Thesis

Hydrodynamic limit of lattice Boltzmann equations

LATT, Jonas

Abstract

An analysis of the lattice Boltzmann (LB) method is conducted, and various conclusions are drawn on how to exploit this method for the numerical resolution of the Navier-Stokes equation. A novel LB model is introduced, first for the simulation of advection-diffusion problems, and then for the resolution of the Navier-Stokes equation. The latter model, presented under the name of "regularized lattice Boltzmann", is shown to substantially increase the stability and accuracy of LB models in numerical simulaiton. The model is furthermore found to be innovative due to the fact that it possesses an accurate dimensionless formulation in terms of macroscopic variables. Based on this observation, the regularized model is investigated to address various challenges of LB, such as the implementaiton of boundary conditions and spatially refined grids.

Université de Genève
Département d'informatique
Département de physique

Faculté des sciences
Professeur Bastien Chopard
Professeur Michel Droz

Hydrodynamic Limit of Lattice Boltzmann Equations

THÈSE

présentée à la Faculté des sciences de l'Université de Genève
pour obtenir le grade de Docteur ès sciences, mention interdisciplinaire

par

Jonas Lätt

de Mühledorf (SO)

Thèse N^o 3846

Genève
Atelier de reproduction de la Section de physique
2007

FACULTÉ DES SCIENCES

Doctorat ès Sciences
mention interdisciplinaire

Thèse de *Monsieur Jonas LÄTT*

intitulée:

Hydrodynamic Limit of Lattice Boltzmann Equations

La faculté des sciences, sur le préavis de Messieurs B. CHOPARD, professeur adjoint et codirecteur de thèse (Département d'informatique), M. DROZ, professeur titulaire et codirecteur de thèse (Département de physique théorique), S. SUCCI, professeur (Istituto per le Applicazioni del Calcolo "Mauro Picone", Rome, Italie), autorise l'impression de la présente thèse, sans exprimer d'opinion sur les propositions qui y sont émises.

Genève, le 23 avril 2007

Thèse -3846-

Le Doyen, Pierre SPIERER

This dissertation was presented on March 9, 2007, at the Faculty of Science of the University of Geneva.

The members of the comittee were the following:

- Prof. Bastien Chopard, Supervisor
- Prof. Michel Droz, Supervisor
- Prof. Sauro Succi, Istituto Applicazioni Calcolo, Rome

Remerciements

En prenant modèle sur la dynamique des fluides émergeante d'une microscopie riche et abondante, cette thèse est issue d'un tissu d'interactions complexe avec mon entourage qui m'a soutenu à un niveau scientfique, technique et émotionnel. Je suis reconnaissant à tous les gens qui m'ont suivi durant ma thèse et espère qu'ils trouveront plaisir, en lisant ce document, à y reconnaitre une contribution, une idée ou un biais qui leur est dû.

Je pense en particulier à mes directeurs de thèse *Bastien Chopard* et *Michel Droz* qui m'ont suivi au long du travail, et à *Sauro Succi* qui a été conseiller et juré de thèse. Des conversations stimulantes ont permis de situer plusieurs sujets dans un contexte plus large, et j'aimerais faire valoir entre autres les contributions de *Peter Wittwer*, de *Vincent Heuveline* et de *Michel Deville*.

J'ai profité d'un environnement de travail inspirant grâce à *Jean-Luc* et *Bernhard* qui m'ont aidé à affronter le monde hostile de l'informatique, *Fokko* qui a soigneusement relu ma thèse, *Orestis* qui a prêté une oreille critique à mes idées farfelues et mes autres collègues qui m'ont tous donné un soutien remarquable.

La contribution la plus précieuse vient sans aucun doute de mon entourage proche, et je remercie *Roxane*, ma mère, les parents de Roxane et toute ma famille pour leur patience et leur appui.

Contents

Abstract

An analysis of the lattice Boltzmann (LB) method is conducted, and various conclusions are drawn on how to exploit this method for the numerical resolution of the Navier-Stokes equation. This focus contrasts with the traditional approach to the LB method, which usually considers complex fluids with extended physical properties. The specific scope of the present thesis is motivated by current practice in the domain of computational fluid dynamics. It is indeed observed that an increasing number of scientists and engineers simply use the LB method as an alternative to conventional numerical solvers for the Navier-Stokes equation. The conceptual framework presently available for an application of the LB method in this area is however still weak. This leads to a number of limitations, to which some workarounds are presented in the thesis.

In the first part, the LB method is reformulated such as to put the main emphasis on macroscopically relevant variables, instead of microscopical quantities originated from the kinetic theory of gases. From this perspective, the accuracy and the limitations of the lattice Boltzmann method are discussed in simple terms.

The second part introduces novel LB models, first for the simulation of advection-diffusion problems, and then for the resolution of the Navier-Stokes equation. The latter model, presented under the name of "regularized lattice Boltzmann" is shown to substantially increase the stability and accuracy of LB models in numerical simulation. The model is furthermore found to be innovative due to the fact that it possesses an accurate dimensionless formulation in terms of macroscopic variables.

The technical implications of this observation are investigated in the third part: the regularized LB method is used to solve various problems with varying time and space scales, ranging from the use of spatially refined grids to the coupling of the LB method with a finite difference method. The point of view adopted throughout this discussion is that practically all implementation issues in the LB method can be addressed properly by systematically applying the results from the corresponding Chapman-Enskog analysis.

The last part of the thesis deliberately changes the focus and utilizes the original, non-regularized LB method for the simulation of a complex fluid. Based on the microscopic LB variables, a model for the simulation of fluid turbulence is formulated.

This document presents a selective choice of the topics investigated during the thesis, aiming at a succinct overview of the main findings. It focuses on theoretical aspects and does not reflect another important part of the thesis, related to implementation issues of the numerical models. In can be pointed out in particular that the thesis led to the creation of three independent software projects. They introduce new programming paradigms for the implementation of parallel programs and LB models and provide freely available and well documented open source codes. This part of the work is summarized at the end of the present document, along with a listing of achieved publications.

Résumé en Français

Aperçu de la problématique

Introduction

La présente thèse de doctorat propose une étude de la méthode de Boltzmann sur réseau, dénotée par l'acronyme LB pour son nom anglais de "lattice Boltzmann". Diverses conclusions sont tirées sur les possibilités d'exploiter cette méthode pour la résolution numérique de l'équation de Navier-Stokes. Cette manière restrictive d'utiliser la méthode LB doit être vue par opposition à son utilisation traditionnelle, se situant dans le contexte de l'étude de fluides complexes avec des propriétés physiques étendues. Le choix adopté dans la thèse est motivé par l'état de l'art dans le domaine de la simulation de fluides (CFD pour l'expression anglaise "computational fluid dynamics"). En effet, on observe qu'un nombre croissant de scientifiques et ingénieurs utilisent la méthode de LB comme outil alternatif aux moyens de résolution numérique conventionnels pour l'équation de Navier-Stokes. Le cadre conceptuel à disposition actuellement pour une application du LB dans ce domaine est par contre toujours faible. Il en ensuit un nombre de limitations, auxquelles certains remèdes sont proposés dans la thèse.

Dans la première partie, la méthode LB est reformulée d'une telle manière à mettre en évidence les variables de nature macroscopique, au lieu des quantités microscopiques issues de la théorie cinétique des gaz. De cette nouvelle perspective, la précision ainsi que les limitations de la méthode de Boltzmann sur réseau sont étudiés en termes simples.

La deuxième partie introduit de nouveaux modèles LB, d'abord pour la simulation de problèmes à convection-diffusion, et ensuite pour la résolution de l'équation de Navier-Stokes. Il est démontré que ce dernier modèle, présenté sous le nom de "lattice Boltzmann régularisé", augmente de manière substantielle la stabilité et la précision des modèles LB au cours de simulations. Un autre aspect innovant de ce modèle se réfère au fait qu'il s'écrit de manière exacte en terme de variables macroscopiques.

Les implications techniques de cette observation sont étudiées dans la troisième partie: la méthode LB régularisée est utilisée pour résoudre divers problèmes possédant des échelles temporelles et spatiales variables. Il s'agit là par exemple de grilles numériques localement raffinées ou de couplages entre une méthode LB et une méthode de différences finies. Le point de vue adopté à travers cette étude s'appuie sur l'idée que pratiquement toute question d'implémentation pratique en LB se résoud à travers une analyse de Chapman-Enskog correspondante.

Cette optique est délibérément modifiée lors de la dernière partie de la thèse, dans laquelle la méthode LB originale, non-régularisée est utilisée pour la simulation d'un fluide complexe. Sur la base de variables LB microscopiques, un modèle pour la simulation de fluides turbulents est formulé.

Ce document contient un choix sélectif des sujets étudiés au cours de la thèse, afin de résumer brièvement les découvertes majeures. Il se concentre sur les aspects théoriques et omet de détailler une autre partie importante de la thèse, relative aux questions d'implémentation numérique. On peut en particulier mettre faire valoir le travail investi dans la conception de trois projets de logiciel indépendants, introduisant de nouveaux paradigmes de programmation pour l'implantation de programmes parallèles et de modèles LB. En outre, ces projets proposent des codes source libres et pleinement documentés. Cette partie du travail est résumée à la fin de la thèse, accompagnée d'une liste des articles publiés au cours de la thèse.

Simulations en dynamique des fluides

La recherche en simulation de dynamique des fluides (CFD) se concentre sur une équation a différences partielles (PDE pour l'anglais "partial differential equation") connue sous le nom d'équation de Navier-Stokes incompressible qui exprime une loi de conservation locale pour la quantité de mouvement du système. Elle représente un modèle largement simplifié pour un fluie inviscide à une seule composante dans un environnment sans variations de densité ou de température, et sans autre ingrédient physique. Bien que cette équation n'adresse que partiellement la complexité de la plupart des fluides considérés en ingénierie, elle est utilisée avec beaucoup de succès dans différents domaines afin de prédire qualitativement le comportement de systèmes immergés dans un fluide. Comme exemple, on peut nommer les problèmes d'optimisation en aérodynamique, tel que la conception d'une carrosserie de voiture ou la création d'ailes d'avion. L'équation de Navier-Stokes incompressible n'arrive bien évidemment à décrire la nature de l'air compressible que de manière très approximative. Néanmoins, cette équation mène à des prédiction de très bonne qualité qu'on utilise fréquemment, conjointement aux résultats de la simulation expérimentale. On peut par ailleurs noter que la résolution de cette PDE représente une tâche de calcul intensive, malgré la simplicité apparente de l'équation. On investit donc beaucoup d'efforts dans le développement de techniques de résolution numérique, plutôt que de rajouter davantage de complexité en ajoutant des termes physiques additionnels.

D'un autre côté, les domaines de recherche se concentrant sur des fluides dits complexes ont également une grande importance dans le domaine. Les fluides en question possédent une physique enrichie: pour ceux-ci, l'approximation de l'équation de Navier-Stokes est donc insuffisante. L'approche classique adoptée en CFD pour traiter ce type de fluides consiste en la description de nouvelles propriétés physiques en terme de phénomènes de transport reliées à une propriété macroscopique. Une nouvelle PDE est formulée pour la dynamique de cette propriété, qu'on résout ensuite par une technique numérique appropriée. L'interaction entre les différents phénomènes de transport est traitée à travers de termes ajoutés dans les PDE's respectives. Dans un fluide avec d'importantes variations de température par exemple, la température est introduite comme une nouvelle propriété, et sa dynamique est décrite par une équation de transport pour la chaleur. En plus, l'équation de conservation du moment est étendue pour tenir compte de la compressibilité du fluide. La température et la vitesse sont ensuite liés à travers un terme convectif dans l'équation de la chaleur et un terme de poussée verticale dans

l'équation du moment. Une équation d'état additionnelle couple la température avec la densité. On peut dire que le domaine traditionnel de la CFD adopte une approche de haut niveau à la modélisation d'un fluide, en commençant par une loi de conservation simple, et en rajoutant de nouvelles équations sur demande, basées sur des considérations théoriques et phénoménologiques.

Méthode de Boltzmann sur réseau

Le sujet principal de la thèse se traduit par une approche numérique du nom de Boltzmann sur réseau (LB pour le terme anglais "lattice Boltzmann"). Il s'agit d'un méthode realtivement récente qui se distingue méthodes traditionelle en adoptant une approche de bas niveau à la modélisation de fluides. A cet effet, elle décrit le fluide à un niveau moléculaire et élabore des modèles pour l'interaction entre molécules. La physique complète du fluide à un niveau du continu se trouve implicitement décrite par le modèle, et le travail conceptuel qu'on doit effectuer se trouve à l'opposé de ce qu'on ferait dans l'approche de haut niveau classique. En effet, les ingrédients physiques du modèle doivent être identifiés un par un et explicités propement, afin de permettre une séparation entre propriétés relevantes et négligeables. En se basant sur ces considérations, le modèle de collision est largement simplifié pour les besoins du traitement numérique. En tant que tel, le LB est une méthode bien adaptée à la simulation de fluides complexes, étant donné que les ingrédients physiques relevants sont pris en charge très naturellement par la méthode. Il est mis en évidence à quel point une formulation LB d'un problème de dynamique des fluides peut être élégante lorsqu'on on simule par exemple des fluides à deux phases. En CFD traditionnelle, une PDE est formulée pour chacune des deux phases, ainsi que pour la dynamique d'interaction sur l'interface entre celles-ci. L'implémentation numérique d'un tel modèle requiert l'utilisation de techniques avancées en génie logiciel, afin de correctement tracer la position de l'interface et implémenter la dynamique correspondante. Ces difficultés n'apparaissent pas dans la méthode LB, car les deux phases peuvent être modélisées par un modèle de fluide unique, et l'interface entre les phases est traitée automatiquement, en ingrédient naturel du modèle.

La méthode LB appliquée au domaine de la CFD

Lorsque l'approche LB à la modélisation physique est importée dans le domaine de CFD classique, deux mondes très différents, possédant leurs propres convictions et traditions sont confrontés. Ceci mène à de nombreux malentendus, d'un côté de la part de la communauté de la CFD qui manque de reconnaitre le potentiel de la méthode LB par sa juste valeur, et d'un autre côté de la communauté LB qui éprouve des difficultés à reconnaitres les enjeux réels de la CFD, ou à comprendre quels sont les domaines dans lesquels le LB peut être compétitifs avec d'autres approches. Un obstacle majeur à un échange mutuel entre ces deux méthodes vient du fait qu'elles décrivent le fluide à travers un choix de variables différent. Les variables centrales en CFD sont les quantités macroscopiques, observables. Dans l'équation de Navier-Stokes incompressible, les degrés de liberté sont définis par la vitesse et la pression, dépendants de l'espace et du temps. Les outils de résolution numérique

en CFD classique modèlisent souvent directement ces variables-ci, c'est-à-dire, ils représentent la valeur adoptée par ces variables sur les noeuds d'une grille numérique donnée. La méthode LB d'un autre côté simule la dynamique d'une quantité appelée fonction de distribution de particules, dérivée de la théorie cinétique des gaz. S'il est bien connu comment cette quantités peut être convertie en variables d'intérêt macroscopique, la procédure inverse par contre est plus compliquée. De fait, il n'est pas toujours clair de quelle manière une configuration donnée des variables macroscopiques est utilisée pour initialiser la fonction de distribution. Ce problème est rarement mis en évidence par une communauté habituée à penser en terme de quantités cinétiques uniquement. Il est néanmoins à la source de complications substantielles lorsque des variables macroscopiques sont évoquées explicitement dasn la simulation, ou lorsque les échelles d'espace et de temps macroscopiques du modèles sont ajustées.

Comment traiter des échelles temporelles et spatiales variables?

Les problèmes liés à la conversion entre variables macroscopiques et fonctions de distribution apparaissent par exemple lorsque l'on applique une technique de raffinement de grille, au cour de laquelle le domaine de la simulation est partitionné en sous-domaines, dont chacun est représenté par une grille numérique possédant un degré de résolution variable. Le but de cet exercice est d'adapter la structure de la grille à la géométrie du problème, et par conséquent, d'augmenter l'efficacité du calcul. Lors de l'exécution d'une simulation, il s'avère nécessaire de communiquer les valeurs calculées au fur et à mesure d'une grille à l'autre. Ce procédé s'applique sans difficulté dans le contexte des outils de CFD classiques. En effet, ces méthodes gardent en mémoire la valeur sans dimensions des variables macroscopiques, indépendante donc de la résolution de la grille ou du pas temporel, s'échangeant entre différente grilles sans besoin de conversion. La fonction de distribution de particules utilisée en LB par contre n'est pas adimensionnelle par rapport au choix d'unités macroscopiques. Elle doit donc être remise à l'échelle lors du transfer d'une grille à une autre, puisque sa valeur dépend de l'espacement de la grille. La règle qui détermine cette remise à l'échelle n'est aucunement évidente, et doit être appliquée avec beaucoup de soins. Par conséquent, une certaine confusion règne à ce sujet dans la litérature, et de nombreuses approches différentes ont été proposées, dont les idées de base diffèrent substantiellement.

Le raffinement n'est qu'un exemple particulier, introduit ici dans un souci d'illustration. La figure 1 affiche d'autres problèmes à échelles d'espace et de temps variables, nécessitant une transformation entre les unités macroscopiques et celles du LB: l'implantation de conditions aux bords et de conditions initiales, et l'implantation de modèles hybrides, faisant appel en même temps à une méthode LB et à un outil classique de la CFD.

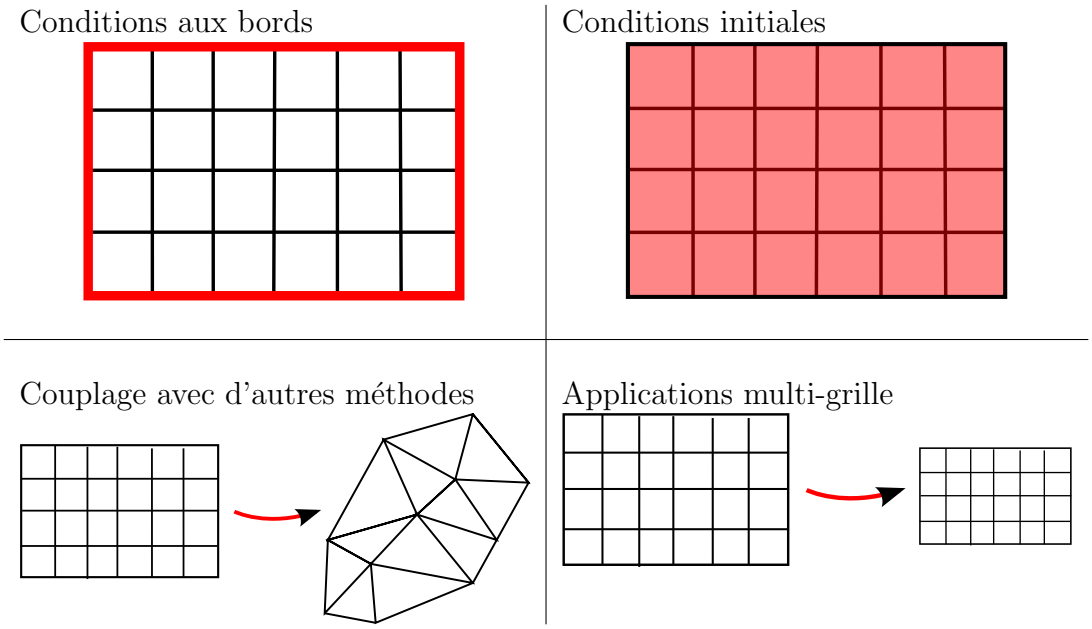


Figure 1: Divers problèmes se posent lors de l'implantation d'un modèle LB, pour lesquels une conversion entre variables macroscopiques et fonction de distribution est requise.

