constant relaxation time τ, on the other hand, preserves the genuine functions f. In this case, $\theta = 0$ corresponds to an explicit approximation that yields the first-order LBE (3.5).

3.1.1 Stability and CFL number

Constituting only an approximation of the continuous Boltzmann equation, the LBM is susceptible to numerical instabilities. The explicit time-stepping puts the approach under classical stability constraints, such as the CFL condition and the amplification factor. Moreover, due to the discretization of velocity space that is associated with an u^3-error term (c.f. § 2.2.6), the stability of the method also depends on the mean flow velocity. As a consequence, the classical stability requirements only constitute a necessary condition for the overall stability; i.e. the scheme can still be unstable. In order to define the parametric setting, under which the method is always stable, the stability is usually assessed by a von Neumann analysis [Sterling and Chen, 1996, Niu et al., 2004, Siebert et al., 2008]. Besides the mean velocity, the linear stability of the method also depends on the weights w_α, the relaxation time τ and the wave number.

Assuming that the weights of a particular lattice are constant (e.g. the D2Q9 lattice in § 2.3) stability depends on the interplay between the mean flow velocity and the relaxation time for a given wavenumber. The maximum achievable velocity $|\boldsymbol{u}|_{max}(\tau)$ is given by a stability map. According to the definition of τ, a value of zero implies the absence of shear viscosity, which naturally defines a first stability boundary, i.e. $\tau \geq 0$ and $\tau_g \geq 0.5$, respectively. The relaxation time that corresponds to the viscosity of air or water is of the order of 10^{-10} s. The corresponding maximum velocity that still yields stable results in a standard BGK-LBM simulation lies near a Mach number of 0.42 for the most unstable wave number [Sterling and Chen, 1996]. This is the velocity limit we usually find in the classical LB simulation on standard lattices. The velocity limit may be increased by using an higher (but artificial) viscosity.

In order to demonstrate the advantage of a semi-implicit discretization of the DVBE, it suffice to employ a classical stability analysis since the relaxation time and weight are identical in both schemes (explicit and semi-implicit). As previously pointed out, the advection of particles in the LBE constitutes a perfect shift without dissipation or amplitude error, defining a Courant-Friedrichs-Lewy (CFL) number of unity. Since in the Lagrangian approach, the CFL condition is locked due to Eq. (3.3), stability issues can only arise from the local collision term. In order to assess the stability we work on a simple model equation with time-dependency only

$$\frac{\mathrm{d}f_\alpha}{\mathrm{d}t} = -\frac{1}{\tau} f_\alpha + \frac{1}{\tau} f_\alpha^{eq}.$$

Following the argumentation in [Lee and Lin, 2003], the stability essentially depends on the transient, homogeneous part of the solution that satisfies

$$\frac{\mathrm{d}f_\alpha}{\mathrm{d}t} = -\frac{1}{\tau} f_\alpha.$$

By approximating the above equation with either the first-order rectangular or the second-order trapezoidal method, we obtain

$$\phi^{t+\Delta t} = \left(1 - \frac{\Delta t}{\lambda}\right) \phi^t,$$

with $(\phi, \lambda) = \{(f, \tau);\ (g, \tau_g)\}$. The term in the brackets is identified as the amplification factor. Numerical stability requires this factor to be equal or less than unity

$$\left| \frac{\lambda - \Delta t}{\lambda} \right| \leq 1.$$

While for the first-order approximation, the time step is limited to $\Delta t < 2\tau$, i.e.

$$\left| \frac{\tau - \Delta t}{\tau} \right| \leq 1,$$

a second-order approximation is unconditionally stable, given that $\tau \geq 0$, i.e.

$$\left| \frac{\tau - \frac{\Delta t}{2}}{\tau + \frac{\Delta t}{2}} \right| \leq 1,$$

with $\tau_g = \tau + \Delta t/2$. With respect to the error term that is introduced by a first-order approximation, the above stability condition suggests that the respective lattice Boltzmann model is stable as long as the physical diffusion counterbalances the numerical diffusion. A physical time step of the order of 10^{-10} s is computationally inefficient and substantiates our previous statement that the first-order LB method should not be used. The improved stability of the semi-implicit LBE therefore manifests in a relaxation of the populations with $\tau + \Delta t/2$.

3.2 Space-time discretization in the Eulerian sense

The Eulerian approach observes the motion of a fluid from a fixed point in space. Defining a small control volume, it is possible to distinguish between changes that occur as a function of time and those that are space dependent. Instead of pathlines as in the previous approach, we are now looking at streamlines that deform over the volume. This change is governed by the spatial derivative. In an unsteady flow, such as the wake behind a cylinder at supercritical Reynolds number ($\mathrm{Re} > 40$), the alternate formation of vortices constantly changes the streamlines that the observer sees in the control volume. This change is described by the temporal derivative. It is thus possible to consider the space and time derivatives separately, if a fixed position in space is chosen. With respect to the lattice Boltzmann equation, this means that the constraint $\Delta x = \boldsymbol{\xi}_\alpha \Delta t$ no longer applies and space can be discretized independently of the lattice. In literature, this is generally referred to as *off-lattice Boltzmann method* (OLBM) [Rao and Schaefer, 2015]. In order to derive a fully-discrete off-lattice Boltzmann equation, we would naturally return to the DVBE (3.1) and find discrete expressions for the temporal and spatial derivative. A second possibility, however, exists that departs from the LBE (3.10), which was derived in the previous section. This is possible, because the approximation of the advection term in the Lagrangian approach is exact. So even if the initial equation in this approach is discrete in space and time, it has the same accuracy as the continuous advection term of Eq.(3.1). Using this characteristic-based approach inevitably leads to a coupled scheme of type Lax-Wendroff (detailed later), whereas the classical approach mostly uses explicit single-stage or multi-stage time-marching

belonging to the family of Runge-Kutta schemes. Based on these considerations, the two approaches may therefore be categorized as *characteristics-based* and *Runge-Kutta-based* [Rao and Schaefer, 2015].

3.2.1 Runge-Kutta-based methods

First attempts to derive the LBE from the DVBE actually used the finite-difference method in the Eulerian sense [Sterling and Chen, 1996, Cao et al., 1997, Abe, 1997] and not the method of characteristics in the Lagrangian sense (cf. § 3.1). So in order to discretize Eq. (3.1), a simple Euler time stepping was used in conjunction with an upwind spatial discretization on a finite-difference grid. The particle velocities $\boldsymbol{\xi}_\alpha$ of the advection term in Eq. (3.1) were chosen such that a space-filling grid is obtained, i.e.

$$\boldsymbol{\xi}_\alpha = c\,\boldsymbol{e}_\alpha,$$

where $\boldsymbol{e}_\alpha = [e_x,\, e_y]_\alpha$ is the natural basis and c is the characteristic particle speed that is proportional to the speed of sound c_0. The discrete expressions for the space and time derivatives then read

$$\frac{f_\alpha\left(\boldsymbol{x}, t+\Delta t\right) - f_\alpha\left(\boldsymbol{x}, t\right)}{\Delta t} + c\,\frac{f_\alpha(\boldsymbol{x}+\boldsymbol{e}_\alpha\Delta x, t+\Delta t) - f_\alpha(\boldsymbol{x}, t+\Delta t)}{\Delta x} = - \frac{1}{\tau}\left(f_\alpha(\boldsymbol{x}, t) - f_\alpha^{eq}(\boldsymbol{x}, t)\right).$$

Note that $\boldsymbol{e}_\alpha$ is dimensionless, so that $\boldsymbol{e}_\alpha\Delta x$ gives a direction in space. Introducing the CFL condition $\Delta x = c\Delta t$, inherent to the original LBE, and multiplying the above equation by Δt, leads to the first-order LBE (3.5) that was previously obtained via integration along the lattice characteristics

$$f_\alpha\left(\boldsymbol{x} + \boldsymbol{\xi}_\alpha\Delta t, t+\Delta t\right) = f_\alpha\left(\boldsymbol{x}, t\right) - \frac{\Delta t}{\tau}\left(f_\alpha\left(\boldsymbol{x}, t\right) - f_\alpha^{eq}\left(\boldsymbol{x}, t\right)\right).$$

Compared to the original LBE, it is argued that in order to achieve second-order accuracy for the above equation, the relaxation time τ must be replaced by the relaxation parameter $\lambda = \tau + \Delta t/2$ [Mei and Shyy, 1998]. Although this is true, it neglects the fact that this modification entails a change of variable as demonstrated in § 3.1. However, in order to apply the same change of variable, the collision integral of the DVBE has to be integrated not only over Δt but also along the characteristic defined by Eq. (3.3). As this procedure relies on a discretization in the Lagrangian sense, it is mathematically not possible to obtain the original, second-order LBE from a derivation of the DVBE in the Eulerian sense. While this might be an interesting fact, it is actually irrelevant, because the motivation behind the Eulerian approach is to discretize the

DVBE independently from the underlying lattice, i.e. $\Delta x \neq \boldsymbol{\xi}_\alpha \Delta t$, in order to gain geometrical flexibility.

First experimentations with an Eulerian type discretization used a simple explicit time marching that may be written under following form

$$f_\alpha^{t+\Delta t} = f_\alpha^t - \Delta t \mathcal{R}^t - \frac{\Delta t}{\tau}\left(f_\alpha^t - f_\alpha^{eq,t}\right)$$

[Nannelli and Succi, 1992, Mei and Shyy, 1998], with $\square^{t+\Delta t} = \square(\boldsymbol{x}, t+\Delta t)$ and $\square^t = \square(\boldsymbol{x}, t)$. $\mathcal{R}$ denotes the spatial discretization operator. A superscript specifies the discretization approach in space, namely FD = finite-difference, FV = finite-volume. A representation in finite differences then yields

$$\mathcal{R}^{\mathrm{FD},t} = \left(\boldsymbol{\xi}_\alpha \cdot \boldsymbol{\nabla}_h\right) f_\alpha^t,$$

where $\boldsymbol{\nabla}_h$ is the discretized form of the operator $\boldsymbol{\nabla}$ in Eq. (3.1). The above equation thus represents the off-lattice, finite-difference counterpart to the first-order LBE (3.5) and is generally referred to as FD-LBE. Equally, the spatial derivative may be approximated in a finite-volume approach such that

$$\mathcal{R}^{\mathrm{FV},t} = \frac{S_\beta}{V}\left(\boldsymbol{\xi}_\alpha \cdot \boldsymbol{n}_\beta\right)\left[f_\alpha\right]_\beta,$$

where V denotes the control volume confined by the surfaces S. β is the surface index, so that n_β is the respective surface normal vector and $[f_\alpha]_\beta$ the function value on the particular surface. The above equation is the finite-volume equivalent of the FD-LBE and termed FV-LBE. The finite-element method is mainly found in conjunction with the characteristics-based schemes that will be discussed in § 3.2.2.

Compared to the Lagrangian approach, the Eulerian approach leads to the appearance of the advection operator that is no longer solved in an exact manner. It is subject to the CFL stability condition that is somewhat hidden in the terms $\Delta t \boldsymbol{\xi}_\alpha \cdot \boldsymbol{\nabla}$ and $\Delta t \boldsymbol{\xi}_\alpha S_\beta / V$ for the FD-LBE and FV-LBE, respectively. In addition, the explicit time approximation of the collision term limits the time step to $\Delta t < 2\tau$ (cf. § 3.1.1). First off-lattice models were thus plagued by two severe limitations that do not apply to the original LBE. In order to improve the CFL stability range, multi-stage time-marching was introduced by using Runge-Kutta schemes of up to fifth-order [Cao et al., 1997, Reider and Sterling, 1995, Zarghami et al., 2012]. Even though some of these schemes heaved the stability threshold above a CFL number of unity – along with an enormous increase in computational cost – , the restriction on the time step of the order of the relaxation time τ persisted. Different techniques had been tested, trying to treat the collision term implicitly, including extrapolation in time [Mei and Shyy, 1998] and a predictor-corrector scheme [Lee and Lin, 2001]. Lacking either stability

or computational efficiency, it was concluded that the Eulerian approach, despite the gain in geometrical flexibility, does not constitute an alternative to the very efficient, *stream and collide* algorithm [Stiebler et al., 2006, Prestininzi et al., 2014]. The situation only improved when Guo and Zhao [2003] showed that the change of variable in the Lagrangian approach, which allows a semi-implicit treatment of the collision term, can equally be applied to the Eulerian approach. The equation they obtained reads

$$g^{t+\Delta t} = f^t - \Delta t \mathcal{R}^t - \frac{\Delta t}{2\tau}\left(f_\alpha^t - f_\alpha^{eq,\,t}\right),$$

with $g^{t+\Delta t} = f^{t+\Delta t} + \Delta t/2\tau \left(f^{t+\Delta t} - f^{eq,\,t+\Delta t}\right)$. Using the variable transformation in the opposite sense (cf. Eq. (3.9)), we may further substitute f on the RHS. In particular,

$$\begin{aligned} f &= g - \frac{\Delta t}{2\tau_g}\left(g - g^{eq}\right) \\ f - f^{eq} = f^{neq} &= \frac{\tau}{\tau_g} g^{neq} = \frac{\tau}{\tau_g}\left(g - g^{eq}\right) \quad \text{cf. § 3.3.1.1.} \end{aligned}$$

One finally obtains

$$g_\alpha^{t+\Delta t} = g_\alpha^t - \Delta t \mathcal{R}^t - \frac{\Delta t}{\tau_g}\left(g_\alpha^t - g_\alpha^{eq,\,t}\right), \tag{3.14}$$

with $\tau_g = \tau + \Delta t/2$. Unaffected by the variable transformation, it should be noted that the advection operator in the above equation is still explicit and $\mathcal{O}(\Delta t)$. In order to obtain the Eulerian approach equivalent of the classical LBE, global second-order accuracy is required. In space, this is simply achieved with a centred-difference scheme. In time, however, second-order accuracy is more delicate. The semi-implicit treatment of the collision term encourages the approximation of the advection term at a temporal mid-point, i.e. $\Delta t \mathcal{R}^{t+\Delta t/2}$. Requiring implicit information on f, literature provides two conceptionally different solutions presented for the FV-LBM. In a series of papers dealing with so-called discrete unified gas kinetic schemes (DUGKS) [Guo et al., 2013, 2015, Zhu et al., 2017], the authors propose to use the classical *stream and collide* algorithm within a half-time step. A more recent alternative was proposed by Shrestha [2015], Shrestha et al. [2016]. Here, a Heun predictor-corrector scheme is applied to achieve a second-order accuracy in time of the flux term. Both studies present stable results, where the additional expense in computational costs is outweighed by a gain in geometrical flexibility. The two strategies are elucidated in the following.

3.2.1.1 DUGKS

The flux term

$$\mathcal{R}^{t+\Delta t/2} = \frac{S_\beta}{V}\boldsymbol{\xi}_\alpha \cdot \boldsymbol{n}_\beta \left[f_\alpha(\boldsymbol{x}_\beta, t + \Delta t/2)\right]$$

still contains the genuine probability distributions f_α of the DVBE. Considering, for now, a regular Cartesian grid, one possibility to obtain $f_\alpha(\boldsymbol{x}_\beta, t + \Delta t/2)$ is thus to integrate Eq. (3.1) within half a time step $h = \Delta t/2$ along the lattice vectors that terminate in $\boldsymbol{x}_\beta$ as shown in Fig. 3.2. We thus obtain the *stream and collide* formulation

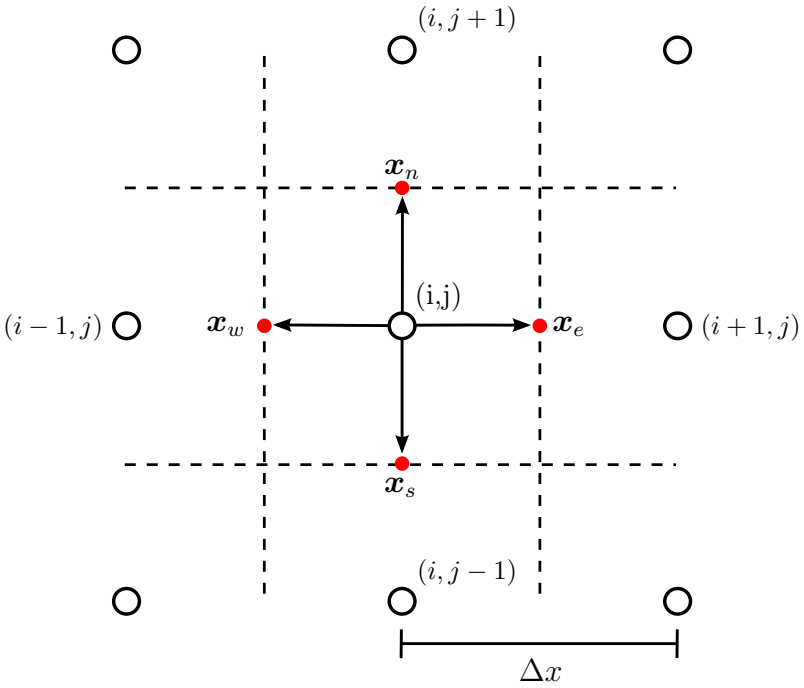


Figure 3.2: Computation of the surface fluxes by a streaming step over a half-time step.

of the mesh-specific populations g_α for a refined (by a factor of two) lattice, which is indicated by the superscript f:

$$g^f_\alpha(\boldsymbol{x}_\beta, t + h) = \widehat{g}^f_\alpha(\boldsymbol{x}_\beta - \boldsymbol{\xi}_\alpha h, t). \tag{3.15}$$

Due to the refinement in lattice space, the post-collision term on the RHS of Eq. (3.15) is not known for the entire set of lattice directions. In particular, immediate neighbours only exist in surface normal direction. Following Guo et al. [2013, 2015], the post-collision term can be approximated by a Taylor expansion

$$\widehat{g}^f_\alpha(\boldsymbol{x}_\beta - \boldsymbol{\xi}_\alpha h, t) = \widehat{g}^f_\alpha(\boldsymbol{x}_\beta, t) - \boldsymbol{\xi}_\alpha h \cdot \boldsymbol{\sigma}_\beta, \tag{3.16}$$

with $\boldsymbol{\sigma}_\beta$ being the gradient of $\widehat{g}_\alpha$ in $\boldsymbol{x}_\beta$. The original algorithm was presented for the one-dimensional case. In two dimensions, the dot product in Eq. (3.16) requires that in a surface point, for example $\boldsymbol{x}_n$, with $n = j + 1/2$ and j being the mesh index, the gradients are known in the respective coordinate directions x_i and x_j (cf. Fig. 3.2). Since in case of the northern surface flux, there are no immediate neighbours in i-direction, the gradient has to be interpolated with the information from adjacent nodes. Having followed so far the reasoning in Guo et al. [2013, 2015], we will now slightly deviate from the original algorithm. We assume that, at time t, the mesh-specific distribution functions g_α are known, everywhere. Still in accordance with the DUGK schemes, we thus apply the following equation to obtain the refined post-collision function $\widehat{g}^f_\alpha$

$$\widehat{g}^f_\alpha = \frac{4\tilde{\tau}_g - 3}{4\tilde{\tau}_g} g_\alpha + \frac{3}{4\tilde{\tau}_g} g^{eq}_\alpha, \tag{3.17}$$

(see appendix A.1 for derivation), where $\tilde{\tau}_g = \dfrac{\tau}{\Delta t} + \dfrac{1}{2}$ is the lattice parameter. Combining Eq. (3.15) with Eq. (3.16), a relation is established between the pre- and the post-collisional state of the mesh-specific populations g_α on the surface $\boldsymbol{x}_\beta$, i.e.

$$g^f_\alpha(\boldsymbol{x}_\beta, t + h) = \widehat{g}^f_\alpha(\boldsymbol{x}_\beta, t) - \boldsymbol{\xi}_\alpha h \cdot \boldsymbol{\sigma}_\beta.$$

As a next step, the generic transfer function (3.9) is used to obtain the genuine distribution function f_α

$$\begin{aligned} f_\alpha &= g^f - \frac{\Delta t^f}{2\tau + \Delta t^f} \left(g^f - g^{eq}\right) \\ &= \frac{1}{2\tilde{\tau}^f_g} \left(\left(2\tilde{\tau}^f_g - 1\right) g^f_\alpha + g^{eq}_\alpha\right). \end{aligned} \tag{3.18}$$

The algorithm employed to achieve a second-order temporal approximation of the flux term can thus be summarized as followed

$$\begin{aligned} g_\alpha(\boldsymbol{x}, t) &\xrightarrow{(3.17)} \widehat{g}^f_\alpha(\boldsymbol{x}, t) \xrightarrow{(3.16)} \widehat{g}^f_\alpha(x_\beta, t) \text{ and } \sigma_\beta \\ &\xrightarrow{(3.15)} g^f_\alpha(x_\beta, t + h) \xrightarrow{(3.18)} f_\alpha(x_\beta, t + \Delta t/2). \end{aligned}$$

In the original work, the spatial fluxes were approximated with a second-order centred scheme. The performance of the DUGK scheme was evaluated for multi-scale flow physics on uniform grids [Guo et al., 2015] and boundary layer flows on a structured, non-uniform grid [Zhu et al., 2017]. Results have shown that the scheme is stable for CFL < 1. At a CFL of 0.95 it was reported to achieve a better stability than the classical LBM at unit CFL [Guo et al., 2013].

3.2.1.2 Heun predictor-corrector scheme

The second possibility was proposed by Shrestha [2015]. Here, a so-called Heun predictor-corrector scheme is used to achieve a second-order time approximation of the surface fluxes. Following Eq. (3.9) with $\tilde{\tau}_g = \tau/\Delta t + 0.5$, we substitute for the mesh-specific probability distributions g_α and compute a prediction, then denoted g^*_α, by solving Eq. (3.14) with a first-order explicit Euler scheme ($\mathcal{R}^t$) for the distribution function at the surface

$$\mathcal{R}^t = \frac{S_\beta}{V}\boldsymbol{\xi}_\alpha \cdot \boldsymbol{n}_\beta \left[\frac{1}{2\tilde{\tau}_g}\left(\left(2\tilde{\tau}_g - 1\right) g_\alpha(\boldsymbol{x}_\beta, t) + g^{eq}_\alpha(\boldsymbol{x}_\beta, t)\right)\right]. \tag{3.19}$$

This intermediate solution is then used to construct an semi-implicit trapezoidal rule for the flux term

$$\mathcal{R}^{t+h} = \frac{S_\beta}{V}\boldsymbol{\xi}_\alpha \cdot \boldsymbol{n}_\beta \left[\frac{\frac{1}{2\tilde{\tau}_g}\left(\left(2\tilde{\tau}_g - 1\right) g^*_\alpha(\boldsymbol{x}_\beta, t+\Delta t) + g^{*eq}_\alpha(\boldsymbol{x}_\beta, t+\Delta t)\right)}{2}\right] + \frac{\mathcal{R}^t}{2}. \tag{3.20}$$

In contrast to the DUGK schemes, the authors proposed the QUICK scheme [Leonard, 1979] for the spatial approximation of the surface fluxes. The algorithm was tested against the classical LBM algorithm in a series of test cases. While the computational cost per time step was estimated being eight to ten times higher than the streaming of the distribution functions, efficiency was recovered by using specific non-uniform grids that are optimized to the expected flow field [Shrestha et al., 2016].

In general, the second-order accurate discrete Boltzmann equation in finite-volume formulation reads

$$g_\alpha(\boldsymbol{x}, t+\Delta t) = \widehat{g}_\alpha(\boldsymbol{x}, t) - \Delta t \mathcal{R}^{t+h}, \tag{3.21}$$

where the flux term can be evaluated by either of the above presented algorithms. Due to the presence of the pre- and post-collisional state on the right-hand side, it becomes obvious that (without further adjustment) this algorithm has to be solved as a single equation. This contrasts the *stream and collide* algorithm, where the two processes are splitted. We will nevertheless demonstrate in § 4.1 that a separate treatment of collision and advection is possible for the Heun predictor-corrector scheme.

3.2.2 Characteristics-based scheme

It might seem contradictory to categorize a characteristics-based scheme as a discretization in the Eulerian sense. Nevertheless, we have learnt that essential to the Eulerian approach is the fixed observer. The idea behind the characteristics-based discretization is therefore to expand the streaming step in a Taylor series around a point $\boldsymbol{x}$, so that

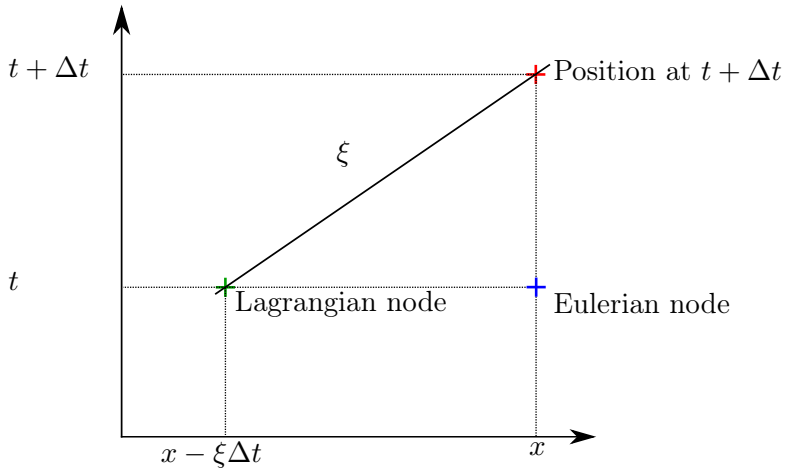


Figure 3.3: Iteration in a fixed (Euler) and moving (Lagrange) frame of reference. In order to obtain the solution in the Eulerian node at time t, the solution is approximated from the adjacent Lagrangian node.

the temporal evolution of the advection term in considered in a fixed position (cf. Fig. 3.3). The second-order LBE in pull-configuration may therefore be approximated by

$$\begin{aligned} g_\alpha(\boldsymbol{x}, t+\Delta t) &= \widehat{g}_\alpha(\boldsymbol{x} - \boldsymbol{\xi}_\alpha \Delta t, t) \\ &\approx \widehat{g}_\alpha(\boldsymbol{x}, t) - \xi_{\alpha i}\Delta t \frac{\partial \widehat{g}_\alpha}{\partial x_i} + \xi_{\alpha i}\xi_{\alpha j}\frac{\Delta t^2}{2}\frac{\partial^2 \widehat{g}_\alpha}{\partial x_i \partial x_j} + \mathcal{O}(\Delta t^3). \end{aligned}$$

If we now substitute for $\widehat{g} = g - \dfrac{\Delta t}{\tau_g}(g - g^{eq})$, one obtains

$$\begin{aligned} g_\alpha^{t+\Delta t} = g_\alpha^t &- \Delta t \left\{ \xi_{\alpha,i}\frac{\partial g_\alpha^t}{\partial x_i} + \frac{1}{\tau_g}\left(g_\alpha^t - g_\alpha^{eq,t}\right) \right\} \\ &+ \frac{\Delta t^2}{2}\xi_{\alpha,i}\frac{\partial}{\partial x_i}\left\{ \xi_{\alpha,j}\frac{\partial g_\alpha^t}{\partial x_j} + \frac{2}{\tau_g}(g_\alpha^t - g_\alpha^{eq,t}) \right\} \\ &- \frac{\Delta t^3}{2\tau_g}\xi_{\alpha,i}\frac{\partial}{\partial x_i}\left\{ \xi_{\alpha,j}\frac{\partial (g_\alpha^t - f_\alpha^{eq,t})}{\partial x_j} \right\} + \mathcal{O}(\Delta t^3). \end{aligned}$$

Note the last term on the RHS is of $\mathcal{O}(\Delta t^2)$ since the relaxation time $\tau_g = \tau + \Delta t/2$ is of $\mathcal{O}(\Delta t)$. The above equation was proposed by [Bardow et al., 2006, 2008] and is usually referred to a BKG scheme (not to be confused with the abbreviation BGK for the collision operator). Approximating the spatial derivatives with finite elements, this scheme joins the group of globally second-order off-lattice equations that constitute an alternative to the classical LBE. Similar to the Runge-Kutta based schemes there is a little bit of history around the development of this approach that was intentionally withhold for the following reasons. First of all, in the development of this method,

there is a lack of consistency in the use of f and g for the reasons previously discussed in § 3.1. As a result, Eq. (13) and Eq. (16) in [Rao and Schaefer, 2015] are identical despite the diverging variable notation. Secondly, for the derivation of the discrete formulation, there is no need to pass via the genuine functions f as in the Runge-Kutta based schemes. The BKG scheme is directly obtained from the semi-implicit LBE (3.10), which elegantly circumvents the hassle of variable transformation. In [Zhu et al., 2017], the authors compared the BKG scheme with the previously presented DUGK scheme. For the sake of better comparability, the BKG scheme was expressed in a finite-volume formulation. Due to a first-order approximation of the collision integral of the flux term, the BKG scheme is, in general, less accurate than the DUGK scheme. This was underlined in a convergence study, where the performance of the BKG scheme deteriorated comparatively more with increasing grid size [Zhu et al., 2017].

3.3 Mesh refinement in the Lattice Boltzmann method

In the previous sections, two fundamentally different approaches have been presented to discretize the DVBE (3.1). The Lagrangian approach yields the classical LBE that was originally derived from the lattice gas automata. Defining the advection velocity as the ratio of lattice spacing Δx over time step Δt, the approximation of the advection term is exact and the streaming translates into an extremely efficient index shift in memory. The penalty paid for these benefits is the restriction to uniform Cartesian grids. In the Eulerian approach, the advection operator is evaluated explicitly for each lattice direction. Moreover, multi-stage time-marching schemes require its repeated evaluation during one time step, making this approach computationally more expensive . Single-stage explicit schemes, on the other hand, have a comparatively low CFL stability range. The great advantage of this approach, however, is the independence from the uniform spacetime crystal (lattice), which allows for full geometrical flexibility in terms of spatial discretization of the fluid domain. The ultimate goal is therefore to converge towards an efficient algorithm that is capable to solve the lattice Boltzmann equation for arbitrary spatial grids. Due to the additional evaluation of the advection operator, it is not granted to the Eulerian approach, to achieve the same efficiency as the Lagrangian approach. For the Lagrangian approach, on the other hand, different strategies exist to expand the method to non-uniform grids. A family of schemes, including interpolation-supplemented LBM [He et al., 1996, He and Doolen, 1997b,a], semi-Lagrangian [Krämer et al., 2017] and Taylor-expansion LBM [Han et al., 2007] is based on the assumption that the distribution functions are continuous in space and time and may therefore be obtained in any position by spatial interpolation or series

expansion. The additional approximation step, however, deprives the Lagrangian approach of its efficiency. Moreover, the resolved fluid dynamics scales can not contain more information than what is supported by the underlying lattice spacing. A more efficient approach, which has by far received the biggest attention, constitutes the class of *multi-scale lattice Boltzmann schemes* [Filippova et al., 2001]. The basic idea behind these schemes is to solve the LBE on multiple uniform subdomains with different resolutions that communicate via refinement interfaces. Being the main mesh refinement strategy in a range of commercial and non-commercial LBM solver (PowerFLOW, XFlow, ProLB (former LaBS), Palabos, waLBerla), this section shall convey the main steps of the mesh refinement algorithms and highlight existing challenges.

3.3.1 Multiscale lattice Boltzmann scheme

Before presenting the lattice Boltzmann method for multiple scales, it is helpful to recall the limitations that come along with the Lagrangian approach in the light of mesh refinement. One might say that the "root of all evil" lies in the seemingly inconspicuous Eq. (3.3). It ties the spatial mesh to the lattice and also the grid size to the time step. Lattice crystals constitute rigid structures that can be isotropically scaled but not changed topologically. In order to achieve connectivity at the interface when refining the mesh, the cell dimensions in each direction thus have to be decreased by a factor of two (or a multiple of two), creating quad-tree and octree data structure in two and three dimensions, respectively. It also implies that the time step on a refined mesh is divided by the same factor. In § 3.1, we have learnt that the distribution functions g_α of the second-order LBE (3.10) depend on the time step. They are thus discontinuous over a refinement interface and need to be scaled before being transferred between grids of different mesh size. While this scaling operation constitutes an element of any second-order algorithm that involves varying time levels Δt, connecting different resolutions of space and time may be managed in two ways that are conceptually distinct. The *volumetric* or rather *cell-based* formulation introduced by [Chen, 1998, Chen et al., 2006] considers the lattice nodes to be located in the cell centres of the underlying mesh (Fig. 3.4a). The *vertex-* or *node-based* formulation [Filippova and Hänel, 1998a,b], on the other hand, assumes that the lattice nodes coincide with the vertices of the spatial mesh (Fig. 3.4b). In practice, PowerFLOW and waLBerla rely on the first approach, whereas ProLB and Palabos apply the latter.

3.3.1.1 Scaling

To begin with, we are going to present the scaling operation that is an element of both formulations (cell-based, node-based). Given that $g^{eq} = f^{eq}$, the variable transforma-

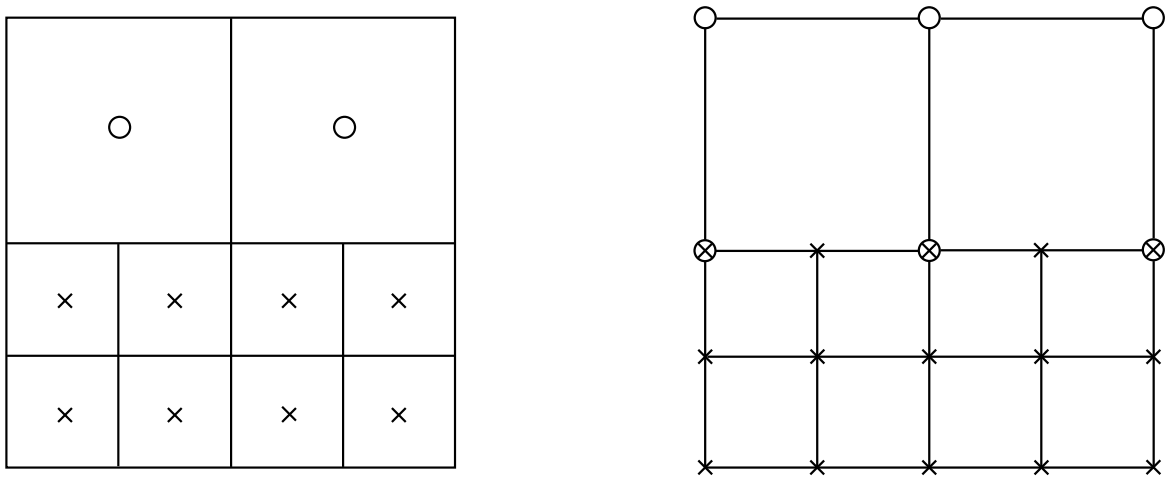


Figure 3.4: Sketch of the cell-based (left) and node-based (right) approach in two dimensions.

tion (3.9) reduces to

$$f^{neq} = \left(1 - \frac{\Delta t}{2\tau + \Delta t}\right) g^{neq} \equiv \left(\frac{2\tau}{2\tau + \Delta t}\right) g^{neq} \equiv \left(\frac{\tau}{\tau_g}\right) g^{neq}.$$

Let the superscript c and f denote a variable on a coarse and a refined grid, respectively, such that $\Delta t^c = m\Delta t^f$, where $m = 2^n$ is the refinement factor with $n \in \mathbb{N}_{>0}$. Given that the non-equilibrium part of the genuine distribution function is continuous over a refinement interface, we may write

$$f^{neq,c} = f^{neq,f} \equiv \frac{\tau}{\tau_g^c} g^{neq,c} = \frac{\tau}{\tau_g^f} g^{neq,f}. \tag{3.22}$$

Substituting for the non-dimensional relaxation parameter $\tilde{\tau}_g = \tau_g / \Delta t$ yields

$$g^{neq,c} = \underbrace{m \frac{\tilde{\tau}_g^c}{\tilde{\tau}_g^f}}_{\substack{\zeta \\ f \to c}} g^{neq,f} \quad \text{or rather} \quad g^{neq,f} = \underbrace{\frac{1}{m} \frac{\tilde{\tau}_g^f}{\tilde{\tau}_g^c}}_{\substack{\zeta \\ c \to f}} g^{neq,c}.$$

By using this relation, it is now possible to calculate a coarse-grained solution on a refined domain and vice versa

$$g^c = g^{eq} + \underset{f \to c}{\zeta}\, g^{neq,f} \quad \text{or rather} \quad g^f = g^{eq} + \underset{c \to f}{\zeta}\, g^{neq,c}, \tag{3.23}$$

where ζ represents the respective scaling parameter. The fact that the viscosity must be independent of the refinement level, since it is a physical property of the fluid, is

used to find an expression for the scaling parameter as a function of either τ_g^c or τ_g^f, only. In particular, we may write

$$\nu = c_0^2 \left(\tilde{\tau}_g^f - \frac{1}{2} \right) \Delta t_f = c_0^2 \left(\tilde{\tau}_g^c - \frac{1}{2} \right) \Delta t_c.$$

By choosing $\Delta t_c = m \Delta t_f$ we obtain

$$\tilde{\tau}_g^f = m \left(\tilde{\tau}_g^c - \frac{1}{2} \right) + \frac{1}{2} \quad \text{or rather} \quad \tilde{\tau}_g^c = \frac{1}{m} \left(\tilde{\tau}_g^f - \frac{1}{2} \right) + \frac{1}{2}. \tag{3.24}$$

At last, we state the scaling of the distribution function as a function of the respective relaxation parameter. For the transfer of information from the fine to the coarse grid this is

$$g_\alpha^c = g_\alpha^{eq} + \frac{2\tilde{\tau}_g^f + m - 1}{2\tilde{\tau}_g^f} g_\alpha^{neq,f}.$$

In terms of numerical implementation, it is more practical to express the scaling function with respect to the full distribution function g and its equilibrium component g^{eq}. Thus, one obtains

$$\begin{aligned} g_\alpha^c &= \frac{2\tilde{\tau}_g^f + m - 1}{2\tilde{\tau}_g^f} g_\alpha^f - \frac{m-1}{2\tilde{\tau}_g^f} g_\alpha^{eq} \\ &= \frac{1}{2\tilde{\tau}_g^f} \left[\left(2\tilde{\tau}_g^f + m - 1 \right) g_\alpha^f - (m-1)\, g_\alpha^{eq} \right]. \end{aligned}$$

Similarly, for the transfer of distribution functions from the coarse to the fine grid we obtain

$$\begin{aligned} g_\alpha^f &= g_\alpha^{eq} + \frac{2\tilde{\tau}_g^c + m^{-1} - 1}{2\tilde{\tau}_g^c} g_\alpha^{neq,c} \\ &= \frac{2\tilde{\tau}_g^c + m^{-1} - 1}{2\tilde{\tau}_g^c} g_\alpha^c + \frac{m^{-1} - 1}{2\tilde{\tau}_g^c} g_\alpha^{eq} \\ &= \frac{1}{2\tilde{\tau}_g^c} \left[\left(2\tilde{\tau}_g^c + m^{-1} - 1 \right) g_\alpha^c + \left(m^{-1} - 1 \right) g_\alpha^{eq} \right] \end{aligned}$$

Depending on the spatial resolution of the flow problem, not all turbulent dynamic scales are resolved. These subgrid scale effects are usually taken into account with an additional (subgrid-scale) viscosity ν_{sgs}. Within the LB approach, this heuristic assumption translates into the correction of the relaxation time so that

$$\tau = \frac{\nu_{tot}}{c_0^2},$$

with $\nu_{tot} = \nu + \nu_{sgs}(\boldsymbol{x}, t)$. Time and space dependency of the total viscosity and the relaxation time is omitted in the following. Under this assumption the lattice based non-dimensional relaxation time $\tilde{\tau}_g$ is then expressed as

$$\tilde{\tau}_g = \frac{1}{2} + 3\tilde{\nu}_{tot},$$

with the viscosity normalized by $\tilde{\nu}_{tot} = \nu_{tot}/3c_0^2\Delta t$. From the above equation it follows that

$$\tilde{\nu} = \frac{\tilde{\tau}_g - 0.5}{3} - \tilde{\nu}_{sgs}.$$

Since $\Delta t^c = m\Delta t^f$, this implies that $\tilde{\nu}^f = m\tilde{\nu}^c$ and one finally obtains

$$\tilde{\tau}_g^c = \frac{1}{2} + 3\left(\underbrace{\frac{1}{m}\left(\frac{\tilde{\tau}_g^f - \frac{1}{2}}{3} - \tilde{\nu}_{sgs}^f\right)}_{\tilde{\nu}^c} + \tilde{\nu}_{sgs}^c\right),$$

which can be simplified to

$$\tilde{\tau}_g^c = \frac{1}{2} + \frac{1}{m}\left(\tilde{\tau}_g^f - \frac{1}{2}\right) + 3\left(\tilde{\nu}_{sgs}^c - \frac{\tilde{\nu}_{sgs}^f}{m}\right)$$

(see also in reference [Touil et al., 2014], Eq. (17) with $m = 2$). "Generally speaking, a kinematic viscosity is a diffusion coefficient that may be viewed as the product of a characteristic length scale and a characteristic velocity (at that scale). Concerning the subgrid-scale viscosity, these two characteristic quantities should be identified with the local grid spacing, Δx and the typical velocity difference at that grid scale, $\delta u(\Delta x)$. According to Kolmogorov's theory, one can establish that $\delta u(\Delta x) \sim (\Delta x)^{1/3}$" [Touil et al., 2014]. The change in resolution by a factor m then leads to

$$\tilde{\nu}_{sgs}^c(x,t) = m^{\frac{1}{3}}\tilde{\nu}_{sgs}^f(x,t)$$

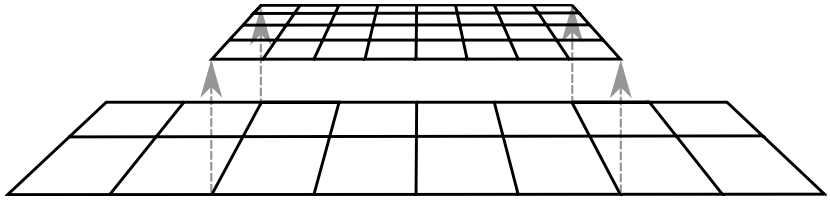
and finally

$$\tilde{\tau}_g^c(x,t) = \frac{1}{2} + \frac{1}{m}\left(\tilde{\tau}_g^f(x,t) - \frac{1}{2}\right) + \frac{3}{m}\left(\tilde{\nu}_{sgs}^f(x,t)\left(m^{\frac{1}{3}} - 1\right)\right).$$

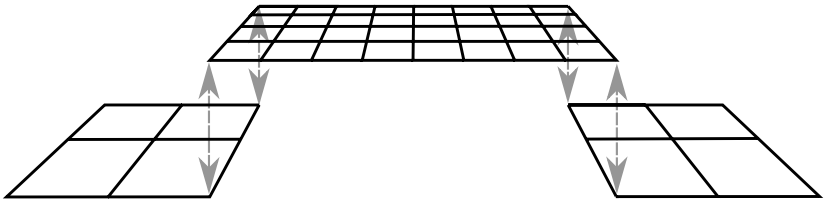
3.3.1.2 Grid coupling

Before diving into the specific steps of the *vertex-based* and the *cell-based* algorithm, there is one further choice to be made. According to Lagrava et al. [2012], two possi-

bilities exist to integrate a refined patch into a uniform root mesh. In the *multi-grid* approach, the coarse mesh fills the entire computational domain and provides the boundary conditions for the refined patch as seen in Fig. 3.5a. It can be considered as the donor grid, whereas the refined patch only receives information, but does not provide it. The *multi-grid* approach is straightforward to implement, since only one-directional transfer of information is considered. It bears, however, the drawback of becoming computationally inefficient when multiple refinement levels are applied. The alternative is a *multi-domain* approach. Here, the refined patch is locally embedded into the coarse mesh (Fig. 3.5b). Both domains are thus donor and receiver, which adds an additional step to the refinement algorithm. Apart from the interface nodes, only a single solution exists though, which improves the CPU performance.



(a) multi-grid



(b) multi-domain

Figure 3.5: Schematic view of the two possible decomposition techniques in the multiscale LBM in two dimensions.

To the author's best knowledge, the *multi-grid* approach as it is described in Lagrava et al. [2012], rarely applies in practice. The one-directional data transfer prevents small-scale information obtained in a refined area to be communicated to the root domain. As a consequence, far field propagation to a region with a particular refinement level yields the same solution as a single-grid approach with a resolution of the respective level, usually the coarsest. This approach is therefore only suitable for near field flow phenomena that are locally confined to a single resolution domain. However, from a fundamental point of view, the *multi-grid* approach allows us to test, independently, the transfer of information form coarse to fine and vice versa. Moreover, this approach relies on multiple structured grids that are hierarchically ordered in a tree, whereas the *multi-domain* approach relies on a single Cartesian, but unstructured grid. Extended to bidirectional coupling, the *multi-grid* approach forms the basis for adaptive mesh refinement (AMR) techniques. Without going into too much detail, this method adapts

the grid resolution in response to the local flow features, based on some error indicator that is usually linked to vorticity [Fakhari and Lee, 2014]. With a communication in both directions, the solution is computed only on the grid with the highest refinement level (smallest grid spacing). Underlying grids of lower order in the refinement hierarchy are ignored expect for the interface nodes [Eitel-Amor et al., 2013]. One may further distinguish between patch-based AMR and tree-based AMR [Drui, 2017]. The former locally introduces refined patches to regions that have been flagged for refinement. In tree-based AMR, on the other hand, single cells are independently refined into four cells generating a quadtree or rather eight cells generating an octree, in 2D and 3D, respectively. Hence, this algorithm makes use of the tree structure to store the mesh, modify it and go through all its cells, whereas the patch-based AMR and the *multi-grid* approach only use the tree to handle the grid hierarchy. The *multi-domain* approach makes even less use of the tree structure and only shares the factor two refinement constraint. Nevertheless, the multiscale lattice Boltzmann method is usually associated with a tree data structure, even though this applies only to some extend.

In the following two paragraphs, grid coupling will be described for the *multi-domain* approach, which contains all the steps of the *multi-grid* approach, as data is transferred in both directions. The coupling is illustrated for the cell-based and the vertex-based approach. The subject of adaptive mesh refinement will no further be covered in this manuscript.

Vertex-based approach: Mesh refinement in the vertex based approach generates an interface of collocated (C) and hanging nodes (H). A coarse interface node can only be of type C, whereas fine interface nodes are both C and H, alternately. Common to all interface nodes is the lack of a complete set of neighbours, which prevents the natural update via the streaming process on the interface (Fig. 3.6).

For the purpose of illustration, the streaming step in Fig. 3.6 is presented in a pull-configuration. The set of distribution functions are pulled towards a single node, instead of scattered to their neighbour nodes

$$g_\alpha(\boldsymbol{x}, t+\Delta t) = \widehat{g}(\boldsymbol{x} - \boldsymbol{\xi}_\alpha \Delta t, t) \quad \text{with} \quad \widehat{g} = g - \frac{\Delta t}{\tau_g} g^{neq}.$$

The above formulation of the streaming step leads to exactly the same result as the push-configuration that was derived in Eq. (3.10).

For the coarse lattice, the position of each missing neighbour coincides with a fine node, whereas the invalid characteristics of the fine lattice arise from "no man's land" (compare the left and the right set of velocity vectors in Fig. 3.6). This has consequences for the computation of the respective interface nodes. While coarse interface nodes are updated by completing the full set of neighbours with the populations of the co-located fine nodes (bottom row in the left part of Fig. 3.6) followed by a streaming,

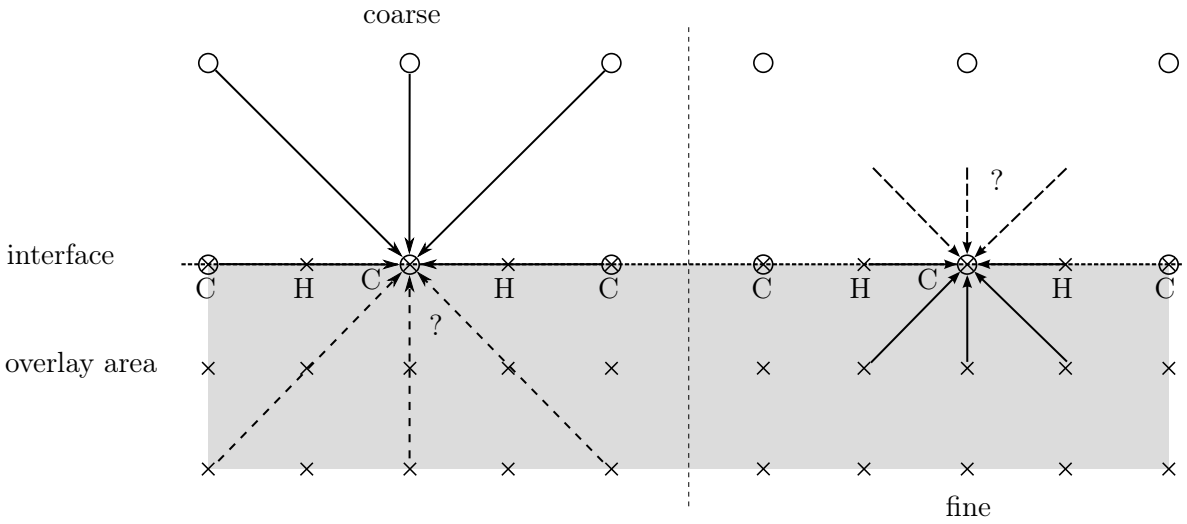


Figure 3.6: Initial state: all nodes (×: fine; ◯: coarse) contain the post-collision states a time t. During the successive streaming step, the entire set of functions in a node leaves (apart from g_0) and a new set comes in. In the boundary nodes, certain incoming distributions (dashed arrows) are missing due to a failure of connectivity between nodes at the interface.

the respective information in the fine interface nodes is obtained (via interpolation if necessary) of the updated coarse interface nodes, directly.

It follows a step-by-step presentation of the mesh refinement algorithm in the vertex-based approach that will be accompanied by illustrative, one-dimensional, space-time diagrams. The abscissa in these diagrams is the line that runs vertically through the center of the lattices shown in Fig. 3.6. The scale of the ordinate represents a single coarse time step. It should be noted that the only difference compared to a representation in two or three dimensions is the absence of hanging nodes in one dimension. The principle steps and their order of execution, however, remain unaffected by this simplification.

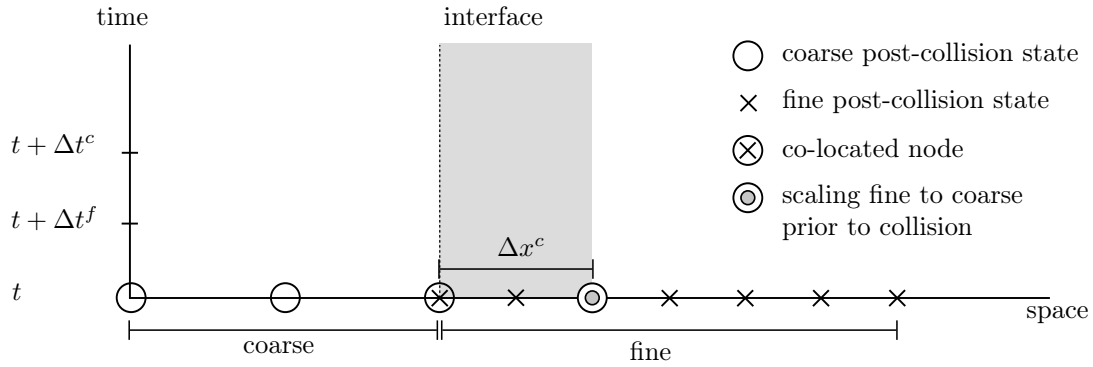


Figure 3.7: Step 1 – Decimation of the distribution functions in the fine nodes (scaling fine to coarse and filtering) followed by a copying of the data to the coarse grid, which creates a co-located node.

Step 1 of the algorithm constitutes the creation of an overlay area (cf. Fig. 3.6), which adds, in 2D, a row of nodes to the coarse grid based on the information from the underlying fine grid. Consequently, in one dimension, only a single node is added to the coarse domain. This process is usually referred to as *decimation* [Lagrava et al., 2012] and involves two stages that are summarized in Fig. 3.7. We assume that an additional node (in 1D) is allocated to the coarse domain. First, the information in respective co-located fine nodes is scaled from *fine to coarse* according to relation (3.23). Given that the spectrum of supported wavelengths is reduced on the coarse grid, the distribution function are also subject to a restriction, which can, for example, be achieved with a low pass filter [Lagrava et al., 2012, Touil et al., 2014]. Finally, the information is copied to the coarse node, creating an overlay area of one grid cell, where two solutions co-exist. In the original algorithm [Filippova and Hänel, 1998b], the decimation was applied to a post-collision distribution. Dupuis and Chopard [2003], nevertheless, pointed out that a singularity for $\tau_g = \Delta t/2$ can be avoided, if the decimation is applied to the pre-collision populations. Illustrated in Fig. 3.7 is therefore the decimation as the creation of an overlay area, followed by the collision of the distribution functions in all nodes.

During *step* 2, all nodes are subject to the streaming process (3.8). The coarse domain, including the interface, is thus entirely updated to time $t + \Delta t^c$. The fine domain is updated to time $t + \Delta t^f$ everywhere except on the interface. We realize that the co-located node on the transition interface (left $\otimes$ in Fig. 3.8) has received both coarse distribution functions, but no fine distribution [4].

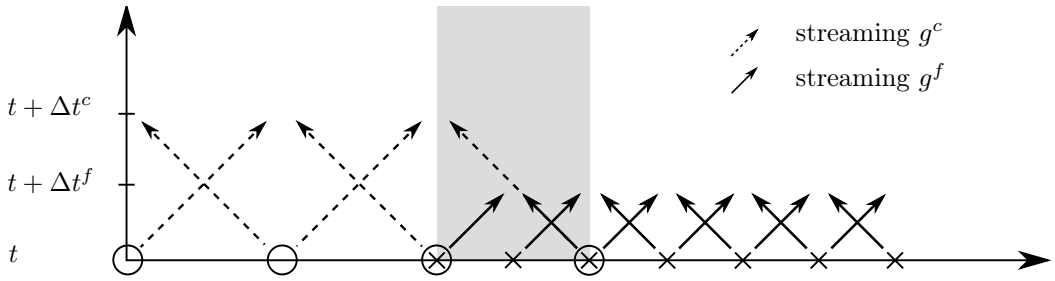


Figure 3.8: Step 2 – Streaming of all distribution functions.

Before equalizing the time levels with a second iteration on the fine grid, the information in the fine interface node is reconstructed in *step* 3 that is shown in Fig. 3.9. This is achieved by a temporal interpolation using the coarse grained distribution functions on the interface at time $t + \Delta t^c$. Prior to this operation, the populations g^c have to be scaled from *coarse to fine* according to relation (3.23). This process is usually referred to as *reconstruction* [Lagrava et al., 2012]. In analogy to the decimation, experiments have been made to enrich the information by deconvolution before passing

[4] The fine interface node could actually have received a distribution from the right-hand node. This, however, is omitted because the node cannot be updated via streaming, anyway.

it from the coarse to the fine domain. These attempts, however, remained without success [Lagrava et al., 2012], so that the *reconstruction* only describes the scaling of the populations.

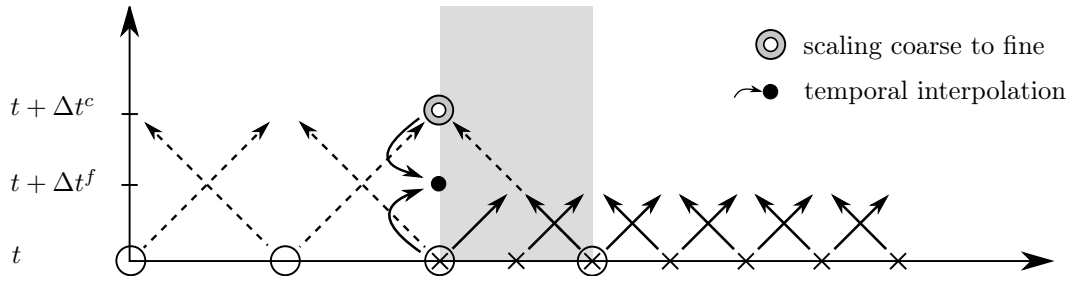


Figure 3.9: Step 3 – Reconstruction of the fine distribuiton functions (scaling coarse to fine) at time $t + \Delta t^c$ followed by a temporal interpolation.

Once all nodes of the refined domain are updated to time $\Delta t + t^f$, the fine distribution functions are collided and streamed one more time to reach the time level $t + \Delta t^c$ (Fig. 3.10). This constitutes *step* 4 of the algorithm.

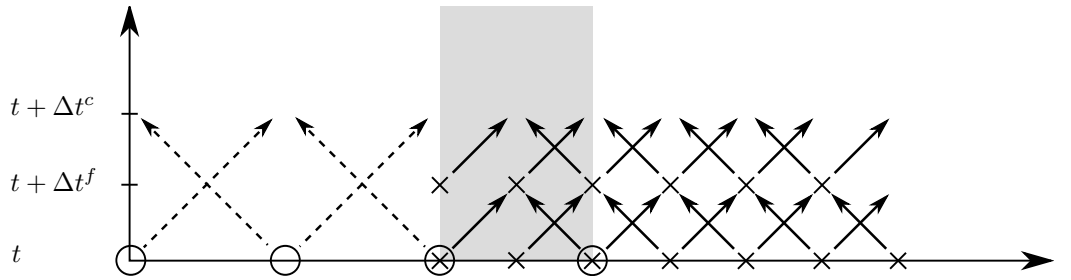


Figure 3.10: Step 4 – Collide and steam of the fine populations.

With the fine distribution functions updated to time $t + \Delta t^c$ it is now possible to proceed to *step* 5: the creation of an additional coarse node via *decimation* (Fig. 3.11). This closes the circle. The *reconstruction* of the interface nodes (scaling *coarse to fine*) is here only indicated by dashed circle lines because it was already done in *step* 3 of the algorithm (cf. Fig. 3.9).

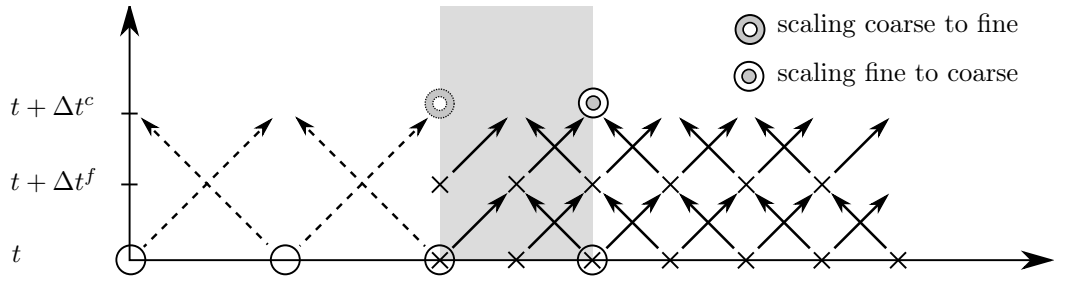


Figure 3.11: Step 5 – Decimation (scaling fine to coarse plus filtering) and end of cycle.

Cell-based approach: The cell-based approach pursues a very different strategy that arises from the setting in Fig. 3.12. The definition of co-located and hanging nodes does not apply. Moreover, the invalid lattice vectors (dashed arrows) pointing to the coarse and the fine transition nodes, respectively, all originate from unpopulated space.

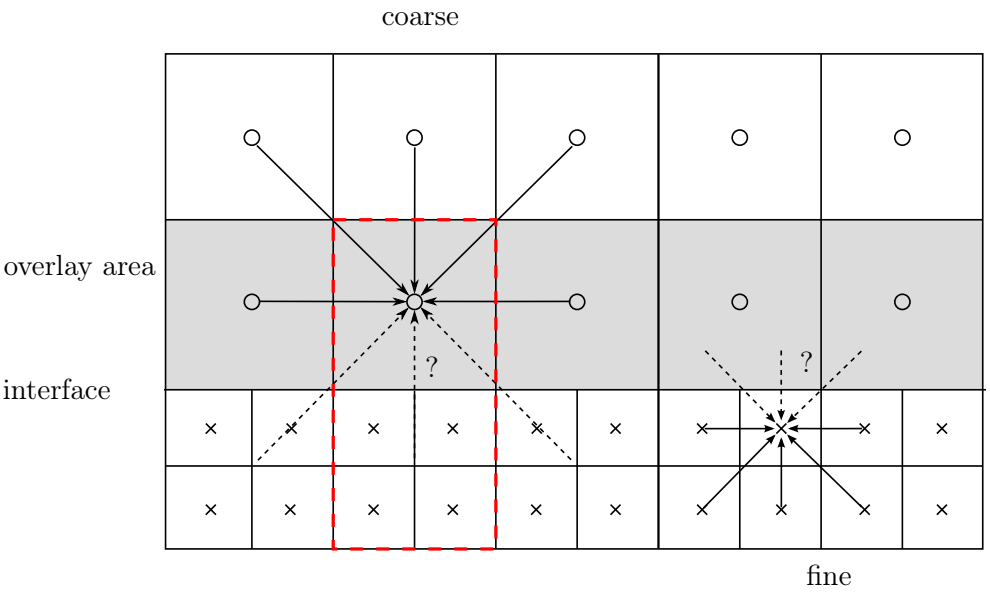
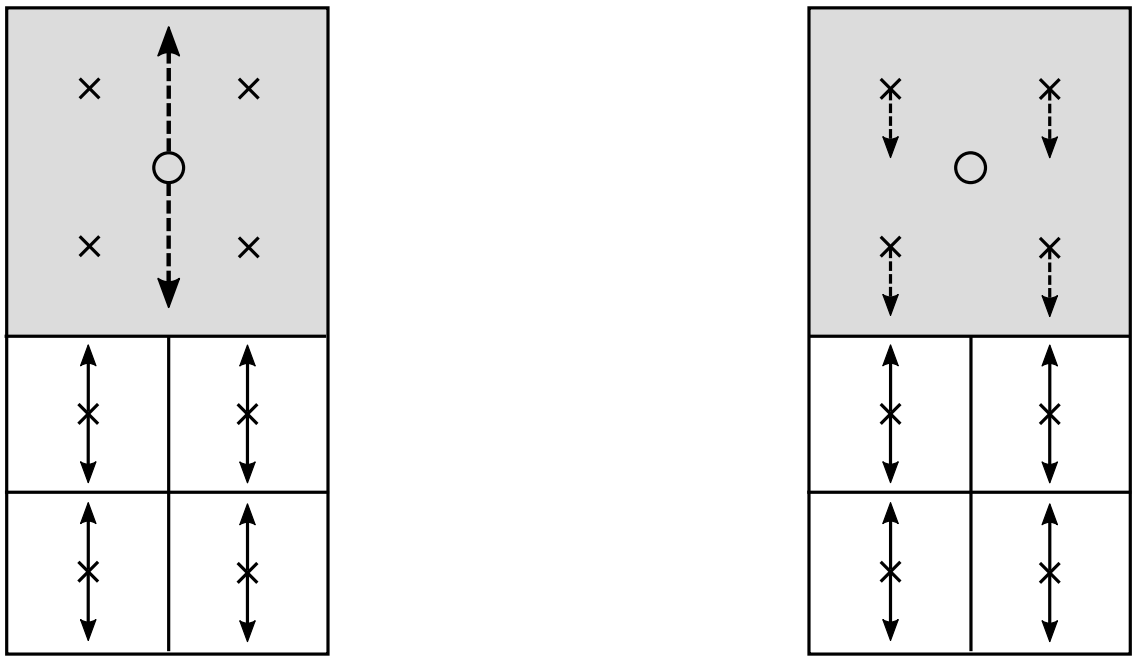


Figure 3.12: Initial state: all distribution functions are in a post-collision state at time t and ready for the successive streaming to the neighbour nodes. Missing incoming functions are indicated by dashed arrows.

In order to walk the reader through the algorithm, each step is accompanied by a simple two-dimensional sketch. The integration of a time axis into these diagrams is omitted for simplicity and the design is largely inspired by Rohde et al. [2006]. In particular, we zoom onto the cell that contains the central node of the coarse lattice in Fig. 3.12 and the four fine cells below (red dashed rectangle). Following symbolism applies to all subsequent diagrams (Figs. 3.13a - 3.15b) and is necessary to grasp the different steps of the algorithm: arrows have the character of a probability distribution. Pointing towards a node indicates a pre-collision state, while pointing away represents a post-collision state, ready for streaming. A dashed arrow shaft stands for an origin in the coarse domain, leaving the continuous shaft to indicate an origin in the fine domain.

In the previous approach, information was, at first, transferred from the fine grid to the coarse grid, while the transfer in the opposite sense occurred at the next coarse time level. In this approach, it is the other way around. A first step is the creation of an overlay area of (empty) fine nodes in the coarse domain (Fig. 3.13a).

A subsequent *explosion* distributes the post-collision populations of the underlying coarse cell to the centres of the fine cells (Fig. 3.13b). In particular, only those popula-

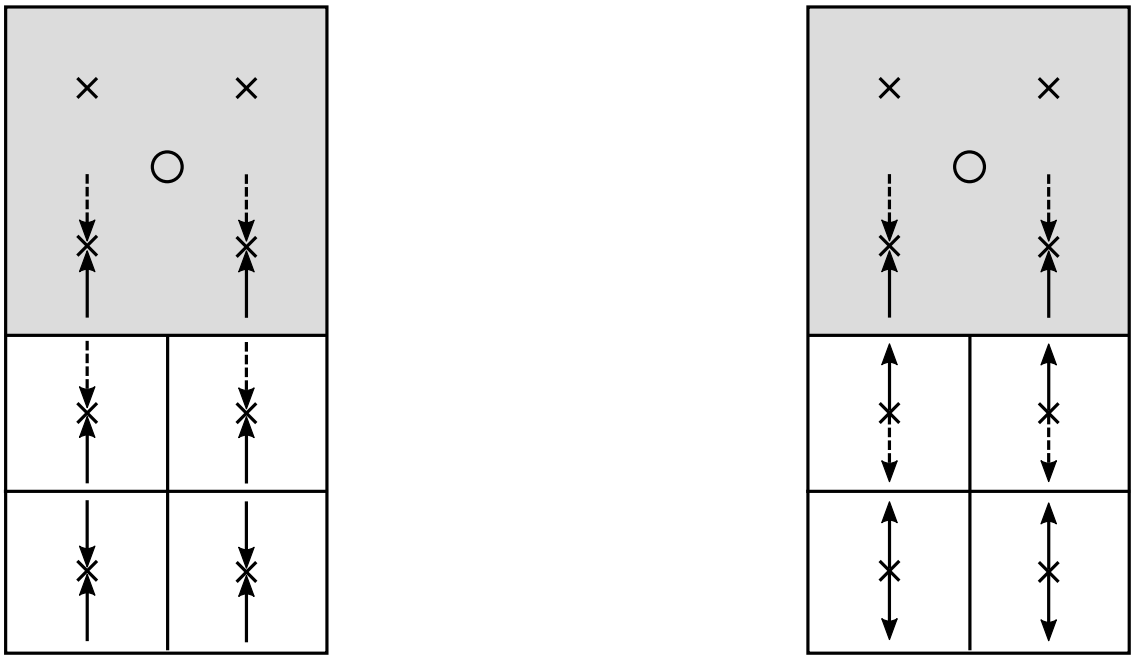


(a) Creation of overlay area **(b)** Explosion

Figure 3.13: Steps one and two of the algorithm that are accomplished at time level t.

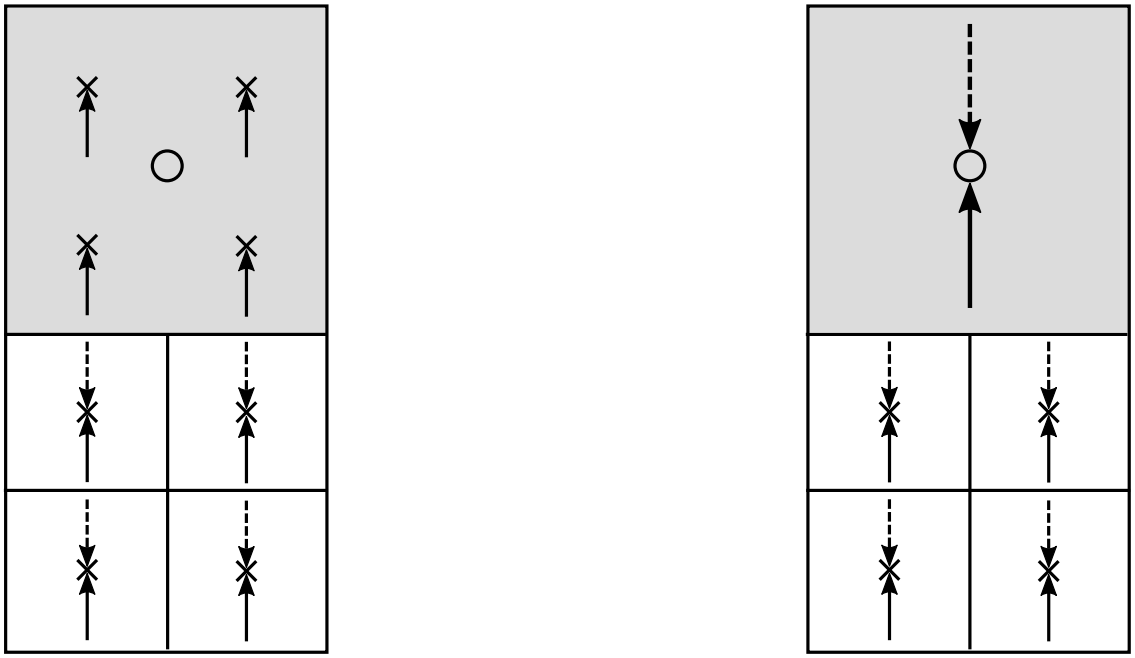
tions are scatters that are oriented towards the fine grid. This explosion demands the scaling *coarse to fine* of the populations according to Eq. (3.23). After the explosion, all nodes in the fine domain can be updated to time level $t + \Delta t^f$ via a streaming step (Fig. 3.14a), thanks to the fine distributions of the overlay area (cf. grey area in Fig. 3.13b). It should be noted that also the bottom row of the overlay area is updated. Now, the populations can be relaxed, indicated by a change in orientation of the distributions in Fig. 3.14b. A peculiarity of this collision is that only the nodes in the fine domain are relaxed. The orientation of the arrows in the overlay area hence does not change between Figs. 3.14a and 3.14b.