Méthode LB et développement multi-échelle

L'équation de Navier-Stokes incompressible

Dans ce qui suit, les problèmes de la dynamique des fluides sont réduits à un point de vue purement macroscopique. Dans ce contexte, les questions concernant la simulation d'un fluide équivalent à des questions de résolution numérique d'une équation différentielle à dérivées partielles. Un fluide incompressible et isotherme de viscosité ν est par exemple décrit par l'équation suivante, connue sous le nom d'*équation de Navier-Stokes*:

$$\frac{\partial \vec{u}}{\partial t} + \overrightarrow{\nabla} \cdot \left(\vec{u} \otimes \vec{u} + \frac{p}{\rho_0} \boldsymbol{I} - 2\nu \boldsymbol{S} \right) = 0.$$

Elle exprime la conservation de la quantité de mouvement d'un fluide caractérisé par sa vitesse $\vec{u}$ et sa pression p. La PDE s'écrit sous forme d'une divergence d'un terme tensoriel. Il est sous-entendu que la divergence est appliquée au deuxième indice du tenseur. En outre, on désigne par le symbole $\otimes$ le produit tensoriel entre deux vecteurs. Le tenseur $\boldsymbol{I}$ désigne l'identité, et $\boldsymbol{S}$ le tenseur des taux de déformation, défini comme la composante symétrique du tenseur des dérivées premières du champ de vitesse:

$$S = \frac{1}{2} \left(\overrightarrow{\nabla} \otimes \vec{u} + \left(\overrightarrow{\nabla} \otimes \vec{u} \right)^T \right)$$

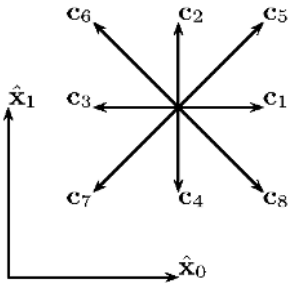


Figure 2: Le réseau D2Q9 pour la simulation de fluides en deux dimensions.

Méthode de Boltzmann sur réseau

La méthode de Boltzmann sur réseau est présentée dans le chapitre 1.3 de la thèse, ainsi que dans les références indiquées à la fin de ce chapitre. Un aperçu rapide de la méthode est reproduit ici, afin d'introduire la notation.

La méthode LB considère une version discrète de la fonction de distribution $f(\vec{x}, \vec{c}, t)$, définie sur l'espace des positions $\vec{x}$ et des vitesses $\vec{c}$. Lors de la discrétisation, l'espace des vitesses $\vec{c}$ est omis en faveur d'une augmentation du nombre de fonctions de distribution, dont un nouvel exemplaire est introduit pour chaque individu d'une population de vecteurs de vitesse discrète. Chacune de ces fonctions de distribution est identifiée par un indice k: $f_k = f_k(\vec{x}, t)$. Les vecteurs vitesse utilisés dans le cas discret représentent la relation spatiale entre une cellule du réseau et ses plus proches voisins. En deux dimensions, on utilise par exemple fréquemment le réseau D2Q9 qui contient neuf vecteurs, un pour chacun des huit voisins, et un pour le vecteur nul, représentant des "particules au repos" ne quittant pas la cellule. Ce réseau est illustré sur la figure 2.

La dynamique discrète d'un modèle LB exprime la propagation de flux de particules d'un noeud $\vec{x}_{ij}$ vers ses voisins. L'équation est écrite dans un système d'unités du réseau, dans lequel l'espacement spatial entre deux noeuds voisins de la grille, ainsi que l'espacement temporel entre deux pas d'itération, sont unitaires. La dynamique LB s'exprime donc par la formule

$$f_k(\vec{x}_{ij} + \vec{c}_k, t+1) - f_k(\vec{x}_{ij}, t) = \Omega_k(\vec{x}_{ij}, t),$$

où le terme Ω désigne l'opérateur de collision entre particules. Dans le modèle le plus couramment utilisé, connu sous le nom de BGK, la collision s'écrit comme une relaxation vers un équilibre local:

$$\Omega_k = -\omega \left(f_k - f_k^{eq}\right),$$

où on a défini l'équilibre, une version discrète de la distribution de Maxwell-Boltzmann, ne dépendant que des variables macroscopiques, de la manière suivante:

$$f_k^{eq} = f_k^{eq}(\rho, \vec{u}) = \rho\, t_k \left(1 + \frac{1}{c_s^2}\vec{c}_k \cdot \vec{u} + \frac{1}{2c_s^4}\left(\vec{c}_k \cdot \vec{u}\right)^2 - \frac{1}{2c_s^2}\left|\vec{u}\right|^2\right).$$

Le paramètre de relaxation ω peut être formellement relié à la viscosité du fluide.

Les quantités macroscopiques peuvent être calculées à partir de fonctions de distribution en calculent les moments d'ordre zéro, et du premier ordre. On trouve ainsi

- La densité: $\rho = \sum_k f_k$.
- La vitesse: $\vec{u} = 1/\rho \sum_k \vec{c}_k f_k$.

Bien qu'en principe la méthode LB décrive un fluide compressible, on retrouve l'équation d'un fluide incompressible dans la limite des faibles nombres de Mach. Dans ce cas, la relation entre la densité et la pression est décrite par l'équation pour un gaz parfait:

$$p = c_s^2 \rho.$$

Développement multi-échelle

Bien qu'on ait montré dans la section précédente comment calculer les variables macroscopiques à partir des fonctions de distribution, la procédure inverse n'a jusqu'alors pas été abordée. Afin de connecter en détail la dynamique LB avec une dynamique macroscopique, la limite du continu doit d'abord être évaluée par un développement de Taylor du deuxième ordre, et une technique de séparation des échelles temporelles et spatiales est utilisée pour évaluer la limite hydrodynamique du modèle. Le développement de Taylor se résume par l'approximation suivante:

$$f_k(\vec{x}_{ij} + \vec{c}_k, t+1) - f_k(\vec{x}_{ij}, t) \approx (\partial_t + \overrightarrow{\nabla} \cdot \vec{c}_k)\, f_k + \frac{1}{2}\left(\partial_t^2 + 2\partial_t \nabla \cdot \vec{c}_k + \left(\overrightarrow{\nabla} \cdot \vec{c}_k\right)^2\right) f_k.$$

Dans le processus de séparation des échelles, les fonctions f_k, ainsi que les dérivées spatiales et temporelles, sont développés en fonction d'un paramètre formel ϵ petit, c'est-à-dire $\epsilon \ll 1$:

$$\begin{aligned} f_k &= f_k^{(0)} + \epsilon f_k^{(1)} + \mathcal{O}(\epsilon^2), \\ \partial_t &= \epsilon \partial_{t1} + \epsilon^2 \partial_{t2} + \mathcal{O}(\epsilon^3) \quad \text{et} \\ \overrightarrow{\nabla} &= \epsilon \overrightarrow{\nabla}^{(1)} + \mathcal{O}(\epsilon^2). \end{aligned}$$

En réintroduisant ces approximations dans le modèle BGK, on retrouve bien l'équation de Navier-Stokes, et on conclut que ce modèle simule proprement la dynamique d'un fluide. Ce résultat est bien connu est représente un des piliers sur lesquels repose la méthode. Ce n'est par contre pas ce résultat en lui-même qui nous intéresse présentement, mais plutôt un résultat intermédiaire, reliant les fonctions de distribution aux champs macroscopiques.

Le terme constant en ϵ est en effet équivalent à la fonction d'équilibre, et ne dépend donc que de ρ et de $\vec{u}$:

$$f_k^{(0)} = f_k^{eq}(\rho, \vec{u}).$$

Le terme d'ordre 1 est lié aux dérivées de la vitesse, ainsi que du carré de la vitesse:

$$\epsilon f_k^{(1)} = -\frac{t_k}{c_s^2 \omega}\Big(\quad \boldsymbol{Q}_k : \rho \overrightarrow{\nabla} \otimes \vec{u} - \vec{c}_k \otimes \overrightarrow{\nabla} : \rho \vec{u} \otimes \vec{u} + \frac{1}{2c_s^2}(\vec{c}_k \cdot \overrightarrow{\nabla})(\boldsymbol{Q}_k : \rho \vec{u} \otimes \vec{u})\Big),$$

où on a défini le tenseur symétrique

$$\boldsymbol{Q}_k = \vec{c}_k \otimes \vec{c}_k - c_s^2 \boldsymbol{I}.$$

Il est important de remarquer que le détail des fonctions de distribution n'est pas relevant dans le calcul de la limite hydrodynamique du modèle. L'importance revient plutôt au moment du deuxième ordre, $\boldsymbol{\Pi}^{(1)}$, calculé selon la formule suivante:

$$\boldsymbol{\Pi}^{(1)} = \sum_k \vec{c}_k \otimes \vec{c}_k f_k^{(1)}.$$

Au cours du développement multi-échelle, on montre que ce tenseur est proportionnel au tenseur des taux de déformation:

$$\boldsymbol{\Pi}^{(1)} = -\frac{2c_s^2}{\rho}\boldsymbol{S}.$$

Lors du calcul du deuxième moment des fonctions de distribution, tous les termes contenus dans la fonction de distribution n'entrent pas en compte. En effet, les termes définis au carré de la vitesse s'éliminent pour des raisons de symétrie. Dans le but d'obtenir le "bon" tenseur $\Pi^{(1)}$ produisant une hydrodynamique appropriée, il est donc suffisant de considérer une version réduite des fonctions f_k:

$$\epsilon \bar{f}_k^{(1)} = -\frac{t_k}{c_s^2 \omega}\left(\boldsymbol{Q}_k : \rho \overrightarrow{\nabla} \otimes \vec{u}\right).$$

Cette expression relie de manière simple le champs macroscopique aux valeurs des fonctions de distribution.

Modèle de Boltzmann sur réseau régularisé

Procédé de régularisation

Dans le chapitre précédent, on a introduit une version simplifiée des fonctions de distribution f_k. Elle se désigne par le terme $\bar{f}_k$ et représente proprement les composants hydrodynamiques du modèle. Ces variables ont été reliées par une relation simple aux dérivées du champ de vitesse. Pour l'utilisation courante de cette expression, il serait par contre plus utile que les fonctions de distribution soient directement liées aux fonctions macroscopiques simulées, donc la densité ρ, la vitesse $\vec{u}$ et le tenseur des taux de déformation $\boldsymbol{S}$. En effet, il a été montré que ces trois grandeurs peuvent être évaluées localement en chaque noeud du réseau LB, en calculant les moments des fonctions de distribution.

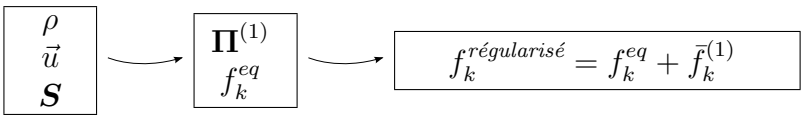
Une telle relation se trouve en exploitant la symétrie du tenseur $\boldsymbol{Q}$:

$$\begin{aligned}
\bar{f}_k^{(1)} &= -\frac{t_k \rho}{c_s^2 \omega}\boldsymbol{Q}_k : \overrightarrow{\nabla} \otimes \vec{u} \\
&= -\frac{t_k \rho}{c_s^2 \omega}\boldsymbol{Q}_k : \frac{1}{2}\left[(\overrightarrow{\nabla} \otimes \vec{u}) + (\overrightarrow{\nabla} \otimes \vec{u})^T\right] \\
&= -\frac{t_k \rho}{c_s^2 \omega}\boldsymbol{Q}_k : \boldsymbol{S} \\
&= \frac{t_k}{2c_s^4 \omega}\boldsymbol{Q}_k : \boldsymbol{\Pi}^{(1)}
\end{aligned}$$

Cette relation est centrale dans pratiquement tous les sujets développés dans la thèse. Elle relie les champs macroscopiques aux valeurs des fonctions de distribution, et peut donc être exploitée systématiquement, à chaque fois qu'une telle transformation est nécessitée. Pour souligner cette importance, la formule de régularisation est reproduite, en omettant cette fois-ci les calculs qui ont permis de la trouver:

$$\boxed{\bar{f}_k^{(1)} = \frac{t_k}{2c_s^4\omega}\boldsymbol{Q}_k : \boldsymbol{\Pi}^{(1)}}$$

En suivant l'enchainement suivant, la fonction de régularisation est utilisée pour évaluer les valeurs des fonctions de distribution à partir des variables macroscopiques:



Modèle LB régularisé

La formule de régularisation est utilisée dans plusieurs domaines qui sont présentés dans la thèse. A titre d'exemple, l'application probablement la plus importante est expliquée dans ce qui suit: le modèle de Boltzmann sur réseau modifié, dont les fonctions de distribution sont régularisées.

L'algorithme pour la collision entre particules dans ce modèle se décompose en trois étapes:

1. Calcul de ρ, $\vec{u}$ et $\boldsymbol{\Pi}^{neq} = \sum_k \vec{c}_k \otimes \vec{c}_k (f_k - f_k^{eq})$.
2. Régularisation $f_k \leftarrow f_k^{régularisé}$
3. Exécution de la collision BGK sur les fonctions de distribution régularisées.

Afin d'apprécier les qualités de ce nouveau modèle, un flux bidimensionnel connu sous le nom de *flux de Kovasznay* a été implanté numériquement, une fois en utilisant la méthode BGK, et une fois en s'appuyant sur le modèle régularisé. Une solution analytique pour ce flux est connue, et on peut donc vérifier la précision de la solution numérique, en la comparant à l'expression analytique. Le résultat de ce test a montré que le modèle régularisé est nettement plus précis que le modèle BGK, et que le gain en précision augmente au fur et à mesure que la grille est raffinée. Sur des grilles de taille commune, on peut gagner un ordre de précision en utilisant le modèle régularisé.

Un autre modèle bidimensionnel a été implanté, désigné par le nom de *flux de cavité*. Il représente un fluide contenu dans une boite quadratique, dont la paroi supérieure est entrainée à une vitesse constante. Cette fois-ci, la stabilité numérique du modèle a été testée, c'est-à-dire, la valeur maximale du nombre de Reynolds pouvant être atteinte avant que la simulation soit numériquement instable. Dans les deux cas, BGK et régularisé, une relation linéaire entre le nombre de Reynolds maximal et le paramètre de résolution N d'une grille $N \times N$ est établie. La croissance

linéaire est par contre sept fois plus rapide dans le cas du modèle BGK. Dans les détails, on trouve

Pour le modèle BGK: $Re^{max} = 0.4 + 1.0 * N$

Pour le modèle régularisé: $Re^{max} = 16 + 7.4 * N$

Il est donc établi que pour ce type de problèmes, le modèle régularisé est nettement plus précis et plus stable que le modèle BGK. Cet avantage s'explique par l'origine de la méthode régularisée. Elle a été construite en imposant une séparation des échelles au modèle original, le modèle BGK. Uniquement les termes hydrodynamiques, c'est-à-dire, le termes contribuant à la dynamique de l'équation de Navier-Stokes, ont été préservés, et les termes d'ordre supérieur sont négligés. Il est vrai qu'à cause de cette simplification, la méthode régularisée s'est apauvrie. On s'attend en particulier à ce qu'elle soit moins apte à présenter spontanément des ingrédients physiques d'un fluide complexe. Dans un but particulier par contre, celui de la résolution de l'équation de Navier-Stokes, elle est plus concise, plus pratique à manier, plus précise et plus stable.

Introduction

Computational fluid dynamics

The present thesis investigates a methodology in the domain of *computational fluid dynamics* (CFD), that is, the numerical simulation of fluid flows. The core of this research area is represented by a partial differential equation (PDE) widely known as the *incompressible Navier-Stokes equation*, which expresses a local conservation law for the momentum in the system. It constitutes a largely simplified model for a viscid, single component fluid in an environment without density or temperature variations nor any other specific physical ingredients. Although this equation only partially addresses the complexity of most fluids of interest in engineering applications, it is successfully applied in different areas for qualitative predictions of fluid flows. A case in point are optimization problems in aerodynamics, as for example the design of a car body, or the creation of airfoils. It is obvious that the incompressible Navier-Stokes equation only roughly approximates the nature of compressible air. Nevertheless, this equation leads to predictions of high quality that are successfully exploited, jointly with the results of physical experiments. Solving the Navier-Stokes equation is actually a numerically very challenging task, in spite of the apparent simplicity of the PDE. Much effort is therefore invested in developing numerical approaches to the resolution of this equation, rather than adding even more complexity by additional physical terms.

On the other hand, some research branches investigate so-called *complex fluids* that exhibit extended physical properties, and to which the basic Navier-Stokes equation is an insufficient approximation. The classical approach in CFD to the treatment of such fluids is to describe the new physical properties in terms of *transport phenomena* related to a new observable, macroscopic property. A new PDE is written down for the dynamics of this property, which then is solved by an appropriate numerical technique. The interaction between the different transport phenomena is handled via interaction terms between the respective PDE's. In a fluid with important temperature variations for example, a new observable property, the temperature, is introduced and its dynamics is described by a heat transport equation. Furthermore, the momentum equation is extended to comprehend compressible fluids. The temperature and the velocity are then linked through a convective term in the heat equation and a buoyancy term in the momentum equation. An additional equation of state links the temperature to the density. One can view the traditional field of CFD as a top-down approach to fluid modeling, starting with a simple conservation law, and adding new equations on demand, based on both theoretical and phenomenological considerations.

Lattice Boltzmann method

The main topic of the thesis is a numerical approach known under the name of *lattice Boltzmann method*. This method is relatively new and contrasts with the traditional approach to CFD by adopting a bottom-up approach to fluid modeling. To achieve this, it describes the fluid at a molecular level and proposes models for the collision between molecules. The full continuum-level physics of the fluid is implicitly contained in this model, and the conceptual work to be effectuated is opposite to the one in the top-down approach of classical CFD. Indeed, the various physical ingredients contained in the model need to be identified one by one and properly explicited in order to allow for a segregation between relevant and negligible properties. Based on those considerations, the collision model is substantially simplified for the needs of the numerical treatment. As such, the lattice Boltzmann (LB) method is well adapted to the simulation of complex fluids, as the relevant physical ingredients are taken in charge very naturally by the method. How elegant a LB formulation of a fluid problem can be becomes apparent for example in the context of two-phase flows. In traditional CFD, a PDE is written down for each of the two phases, as well as for the interaction of the phases on their common interface. The implementation of such a model requires advanced software engineering techniques to track the position of this interface and to implement its dynamics. Those problems don't appear in the LB method, where the two phases can be represented by the same fluid model, and the interface between the phases is handled automatically, as a natural ingredient of the model.

In spite of this apparent adequacy of the LB model for complex fluids, the recent historical development shows that much effort is invested in solving the raw Navier-Stokes equation with this method. This is reflected by numerous publications and technical reports, in which the LB method is used to solve benchmark applications of classical CFD, and its relative efficiency is compared to the one of other methods. A reason for the success of the LB method in this domain can possibly be found in the relative simplicity of its numerical implementation. As a matter of fact, the physical principles that stand behind the method can be intimately connected with classical programming paradigms of high performance computing. For example, the fact that collisions between particles are local in their nature encourages the use of parallel architectures in which spatial subdomains of the problem are solved locally on different computational nodes.

The LB method applied to CFD

When the LB approach to physical modeling is imported in the domain of classical CFD, two very different worlds with different traditions and convictions are confronted. This leads to numerous misunderstandings, on the one hand from the CFD community that fails to recognize the potential of the LB method by its right value, and on the other hand from the LB community that has difficulties to recognize the real challenges of CFD, or the areas in which the LB method can prove competitive with other approaches. One main obstacle to a mutual exchange between those two methods stems from the fact that they describe the fluid via a different choice of vari-

ables. The variables of concern in CFD are the macroscopic, observable quantities. In the incompressible Navier-Stokes equation, the degrees of freedom are defined by the space and time dependent velocity $\vec{u} = \vec{u}(\vec{r}, t)$ and the pressure $p = p(\vec{r}, t)$. Classical CFD solvers most often calculate directly a discrete representation of these variables, that is, they store the value adopted by the variables on the nodes of a given numerical grid. The LB method on the other hand simulates the dynamics of a so-called particle distribution function, a quantity derived from the kinetic theory of gases that represents a probability density for the presence of particles in phase space. Although it is well known how the macroscopic variables are computed from a given state of the particle distribution function, the reverse procedure is much more contrived. It is actually not always clear how a given configuration of the macroscopic variables can be used to initialize the distribution function. This fact is rarely recognized as a problem by a community that is used to thinking exclusively in terms of kinetic quantities. It does however generate many complications whenever macroscopic variables are explicitly evoked in the simulation, or when the macroscopic length and time scales of the model need to be adapted.

How to treat varying space and time scales?