Finally, a second streaming step is applied to the fine nodes, including the overlay area, which casts the totality of fine nodes into a pre-collision state at time $t + \Delta t^c$ (Fig. 3.15a). The last step of the algorithm contains the *coalescence* of the distribution functions in the fine nodes of the overlay. The fine pre-collision distributions are scaled *fine to coarse*, according to Eq. (3.23), and gathered in the central coarse node using an arithmetic average. During the streaming step of the coarse domain, which can be executed arbitrarily between the previously described steps of the algorithm, the coarse node has already received a population from the upper neighbour (exceeding the domain depicted in the sketches). The set of distribution functions is thus complete and a global collision brings us back to the point of departure (Fig. 3.13a).



(a) Streaming in fine domain **(b)** Collision in fine domain

Figure 3.14: Algorithm steps at time level $t + \Delta t^f$.



(a) streaming in fine nodes **(b)** coalescence

Figure 3.15: Algorithm steps at time level $t + \Delta t^c$.

3.3.1.3 Concluding Remarks

The above presentation outlines the main strategies that are pursued in the *vertex-based* and the *cell-based* algorithms. In detail, the individual steps of each algorithm may, nevertheless, differ. Particularly, the spatial and temporal interpolation leaves some freedom for variability. In order to maintain the overall second-order accuracy of the method, the lowest interpolation schemes are linear in space [Marié, 2008, Rohde et al., 2006] and time [Lagrava et al., 2012, Crouse, 2003] [5]. In most of the studies, however, a polynomial approximation is chosen; usually fourth-order cubic for space and third-order quadratic for time approximation [Gendre et al., 2017].

The variables that are transferred between domains in the above algorithms are naturally the distribution functions g_α. An alternative, however, is to copy the moments of g_α. The advantage is that the first two moments, ρ and $\rho\boldsymbol{u}$, are continuous over the interface. The momentum flux tensor

$$\Pi_{ij} = \underbrace{\sum_\alpha \xi_{\alpha i}\xi_{\alpha j} g_\alpha^{eq}}_{\Pi_{ij}^{eq}} + \underbrace{\sum_\alpha \xi_{\alpha i}\xi_{\alpha j} g_\alpha^{neq}}_{\Pi_{ij}^{neq}}, \tag{3.25}$$

however, requires scaling in order to maintain a second-order accuracy of the stress components that are contained in the off-equilibrium part Π_{ij}^{neq}. It should be noted that in the previous chapter, the momentum flux tensor was denoted as $\boldsymbol{P}$. This notation is usually reserved for the second-order moment of the genuine distribution functions f_α, which is continuous over a refinement interface. The second-order moment of g_α, on the other hand, is not continuous based on relation (3.22) and therefore explicitly marked as Greek letter. According to Eq. (3.23)

$$\boldsymbol{\Pi}^{neq,c} = \underset{f\to c}{\zeta}\, \boldsymbol{\Pi}^{neq,f} \quad \text{or rather} \quad \boldsymbol{\Pi}^{neq,f} = \underset{c\to f}{\zeta}\, \boldsymbol{\Pi}^{neq,c}. \tag{3.26}$$

A problem emerges when returning to the mesoscopic level. We recall that the distribution functions may be expanded in terms of the small parameter ϵ

$$g_\alpha = g_\alpha^{(0)} + \epsilon g_\alpha^{(1)} + \mathcal{O}(\epsilon^2).$$

While $g^{(0)}$ can be calculated from the first two moments ρ and $\rho\boldsymbol{u}$ via (2.21), it is *a priori* not possible to obtain $g_\alpha^{(1)}$ from $\boldsymbol{\Pi}^{(1)}$, which would be required to determine the populations from the hydrodynamic variables in a unique fashion. This can be explained with the disparity in the number of distribution functions and hydrodynamic variables. In two dimensions, six independent variables, ρ, u_x, u_y, Π_{xx}, Π_{xy} and Π_{yy}, are necessary to recover the Navier-Stokes equations, given that the momentum flux

[5] A linear scheme is sufficient due to interpolation between equidistant points.

tensor is symmetric: $\Pi_{xy} = \Pi_{yx}$. The two-dimensional, isotropic lattice that fills space with simple squares (D2Q9), on the other hand, includes nine distribution functions. The three extra variables are associated with non-hydrodynamic, so-called "ghost" modes that are of no significance to the Navier-Stokes hydrodynamics. For a thorough discussion on this subject, the interested reader may refer to Dellar [2002, 2003]. Based on these considerations, we may assume that the definition of $g_\alpha^{(1)}$ contains relevant (hydrodynamic) and irrelevant (non-hydrodynamic) contributions with respect to the governing macroscopic equations [Latt, 2007, pp. 45-46] [6]. It was therefore suggested by Latt and Chopard [2005], to substitute $g_\alpha^{(1)}$ for a regularized expression $\overline{g}_\alpha^{(1)}$ that solely depends on $\mathbf{\Pi}$. In particular

$$\epsilon \overline{g}_\alpha^{(1)} \approx -w_\alpha \frac{1}{2c_0^4} \mathscr{H}_{\alpha ij}^{(2)} \epsilon \Pi_{ij}^{(1)}, \tag{3.27}$$

with $\mathscr{H}_{ij}^{(2)} = \xi_i \xi_j - \delta_{ij}$ being the second-order Hermite polynomial (cf. appendix C). In chapter 2, it was shown that under the low Mach number assumption, $\epsilon \Pi_{ij}^{(1)}$ can be related to the strain rate tensor $S_{ij} = 0.5(\partial u_i/\partial x_j + \partial u_j/\partial x_i)$ so that one equally obtains

$$\epsilon \overline{g}_\alpha^{(1)} \approx -w_\alpha \frac{\rho \tau_g}{c_0^2} \mathscr{H}_{\alpha ij}^{(2)} S_{ij}.$$

The regularized populations then read

$$g_\alpha \approx g_\alpha^{(0)} + \epsilon \overline{g}_\alpha^{(1)}, \quad \text{or rather} \quad g_\alpha \approx g_\alpha^{eq} + \overline{g}_\alpha^{neq}, \tag{3.28}$$

and are uniquely defined from the hydrodynamic variables of the Navier-Stokes equations. With respect to the mesh refinement algorithm, following differences apply when regularization is used. Instead of transferring the distribution functions, the moments ρ, $\rho \boldsymbol{u}$ and $\mathbf{\Pi}$ are constructed according to Eqs. (2.22), (2.23) and (3.25). The non-equilibrium part $\mathbf{\Pi}^{neq}$ of the momentum flux tensor is then scaled with respect to the direction of transfer (Eq. (3.26)). Finally, $g_\alpha^{(1)}$ is calculated using Eq. (3.27) and g_α is reconstructed with Eq. (3.28). Instead of simply redefining the distribution functions, the regularized non-equilibrium distribution is often injected during the BGK collision [Latt and Chopard, 2005] such that

$$\widehat{g}_\alpha = g_\alpha^{eq} + (1 - \omega_g) \overline{g}_\alpha^{neq}. \tag{3.29}$$

[6] An expression for $g_\alpha^{(1)}$ can either be obtained by expanding the LHS of the LBE (3.10) in a first-order Taylor series and applying a first-order Chapman-Enskog procedure; i.e. $g_\alpha^{(1)} = \tau_g \left(\frac{\partial}{\partial t_1} + \boldsymbol{\xi}_\alpha \cdot \nabla_1 \right) g_\alpha^{(0)}$ or from the polynomial expression (2.15).

3.3.1.4 Node-based versus cell-based

A final paragraph is dedicated to the differences between the vertex- and cell-based approach. A general difference lies in the number of computation nodes, which becomes apparent from Fig. 3.4. In order to cover the same two-dimensional square domain L^2, the cell-based approach requires $2N-1$ data points less, with $N = L/\Delta x$. It is thus, by nature, sightly more efficient than the node-based approach [Fakhari and Lee, 2015]. This difference, however, becomes negligible in computations with a large number of degrees of freedom. The presence of co-located nodes in the node-based approach is a double-edged sword. It introduces a "strong" coupling in the sense that the solution is mutually imposed in the respective other node, despite the fact that the macroscopic solution on the fine and coarse grid are slightly different. Due to the second-order accuracy of the lattice Boltzmann method, the error with respect to an analytical solution may be quantified as

$$E(\Delta x, \Delta t) \approx C_x \Delta x^2 + C_t \Delta t^2,$$

where C_x and C_t denote some constants that are independent of Δx and Δt. The difference between a solution on a coarse (Δx^c) and a refined grid ($\Delta x^f = 0.5\Delta x^c$) is then estimated as

$$\begin{aligned}\Delta\phi &= E(\Delta x^c, \Delta t^c) - E(\Delta x^f, \Delta t^f) \\ &= 3C_x \left(\frac{\Delta x^c}{2}\right)^2 + 3C_t \left(\frac{\Delta t^c}{2}\right)^2 \\ &= \frac{3}{4} E(\Delta x^c, \Delta t^c),\end{aligned}$$

where ϕ represents any macroscopic variable that is transferred between co-located nodes. In order to illustrate this difference, a two-dimensional pulse is enforce at the center $(x, y) = (0, 0)$ of a quadratic domain of dimension 1 m $\times$ 1 m.

$$\begin{cases} u(x, y, t_0) = 0.0 \\ v(x, y, t_0) = 0.0 \\ p(x, y, t_0) = p_0 + 1000 \times \exp\left(-\dfrac{x^2 + y^2}{r^2}\right), \end{cases} \tag{3.30}$$

with $p_0 = 117786$ Pa and $r = 0.05$ m. Each direction is discretized with $N = 51$ points in order to obtain a coarse grid with $\Delta x^c = 0.02$ m. Accordingly, the fine grid contains $N = 101$ discrete points in each direction with $\Delta x^f = 0.01$ m. Fig 3.16 compares the two solutions after a physical time of $T = 6.73 \times 10^4$ s, which corresponds to 10

iterations on the coarse grid and 20 iterations on the fine grid, respectively. After a few iterations, the two solutions already differ by 2% of the initial pressure perturbation.

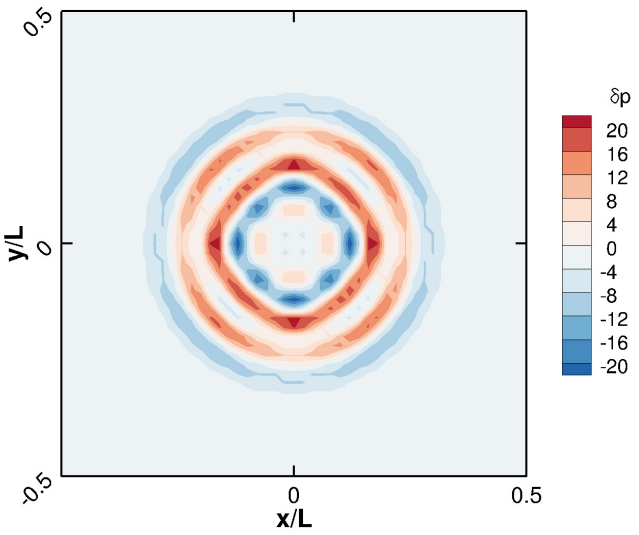


Figure 3.16: Difference in the solution of the pressure for two different grids.

The two mesoscopic solutions in the co-located nodes might be even more divergent than this 2%-error, which has to do with the previously mentioned "ghost" modes. The populations can be considered overdetermined in the sense that an infinite number of lattice molecules (the total set of distribution functions contained in one lattice node) exist that lead to the same macroscopic variables ρ, $\boldsymbol{u}$ and $\boldsymbol{\Pi}$. This is illustrated in Fig. 3.17. Suppose the macroscopic solutions are identical in the co-located nodes ($\otimes$). Then, the mesoscopic solution contained in each node may still be different because there is no constraint, which dictates that the fine and the coarse functions are identical. In the standard mesh refinement algorithm, the distributions at the interface are transferred from the coarse to the fine nodes. From the transferred distributions, only those are retained that have a neighbour in their assigned directions (red distributions in the left molecule of Fig. 3.17). The others are discarded (black distributions in the left molecule of Fig. 3.17). Now, imagine there was no interface and the fine co-located node has, instead, received the distributions from its fine, neighbouring node (right molecule in Fig. 3.17). If the sum of the five functions that are retained in case of a mesh transition is not the same, as the sum of the corresponding five functions obtained from the fine neighbouring nodes (red arrows in the right molecule of Fig. 3.17), then the conservation of mass is violated. Since no constraint exists that enforces the equality of the sums of a random subgroup of functions, the conservation of mass in this algorithm can not be guaranteed.

It is for this reason that the node-based approach is not mass-conservative on non-uniform grids. Regularization enforces the same shape of the fine and the coarse

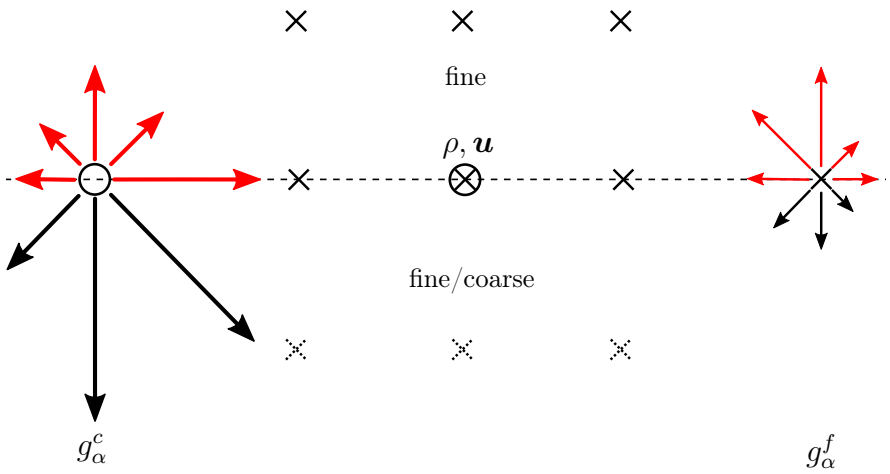


Figure 3.17: Identical macroscopic solution (ρ and $\boldsymbol{u}$) in co-located nodes ($\otimes$). Left: Coarse, mesoscopic solution g_α^c that is imposed in a fine interface node ($\bigcirc \rightarrow \times$) in case of mesh refinement. Right: (Theoretical) fine, mesoscopic solution g_α^f in the same fine node, if there was no transition and the distributions were obtained from the surrounding neighbours (dashed and solid crosses).

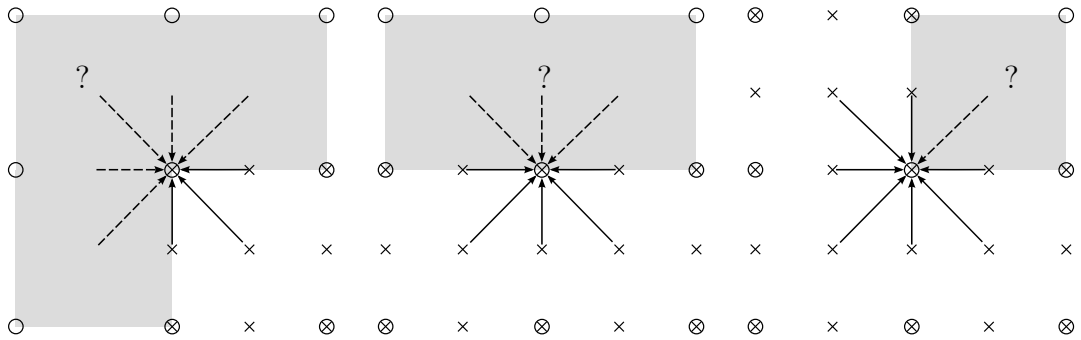
molecule, when they are reconstructed from identical macroscopic variables. Under these consideration, using regularization reduces the error at the interface to the error between the macroscopic variables.

In the cell-based approach, cell boundaries are co-located, introducing a "weak" coupling. Due to the balance of the fluxes, the cell-based approach is intrinsically mass-conservative. The advantage of the node co-localization, on the other hand, is revealed during spatial interpolation of the fine nodes. Already in the co-located nodes, no approximation is required. Information in the hanging nodes may then obtained by linearly interpolating through the equidistant co-located nodes, which is second-order accurate. In the cell-based D2Q9 model, two-dimensional barycentric interpolation is mandatory. More importantly, linear interpolation in this case is only first-order accurate, so that quadratic schemes should be used [Fakhari and Lee, 2015] to maintain the overall second-order accuracy of the method.

3.3.2 Partial reconstruction: An alternative to regularization?

Regularization is one possibility to compute the distribution functions from the hydrodynamic variables of the isothermal Navier-Stokes equations in a unique fashion. The non-equilibrium part g^{neq} is forced into a symmetric corset, while the asymmetric contributions are ignored. An alternative is provided by the generalized LBM, which will be presented in § 4.2. Here, a third possibility in 2D is examined, which constitutes a first original work of this PhD thesis.

A 2D lattice can not be reduced to six discrete velocities to match the number of hydrodynamic variables. With regard to mesh refinement on a D2Q9 lattice, it should, however, be noted that in an interface node, the number of unknown functions is always smaller than six. In particular, there is a maximum of five unknown functions in a reflex corner interface node (Fig. 3.18a). If the interface is a straight line or a right angle (with respect to the unknown functions), the number of unknown functions reduces to three and one, respectively (Fig. 3.18b and 3.18c).



(a) reflex corner **(b)** straight interface **(c)** right corner

Figure 3.18: Three types of interface nodes in the D2Q9 model. Shaded area symbolizes the set of unknown functions.

It should thus be possible to derive a system of equations that allows to reconstruct the missing functions from the hydrodynamic variables together with the populations that are known. For the sake of simplicity, we will restrict the following consideration to a node on a straight interface that lacks neighbours in positive y-direction, as it is the case in Fig 3.18b. In accordance with our definition in § 2.3, the missing functions are therefore g_4, g_7 and g_8. Moreover, the idea of partial reconstruction will be exemplarily presented for a co-located fine interface node ($\times$). Identical to the reconstruction via regularization, we assume that the macroscopic variables ρ, $\boldsymbol{u}$ and $\boldsymbol{\Pi}^f$ in the co-located fine interface node are always available at the next small time step. This reasoning is based on the fact that the co-located coarse interface node in Fig. 3.18b possesses the complete set of neighbours and may therefore provide information at time t and $t + \Delta t^c$. In addition, temporal interpolation yields the solution at the intermediate time level $t + \Delta t^f$. Remember that the solution of the stress tensor is discontinuous. In the following, $\boldsymbol{\Pi}$ represents $\boldsymbol{\Pi}^f = \boldsymbol{\Pi}^{eq} + \left(\tau_g^f/\tau_g^c\right) \boldsymbol{\Pi}^{neq,c}$. Then, we obtain the following system of equations, where k stands for the populations that are known at the next fine time level. Note that the velocity moments are given in lattice units, which is not indicated explicitly. An asterisk indicates a macroscopic solution that was transferred

from the coarse grid

$$\begin{cases} g_4 + g_7 + g_8 = \rho^* - \sum g_k & (3.31a) \\ g_4\xi_{x,4} + g_7\xi_{x,7} + g_8\xi_{x,8} = \rho u_x^* - \sum g_k\xi_{x,k} & (3.31b) \\ g_4\xi_{y,4} + g_7\xi_{y,7} + g_8\xi_{y,8} = \rho u_y^* - \sum g_k\xi_{y,k} & (3.31c) \\ g_4\xi_{x,4}\xi_{x,4} + g_7\xi_{x,7}\xi_{x,7} + g_8\xi_{x,8}\xi_{x,8} = \Pi_{xx}^* - \sum g_k\xi_{x,k}\xi_{x,k} & (3.31d) \\ g_4\xi_{x,4}\xi_{y,4} + g_7\xi_{x,7}\xi_{y,7} + g_8\xi_{x,8}\xi_{y,8} = \Pi_{xy}^* - \sum g_k\xi_{x,k}\xi_{y,k} & (3.31e) \\ g_4\xi_{y,4}\xi_{y,4} + g_7\xi_{y,7}\xi_{y,7} + g_8\xi_{y,8}\xi_{y,8} = \Pi_{yy}^* - \sum g_k\xi_{y,k}\xi_{y,k} & (3.31f) \end{cases}$$

Substituting for the lattice velocities with $c = 1$, the system simplifies to

$$\begin{cases} g_4 + g_7 + g_8 = \rho^* - \sum g_k & (3.32a) \\ g_7 - g_8 = \rho u_x^* - \sum g_k\xi_{x,k} & (3.32b) \\ -g_4 - g_7 - g_8 = \rho u_y^* - \sum g_k\xi_{y,k} & (3.32c) \\ g_7 + g_8 = \Pi_{xx}^* - \sum g_k\xi_{x,k}\xi_{x,k} & (3.32d) \\ -g_7 + g_8 = \Pi_{xy}^* - \sum g_k\xi_{x,k}\xi_{y,k} & (3.32e) \\ g_4 + g_7 + g_8 = \Pi_{yy}^* - \sum g_k\xi_{y,k}\xi_{y,k}. & (3.32f) \end{cases}$$

We see that the system is dependent through Eq. (3.32a), Eq. (3.32c) and Eq. (3.32f) on the one hand, and Eq. (3.32b) and Eq. (3.32e) on the other. Imposing that the following assumptions hold

$$\rho - \sum g_k \equiv \Pi_{yy} - \sum g_k\xi_{y,k}\xi_{y,k} \equiv -\rho u_y + \sum g_k\xi_{y,k},$$

and

$$\rho u_x - \sum g_k\xi_{x,k} \equiv -\left(\Pi_{xy} - \sum g_k\xi_{x,k}\xi_{y,k}\right),$$

the system can be reduced to a system of only three equations

$$\begin{cases} g_4 + g_7 + g_8 = \rho^* - \sum g_k & (3.33a) \\ g_7 - g_8 = \rho u_x^* - \sum g_k\xi_{x,k} & (3.33b) \\ g_7 + g_8 = \Pi_{xx}^* - \sum g_k\xi_{x,k}\xi_{x,k}, & (3.33c) \end{cases}$$

so that g_4, g_7 and g_8 can be determined. From Eqs. 3.33b and 3.33c we obtain

$$g_7 = 0.5\left(\rho u_x^* - \sum g_k \xi_{x,k} + \Pi_{xx}^* - \sum g_k \xi_{x,k}\xi_{x,k}\right),$$

and

$$g_8 = -0.5\left(\rho u_x^* - \sum g_k \xi_{x,k} - \Pi_{xx}^* + \sum g_k \xi_{x,k}\xi_{x,k}\right).$$

Using the above information together with equation 3.33a, one finally obtains

$$g_4 = \rho^* - \sum g_k - \Pi_{xx}^* + \sum g_k \xi_{x,k}\xi_{x,k}.$$

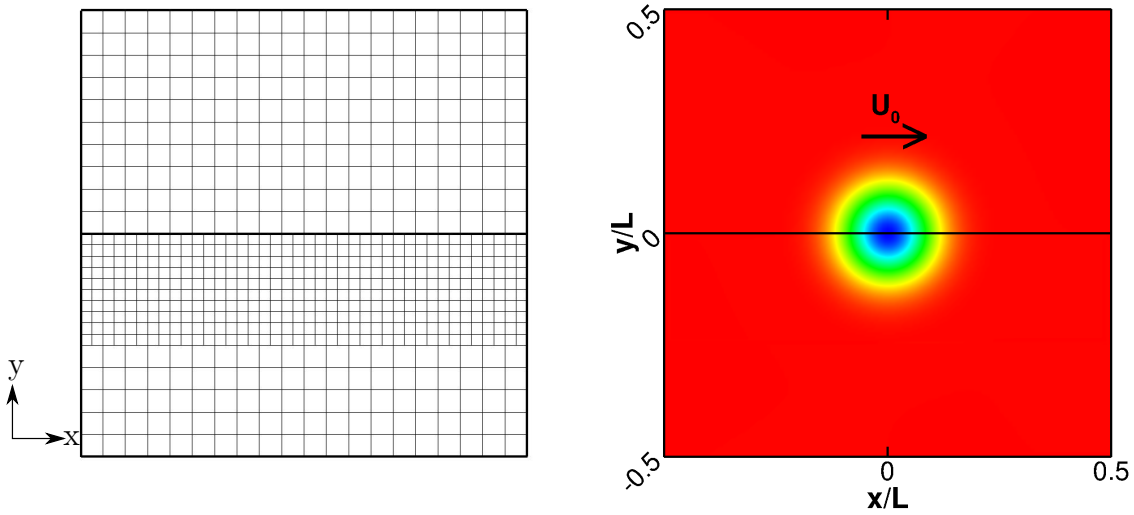
The algorithm has been tested in a multi-grid approach, which bears the advantage of an isolated consideration of the partial reconstruction step. The computational domain, depicted in Fig. 3.19a, is composed of a quadratic root grid of size 1 m^2 and a refined patch of dimensions 1 m $\times$ 0.25 m, whose upper boundary coincides with the horizontal centreline of the root domain. The spatial resolution of the actual mesh is five times higher, than shown in Fig. 3.19a, with $\Delta x^c = 0.01$ m and $\Delta x^f = 0.005$ m. This relates to a physical time step of $\Delta t^c = 1.68 \times 10^{-5}$ s and $\Delta t^f = 0.84 \times 10^{-5}$ s, respectively. As a test case, a 2D *pseudo-isentropic* vortex [Gendre et al., 2017] is initialized in the center of the domain (Fig. 3.19b) and advected in positive x-direction.

$$\begin{cases} u(x,y,t_0) = U_0 - \kappa c_0 \dfrac{y}{R} exp\left(\dfrac{1}{2} - \dfrac{r^2}{2R^2}\right) \\ v(x,y,t_0) = \quad\quad \kappa c_0 \dfrac{x}{R} exp\left(\dfrac{1}{2} - \dfrac{r^2}{2R^2}\right) \\ \rho(x,y,t_0) = \rho_0 - \dfrac{\rho_0 \kappa^2}{2} exp\left(1 - \dfrac{r^2}{R^2}\right), \end{cases} \tag{3.34}$$

where the parameter κ controls the vortex strength, c_0 is the speed of sound, ρ_0 is the density and r/R the normalized radius with $r = \sqrt{x^2 + y^2}$. For this simulation, we choose

$$R = 0.1 \text{ m}, \quad \kappa = 0.14, \quad c_0 = 343.2 \text{ m/s}, \quad U_0 = 0.2c_0, \quad \rho_0 = 1 \text{ kg/m}^3,$$

leading to a maximum velocity of $u_{max} = 0.34c_0$. The boundary conditions are periodic in both directions.



(a) Sketch of the computaitonal mesh. A single grid cell reprents $5\Delta x^2$ cells in the actual simulation.

(b) Initialization of the vortex at time t_0. Isocontours show the density fluctuations $\rho^{'} = \rho_0 - \rho(\boldsymbol{x}, t)$ in the range -0.02 kg/m$^3 < \rho^{'} < 0$ kg/m^3.

Figure 3.19: Computational domain and initial state of the advection of a *pseudo-isentropic* vortex.

In order to evaluate the algorithm, results are compared to the solution of a reference code that uses regularized reconstruction in a multi-grid approach. The main difference in the two algorithms is thus the reconstruction of the distribution functions in the fine interface nodes. Snapshots of the solution are recorded at $t^* = 870$, $t^* = 1740$ and $t^* = 2610$, where $t^* = t/\Delta t$ denotes the ratio of total physical time t over the time step. With the distance defined as $d = t^* \Delta t U_0$, this corresponds approximately to three subsequent passages of the center point of the domain. Results of the reference algorithm with regularized reconstruction are depicted in the first row of Fig 3.20. We see that over the course of three domain passages, the fine solution remains stable. Apart from a slight distortion near the interface, it represents the mirror image of the coarse solution. The row underneath in Fig 3.20 contains the results from the partial reconstruction algorithm. The solution at $t^* = 870$ appears perfectly symmetric with respect to the interface. One passage further, however, we observe the distortion of the vortex core on the fine domain. Somewhere between the second and the third passage, the solution becomes unstable, so that at $t^* = 2610$ the vortex has disappeared. In a last experiment, we have switched the resolutions. In this case, the missing populations in the boundary nodes of a coarse patch are reconstructed from the hydrodynamic variables of an underlying fine root grid. This algorithm was not discussed previously. Results are presented in the bottom row of Fig. 3.20. Already after a single domain passage, small zick-zack structures become visible in the contour lines. The vortex

becomes unstable before its second passage.

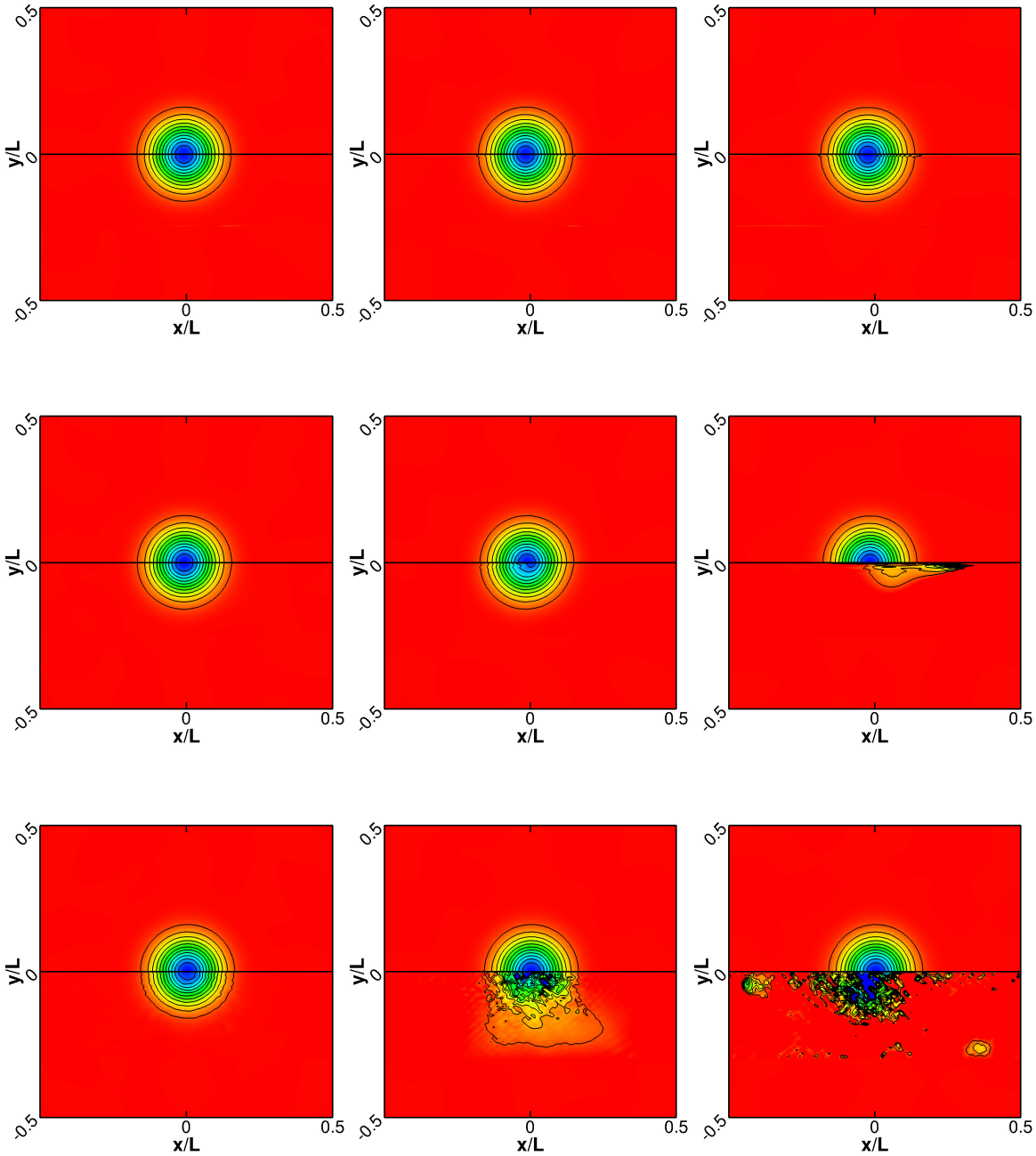


Figure 3.20: Snapshots of the pseudo-isentropic vortex computed with the regularized reconstruction (top row), partial reconstruction (middle row) and reversed partial reconstruction (bottom row). Isocontours show the density fluctuations $\rho^{'}(\boldsymbol{x}, t) = \rho_0 - \rho(\boldsymbol{x}, t)$ in the range $-0.02\ \mathrm{kg/m^3} < \rho^{'} < 0\ \mathrm{kg/m^3}$.

The fact that a partial reconstruction technique becomes unstable, agrees with our

previous findings that the solution on the fine and the coarse grid are not identical. The question now arises, why the regularization technique is the better choice. First of all, let $\delta\phi = \phi^* - \phi$ denote the difference between the two solutions, where ϕ represents the first three moments of g_α. The asterisk indicates the solution from the respective other grid, which is used to reconstruct the populations in the interface nodes. Applying regularization, the entire set of distribution functions is thus reconstructed with slightly erroneous hydrodynamic variables, in particular $f_\alpha^{eq}(\rho+\delta\rho, \boldsymbol{u}+\delta u)$ and $\overline{f}^{neq}(\boldsymbol{\Pi}+\delta\boldsymbol{\Pi})$. Only a fraction of the error is thus allocated to each distribution function. In our example of partial reconstruction, there are six populations that are exact (with respect to the actual grid), two populations (g_7, g_8) that were reconstructed as a functions of $\boldsymbol{u}+\delta u$ and $\boldsymbol{\Pi}+\delta\boldsymbol{\Pi}$, and finally, g_4, which was calculated with $\rho+\delta\rho$ and $\boldsymbol{\Pi}+\delta\boldsymbol{\Pi}$. There is thus only a single distribution function that absorbs the difference in the density $\delta\rho$ between the two grids. We may thus write

$$g_4^* = g_4 + \delta\rho.$$

Let us now assume that after the reconstruction and a collision step, the g_4^* function streams into the fine domain (cf. Fig 3.18b). At the new destination, g_4^* will constitute the only reconstructed function in the entire set of populations g_α. The macroscopic variables at this position are thus

$$\begin{aligned}
\rho^* &= \rho + \delta\rho \\
\rho\boldsymbol{u}^* &= \rho\boldsymbol{u} + \delta\rho\xi_4 \\
\boldsymbol{\Pi}^* &= \boldsymbol{\Pi} + \delta\rho\xi_4\xi_4,
\end{aligned}$$

with $\boldsymbol{\Pi} = \rho\left(u_i u_j + c_0^2\delta_{ij}\right) - 2\rho c_0^2\tau_g S_{ij}$. As a measure of comparability, we introduce the relative error ϵ so that

$$\begin{aligned}
\epsilon_\rho &= \frac{\delta\rho}{\rho} \\
\epsilon_{\rho u} &= \frac{\delta\rho\xi_4}{\rho\boldsymbol{u}} \simeq \epsilon_\rho \times \mathcal{O}\left(\frac{1}{\mathrm{M}}\right) \\
\epsilon_\Pi &= \frac{\delta\rho\xi_4\xi_4}{\boldsymbol{\Pi}} \simeq \epsilon_\rho \times \left[\mathcal{O}\left(\frac{1}{\mathrm{M}^2}\right) + \mathcal{O}\left(1\right) + \mathcal{O}\left(\frac{1}{\mathrm{M}}\frac{L}{\Delta x}\right)\right],
\end{aligned}$$

where $\mathrm{M} = U/c_0$ denotes the Mach number. The above analysis yields some interesting insights that help to explain the results obtained with the partial reconstruction (cf. Fig. 3.20). A general observation is that the error introduced by a variation in mass between the two grids is amplified with a factor $\mathcal{O}\left(1/\mathrm{M}\right)$ and $\mathcal{O}\left(1/\mathrm{M}^2\right)$ for the computation of the first-order and second-order moment, respectively. The lower the Mach number, the more this error therefore amplifies in the higher moments. It thus stands

in opposition to the error introduced by the truncation of the equilibrium function, which becomes non-negligible at increasing Mach numbers. The same reasoning can actually be applied to the regularization. The important difference is the fact that here the entire set of density functions is reconstructed. Already about half of the error $\delta\rho$ is therefore attributed to g_0 (weighted with $w_0 = 4/9$ in the summation), which is not propagated. The respective regularized function $\overline{g}_4$ carries only a fraction of $\delta\rho$ into the fine domain that is approximately one order of magnitude lower compared to that of a partial reconstruction.

3.3.3 Conclusion and Perspective

The classical mesh refinement algorithm in the node-based approach relies on a redundant overlay region. At the edges of this region, which usually covers one coarse grid cell Δx^c, the solution is imposed in co-located nodes of the respective other grid (Fig. 3.21).

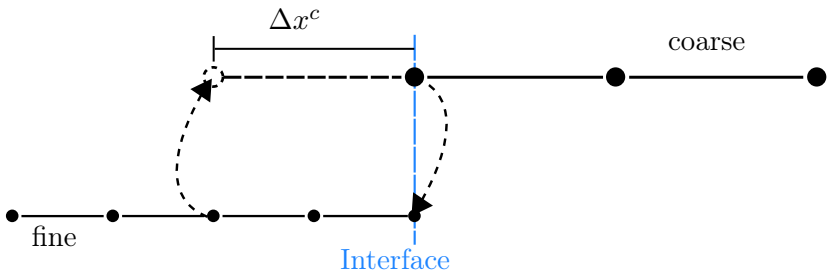


Figure 3.21: Sketch of the refinement algorithm with overlay.

Mesoscopic as well as macroscopic variables can be exchanged between the domains. Latter requires the reconstruction of the populations. This is usually achieved via regularization or generalized LBM (not dicussed here). Moreover, we have introduced the possibility of a partial reconstruction step, which actually deteriorates the mesh refinement algorithm. Based on our findings, we believe that the problem originates from the fact that the two solutions in the co-located nodes are slightly different. Using partial reconstruction, the error is concentrated in a few distribution functions. This would explain the fast blow-up compared to the algorithms using regularized reconstruction or the direct transfer of the populations. Based on these considerations, a possible road for improvement would be a direct matching of the domains, without overlay as depicted in Fig. 3.22. In this case, the solution in the interface node has the character of an intermediary between the two resolution domains, because neither the coarse nor the fine solution is directly imposed. The challenge is thus to find a solution that constitutes a valid boundary condition for both resolution domains. One possibility was presented in Lagrava [2012]. Here, the fine and coarse distributions in the interface node are used to construct a system of equation that solves for ρ and $\boldsymbol{u}$,

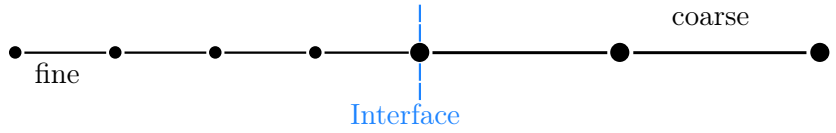


Figure 3.22: Sketch of the refinement algorithm without overlay.

which constitutes an intermediary macroscopic solution. The algorithm was validated against the analytical solution to a two-dimensional Poiseuille flow. The results are in very good agreement with the analytical solution and confirm the conservation of mass. For three-dimensional lattices, the system of equations can not be solved analytically and requires an iterative solver. Another possibility is to reconstruct the intermediary macroscopic solution in the interface nodes based on the conservation of mass and momentum. To our best knowledge, this idea has not been tested so far and involves the construction of a control volume around the interface node. In doing so, the macroscopic solutions in the interface nodes are updated based on the mass and momentum fluxes in and out of the cell. During the present PhD thesis, this particular idea has not been tested. It gave, however, crucial impulses to the decision to couple the classical LBM with a finite-volume formulation of the DVBE and the Navier-Stokes equations, respectively. This topic constitutes the main work of this thesis and will be presented in the next chapter.

3.4 Summary

In the previous chapter, the LBM was presented as a descendant of the LGA. Here, it was shown that the LBM can also be derived from the continuous BE, or rather the DVBE, which considers a discrete velocity-space (lattice), but is continuous in space and time. When velocity-space, physical space and time are coupled, which is the case in a Lagrangian discretization, one obtains the classical LBM with the well-known *stream and collide* kernel. Here, we highlight that it is equally possible to choose a discretization in the Eulerian sense. The spatial derivative in this case is treated explicitly by subdividing continuous space into a finite number of differences (nodes), volumes or elements that can be chosen independently from the velocity-space. This implies that structured as well as unstructured grids come into consideration, which can be arbitrarily refined or coarsened where necessary. In the Lagrangian approach, mesh refinement becomes a real issue, because it is confined to a quadtree (2D) or rather octree (3D) data structure imposed by the lattice. The abrupt transition in resolution is prone to trigger instabilities and cause the emission of spurious noise. A third part of this chapter therefore sheds some light on mesh refinement in the classical LBM. In particular, the node-based and the cell-based approach are presented. The chapter terminates with the presentation of a novel idea for a node-based refinement

algorithm. It is based on the fact, that certain density functions are known at the interface. Together with the macroscopic solution from the co-located nodes of the respective other domain, it is possible to construct a set of equations that solve for the missing populations in a unique manner. While the results showed that this idea rather deteriorates the stability of the solver, it helped to pinpoint the problem of the existing algorithm. Against this background a perspective for further development is given.

Chapter 4

HLBM - A new mesh refinement algorithm

While the multi-scale lattice Boltzmann scheme preserves the efficiency of the classical *stream and collide* algorithm, it holds certain drawbacks. Despite including domains with different spatial resolutions, the cell geometry remains that of the lattice crystal, which prevents the adaptation to curved boundaries. Moreover, the mesh refines isotropically, although many flow situations exist, where a refinement in one coordinate direction is sufficient (e.g. wall-bounded flows, elongated geometires, etc.). In addition, the tree-type mesh refinement generates an abrupt transition of the space-time discretization step, which is prone to contaminate aeroacoustic simulations and trigger instabilities. Such a sudden refinement usually does not apply to the underlying flow scales. A gradual mesh refinement over multiple grid cells is therefore more adapted to the flow. This brings us back to the discretization in the Eulerian sense (cf. § 3.2), where physical space is discretized independently from the lattice, enabling gradual mesh refinement and body-fitted grids. These benefits are overshadowed by a considerable increase in computational costs and numerical dissipation. This gave rise to the question, of whether it is possible to couple the two discretization schemes in order to combine their positive features [Shrestha, 2015, p. 135].

A first answer was given very recently by Di Ilio et al. [2017] with their pioneering work on a hybrid lattice Boltzmann method (HLBM). Using a domain-decomposition technique, a peripheral, uniform Cartesian grid was coupled to a central, unstructured domain enclosing a cylinder. Around the obstacle, the distribution functions are solved with the LB method in finite-volume formulation, while in the periphery the classical *stream and collide* algorithm is applied. The FV-LB equations are solved with a simple explicit time marching, thus degrading the global accuracy of the method and accepting a limitation on the time step imposed by the relaxation time τ.

Previously in the manuscript, two finite-volume formulations of the lattice Boltz-

mann equation were presented that are second-order accurate and only limited by the CFL stability condition. It should therefore be possible to construct a HLBM that has the same accuracy as the classical LBM. Investigating this hypothesis constitutes one of the principal work of this thesis and will be presented and discussed in the first part of this chapter.

So far, the term *hybrid* referred to the combination of two different discretization schemes for the discrete velocity Boltzmann equation, e.g. **LBM – FV-LBM**. In literature, *hybrid* may also refer to the coupling of the classical LBM with the Navier-Stokes equations (NSE) [Feiz, 2006, Yeshala, 2010], i.e. **LBM – NS**. Here and in the following, this approach will be referred to as HLBM_{NS}. The motivation behind such a coupling is the same: in isotropic flow regions, the Navier-Stokes solver is replaced by a lattice Boltzmann solver in order to improve the efficiency of the computation. In the second part of this chapter, we will show how the two hybrid schemes relate to each other. Moreover, a novel strategy is presented to enable the communication between the macroscopic solutions of the Navier-Stokes solver and the mesocopic variables of the LB method.

4.1 HLBM: LBM – FV-LBM

The present section is based around the author's co-written article **"Hybrid simulation combining two space-time discretization of the discrete-velocity Boltzmann equation"** that was published in the November 2017 issue of the *Journal of Computational Physics*. In a first instance, we are going to identify the second-order, finite-volume formulation that is best suited to be combined with the classical LBM. Then the coupling algorithm is defined, which relies on a domain decomposition technique (multi-domain). Finally, the developed HLBM is tested for two-dimensional, periodic case studies on a uniform and gradually refined mesh.

4.1.1 Performance of single schemes

In order to choose a candidate for the coupling, the **DUGKS** and the **Heun predictor-corrector** scheme that were introduced in chapter 3 are tested in terms of numerical stability and dissipation for varying CFL numbers and spatial flux interpolations. As a third criterion, the efficiency of each algorithm is taken into account. The CFL condition is controlled via the time step. Results are compared to the *stream and collide* algorithm and the analytical solution. As a test case, the 2D *pseudo-isentropic* vortex Eq. (3.34), with $U_0 = 0$, is enforced at initial time t_0 in a uniform square domain of size 1 m^2. For this simulation, we choose

$$R = 0.1 \text{ m}, \quad \kappa = 0.04, \quad c_0 = 343.2 \text{ m/s}, \quad \rho_0 = 1.0 \text{ kg/m}^3,$$

leading to a maximum velocity of $u_{max} = v_{max} = 0.04c_0$. The benchmark for the evaluation of the two schemes is a stable solution of the vortex after a physical time of about 0.1 s. The initial solution is discretized with $R = 5\Delta x^c$ and $R = 20\Delta x^f$, defining a coarse ($\Delta x^c = 0.02$ m) and a fine mesh ($\Delta x^f = 0.005$ m), respectively. The computational domain then contains $N = 51$ coarse, or rather $N = 201$ fine grid points in each direction. At a CFL of unity, the physical time steps amount to $\Delta t^c = 3.326 \times 10^{-5}$ s and $\Delta t^f = 0.831 \times 10^{-5}$ s for the respective discretization in space, requiring 3000 and 12000 iterations to simulate 0.1 s of the flow. Time step and number of iterations scale with 1/CFL depending on the stability of the numerical scheme.

The parametric study hence comprises the two different second-order temporal flux evaluation that were presented in the § 3.2. In order to mimic the characteristics of the *stream and collide* algorithm, a low dissipative scheme is required that still allows for a maximal time step. Each scheme, therefore starts with a CFL number of order unity that is successively decreased by a factor of two until stable results are obtained, both for a centred and quadratic upwind interpolation (QUICK). The computational costs are given as CPU/iteration and as total CPU per simulation relative to the total CPU of the *stream and collide* algorithm

$$\text{relative cost} = \frac{\text{CPU}}{\text{CPU}_{\text{LBM}}} \times \frac{1}{\text{CFL}}.$$

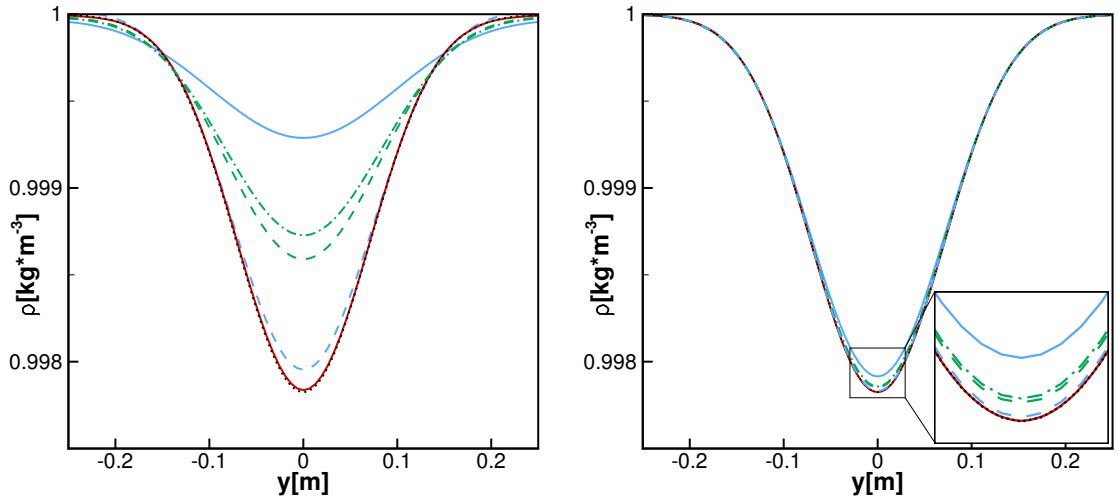
The resulting parametric settings are presented in table 4.1.

Table 4.1: Discretization schemes and parameters

scheme		LBM	FV-LBM			
			DUGKS		Heun predictor-corrector	
flux approximation		-	centred	QUICK	centred	QUICK
$N = 51$	CFL	1	0.5	0.5	0.25	1
	CPU/it	0.002 s	0.011 s	0.011 s	0.013 s	0.014 s
	relative cost	1	11.3	11.4	27.7	7.4
$N = 201$	CFL	1	0.5	0.5	0.5	1
	CPU/it	0.029 s	0.168 s	0.169 s	0.211 s	0.221 s
	relative cost	1	11.4	11.5	14.4	7.5

We observe that with one exception, the CFL condition that yields stable results on the fine grid, is also applicable to the coarse grid. In case of the Heun predictor-corrector scheme with a centred approximation in space, the time step has to be decreased by a factor of two in order to obtain a stable solution on the coarse grid. By looking at the computational time per iteration, the *stream and collide* algorithm emerges as by

far the most efficient method. With respect to the finite-volume kinetic schemes, a QUICK evaluation adds slightly to the costs and the DUGK scheme performs better per iteration than the Heun predictor-corrector scheme. However, in terms of relative costs, the CPU required to compute a physical time of about 0.1s is lower in case of the Heun predictor-corrector scheme with QUICK approximation in space than for the DUGK schemes, which require more iterations.



(a) $N = 51$ **(b)** $N = 201$

Figure 4.1: Density profiles of the static vortex at physical time 0.1 s for different flux approximations: (·····) analytical solution; (———) streaming; (———) Heun QUICK; (– – –) Heun centred; (– – –) DUGKS centred; (– · – · –) DUGKS QUICK.

The corresponding graphs that represent the density of the vortex at $t = 0.1$ s for the two different resolutions in space are shown in Fig. 4.1. Generally, the order by which the different schemes dissipate remains unchanged for the two grid resolutions. As expected, the *stream and collide* algorithm shows the lowest numerical dissipation and by the end of the simulation it has only marginally diverged from the analytical solution ($N = 51$), or rather still agrees with it ($N = 201$). A low dissipation rate can also be observed for the finite-volume formulation using the Heun predictor-corrector scheme, with a centred flux evaluation and a CFL number equal to 0.25 ($N = 51$), or rather 0.5 ($N = 201$). The latter time step yields a more dissipated solution when the DUGK scheme is applied, both for the QUICK and a centred scheme. With the QUICK approximation, the Heun predictor-corrector methods yields stable results at a CFL number of unity. However, the dissipation is the highest among all results.

The Heun predictor-corrector scheme with a centred approximation in space is visibly a good candidate in terms of dissipation, but comes along with a higher susceptibility to numerical instabilities. The DUGK schemes show good dissipative behaviour at reasonable cost. For the construction of a hybrid algorithm, we nevertheless prioritized

the scheme that shares the same CFL number as the *stream and collide* algorithm, with the lowest relative cost among the finite-volume kinetic schemes, despite its lower performance in terms of dissipation. The main motivation behind this choice is the fact that both algorithms calculate with the same physical time step, which greatly facilitates the coupling of the two schemes. Hence, the HLBM established here combines the traditional vertex-based LBM with a FV-LBM that uses a Heun predictor-corrector scheme for a second-order flux approximation in time and a third-order upwind approximation (QUICK) in space, with a global CFL number equal to one, excluding the need for any subiterations. The next section presents the hybrid algorithm for one temporal iteration, followed by results for three different test cases.

4.1.2 HLBM algorithm

In order to make the coupling more evident, one can separate the collision from the advection in the finite-volume formulation (Eq. (3.21)), which brings it in accordance with the two-step *stream and collide* algorithm. In general, this means that the advection term has to be reformulated for the post-collision function $\widehat{g}$. We recall that the advection term initially contained the genuine distributions f_α. Instead of using Eq. (3.9) to convert between the pre-collision states of f_α and g_α, a new change of variable is required that relates f_α to $\widehat{g}_\alpha$, i.e.

$$f_\alpha = \frac{1}{2\left(\tilde{\tau}_g - 1\right)} \left(\left(2\tilde{\tau}_g - 1\right) \widehat{g}_\alpha - g_\alpha^{eq} \right) .$$

Details are given in appendix A.2. The post-collision advection term then reads

$$\widehat{\mathcal{R}}^t = \boldsymbol{\xi}_\alpha \cdot \boldsymbol{n}_\beta \left[\frac{1}{2\left(\tilde{\tau}_g - 1\right)} \left(\left(2\tilde{\tau}_g - 1\right) \widehat{g}_\alpha(\boldsymbol{x}_\beta, t) - g_\alpha^{eq}(x_\beta, t) \right) \right] . \tag{4.1}$$

For the preceding redistribution of the distribution functions, we choose the classical BGK operator (c.f. Eq. (3.11a)). It is, nevertheless, equally possible to choose the more stable regularized collision operator that was described in the previous chapter (c.f. Eq. (3.29)). The regularized collision can also be expressed as a particular case of multi-relaxation time model (MRT), where the non-physical modes are relaxed by $\tilde{\tau}_{\text{num}} = 1$ to impose their equilibrium value (this will be discussed in more detail in § 4.2.2). In this case it should be noted that the numerical relaxation parameter leads to a singularity in Eq. (4.1). The separation of the collision from the advection step in the proposed algorithms thus adds a constraint on the relaxation time when used with MRT models.

For the particular case of the *Heun* predictor-corrector scheme, we employ formulation (4.1) in order to calculate a first-order accurate prediction g^* at time $t + \Delta t$. Since for the correction step one can directly use the pre-collision state of the predicted

populations, we may apply formulation (3.19) so that the semi-implicit flux evaluation now reads

$$\mathcal{R}^{t+h} = \boldsymbol{\xi}_\alpha \cdot \boldsymbol{n}_\beta \left[\frac{\frac{1}{2\tilde{\tau}_g}\left(\left(2\tilde{\tau}_g - 1\right) g^*_\alpha(\boldsymbol{x}_\beta, t+\Delta t) + g^{*eq}_\alpha(\boldsymbol{x}_\beta, t+\Delta t)\right)}{2} \right] + \frac{\widehat{\mathcal{R}}^t}{2}. \tag{4.2}$$

Mono- or multi-domain approach?

Due to the possibility to separate the collision from the advection step in the Heun predictor-corrector scheme and the fact that this scheme can acquire stable results at a CFL of unity, it would be possible to match directly the *stream and collide* method with a finite-volume formulation in a mono-domain approach. Such an algorithm is summarized in Table 4.2.

Table 4.2: HLBM algorithm for a mono-domain approach

Step 1:	Collision in every node of the domain from Eq. (3.11a)
Step 2a:	Memory shift in the LBM nodes according to Eq. (3.11b)
Step 2b:	Prediction of g^*_α in the FV-LBM nodes with a simple Euler scheme applied to the post-collision populations based on Eq. (3.21) and Eq. (4.1)
Step 3:	Correction of g_α in FV-LBM nodes with modified flux evaluation from Eq. (3.21) and Eq.(4.2)
Step 4:	Back to step 1

The above algorithm implies that it is possible to assign each node individually to be either of type FV-LBM or LBM and that the distribution functions *a priori* commute regardless of the discretization method they have been iterated with. Tests revealed that such an approach is unstable. It was hence decided to use a multi-domain approach for the implementation of the HLBM algorithm, in which the *stream and collide* algorithm and the finite-volume kinetic scheme are solved on separated domains. In this case, the entire set of distribution functions is transferred between boundary nodes of the two sub-domains as it is seen in Fig. 4.2. The interface thickness that is necessary to stabilize the solution is a key parameter and must not be smaller than two grid cells. Strong gradients over the interface can require an overlap region of up to $4\Delta x$.

The multi-domain algorithm requires an additional step to transfer the complete set of distribution functions between the transition nodes of the respective domains. In particular, just before the collision of the next time step is carried out, the populations in each domain are completed with the information from the respective other grid (Table 4.3).

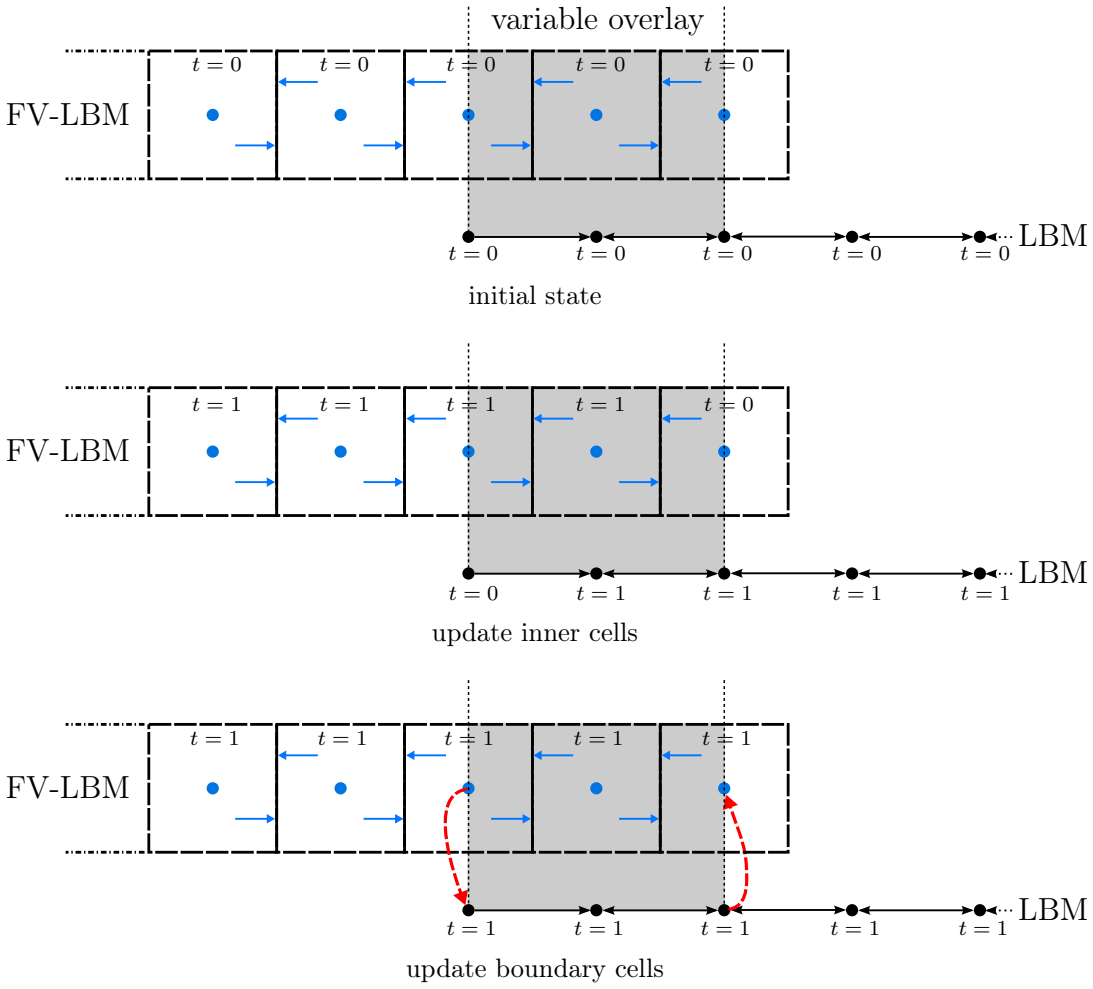


Figure 4.2: 1D schema of the hybrid LBM algorithm in a multi-domain approach with uniform grid spacing and an variable overlap region (grey) that is here set to $2\Delta x$. *Initial state:* at the initial step all nodes, including the boundary nodes of the respective domains are at the same time level. *Update inner cells:* The inner nodes are updated via collision and a subsequent streaming or rather flux evaluation step. *Update boundary nodes:* The red arrows indicate the tranfer of updated populations ($t = 1$) in pre-collision state to the boundary nodes of the respective other domain.

Table 4.3: HLBM algorithm for a multi-domain approach

Step 1:	Collision in every node of the domain from Eq. (3.11a)
Step 2a:	Memory shift in the domain where streaming is used according to Eq. (3.11b)
Step 2b:	Prediction of g_α^* with a simple Euler scheme applied to the post-collision populations based on Eq. (3.21) and Eq. (4.1)
Step 3:	Correction of g_α with modified flux evaluation from Eq. (3.21) and Eq.(4.2)
Step 4:	Mutual transfer of information in the transition nodes
Step 5:	Back to step 1

This algorithm was then implemented into our in-house LBM code and analysed in the framework of three distinct test cases that are presented in the next section.

4.1.3 Numerical validation

For the numerical validation, following test cases were solved on a 2D Cartesian grid with periodic boundary conditions in both x- and y-directions:

1. Kelvin-Helmholtz instability of a double shear layer on a uniform grid,
2. Advection of a *pseudo-isentropic* vortex on a uniform grid,
3. Kelvin-Helmholtz instability of an axial jet on a gradually refined grid.

The initial states are depicted in Fig. 4.3. The mesh of the HLBM algorithm is

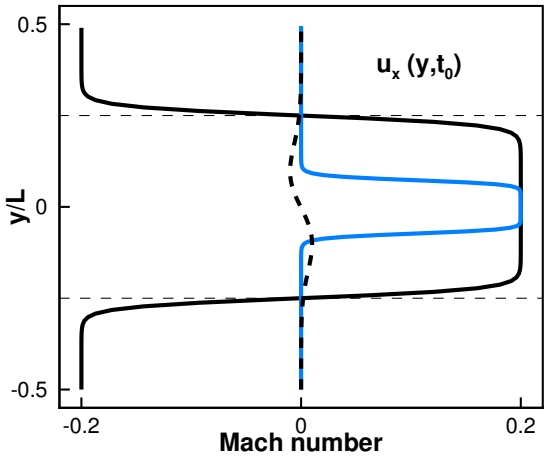


Figure 4.3: $u_x(y, t_0)$-profile of the three test cases on a uniform (black) and refined (blue) mesh, respectively. (——): double shear layer, (– – –): vortex, (——): jet. The two horizontal dashed lines indicate the position of the interfaces.

subdivided into two domains with an equal number of nodes. A continuous central part is framed by two peripheral regions that are linked via the periodicity condition in y-direction, as depicted in Fig. 4.4. The long dashed lines of the interface represent the transfer from the outer to the inner domain, whereas the short dashes indicate the transfer in the opposite direction.

The classical D2Q9 lattice (see Fig.(2.4)) is considered, with

$$g_\alpha^{eq} = \rho w_\alpha \left(1 + \frac{\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u}}{c_0^2} + \frac{(\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u})^2}{2c_0^4} - \frac{u^2}{2c_0^2}\right), \quad \alpha = 0, ...q-1. \tag{4.3}$$

We recall that in macroscopic space, this model is compliant to the isothermal weakly-compressible Navier-Stokes equation. The pressure is calculated via the isentropic equation of state (2.11). In a first instance, the control volumes in the FV-LBM were quadratic with dimensions that correspond to the lattice unit, i.e. $\Delta t_l S_\beta / V =$

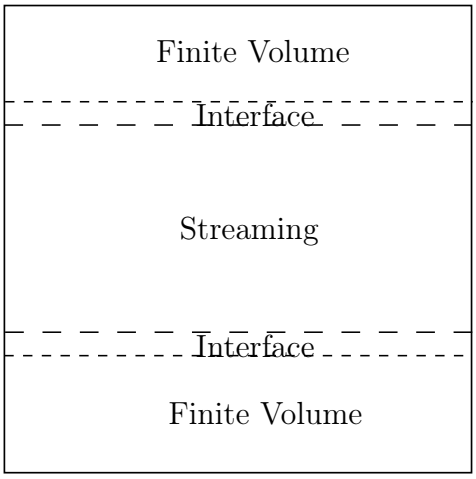


Figure 4.4: Domain configuration for the simulation of the first two test cases. In the last test case the order of the schemes is reversed. The upper and lower regions are linked via the periodicity condition in y-direction and have the same number of points as the central patch.

$\Delta t_l/\Delta x_l = 1$. The thickness of the overlap region varied between $2\Delta x_l$ and $4\Delta x_l$. In dimensional space $\Delta x = L/(N-1)$, where L and N denote the domain length and the number of grid points respectively, in both, x- and y-direction. The domain dimension was kept at $L = 1$ m for the first two test cases and was changed to $L = 12$ m in the last case. We defined a coarse ($N = 101$), intermediate ($N = 201$) and fine ($N = 401$) grained mesh, such that the discretization is referred to as Δx^c, Δx^m and Δx^f. The physical time step is then given as $\Delta t = \Delta x/\sqrt{3}c_0$. Each simulation was initialized via the equilibrium distribution function Eq.(2.21), where the macroscopic velocity $\mathbf{u}$ is linked to the first moment of g_α via $\mathbf{u}_l = \mathbf{u}/\sqrt{3}c_0$. The initialization with the relaxed state of the distribution functions hence dictates the execution of the propagation step prior to the collision (pull-scheme) in our algorithm.

4.1.3.1 Double shear layer

A first study concerns the stability of the hybrid scheme. A classical test case in this context is the periodic double shear layer [Dellar, 2001] in an under-resolved ($N^2 = 101 \times 101$) simulation. The initial state is given by the following equations

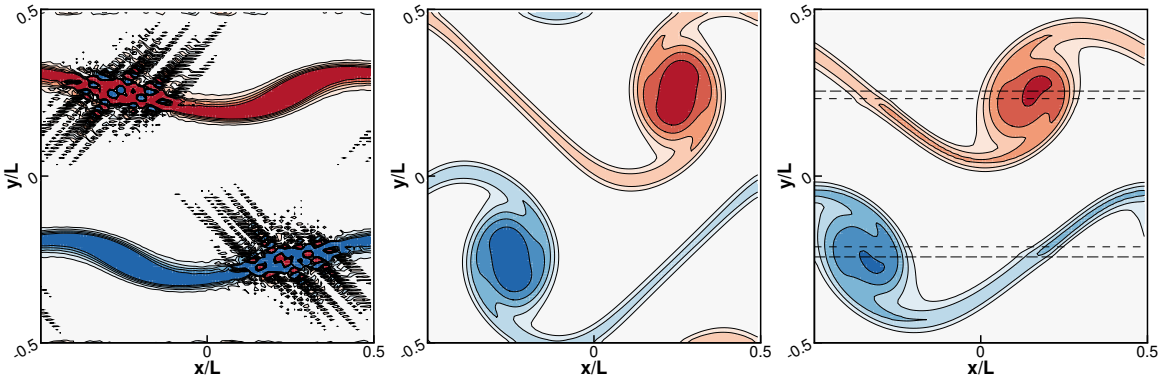
$$\begin{cases} u(y,t_0) = U_0 \tanh\left(\dfrac{y-\frac{L}{4}}{d_0}\right) \tanh\left(\dfrac{\frac{3}{4}L-y}{d_0}\right) \\ v(x,t_0) = U_0 a_0 \sin\left(2\pi\left(\dfrac{x}{L}+0.25\right)\right), \end{cases} \tag{4.4}$$

with $U_0 = 0.2c_0$ and $\rho_0 = 1.0$ kg/m^3. The thickness of the shearlayer is set to $d_0 = 2.5\Delta x$. The flow is perturbed by a sinusoidal cross-flow with an amplitude of $a_0 =$

$0.01p_0$. Using Crocco's relation, an expression for the density as a function of M_0 and $u(y, t_0)$ can be obtained:

$$\rho(y, t_0) = \left[1 + \frac{\gamma - 1}{2}\mathrm{M}_0^2 u(y, t_0)\left(1 - u(y, t_0)\right)\right]\rho_0,$$

which can be approximated by $\rho(x, y, t_0) \approx \rho_0$ in the low Mach number regime. The test case was simulated with each algorithm, *i.e.* LBM, FV-LBM and HLBM on the coarse grid $\Delta x^c = 0.01$ m. For the hybrid LBM algorithm, the different domains were arranged in the order that is shown in Fig. 4.4. We recall that the classical LBM is coupled to a FV-LBM with a Heun predictor-corrector scheme in time and a QUICK approximation in space that allows us to maintain a stable solution at CFL = 1. Thus, the time step was fixed to $\Delta t^c = 2.49 \times 10^{-5}$ s in all computations. The number of temporal iterations was equal to $t^* = t/\Delta t^c = 3000$, where t denotes the physical time of the simulation. It is known from literature, e.g. [Dellar, 2001] that the *stream and collide* algorithm with a single relaxation time does not yield stable results for this test case. This is confirmed in Fig. 4.5a.



(a) classical LBM **(b)** FV-LBM **(c)** HLBM

Figure 4.5: Vorticity field in the range $-2100\ \mathrm{s}^{-1} < \omega_z < 2100\ \mathrm{s}^{-1}$ of the double shear layer on a coarse mesh ($N = 101$) at $t^* = 858$ (LBM) and $t^* = 1400$ (FV-LBM, HLBM), respectively. The three simulations share the same initial condition.