As an example, the difficulty becomes apparent when a grid refinement technique is used, in which the computational domain is partitioned into subdomains that are represented by grids with a varying degree of resolution. The purpose of this procedure is to adapt the structure of the grid to the geometry of the problem, and thus to gain computational efficiency. During the simulation, it is necessary to communicate successively computed values from one grid to another. This is easily performed in classical fluid solvers. Indeed, they store the value of dimensionless macroscopic variables that do not depend on a given resolution of the grid and can therefore communicate those values without the need for conversions. On the other hand, the particle distribution function used in LB methods is not dimensionless with respect to a choice of macroscopic units. It must therefore be rescaled during the transfer from a grid to another, because its value depends on the specific grid spacing. The rules that dictate this rescaling are in no way straightforward, and much care must be used to get them right. A certain confusion reigns consequently in the literature on this point, and numerous different approaches have been proposed, based on substantially different points of view. Grid refinement is just a case in point to illustrate the encountered difficulties, but similar problems are encountered with essentially all practical implementation issues of LB, such as the definition of initial conditions or boundary conditions, the implementation of an adaptive time interval, the coupling of LB methods with other techniques, and many more.

Multiscale expansion of lattice Boltzmann

The value of the distribution function can be formally related to the macroscopic variables through an infinite series via the so-called Chapman-Enskog multi-scale expansion. This expansion is commonly used to verify that a given LB method effectively solves the macroscopic PDE's asymptotically. To do this, the infinite

series is truncated. The remaining lower order terms, sufficient to recovering the Navier-Stokes equation, are called *hydrodynamic terms*. This technique is however not so frequently used to actually develop new methods in the domain of LB. The point of view adopted in the present thesis is that practically all implementation issues in the LB method can be addressed properly by systematically taking into account the results of the Chapman-Enskog analysis. In this approach, intuitive reasoning is underlined by formal proofs, and a uniform strategy is devised to solve different problems in a common framework.

Content of the thesis

In the first part of the thesis, various theoretical results from the literature are gathered to show that a LB implementation effectively produces results equivalent to those of a conventional CFD solver. The necessity to include some less known terms into the model is highlighted, and the limitations of the approach are emphasized. This part includes also a novel contribution that shows how to properly implement the application of an external force term in a compressible fluid model with adaptable bulk viscosity. Given that the theoretical developments can be somewhat hard to follow, they are preceded, for illustration purposes, by similar-looking but simpler discussions of a LB model for advection-diffusion problems. On the way, it is shown that this advection-diffusion model is treated erroneously in the literature, and a workaround is suggested.

In the second part, a modified LB model is introduced under the name of *regularized lattice Boltzmann* model. In this model, the original LB distribution function is replaced by a regularized function, inspired by the hydrodynamic terms found in the Chapman-Enskog expansion. The specificity of the regularized distribution function is that it can be fully expressed in terms of macroscopic, observable variables. It can therefore be decomposed into components that are dimensionless in a system of macroscopic variables, and is rescaled in a controllable way when the parameters of the space or time discretization vary. Furthermore, as it is shown on selected CFD applications, the resulting numerical model is more stable and accurate than the original one. The ability to rescale the variables of the regularized model in a controllable way is exploited in the subsequent chapters. Through a unified approach, it is explained how to implement grid refinement, an adaptive time interval, boundary conditions, initial conditions, and a coupling of the LB model with another CFD tool. In the last chapter finally, a new point of view is adopted, and the focus turns back to the original, non-regularized LB method. It is discussed how the non-hydrodynamical terms can be purposely exploited to create a model of fluid turbulence. The theoretical considerations are accompanied by results of direct numerical simulations of a turbulent fluid.

Chapter 1

Computational fluid dynamics and lattice Boltzmann

1.1 Vector and tensor notation

Throughout this thesis, vectors in the 2D or 3D Euclidean space are noted with an arrow on top of them as in $\vec{a}$, and higher order tensors by a bold face notation as in $\boldsymbol{A}$. Their indexes are labeled by Greek letters, as in a_α and in $A_{\alpha\beta}$. Two types of notation are used to express algebraic operations on vectors and tensors. In the index notations, all indexes are explicited, but the sums are skipped: it is implicitly understood that they must be taken whenever an index is repeated twice in the same term. With this notation, a scalar product between $\vec{a}$ and $\vec{b}$ is written as $\lambda = a_\alpha\, b_\alpha$. In a full vector notation, the indexes are skipped, and the scalar product is written as $\lambda = \vec{a} \cdot \vec{b}$. The shorter and more readable vector/tensor notation is generally preferred. The more explicit index notation is however still used in complicated expressions involving higher order tensors, to make sure they are properly interpreted. The following list shows through several examples how operations between vectors and tensors are denoted:

	Vector/tensor notation	Index notation
Scalar product	$\lambda = \vec{a} \cdot \vec{b}$	$\lambda = a_\alpha b_\alpha$
Tensor contraction	$\lambda = \boldsymbol{A} : \boldsymbol{B}$	$\lambda = A_{\alpha\beta} B_{\alpha\beta}$
Dyadic vector product	$\boldsymbol{A} = \vec{a}\vec{b}$	$A_{\alpha\beta} = a_\alpha b_\beta$
Gradient	$\vec{a} = \overrightarrow{\nabla} \lambda$	$a_\alpha = \partial_\alpha \lambda$
Divergence of vector	$\lambda = \overrightarrow{\nabla} \cdot \vec{a}$	$\lambda = \partial_\alpha a_\alpha$
Divergence of tensor (order 2)	$\vec{a} = \overrightarrow{\nabla} \cdot \boldsymbol{A}$	$a_\alpha = \partial_\beta A_{\alpha\beta}$
Divergence of tensor (order 3)	$\boldsymbol{A} = \overrightarrow{\nabla} \cdot \boldsymbol{T}$	$A_{\alpha\beta} = \partial_\gamma T_{\alpha\beta\gamma}$

Another vector space, the space $\mathbb{R}^q$ of the particle distribution functions, sometimes appears in the thesis. In order to clearly distinguish between the two types of vectors, the indexes of vectors in $\mathbb{R}^q$ are labeled by a Latin index, and none of the shorthand notations introduced above is used for them. Occasionally, a scalar product in $\mathbb{R}^q$ is executed, for which a bracket notation $< \cdot | \cdot >$ is used. Furthermore, Section 3.3.1 makes a short exception and uses a full vector notations for the vectors in $\mathbb{R}^q$, in order to underline the origin of the computations as linear algebra

expressions.

1.2 Macroscopic equations

1.2.1 Governing equations

Computational Fluid Dynamics (CFD) provides a qualitative, and in some cases even quantitative, prediction of fluid flows by means of mathematical and numerical tools. The chain leading to a full numerical model includes the following steps:

The mathematical model consists most often of a set of partial differential equations (PDE's). They describe the evolution of the observable, physical quantities in the flow and are based on phenomenological and/or theoretical considerations. As an alternative, a simplified microscopic model can be directly simulated on a computer, without referring explicitly to a PDE.

The Numerical method provides a way to find approximate solutions of the PDE, or to solve the dynamics of a microscopic model, with the help of computer simulations.

Some Software tools , consisting of specialized algorithms, are used to implement parts of the numerical model. Pre- and postprocessing utilities are required to handle the setup of a simulation and the analysis of the simulated data.

The present section introduces the most commonly used mathematical model, which consists of a set of PDE's. It describes the fluid at a continuum level, at which its particulate nature can be neglected. Instead, one takes interest in the dynamics of a reduced set of *macroscopic variables* that describe the state of the fluid sufficiently well. Whenever the fluid exhibits a physical behavior that cannot be described by the macroscopic model, new macroscopic variables and PDE's are introduced to adapt the model. This technique is opposed to an approach in which one tries to obtain an understanding of the microscopic nature of the fluid. In that case, the observed macroscopic physics of the fluid is an emergent property that arises naturally from the microscopic model.

The state of a simple fluid is described by the following macroscopic variables:

the fluid density	ρ,	
the flow velocity	$\vec{u}$,	
the pressure	p,	
the energy	E	and
the temperature	T.	

The governing equations for those variables can be derived by methods of statistical physics from the equations of motion of the microscopic model. Alternatively, they can be interpreted as conservation laws for the concerned variables. To reflect this fact, they are written in a divergence form:

$$\frac{\partial \mathcal{U}}{\partial t} + \overrightarrow{\nabla} \cdot \mathcal{F} = \mathcal{Q}. \tag{1.1}$$

The most commonly used equations express the conservation of

$$\begin{aligned}
&\text{mass:} && \partial_t \rho && + \overrightarrow{\nabla} \cdot (\rho \vec{u}) && = 0, && (1.2)\\
&\text{momentum:} && \partial_t (\rho \vec{u}) && + \overrightarrow{\nabla} \cdot (\rho \vec{u}\vec{u} + p\boldsymbol{I} - \boldsymbol{\tau}) && = \rho \vec{g} \quad \text{and} && (1.3)\\
&\text{energy:} && \partial_t E && + \overrightarrow{\nabla} \cdot \left((\rho E + p)\vec{u} - \kappa \overrightarrow{\nabla} T - (\vec{u} \cdot \boldsymbol{\tau})^T \right) && = \rho (q + \vec{g} \cdot \vec{u}). && (1.4)
\end{aligned}$$

The parameter κ stands for the thermal conductivity, and q for internal heat sources. The tensor $\boldsymbol{\tau}$ represents the deviatoric stresses in the fluid. For a Newtonian fluid it has, by definition, the following form:

$$\boldsymbol{\tau} = -\nu' \rho (\overrightarrow{\nabla} \cdot u)\, \boldsymbol{I} + 2\, \rho\, \nu\, \boldsymbol{S}, \tag{1.5}$$

where the *strain rate tensor* $\boldsymbol{S}$ is defined as

$$\boldsymbol{S} = \frac{1}{2} (\overrightarrow{\nabla} \vec{u} + (\overrightarrow{\nabla} \vec{u})^T). \tag{1.6}$$

The parameters ν and ν' stand for the dynamic shear and bulk viscosity respectively. Based on a microscopic model, in which the molecules of a gas are represented by a single rigid sphere, one can derive the following theoretical relationship between those parameters (see *e.g.* [1]):

$$\nu' = \frac{2}{3}\, \nu. \tag{1.7}$$

In real gases and liquids however, the actual value of ν' can vary substantially from this prediction. Equation (1.3) is commonly known as the *compressible Navier-Stokes equation*.

Eqs. (1.2,1.3,1.4) represent a system of $d+2$ equations for $d+4$ unknowns, in a d-dimensional space. Is must be closed by additional equations of state that relate the pressure with the temperature and the temperature with the energy.

1.2.2 Isothermal and incompressible flows

In an isothermal flow, effects of the temperature are neglected, and Eq. 1.4 is not taken into account. In that case, an equation of state must be devised that relates the pressure and the density in the fluid. In the following, an equation for ideal gases is used, in which the pressure scales linearly with the density:

$$p = c_s^2\, \rho, \tag{1.8}$$

where c_s is the speed of sound.

In an incompressible fluid, the density takes a constant value $\rho = \rho_0$ which does not vary in time and space. The conservation law for the mass is simplified, and states that the velocity field is divergence-less (solenoidal):

$$\overrightarrow{\nabla} \cdot \vec{u} = 0. \tag{1.9}$$

Eq. (1.9) is often used as a definition for fluid incompressibility. The conservation law for the momentum is also simplified and leads to the incompressible Navier-Stokes equation, often called "Navier-Stokes equation" for short:

$$\partial_t \vec{u} + (\vec{u} \cdot \vec{\nabla})\vec{u} = -\frac{1}{\rho_0}\overrightarrow{\nabla} p + \nu \nabla^2 \vec{u}. \tag{1.10}$$

In order to find the value of p, one takes the divergence of Eq. (1.10) and makes use of Eq. (1.9). This yields the *Poisson equation*:

$$\nabla^2 p = -\rho_0 (\overrightarrow{\nabla}\vec{u}) : (\overrightarrow{\nabla}\vec{u})^T. \tag{1.11}$$

This time-independent equation replaces Eq. (1.9). When the evolution of an incompressible flow is computed numerically, Eq. (1.11) needs to be solved by an iterative procedure at every discrete time step. It adjusts the value of the pressure in such a way that the velocity field remains divergence-less during its time evolution.

1.2.3 Dimensionless formulation

Before the governing equations for fluid motion can be solved on a computer, one needs to get rid of the physical units of the macroscopic variables. This leads to a set of PDE's that act on dimensionless variables, and whose properties are tuned via generic, dimensionless parameters. For the purpose of illustration, Eqs. (1.9) and (1.10) are now cast in a dimensionless form. For this, a length scale l_0 and a time scale t_0 are introduced that are representative for the flow configuration. The length l_0 could for example stand for the size of an obstacle which is immersed in the fluid, and t_0 could be the time needed by a passive scalar in the fluid to travel a distance l_0. The physical variables for time and position, t and $\vec{r}$, are replaced by their dimensionless counterpart $t^* = t/t_0$ and $\vec{r}^* = \vec{r}/l_0$. In the same manner, a change of variables for $\vec{u} = l_0/t_0 \vec{u}^*$, $\partial_t = 1/t_0 \partial_t^*$, $\overrightarrow{\nabla} = 1/l_0 \overrightarrow{\nabla}^*$ and $p = \rho_0 l_0^2/t_0^2 p^*$ is performed. This leads to the following dimensionless Navier-Stokes equation:

$$\partial_t^* \vec{u}^* + (\vec{u}^* \cdot \overrightarrow{\nabla}^*)\vec{u}^* = -\overrightarrow{\nabla}^* p^* + \frac{1}{Re}\nabla^{2^*}\vec{u}^* \quad \text{and} \tag{1.12}$$

$$\overrightarrow{\nabla}^* \cdot \vec{u}^* = 0, \tag{1.13}$$

where $Re = l_0^2/(t_0\,\nu)$ is the dimensionless *Reynolds number*. Two flows that obey the same Navier-Stokes equation and which have the same Reynolds number are equivalent. In this way, it is possible to solve the dimensionless equations on a computer, and to relate the results with a physical flow that possesses the same Reynolds number and the same flow geometry.

1.2.4 Further reading

A large number of books discuss the numerical resolution of PDE's and the topic of computational fluid dynamics. Only a few are mentioned here to guide the interested reader.

A well written and theoretically sound introduction to the numerical treatment of PDE's can be found in Ref. [2]. A general introduction to the problems of computational fluid dynamics is available on the web presented as a one-semester introduction course [3]. An undergraduate-level, practically oriented introduction to computational fluid dynamics with a specific emphasis on the finite difference method is presented in Ref. [4]. Reference [5] finally gives an extended overview on various methods of computational fluid dynamics in general, and Ref. [6] on the finite element method in particular.

1.3 Lattice Boltzmann method

1.3.1 Discrete variables

The problems of fluid dynamics possess an infinite number of degrees of freedom, because the considered quantities (velocity, pressure, *etc.*) are not just single values, but spatially extended fields. Computers can however only handle a finite amount of variables, and represent them with a finite accuracy. The spatially extended fields must therefore be replaced by a finite set of scalar values that are appropriate for the numerical investigation of the problem. This step is called the *space discretization* of the problem. To keep the discussion simple, it is restricted to problems whose domain of definition is confined within a regular box of finite extent. A possible way to discretize this space is to partition it into cells. In that case, a numerical representation of the fluid consists of a set of numbers, of which each stands for the average value of a fluid variable over a cell. Another approach to space discretization consists in replacing spatially extended fields by their value on a chosen finite population of space points. The lattice Boltzmann method, as it is presented here, adopts the latter point of view. The space is hence discretized on a regular lattice with fixed grid spacing. The present discussion concentrates on a cubic subsample of a three-dimensional system whose size is $l_0 \times l_0 \times l_0$ (l_0 is a characteristic length of the system, introduced in Section 1.2.3). The coordinate system is defined in such a way that one corner of the cube is situated at the origin, and the opposite corner at the position $l_0\,\vec{e}_0 + l_0\,\vec{e}_1 + l_0\,\vec{e}_2$. The cube is represented numerically by N^3 points $\vec{r}_{ijk}$, with $i, j, k = 0 \cdots N-1$, situated at the position $\vec{r}_{ijk} = i\vec{e}_0\,\delta_x + j\vec{e}_1\,\delta_x + k\vec{e}_2\,\delta_x$. Those positions are called the *lattice sites*. The *lattice spacing* $\delta_x = 1/(N-1)$ is equal to the distance between two neighboring lattice sites. A scalar field $\mu(\vec{r}, t)$ is represented by N^3 values $\mu_{ijk}(t)$ that are numerical approximations of $\mu(\vec{r}_{ijk}, t)$.

The time axis is discretized in the same manner by a set of equidistributed finite time steps $t_n = n\,\delta_t$. In the numerical representation that is adopted here, a time-dependent variable $f(t)$ is replaced by a numerical approximation $f_n \approx f(t_i)$. Finally, to combine the notation for space and time discretization, the somewhat obscure index notation is again skipped for the more readable notation with parentheses. That is, the expression $f(\vec{r}_{ijk}, t_n)$ is used to represent the numerical approximation (and not the exact value) of f at the position $\vec{r}_{ijk}$ and at the time step t_n.

1.3.2 Lattice Boltzmann model

Each node of a lattice Boltzmann simulation holds a set of q variables f_i, $i = 0 \cdots q-1$, called the *particle distribution functions*. Each of those functions is responsible for carrying information from a node to one of its neighbors. The position of this neighbor is defined by a vector $\vec{c}_i$, which points from a lattice node to the corresponding lattice node. It is characteristic for the lattice and does not depend on space or time. The following general rule for the evolution of a LB model explicits the role of the vector $\vec{c}_i$:

$$f_i(\vec{r}+\vec{c}_i, t+1) - f_i(\vec{r}, t) = \Omega_i. \tag{1.14}$$

This equation has been written in *lattice units*, in which the spacing between two adjacent lattice nodes, as well as the time interval from one iteration to the next, are unitary. In this way, a close relationship between the theory and the application is kept, as it is common to write numerical implementations of LB models in lattice units. This approach is opposed to the one of other numerical models that are most often implemented in a system of dimensionless units introduced in Section 1.2.3). The term Ω_i on the right hand side of Eq. 1.14 is the *collision operator* that describes how the q values f_i defined on the same node at given time step interact. The macroscopic variables, the density ρ and the velocity $\vec{u}$ are defined locally as moments of the distribution functions:

$$\rho = \sum_{i=0}^{q-1} f_i \quad \text{and} \tag{1.15}$$

$$\vec{u} = \frac{1}{\rho} \sum_{i=0}^{q-1} \vec{c}_i f_i. \tag{1.16}$$

In the following, whenever a sum over all distribution functions is taken, the range of the variable i is not explicited any more to keep the notation short: the sum over the q elements of a variable E_i, $\sum_{i=0}^{q-1} E_i$, is simply written as $\sum_i E_i$.

The most commonly used collision operator, known under the name of *BGK collision* and discussed extensively in Ref. [7], implements a relaxation dynamics towards a local equilibrium with a relaxation parameter ω:

$$\Omega_i^{BGK} = -\omega(f_i - f_i^{eq}). \tag{1.17}$$

The local equilibrium is defined as

$$f_i^{(eq)} = \rho\, t_i \left(1 + \frac{1}{c_s^2}\vec{c}_i \cdot \vec{u} + \frac{1}{2c_s^4}\boldsymbol{Q}_i : \vec{u}\vec{u}\right), \tag{1.18}$$

and the tensor $\boldsymbol{Q}$ is defined as follows:

$$Q_i = \vec{c}_i\vec{c}_i - c_s^2\boldsymbol{I}. \tag{1.19}$$

The constant c_s is the speed of sound of the model. This parameter, as well as the q parameters t_i are lattice constants. Conceptually speaking, there is some freedom in the choice of the constant c_s. All that needs to be done is to modify the value of

the rest particle weight t_0 correspondingly, in order to recover the lattice symmetries described in Section 2.1. In practice, a value of $c_s^2 = 1/3$ is found to be numerically most stable, and this choice is therefore most commonly adopted. In conclusion, there are two ingredients to the definition of a lattice Boltzmann model: on the one hand the details of the collision operator Ω_i, and on the other hand the structure of the lattice, defined by the constants q, $\vec{c}_i$, c_s and t_i. Some of the most commonly used lattice structures are summarized in Section 1.3.3.