At the beginning of the roll-up, an instability arises from the region of the free shear layer, where the generation of two spurious vortices is expected. If, however, the FV-LBM is employed, the dissipation of the scheme damps the amplification of the non-hydrodynamic modes – responsible for the failure of the *stream and collide* algorithm – and the solution remains stable as can be seen in Fig. 4.5b. In order to better assess the stability of the HLBM, the simulation is initialized such that each shear layer is placed on an interface. Results show that the higher dissipation of the FV-LBM has a stabilizing effect on the solution of the traditional LBM. In addition, we

confirm that the coupling of the two algorithm does not pose an additional trigger for the numerical instability that is reported in Fig. 4.5a. The comparison of the energy

$$E(t) = \int\int \left(u^2(t) + v^2(t)\right) dxdy$$

in Fig. 4.6 reveals that the hybrid scheme is slightly less dissipative than the FV-LBM. The solution of the LBM algorithm blows up right at the beginning of the the roll-up indicated by the energy drop.

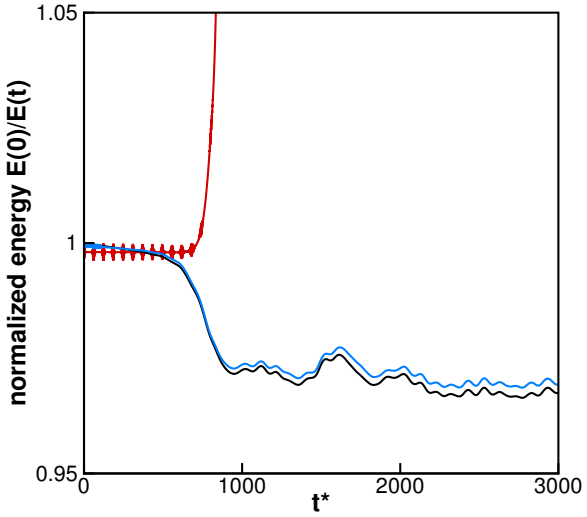
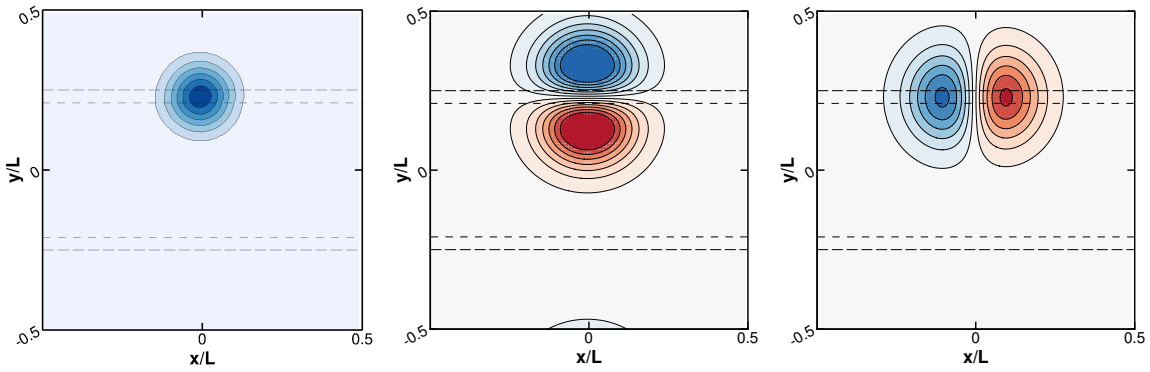


Figure 4.6: Normalized energy during the shearlayer roll-up: ( ) LBM; ( ) FV-LBM; ( ) HLBM.

4.1.3.2 Advection of a vortex

For a more thorough error analysis, the *pseudo isentropic* vortex (Eq. (3.34)) was used. It is now advected normal to the interfaces with $V_0 = 0.2c_0$ and a reduced parameter for the vortex strength of $\kappa = 0.01$. Such a low strength has the effect that the difference between the free stream density ρ_0 and the density in the core of the vortex $\rho(r = 0)$ only amounts to $\delta\rho = 1.35 \times 10^4$ kg/m^3. The corresponding pressure difference $\delta p = \delta\rho c_0$ is about 16 Pa. The density isocontours in Fig. 4.7 are therefore sensitive enough to detect anomalies relevant to the aeroacoustic level. The characteristic radius ranged from $R = 10\Delta x^c$ for the coarse mesh ($N = 101$, $\Delta x^c = 0.01$ m) to $R = 40\Delta x^f$ for the fine mesh ($N = 401$, $\Delta x^f = 0.0025$ m). The test case was simulated with the three different algorithms: LBM, FV-LBM and HLBM. The time step Δt was kept constant among the different solvers, yielding a CFL number equal to one.

The solution of the vortex was initialized in the centre of the domain. In Fig. 4.7, isocontours of the macroscopic variables are shown for the coarse mesh ($N = 101$) at



$0.9998\ \text{kg/m}^3 < \rho < 1.0\ \text{kg/m}^3$ $\quad -3.0\ \text{m/s} < u < 3.0\ \text{m/s}$ $\quad 65.0\ \text{m/s} < v < 72.0\ \text{m/s}$

Figure 4.7: Isocontours of the macroscopic variables (ρ, u, v) computed with the HLBM algorithm on the coarse mesh ($\Delta x^c = 0.01$ m) at $t^* = 220$. The vortex moves vertically with $V_0 = 0.2c_0$.

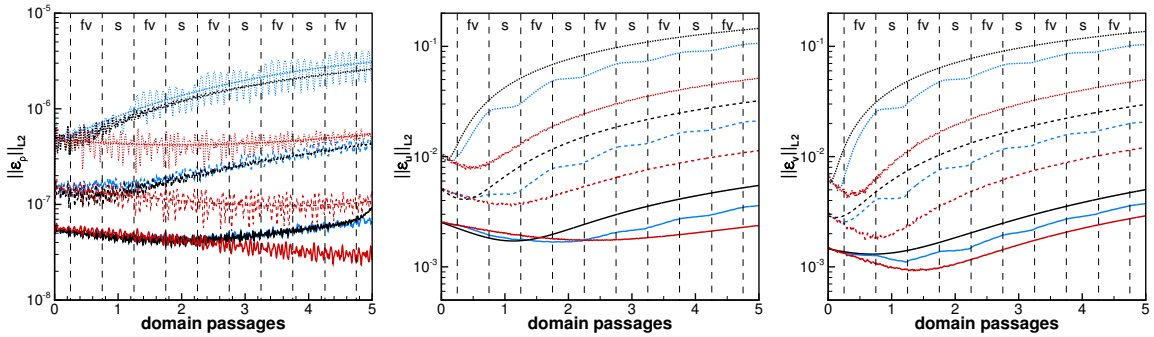
$t^* = 220$. For the HLBM, this time instant corresponds to the displacement between the center of the domain and the interface. Qualitatively, no distortion is notable when the vortex crosses the interface and an error analysis shall give a better estimation of the error introduced by the HLBM.

The error is studied by calculating the L_2 norm, defined as

$$||\epsilon_\phi||_{L_2}(t) = \sqrt{\frac{\sum_{x,y} \left(\phi(x,y,t) - \phi_{al}(x,y,t)\right)^2 \Delta x \Delta y}{\sum_{x,y} \Delta x \Delta y}},$$

where ϕ_{al} is the analytical solution and $\phi = (\rho, u, v)$. The analysis was conducted for the two individual and hybrid algorithms. The error was recorded over 5 domain passages, corresponding to $t^* = 4400$, 8000 and 17600 on the grid with spacing Δx^c, Δx^m and Δx^f, respectively. The results for each macroscopic variable are presented in Fig. 4.8.

A third-order polynomial fit is used to smooth out the high-frequency fluctuations of $||\epsilon_\rho||_{L_2}(t)$ observed for the LBM and HLBM algorithms with coarse and intermediate spacing. We observe that the deviations from the analytical solution for the density are similar for the two algorithms that contain a FV formulation (HLBM, FV-LVM), whereas the value of $||\epsilon_\rho||_{L_2}(t)$ is considerably smaller when the classical LBM algorithm is used (Fig. 4.8a). Regular oscillations of the HLBM algorithm, especially for the coarse mesh (Δx^c), nevertheless, are not negligible and will be discussed later. The error in the velocities $||\epsilon_u||_{L_2}$ and $||\epsilon_v||_{L_2}$ of the hybrid scheme is bounded between the error of the *stream and collide* algorithm and the error of the pure finite-volume



(a) $||\epsilon_\rho||_{L_2}$ **(b)** $||\epsilon_u||_{L_2}$ **(c)** $||\epsilon_v||_{L_2}$

Figure 4.8: L_2-error of macroscopic flow variables. The different algorithms are distinguished by color: (———) LBM, (———) FV-LBM, (———) HLBM. The mesh resolution is represented by different line patterns: (· · · · ·) $N = 101$; (– – –) $N = 201$; (———) $N = 401$.

algorithm. It can be seen that the error curve for the HLBM takes on the slope of the respective individual scheme while traversing the different domains.

If we now trace the value of the L_2-norm after the vortex has passed the domain five times against the mesh resolution, the global accuracy of the different schemes is obtained (Fig. 4.9). The dashed line in each figure represents the (-2)-slope in a double logarithmic representation. The order-two convergence is confirmed for the individual schemes as well as for the hybrid algorithm, as expected.

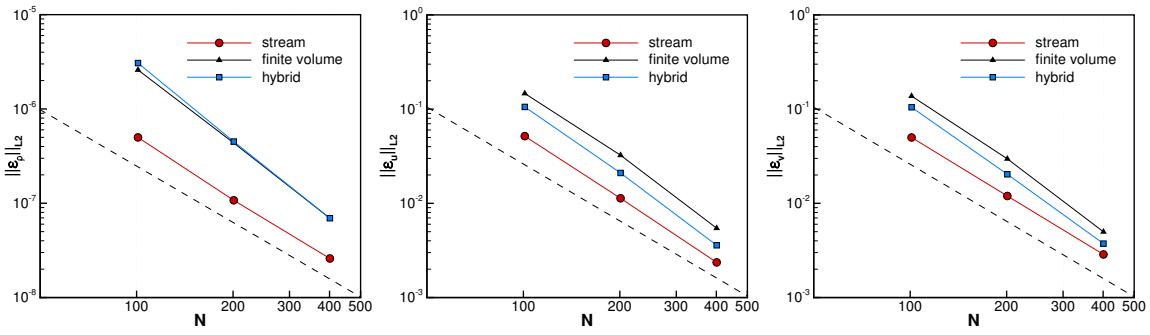


Figure 4.9: L_2-error of macroscopic variables for varying mesh size N after 5 domain passages. The different algorithms are distinguished by color: (——●——) LBM; (——▲——) FV-LBM; (——■——) HLBM. The dashed line indicates the slope -2.

The superior performance of the classical LBM compared to the FV-LBM is clearly due to a lower dissipation. It is reassuring to find the error of the HLBM in the first-order moment (momentum) in between the norms of the single algorithms, suggesting that no significant error is introduced due to the coupling. This situation, however, is

different for the zeroth-order moment (density), which requires further investigation. Given that $||\epsilon_\rho||_{L_2}$ oscillates continuously indicates that the error – as a consequence of the coupling – is locked inside the domain due to the periodic boundary conditions in x- and y-direction. In order to reduce these fluctuations, the domain was extended in x-direction by a factor 5 and the L_2 error analysis was repeated for the coarse grid (Δx^c), whereas the integration area remained unchanged (L^2) and centred around the vortex. The results are reported in Fig. 4.10b. With respect to the results from a square domain (Fig. 4.10a), we observe that the error of the classical LBM and the FV-LBM are smoothed out but have maintained their magnitude. This situation is different for the error of the HLBM, which has now dropped below the error of the FV-LBM apart from a double peak for each time that the vortex leaves the central domain (*stream and collide*) to enter the FV-LBM mesh. In order to localize the origin of the error that is attributed to the presence of an interface, the maximum of the relative error was calculated according to Eq.(4.5) and presented for two instances, in which the vortex crosses the upper and lower interface respectively.

$$\epsilon_\rho\left(x,y,t\right) = \frac{|\rho(x,y,t) - \rho_{al}(x,y,t)|}{\rho_{al}(x,y,t)} \tag{4.5}$$

Fig. 4.10c, which represents the maximum of the relative error at $t^* = 660$, when the center of the vortex is located on the lower edge of the central domain, shows the dissipation of the vortex as the only source of error. When the vortex center is, however, located close to the upper edge of the central domain, two patches emerge, indicating the deviation from the analytical solution by 0.1%. We can therefore show that the source of the increased error in ρ of the HLBM arises from the interface passage LBM $\rightarrow$ FV-LBM. This reflection from the interface results in an oscillating error due to the periodicity condition in x-direction. In order to exclude any error in the algorithm, we reversed the direction of the flow and obtained an identical error curve. This implies that the HLBM algorithm is direction-sensitive. In Di Ilio et al. [2017], the aforementioned transition (LBM $\rightarrow$ FV-LBM) only encounters free stream flow, whereas the wake of the cylinder is transported over an interface FV-LBM $\rightarrow$ LBM. This might explain why this problem was not observed in their study. The comparison of the HLBM algorithm presented here and the one in Di Ilio et al. [2017] is, however, not that straightforward, as they differ in important aspects, such as node evaluation, cell geometry, interface thickness and number of subiterations.

A possible explanation for the reflections lies in the different dissipative behaviour of the two single algorithms. In Fig. 4.10a, we observe that the oscillation begins only when the vortex passes the interface LBM $\rightarrow$ FV-LBM for a second time. That is when the errors of the single algorithms, *i.e.* the dissipation of the solution, already differ by half an order of magnitude. As a consequence, the solution in the LBM domain will

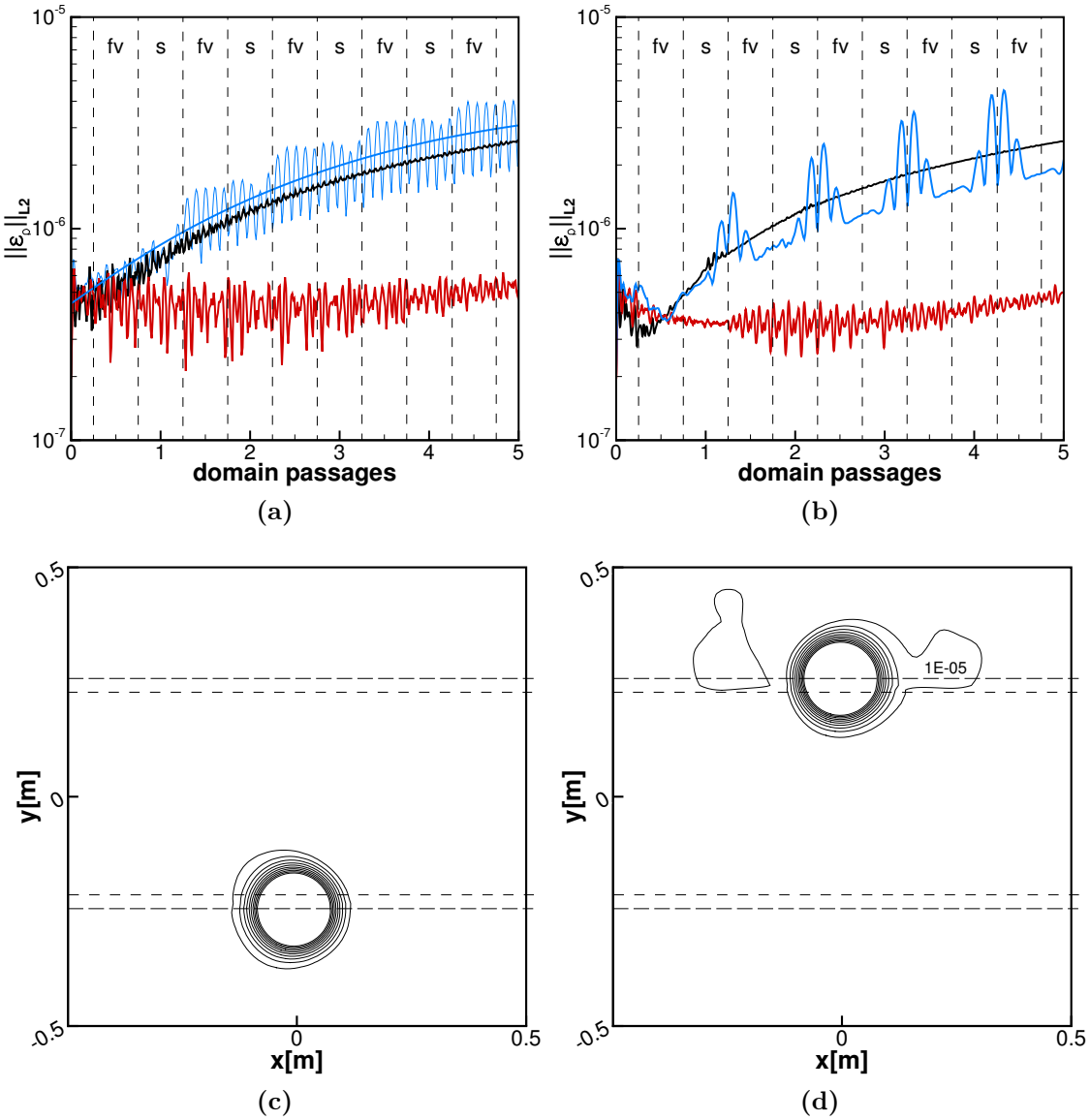


Figure 4.10: Error L_2 and maximum of the relative error of the density on the coarse mesh ($N = 101$). a) Error $||\epsilon_\rho||_{L_2}$ on a doubly periodic mesh and b) a quasi-infinite mesh in x as a function of the normalized convected distance of the vortex. Algorithms are distinguihed by color: (———) LBM; (———) FV-LBM; (———) HLBM. c)– d): Maximum of the relative error during interface crossing at $t^* = 660$ and $t^* = 1100$.

contain scales that have already disappeared from the FV-LBM mesh. When entering the FV domain, the non-supported structures will be reflected back into the LBM domain, similar to a mesh transition towards a coarser grid with a single algorithm.

Using the Heun scheme with QUICK approximation to construct the HLBM, we combined the two algorithms that mark the opposite extremes of the dissipation diagram (see Fig. 4.1). In an attempt to construct a hybrid method with a less dissipative scheme, such as the Heun scheme with a centred approximation of the surface fluxes and a CFL number of 0.5, the overall quality of the interface was, however, not improved. Although the discrepancy between the solutions was reduced, the low dissipation gave rise to the development of instabilities.We thus conclude that a certain degree of dissipation is required to stabilize the solution at the interface. The concomitant refection is an inevitable artefact. Despite this error, we nevertheless emphasize that our HLBM algorithm delivers reliable results at the scale relevant to aerodynamic applications. This shall be illustrated in the following section with the example of a two-dimensional axial jet.

4.1.3.3 Axial jet on a gradually refined mesh

In order to demonstrate a possible scope of the HLBM algorithm, the arrangement of the underlying algorithms is swopped in the following test case and the height of the domain is increased to $L_y = 12$ m. A two dimensional jet with a y-dependent velocity profile is placed in the central domain, where now the FV-LBM is solved

$$\begin{cases} u(y,t_0) = \dfrac{1}{1+\sinh^{2n}(y)}U_0 \\ v(x,t_0) = U_0 a_0 \sin\left(2\pi\left(\dfrac{x}{L}+0.25\right)\right), \end{cases} \tag{4.6}$$

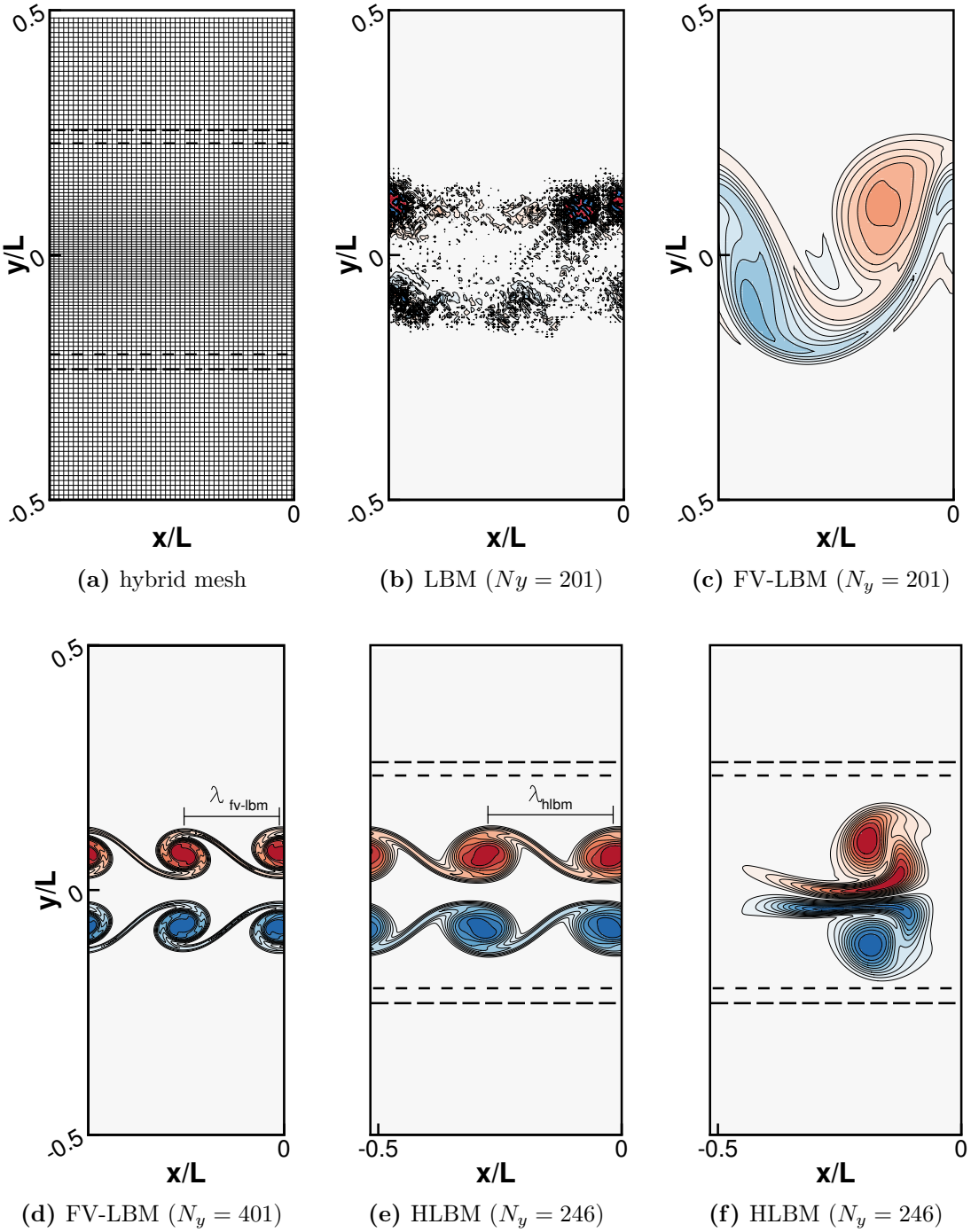
with $U_0 = 0.2c_0$ and $a_0 = 0.01p_0$ being the amplitude of the sinusoidal perturbation of the jet. As in the first test case, the density field was simplified to $\rho(x,y,t_0) = \rho_0$ under the low Mach number hypothesis. The two inflection points of the $u(y,t_0)$-profile are located at $y = \pm 0.8814$ m. The parameter n controls the width of the shear layer and was set to $n = 5$, which results in a value of $\delta_w = U_0/\max(du/dy) = 0.283$ m. The ratio of jet to shear layer thickness then amounts to $r_t = 6.2$. This value was chosen intentionally, as it is slightly higher than the critical ratio r_c of 4 to 5 (with respect to the radius) for two counter-rotating vortices to merge. Under the hypothesis that the radius of the vortices, emerging from the shear layers corresponds roughly to δ_w, we would in theory expect a symmetric roll-up of the two shear layers. Apart from a correct representation of the expected flow physics, we put the algorithm again in the light of numerical stability to evaluate its performance. In a first instance, we applied the single algorithms (LBM and FV-LBM) on the fine ($\Delta x^f = 0.03$ m) and intermediate ($\Delta x^m = 0.06$ m) mesh to discretize the shear layers in y-direction with

9.5 and 4.75 points, respectively. The domain extension in x-direction should be ideally a multiple of the distance λ between two vortices. We therefore chose a larger domain extension in x-direction ($L_x = 2L_y$) to reduce the remainder of $n_\lambda = L_x/\lambda$, where n_λ is the number of wavelength in x. Once the wavelength was obtained experimentally, the computation was repeated with a domain extension $L_x = 2\lambda$. Results for this configuration are presented in Fig. 4.11.

The axial jet – similar to the previously discussed double shear layer – leads to the failure of the classical BGK *stream and collide* algorithm (Fig. 4.11b). The FV-LBM algorithm, on the other hand, delivers stable results due to a higher dissipation (Fig. 4.11c). Nevertheless, the fine mesh-size Δx^f is required to capture the expected physics, *i.e.* two symmetric double shear layers (Fig. 4.11d). The higher dissipation inherent to the intermediate mesh size Δx^m increases the thickness of the shear layer such that the ratio r_t for the two shear layers to develop independently falls below the critical value r_c. The result is an inverted von Kármán vortex trail, where the initially separated shear layers have coalesced to form a single mixing layer (Fig. 4.11c). It can be concluded that the solution is very sensitive to the resolution in the region of the two shear layers, particularly in y-direction. We thus apply the HLBM on the intermediate mesh (Δx^m) in such a way that the classical LBM is solved in the outer regions of the computational domain and the FV-LBM is used for the central part. Such a configuration enables a gradual mesh refinement in the region of the jet. Between the interfaces, the dimension of the control volume in y-direction varied on the interval $[0.5\Delta y^m; \Delta y^m]$ according to

$$\Delta y = \left(1 - 0.5 \sin\left(\frac{i\pi}{N-1}\right)\right) \Delta y^m \quad i = 0, ..., N-1, \tag{4.7}$$

where $\Delta y^m = \Delta x^m$. The vertical dimension of the domain is now discretized by $N_y = 246$. The change in mesh size leads to a maximal CFL number of 2 so that the time step of the HLBM was reduced by a factor two to recover CFL = 1 in the smallest cells, *i.e.* the cells that lie on the x-axis. The physical time step Δt^f is hence the same as in the FV-LBM simulation with $N_y = 401$. Note that in the LBM domain, the time step is still $\delta t^m = 2\Delta t^f$, so that in this particular case linear temporal interpolation is required at the interface to couple the two domains. The result of the HLBM on a gradually refined intermediate mesh ($N_y = 246$) at two different solution times are shown in Fig. 4.11e and Fig. 4.11f. We confirm the symmetric roll-up of the shear layer similar to the result of the FV-LBM on the fine mesh. The vortices appear slightly more dissipated, which also manifests in a greater vortex displacement, $\lambda_{hlbm} = 0.284$ m compared to $\lambda_{fv-lbm} = 0.197$ m. With respect to the results from the FV-LBM algorithm on the intermediate mesh ($N_y = 201$), we observe that the gradual mesh refinement towards the abscissa visibly reduces the smoothing effect of dissipation on the gradients. As a consequence, the ratio r_t remains above the critical



(a) hybrid mesh **(b)** LBM ($Ny = 201$) **(c)** FV-LBM ($N_y = 201$)

(d) FV-LBM ($N_y = 401$) **(e)** HLBM ($N_y = 246$) **(f)** HLBM ($N_y = 246$)

Figure 4.11: HLBM with gradual mesh refinement applied to a two-dimensional axial jet. a) Detail of the hybrid mesh ($N_x = 51$, $N_y = 123$) with coarse spacing Δx^c. Dashed lines indicate the interface. b–f) Vorticity contours $-1800\ \mathrm{s}^{-1} < \omega_z < 1800\ \mathrm{s}^{-1}$ obtained with the algorithms at different instants in time: b) $t = 0.36$ s; c–e) $t = 0.46$ s; f) $t = 0.7$ s.

ratio r_c before the K-H instability is triggered leading to an independant roll-up of the two layers. In Fig. 4.11f it is confirmed that the algorithm remains stable even when secondary instabilities set in (leap-frogging). We are thus able to capture the same physical phenomenon by using 30% of the nodes compared to a FV-LB simulation on a uniform fine grid (Δx^f). In addition the more efficient *stream and collide* algorithm is solved in about 40% of the nodes, which leads to a total saving of 80% in CPU.

4.2 HLBM$_{\text{NS}}$: LBM – NS

In this section, we present the theory and first results of a hybrid lattice Boltzmann method that couples the second-order lattice Boltzmann equations with the isothermal Navier-Stokes equations (HLBM$_{\text{NS}}$). Due to the non-linearity of the advection term ($\boldsymbol{u} \neq$ cst.), the Navier-Stokes equations are typically discretized within the Eulerian framework. With respect to the classical *stream and collide* algorithm, we therefore expect differences between the two schemes in terms of mesh structure, dissipation and efficiency. As in case of the HLBM, the Eulerian part of the algorithm relies on a discretization of physical space in finite volumes. While the coupling of the LB and the NS mechanics combines two different perceptions of a fluid (kinetic and discrete *versus* continuum-based), it can be shown that the FV-NSE can be interpreted as a moment formulation of the FV-LBE. In order to make this clear, we will begin the development of the algorithm by restating the DVBE

$$\frac{\partial f_\alpha}{\partial t} + (\boldsymbol{\xi}_\alpha \cdot \nabla) f_\alpha = -\frac{1}{\tau}\left(f_\alpha - f_\alpha^{eq}\right).$$

Equally, the above equation may be expresses as an infinite series of statistical equations, by taking moments of f in increasing order

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \nabla \cdot \rho u_i = 0 \\ \dfrac{\partial \rho u_i}{\partial t} + \nabla \cdot P_{ij} = 0 \\ \dfrac{\partial P_{ij}}{\partial t} + \nabla \cdot Q_{ijk} = -\dfrac{1}{\tau} P_{ij}^{neq} \\ \qquad \vdots \end{cases} \tag{4.8}$$

with

$$\rho = \boldsymbol{M}^{(0)}, \quad \rho u_i = \boldsymbol{M}^{(1)}, \quad P_{ij} = \boldsymbol{M}^{(2)}, \quad Q_{ijk} = \boldsymbol{M}^{(3)},$$

and

$$\sum_\alpha f_\alpha^{neq} = 0, \quad \sum_\alpha f_\alpha^{neq} \boldsymbol{\xi}_\alpha = 0.$$

One immediately recognizes that the set of equations represents a moment cascade, in which a particular moment $\boldsymbol{M}^n$ requires knowledge about the moment of order $\boldsymbol{M}^{n+1}$, to be solved. System (4.8) can also be interpreted as a set of macroscopic conservation equations taking into account an infinite range of physical time scales. This brings us back to the Chapman-Enskog analysis discussed in § 2.2.6. Instead of solving a transport equation for P_{ij}, we may use an $\mathcal{O}(\epsilon^2)$-truncated multi-timescale

approximation

$$P_{ij} \approx P_{ij}^{(0)} + \epsilon P_{ij}^{(1)} + \mathcal{O}(\epsilon^2)$$

to intercept the moment cascade (4.8) after the second equation. Expressions for $P_{ij}^{(0)}$ and $\epsilon P_{ij}^{(1)}$ as a function of ρ and $\boldsymbol{u}$ were previously derived (cf. § 2.2.6), i.e.

$$\begin{aligned} P_{ij}^{(0)} &= c_0^2 \rho \delta_{ij} + \rho u_i u_j \\ \epsilon P_{ij}^{(1)} &= -\mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \underbrace{\tau \frac{\partial \rho u_i u_j u_k}{\partial x_k}}_{\text{error}}, \end{aligned}$$

so that the system (4.8) is closed. Under the low Mach number assumption, the error term in the expression for $P_{ij}^{(1)}$ is neglected and we may write the above system in the following form

$$\begin{aligned} &\frac{\partial \rho}{\partial t} + \nabla \cdot \rho u_i = 0 \\ &\frac{\partial \rho u_i}{\partial t} + \nabla \cdot \left[\underbrace{\rho u_i u_j + p \delta_{ij}}_{\boldsymbol{P}^{(0)}} - \underbrace{\mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)}_{\epsilon \boldsymbol{P}^{(1)}} \right] = 0, \end{aligned}$$

where $p = c^2 \rho$. We have thus retrieved the isothermal Navier-Stokes equations consisting of the conservation equations for mass and momentum under the assumption that the bulk viscosity equals $\mu' = \frac{2}{3}\mu$ (cf. Eq. (2.36)). With respect to the previous HLBM algorithm on a D2Q9 lattice (§ 4.1), we notice that instead of solving nine equations for each discrete distribution function f_α of the DVBE, this approach only requires the evaluation of three conserved quantities ρ, ρu_x and ρu_y. A further advantage of solving the statistical equations is the increased stability margin of the finite-volume scheme. Other than the FV-LB method, the propagation speed to calculate the CFL number, in the following denoted as $\mathrm{CFL_{NS}}$, is based on the macroscopic fluid velocity $\boldsymbol{u}$. Taking acoustics into account, the propagation velocity is set to $\boldsymbol{u}_{\mathrm{prop}} = c_0 + |\boldsymbol{u}|$, which yields following definition of the local $\mathrm{CFL_{NS}}$ number

$$\mathrm{CFL_{NS}} = \frac{\boldsymbol{u}_{\mathrm{prop}} \Delta t}{\Delta x} = \frac{(c_0 + |\boldsymbol{u}|)\Delta t}{\Delta t \sqrt{3} c_0} = \frac{1 + \mathrm{M}}{\sqrt{3}}, \tag{4.10}$$

with $\mathrm{M} = |\boldsymbol{u}|/c_0$ being the local Mach number. It becomes obvious that the $\mathrm{CFL_{NS}}$ number is smaller or equal to one for $\mathrm{M} \leq \sqrt{3} - 1$. Since in the D2Q9 model we are limited to the small Mach number range, one can generally say that $\mathrm{CFL_{NS}} < \mathrm{CFL_{LBM}}$ for equal time steps and discretization of space.

4.2.1 Navier-Stokes solver

A Navier-Stokes solver that falls within the framework of the classical LBM is fairly simple to construct. Due to the low Mach number constraint, the flow regime is considered weakly compressible, which avoids the need for a supplementary pressure equation (e.g. Poisson). On the other hand, the flow is considered isothermal, since the characteristic particle speed c in the discrete Boltzmann model is fixed and proportional to the speed of sound $c_0 = k_B T_0/m$, where T_0 denotes a constant temperature (cf. (2.10)). Instead of solving the energy equation, which can be interpreted as the linkage between pressure and density, the isentropic equation of state $p = \rho c_0^2$ can be used instead, as it was elaborated in § 2.2.1. We are therefore left with the conservation equations for mass and momentum, which can formally be written as

$$\frac{\partial \phi}{\partial t} + \nabla \cdot F(\phi) = 0, \tag{4.11}$$

with

$$\phi = \begin{pmatrix} \rho \\ \rho \boldsymbol{u} \end{pmatrix} \quad \text{and} \quad F(\phi) = \begin{pmatrix} \rho \boldsymbol{u} \\ \rho \boldsymbol{u} \otimes \boldsymbol{u} + p\boldsymbol{I} - 2\mu \boldsymbol{S} \end{pmatrix}.$$

For the temporal discretization of the weakly compressible Navier-Stokes equations (4.11), a 4-step Runge-Kutta time-marching scheme is chosen, which is theoretically stable at a CFL number of up to 2.8 [Eckhart and Oertel Jr., 2009]. Let us assume that the solution ϕ^t and $F(\phi^t) = F^t$ are known at time t. Then, the update of ϕ may be written as

$$\phi^{t+\Delta t} = \phi^t - \frac{\Delta t}{6}\left(\mathcal{R}(F^t) + 2\mathcal{R}(F^{(1)}) + 2\mathcal{R}(F^{(2)}) + \mathcal{R}(F^{(3)})\right), \tag{4.12}$$

which represents the final step of the updating cycle. $\mathcal{R}$ it the spatial discretization operator. The necessary sub-steps to obtain $F^{(1-3)}$ are summarized as

$$\begin{aligned}
&\text{step 1: } \phi^{(1)} = \phi^t - \frac{\Delta t}{2}\mathcal{R}(F^t), \\
&\text{step 2: } \phi^{(2)} = \phi^t - \frac{\Delta t}{2}\mathcal{R}(F^{(1)}), \\
&\text{step 3: } \phi^{(3)} = \phi^t - \Delta t \mathcal{R}(F^{(2)}).
\end{aligned}$$

Note that by updating ϕ, only the first row of F is immediately obtained, i.e. $F_1 = \phi_2$, where the subscript indicates the row number. The second entry, F_2, is obtained in a separate step. The calculation of the momentum flux $\rho \boldsymbol{u} \otimes \boldsymbol{u}$ and the pressure $p\boldsymbol{I}$ is straightforward. The space derivatives of the strain rate tensor $\boldsymbol{S}$ are obtained by a second-order central-difference approximation.

The choice of a 4-step Runge-Kutta scheme was largely motivated by our previous experiences with the hybrid algorithm combining two different LB schemes (HLBM). In order to match the unity CFL condition imposed by the classical LBM, a dissipative upwind scheme was employed to stabilize the FV-LB scheme. This led to the coupling of two schemes with very opposing dissipation (cf. Fig. 4.1). Using a higher-order approximation for the temporal derivative allows the use of a less dissipative scheme for approximation in space. In particular, the fluxes over the cell surfaces, contained within the operator $\mathcal{R}$ in Eq. (4.12) are calculated with a second-order central scheme. In order to get a fair idea of the dissipation behaviour of the Navier-Stokes solver, the same test case was used as for the evaluation of the different FV-LBM schemes in § 4.1.1. We recall that the *pseudo-isentropic* vortex was enforced in the centre of a 1 m $\times$ 1 m square domain, with $N = 51$ and $\Delta x = 0.02$ m, respectively. The same parametric setting is chosen as before, i.e.

$$R = 0.1 \text{ m}, \quad \kappa = 0.04, \quad c_0 = 343.2 \text{ m/s}, \quad \rho_0 = 1.0 \text{ kg/m}^3,$$

leading to a maximum flow velocity of $u_{max} = v_{max} = 0.04 c_0$. The density is traced after a physical time of 0.1 s along the y-axis running through the vortex centre, i.e. $\rho(x = 0, y)$. Results are shown in Fig. 4.12. For better comparability, the behaviour of the two FV-LBM solver with the Heun predictor-corrector scheme is depicted as well.

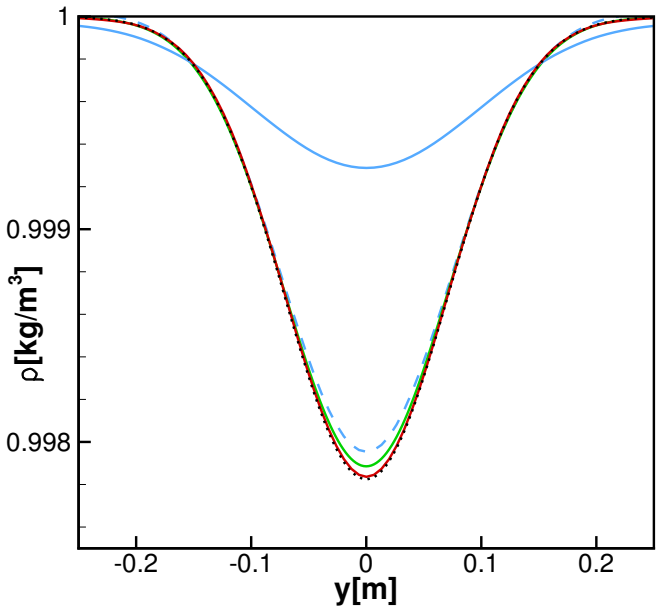


Figure 4.12: Density profiles of the static vortex at physical time 0.1 s obtained from a classical LBM solver, a FV-LBM solver with Heun predictor-corrector scheme and a Runge-Kutta Navier-Stokes solver: (·····) analytical solution; (——) streaming $\text{CFL}_{\text{LBM}} = 1$; (——) Heun QUICK $\text{CFL}_{\text{LBM}} = 1$; (– – –) Heun centred $\text{CFL}_{\text{LBM}} = 0.5$; (——) Navier-Stokes RK4 centred $\text{CFL}_{\text{LBM}} = 1$ ($\text{CFL}_{\text{NS}} \approx 0.6$).

With regard to Fig. 4.12, we realize that the Navier-Stokes solver performs even better in terms of dissipation than the least dissipative FV-LBM solver (centred approximation in space with a Heun predictor corrector scheme for the time derivative). In addition, the Navier Stokes solver yields stable results at a $\mathrm{CFL}_{\mathrm{LBM}}$ number of unity. This is, however, not surprising since the $\mathrm{CFL}_{\mathrm{NS}}$ number, which controls the stability of the NS scheme, is only about 0.6 according to Eq. (4.10). On the downside, a multi-stage scheme for the coupling requires solutions at intermediate time levels, which are *a prior* not available at the interface with the LBM solver. Two solutions exist to this problem, which will be discussed in § 4.2.3.

4.2.2 Communication between the solvers

In contrast to the previous study, the communication between the two algorithm includes the macroscopic variables. The transfer of information from the LB routine to the Navier Stokes solver is straightforward, as the solutions to the NS equations are the velocity moments of the mesh specific distribution functions g_α up to first order.

variable LBM solver		moment space		variables NS solver
g_α	$\rightarrow$	ρ	$\rightarrow$	ρ
		$\rho\boldsymbol{u}$	$\rightarrow$	$\boldsymbol{u}$
		$\boldsymbol{\Pi}$	$\overset{scaling}{\rightarrow}$	$\boldsymbol{P}' = \boldsymbol{P}^{(0)} + \epsilon\boldsymbol{P}^{(1)}$

In order to obtain $\boldsymbol{P}^{(0)}$ and $\epsilon\boldsymbol{P}^{(1)}$ at the interface we assume that $\boldsymbol{P}' \approx \boldsymbol{P}$, where $\boldsymbol{P}$ denotes the second-order moment of the genuine distribution function f_α, and $\boldsymbol{P}'$ its multi-timescale expansion, truncated at second order in the Knudsen number, i.e. $\boldsymbol{P}' = \boldsymbol{P}^{(0)} + \epsilon\boldsymbol{P}^{(1)} + \mathcal{O}(\epsilon^2)$. The momentum flux tensor at the interface is thus taken as the second-order moment of the distribution functions. The fact that a second-order LBM works with the mesh-specific distribution functions g_α, however, requires a scaling operation for the non-equilibrium part of $\boldsymbol{\Pi}$, such that

$$\epsilon\boldsymbol{P}^{(1)} = \frac{\tau}{\tau_g}\boldsymbol{\Pi}^{neq}.$$

The coupling in the opposite sense makes reappear the mapping problem from the macroscopic to the mesoscopic space. In the two-dimensional case, the Navier Stokes solver provides six updated macroscopic variables $\{\rho, u_x, u_y, P'_{xx}, P'_{xy}, P'_{yy}\}^{t+\Delta t}$ to reconstruct nine distribution functions. Literature provides different possibilities to overcome this shortage. In the pioneering work of Bourgat et al. [1996], Le Tallec and Mallinger [1997], the velocity distribution function are reconstructed with a first-order approximation of g_α in ϵ

$$g_\alpha \approx g_\alpha^{(0)} + \mathcal{O}(\epsilon),$$

and calculated with the Maxwellian distribution (2.21) that is a function of ρ and $\boldsymbol{u}$. Despite the simplicity, this assumption corresponds to an inviscid fluid at the macroscopic level (Euler flow). Feiz [2006] suggested to use a partial reconstruction, similar to that discussed in § 3.3.2. Symmetry conditions were imposed on certain populations, in order to reduce the number of unknown populations to the number of hydrodynamic variables. Nevertheless, our tests in § 3.3.2 have revealed that the regularized reconstruction, in which the entire set of distribution functions are modified, is more stable. Here, we present a fourth possibility to establish a uniquely defined mapping from the macroscopic to the mesoscopic world using multi-relaxation time LBM (MRT-LBM) [D'Humières, 1992]. First of all, a theoretical overview in 2D shall be presented. An extension to 3D is straightforward but was omitted for the sake of simplicity.

The generalized LBM enables the transformation between the population basis and the moment basis by using a non-unique transformation matrix $\boldsymbol{\mathcal{T}}$ such that

$$\boldsymbol{\mathcal{T}}|g_\alpha\rangle = |m_\alpha\rangle \quad \text{with} \quad \alpha = 0, .., q-1,$$

where $|m_\alpha\rangle$ is the vector of moments. Let us define the entries of $|m_\alpha\rangle$ by

$$M_{x^m y^n} = \sum_\alpha g_\alpha \xi_{\alpha,x}^m \xi_{\alpha,y}^n,$$

with $m = 0, 1, 2$ and $n = 0, 1, 2$. Combining the quantities $\xi_{\alpha,x}^m \xi_{\alpha,y}^n$ is ascending order, the following vector is obtained

$$|m_\alpha\rangle = [\rho, \rho u_x, \rho u_y, \Pi_{xy}, \Pi_{xx}, \Pi_{yy}, Q_{xxy}, Q_{xyy}, A_{xxyy}]^T,$$

given that $Q_{xxx} = \rho u_x$ and $Q_{yyy} = \rho u_y$. Again all moments are given in lattice units with $c = 1$, which is not indicated explicitly. The first six variables have a physical meaning in the hydrodynamic Navier-Stokes model, while the last three entries are the previously mentioned non-hydrodynamic modes. The corresponding transformation matrix then reads

$$\boldsymbol{\mathcal{T}} = [|\mathcal{T}_0\rangle, \ldots, |\mathcal{T}_\alpha\rangle, \cdots, |\mathcal{T}_8\rangle], \tag{4.13}$$

with

$$
\begin{aligned}
|\mathcal{T}_0\rangle &= ||\boldsymbol{\xi}_\alpha|^0\rangle && = [1, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1]^T, \\
|\mathcal{T}_1\rangle &= |\xi_{\alpha x}\rangle && = [0, \quad 1, \quad 0, -1, \quad 0, \quad 1, -1, -1, \quad 1]^T, \\
|\mathcal{T}_2\rangle &= |\xi_{\alpha y}\rangle && = [0, \quad 0, \quad 1, \quad 0, -1, \quad 1, \quad 1, -1, -1]^T, \\
|\mathcal{T}_3\rangle &= |\xi_{\alpha x}\xi_{\alpha y}\rangle && = [0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 1, -1, \quad 1, -1]^T, \\
|\mathcal{T}_4\rangle &= |\xi_{\alpha x}^2\rangle && = [0, \quad 1, \quad 0, \quad 1, \quad 0, \quad 1, \quad 1, \quad 1, \quad 1]^T, \\
|\mathcal{T}_5\rangle &= |\xi_{\alpha y}^2\rangle && = [0, \quad 0, \quad 1, \quad 0, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1]^T, \\
|\mathcal{T}_6\rangle &= |\xi_{\alpha x}^2\xi_{\alpha y}\rangle && = [0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad 1, -1, -1]^T, \\
|\mathcal{T}_7\rangle &= |\xi_{\alpha x}\xi_{\alpha y}^2\rangle && = [0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 1, -1, -1, \quad 1]^T, \\
|\mathcal{T}_8\rangle &= |\xi_{\alpha x}^2\xi_{\alpha y}^2\rangle && = [0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad 1, \quad 1, \quad 1]^T.
\end{aligned}
$$

Physically, the relaxation towards the equilibrium acts on the moments, so that different relaxation parameters can be individually assigned to each moment. Based on these considerations, a more generalized presentation of the LB equations therefore uses a diagonal relaxation matrix instead of a single parameter. The classical BGK model then only constitutes the particular case, in which all entries of the diagonal are $\Delta t/\tau_g$ [D'Humières et al., 2002], i.e.

$$
g_\alpha^{t+\Delta t} = g_\alpha^t - \sum_\beta \boldsymbol{\Lambda}_{\alpha\beta} \left(g_\beta^t - g_\beta^{eq,\,t} \right),
$$

with $\square^{t+\Delta t} = \square\left(\boldsymbol{x} + \boldsymbol{\xi}_\alpha \Delta t, t + \Delta t\right)$, $\square^t = \square\left(\boldsymbol{x}, t\right)$ and

$$
\boldsymbol{\Lambda}_{\alpha\beta} = \frac{\Delta t}{\tau_g} \delta_{\alpha\beta}, \quad \text{where} \quad \beta = 0, .., q-1.
$$

A respective collision in moment space then reads

$$
g_\alpha^{t+\Delta t} = \boldsymbol{\mathcal{T}}^{-1} \left(\underbrace{m_\alpha^t - \sum_\beta \boldsymbol{\Lambda}_{\alpha\beta} \left(m_\beta^t - m_\beta^{eq,\,t} \right)}_{\widehat{m}^t} \right), \tag{4.14}
$$

where $\boldsymbol{\mathcal{T}}^{-1}$ transforms the relaxed moments back into velocity space. It should now become apparent, how the macroscopic solutions of the Navier Stokes solver can be injected into the LBM algorithm. With the density and the velocity, the first three entries of m_α are immediately obtained. Making the same assumption to the second-

order moment as before, i.e. $\boldsymbol{P}' \approx \boldsymbol{P}$, we obtain the moments m_3 to m_5. We recall that the variables being a solution to the Navier-Stokes equations, are the velocity moments of the (space and time continuous) DVBE. Following scaling operation is therefore mandatory before the variables can be inserted into the collision kernel of the (space and time discrete) LBE

$$\boldsymbol{\Pi}^{neq} = \frac{\tau_g}{\tau}\epsilon\boldsymbol{P}^{(1)}.$$

The non-hydrodynamic moments m_{6-8} are obviously not a solution to the Navier-Stokes equations, which makes clear, one more time, the bidirectional mapping problem between the meso- and the macroscopic fluid models. However, now that the collision occurs in moment space, it is possible to individually tune the relaxation time in such a way that the post-collision, non-hydrodynamic moments correspond to their equilibrium state, i.e. $\widehat{m}_{6-8} = m^{eq}_{6-8}$. According to Eq. (4.14), this is the case when we impose the numerical relaxation time $\tilde{\tau}_{6-8} = \tilde{\tau}_{num} = 1$, so that the relaxation matrix reads

$$\boldsymbol{\Lambda}_{\alpha\beta} = diag\left(\tilde{\omega}_g, \tilde{\omega}_g, \tilde{\omega}_g, \tilde{\omega}_g, \tilde{\omega}_g, \tilde{\omega}_g, 1, 1, 1\right).$$

Given that the equilibrium state of any moment can be calculated from ρ and $\boldsymbol{u}$, it is possible to recover the entire set of moments from the hydrodynamic variables. This is schematically presented in the following table.

variables NS solver		moment space		variable LBM solver
ρ, $\boldsymbol{u}$	$\rightarrow$	m_{0-2}, m^{eq}_{6-8}	$\rightarrow$	$\widehat{g}_\alpha$
$\boldsymbol{P}'$	$\overset{scaling}{\rightarrow}$	m_{3-5}		

The above strategy to feed the hydrodynamics variables into the LBM solver does not yet correspond to the novel algorithm that was developed in this study. Instead, it is actually the equivalent of the regularized reconstruction (cf. § 3.3.1.3) in MRT formulation. This becomes evident from the fact that in both approaches, the distribution functions are reconstructed from the variables ρ, $\boldsymbol{u}$, $\boldsymbol{P}^{(0)}$ and $\epsilon\boldsymbol{P}^{(1)}$, while the asymmetric components of $\boldsymbol{P}$ are ignored. Before we proceed to the coupling strategy that was employed here, there are already some general observations to be made. First of all, the distribution functions can only be reconstructed from collided moments, i.e.

$$|\widehat{g}_\alpha\rangle = \boldsymbol{\mathcal{T}}^{-1}|\widehat{m}_\alpha\rangle,$$

because the non-hydrodynamic modes are not known in their pre-collision state. Secondly, in order to reconstruct g_α at the interface, we are obligated to leave the classical BGK framework. As a consequence, the multi-relaxation time model has to be applied

throughout the entire LBM domain, to be congruent with the collision model at the interface.

Despite the often improved stability of the MRT model over a classical BGK collision, it still has its limitations in the low viscosity range. This flaw was attributed by Geier et al. [2006] to an insufficient Galilean invariance and a so-called "crosstalk" between moments. The former can be improved by using central moments, instead of raw moments, which move within the reference frame of the fluid, i.e.

$$\overline{M}_{x^m y^n} = \sum_{\alpha} g_\alpha v_{\alpha,x}^m v_{\alpha,y}^n,$$

where $\overline{\square}$ indicates a central moment and $\boldsymbol{v}_\alpha = [v_{\alpha,x},\, v_{\alpha,y}]^{\mathrm{T}}$ is the peculiar velocity vector with

$$\begin{aligned} v_{\alpha,x} &= \xi_{\alpha,x} - u_x, \\ v_{\alpha,y} &= \xi_{\alpha,y} - u_y. \end{aligned}$$

These central moments are therefore build from raw moments of the same order and lower. For example, $\overline{M}_{xy} = M_{xy} - \rho u_x u_y$. Following this line of reasoning, relaxing a raw moments therefore affects higher-order central moments. It is this crosstalk between the moments that is suggested as a source of instability. Remedy to this problem is provide by relaxing the moments in a cascade from low to high order. The effect of a particular raw moment on a central moment is known and can thus be removed before that higher moment is itself relaxed. Following [Lycett-Brown and Luo, 2014] and [De Rosis, 2016], the generic transformation matrix $\boldsymbol{\mathcal{T}}$ (Eq. (4.13)) is slightly modified such that

$$\begin{aligned} |\mathcal{T}_4^*\rangle = |\xi_{\alpha x}^2 + \xi_{\alpha y}^2\rangle \quad &= [0,\ \ 1,\ \ 1,\ \ 1,\ \ 1,\ \ 2,\ \ 2,\ \ 2,\ \ 2]^T, \\ |\mathcal{T}_5^*\rangle = |\xi_{\alpha x}^2 - \xi_{\alpha y}^2\rangle \quad &= [0,\ \ 1, -1,\ \ 1, -1,\ \ 0,\ \ 0,\ \ 0,\ \ 0]^T, \end{aligned}$$

leading to the trace and the normal difference of the momentum flux tensor $\boldsymbol{\Pi}$. This bears the advantage that the bulk viscosity, acting on the trace of the deviatoric stress tensor (cf. Eq. (2.36)), can be set independently of the kinematic viscosity. In this particular case, it amounts to modifying $\tilde{\omega}_4$. This is of interest, because the bulk viscosity can be used to stabilize the computation.

Let us now introduce the matrix $\overline{\boldsymbol{\mathcal{T}}}$, where the component $|\overline{\mathcal{T}_\alpha}\rangle$ is computed from $|\mathcal{T}_\alpha\rangle$ by substituting for the peculiar velocity $\boldsymbol{v}_\alpha$. In this case $|\overline{\mathcal{T}}_4^*\rangle$ and $|\overline{\mathcal{T}}_5^*\rangle$ constitute the trace and the normal difference of the pressure tensor, respectively. Since the difference in the momentum flux tensor and the pressure tensor is confined to the equilibrium part $(\rho \boldsymbol{u} \otimes \boldsymbol{u})$, the relaxation of the two tensors has the identical effect.

The vector of central moments then reads

$$|\overline{m}_\alpha\rangle = [\overline{m}_0, \ldots, \overline{m}_\alpha, \ldots, \overline{m}_8]$$

with

$$\begin{aligned}
\overline{m}_0 &= \rho, \\
\overline{m}_1 &= 0, \\
\overline{m}_2 &= 0, \\
\overline{m}_3 &= \overline{\Pi}_{xy} && = \Pi_{xy} - \rho u_x u_y, \\
\overline{m}_4 &= \overline{\Pi}_{xx} + \overline{\Pi}_{yy} && = \Pi_{xx} + \Pi_{yy} - \rho\left(u_x^2 + u_y^2\right), \\
\overline{m}_5 &= \overline{\Pi}_{xx} - \overline{\Pi}_{yy} && = \Pi_{xx} - \Pi_{yy} - \rho\left(u_x^2 - u_y^2\right), \\
\overline{m}_6 &= \overline{Q}_{xxy} && = Q_{xxy} - \rho u_x^2 u_y, \\
\overline{m}_7 &= \overline{Q}_{xyy} && = Q_{xyy} - \rho u_x u_y^2, \\
\overline{m}_8 &= \overline{A}_{xxyy} && = A_{xxyy} - \rho u_x^2 u_y^2
\end{aligned}$$

[De Rosis, 2016]. In analogy to the raw moment MRT model, $\overline{m}_{6-8}$ correspond to non-hydrodynamic modes that are relaxed with $\tilde{\tau}_{num} = 1$. The result being that their post-collision state corresponds to the equilibrium state. In order to calculate the equilibrium state of the central moments we use an extended Maxwell distribution

$$g_\alpha^{eq}(\boldsymbol{x}, t) = \rho w_\alpha \left(1 + \frac{\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u}}{2c_0^2} + \frac{(\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u})^2}{4c_0^4} - \frac{u^2}{2c_0^2} + \frac{\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u}}{6c_0^2}\left(\frac{(\boldsymbol{\xi}_\alpha \cdot \boldsymbol{u})^2}{c_0^4} - \frac{3u^2}{c_0^2}\right)\right),$$

which results from a third-order truncation of Eq. (2.17). The attentive reader might find this contradicting to what was said in § 2.2.4. There, it was stated that the order of truncation of the Maxwellian is limited by the number of discrete velocities provided by the lattice. Then it was shown that the D2Q9 lattice only permits a second-order truncation. This is true, if we request that the moments of the discrete and continuous Maxwellian are identical up to the order of truncation. In other words, using a third-order truncation without increasing the number of degrees of freedom (to D2Q17 in that case)) means that continuous and discrete moments remain identical only up to

the second order, i.e.

$$\begin{cases} \boldsymbol{M}^{(0)}: & \sum g_\alpha^{eq}(\boldsymbol{x},t) = \int g^{eq}(\boldsymbol{x},t,\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} \\ \boldsymbol{M}^{(1)}: & \sum \boldsymbol{\xi}_\alpha g_\alpha^{eq}(\boldsymbol{x},t) = \int \boldsymbol{\xi} g^{eq}(\boldsymbol{x},t,\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} \\ \boldsymbol{M}^{(2)}: & \sum \boldsymbol{\xi}_\alpha\boldsymbol{\xi}_\alpha g_\alpha^{eq}(\boldsymbol{x},t) = \int \boldsymbol{\xi}\boldsymbol{\xi} g^{eq}(\boldsymbol{x},t,\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} \\ \boldsymbol{M}^{(3)}: & \underbrace{\sum \boldsymbol{\xi}_\alpha\boldsymbol{\xi}_\alpha\boldsymbol{\xi}_\alpha g_\alpha^{eq}(\boldsymbol{x},t)}_{Q_{ijk}^{eq}} \neq \underbrace{\int \boldsymbol{\xi}\boldsymbol{\xi}\boldsymbol{\xi} g^{eq}(\boldsymbol{x},t,\boldsymbol{\xi})}_{Q_{ijk}^{eq,\mathrm{cont}}}\,\mathrm{d}\boldsymbol{\xi}. \end{cases}$$

In particular, the diagonal entries of the discrete, third-order moment Q_{ijk}^{eq} correspond to the first-order moments ρu_x and ρu_y, since $\xi_{\alpha i}^3 = \xi_{\alpha i}$. In the continuous case, all elements of $Q_{ijk}^{eq,\mathrm{cont}}$ are independent of the inferior moments ($\boldsymbol{M}^{(0)}$-$\boldsymbol{M}^{(3)}$). Using the third-order truncation of the Maxwellian distribution with a D2Q9 lattice, however, correctly represents the off-diagonal elements of Q_{ijk}^{eq}, which has a stabilizing effect on the algorithm as shown in [Dellar, 2014].

Using a third-order truncation of the Maxwell distribution also has the advantage that the the equilibrium states of the non-physical, central moments $\overline{m}_{6-8}^{eq}$ simply become

$$\overline{m}_6^{eq} = 0$$

$$\overline{m}_7^{eq} = 0$$

$$\overline{m}_8^{eq} = -\rho\frac{9u_x^2u_y^2 - 1}{9},$$

which is very convenient.

We finally obtain the vector of post-collision, central moments

$$\widehat{\overline{m}}_\alpha = \left[\rho,\, 0,\, 0,\, \widehat{\overline{m}}_3,\, \widehat{\overline{m}}_4,\, \widehat{\overline{m}}_5,\, 0,\, 0,\, \widehat{\overline{m}}_8\right]$$

As a last step, we solve the system

$$|\widehat{g}_\alpha\rangle = \overline{\boldsymbol{T}}^{*-1}|\widehat{\overline{m}}_\alpha\rangle,$$

where $\overline{\boldsymbol{T}}^*$ is the modified central-moment transformation matrix. The distribution functions for the D2Q9 model are given in appendix B.

4.2.3 The LBM – NS interface

The theory for the communication between the macroscopic and mesoscopic solver being established, we are now having a closer look at the actual coupling. The following consideration is limited to uniform regular grids. While in the HLBM algorithm, the connecting elements were the nodes, here the grid cells were chosen as seen in Fig. 4.13. Of course, this only makes sense, if the grid cells of the connected domains are identical at the interface. The advantage is that the global mesh remains regular. If a

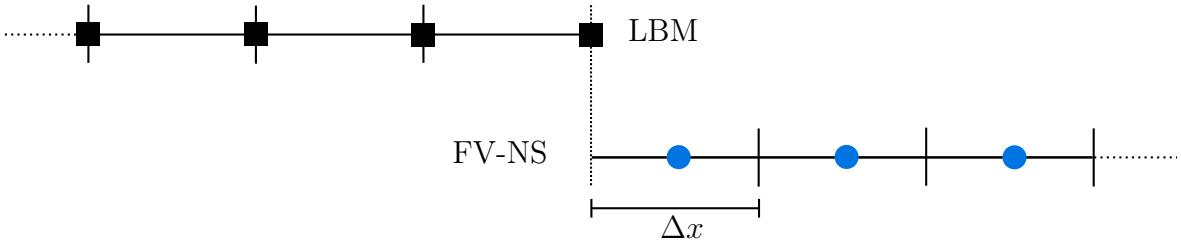


Figure 4.13: Domain linkage via grid cells. Color codes the position of the node: blue=center; black=vertex. Symboles indicate the origin of the data: square = LBM; circle = NS.

node-based and a cell-based approach are combined, as it is the case here, an averaging step, nevertheless, is required to project the information from the cell-vertices onto the cell-centres and vice-versa. Whenever this is necessary in the following algorithm, a simple arithmetic average of the the nearest neighbours is applied

$$\overline{\boldsymbol{\phi}}(\boldsymbol{x},t) = \frac{1}{4}\sum_{i=1}^{2}\sum_{j=1}^{2}\boldsymbol{\phi}(x+\frac{\Delta x}{2}(-1)^i, y+\frac{\Delta x}{2}(-1)^j, t),$$

with $\boldsymbol{\phi} = \{\rho, \boldsymbol{u}, \boldsymbol{P}\}$.

In accordance with § 4.2.2, we begin with the transfer of the hydrodynamic variables from the LBM routine to the NS solver. We recall that due to the 4-step Runge-Kutta scheme of the NS solver, information is required at four intermediate time steps that is not available at the interface with the LB domain. There are two strategies for the coupling of the LB algorithm with the NS solver that basically differ from each other in terms of interface width. A first possibility that manages without overlap area, is the temporal interpolation of the velocity moments in the LB domain for every time instant of the RK cycle. These instants, however, are not clearly defined. A second possibility, which was chosen in this study, does not rely on information from the LB domain during the iteration cycle of the RK scheme. Without update of the boundary nodes, the number of updated data points in the NS domain is therefore reduced by one cell layer after every subiteration as shown by the blank circles in Fig. 4.14. By the end of the iteration cycle, each boundary has four layers of cells that have not been updated.

These areas have to be "refilled" with data from the LB domain, indicated by the blue squares in the right hand sub-figure of Fig. 4.14. At this point, one can conclude that already for the transfer of the hydrodynamic variables onto the Navier-Stokes domain, an overlay area of four grid cells is necessary.

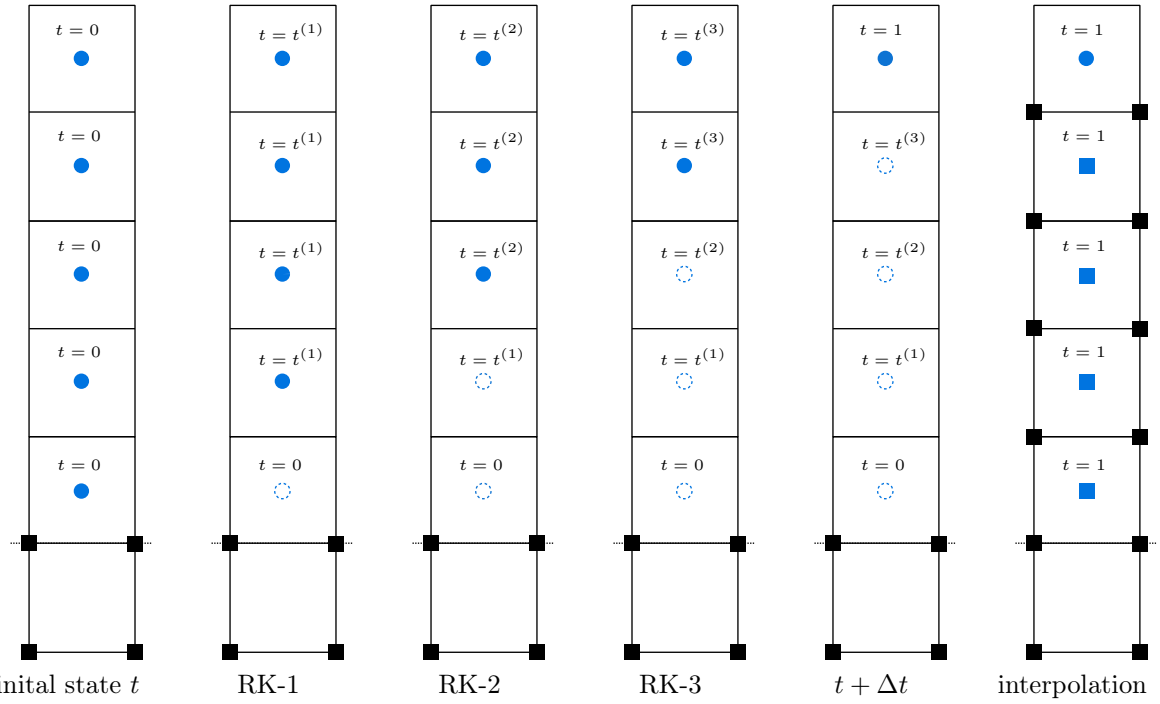


Figure 4.14: Boundary treatment of the finite-volume domain: squares represent data from the LBM domain and circles stand for the cell data of the NS-solver. Blue symboles denote discrete points of the FV mesh and black symbols stand for grid points of the LBM mesh. Empty symbols represent unvalid data.

Transferring data from the FV-NS domain onto the LBM domain only requires two centred nodes in direction perpendicular to the transition. In particular, an interpolation square, indicated by the grey area in Fig. 4.15, is spanned between nodes from the first and the second row of the *trimmed* NS domain (valid nodes at time $t + \Delta t$ in Fig. 4.14). In order to get an idea of the interface zone of a multi-domain approach, i.e. data transfer in both senses, the two transition zones are brought into perspective in Fig 4.15. It becomes clear now that a mutual transfer of hydrodynamic variables between the solvers requires an overlap of at least five grid cells, provided that the Navier-Stokes equations are solved with a 4 step Runge-Kutta scheme.

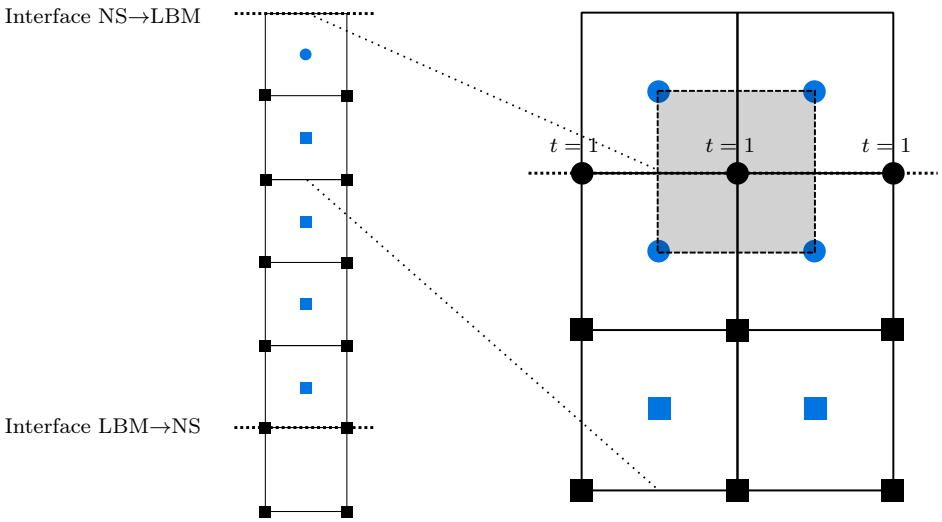


Figure 4.15: Boundary treatment of the LBM domain. The Navier-Stokes domain extends in top-direction and the LBM grid in bottom-direction, accordingly. The horizontal dashed lines frame the overlay area of the HLBM_{NS} algorithm. Colors code the positon: blue = cell-based; black = vertex based. Symbols indicate origin of the data: square = LBM; circle = NS

4.2.4 Preliminary results

The hybrid algorithm was assessed in the light of two test cases, previously studies in § 4.1.3, on a uniform two-dimensional grid :

1. Kelvin-Helmholtz instability of a double shear layer,
2. Advection of a *pseudo-isentropic* vortex.

The initial states are the same as given in Fig. 4.3 and the domain composition is shown in Fig. 4.16.

When referring to the hybrid algorithm that couples the LB method with the Navier-Stokes equations, the respective acronym is labelled by a subscript, i.e. HLBM_{NS}. As the section headline suggests, the results that are presented here are less exhaustive than those obtained for the HLBM. In particular, only the coarse discretization in space is considered with $\Delta x = 0.01$ m, so that the square domain of size 1 m^2 is discretized by $N = 101$ nodes in each dimension. Under a CFL_{LBM} condition of unity imposed by the *stream and collide* algorithm, the time step amounts to $\Delta t = \Delta x/\sqrt{3}c_0 = 2.49 \times 10^{-5}$ s with the speed of sound c_0 fixed at 343.2 m/s in all simulations.