The numerical model defined by Eqs.(1.14,1.18) solves asymptotically the equations of motion for a compressible fluid, when the discrete steps δ_t and δ_x, as well as the relative velocity $\vec{u}/c_s$ are small enough. This fact is proved with some reservations in Section 2.3. It is also shown that the relaxation parameter ω is directly related to the dynamic shear viscosity ν of the fluid via Eq. 2.73:

$$\nu = c_s^2 \left(\frac{1}{\omega} - \frac{1}{2} \right). \tag{1.20}$$

The formulas of the lattice Boltzmann method can be derived theoretically via different approaches. One approach, advocated for example in Ref. [8], views the LB method as a continuous version of a microscopic model known as a *Cellular Automata*. In another approach, described for example in Ref. [9], the dynamics of the LB method is derived from the continuum Boltzmann equation. This equation considers the movement of molecules in a gas and describes their behavior statistically at a continuum level (see *e.g.* Refs. [10, 1]). For this reason, the theory behind the Boltzmann equation is often called *kinetic theory*. By extension, the LB method is sometimes called a *lattice kinetic scheme*, and the quantities f_i used in the method are referred to as *kinetic variables*.

This concludes the presentation of the lattice Boltzmann method. An overview of lattice structures that can be used in common with the BGK model is presented in the next section. A discussion of how to choose the system of units and relate the lattice variables to macroscopic ones is found in Section 2.4. For a more detailed discussion of implementation issues related to the LB method, the reader is referred to Refs. [11, 12].

1.3.3 Lattice structures

A lattice structure with q lattice directions, defined on a d-dimensional space, is commonly identified by the name "DdQq lattice". A few 2D and 3D lattice structures that are commonly used for isothermal flows are listed in this section. For those flows, it is sufficient to consider structures in which the lattice vectors $\vec{c}_i$ point to the neighbors included in the immediate neighborhood, so-called *nearest neighbors*. The lattice weights t_i are used to account for the fact that not all lattice vectors have the same length. Each lattice structure has three types of lattice weights, the weight t_0 corresponding to the zero velocity $\vec{c}_0 = 0$, the weight t_s corresponding to the short velocities and the weight t_l corresponding to the long velocities. In 2D for example, the short velocities, parallel to the lattice edges are of length 1, and the long velocities following diagonal directions are of length $\sqrt{2}$.

D2Q9 lattice

$c_s^2 = \frac{1}{3}$

$t_0 = \frac{4}{9} \quad t_s = \frac{1}{9} \quad t_l = \frac{1}{36}$

$\vec{c}_0 = (0,0)$

$\vec{c}_1 = (-1,1) \quad \vec{c}_2 = (-1,0) \quad \vec{c}_3 = (-1,-1) \quad \vec{c}_4 = (0,-1)$

$\vec{c}_5 = (1,-1) \quad \vec{c}_6 = (1,0) \quad \vec{c}_7 = (1,1) \quad \vec{c}_8 = (0,1)$

D3Q15 lattice

$c_s^2 = \frac{1}{3}$

$t_0 = \frac{2}{9} \quad t_s = \frac{1}{9} \quad t_l = \frac{1}{72}$

$\vec{c}_0 = (0,0,0)$

$\vec{c}_1 = (-1,0,0) \quad \vec{c}_2 = (0,-1,0) \quad \vec{c}_3 = (0,0,-1)$

$\vec{c}_4 = (-1,-1,-1) \quad \vec{c}_5 = (-1,-1,1) \quad \vec{c}_6 = (-1,1,-1) \quad \vec{c}_7 = (-1,1,1)$

$\vec{c}_8 = (1,0,0) \quad \vec{c}_9 = (0,1,0) \quad \vec{c}_{10} = (0,0,1)$

$\vec{c}_{11} = (1,1,1) \quad \vec{c}_{12} = (1,1,-1) \quad \vec{c}_{13} = (1,-1,1) \quad \vec{c}_{14} = (1,-1,-1)$

D3Q19 lattice

$c_s^2 = \frac{1}{3}$

$t_0 = \frac{1}{3} \quad t_s = \frac{1}{18} \quad t_l = \frac{1}{36}$

$\vec{c}_0 = (0,0,0)$

$\vec{c}_1 = (-1,0,0) \quad \vec{c}_2 = (0,-1,0) \quad \vec{c}_3 = (0,0,-1)$

$\vec{c}_4 = (-1,-1,0) \quad \vec{c}_5 = (-1,1,0) \quad \vec{c}_6 = (-1,0,-1)$

$\vec{c}_7 = (-1,0,1) \quad \vec{c}_8 = (0,-1,-1) \quad \vec{c}_9 = (0,-1,1)$

$\vec{c}_{10} = (1,0,0) \quad \vec{c}_{11} = (0,1,0) \quad \vec{c}_{12} = (0,0,1)$

$\vec{c}_{13} = (1,1,0) \quad \vec{c}_{14} = (1,-1,0) \quad \vec{c}_{15} = (1,0,1)$

$\vec{c}_{16} = (1,0,-1) \quad \vec{c}_{17} = (0,1,1) \quad \vec{c}_{18} = (0,1,-1)$

D3Q27 lattice

The D3Q27 lattice uses lattice vectors of three different lengths: the short ones, such as $(1,0,0)$ with weight t_s, the medium ones like $(1,1,0)$ with weight t_m and the long ones like $(1,1,1)$ with weight t_l.

$c_s^2 = \frac{1}{3}$

$t_0 = \frac{8}{27} \quad t_s = \frac{2}{27} \quad t_m = \frac{1}{54} \quad t_l = \frac{1}{216}$

$$
\begin{array}{llll}
\vec{c}_0 = (0,0,0) & & & \\
\vec{c}_1 = (-1,0,0) & \vec{c}_2 = (0,-1,0) & \vec{c}_3 = (0,0,-1) & \\
\vec{c}_4 = (-1,-1,0) & \vec{c}_5 = (-1,1,0) & \vec{c}_6 = (-1,0,-1) & \\
\vec{c}_7 = (-1,0,1) & \vec{c}_8 = (0,-1,-1) & \vec{c}_9 = (0,-1,1) & \\
\vec{c}_{10} = (-1,-1,-1) & \vec{c}_{11} = (-1,-1,1) & \vec{c}_{12} = (-1,1,-1) & \vec{c}_{13} = (-1,1,1) \\
\vec{c}_{14} = (1,0,0) & \vec{c}_{15} = (0,1,0) & \vec{c}_{16} = (0,0,1) & \\
\vec{c}_{17} = (1,1,0) & \vec{c}_{18} = (1,-1,0) & \vec{c}_{19} = (1,0,1) & \\
\vec{c}_{20} = (1,0,-1) & \vec{c}_{21} = (0,1,1) & \vec{c}_{22} = (0,1,-1) & \\
\vec{c}_{23} = (1,1,1) & \vec{c}_{24} = (1,1,-1) & \vec{c}_{25} = (1,-1,1) & \vec{c}_{26} = (1,-1,-1)
\end{array}
$$

1.3.4 Further reading

It is more difficult to find well written textbooks on the lattice Boltzmann method than on other topics of computational fluid dynamics. A few are however worth mentioning and are listed in the following.

Some general information on the LB method, links to further literature and an open source LB simulation code are found on the web page of the `OpenLB` project [13]. As interesting introductory textbooks with some theoretical backgrounds Refs. [8, 14, 1] should be mentioned. The Refs. [11, 12] complete the list with a more practically oriented approach and a discussion of implementation details. An overview of LB methods, including multi relaxation time models, is contained in the review paper [9], of which a reprint can be downloaded from the internet.

Chapter 2

Multiscale Chapman-Enskog analysis

2.1 Series expansion with scale separation

In this section, the generic evolution equation of LB methods is inspected by means of a truncated Taylor expansion and a multi-scale analysis. The results of this study can be applied to show that a specific LB method solves asymptotically the dynamics of the desired macroscopic PDE. In the present thesis, they are furthermore used systematically, whenever a LB model needs to be adapted to a specific situation. All developments contained in this section are largely inspired from corresponding sections in Ref. [8].

2.1.1 Lattice symmetries

Not all lattice types are adequate for the execution of a lattice Boltzmann simulation. In order for the simulation to yield the desired asymptotic PDE, the lattice must verify a set of symmetry conditions. The BKG model for a fluid for example requires that there exists a constant c_s and a set of weights t_i for the lattice velocities $\vec{c}_i$ such that the following relations are verified:

$$\begin{aligned} &(a)\sum_i t_i = 1 \qquad (c)\sum_i t_i c_{i\alpha}c_{i\beta} = c_s^2\delta_{\alpha\beta} \qquad (e)\sum_i t_i c_{i\alpha}c_{i\beta}c_{i\gamma}c_{i\delta} = \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad c_s^4\left(\delta_{\alpha\beta}\delta_{\gamma\delta}+\delta_{\alpha\gamma}\delta_{\beta\delta}+\delta_{\alpha\delta}\delta_{\beta\gamma}\right) \\ &(b)\sum_i t_i c_{i\alpha} = 0 \qquad (d)\sum_i t_i c_{i\alpha}c_{i\beta}c_{i\gamma} = 0 \qquad (f)\sum_i t_i c_{i\alpha}c_{i\beta}c_{i\gamma}c_{i\delta}c_{i\epsilon} = 0. \end{aligned} \tag{2.1}$$

The value of c_s may vary from one lattice to another. In the following, this parameter will be shown to be equal to the speed of sound in a simulation. The symmetries displayed in Eq. (2.1) are present in all types of lattices listed in Section 1.3.3.

Note 2.1: Extended lattice structures *The lattice symmetries shown in Eq. (2.1) are milestones to the theoretical developments that follow and are fundamental to the presented BGK lattice Boltzmann models. There exist however specific lattice Boltzmann models that work on lattices with weaker symmetries. A case in point is the model presented in Ref. [15] that works fine on a* 13*-velocity*

3D lattice (D3Q13), although this lattice lacks sufficient symmetries to be used with BKG dynamics. On the other hand, there exist models that require additional symmetry properties to those of Eq. (2.1). So-called thermal LB models for example use an extended set of lattice velocities that include interaction with next-to-nearest neighbors.

Equations (2.1) are sometimes viewed as orthogonality properties between *lattice vectors* defined on the vector space $\mathbb{R}^q$. The first of those lattice vectors is called $\mathcal{C}^0$ and is defined through

$$\mathcal{C}^0_i = 1 \quad \forall i = 0 \cdots q. \tag{2.2}$$

It follows a set of d vectors $\mathcal{C}^1_\alpha$ defined as

$$\mathcal{C}^1_{i,\alpha} = c_{i\alpha} \quad \text{for} \quad \alpha = 0 \cdots d-1 \quad \text{and} \quad i = 0 \cdots q. \tag{2.3}$$

Finally, a set of d^2 lattice vectors $\mathcal{C}^2_{\alpha\beta}$ is introduced that obey the relation

$$\mathcal{C}^2_{i,\alpha\beta} = Q_{i\alpha\beta} \quad \text{for} \quad \alpha, \beta = 0 \cdots d-1 \quad \text{and} \quad i = 0 \cdots q, \tag{2.4}$$

where the tensor $\boldsymbol{Q}$ is defined in Eq. (1.19). It is pointed out that by symmetry of the tensor $\boldsymbol{Q}$, the vectors $C^2_{i\alpha\beta}$ verify the relation $\mathcal{C}^2_{i\alpha\beta} = \mathcal{C}^2_{i\beta\alpha}$. Equation (2.4) defines therefore only $d(d+1)/2$ independent vectors, instead of d^2. A scalar product between two vectors a and b in $\mathbb{R}^q$ is finally defined as follows:

$$\langle a|b\rangle = \sum_i t_i \, a_i b_i. \tag{2.5}$$

With this, it is easily concluded from Eq. (2.1) that the lattice vectors defined above are orthogonal to each other, but not necessarily unitary:

$$\langle \mathcal{C}^0|\mathcal{C}^1_\alpha\rangle = 0, \tag{2.6}$$
$$\langle \mathcal{C}^0|\mathcal{C}^2_{\alpha\beta}\rangle = 0, \tag{2.7}$$
$$\langle \mathcal{C}^1_\alpha|\mathcal{C}^2_{\beta\gamma}\rangle = 0, \tag{2.8}$$
$$\langle \mathcal{C}^0|\mathcal{C}^0\rangle = 1, \tag{2.9}$$
$$\langle \mathcal{C}^1|\mathcal{C}^1\rangle = c_s^2 \boldsymbol{I} \quad \text{and} \tag{2.10}$$
$$\langle \mathcal{C}^2_{\alpha\beta}|\mathcal{C}^2_{\gamma\delta}\rangle = c_s^4 \, (\delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}). \tag{2.11}$$

The fact that Eq. (2.11) expresses an orthogonality relation between all vectors of the family $\mathcal{C}^2$ becomes clear when the tensor resulting from a scalar product between two of those vectors is contracted with an arbitrary tensor $\boldsymbol{T}$:

$$\langle \mathcal{C}^2_{\alpha\beta}|\mathcal{C}^2_{\gamma\delta}\rangle \, T_{\gamma\delta} = c_s^4 \, (T_{\alpha\beta} + T_{\beta\alpha}) \tag{2.12}$$

The q-dimensional vector space introduced in the previous paragraph is of both theoretical and technical interest. To see this, the *unweighted particle distribution functions*

$$g_i = f_i / t_i \tag{2.13}$$

are introduced. They are interpreted as vectors in $\mathbb{R}^q$, and the index i is consequently skipped. All hydrodynamic variables in a LB model, which were previously introduced as moments of the distribution functions, are now reinterpreted as projections of g on the lattice vectors. The mass density for example is computed as $\rho = \langle \mathcal{C}^0 | g \rangle$, and the momentum as $\rho \vec{u} = \langle \mathcal{C}^1 | g \rangle$. Most of the algebra shown in the next section can be viewed as the computation of similar projections. This point of view is useful, because vanishing terms in the computation are at once identified as projections between orthogonal vectors.

Note 2.2: Hermite polynomials

The vectors $\mathcal{C}^0$, $\mathcal{C}^1$ and $\mathcal{C}^2$ introduced in this section can be viewed as tensor Hermite polynomials. In some theoretical approaches to the LB method, the equilibrium distribution function in Eq. (1.18) is derived by means of a truncated expansion of the velocity in tensor Hermite polynomials. The key ideas of this approach are developed in Ref. [16].

2.1.2 Multi-scale expansion

LB models with a generic collision operator Ω are defined by the evolution equation (1.14) on page 10. The left-hand side of this equation can be expanded in a second-order Taylor series as follows:

$$\begin{aligned} \Omega_i &= f_i(\vec{r} + \vec{c}_i, t+1) - f_i(\vec{r}, t) \\ &\approx (\partial_t + \vec{\nabla} \cdot \vec{c}_i)\, f_i + \frac{1}{2}(\partial_t^2 + 2\partial_t \vec{\nabla} \cdot \vec{c}_i + \overrightarrow{\nabla}\overrightarrow{\nabla} : \vec{c}_i \vec{c}_i)\, f_i. \end{aligned} \tag{2.14}$$

In order to relate the LB equation with a macroscopic PDE, it is necessary to formally separate different time scales. In this way, physical phenomena occurring at different scales are discussed separately and contribute individually to the final equations of motion. To obtain this, the time derivative is expanded in terms of a formal, "small" parameter ϵ:

$$\partial_t = \epsilon\, \partial_{t1} + \epsilon^2 \partial_{t2} + \mathcal{O}(\epsilon^3). \tag{2.15}$$

The idea behind this expansion is that all involved terms (∂_{t1} and ∂_{t2}) are of the same order of magnitude, and the smallness is introduced via the parameter ϵ. The space derivative is not expanded beyond its first-order term:

$$\overrightarrow{\nabla} = \epsilon \overrightarrow{\nabla}^{(1)} + \mathcal{O}(\epsilon^2). \tag{2.16}$$

The particle distribution functions are likewise expanded, starting with a zeroth-order contribution $f^{(0)}$:

$$f_i = f_i^{(0)} + \epsilon f_i^{(1)} + \epsilon^2 f_i^{(2)} + \mathcal{O}(\epsilon^3). \tag{2.17}$$

It is concluded from Eq. (2.14) that the collision term Ω_i does not contain constant contributions with respect to the parameter ϵ: $\Omega_i^{(0)} = 0$. It is therefore expanded as follows, starting with the $\mathcal{O}(\epsilon)$ term:

$$\Omega_i = \epsilon \Omega_i^{(1)} + \epsilon^2 \Omega_i^{(2)} + \mathcal{O}(\epsilon^3). \tag{2.18}$$

This expansion, up to a second order accuracy with respect to ϵ, appears to be sufficient to find the Navier-Stokes equation. However, higher order terms, known under the name of *Burnett terms* are present in the LB dynamics and can be computed. In the following, all developments are written down separately for the $\mathcal{O}(\epsilon)$ and the $\mathcal{O}(\epsilon^2)$ scales, and all terms of the order $\mathcal{O}(\epsilon^3)$ or higher are neglected.

The scale separated version of Eq. (2.14) reads:

$$\epsilon\,\Omega_i^{(1)}+\epsilon^2\Omega_i^{(2)} = (\epsilon\,\partial_{t1}+\epsilon^2\partial_{t2}+\epsilon\vec{\nabla}_1\cdot\vec{c}_i+\frac{1}{2}\epsilon^2\partial_{t2}^2+\epsilon^2\partial_{t1}\vec{\nabla}_1\cdot\vec{c}_i+\frac{1}{2}\epsilon^2\overrightarrow{\nabla}_1\overrightarrow{\nabla}_1 : \vec{c}_i\vec{c}_i)(f_i^{(0)}+\epsilon\,f_i^{(1)}). \tag{2.19}$$

Note 2.3: Knudsen number

The parameter ϵ is often identified as the Knudsen number, that is, the ratio λ_f/l_0 between the mean free path of a gas molecule and a macroscopic length scale. This terminology is motivated by a dimensional analysis of the Boltzmann equation. Indeed, when all terms of this equation are made dimensionless with respect to microscopic scales, the collision term is left with a trailing factor λ_f/l_0 (see e.g. Ref. [14]). As a result of the truncated multi-scale expansion, the LB equation (as well as the Navier-Stokes equation) is valid only for low Knudsen number flows. It does not apply well for example to dilute gases or microfluids.

2.1.3 Conservation laws

The macroscopic variables of the flow are defined as moments of the particle distribution functions, that is, as the projection of the unweighted distribution functions g_i on the base vectors $\mathcal{C}^0$, $\mathcal{C}^1$ and $\mathcal{C}^2$. This leads to the definition of the zeroth-order scalar moment ρ, the first-order vector moment $\vec{j}$ and the second order tensor moment $\mathbf{\Pi}$:

$$\rho = \sum_i f_i, \tag{2.20}$$

$$\vec{j} = \sum_i \vec{c}_i f_i \quad \text{and} \tag{2.21}$$

$$\mathbf{\Pi} = \sum_i \boldsymbol{Q}_i f_i. \tag{2.22}$$

The dynamics of a flow can be expressed by means of conservation laws, applied to some of those moments. A global conservation of a macroscopic quantity is expressed locally by a collision invariant. As an example, conservation of mass in a fluid is enforced by a local conservation of mass during the collision between particles. At the level of the LB equations, a collision invariance means that the projection of the unweighted collision operator Ω_i/t_i on the corresponding base vector vanishes.

Zeroth order moment balance. Conservation of the zeroth order moment is ensured by the collision invariant

$$\sum_i \Omega_i = 0, \tag{2.23}$$

and by the requirement that first-order contributions to ρ vanish:

$$\rho = \sum_i f_i = \sum_i f_i^{(0)}. \tag{2.24}$$

This latter condition can be understood intuitively as follows: the conserved variables are moments of $f^{(0)}$ only, whereas the collision operator acts on $f^{(1)}$, which leaves the conserved variables unchanged during the collision. Alternatively, Eqs. (2.23) and (2.24) can simply be taken as technical requirements that serve the final aim, namely to find a LB scheme leading to the Navier-Stokes equation. In Sections (2.2) and (2.3), LB collision operators are explicitly such as to respect the conservation laws.