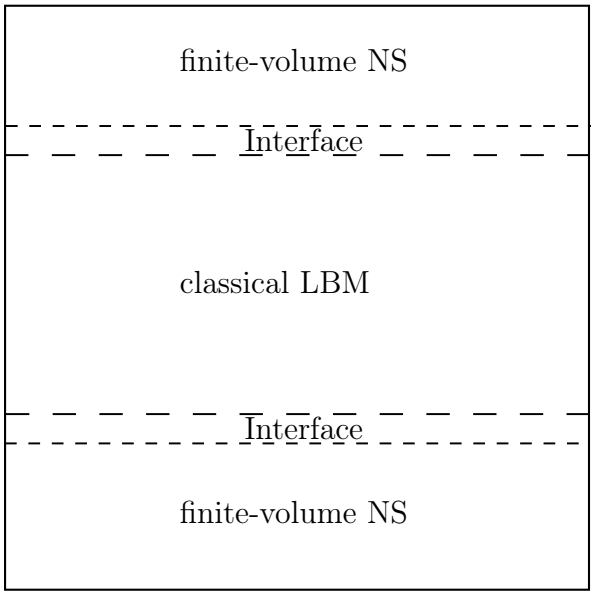


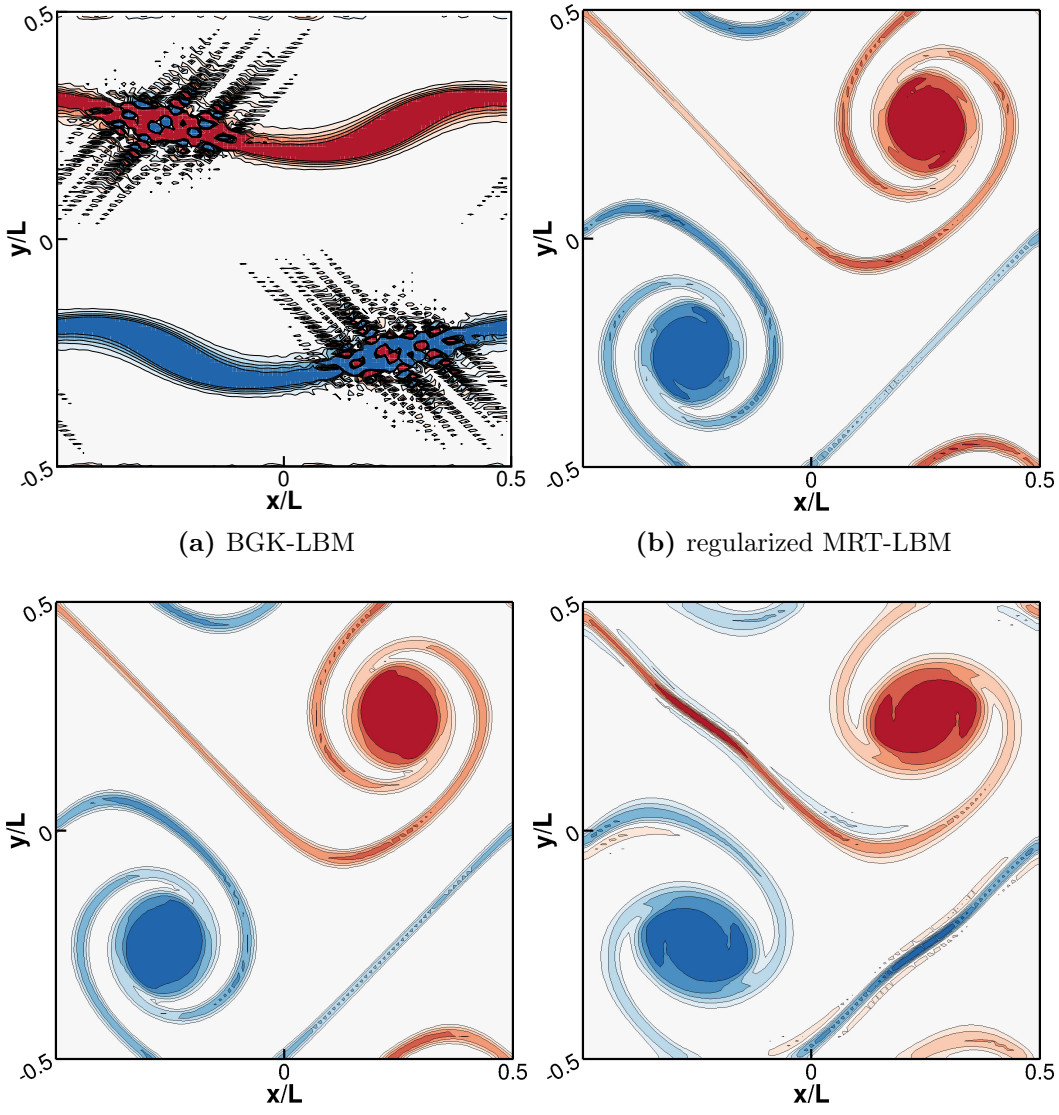
Figure 4.16: Domain configuration for the simulation of the two test cases with the HLBM$_{NS}$ algorithm. The upper and lower regions are linked via the periodicity condition in y-direction and have the same number of points as the central patch.

4.2.4.1 Double shear layer

We begin the evaluation of the HLBM$_{NS}$ with an under-resolved periodic double shear layer (cf. Eq. (4.4)), which was initialized such that the two layers are aligned with the transition interfaces. If the coupling of the two approaches introduces an error, it will immediately trigger instabilities in the highly sensitive shear regions. As in § 4.1.3.1, the shear layer thickness was set to $d_0 = 2.5\Delta x$, over which the velocity in y-direction varied within the range $-0.2c_0 < u_x < 0.2c_0$.

Compared to the previous HLBM, not only the finite-volume algorithm has changed, but also the *stream and collide* algorithm was modified by using a central moment collision model. In a first instance, we therefore consider only the individual algorithms. In addition to the cascaded MRT and the finite-volume Navier Stokes solver, we also ran a simulation with the regularized LBM in moment space. In the absence of numerical instabilities, each computation was stopped after 1400 iterations. The results that are presented in Fig 4.17, also contain the BGK-LBM model as a reference.

As previously discussed, the classical *stream and collide* algorithm with BGK collision operator collapses as soon as the two shear layers begin to roll-up at around $t^* = 858$. In contrast, both MRT models and the Navier-Stokes solver remain stable until the last iteration. The two MRT kinetic schemes do not shown visibly different results. The solution of the NS solver seems minimally less advanced in the roll-up process. This might be related to the slightly higher dissipation that counteracts the Kelvin-Helmholtz instability. We also notice that the vorticity is higher in the free



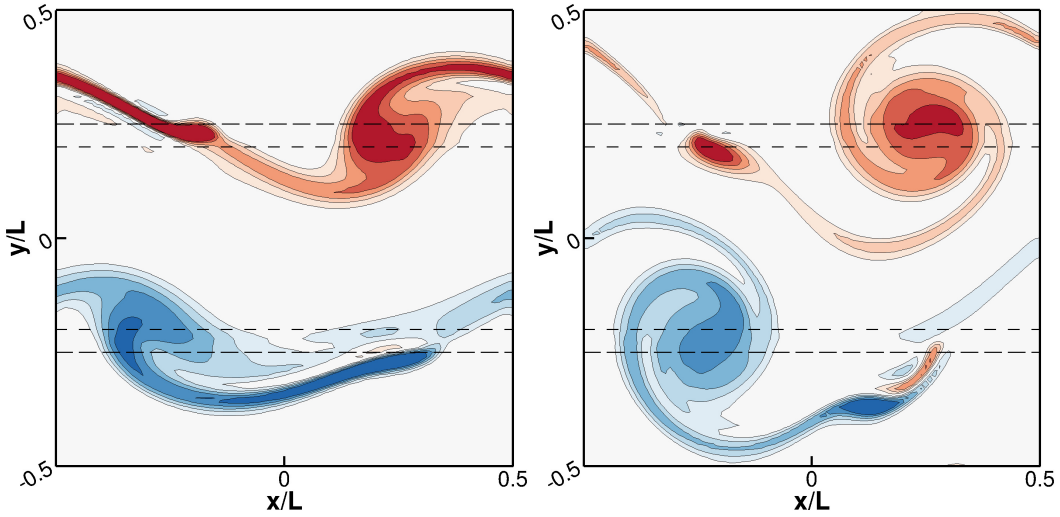
(a) BGK-LBM

(b) regularized MRT-LBM

(c) cascaded MRT-LBM

(d) finite-volume NS

Figure 4.17: Vorticity field in the range $-2100\ \mathrm{s}^{-1} < \omega_z < 2100\ \mathrm{s}^{-1}$ of the double shear layer discretized with $\Delta x = 0.01$ m at $t^* = 858$ (BGK-LBM) and $t^* = 1400$ (regularized MRT-LBM, cascased MRT-LBM, finite-volume NS). All simulations share the same initial conditions.



(a) $t^* = 1070$ **(b)** $t^* = 1400$

Figure 4.18: Vorticity field of the double shear layer obtained from the HLBM$_{NS}$ at two different instants of time. $-2100\ \mathrm{s}^{-1} < \omega_z < 2100\ \mathrm{s}^{-1}$ and $\Delta x = 0.01$ m.

shear layers compared to the two stable lattice Boltzmann algorithms.

This time, the hybrid algorithm (HLBM$_{NS}$) is hence composed of two schemes that are individually stable and produce comparable solutions. The vorticity field of a combined simulation at two instants is shown in Fig. 4.18. The final snapshot (Fig. 4.18b) of the hybrid algorithm indicates that the presence of a coupling interface has a rather deteriorating effect on the solution. The results remain stable, but the vorticity in the vortex cores is weaker compared to the individual results. Moreover, the free shear layers are interrupted and show small vortices at their loose ends. Examining the results at an earlier instant (Fig. 4.18a), we observe that the velocity gradients in the NS domain are stronger than in the LBM domain. This difference probably creates the vortices at the interface that later migrate into the NS domain.

Given that the individual solvers are already stable, it is not possible to attribute a stabilizing effect to the HLBM$_{NS}$, as it was the case for the HLBM. Moreover, the presence of the coupling interface is clearly visible in the results this time. Here, two schemes with comparable dissipative behaviours were combined. The fact, that the HLBM shows better results despite the fact that it combined two schemes with very opposite numerical diffusion, suggests that the problem here arises from the coupling strategy.

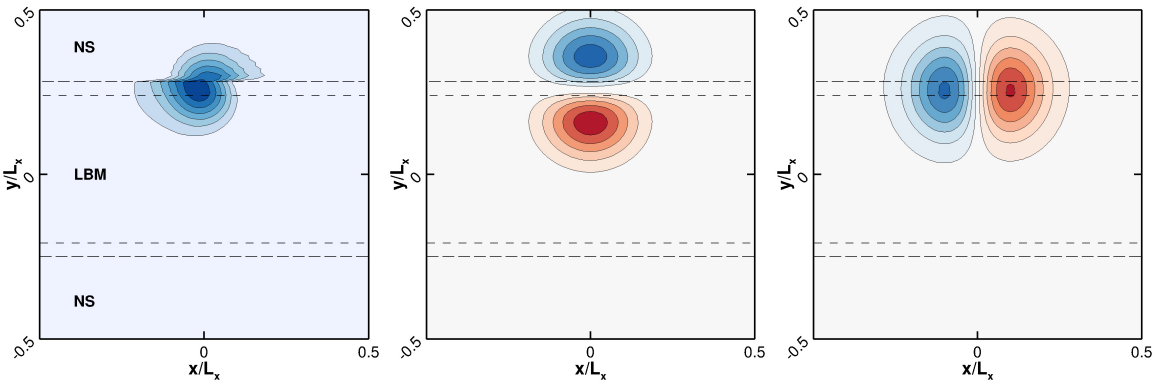
4.2.4.2 Advection of a vortex

The intention of the previous case study was to assess the robustness of the HLBM$_{NS}$ in shear driven flows. Here, we continue the analysis with the advection of a Gaussian

source of vorticity Eq. (3.34) across the interface. The first test was conducted with the exact same parameters that were used for the testing of the HLBM algorithm and are restated in the following

$$R = 0.1 \text{ m}, \quad V_0 = 0.2c_0, \quad \kappa = 0.01, \quad c_0 = 343.2 \text{ m/s}, \quad \rho_0 = 1 \text{ kg/m}^3.$$

Isocontours of the density and the two velocity components are shown in Fig. 4.19. The velocity fields in x- and y-direction appear clean and continuous over the interface. The density field, on the other hand, is quite distorted, which was not the case for the HLBM. We remind that the strength of the vortex is very low. The initial density drop between the far field and the vortex center amounts to $\delta\rho = 1.35 \times 10^4$ kg/m^3, corresponding to a pressure difference of $\delta p = 16$ Pa. The difference between the single isocontours is 2 Pa. Previously, it was argued that this scale is fine enough to reveal anomalies that are relevant to acoustics. In this regard, the current version of the HLBM$_{\text{NS}}$ fails. It should, however, be added that the density field of the vortex is only locally distorted. The vortex recovers its shape after the passage of the transition even though its intensity is weaker (not shown). It should also be mentioned that no spurious noise waves are emitted into the domain. The question that arises now, is

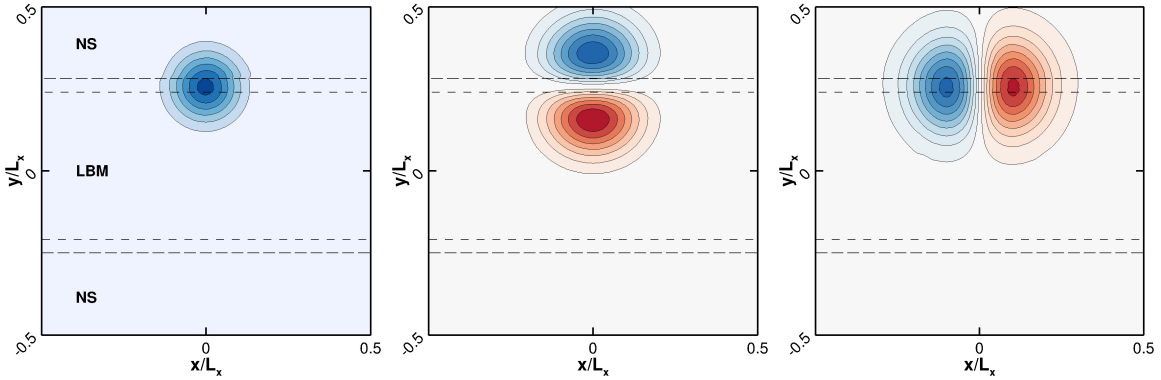


$0.9998 \text{ kg/m}^3 < \rho < 1.0 \text{ kg/m}^3$ $\quad -3.0 \text{ m/s} < u < 3.0 \text{ m/s}$ $\quad 65.0 \text{ m/s} < v < 72.0 \text{ m/s}$

Figure 4.19: Isocontours of the macroscopic variables (ρ, u, v) computed with the hybrid HLBM$_{\text{NS}}$ algorithm on the coarse mesh ($\Delta x^c = 0.01$ m) at $t^* = 220$. The vortex moves vertically with $V_0 = 0.2c_0$ and the circulation parameter is set to $\kappa = 0.01$.

whether this behaviour is also observed when the strength of the vortex is increased. The above presented test case was therefore repeated with the same settings, only that the parameter κ was changed to 0.14. This is the original value used in [Gendre et al., 2017]. The density in the core of the resulting vortex then differs by $\delta\rho = 0.024$ kg/m^3 to the ambient density field. The respective pressure difference is $\delta p = 2827$ Pa. Here, we are clearly in the regime of turbulent pressure fluctuations. The results are shown

in Fig. 4.20. The velocity fields in x- and y-direction resemble that of the weak vortex. This time, the density field captures well the circular shape of the vortex and only a slight distortion is observable. We conclude that the negative effects of the coupling do not scale with the pressure, or the strength of the vortex.



$0.976\ \mathrm{kg/m^3} < \rho < 1.0\ \mathrm{kg/m^3}$ $\quad -50\ \mathrm{m/s} < u < 50\ \mathrm{m/s}$ $\quad 27\ \mathrm{m/s} < v < 115\ \mathrm{m/s}$

Figure 4.20: Isocontours of the macroscopic variables (ρ, u, v) computed with the HLBM$_{\mathrm{NS}}$ algorithm on the coarse mesh ($\Delta x^c = 0.01$ m) at $t^* = 220$. The vortex moves vertically with $V_0 = 0.2c_0$ and the circulation parameter is set to $\kappa = 0.14$.

The previous section revealed that the HLBM shows a very sensitive response to the interface thickness and that a too small interface may even lead to the failure of the algorithm. Here, the code remains stable even at an interface thickness of Δx (in addition to the five grid cells that are always required) but the distortion of the density field of the weak vortex persists – although less pronounced – when the interface is expanded to $> 4\Delta x$ (in addition to the five grid cells that are required anyway). These observations gave rise to a final test, in which the single transitions, i.e. NS $\rightarrow$ LBM and LBM $\rightarrow$ NS, are considered separately in a multi-grid approach and compared to the result of a multi-domain (NS$\leftrightarrow$LBM) approach. Such an isolated consideration allows us to narrow down the potential cause of the distortion. In addition, the mesh size was reduced to $\Delta x = 0.005$ m and the domain was expanded to 1.5 m $\times$ 3 m as shown in Fig. 4.21.

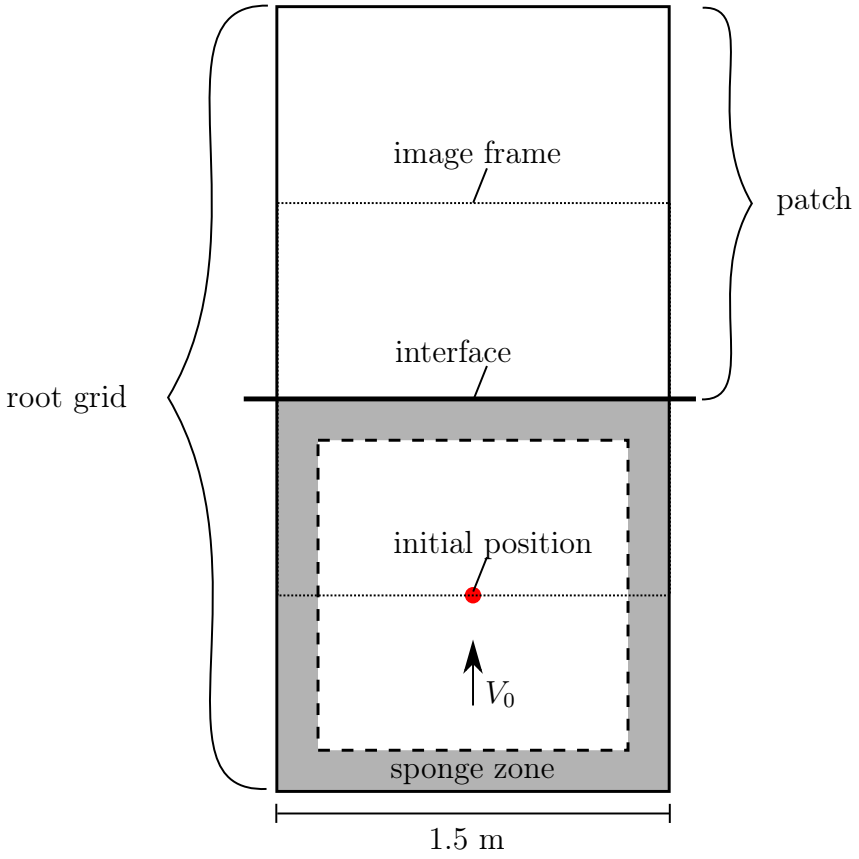


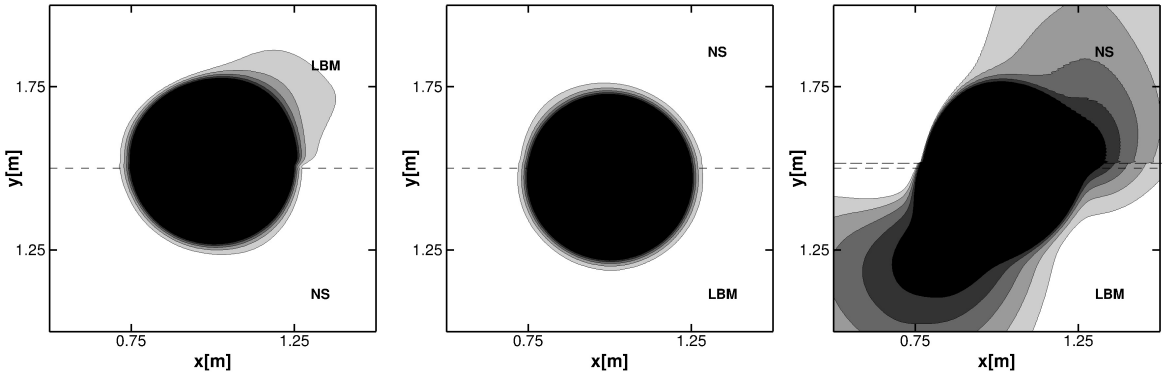
Figure 4.21: Grid strategy for the testing of the HLBM_{NS} algorithm in a multi-grid approach.

A finer mesh is motivated by the fact that in a well-resolved simulation any anomaly must arise from the coupling of the algorithms. The expansion of the domain is necessary to implement sponge zones that absorb the acoustic wave generated at the beginning of the simulation due to an imperfect initialisation. Here the type II absorbing layers [Xu and Sagaut, 2013] with $\chi = 1.999$ are used. The two grids were arranged such that the root grid fills the entire simulation domain with $N_x = 300$ nodes in x-direction and $N_y = 600$ nodes in y-direction. The patch that receives the information from the root grid covers the upper half of the computational domain, so that in this region the two grids are superimposed. The vortex was initialized in the center of the lower half of the computational domain (red dot), at position $x = 0.75$ m and $y = 0.75$ m. The computational settings were adopted from the test case before, i.e.

$$R = 0.1 \text{ m}, \quad V_0 = 0.2c_0, \quad \kappa = 0.14, \quad c_0 = 343.2 \text{ m/s}, \quad \rho_0 = 1 \text{ kg/m}^3.$$

The distance between the initial position of the vortex and the interface is 0.75 m. An acoustic signal travelling at the speed of sound c_0 covers this distance in $t =$

2.19×10^{-3} s. A spatial discretization step of $\Delta x = 0.005$ m gives a time step of $\Delta t = 1.25 \times 10^{-5}$ s. The spurious wave caused by the initialization therefore reaches the interface at about $t^* = 175$ and the vortex itself approximately at $t^* = 875$. The sponge zones were thus deactivated after 300 iterations of the simulation. This left the density field of the vortex unaffected by simultaneously absorbing the spurious wave even in the corners of the domain. Highly saturated isocontours of the pressure fluctuations $p' = \rho' c_0^2$ are illustrated in Fig. 4.22.



(a) NS → LBM **(b)** LBM → NS **(c)** LBM ↔ NS

Figure 4.22: Isocontours of the pressure fluctuations in the range -5 Pa $< p' < 0$ Pa of a pseudo-isentropic vortex crossing a coupling interface. The mesh size is $\Delta x = 0.005$ m and the vortex is advected with $V_0 = 0.2c_0$. The left and central images represent results of the multi-grid approach, where the lower domain is the rootgrid. The right figure shows the results of a multi-domain approach.

In a first setup, which is depicted in Fig. 4.22a, the Navier-Stokes solver is applied to the root grid providing boundary conditions for the LB routine, which is solved on the patch. The transition is clean and no spurious sound emission is visible. The vortex forms an unnatural lobe in the LB domain though. In a second test, the routines are reversed, which is depicted in Fig. 4.22b. The transition is perfectly smooth. When the two algorithms are coupled in a bidirectional way (multi-domain), the previously discussed distortion becomes visible. It should be noted that the pressure field in Fig. 4.22c represents the same test case as the results shown in Fig. 4.20 only at a different scale.

We thus conclude that the coupling of a the isothermal Navier-Stokes equations with the *stream and collide* algorithm in a multi-grid approach shows good results, in particular when the solution passes from the LBM domain to the NS domain. When the routines are coupled in a multi-domain approach, it was observed that the velocity field remains unaffected by the coupling. The density field, however, gets unacceptably distorted. Since the isolated transitions seem to work, there must be some kind of feedback when both transitions are active, which is not alleviated by increasing the

interface width.

4.3 Summary

In this chapter, two hybrid algorithms are presented that combine the classical *stream and collide* algorithm with a finite-volume formulation of the lattice Boltzmann equation (HLBM) and the Navier-Stokes equations (HLBM_{NS}), in a domain decomposition approach. Hybrid lattice Boltzmann methods respond to certain shortcomings of the classical lattice Boltzmann method. In particular, they address the matter of mesh refinement.

The classical multi-scale lattice Boltzmann scheme decomposes the computational domain into uniform, Cartesian subdomains of different resolution. In each subdomain, the *stream and collide* algorithm is solved. Owing to connectivity, the resolution factor between adjacent subdomains is two, or a multiple of two (quadtree/octree grids in 2D/3D). Such a technique introduces an abrupt change in the space-time discretization step, which makes the information transfer between the subdomains an intricate matter (spatial and temporal interpolation, etc.). Moreover, the sudden change in resolution can lead to undesired noise emission and instabilities.

Hybrid algorithms also rely on a domain decomposition. Here, the idea is to solve different algorithms in the subdomains that arise from different discretization techniques of the DVBE (HLBM), or different fluid models (HLBM_{NS}). Solving the classical *stream and collide* algorithm, we are confined to uniform Cartesian grids imposed by the lattice. Using a finite-volume algorithm (FV-LBM or FV-NS), these constraints do not apply accepting, nevertheless, a loss in efficiency. Coupling the two algorithms, the change in resolution is not any more confined to the interface but can take place anywhere in the finite-volume domain. Subdomains can thus be connected by matching the spatial resolution at the interface and allowing a smooth change of the spatial resolution in the finite-volume domain.

The feasibility of the HLBM and the HLBM_{NS} algorithm was tested in this chapter. Former was evaluated in the course of three different case studies including gradual mesh refinement. The results show that the HLBM combines the positive features of the individual algorithms without undesirable side effects. It therefore constitutes a valuable alternative to classical multi-scale lattice Boltzmann schemes.

The HLBM_{NS} algorithm is more intricate than the HLBM algorithm, because it requires a coupling in moment space. This, on the other hand, widens the spectrum of possible coupling strategies. Here, a novel approach was tested that uses a multi-relaxation time, central moment collision to integrate the hydrodynamic solutions into the kinetic Boltzmann scheme. The HLBM_{NS} was only tested on uniform Cartesian grids. Qualitative results of two different test-cases indicate that the algorithm is still in a premature state. While the algorithm is stable, the presence of the hybrid interface

is usually visible in the pressure/density field as well as in the velocity/vorticity field.

Chapter 5

Conclusion and Perspective

The present work discusses the lattice Boltzmann method for non-uniform grids. Evolved from lattice gas automata, this method is usually associated with the classical *stream and collide* formulation of discrete distribution functions on a defined lattice. Considered as the probability to find, at a given time instant t and position in space $\boldsymbol{x}$, a particle with discrete velocity $\boldsymbol{\xi}_\alpha$, i.e. $f(\boldsymbol{x}, \boldsymbol{\xi}_\alpha, t)$, these populations are streamed between nodes of a lattice according to their assigned molecular velocities. At the nodes, collision between the populations takes place, which is usually modelled as a relaxation towards a thermodynamic equilibrium described by the Maxwell distribution.

Mathematically, the streaming step describes an exact solution of the advection term of the discrete velocity Boltzmann equation, expressed as a Lagrangian derivative along the discrete velocity vector. This is possible, because the propagation velocity $\boldsymbol{\xi}_\alpha$ in discrete velocity space is constant, so that the advection term in linear. As a consequence, numerical dissipation in this approach is extremely low and confined to the local collision operator. A further advantage of the streaming step resides in its simplicity of numerical implementation. The scattering of the post-collision state to the neighbouring nodes simply corresponds to an index shift in memory, which makes this method also extremely efficient.

Low numerical dissipation and high efficiency, however, come at a price. Inevitable to the *stream and collide* formulation is the coupling of physical and velocity space. The advection in discrete phase space dictates that during the time interval Δt, the populations cover exactly the distance between two adjacent lattice nodes defined by $\Delta x = \boldsymbol{\xi}_\alpha \Delta t$. Discrete space points are thus confined to the lattice nodes, which produces a regular Cartesian grid. While it makes the meshing of complex geometries straightforward, the classical LB method is not well suited to body-fitted coordinates and adaptive time-stepping. In addition, mesh refinement is restricted to quadtrees in 2D and octrees in 3D, where the mesh size Δx changes by a factor of two in each dimension. Mesh transitions are therefore delicate regions, because here, two different

solutions are matched over a very small interface area. The miss-match between the two solutions amounts to $(1-0.5^{\mathcal{O}})E_{al}$, where E_{al} is the total error between the coarse and the analytical solution and $\mathcal{O}$ denotes the global order of the scheme, which is two for the classical LB method. Usually, mesh refinement is achieved with a redundant overlay region. At the edges parallel to the transition, the two different solutions are mutually imposed on the respective other grid. In this study, it was indicated that in a particular case of mesh refinement (partial reconstruction), the difference in the solutions is sufficient to trigger instabilities that lead to the failure of the method at moderate Mach numbers ($\approx$ 0.3 M). As a consequence, within the framework of the classical LB method, it was suggested to reduce the overlay to zero (direct matching of the resolution domains) and establish an intermediate solution in the transition nodes that converges the different solutions of two resolution domains. This idea, however, was not further pursued in this study. Instead, the subject of mesh refinement was tackled at its origin: is there a way to avoid octree grids by simultaneously preserving the previously discussed benefits of the LB method?

The only way to avoid an octree data structure is to discretize physical space and velocity space independently. Otherwise the spatial mesh is imposed by the lattice. This is achieved by discretizing the advection term in the Eulerian framework, such that the space and time derivatives are treated separately. Of course, certain advantages of the classical LBM are compromised. First, the advection term is no longer solved in an exact manner, which raises the level of numerical dissipation. Secondly, the efficiency of the streaming that translates into a simple index shift in memory is lost, due to the evaluation of an additional flux term.

In conclusion, both, the Lagrangian (*stream and collide*) the Eulerian approach have a spot of bother. While the classical *stream and collide* algorithm is little dissipative and highly efficient, it is restricted to Cartesian uniform grids. Choosing a discretization in the Eulerian sense increases numerical dissipation and lowers efficiency, but physical space may be discretized in any arbitrary way (e.g. unstructured, gradually refined, body-fitted grids).

In this work, the two approaches were coupled in a hybrid lattice Boltzmann method with the intention to combine their positive features in a single algorithm. This idea was evoked simultaneously in a study that coupled the classical *stream and collide* algorithm with a first-order unstructured finite-volume LB formulation [Di Ilio et al., 2017]. Here, important improvement to the pioneering work were achieved. Choosing the finite-volume approach, too, the collision term in this study was treated semi-implicitly to construct a second-order FV-LBM algorithm that agrees with the order of accuracy of the classical LBM. It was shown that the resulting HLBM algorithm maintains a global second-order accuracy. Moreover, a third-order upwind approximation in space of the surface fluxes allows for a CFL number of unity so that the two algorithms were coupled without sub-iteration and temporal interpolation, which was a highly

desirable feature. The discretization of space into finite volumes was achieved on a Cartesian grid and the cell size and geometry at the interface of the LBM domain and FV-LBM domain are perfectly matched. In doing so, we address the issue of solution miss-match at the refinement interface of the classical LB method using octrees. The solution of the two domains are identically resolved at the interface, which enables a smooth passage. The finite-volume domain can then be gradually refined away from the interface, which was demonstrated for an axial jet. The two methods are linked using a domain decomposition technique, where the thickness of the interface turned out to play a crucial role. The minimum required to yield stable results has to be at least $2\Delta x$. We believe this buffer is necessary to avoid a feedback between adjacent transition nodes that immediately leads to an instability.

The proposed algorithm was assessed in terms of stability, numerical error and potential applications within the scope of three standard test cases for periodic boundary conditions. Outside the stability limit of the classical LBM with BGK collision operator the coupling can have a stabilizing effect, due to the dissipation introduced with the finite-volume LB scheme. Concerning the first-order moment of the distribution functions, the interface does not contribute to the error and the results from the HLBM scheme lie well within the error margins spanned by the two individual schemes. In one case, the error in the density was influenced by the presence of the interface. This occurred when a vortex crossed the transition between the algorithms, with the classical LBM upstream and the FV-LBM downstream of the interface. It was argued that the abrupt increase in dissipation can equally be obtained by a transition towards a coarse grid, which is knowingly accompanied by the reflection of non-supported structures back into the domain. In the last test case, we used gradual mesh refinement to demonstrate the advantages of the HLBM. Its stabilizing effect compared to the individual LBM algorithm was confirmed and a reduction in CPU compared to the FV-LBM could be demonstrated.

The hybrid LBM constitutes an interesting numerical tool. It has the ability to adapt to anisotropic flows by simultaneously using the efficient *stream and collide* algorithm in far-field regions. This is of particular interest for aeroacoustic simulations. The low dissipation and high efficiency of the classical LB method are ideally suited to transport acoustic signals over long distances. A hurdle is usually the abrupt mesh refinement by a factor of two. Using a HLBM approach, it would be possible to resolve the hydrodynamics around and object on a body-fitted unstructured grid that is progressively coarsened from $\Delta x \propto \boldsymbol{u}/f$ to $\Delta x_{ac} \propto c_0/f$, for a given frequency f. At the domain boundary, data would be transferred to a uniform Cartesian regular grid of grid size Δx_{ac}, allowing a smooth transition and efficient propagation of the aeroacoustic signals. Further development thus implies the coupling with an unstructured FV formulation and the implementation of solid boundaries to test the performance in wall-bounded flows. Here the matching with a FV formulation in near-wall regions

will allow us to benefit from the armada of numerical methods for the treatment of wall-boundary modelling.