Expanding Eq. (2.23) on two different scales ϵ and ϵ^2, and using Eq. (2.19), yields

$$\sum_i \Omega_i^{(1)} = \partial_{t1} \sum_i f_i^{(0)} + \overrightarrow{\nabla}_1 \cdot \sum_i \vec{c}_i f_i^{(0)} = \partial_{t1}\rho + \overrightarrow{\nabla}_1 \cdot \vec{j}^{(0)} = 0 \tag{2.25}$$

and

$$\begin{aligned}\sum_i \Omega_i^{(2)} &= \partial_{t1} \sum_i f_i^{(1)} + \partial_{t2} \sum_i f_i^{(0)} + \overrightarrow{\nabla}_1 \cdot \sum_i \vec{c}_i f_i^{(1)} + \frac{1}{2}\partial_{t1}^2 \sum_i f_i^{(0)} \\ &\quad + \partial_{t1} \overrightarrow{\nabla}_1 \cdot \sum_i \vec{c}_i f_i^{(0)} + \frac{1}{2}\overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \sum_i (\boldsymbol{Q}_i + c_s^2 \boldsymbol{I}) f_i^{(0)} \\ &= \partial_{t2}\rho + \overrightarrow{\nabla}_1 \cdot \vec{j}^{(1)} + \frac{1}{2}\partial_{t1}^2 \rho + \partial_{t1} \overrightarrow{\nabla}_1 \cdot \vec{j}^{(0)} + \frac{1}{2}\overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \boldsymbol{\Pi}^{(0)} + \frac{1}{2} c_s^2 \nabla_1^2 \rho.\end{aligned} \tag{2.26}$$

Equations (2.25) and (2.26) are combined to eliminate the second-order time derivative $\partial_{t1}^2\rho$:

$$\partial_{t2}\rho + \overrightarrow{\nabla}_1 \cdot \vec{j}^{(1)} + \frac{1}{2}\partial_{t1} \overrightarrow{\nabla}_1 \cdot \vec{j}^{(0)} + \frac{1}{2}\overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \boldsymbol{\Pi}^{(0)} + \frac{1}{2} c_s^2 \nabla_1^2 \rho = 0. \tag{2.27}$$

First order moment conservation. At the first order moment, a source term

$$\vec{F} = \epsilon \vec{F}^{(1)} \tag{2.28}$$

is added. It will be identified in the following as an external force acting on a fluid. Depending on the specific requirements for the numerical model, a corresponding source term can of course also be added to the zeroth order moment. This could for example lead to an advection-diffusion equation with source terms. The first order collision invariant reads

$$\sum_i \vec{c}_i \Omega_i = \vec{F}. \tag{2.29}$$

In analogy with the zeroth-order moment, the first-order moment is required to depend only on zeroth-order particle distribution functions. For reasons that become clear shortly, a correction term due to the source $\vec{F}$ must be added:

$$\vec{j} = \sum_i \vec{c}_i f_i = \sum_i \vec{c}_i f_i^{(0)} - \frac{\vec{F}}{2}. \tag{2.30}$$

The $\mathcal{O}(\epsilon)$ scale contribution to Eq. (2.29) reads

$$\begin{aligned}\sum_i \vec{c}_i \Omega_i^{(1)} &= \partial_{t1} \sum_i \vec{c}_i f_i^{(0)} + \overrightarrow{\nabla}_1 \cdot \sum_i (Q_i + c_s^2 \boldsymbol{I}) f_i^{(0)} \\ &= \partial_{t1} \vec{j}^{(0)} + \overrightarrow{\nabla}_1 \cdot \Pi^{(0)} + c_s^2 \overrightarrow{\nabla}_1 \rho = \vec{F}^{(1)}.\end{aligned} \tag{2.31}$$

Two $\mathcal{O}(\epsilon^2)$ terms appearing in Eq. (2.27) can now be evaluated and inserted into this equation:

$$\partial_{t1} \overrightarrow{\nabla}_1 \vec{j}^{(0)} = \overrightarrow{\nabla}_1 \cdot \vec{F}^{(1)} - \overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \boldsymbol{\Pi}^{(0)} - c_s^1 \nabla_1^2 \rho \qquad \text{by (2.31), and} \tag{2.32}$$

$$\overrightarrow{\nabla}_1 \cdot \vec{j}^{(1)} = -\frac{1}{2} \overrightarrow{\nabla}_1 \cdot \vec{F} \qquad \text{by (2.30).} \tag{2.33}$$

Eqs. (2.27), (2.32) and (2.33) are combined to obtain

$$\partial_{t2} \rho = 0, \tag{2.34}$$

which completes Eq. (2.25) to form the continuity equation:

$$\partial_t \rho + \overrightarrow{\nabla} \cdot \vec{j}^{(0)} = 0. \tag{2.35}$$

It is now clear that the particular form of Eq. (2.30), including a force term, was required to cancel erroneous force contributions to the continuity equation.

Contributions of the order $\mathcal{O}(\epsilon^2)$ to Eq. (2.29) yield:

$$\begin{aligned}\sum_i \vec{c}_i \Omega_i^{(2)} &= \partial_{t1} \sum_i \vec{c}_i f_i^{(1)} + \partial_{t2} \sum_i \vec{c}_i f_i^{(0)} + \overrightarrow{\nabla}_1 \sum_i (\boldsymbol{Q}_i + c_s^2 \boldsymbol{I}) f_i^{(1)} \\ &\quad + \frac{1}{2} \partial_{t1}^2 \sum_i \vec{c}_i f_i^{(0)} + \partial_{t1} \overrightarrow{\nabla}_1 \cdot \sum_i (\boldsymbol{Q}_i + c_s^2 \boldsymbol{I}) f_i^{(0)} + \frac{1}{2} \overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \boldsymbol{R}^{(0)} \\ &= \partial_{t1} \vec{j}^{(1)} + \partial_{t2} \vec{j}^{(0)} + \overrightarrow{\nabla}_1 \cdot \boldsymbol{\Pi}^{(1)} + \frac{1}{2} \partial_{t1}^2 \vec{j}^{(0)} + \partial_{t1} \overrightarrow{\nabla}_1 \cdot \boldsymbol{\Pi}^{(0)} \\ &\quad + c_s^2 \partial_{t1} \overrightarrow{\nabla}_1^2 \rho + \frac{1}{2} \overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \boldsymbol{R}^{(0)} = 0,\end{aligned} \tag{2.36}$$

where the tensor $\boldsymbol{R}$ with components $R_{\alpha\beta\gamma} = \sum_i c_{i\alpha} c_{i\beta} c_{i\gamma} f_i$ has been introduced. The second order time derivative is canceled with the help of Eq. (2.31),

$$\partial_{t1}^2 \vec{j}^{(0)} = \partial_{t1} \left(\vec{F}^{(1)} - \overrightarrow{\nabla}_1 \cdot \boldsymbol{\Pi}^{(0)} - c_s^2 \overrightarrow{\nabla}_1 \rho \right), \tag{2.37}$$

which simplifies Eq. (2.36) as follows:

$$\partial_{t2} \vec{j}^{(0)} + \overrightarrow{\nabla}_1 \cdot \boldsymbol{\Pi}^{(1)} + \frac{1}{2} \partial_{t1} \overrightarrow{\nabla}_1 \cdot \boldsymbol{\Pi}^{(0)} + \frac{1}{2} c_s^2 \, \partial_{t1} \overrightarrow{\nabla}_1 \rho + \frac{1}{2} \overrightarrow{\nabla}_1 \overrightarrow{\nabla}_1 : \boldsymbol{R}^{(0)} = 0. \tag{2.38}$$

Finally, the $\mathcal{O}(\epsilon)$ term in Eq. (2.31) and the $\mathcal{O}(\epsilon^2)$ term in Eq. (2.36) are combined and form the full momentum conservation equation:

$$\epsilon \, \partial_t \vec{j}^{(0)} + \epsilon \overrightarrow{\nabla} \cdot \left(\boldsymbol{\Pi} + c_s^2 \, \rho \, \boldsymbol{I} + \frac{\epsilon}{2} \left(\partial_{t1} \left(\boldsymbol{\Pi}^{(0)} + c_s^2 \, \rho \, \boldsymbol{I} \right) + \overrightarrow{\nabla}_1 \cdot \boldsymbol{R}^{(0)} \right) \right) = \epsilon \vec{F}. \tag{2.39}$$

This equation is written in a divergence form, just like Eq. (1.1) on page 6. In Section 2.3, a collision operator is developed in such a way that an exact match between the terms in Eq. (2.39) and the macroscopic conservation equation (1.3) is obtained. More precisely, the $\mathcal{O}(\epsilon)$ term $\boldsymbol{\Pi}^{(0)} + c_s^2 \, \rho \, \boldsymbol{I}$ is identified with $\rho \, \vec{u}\vec{u} + p \, \boldsymbol{I}$, and the remaining $\mathcal{O}(\epsilon^2)$ terms with $-\boldsymbol{\tau}$.

Note 2.4: Conservation laws *A striking feature of LB models is the fact that they implement conservation laws without error. Indeed, the conservation laws prescribed by Eqs. (2.23) and (2.29) can be interpreted as follows: the conserved variables ρ and $\vec{j}$ yield the same value, independently on whether they are computed from the "incoming distribution functions" f_i or the "outgoing distribution functions" $f_i+\Omega_i$. This implies that the space average of those variables does not change during the time evolution, unless a source term is introduced by the domain boundaries or by the external momentum source $\vec{F}$. Of course, a numerical implementation of the LB model exhibits errors in the conservation laws anyway, due to the limited precision of floating point representations in a computer. By "exact conservation" it is meant that none of the typical error terms that scale at an order $\mathcal{O}(\delta_x^3)$, $\mathcal{O}(\delta_t^3)$, $\mathcal{O}(\epsilon^3)$ or $\mathcal{O}(Ma^3)$ is present in the conservation laws.*

2.2 Advection-diffusion: Chapman-Enskog ansatz

The advection-diffusion equation describes the evolution of a scalar quantity ρ that is diffused and exposed to the advective influence of an external term $\vec{u}$:

$$\partial_t \rho + \overrightarrow{\nabla} \cdot (\rho \vec{u}) = d\, \nabla^2 \rho, \tag{2.40}$$

where d is the diffusivity constant. The scalar ρ represents for example the density of a component in a multi-component fluid, or the temperature distribution is a fluid. It is possible to solve this equation by means of a lattice Boltzmann method. This is actually a relatively simple task, because the governing equation is linear, and only one quantity, ρ, is conserved. For this reason, this problem is used as an introductory example to explain the ideas behind the Chapman-Enskog expansion.

The collision operator for the advection-diffusion LB model can be written as a BGK term, a relaxation towards a local equilibrium distribution:

$$\Omega_i = -\frac{1}{\lambda}\left(f_i - f_i^{(eq)}\right) + t_i\, \vec{c}_i \cdot \delta\vec{j}. \tag{2.41}$$

Here, λ is the relaxation time, and $\delta\vec{j} = \epsilon\, \delta\vec{j}^{(1)}$ an external source term for the non-conserved first order moment. It is introduced formally and will be used later to correct errors in the numerical model. The equilibrium distribution reads

$$f_i^{eq} = \rho\, t_i (1 + \frac{1}{c_s^2}\vec{c}_i \cdot \vec{u}). \tag{2.42}$$

It is clear that this collision operator verifies the conservation requirements of Eqs. (2.23) and (2.24).

The governing equation for this model is given by the conservation laws for the zeroth-order moment, Eqs. (2.25) and (2.27), that are combined as follows:

$$\partial_t \rho + \epsilon \overrightarrow{\nabla}_1 \cdot \vec{j}^{(0)} = -\epsilon^2 \overrightarrow{\nabla}_1 \cdot \left(\vec{j}^{(1)} + \frac{1}{2}\partial_{t1}\vec{j}^{(0)} + \frac{1}{2}\overrightarrow{\nabla}_1 \cdot \mathbf{\Pi}^{(0)} + \frac{1}{2}c_s^2 \overrightarrow{\nabla}_1 \rho\right). \tag{2.43}$$

In order to find the unknown term in this equation, a scale separation for the particle distribution function f_i must be explicitly defined. In the Chapman-Enskog approximation, the model is expanded around the local equilibrium term. That is, the equilibrium term is identified with $f^{(0)}$:

$$f_i^{(0)} = f_i^{eq}. \tag{2.44}$$

It is immediately concluded, using the symmetry properties (a)-(d) in Eq. (2.1), that

$$\vec{j}^{(0)} = \rho\vec{u} \quad \text{and} \tag{2.45}$$

$$\mathbf{\Pi}^{(0)} = 0. \tag{2.46}$$

First-order terms with respect to ϵ are computed by rewriting Eq. (2.41) as follows (the equation is divided by ϵ, by which this parameter is skipped):

$$f_i^{(1)} = -\lambda\,\Omega_i + t_i\,\vec{c}_i \cdot \delta\vec{j}. \tag{2.47}$$

The collision term Ω_i is expanded in a Taylor series as in Eq. (2.19). Only first-order derivatives need to be retained at the $\mathcal{O}(\epsilon)$ scale:

$$\begin{aligned} f_i^{(1)} &= -t_i\lambda\,(\partial_{t1} + \overrightarrow{\nabla}_1 \cdot \vec{c}_i)\left[\rho\left(1 + \frac{1}{c_s^2}\vec{c}_i \cdot \vec{u}\right)\right] + \lambda\, t_i \vec{c}_i \cdot \delta\vec{j} \\ &= -\frac{\lambda}{c_s^2} t_i\, \boldsymbol{Q}_i : \overrightarrow{\nabla}_1(\rho\vec{u}) - \lambda\, t_i\, \vec{c}_i \cdot \overrightarrow{\nabla}_1\rho - \frac{\lambda}{c_s^2} t_i\, \partial_{t1}\vec{c}_i \cdot (\rho\vec{u}) + \lambda\, t_i\, \vec{c}_i \cdot \delta\vec{j}. \end{aligned} \tag{2.48}$$

Projection on the first-order base vectors $\mathcal{C}^1$ yields

$$\begin{aligned} \vec{j}^{(1)} = \sum_i \vec{c}_i f_i^{(1)} &= -\lambda\left(\frac{1}{c_s^2}\partial_{t1}\left(\sum_i t_i \vec{c}_i \vec{c}_i\right)\rho\vec{u} + \overrightarrow{\nabla}_1 \cdot \left(\rho \sum_i t_i \vec{c}_i \vec{c}_i\right)\right) + \lambda\left(\sum_i \vec{c}_i \vec{c}_i\right)\delta\vec{j} \\ &= -\lambda\left(\partial_{t1}(\rho\vec{u}) + c_s^2 \overrightarrow{\nabla}_1\rho\right) + \lambda\, c_s^2\, \delta\vec{j}. \end{aligned} \tag{2.49}$$

Eqs. (2.49), (2.45) and (2.46) are inserted into Eq. (2.43), which finally yields the $\mathcal{O}(\epsilon^2)$ scale of the dynamics:

$$\partial_t\rho + \overrightarrow{\nabla} \cdot (\rho\vec{u}) = d\,\overrightarrow{\nabla}^2\rho + \underbrace{d/c_s^2\,\partial_t \overrightarrow{\nabla} \cdot (\rho\vec{u})}_{\text{unwanted term}} - \underbrace{\lambda\, c_s^2 \overrightarrow{\nabla} \cdot \delta\vec{j}}_{\text{correction term}}, \tag{2.50}$$

where the diffusivity constant d has been defined as

$$d = c_s^2\left(\lambda - \frac{1}{2}\right). \tag{2.51}$$

Equation (2.50) is almost equivalent to the desired PDE, Eq. (2.40), except for a correction term on the right hand side. To the knowledge of the author, this term has been overlooked in the literature so far. Reference [17] for example presents a full Chapman-Enskog development of the temperature model (2.41) but somehow fails to find the error term. In Section 2.4, numerical evidence for the negative

influence of error term on the accuracy of a simulation is presented. It is shown in Section 3.1 that the free variable $\delta\vec{j}$ can be adjusted to eliminate the unwanted term.

This section concludes with a discussion of lattice structures that are appropriate for the implementation of the LB advection-diffusion model. It is pointed out that among the symmetry relations in Eq. (2.1), only the properties (a)-(d) were used in the previous theoretic development, and the property (e) in particular is not required. Additionally to the lattice structures listed in Section 1.3.2, one may use settings with weaker symmetry properties. A case in point are topologies that include only velocity vectors parallel to the space axes. In 2D, this includes the D2Q4 model with two horizontal vectors $\vec{c}_h = (\pm 1, 0)$ and vertical vectors $\vec{c}_v = (0, \pm 1)$. The lattice weights are all equal and take the value $t_i = 1/4$, and the lattice constant c_s^2 is defined as $c_s^2 = 1/2$. Another possible choice is the D2Q5 model that includes the zero-speed velocity $\vec{c}_0$. In this model, the constant c_s^2 is a free parameter. The lattice weights take the value $t_0 = 1 - 2\,c_s^2$ for the zero-speed velocity, and $t_i = c_s^2/2$ for the four other ones. It is common to choose $c_s^2 = 1/3$, with corresponding lattice weights $t_0 = 1/3$ and $t_i = 1/6$ for $i = 1 \cdots 4$. The same arguments lead to the 3D lattice structures D3Q6 ($t_i = 1/6$, $c_s^2 = 1/3$) and D3Q7 ($t_0 = 1 - 3\,c_s^2$, $t_i = c_s^2/2$ for $i > 0$).

2.3 Fluid flows: Chapman-Enskog ansatz

The arguments leading to a LB model for fluid flows are similar to those developed in the previous section for the advection-diffusion model. The related algebra is however more involved, because both the zeroth order moment (mass) and the second-order moments (momentum) are conserved. The collision operator is again represented by a BGK relaxation term with corrections:

$$\Omega_i = -\omega(f_i - f_i^{(eq)}) + FT_i + CT_i. \tag{2.52}$$

The force term FT_i and the correction term CT_i are developed on the first and second order vectors $\mathcal{C}^1$ and $\mathcal{C}^2$:

$$FT_i = t_i \frac{a}{c_s^2}\,\vec{c}_i \cdot \vec{F} + t_i \frac{b}{2\,c_s^4}\,\boldsymbol{Q}_i : (\vec{F}\vec{u} + \vec{u}\vec{F}) \quad \text{and} \tag{2.53}$$

$$CT_i = t_i \frac{c}{2\,c_s^4}\,\boldsymbol{Q}_i : \delta\boldsymbol{\Pi}. \tag{2.54}$$

The term multiplied by a factor a adds to the total momentum of the system and represents the external force contribution in the momentum conservation law. The term preceded by a factor b acts on the higher order moment and introduces a correction to numerical errors of the force term, due to the second order time derivative. This approach has first been described in Ref. [18]. Finally, a formal term $\delta\boldsymbol{\Pi}$ is introduced. It is used in Section 3.2 to correct a numerical deficiency of the model. This term is introduced, just like the correction term $\delta\vec{j}$ in Eq. (2.41), at the level of the first non-conserved moment. In this way, it does not interfere with the exact conservation laws of the numerical model, which are explained in Note 2.4 on page 21.

The equilibrium distribution function f_i^{eq} is constructed in such a way that the collision term Ω respects all the conservation laws listed in Section 2.1.2, and the moments of the particle distribution functions are such that the momentum conservation equation (2.39) is equivalent to the Navier-Stokes equation. For this, the moments of the distribution functions are expanded on the base vectors $\mathcal{C}^0$, $\mathcal{C}^1$ and $\mathcal{C}^2$, which leads to the expression defined in Eq. 1.18. Actually, the dynamics described by Eqs. (2.52), (2.54) and (1.18) can only be used to solve the Navier-Stokes equation when the relative velocity $\vec{u}/c_s$ is small. More precisely, the quantity $\vec{u}/c_s$ is said to be of the order of magnitude of the *Mach number* (Ma), and the model contains a numerical error that scales at the third order of the Mach number, $\mathcal{O}(Ma^3)$. In the following development, all terms that scale at the third order of the Mach number are therefore neglected. An additional assumption is used, stating that the time-variations of the density are of the same order of magnitude as the Mach number, or smaller:

$$\partial_t \rho = \mathcal{O}(Ma). \tag{2.55}$$

This assumption is compatible with results from the kinetic theory of gases close to the isothermal limit [1]. Close to the limit of fluid incompressibility, the density variations are actually even found to scale like the square Mach number ($\partial_t \rho = \mathcal{O}(Ma^2)$ and $\partial_\alpha \rho = \mathcal{O}(Ma^2)$). The assumption of Eq. (2.55) is less restrictive and leaves space for the model to be pushed further away from its limit of incompressibility .

Note 2.5: Discrete Maxwellian *The lattice Boltzmann equation is sometimes derived by discretizing the continuous Boltzmann equation. In this derivation, the equilibrium distribution corresponds to a Maxwellian distribution of the velocity, and it is obtained by expanding this distribution for small Mach numbers, up to a second order. This point of view is for example developed in Ref. [14].*

It is obvious that the collision term in Eq. (2.52) conserves the zeroth order moment (mass). Conservation of the first order moment (momentum) reads

$$\sum_i \vec{c}_i \Omega_i = \left(a + \frac{\omega}{2}\right) \vec{F} \quad \text{by Eqs. (2.52) and (2.30).} \tag{2.56}$$

For Eq. (2.29) to be respected, the value of the constant a must be set to

$$a = 1 - \frac{\omega}{2}. \tag{2.57}$$

The governing equations for the LB fluid model are Eqs. (2.35) and (2.39). To obtain the missing terms in these equations, the dynamics is developed around the equilibrium distribution function, as it was done in the previous section for the advection-diffusion model. This time, all symmetry properties listed in Eq. (2.1) are used. The zeroth order terms read

$$\vec{j}^{(0)} = \rho \vec{u}, \tag{2.58}$$

$$\boldsymbol{\Pi}^{(0)} = \rho\, \vec{u}\vec{u} \quad \text{and} \tag{2.59}$$

$$\overrightarrow{\nabla}_1 \cdot \boldsymbol{R} = c_s^2 \left(\overrightarrow{\nabla}_1 (\rho \vec{u}) + (\overrightarrow{\nabla}_1 (\rho \vec{u}))^T + \overrightarrow{\nabla}_1 \cdot (\rho \vec{u})\, \boldsymbol{I} \right). \tag{2.60}$$

At the first order with respect to ϵ, it follows from Eq. (2.52) that

$$\begin{aligned} f_i^{(1)} &= -\frac{t_i}{\omega}(\partial_{t1} + \overrightarrow{\nabla}_1 \cdot \vec{c}_i)\left[\rho\left(1 + \frac{1}{c_s^2}\vec{c}_i\cdot\vec{u} + \frac{1}{2c_s^4}\boldsymbol{Q}_i : \vec{u}\vec{u}\right)\right] + \frac{1}{\omega}(FT_i + CT_i) \\ &= -\frac{t_i}{\omega}\Big(\partial_{t1}\rho + \frac{1}{c_s^2}\partial_{t1}(\vec{c}_i\cdot\rho\vec{u}) + \frac{1}{2c_s^4}\partial_{t1}(\boldsymbol{Q}_i : \rho\vec{u}\vec{u}) \\ &\quad + \vec{c}_i\cdot\overrightarrow{\nabla}_1\rho + \frac{1}{c_s^2}\vec{c}_i\vec{c}_i : \overrightarrow{\nabla}_1(\rho\vec{u}) + (\vec{c}_i\cdot\overrightarrow{\nabla}_1)(\boldsymbol{Q}_i : \rho\vec{u}\vec{u})\Big) + \frac{1}{\omega}(FT_i + CT_i). \end{aligned} \tag{2.61}$$

The time-derivatives in this equation are replaced by space derivatives through the following substitutions:

$$\partial_{t1}\rho = -\overrightarrow{\nabla}_1\rho\vec{u} \qquad \text{by (2.25,2.58)}, \tag{2.62}$$

$$\frac{1}{c_s^2}\partial_{t1}(\vec{c}_i\cdot\rho\vec{u}) = \frac{1}{c_s^2}\vec{c}_i\cdot\vec{F}^{(1)} - \frac{1}{c_s^2}\vec{c}_i\cdot\overrightarrow{\nabla}_1 : \rho\vec{u}\vec{u} - \vec{c}_i\cdot\overrightarrow{\nabla}_1\rho \qquad \text{by (2.31,2.59)}, \tag{2.63}$$

$$\begin{aligned} \frac{1}{2\,c_s^4}\boldsymbol{Q}_i : \partial_{t1}(\rho\,\vec{u}\vec{u}) &= \frac{1}{2\,c_s^4}\boldsymbol{Q}_i : \Big(\partial_{t1}(\rho\vec{u})\vec{u} + \vec{u}\,\partial_{t1}(\rho\vec{u}) - \underbrace{(\partial_{t1}\rho)\vec{u}\vec{u}}_{\mathcal{O}(Ma^3)}\Big) \\ &= \frac{1}{c_s^4}\boldsymbol{Q}_i : \partial_{t1}(\rho\vec{u})\vec{u} \qquad \text{(because } \boldsymbol{Q} \text{ is symmetric)} \\ &= \frac{1}{c_s^4}\boldsymbol{Q}_i : \Big(\vec{F}^{(1)} - \underbrace{\overrightarrow{\nabla}_1\cdot(\rho\vec{u}\vec{u})}_{\mathcal{O}(Ma^3)} - c_s^2\overrightarrow{\nabla}_1\rho\Big)\vec{u} \qquad \text{by (2.31)} \\ &= \frac{1}{c_s^4}Q_i : \left(\vec{F}^{(1)} - c_s^2\overrightarrow{\nabla}_1\rho\right)\vec{u}. \end{aligned} \tag{2.64}$$

Inserting all these terms into Eq. (2.61) yields

$$\begin{aligned} f_i^{(1)} = &-\frac{t_i}{c_s^2\,\omega}\left(\boldsymbol{Q}_i : \rho\overrightarrow{\nabla}_1\vec{u} - \vec{c}_i\overrightarrow{\nabla}_1 : \rho\,\vec{u}\vec{u} + \frac{1}{2c_s^2}(\vec{c}_i\cdot\overrightarrow{\nabla}_1)(\boldsymbol{Q}_i : \rho\,\vec{u}\vec{u})\right) \\ &- \frac{1}{2}\frac{t_i}{c_s^2}\vec{c}_i\cdot\vec{F} + \frac{(b-1)t_i}{2\,c_s^4\,\omega}\boldsymbol{Q}_i : (\vec{F}\vec{u} + \vec{u}\vec{F}) + \frac{c\,t_i}{2\,c_s^4\,\omega}\boldsymbol{Q}_i : \delta\boldsymbol{\Pi}. \end{aligned} \tag{2.65}$$

When this equation is projected on the vectors of $\mathcal{C}^2$, using Eq. (2.12), only two terms are left:

$$\boldsymbol{\Pi}^{(1)} = \sum_i \boldsymbol{Q}_i f_i^{(1)} = -\frac{2\,c_s^2}{\omega}\rho\,\boldsymbol{S} + \frac{b-1}{\omega}(\vec{F}\vec{u} + \vec{u}\vec{F}) + \frac{c}{\omega}\delta\boldsymbol{\Pi}, \tag{2.66}$$

where the symmetric tensor $\boldsymbol{S}$ is defined by Eq. (1.6) on page 7. In order to find the last missing term, the time derivative of the $\boldsymbol{\Pi}^{(0)}$ tensor, the same approximations are used as those that lead to Eq. (2.64):

$$\partial_{t1}\boldsymbol{\Pi}^{(0)} = \partial_{t1}(\rho\vec{u}\vec{u}) - \vec{F}^{(1)}\vec{u} + \vec{u}\vec{F}^{(1)} - c_s^2\left((\overrightarrow{\nabla}_1\rho)\vec{u} + \vec{u}(\overrightarrow{\nabla}_1\rho)\right). \tag{2.67}$$

Equations ((2.58)-(2.60)), (2.62), (2.66), and (2.67) are now collected and inserted into Eqs. (2.35) and (2.39). The resulting momentum equation contains among others an unwanted term that depends on the external force $\vec{F}$, introduced by Eqs. (2.66) and (2.67):

$$\mathbf{\Pi}^{(1)} + \frac{1}{2}\partial_{t1}\mathbf{\Pi}^{(0)} = \left(\frac{b-1}{\omega} + \frac{1}{2}\right)(\vec{F}\vec{u} + \vec{u}\vec{F}) - \frac{2\,c_s^2}{\omega}\rho\,S + \frac{1}{2}c_s^2\,\overrightarrow{\nabla}\cdot(\rho\vec{u})\boldsymbol{I}. \tag{2.68}$$

This force term is eliminated by setting the constant b to

$$b = 1 - \frac{\omega}{2}. \tag{2.69}$$

For the present discussion, the correction term $\delta\mathbf{\Pi}$ is left out, and thus the constant c is set to 0. This choice of the constants leads to the following equations:

$$\partial_t\rho + \overrightarrow{\nabla}\cdot(\rho\vec{u}) = 0 \quad \text{and} \tag{2.70}$$

$$\partial_t(\rho\vec{u}) + \overrightarrow{\nabla}\cdot\left(c_s^2\rho\boldsymbol{I} + \rho\vec{u}\vec{u} - \tau\right) = \vec{F}, \tag{2.71}$$

where the deviatoric stress tensor $\boldsymbol{\tau}$ reads

$$\boldsymbol{\tau} = 2\,\nu\rho\,\boldsymbol{S}, \tag{2.72}$$

and the dynamic shear viscosity ν is related to the relaxation parameter ω as follows:

$$\nu = c_s^2\left(\frac{1}{\omega} - \frac{1}{2}\right). \tag{2.73}$$

This set of PDE's is identical to the equations of fluid motion, introduced in Section 1.2.1. The bulk viscosity ν' can however not be chosen as a free parameter in the LB method, and it takes the fixed value $\nu' = 0$. In order to vary this parameter, one needs to exploit the degree of freedom introduced by the correction term $\delta\mathbf{\Pi}$. Section 3.2 explains how this is done.

For convenience, the force term FT_i in Eq. (2.53) is finally rewritten using the choice of the constants a and b derived in the present section:

$$FT_i = \left(1 - \frac{\omega}{2}\right)t_i\left(\frac{(\vec{c}_i - \vec{u})}{c_s^2} + \frac{\vec{c}_i\cdot\vec{u}}{c_s^4}\,\vec{c}_i\right)\cdot\vec{F}. \tag{2.74}$$

This form of the force term has been suggested in Ref. [18] and shown to be superior to other commonly used terms. Section 3.2 shows that one more correction needs to be added to the force when fluids with adaptable bulk viscosity are simulated.

2.4 Dimensionless formulation and accuracy

In the multi-scale analysis, the equivalence between the LB method and its associated macroscopic PDE is explicited via a finite development of the difference $f_i(\vec{r}+\vec{c}_i, t+1) - f_i(\vec{r}, t)$ into a Taylor series. Because this series is truncated after the second order term, it is common to say that "lattice Boltzmann is a second order accurate method in space and time". The present section discusses in some detail what is meant by this statement, and why it should be appreciated with care.

2.4.1 Dimensionless formulation

During the implementation of a simulation, LB users are not always aware of the time and space step δ_t and δ_x they are actually using. This question is therefore clarified to begin with. The approach adopted here is to relate lattice units to dimensionless units (see Section 1.2.3) through those parameters. Another approach would be to relate the lattice units directly with the units of a given physical system, but this is less general and does not serve the purpose of the present discussion. It is supposed that a reference length in the simulation has been resolved by means of N lattice sites, and a reference laps of time is simulated by T iteration steps. The corresponding scales are, by definition, unitary in the dimensionless system of units: $T^* = 1$ and $L^* = 1$. This leads to the definition of the discrete time and space steps:

$$\delta_t = \frac{T^*}{T} = \frac{1}{T} \quad \text{and} \tag{2.75}$$

$$\delta_x = \frac{L^*}{N-1} = \frac{1}{N-1}. \tag{2.76}$$

The term $N-1$ in the definition of δ_x stems from a *vertex-centered* interpretation of the location of lattice sites, as opposed to a *cell-centered* interpretation. The lattice sites are in that case situated at the intersection between two cell edges, and the number of cells spanned by N lattice sites amounts to $N-1$. It is common to specify a reference speed U instead of a reference time T to set up a simulation. The reference speed relates to the other parameters as follows:

$$U = \frac{N-1}{T} = \frac{\delta_t}{\delta_x} \tag{2.77}$$

Furthermore, the fluid viscosity $\nu = c_s^2(\tau - 1/2)$, expressed in lattice units, is related to the Reynolds number. To simplify the discussion, only the incompressible Navier-Stokes equation (1.10) is considered. By casting this equation into its dimensionless form, given by Eq. (1.12), one obtains:

$$\nu = \frac{1}{Re}\frac{\delta_t}{\delta_x^2}. \tag{2.78}$$

Note 2.6: Dimensionless or not? *The expression $U = \delta_t/\delta_x$ in Eq. (2.77) is somewhat puzzling, as the right hand side of the equation does not seem to have the dimensions of a velocity. To be sure it is understood properly, let us go through this once more. The velocity U can be viewed in two different systems of units. One of them is the "dimensionless" system: it might be more appropriate presently to call it "physical" system, to emphasize that it is independent of the lattice paramters. The other one is the system of lattice units. The lattice parameters δ_t and δ_x are used to restore the units of the physical velocity $U_{physical}$ from the lattice velocity $U_{lattice}$: $U_{physical} = \delta_x/\delta_t\, U_{lattice}$. Equation (2.77) is deduced when the "physical" system of units is chosen such that $U_{physical} = 1$.*

2.4.2 Accuracy of LB methods

The signification of the expression "space accuracy of a numerical method" is best understood when a stationary (time independent) system is simulated, and the obtained numerical solution solution $\vec{u}(\vec{r})$ is compared with an exact solution $\vec{v}(\vec{r})$ of the problem. A common approach to defining the numerical accuracy is to say that a method is of order k in space when there exists a constant κ for which the following relation holds:

$$\sqrt{\frac{\sum_{\vec{r}}(|\vec{u}(\vec{r}) - \vec{v}(\vec{r})|)^2}{N^d}} < \kappa^k, \tag{2.79}$$

where the sum is taken over all sites of a grid of size N^d. The time accuracy can be defined similarly for flows that are time periodic with period T. The accuracy is then said to be of the order k in time when there exists a constant κ' by which the error between the numerical solution $\vec{u}(\vec{x})$ and the exact solution $\vec{v}(\vec{x})$ is constrained as follows:

$$\frac{1}{T}\sum_t \sqrt{\frac{\sum_{\vec{r}}|\vec{u}(\vec{r},t) - \vec{v}(\vec{r},t)|^2}{N^d}} < \kappa'^k, \tag{2.80}$$

provided the space discretization error is small enough to be negligible.

Additionally to the LB approach, there exist many numerical methods for the resolution of PDE's in which the space derivatives are replaced by a finite Taylor series up to the order k_x, and the time derivatives by a series up to the order k_t. It can be formally shown that those methods have a space accuracy of the order k_x and a time accuracy of the order k_t, in the sense of Eqs. (2.79) and (2.80). Although no such formal proof exists for the LB methods, it is often argued on intuitive arguments that a similar relation must hold for this method too. This is further underlined by numerical evidence, such as the one presented in Section 3.2.

2.4.3 Accuracy of specific models

Some care must however be taken when the LB method is used to simulate incompressible flows. In those cases, the Mach number is chosen to be very small, and it is concluded that the time derivative of the density scales then like the square Mach number ($\partial_t\rho = \mathcal{O}(Ma^2)$), by which the density is asymptotically described as a time independent constant. In the LB method, the speed of sound is a lattice constant, and the Mach number of the system is therefore equivalent to the reference speed U expressed in lattice units: $Ma = U/c_s$. Based on Eq. (2.77), it is concluded that the error due to the compressibility effects scales like $E_{Ma} = \mathcal{O}(\delta_t^2/\delta_x^2)$. There are therefore in total three terms that contribute to the overall error E:

$$E = E_{\delta_x} + E_{\delta_t} + E_{Ma} = \mathcal{O}(\delta_x^2) + \mathcal{O}(\delta_t^2) + \mathcal{O}(\delta_t^2/\delta_x^2). \tag{2.81}$$

The additional error term E_{Ma} introduces a coupling between the space and the time discretization error. When the space step δ_x is decreased, the time step must be readapted for the overall error to decrease at a rate of $\mathcal{O}(\delta_x^2)$. That is, the E_{Ma} error term must scale as $E_{Ma} = \mathcal{O}(\delta_x^2)$, and the time and space step are therefore related through $\delta_t \sim \delta_x^2$. This in turn means that the time discretization error really

behaves as in numerical methods with first order accurate time stepping. Several numerical case studies are conducted in Sections 5.3 and 6.2 that show that the error term E_{Ma} has severe drawbacks on the efficiency of the LB model.

Note 2.7: Keeping ω fixed

When benchmarks are executed on several simulations with different grid size, a decision needs to be taken on how to adapt the discrete time step to a given value of the space step. A choice often observed in the literature consists in changing the scales in such a way that the Reynolds number Re on one hand and the relaxation parameter ω on the other hand remain identical from one grid to another. This choice is a priori surprising, because it is based on a microscopic property ω whose significance is not simple to interpret from a macroscopic point of view, as would be for example the time step δ_t, or the Mach number. With the help of the results presented in this Section, it is however possible to better understand this choice. Indeed, Eq. (2.78) shows that the parameter ω is proportional to δ_t/δ_x^2. Keeping ω constant thus amounts to varying δ_t at a rate proportional to δ_x^2. It has been argued in the previous paragraph that this is an appropriate way to scale the lattice parameters. It guarantees in particular that the discretization error of a simulation decreases at a second order rate when the grid resolution is increased.

The full power of LB methods is thus best exploited when compressible flows are simulated at low Mach numbers. But even then, the LB method does not possess all features one might expect from a good numerical method. First, the bulk viscosity can not be adjusted in the basic BGK fluid model. This drawback can luckily be fixed by taking into account a correction term discussed in Section 3.2. Another problem however still subsists: the choice of the Mach number is intimately coupled with the choice of the discrete time and space step, from which it is concluded that those three parameters cannot be chosen independently. Indeed, in the LB model that was introduced in this text, and which is systematically used in the community, the speed of sound is a lattice constant. This means that it takes a constant value (independent of δ_x and δ_t) when it is expressed in lattice units. The Mach number is thus identified with the reference speed expressed in lattice units. From Eq. (2.77), the following relationship between the Mach number and the discretization parameters is deduced:

$$Ma \sim \frac{\delta_t}{\delta_x}. \tag{2.82}$$

For consistency, it should be remarked that it is principally possible to vary the speed of sound in LB models. This is done by adapting the rest-particle weight t_0. Choosing a speed of sound different than 1/3 has however been observed to induce numerical instabilities and is therefore very uncommon.

It is interesting to note that the LB equation for advection-diffusion problems introduced by Eqs. (2.41) and (2.42) contains an error term that is very similar to the error term E_{Ma} in the context of fluid dynamics. To see this, it suffices to turn the PDE (2.50) to a dimensionless system, just as was done in Section 1.2.3 for the

Navier-Stokes equations (the correction term $\delta\vec{j}$ is excluded from the discussion):

$$\partial_t^* \rho + \overrightarrow{\nabla}^* \cdot (\rho \vec{u}^*) = d^* \, \overrightarrow{\nabla}^{*2} \rho + \frac{\delta_t^2}{\delta_x^2} \frac{1}{c_s^2} d^* \, \partial_t^* \overrightarrow{\nabla}^* \cdot (\rho \vec{u}^*), \tag{2.83}$$

where $d^* = \delta_x^2 / \delta_t \, d$ is the dimensionless diffusion parameter. Given that c_s^2 is an immutable lattice constant, this equation contains an error term E_{LE} due to lattice effects that scales like δ_t^2 / δ_x^2. As in Eq (2.81), the overall error contains therefore three contributing terms:

$$E = E_{\delta_x} + E_{\delta_t} + E_{LE} = \mathcal{O}(\delta_x^2) + \mathcal{O}(\delta_t^2) + \mathcal{O}(\delta_t^2 / \delta_x^2). \tag{2.84}$$

The effect of those error terms on actual numerical simulations is discussed in Section 3.1, and a strategy is devised to eliminate the error term E_{LE}.

Note 2.8: Summary *All points discussed in this section can be summarized by answering the following*

Question: *What is the time and space accuracy of a LB method?*

Answer: *The LB method is generally thought of as a second order accurate numerical scheme, both in space and time, for the simulation of compressible, isothermal flows at small Mach numbers. This conjecture has however never been proved formally, and it makes sense only when the widely unknown correction term in Section 3.2 is used. Furthermore, the discrete time and space steps cannot be chosen freely in compressible flows, because they are coupled with a physically significant value, the Mach number. This problem does not appear when incompressible flows are simulated, but in that context, the LB model behaves like a numerical scheme with first order accurate time stepping.*

Chapter 3

Corrections to the BGK dynamics

In Chapter 2, two lattice Boltzmann models are analyzed, one that describes advection-diffusion phenomena, and one that simulates fluid motion. In both models, a correction term is formally introduced, which offers an additional degree of freedom for handling deficiencies of the numerical models. Those degrees of freedom are exploited in this chapter. In the advection-diffusion model, the correction term $\delta\vec{j}$ is adjusted so as to eliminate the lattice error E_{LE} (*cf.* Section 2.4). In the fluid model, an appropriate choice of the correction term $\delta\mathbf{\Pi}$ leads to a numerical scheme in which the bulk viscosity ν' is a free parameter (it was shown in Section 2.3 that ν' is immutable in the basic BGK model). A strong analogy in the way those two problems are treated becomes apparent. In particular, it is shown that the deficiency of both numerical models can be eliminated by explicitly referring to the value of the "rest-particle" distribution function f_0.