Another hybrid algorithm was presented that couples the *stream and collide* algorithm with the isothermal Navier-Stokes equations in finite-volume formulation (FV-NS). This algorithm is referred to as HLBM_{NS}. Clearly outside the framework of a hybrid kinetic scheme (coupling of two different space-time discretization of the discrete velocity Boltzmann equations), the main difference between the HLBM and the HLBM_{NS} manifests in a single dimensionless parameter ϵ, called the Knudsen number. In the beginning of this manuscript, we demonstrate that the Navier-Stokes equations can be related to the lattice Boltzmann equation using a multi timescale analysis. The scales are separated by different orders of the Knudsen number. Hence, increasing orders in ϵ yield the Euler equations, Navier-Stokes equations, etc, respectively. A finite-volume formulation of the isothermal Navier-Stokes equations may therefore be interpreted as a moment-representation of the FV-LBM at second order in the Knudsen number, by disregarding some errors of $\mathcal{O}(\text{Ma}^3)$. The advantage lies in the number of variables in each node. In two dimensions, the FV-LBM solves for nine distribution functions, whereas the isothermal FV-NS method only solves for three macroscopic variables (ρ, u_x, u_y) in addition to a finite-difference approximation of the deviatoric stress tensor. Logically, the coupling of the two algorithms is achieved in moment space, which constitutes the main difference to the HLBM. The bilateral communication between the mesoscopic and the macroscopic solver is established through a collision in moment space of the classical LBM. The Navier-Stokes equations are solved with a fourth-order Runge-Kutta scheme that is stable at a CFL of unity. Using central differences to approximate the surface fluxes, the numerical dissipation of the scheme is very low, in the range of the *stream and collide* algorithm, which contrasts the behaviour of the FV-LBM. The performance of the HLBM_{NS} was tested in two different case studies on a uniform grid. The hybrid algorithm shows a similar stability in under-resolved, shear-driven flows, than the two individual algorithms. The advection of a Gaussian source of vorticity over a bilateral coupling interface, however, revealed a distortion of the density field. This error is considerably bigger then the one observed for the HLBM. An isolated consideration of the two coupling strategies (LBM $\rightarrow$ NS and NS $\rightarrow$ LBM) showed good results. It was therefore concluded that the problem arises from a feedback effect. Even though, the dissipative behaviour of the two approaches is much more comparable than that of the two schemes in the HLBM algorithm, their solutions can still be different. We recall that the LBM contains a $\mathcal{O}(\text{M}^3)$-error that arises from the discretization of velocity-space. The Navier-Stokes model does not have this error term and is applicable to compressible flows. It is hence possible that during every iteration, a slightly different solution as "expected" is mutually imposed in the boundary nodes of the overlay region. The error introduced might accumulate in a feedback loop as indicated in Fig. 5.1.

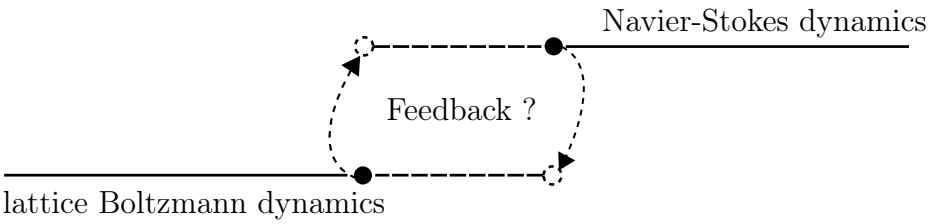


Figure 5.1: Possible explanation for the failure of the HLBM$_{NS}$ algorithm with bi-directional coupling.

We nevertheless stress that the "feedback effect" is not significantly lowered by increasing the interface width, which would be naturally expected. The HLBM$_{\mathrm{NS}}$ that was presented here is therefore still at a premature state. Further development is required before it can be applied to refined grids, where it should unfold its full potential. We are, nevertheless, convinced that the HLBM$_{\mathrm{NS}}$ can achieve the same quality as the HLBM.

In the introduction of the present manuscript, the hybrid lattice Boltzmann approach was suggested for aeroacoustic simulations, where the classical *stream and collide* algorithm is only applied in the periphery to transport the aeroacoustic signal at low cost and little dissipation. The discretization of the region around the geometry then relies on classical (Eulerian) schemes that allow for body-fitted, gradually refined, unstructured grids. Our results strongly encourage the development of such solvers as an alternative to multi-scale lattice Boltzmann schemes.

Appendix A

Relations between pre- and post-collision states

Scaling the mesh specific distribution functions g, or applying the change of variable between g and the genuine functions f, is usually achieved in their pre-collision state. Sometimes, however, a distribution function is required in its post-collision state. It also happens that only the post-collision state is available to do the conversion with. Here, some important relations that occur during the manuscript, are derived.

A.1 Rescaled post-collision state

The DUGK scheme uses a scaling operation that relates a pre-collision default (coarse) population g to a post-collision fine population $\widehat{g}^f$ (Eq. (3.17)). Here, it is demonstrated how this relation is obtained. First, we recall two important relations for a mesh refinement factor of two that were derived in § 3.3 and will be relevant here.

$$g^{neq,f} = \frac{\tilde{\tau}_g^f}{2\tilde{\tau}_g^c} g^{neq,c} \tag{A.1}$$

$$\tilde{\tau}_g^f = 2\tilde{\tau}_g^c - \frac{1}{2} \quad \text{cf. Eq. (3.24) with } m = 2 \tag{A.2}$$

Then, we restate the collision of g^f and substitute for the above expressions, to arrive at Eq. (3.17).

$$\begin{aligned}
\widehat{g}^f &= g^f - \frac{1}{\tilde{\tau}_g^f} g^{neq,f} \\
&= g^{eq} + g^{neq,f} - \frac{1}{\tilde{\tau}_g^f} g^{neq,f} \\
&= g^{eq} + \frac{\tilde{\tau}_g^f}{2\tilde{\tau}_g^c} g^{neq,c} - \frac{1}{\tilde{\tau}_g^f} \frac{\tilde{\tau}_g^f}{2\tilde{\tau}_g^c} g^{neq,c} && \text{using (A.1)} \\
&= g^{eq} + \frac{\tilde{\tau}_g^f - 1}{2\tilde{\tau}_g^c} g^{neq,c} \\
&= g^{eq} + \frac{2\tilde{\tau}_g^c - \frac{3}{2}}{2\tilde{\tau}_g^c} g^{neq,c} && \text{using (A.2)} \\
&= g^{eq} + \frac{4\tilde{\tau}_g^c - 3}{4\tilde{\tau}_g^c} \left(g^c - g^{eq}\right) \\
&= \frac{4\tilde{\tau}_g^c - 3}{4\tilde{\tau}_g^c} g^c + \frac{3}{4\tilde{\tau}_g^c}
\end{aligned} \tag{A.3}$$

A.2 Change of variable with post-collision state

The flux term in the finite-volume scheme contains the genuine distribution functions f. The solutions to the FV-LB method, however, are the mesh specific functions g. Naturally, f is replaced by g according to the change of variable that was introduced in § 3.1, Eq. (3.9). When using the Heun-predictor scheme in the HLBM, it becomes, nevertheless, convenient to express the genuine populations f as a function of the post-collision state of g, i.e. $\widehat{g}$ (cf. Eq. (4.1)). A derivation is provided in the following. Again, we restate two important relations that were established previously in the manuscript and required here.

$$f = g - \frac{\Delta t}{2\tau_g} \left(g - g^{eq}\right), \tag{A.4}$$

$$\widehat{g} = g - \frac{\Delta t}{\tau_g} \left(g - g^{eq}\right). \tag{A.5}$$

Combining Eqs. (A.4) and (A.5), following equation is obtained for f

$$f = \widehat{g} + \frac{\Delta t}{2\tau_g} \left(g - g^{eq}\right) \tag{A.6}$$

Now, g in the above equation has to be expressed as a function of $\widehat{g}$. Using Eq. (A.5), we obtain

$$\widehat{g} = \frac{\tau_g - \Delta t}{\tau_g} g + \frac{\Delta t}{\tau_g} g^{eq} \tag{A.7}$$

$$\begin{aligned} g &= \frac{\tau_g}{\tau_g - \Delta t} \left(\widehat{g} - \frac{\Delta t}{\tau_g} g^{eq} \right) \\ &= \frac{\tau_g}{\tau_g - \Delta t} \widehat{g} - \frac{\Delta t}{\tau_g - \Delta t} g^{eq} \end{aligned} \tag{A.8}$$

Substituting for g now yields

$$\begin{aligned} f &= \widehat{g} + \frac{\Delta t}{2\tau_g} \left(\frac{\tau_g}{\tau_g - \Delta t} \widehat{g} - \frac{\Delta t}{\tau_g - \Delta t} g^{eq} - g^{eq} \right) \\ &= \widehat{g} + \frac{\Delta t}{2\tau_g} \left(\frac{\tau_g}{\tau_g - \Delta t} \widehat{g} - \frac{\tau_g}{\tau_g - \Delta t} g^{eq} \right) \\ &= \widehat{g} + \frac{\Delta t}{2 \left(\tau_g - \Delta t \right)} \left(\widehat{g} - g^{eq} \right) \\ &= \widehat{g} + \frac{1}{2 \left(\tilde{\tau}_g - 1 \right)} \left(\widehat{g} - g^{eq} \right) \\ &= \frac{1}{2 \left(\tilde{\tau}_g - 1 \right)} \left[\left(2\tilde{\tau}_g - 1 \right) \widehat{g} - g^{eq} \right] \end{aligned} \tag{A.9}$$

Appendix B

Central moment reconstruction of $\widehat{g}$

Here, the post-collision states of the distribution functions $\widehat{g}_\alpha$ are presented that are obtained by solving

$$|\widehat{g}_\alpha\rangle = \boldsymbol{\mathcal{T}}^{*-1}|\widehat{\overline{m}}_\alpha\rangle, \tag{B.1}$$

where $|\widehat{\overline{m}}_\alpha\rangle$ denotes the central moment, post-collision vector

$$\widehat{\overline{m}}_\alpha = \left[\rho,\, 0,\, 0,\, \widehat{\overline{m}}_3,\, \widehat{\overline{m}}_4,\, \widehat{\overline{m}}_5,\, \overline{m}_6^{eq},\, \overline{m}_7^{eq},\, \overline{m}_8^{eq}\right]. \tag{B.2}$$

with

$$\overline{m}_6^{eq} = 0$$

$$\overline{m}_7^{eq} = 0$$

$$\overline{m}_8^{eq} = -\rho\frac{9u_x^2u_y^2 - 1}{9}.$$

The distributions then read

$$\widehat{g}_0 = \rho\left(\frac{10}{9} - u_x^2 - u_y^2\right) + \widehat{\overline{m}}_3 4u_xu_y + \widehat{\overline{m}}_4\left(\frac{u_x^2u_y^2}{2} - 1\right) + \widehat{\overline{m}}_5\left(\frac{u_x^2u_y^2}{2}\right) \tag{B.3}$$

$$\begin{aligned}\widehat{g}_1 = &\frac{\rho}{2}\left(-\frac{1}{9} + u_x + u_x^2 - u_xu_y^2\right) - \widehat{\overline{m}}_3\left(u_y + 2u_xu_y\right)\\ &+ \frac{\widehat{\overline{m}}_4}{4}\left(1 - u_x - u_x^2 - u_y^2\right) + \frac{\widehat{\overline{m}}_5}{4}\left(1 + u_x + u_x^2 - u_y^2\right)\end{aligned} \tag{B.4}$$

$$\widehat{g}_2 = \frac{\rho}{2}\left(-\frac{1}{9} + u_y + u_y^2 - u_x^2 u_y\right) - \widehat{\overline{m}}_3\left(u_x + 2u_x u_y\right) + \frac{\widehat{\overline{m}}_4}{4}\left(1 - u_y - u_y^2 - u_x^2\right) - \frac{\widehat{\overline{m}}_5}{4}\left(1 + u_y - u_x^2 + u_y^2\right) \tag{B.5}$$

$$\widehat{g}_3 = \frac{\rho}{2}\left(-\frac{1}{9} - u_x + u_x^2 + u_x u_y^2\right) - \widehat{\overline{m}}_3\left(u_y - 2u_x u_y\right) + \frac{\widehat{\overline{m}}_4}{4}\left(1 + u_x - u_x^2 - u_y^2\right) + \frac{\widehat{\overline{m}}_5}{4}\left(1 - u_x + u_x^2 - u_y^2\right) \tag{B.6}$$

$$\widehat{g}_4 = \frac{\rho}{2}\left(-\frac{1}{9} - u_y + u_y^2 - u_x^2 u_y\right) - \widehat{\overline{m}}_3\left(u_x - 2u_x u_y\right) + \frac{\widehat{\overline{m}}_4}{4}\left(1 + u_y - u_y^2 - u_x^2\right) - \frac{\widehat{\overline{m}}_5}{4}\left(1 - u_y - u_x^2 + u_y^2\right) \tag{B.7}$$

$$\widehat{g}_5 = \frac{\rho}{4}\left(\frac{1}{9} + u_x u_y + u_x u_y^2 + u_x^2 u_y\right) + \frac{\widehat{\overline{m}}_3}{2}\left(\frac{1}{2}u_x + u_y + u_x u_y\right) + \frac{\widehat{\overline{m}}_4}{8}\left(u_x + u_y + u_x^2 + u_y^2\right) - \frac{\widehat{\overline{m}}_5}{8}\left(u_x - u_y + u_x^2 - u_y^2\right) \tag{B.8}$$

$$\widehat{g}_6 = \frac{\rho}{4}\left(\frac{1}{9} - u_x u_y - u_x u_y^2 + u_x^2 u_y\right) - \frac{\widehat{\overline{m}}_3}{2}\left(\frac{1}{2}u_x + u_y - u_x u_y\right) - \frac{\widehat{\overline{m}}_4}{8}\left(u_x - u_y - u_x^2 - u_y^2\right) + \frac{\widehat{\overline{m}}_5}{8}\left(u_x + u_y - u_x^2 + u_y^2\right) \tag{B.9}$$

$$\widehat{g}_7 = \frac{\rho}{4}\left(\frac{1}{9} + u_x u_y - u_x u_y^2 - u_x^2 u_y\right) + \frac{\widehat{\overline{m}}_3}{2}\left(\frac{1}{2} - u_x - u_y + u_x u_y\right) - \frac{\widehat{\overline{m}}_4}{8}\left(u_x + u_y - u_x^2 - u_y^2\right) + \frac{\widehat{\overline{m}}_5}{8}\left(u_x - u_y - u_x^2 + u_y^2\right) \tag{B.10}$$

$$\widehat{g}_8 = \frac{\rho}{4}\left(\frac{1}{9} - u_x u_y + u_x u_y^2 - u_x^2 u_y\right) - \frac{\widehat{\overline{m}}_3}{2}\left(\frac{1}{2} + u_x - u_y - u_x u_y\right) + \frac{\widehat{\overline{m}}_4}{8}\left(u_x + u_y - u_x^2 + u_y^2\right) - \frac{\widehat{\overline{m}}_5}{8}\left(u_x + u_y + u_x^2 - u_y^2\right) \tag{B.11}$$

Appendix C

Hermite polynomials

The Rodgigues' fomula defines a Hermite polynomial of order n as

$$\mathscr{H}^{(n)}(\boldsymbol{x}) = \frac{(-1)^n}{w(\boldsymbol{x})}\nabla^n w(\boldsymbol{x}), \tag{C.1}$$

where w denotes the weighting function defined in Eq.(2.16). The polynomials for $n \leq 3$ are then given as

$$\mathscr{H}^{(0)}(\boldsymbol{x}) = 1 \tag{C.2a}$$
$$\mathscr{H}^{(1)}(\boldsymbol{x}) = x_i \tag{C.2b}$$
$$\mathscr{H}^{(2)}(\boldsymbol{x}) = x_i x_j - \delta_{ij} \tag{C.2c}$$
$$\mathscr{H}^{(3)}(\boldsymbol{x}) = x_i x_j x_k - (\delta_{ij}x_k + \delta_{ik}x_j + \delta_{jk}x_i) \tag{C.2d}$$

[Grad, 1949].

Bibliography

T. Abe. Derivation of the Lattice Boltzmann Method by Means of the Discrete Ordinate Method for the Boltzmann Equation. *Journal of Computational Physics*, 131:241–246, 1997.

A. Bardow, I. V. Karlin, and A. A. Gusev. General characteristic-based algorithm for off-lattice Boltzmann simulations. *Europhysics Letters (EPL)*, 75(3):434–440, 2006.

A. Bardow, I. V. Karlin, and A. A. Gusev. Multispeed models in off-lattice Boltzmann simulations. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 77 (2):1–4, 2008.

P. L. Bhatnagar, E. P. Gross, and M. Krook. A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems. *Physical Review*, 94(3):511–525, 1954.

J. F. Bourgat, P. Le Tallec, and M. D. Tidriri. Coupling Boltzmann and Navier – Stokes equations by friction. *J. Comp Physics*, 245:227–245, 1996.

N. Cao, S. Chen, S. Jin, and D. Martínez. Physical symmetry and lattice symmetry in the lattice Boltzmann method. *Physical Review E*, 55(1):21–24, 1997.

H. Chen. Volumetric formulation of the lattice Boltzmann method for fluid dynamics: Basic concept. *Physical Review E*, 58(3):3955–3963, 1998.

H. Chen, S. Chen, and W. H. Matthaeus. Recovery of the Navier-Stokes equations using a lattice-gas Soltzmann method. *Physical Review A*, 45(8):5339–5342, 1992.

H. Chen, O. Filippova, J. Hoch, K. Molvig, R. Shock, C. Teixeira, and R. Zhang. Grid refinement in lattice Boltzmann methods based on volumetric formulation. *Physica A: Statistical Mechanics and its Applications*, 362(1):158–167, 2006.

B. Crouse. *Lattice-Boltzmann Strömungssimulationen auf Baumdatenstrukturen*. PhD thesis, Technische Universität München, 2003.

A. De Rosis. Non-orthogonal central moments relaxing to a discrete equilibrium: A D2Q9 lattice Boltzmann model. *EPL (Europhysics Letters)*, 116(4):44003, 2016.

P. J. Dellar. Bulk and shear viscosities in lattice Boltzmann equations. *Pysical Review E*, 2001.

P. J. Dellar. Non-hydrodynamic modes and a priori construction of shallow water lattice Boltzmann equations. *Physical Review E*, 65:1–12, 2002.

P. J. Dellar. Incompressible limits of lattice Boltzmann equations using multiple relaxation times. *Journal of Computational Physics*, 190:351–370, 2003.

P. J. Dellar. Lattice Boltzmann algorithms without cubic defects in Galilean invariance on standard lattices. *Journal of Computational Physics*, 259, 2014.

D. D'Humières. Generalized Lattice-Boltzmann Equations. *Progress in Astronautics and Aeronautics*, 159:450–458, 1992.

D. D'Humières, I. Ginzburg, M. Krafczyk, P. Lallemand, and L.-s. Luo. Multiple-relaxatio-time lattice Boltzmann models in three dimensions. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 360(1792):437–451, 2002.

G. Di Ilio, D. Chiappini, S. Ubertini, G. Bella, and S. Succi. Hybrid Lattice Boltzmann Method on overlapping grids. *Physical Review E*, 013309:1–15, 2017.

F. Drui. *Modélisation et simulation Eulériennes des écoulements diphasiques à phases séparées et dispersées : développement d ' une modélisation unifiée et de méthodes numériques adaptées au calcul massivement parallèle.* PhD thesis, Univesité Paris-Saclay, 2017.

A. Dupuis and B. Chopard. Theory and applications of an alternative lattice Boltzmann grid refinement algorithm. *Physical Review. E, Statistical, nonlinear, and soft matter physics*, 67(066707):1–7, 2003.

L. Eckhart and H. Oertel Jr. *Numerische Strömungslehre.* VIEWEG+TEUBNER, 3 edition, 2009.

G. Eitel-Amor, M. Meinke, and W. Schröder. A lattice-Boltzmann method with hierarchically refined meshes. *Computers and Fluids*, 75:127–139, 2013.

A. Fakhari and T. Lee. Finite-difference lattice Boltzmann method with a block-structured adaptive-mesh-refinement technique. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 89(3):1–12, 2014.

A. Fakhari and T. Lee. Numerics of the lattice boltzmann method on nonuniform grids: Standard LBM and finite-difference LBM. *Computers and Fluids*, 107:205–213, 2015.

H. Feiz. *LES of multiple jets in crossflow using a coupled lattice boltzmann-finite volume solver*. PhD thesis, Georgia Institue of Technology, 2006.

O. Filippova and D. Hänel. Boundary-fitting and local grid refinement for lattice-BGM models. *International Journal of Modern Physics C*, 9(8):1271–1279, 1998a.

O. Filippova and D. Hänel. Grid refinement for lattice-BGK models. *Journal of Computational Physics*, 147:219–228, 1998b.

O. Filippova, S. Succi, F. Mazzocco, C. Arrighetti, G. Bella, and D. Hänel. Multiscale Lattice Boltzmann Schemes with Turbulence Modeling. *Journal of Computational Physics*, 170(2):812–829, 2001. ISSN 00219991.

U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice as Automata for the Navier-Stokes Equation. *Physical Review Letters*, 56(14):1505–1508, 1986.

M. Geier, A. Greiner, and J. G. Korvink. Cascaded digital lattice Boltzmann automata for high Reynolds number flow. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 73(6):1–10, 2006.

F. Gendre, D. Ricot, G. Fritz, and P. Sagaut. Grid refinement for aeroacoustics in the lattice Boltzmann method: A directional splitting approach. *Physical Review E*, 96 (2), 2017.

I. Ginzburg and D. D'Humières. Multi-reflection boundary conditions for lattice Boltzmann models. *Physical Review E*, 68(066614):1–30, 2003.

H. Grad. Note on N-dimensional Hermite polynomials. *Communications on Pure and Applied Mathematics*, 2(4):325–330, 1949.

Z. Guo and T. S. Zhao. Explicit finite-difference lattice Boltzmann method for curvilinear coordinates. *Physical Review E*, 67(6):066709, 2003.

Z. Guo, K. Xu, and R. Wang. Discrete unified gas kinetic scheme for all Knudsen number flows: Low-speed isothermal case. *Physical Review E*, 033305:1–11, 2013.

Z. Guo, R. Wang, and K. Xu. Discrete unified gas kinetic scheme for all Knudsen number flows. II. Thermal compressible case. *Physical Review E*, 033313:1–15, 2015.

Z. L. Guo, C. G. Zheng, and B. C. Shi. Non-equilibrium extrapolation method for velocity and pressure boundary conditions in the lattice boltzmann method. *Chinese Physics (Overseas Edition)*, 11(4):366–374, 2002.

S. L. Han, P. Zhu, and Z. Q. Lin. Two-dimensional interpolation-supplemented and Taylor-series expansion-based lattice Boltzmann method and its application. *Communications in Nonlinear Science and Numerical Simulation*, 12(7):1162–1171, 2007.

X. He and G. D. Doolen. Lattice Boltzmann method on a curvilinear coordinate system: vortex shedding behind a circular cylinder. *Physical Review E*, 56(1):434–440, 1997a.

X. He and L.-s. Luo. Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Physical Review E*, 56(6):6811–6817, 1997.

X. He, L. S. Luo, and M. Dembo. Some progress in lattice Boltzmann method. Part 1. Nonuniform mesh grids, J. *Journal of Computational Physics*, 129:357–363, 1996.

X. He, X. Shan, and G. D. Doolen. Discrete Boltzmann equation model for nonideal gases. *Physical Review E*, 57(1):13–16, 1998.

X. Y. He and G. Doolen. Lattice Boltzmann method on curvilinear coordinates system: Flow around a circular cylinder. *Journal of Computational Physics*, 134(2):306–315, 1997b.

F. J. Higuera and J. Jiménez. Boltzmann approach to lattice gas simulations. *Europhysics Letters (EPL)*, 9(7):663–668, 1989.

F. J. Higuera, S. Succi, and R. Benzi. Lattice gas dynamics with enhanced collisions. *Europhysics Letters*, 9(4):345–349, 1989.

C. Hirsch. *Numerical Computation of Internal and External Flows*. Elsevier BH, 2007.

K. Huang. *Statistical Mechanics*. Wiley & Sons, New York, 2nd edition, 1987.

A. Krämer, K. Küllmer, D. Reith, W. Joppich, and H. Foysi. Semi-Lagrangian off-lattice Boltzmann method for weakly compressible flows. *Physical Review E*, 95(2): 1–12, 2017.

P. K. Kundu, I. M. Cohen, and D. R. Dowling. *Fluid Mechanics*. Elsevier Inc., 5 edition, 2012.

D. Lagrava. *Revisiting grid refinement algorithms for the lattice Boltzmann method*. PhD thesis, Université de Genève, 2012.

D. Lagrava, O. Malaspinas, J. Latt, and B. Chopard. Advances in multi-domain lattice Boltzmann grid refinement. *Journal of Computational Physics*, 231(14):4808–4822, 2012.

J. Latt. *Hydrodynamic limit of lattice Boltzmann equations*. PhD thesis, Université de Genève, 2007.

J. Latt and B. Chopard. Lattice Boltzmann method with regularized non-equilibrium distribution functions. *physics.flu-dym*, 2005.

P. Le Tallec and F. Mallinger. Coupling Boltzmann and Navier-Stokes equations by half fluxes. *Journal of Computational Physics*, 136(1):51–67, 1997.

T. Lee and C.-L. Lin. A Characteristic Galerkin Method for Discrete Boltzmann Equation. *Journal of Computational Physics*, 171(1):336–356, 2001.

T. Lee and C. L. Lin. An Eulerian description of the streaming process in the lattice Boltzmann equation. *Journal of Computational Physics*, 185(2):445–471, 2003.

B. P. Leonard. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computer Methods in Applied Mechanics and Engineering*, 19(1):59–98, 1979.

D. Lycett-Brown and K. H. Luo. Multiphase cascaded lattice Boltzmann method. *Computers and Mathematics with Applications*, 67(2):350–362, 2014.

O. Malaspinas. *Lattice Boltzmann Method for the Simulation of Viscoelastic Fluid Flows*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2009.

S. Marié. *Etude des la Méthode Boltzmann sur Réeseau pour les Simulations en Aéeroacoustiques*. PhD thesis, Université Pierre et Marie Curie, 2008.

G. R. McNamara and G. Zanetti. Use of the boltzmann equation to simulate lattice-gas automata. *Physical Review Letters*, 61(20):2332–2335, 1988.

R. Mei and W. Shyy. On the Finite-Difference-Based Lattice Boltzmann Method in Curvilinear Coordinates. *Journal of Computational Physics*, 143:426–448, 1998.

F. Nannelli and S. Succi. The lattice Boltzmann equation on irregular lattices. *Journal of Statistical Physics*, 68(3-4):401–407, 1992.

X. D. Niu, C. Shu, Y. T. Chew, and T. G. Wang. Investigation of Stability and Hydrodynamics of Different Lattice Boltzmann Model. *Journal of Statistical Physics*, 117(November):665–680, 2004.

P. Prestininzi, M. L. Rocca, and R. Hinkelmann. Comparative study of a Boltzmann-based Finite Volume and a lattice Boltzmann model for shallow water flows in complex domains. *International Journal of Offshore and Polar Engineering*, 24(3):161–167, 2014.

Y. H. Qian, D. D'Humières, and P. Lallemand. Lattice BGK Models for Navier - Stokes Equation. *Europhysics Letters*, 17(6):479–484, 1992.

P. R. Rao and L. A. Schaefer. Numerical stability of explicit off-lattice Boltzmann schemes: A comparative study. *Journal of Computational Physics*, 285, 2015.

M. B. Reider and J. D. Sterling. Accuracy of discrete-velocity BGK models for the simulation of the incompressible Navier-Stokes equations. *Computers and Fluids*, 24 (4):459–467, 1995.

D. Ricot. *Simulation numérique d'un écoulement affleurant une cavité par la méthode Boltzmann sur reseau et application au toit ouvrant de vehicules automobiles*. PhD thesis, Ecole Centrale de Lyon, 2002.

M. Rohde, D. Kandhai, J. J. Derksen, and H. E. A. van den Akker. A generic, mass conservative local grid refinement technique for lattice-Boltzmann schemes. *International Journal for Numerical Methods in Fluids*, 51(4):439–468, 2006.

X. Shan, X.-F. Yuan, and H. Chen. Kinetic theory representation of hydrodynamics: a way beyond the Navier–Stokes equation. *Journal of Fluid Mechanics*, 550:413–441, 2006.

K. Shrestha. *Simulation of wall-bounded turbulent convective flows by Finite Volume Lattice Boltzmann Method*. PhD thesis, Université de Lille, 2015.

K. Shrestha, G. Mompean, and E. Calzavarini. Finite-volume versus streaming-based lattice Boltzmann algorithm for fluid-dynamics simulations: A one-to-one accuracy and performance study. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 93(2):1–14, 2016.

D. N. Siebert, L. A. Hegele, and P. C. Philippi. Lattice Boltzmann equation linear stability analysis: Thermal and athermal models. *Physical Review E*, 77(2):1–9, 2008. ISSN 15393755. doi: 10.1103/PhysRevE.77.026707.

J. D. Sterling and S. Chen. Stability Analysis of Lattice Boltzmann Methods. *Journal of Computational Physics*, 123:196–206, 1996.

M. Stiebler, J. Tölke, and M. Krafczyk. An upwind discretization scheme for the finite volume lattice Boltzmann method. *Computers and Fluids*, 35(8-9):814–819, 2006.

S. Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond*. Oxford University Press, 2001.

S. Succi. Lattice Boltzmann 2038. *EPL (Europhysics Letters)*, 109(5), 2015.

H. Touil, D. Ricot, and E. Lévêque. Direct and large-eddy simulation of turbulent flows on composite multi-resolution grids by the lattice Boltzmann method. *Journal of Computational Physics*, 256:220–233, 2014.

E. M. Viggen. *The lattice Boltzmann method: Fundamentals and acoustics*. PhD thesis, Norwegian University of Science and Technology, 2014.

D. A. Wolf-Gladrow. *Lattice Gas Cellular Automata and Lattice Boltzmann Models*, volume 1725. Springer, 2000.

H. Xu and P. Sagaut. Analysis of the absorbing layers for the weakly-compressible lattice Boltzmann methods. *Journal of Computational Physics*, 245:14–42, 2013.

N. Yeshala. *A coupled lattice-Boltzmann – Navier-Stokes methodology for drag reduction*. PhD thesis, Georgia Institute of Technology, 2010.

A. Zarghami, M. J. Maghrebi, J. Ghasemi, and S. Ubertini. Lattice Boltzmann finite volume formulation with improved stability. *Communications in Computational Physics*, 12(1):42–64, 2012.

L. Zhu, P. Wang, and Z. Guo. Performance evaluation of the general characteristics based off-lattice Boltzmann scheme and DUGKS for low speed continuum flows. *Journal of Computational Physics*, 333:227–246, 2017.

O. Zienkiewicz and R. Codina. A general algorithm for compressible and incompressible flows. Part I: The split, characteristic-based scheme. *International Journal for Numerical Methods in Fluids*, 20:869–885, 1995.

Méthodes numériques hybrides basées sur une approche Boltzmann sur réseau en vue de l'application aux maillages non-uniformes

Résumé: Malgré l'efficacité informatique et la faible dissipation numérique de la méthode de Boltzmann sur réseau (LBM) classique reposant sur un algorithme de *propagation-collision*, cette méthode est limitée aux maillages cartésiens uniformes. L'adaptation de l'étape de discrétisation à différentes échelles de la mécanique des fluides est généralement réalisée par des schémas LBM à échelles multiples, dans lesquels le domaine de calcul est décomposé en plusieurs sous-domaines uniformes avec différentes résolutions spatiales et temporelles. Pour des raisons de connectivité, le facteur de résolution des sous-domaines adjacents doit être un multiple de deux, introduisant un changement abrupt des échelles spatio-temporelles aux interfaces. Cette spécificité peut déclencher des instabilités numériques et produire des sources de bruit parasite rendant l'exploitation de simulations à finalités aéroacoustiques impossible. Dans la présente thèse, nous avons d'abord élucidé le sujet du raffinement de maillage dans la LBM classique en soulignant les défis et les sources potentielles d'erreur. Par la suite, une méthode de Boltzmann sur réseau hybride (HLBM) est proposée, combinant l'algorithme de *propagation-collision* avec un algorithme de flux au sens eulérien obtenu à partir d'une discrétisation en volumes finis des équations de Boltzmann à vitesse discrète. La HLBM combine à la fois les avantages de la LBM classique et une flexibilité géométrique accrue. La HLBM permet d'utiliser des maillages cartésiens non-uniformes. La validation de la méthode hybride sur des cas tests 2D à finalité aéroacoustique montre qu'une telle approche constitue une alternative viable aux schémas Boltzmann sur réseau à échelles multiples, permettant de réaliser des raffinements locaux en H. Enfin, un couplage original, basé sur l'algorithme de *propagation-collision* et une formulation isotherme des équations de Navier-Stokes en volumes finis, est proposé. Une telle tentative présente l'avantage de réduire le nombre d'équations du solveur volumes finis tout en augmentant la stabilité numérique de celui-ci, en raison d'une condition CFL plus favorable. Les deux solveurs sont couplés dans l'espace des moments, où la solution macroscopique du solveur Navier-Stokes est injectée dans l'algorithme de propagation-collision à l'aide de la collision des moments centrés. La faisabilité d'un tel couplage est démontrée sur des cas tests 2D, et les résultas obtenus sont comparés avec la HLBM.

Mots clés: *CFD, méthode Boltzmann sur réseau, volumes finis, equations Navier-Stokes, méthode hybride, raffinement de maillage, maillages non-uniformes.*

Hybrid numerical method based on the lattice Boltzmann approach with application to non-uniform grids

Abstract: Despite the inherent efficiency and low dissipative behaviour of the standard lattice Boltzmann method (LBM) relying on a two step *stream and collide* algorithm, a major drawback of this approach is the restriction to uniform Cartesian grids. The adaptation of the discretization step to varying fluid dynamic scales is usually achieved by multi-scale lattice Boltzmann schemes, in which the computational domain is decomposed into multiple uniform subdomains with different spatial resolutions. For the sake of connectivity, the resolution factor of adjacent subdomains has to be a multiple of two, introducing an abrupt change of the space-time discretization step at the interface that is prone to trigger instabilites and generate spurious noise sources that contaminate the expected physical pressure signal. In the present PhD thesis, we first elucidate the subject of mesh refinement in the standard lattice Boltzmann method and point out challenges and potential sources of error. Subsequently, we propose a novel hybrid lattice Boltzmann method (HLBM) that combines the *stream and collide* algorithm with an Eulerian flux-balance algorithm that is obtained from a finite-volume discretization of the discrete velocity Boltzmann equations. The interest of a hybrid lattice Boltzmann method is the pairing of efficiency and low numerical dissipation with an increase in geometrical flexibility. The HLBM allows for non-uniform grids. In the scope of 2D periodic test cases, it is shown that such an approach constitutes a valuable alternative to multi-scale lattice Boltzmann schemes by allowing local mesh refinement of type H. The HLBM properly resolves aerodynamics and aeroacoustics in the interface regions. A further part of the presented work examines the coupling of the *stream and collide* algorithm with a finite-volume formulation of the isothermal Navier-Stokes equations. Such an attempt bears the advantages that the number of equations of the finite-volume solver is reduced. In addition, the stability is increased due to a more favorable CFL condition. A major difference to the pairing of two kinetic schemes is the coupling in moment space. Here, a novel technique is presented to inject the macroscopic solution of the Navier-Stokes solver into the *stream and collide* algorithm using a central moment collision. First results on 2D tests cases show that such an algorithm is stable and feasible. Numerical results are compared with those of the previous HLBM.

Key words: *CFD, lattice Boltzmann method, finite-volume method, Navier-Stokes equations, hybrid lattice Boltzmann method, mesh refinement, non-uniform grids.*