3.1 Advection-diffusion revisited

3.1.1 Correction term for second order accurate time stepping

In Section 2.2 a BGK model for the modeling of advection-diffusion processes is proposed. Equation (2.50) on page 22 shows the macroscopic PDE related to this model, obtained by a multiscale Chapman-Enskog analysis. This equation contains an unwanted error term, as well as a correction term, depending on a not yet specified correction vector $\delta\vec{j}$. In the present section, this vector is chosen in such a way as to cancel out the unwanted term:

$$\left(\lambda - \frac{1}{2}\right) \partial_{t1} \overrightarrow{\nabla}_1 \cdot (\rho\vec{u}) - \lambda\, c_s^2\, \overrightarrow{\nabla}_1 \cdot \delta\vec{j} = 0, \tag{3.1}$$

which leads to the condition

$$\delta\vec{j} = \frac{1}{c_s^2}\left(1 - \frac{1}{2\lambda}\right) \partial_{t1}(\rho\vec{u}) = \frac{1}{c_s^2}\left(1 - \frac{1}{2\lambda}\right) (\underbrace{\rho\, \partial_{t1}\vec{u}}_{\vec{j}_1} - \underbrace{\vec{u}\, \partial_{t1}\rho}_{\vec{j}_2}). \tag{3.2}$$

The correction term $\delta\vec{j}$ contains two contributions. The first, which will be named $\vec{j}_1$, as indicated in Eq. 3.2, depends on the time derivative of the velocity $\vec{u}$, and the

second, $\vec{j}_2$, on the time derivative of the density. It is emphasized that a first order accurate estimate of those derivatives is sufficient for the present purpose, because the term ∂_{t1} acts at a $\mathcal{O}(\epsilon)$ scale. In Eq. (2.19), this scale is related to the first order terms of the finite Taylor series.

The value of $\vec{j}_1$ can for example be found by a finite difference approximation to the time derivative. The difference of the velocity between two successive time steps is expanded in a Taylor series and scale separated as follows:

$$\vec{u}(t+1) - \vec{u}(t) = \epsilon \partial_{t1} \vec{u} + \epsilon^2 (\partial_{t2} + \frac{1}{2} \partial_{t1}^2) \vec{u} + \mathcal{O}(\epsilon^3), \tag{3.3}$$

which leads to an estimate of the $\mathcal{O}(\epsilon)$ scale of the time derivative:

$$\epsilon \, \partial_{t1} \vec{u}(\vec{r}, t) = \vec{u}(\vec{r}, t+1) - \vec{u}(\vec{r}, t) + \mathcal{O}(\epsilon^2). \tag{3.4}$$

Given that the correction term δ_j scales at an order $\mathcal{O}(\epsilon)$, the first-order finite difference suggested by Eq. (3.4) is sufficient to approximate the time derivative $\partial_{t1} \vec{u}$.

Whether this expression can easily be evaluated or not depends on the origin of the external convective term $\vec{u}$. A typical application area for the advection-diffusion equation is the analysis of thermal flows using the Boussinesq approximation. In that case, the external vector field $\vec{u}$ is obtained from a numerical solver for the Navier-Stokes equation. It is common for such solvers to keep in memory two successive time steps of the velocity field, and the finite difference in Eq. (3.4) can be evaluated efficiently.

The value of $\vec{j}_2$ is directly found from the off-equilibrium part $f_0^{(1)}$ of the rest-particle distribution function. As a matter of fact, Eq. (2.48) leads to the following relation:

$$\partial_{t1} \rho = -\frac{1}{\lambda t_0} f_0^{(1)}. \tag{3.5}$$

Equations (3.4) and (3.5) are finally combined with Eq. (3.2) as follows:

$$\delta \vec{j} = \frac{1}{c_s^2} \left(1 - \frac{1}{2\lambda}\right) \left(\rho \left(\vec{u}(t+1) - \vec{u}(t) \right) - \frac{1}{\lambda \, t_0} f_0^{(1)} \, \vec{u} \right). \tag{3.6}$$

This correction term can of course only be implemented on lattice structures that possess a rest particle distribution function, which rules out the D2Q4 lattice, but works just fine with the D2Q5 lattice.

Note 3.1: Efficiency *When the correction term in Eq. (3.6) is taken into account, the LB method for advection-diffusion problems is unconditionally second order accurate, both in space and time. Furthermore, it belongs to the family of so-called explicit schemes, because the data at a time step $t+1$ is obtained from the data at a time t by an explicit calculation, and no expensive iterative procedure needs to be undertaken. It is quite unusual for numerical methods with second order accurate time stepping to be explicit, and the LB method is exceptional in this sense. Compared to other methods, the LB method can be said to*

be very memory expensive, as it requires for example five variables per lattice site on a D2Q5 lattice, but very fast, as it steps through time at a second order accuracy and with an explicit scheme.

3.1.2 Numerical verification

This section presents the results of a $2D$ numerical experiment on which the LB model for advection-diffusion (2.41,2.42) is tested, both in its original BGK formulation with $\delta\vec{j} = 0$, and with the modification term suggested in Eq. (3.6). The theoretical discussion of the error terms in Eq. (2.84) is numerically verified, and the benefit of the correction term $\delta\vec{j}$ on the accuracy of the model is underlined.

In the numerical example, a fixed, divergence free and time independent velocity field $\vec{u}$ is imposed, and the advection-diffusion equation for the scalar ρ is solved under the advective influence of $\vec{u}$. The velocity field is defined on a square domain whose side length serves as reference for the definition of a dimensionless length scale. It corresponds to a Poiseuille profile whose maximum velocity u_{max} defines the reference scale for the velocity:

$$\begin{aligned} \vec{u^*}(\vec{x^*}) &= (u^*(x^*, y^*), v^*(x^*, y^*)), \quad \text{where} \\ u^*(x^*, y^*) &= 4\, y^*\,(1 - y^*) \quad \text{and} \\ v^*(x^*, y^*) &= 0. \end{aligned} \tag{3.7}$$

The domain is periodically extended in both the x- and the y-direction. The diffused scalar ρ is initially defined through a uniform Gaussian distribution:

$$\rho_{ini}(\vec{r^*}) = \frac{1}{\sqrt{2\pi}\,\sigma}\, e^{-\frac{|\vec{r^*} - \vec{c}|^2}{2\,\sigma^2}}, \tag{3.8}$$

where the distribution, in dimensionless variables, is centered at $\vec{c} = (0.5, 0.4)$ and has a width of $\sigma = 0.1$. All simulations are run on a D2Q5 lattice.

The numerical results are compared to the results of a high accuracy reference simulation. To be sure the time and space steps of the reference simulation are sufficiently small, they are successively decreased until time and space convergence is reached, that is, until a further refinement does not influence any more the conclusions drawn from the simulations. The reference simulation uses the BGK model without correction term, so that no prejudice on the correctness of this approach is made. The simulations are executed on a time interval $0 \leq t < 0.2$. The error of the numerical result at a given time step, as compared to the reference simulation, is defined through a l_2 space norm, just as in Eq. (2.79). Similarly, the error of a simulation on a full time interval is defined on ground of a l_2 time norm, as in Eq. (2.80). This procedure cannot be used to evaluate quantitatively the accuracy of the time evolution, as it is defined in Section 2.4, because the flow in not periodic. It leads however to a qualitative estimation of the error, and an interesting comparison between the BGK approach and the corrected model.

Figure 3.1 displays the numerical results obtained on this problem by means of the original BGK and the corrected model. In Figure 3.1 (a), the space resolution

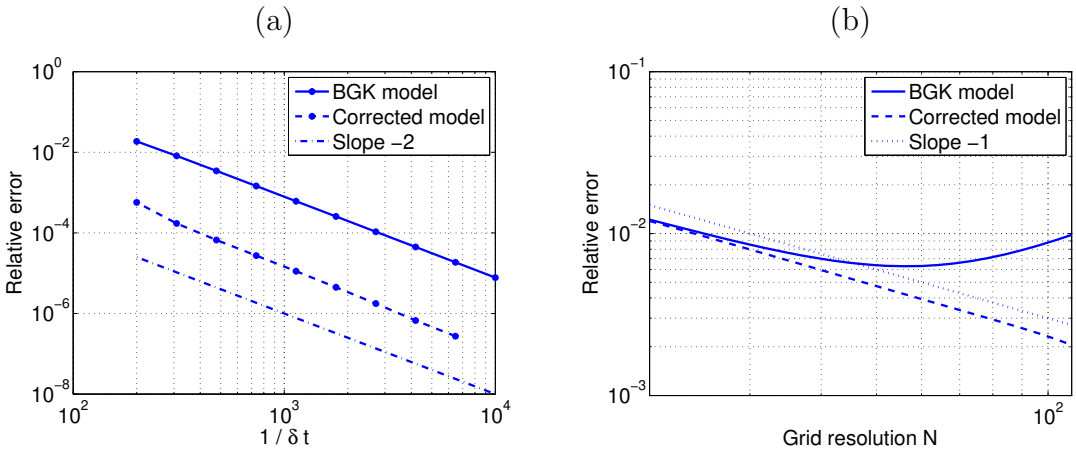


Figure 3.1: Accuracy of the BGK and the corrected advection diffusion models on a chosen 2D problem. (a) Fixed diffusivity $d^* = 0.01$ and grid resolution $N = 100$, varying time step δt. (b) Fixed diffusivity $d^* = 0.05$ and time step $\delta_t = 0.002$, varying grid resolution N.

is kept constant, and the time step is progressively refined. It is appropriate at this point to recapitulate the conclusions drawn in Section 2.4 on the accuracy of the numerical model. In Eq. (2.84) on page 30, three error terms were explicited, the space error $E_{\delta_x} = \mathcal{O}(\delta_x^2)$, the time error $E_{\delta_t} = \mathcal{O}(\delta_t^2)$ and an additional lattice effect $E_{\text{LE}} = \mathcal{O}(\delta_t^2/\delta_x^2)$ that is present only in the uncorrected BGK model. The space error E_{δ_x} is not visible on Figure 3.1 (a), because the lattice resolution has been chosen deliberately high enough. The remaining error terms E_{δ_t} and E_{LE} scale both at a same rate $\mathcal{O}(\delta_t^2)$. This is confirmed on the graph, on which the error visibly decreases at the same rate for both numerical methods. However, the offset between the two curves shows that the term E_{LE} has a more severe influence on the overall error than E_{δ_t}. On Figure 3.1(b), the time step δ_t is kept constant, at a value sufficiently small for the time error E_{δ_t} to be negligible, and the lattice resolution is progressively increased. This time, the BGK simulation contains a decreasing error $E_{\delta_x} = \mathcal{O}(\delta_x^2)$, and an increasing error $E_{\text{LE}} = \mathcal{O}(1/\delta_x^2)$. This is clearly reflected on the graph: the error curve corresponding to BGK reaches a minimum after which it increases again, whereas the error in the corrected model continues to decrease. As a side remark, the rate of decrease is proportional to δ_x and not δ_x^2, as one might have expected. It must be again underlined that those measurements of the accuracy may not be interpreted quantitatively, because the flow is not periodic. It would be interesting to devise a new numerical experiment that is time periodic, and on which the rate of decrease of the numerical error can be verified quantitatively.

3.2 Bulk viscosity for fluid flows

3.2.1 Numerical error in incompressible flows

Equation (2.81) on page 28 shows that there are three contributions to the overall error of a simulation when an incompressible fluid is being simulated: a space error

$E_{\delta_x} = \mathcal{O}(\delta_x^2)$, a time error $E_{\delta_t} = \mathcal{O}(\delta_t^2)$ and a compressibility error $E_{Ma} = \mathcal{O}(\delta_t^2/\delta_x^2)$. In the present section, those theoretical considerations are first underlined by a numerical experiment. Then, a modified BGK model is introduced in which the error term cancels.

The experiment, known under the name of *Womersley flow*, is now shortly introduced and presented in a system of dimensionless variables. Just like the Poiseuille flow used in Section 3.1, the Womersley flow takes place in a channel whose width determines the length scale for the dimensionless variables. A Poiseuille flow is a flow parallel to the channel boundaries with parabolic velocity profile that is driven either by an constant body force or by a constant pressure gradient DP. A pressure driven Poiseuille velocity profile has the following form:

$$u_x^*(y^*) = \frac{Re}{2} DP(y^{*2} - y^*). \tag{3.9}$$

The maximum value of the velocity in the middle of the channel is often taken as a reference value to choose a time scale for the dimensionless variables. In that case, the pressure gradient DP takes the value

$$\frac{\partial p}{\partial x^*} = DP = -\frac{8}{Re}. \tag{3.10}$$

In a Womersley flow, the pressure gradient is varied periodically with a frequency ω^* as follows:

$$\frac{\partial p}{\partial x^*} = A\cos(\omega^* t^*). \tag{3.11}$$

The resulting velocity profile is time dependent and substantially more complex than the Poiseuille profile in Eq. (3.9). When the system is iterated long enough, it enters a time periodic state for which an analytical solution is known. To choose a time scale for this flow, the Poiseuille flow is taken as a reference, and the amplitude A in Eq. 3.11 is identified with the pressure gradient DP in Eq. 3.10: $A = -8/Re$. Furthermore, a parameter α is introduced that is known under the name of *Womersley number* and defined as

$$\alpha = \sqrt{\frac{Re\,\omega^*}{4}}. \tag{3.12}$$

With those definitions, the analytical Womersley profile reads

$$u_x^* = Real\left[\frac{DP}{i\omega} e^{i\omega t}\left(1 - \frac{\cosh\left(\sqrt{2}\left(y^* - \frac{1}{2}\right)(\alpha + i\alpha)\right)}{\cosh\left(\frac{\sqrt{2}}{2}(\alpha + i\alpha)\right)}\right)\right], \tag{3.13}$$

where i is the imaginary number, and the term *Real* indicates that the real component of the right hand side expression must be extracted.

All simulations are executed with an unmodified BGK model, defined by Eqs. (2.52, 2.54, 1.18), with $\delta\Pi = 0$ on a D2Q9 lattice. The parameters are $Re = 1$ and $\alpha = 5$. The analytical solution of the Womersley flow is actually one-dimensional, given that Eq. (3.13) depends only on the y-direction and not the x-direction. Therefore, the x-direction, parallel to the channel walls, is resolved by only two lattice sites (one lattice site would be insufficient for the implementation of the pressure gradient).

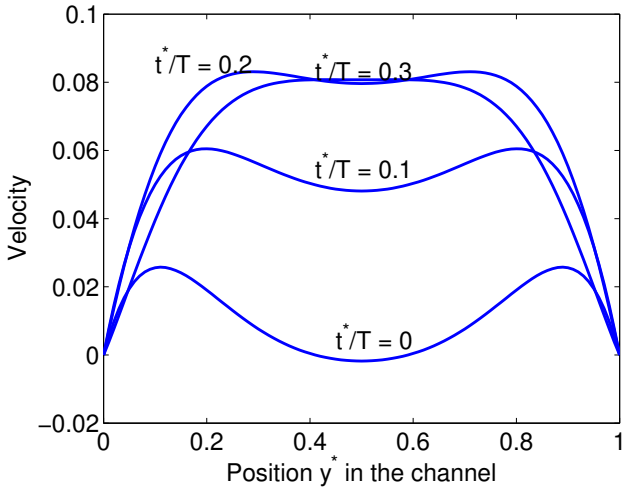


Figure 3.2: Womersley profile, predicted by Eq. (3.13), at four chosen time steps within a period $T = 2\pi/\omega^*$, for $Re = 1$ and $\alpha = 5$.

Figure 3.2 shows the analytical solution of the Womersley problem, obtained from Eq. (3.13) with the mentioned parameters. It is interesting to note that the profile is seriously flattened, compared to a Poiseuille profile whose maximum value would be 1.

The simulations are executed until a time periodic state is reached, and their relative error to the analytical solution in Eq. (3.13) is calculated by a time average, as in Eq. 2.80. It is argued in Section 2.4 that the time step δ_t in the BGK method must be varied like δ_x^2 so as to cancel compressibility effects. The results of applying this procedure, with $\delta_t = 2/5\,\delta_x^2$, is shown in Fig. 3.3(a). As expected, the error decreases exactly at a second order rate with the space step δ_x. This could however be achieved only via a dramatic adaptation of the time step. For that reason, as it is argued in Section 2.4, the BGK method behaves like a numerical scheme with first order accurate time discretization, when used for incompressible flows. Figure 3.3(b) shows what happens when the time step is fixed to a value $\delta_t = 2/5\,100^{-2} = 0.004$. When the grid resolution is refined, the error, dominated by the space error E_{δ_x}, *decreases* with a second order rate. At some point however, the compressibility error E_{Ma}, which scales like $1/\delta_x^2$, becomes dominant. From then on, the error *increases* at a second order rate.

3.2.2 Correction term for the bulk viscosity

The error E_{LE} in the advection diffusion model could be eliminated by an appropriate choice of the correction term CT_i in Section 3.1. It would be tempting to take a similar approach for the fluid model and to look for a correction $\delta\mathbf{\Pi}$ for which the model is fully incompressible: $E_{Ma} = 0$. Things are however not that easy, and to the knowledge of the author, there exists currently no lattice Boltzmann model that simulates incompressible flows with an adequate accuracy. Though a seemingly incompressible fluid model has been presented in Refs. [19, 20], this model has been

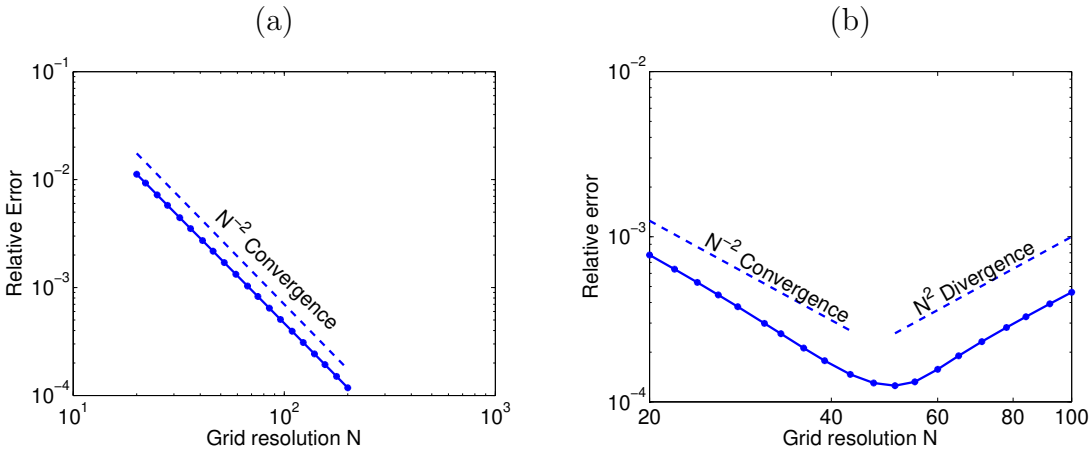


Figure 3.3: Accuracy of the BGK scheme compared to the analytical solution of an incompressible Womersley flow. (a) $\delta_t = 2/5\,\delta_x^2$ (b) $\delta_t = const. = 2/5 \cdot 100^{-2}$

shown to be erroneous (see *e.g.* Ref. [21]), as it eliminates compressibility effects only in stationary flows. For that reason, the LB method is most efficiently used to simulate weakly compressible flows. In this regime at least, it should therefore be able to provide the expected results. This is not the case with the plain BGK model discussed in Section 2.3, because it does not offer the possibility to choose the bulk viscosity in an arbitrary way. In the present section, this problem is circumvented by an appropriate choice of the correction term CT_i. This correction is proposed in the literature in Ref. [22], and it is extended in the present context by the inclusion of a body force term. Indeed, the following developments show that a coupling between the body force and the bulk viscosity correction occurs and requires special care. The section ends with the suggestion of an alternative correction term, which achieves the same effect as the one proposed in [22] but is more efficient to implement numerically.

In order to obtain a LB model with adjustable bulk viscosity, the correction $\delta\mathbf{\Pi}$ is implemented by means of the following ansatz:

$$\delta\mathbf{\Pi} = \frac{1}{d}(\,Tr(\mathbf{\Pi}^{(1)} + k\,\vec{F}\cdot\vec{u})\,\boldsymbol{I}, \tag{3.14}$$

where $Tr(\boldsymbol{T})$ yields the trace of a tensor $\boldsymbol{T}$ and k is a constant to be determined. The constant d corresponds to the number of space dimensions, and it enters the calculations through the identity $Tr(\boldsymbol{I}) = d$. Taking the trace of Eq. (3.14) yields

$$Tr(\delta\mathbf{\Pi}) = Tr(\mathbf{\Pi}^{(1)}) + k\,\vec{F}\cdot\vec{u}. \tag{3.15}$$

From Eqs. (2.66) and (2.69) on page 26 the trace of $\mathbf{\Pi}^{(1)}$ is evaluated as follows:

$$Tr(\mathbf{\Pi}^{(1)}) = \frac{c}{\omega}\,Tr(\delta\mathbf{\Pi}) - \frac{2\,c_s^2}{\omega}\rho\,\overrightarrow{\nabla}_1\cdot\vec{u} - \vec{F}\cdot\vec{u}., \tag{3.16}$$

Inserting Eq. (3.15) leads to

$$Tr(\mathbf{\Pi}^{(1)}) = \frac{1}{1-\frac{c}{\omega}}\left(\left(\frac{c}{\omega}k - 1\right)\vec{F}\cdot\vec{u} - \frac{2\,c_s^2}{\omega}\rho\overrightarrow{\nabla}_1\cdot\vec{u}\right). \tag{3.17}$$

Equation (2.66) shows that the correction to the tensor $\mathbf{\Pi}^{(1)}$ due to $\delta\mathbf{\Pi}^{(1)}$ is of the magnitude $c/\omega\,\delta\mathbf{\Pi}^{(1)}$. With the help of Eqs. (3.14) and (3.17) this expression evaluates to

$$\frac{c}{\omega}\delta\mathbf{\Pi} = \frac{1}{1-c/\omega}\frac{c}{\omega\, d}\left(\left(\left(1+\frac{c}{\omega}\right)k-1\right)\vec{F}\cdot\vec{u} - \frac{2\,c_s^2}{\omega}\rho\overrightarrow{\nabla}_1\cdot\vec{u}\right)\boldsymbol{I}. \tag{3.18}$$

The force term is eliminated by choosing the constant k to be

$$k = \frac{1}{1+\frac{c}{\omega}}. \tag{3.19}$$

Furthermore, by setting

$$c = \omega\frac{1}{1-\frac{2\,c_s^2}{d\,\omega\,\nu'}}, \tag{3.20}$$

Eq. (3.18) becomes

$$\frac{c}{\omega}\,\delta\mathbf{\Pi} = \nu'\,\rho\,\overrightarrow{\nabla}_1\cdot\vec{u}\,\boldsymbol{I}. \tag{3.21}$$

The momentum conservation law derived from the modified model is then the same as in Eq. (2.71), but it includes the full deviatoric stress τ, as in Eq. (1.5).

To explicit the final form of the correction term, Eq. (3.14) is inserted into Eq. (2.54). Using the identity $\boldsymbol{Q}_i : \boldsymbol{I} = |\vec{c}_i|^2 - c_s^2\,d$, one obtains

$$CT_i = \frac{c\,t_i}{2\,d\,c_s^4}\left(|\vec{c}_i|^2 - c_s^2\,d\right)\left(Tr(\mathbf{\Pi}^{(1)}) + k\,\vec{F}\cdot\vec{u}\right), \tag{3.22}$$

where the constants c and k are defined by Eqs. (3.20) and (3.19). The expression $Tr(\mathbf{\Pi}^{(1)})$ is evaluated using the definition of $\mathbf{\Pi}$ in Eq. (2.22), and the value of $\mathbf{\Pi}^{(0)}$ for the BGK model in Eq. (2.59). One finds

$$\mathbf{\Pi}^{(1)} = \sum_i \vec{c}_i\vec{c}_i f_i - \rho\,\vec{u}\vec{u} - c_s^2\,\boldsymbol{I}, \text{ and consequently,} \tag{3.23}$$

$$Tr(\mathbf{\Pi}^{(1)}) = \sum_i |\vec{c}_i|^2\; f_i - \rho\;|\vec{u}|^2 - c_s^2\,d. \tag{3.24}$$

It is interesting to note that the calculations required in Eq. (3.24) can be circumvented by referring explicitly to the rest particle distribution function f_0, as was done in Section 3.1 for the advection-diffusion model. Indeed, the following relation is directly deduced from Eq. (2.65):

$$f_0^{(1)} = f_0 - \rho\,t_0 = -\frac{t_0}{2\,c_s^2}\,Tr(\mathbf{\Pi}^{(1)}). \tag{3.25}$$

The correction term CT_i can therefore alternatively be written as follows:

$$CT_i = \frac{c\,t_i}{2\,d\,c_s^4}\left(|\vec{c}_i|^2 - c_s^2\,d\right)\left(-\frac{2\,c_s^2}{t_0}f_0^{(1)} + k\,\vec{F}\cdot\vec{u}\right). \tag{3.26}$$

It is reminded that the adjustable bulk viscosity ν' enters the equation via the constant c, defined in Eq. (3.20), and that the constant d corresponds to the number of space dimensions (2 or 3) of the system.

3.3 Alternative LB models

3.3.1 LB models with multiple relaxation times

The BGK model introduced in Section 1.3.2 can be extended to a more general class of models, in which the collision operator Ω, defined in Eq. (1.14) on page 10 takes the form of a diagonalizable matrix Ω_{ij} that acts as a linear operator on the elements of the off-equilibrium particle distribution functions:

$$f_i(\vec{x}+\vec{c}_i, t+1) - f_i(\vec{x}, t) = \sum_j \Omega_{ij}\,(f_j - f_j^{eq}). \tag{3.27}$$

The restriction to the off-equilibrium part of the distribution functions is motivated theoretically by the multi-scale analysis in Section (2.1), in which the scale of the collision operator is formally identified with the scale of the off-equilibrium distribution functions. Using the definition of the vector space $\mathbb{R}^q$ introduced in Section 2.1.1, with the scalar product of Eq. (2.5) and the definition of the unweighted particle distribution functions g in Eq. (2.13), Eq. (3.27) is interpreted as a Matrix-Vector product between the matrix $\boldsymbol{\Omega}$ and the vector g. Given that the matrix $\boldsymbol{\Omega}$ is diagonalizable, there exists an invertible matrix $\boldsymbol{M}$, whose lines represent the eigenvectors of $\boldsymbol{\Omega}$, and a diagonal matrix $\boldsymbol{D}$, for which the following relation holds:

$$\boldsymbol{\Omega} = \boldsymbol{M}^{-1}\,\boldsymbol{D}\,\boldsymbol{M}. \tag{3.28}$$

Equation (3.27) is therefore rewritten in matrix notation as follows:

$$g^{out} = g + \boldsymbol{\Omega}\,(g - g^{eq}) = g + \boldsymbol{M}^{-1}\,\boldsymbol{D}\,\boldsymbol{M}(g - g^{eq}), \tag{3.29}$$

where the short-hand notation $g_i^{out} \equiv g(\vec{x}+\vec{c}_i, t+1)$ for "outgoing" particle distribution functions has been used.

Note 3.2: Matrix notation *In this section, a matrix notation is exceptionally used on the space $\mathbb{R}^q$ of the particle distribution functions. The same notation is used in the original papers that introduce multiple relaxation time models, and it underlines the fact that linear algebra tools are used for developing new LB models. One must be careful though not to mistake space vectors for the vectors in $\mathbb{R}^q$, and thus, keep in mind that M and D are matrices of size $q \times q$, whereas the tensors $\boldsymbol{Q}$ and $\boldsymbol{\Pi}$ are represented by matrices of size $d \times d$.*

The space representation of the distribution functions is now replaced by what is called a *moment space representation* for reasons that become clear shortly, by the change of variables $\mathcal{G} = \boldsymbol{M}\,g$. The moment space representation of Eq. (3.29) reads

$$\mathcal{G}^{out} = \boldsymbol{D}\,(\mathcal{G} - \mathcal{G}^{eq}). \tag{3.30}$$

The last step consists in the formulation of a base formed by eigenvectors of the desired collision operator. The choice presented here is taken from Ref. [21] and encourages a physical interpretation of the algebraic operations. In this approach

the base is first populated by the $1+d+d(d+1)/2$ vectors defined by C^0, C^1 and C^2 in Eqs. (2.2), (2.3) and (2.4) respectively. It is now clear that the first components of the the vector $\mathcal{G}$ are constituted by the moments of the distribution functions.

With this new knowledge, Eq. (3.30) can be inspected more closely. It is obvious that in the BGK model, both $\boldsymbol{\Omega}$ and $\boldsymbol{D}$ are defined as $\boldsymbol{\Omega} = \boldsymbol{D} = -\omega \boldsymbol{I}$. Equation (3.30) shows that in this model, all moments are relaxed by a factor ω. An exception is made for the conserved moments ρ and $\vec{u}$, because they are not represented in the vector $\mathcal{G}$. Indeed, the projection of $g - g^{eq}$ on the vectors $\mathcal{C}^0$ and $\mathcal{C}^1$ vanishes by virtue of Eqs.(2.24, 2.30, 2.44).

The idea behind multiple relaxation time (MRT) models is that each moment can be relaxed at a different rate, by modifying the corresponding value on the diagonal of $\boldsymbol{D}$ (see Ref. [23]). This approach can be used either to modify the physics of a model by adapting the relaxation parameter of a hydrodynamically relevant moment, or to increase the stability of the model by adapting the relaxation time of physically irrelevant moments. The latter aim is for example achieved in Ref. [24] via a linear stability analysis of the model.

The moments ρ, $\vec{u}$ and $\boldsymbol{\Pi}$ related to the base vectors introduced so far are all relevant to the hydrodynamics of the model. To form a complete base of $\mathbb{R}^q$, they must be completed by additional, physically irrelevant base vectors known as *ghost vectors*. Their corresponding variables are named *ghost variables*. This point is illustrated with the simple D2Q9 lattice, in which the space of particle distribution functions, and thus also the space of the moments, is nine-dimensional. This moment space can be specified via 6 physically relevant base vectors (1 for the density, 2 for the velocity and 3 for the Π-tensor) and 3 ghost vectors. Guided by Ref. [22], a new set of lattice constants h_i is introduced as follows:

$$h_i = (1, -2, -2, -2, -2, 4, 4, 4, 4) \quad \text{for } i = 0 \cdots 8. \tag{3.31}$$

It is used to define a new lattice vector C^3 by

$$\mathcal{C}^3_i = h_i \tag{3.32}$$

and two lattice vectors C^4_α by

$$\mathcal{C}^4_{i,\alpha} = h_i \, c_{i\alpha}. \tag{3.33}$$

A distribution function f can be projected on this base by purely algebraic means, without inclusion of any physical ingredient. Some additional factors appear during this projection, because the base vectors are not unitary:

$$f_i = t_i \left(\rho + \frac{1}{c_s^2} \rho \, \vec{c}_i \cdot \vec{u} + \frac{1}{2\, c_s^4} \boldsymbol{\Pi} : \boldsymbol{Q} + h_i \left(\frac{1}{4} N + \frac{3}{8} \vec{c}_i \cdot \vec{J} \right) \right), \tag{3.34}$$

where the projections on the ghost vectors have been named $\langle \mathcal{C}^3 | f \rangle = N$ and $\langle \mathcal{C}^4 | f \rangle = \vec{J}$ respectively. It is illuminating to split Eq. (3.34) into an equilibrium and an off-equilibrium contribution:

$$f_i^{eq} = t_i \left(\rho + \frac{1}{c_s^2} \rho \, \vec{c}_i \cdot \vec{u} + \frac{1}{2\, c_s^4} \boldsymbol{\Pi}^{eq} : \boldsymbol{Q} \right) \quad \text{and} \tag{3.35}$$

$$f_i^{neq} = t_i \left(\frac{1}{2\, c_s^4} \boldsymbol{\Pi}^{neq} : \boldsymbol{Q} + h_i \left(\frac{1}{4} N + \frac{3}{8} \vec{c}_i \cdot \vec{J} \right) \right). \tag{3.36}$$

With this notation, the BGK collision is written as follows:

$$\Omega_i = -\omega\, t_i \left(\frac{1}{2\, c_s^4} \boldsymbol{\Pi}^{neq} : \boldsymbol{Q} + h_i \left(\frac{1}{4} N + \frac{3}{8} \vec{c}_i \cdot \vec{J} \right) \right). \tag{3.37}$$

A MRT model has exactly the same form, except for the fact that all non-conserved moments possess their own relaxation parameter, numbered through from ω_3 to ω_8:

$$\begin{aligned} \Omega_i = -t_i \Bigg[& \frac{1}{c_s^4} \left(\omega_3\, \Pi_{xx}^{neq} Q_{xx} + \omega_4\, \Pi_{xy}^{neq} Q_{xy} + \omega_5\, \Pi_{yy}^{neq} Q_{yy} \right) \\ & + h_i \left(\omega_6\, \frac{1}{4} N + \frac{3}{8} \left(\omega_7\, c_{ix}\, J_x + \omega_8\, c_{iy}\, J_y \right) \right) \Bigg]. \end{aligned} \tag{3.38}$$

Equation (3.38) is particularly illuminating, because it displays both the space and moment representations of the dynamics at the same time. In Section 4, a new LB model is introduced whose collision operator is linear and diagonalizable. The MRT interpretation of this model, presented in Section 4.3, will show that the hydrodynamic parameters ω_3 to ω_5 are the same as in the BGK model, whereas ω_6 to ω_8 are modified.

Note 3.3: Terminology *Some confusion in the terminology occurs sometimes when the term "MRT model" is used for all LB models with a linear, diagonalizable collision operator. This puts the emphasis on the wrong place. Such a terminology would actually imply that practically all commonly used LB models are MRT models, including the BGK model, and several other models that were developed even before the term "MRT" existed. In particular, specific LB models based on a linearized collision operator have been described in Refs. [10, 25, 26, 27]. In the literature, the name "MRT" is rather used for models that where explicitly constructed by switching to a moment representation and fine-tuning the relaxation parameters.*

3.3.2 Entropic LB models

The family of *entropic LB models* has emerged over the past decade and is successively gaining importance. Two approaches were developed independently by Karlin, Ansumali *e.a.*, as documented in Refs. [28, 29], and by Boghosian *e.a.*, documented in Ref. [30]. Those models exploit a discrete-space analogue of Boltzmann's H-theorem to achieve unconditional numerical stability. Numerical stability means that the simulated values always remain within a fixed range and never exceed the storage capacity of floating point representations in a computer. It does however not imply that the simulated values systematically stay close to the exact solution of the problem, nor that they even yield a physically meaningful result. Unconditional numerical stability can be very useful when simulations are run close to a regime in which they are underresolved. It might happen that the simulation is locally underresolved during a short time interval, which should only affect the overall quality of the simulation to a small extent. In such a situation, numerical instabilities can however occur in relation with the BGK model, but not with entropic models.

Boltzmann's H-theorem states that each fluid possesses a property H that never increases during the time-evolution of a closed system (a system without energy input). A local version of this theorem additionally states that a locally defined property $H = H(f(\vec{x}, t))$ never increases during particle collisions. Because of numerical errors, this theorem is only approximately true in BGK simulations. In entropic models however, the collision operator is modified so as to obtain an exact discrete version of the H theorem. This can be compared to the local conservation laws for mass and momentum, which are exactly respected in the discrete LB equation.

The unconditional numerical stability is achieved by means of the three following steps:

1. A function $H = H(f_i(\vec{x}))$ is constructed, which is defined over the space of strictly positive particle distribution functions f_i, and which is convex. The minima of this function is given by the local equilibrium f_i^{eq}.

2. The H value for the incoming distribution functions $H^{in} = H(f_i^{in})$ is evaluated, and the set of distribution functions f_i for which $H(f_i) \leq H^{in}$ is evaluated.

3. The collision is modified so as to ensure that the outgoing distribution functions f_i^{out} are contained within the set of distribution functions evaluated previously. This in its turn ensures that all distribution functions remain strictly positive. It has actually been observed empirically that numerical instabilities only occur in the LB method when negative distribution functions appear. This effect is technically prevented in entropic methods by the exact H theorem.

Chapter 4

Regularized lattice Boltzmann

4.1 Introduction

In Section 1.3, a lattice Boltzmann model is introduced that is expressed in a system of lattice units. In this system, the spacing between two adjacent lattice cells, as well as the time span between two successive iteration steps, is unitary. To motivate why this system of units is a natural choice in LB simulation, it is useful to adopt the interpretation of the equilibrium distribution function in Eq. (1.18) on page 10 as a small Mach number expansion of a Maxwellian velocity distribution (see Note 2.5). The Mach number is a dimensionless number computed from the ratio between a characteristic velocity of the system and the speed of sound. As it was shown in Section 2.3, the speed of sound depends on the choice of the discrete space and time steps, and it is constant in a system of lattice units. On a D2Q9 lattice for example, and in a system of lattice units, the speed of sound is defined by $c_s^2 = 1/3$. This shows that apart from this factor $1/3$, the characteristic flow velocity $\vec{u}$ is equivalent to the Mach number. The equilibrium distribution function is thus most naturally written in lattice units, in which it does not explicitly depend on the parameters δ_x and δ_t. This approach contrasts with one of other common CFD tools, where the system of dimensionless units introduced in Section 1.2.3 is the most natural choice for the formulation of a dynamics independent of discretization parameters.

It is of course possible to switch from one system of units to another. To interpret the results of a LB simulation in terms of dimensionless macroscopic variables, the moments of the distribution functions are computed and then rescaled in an appropriate way. The dimensionless velocity is for example obtained through the expression

$$\vec{u}^* = \frac{\delta_x}{\delta_t}\,\vec{u} = \frac{\delta_x}{\delta_t}\,\frac{1}{\rho}\sum_i f_i\,\vec{c}_i. \tag{4.1}$$

It is much more contrived however to initialize the particle distribution functions of a LB simulation from a given prescription of the macroscopic flow fields. The results from the Chapman-Enskog multiscale analysis are used to explicit this relation. From Eqs. (1.18) and (2.65), sparing out the force term and the correction term to

keep the discussion simple, one obtains:

$$\begin{aligned} f_i &= f_i^{eq} + \epsilon f_i^{(1)} + \mathcal{O}(\epsilon^2) \\ &= \rho t_i \left(1 + \frac{1}{c_s^2}\vec{c}_i \cdot \vec{u} + \frac{1}{2c_s^4}\boldsymbol{Q}_i : \vec{u}\vec{u}\right) \\ &\quad - \frac{t_i}{c_s^2\omega}\left(\boldsymbol{Q}_i : \rho\overrightarrow{\nabla}\vec{u} - \vec{c}_i\overrightarrow{\nabla} : \rho\vec{u}\vec{u} + \frac{1}{2c_s^2}(\vec{c}_i \cdot \overrightarrow{\nabla})(\boldsymbol{Q}_i : \rho\vec{u}\vec{u})\right) + \mathcal{O}(\epsilon^2). \end{aligned} \tag{4.2}$$

This equation shows that in order to relate a macroscopic flow field to a LB field, various types of velocity gradients need to be computed. Furthermore, the relatively complicated expression of Eq. (4.2) needs to be evaluated. This is technically feasible, but tends to be very expensive if it has to be done at every iteration step. This conversion is to be executed not only when values from an external flow field are imported into a LB simulations, but also when two LB simulations, executed on different grids, are related to each other. This becomes clear when Eq. (4.2) is rewritten in terms of dimensionless variables:

$$\begin{aligned} f_i = \rho t_i &\left(1 + \frac{\delta_t}{\delta_x}\frac{1}{c_s^2}\vec{c}_i \cdot \vec{u}^* + \frac{\delta_t^2}{\delta_x^2}\frac{1}{2c_s^4}\boldsymbol{Q}_i : \vec{u}^*\vec{u}^*\right) \\ &- \frac{t_i}{c_s^2\omega}\left(\delta_t\,\boldsymbol{Q}_i : \rho\overrightarrow{\nabla}^*\vec{u}^* - \frac{\delta_t^2}{\delta_x}\left(\vec{c}_i\overrightarrow{\nabla}^* : \rho\vec{u}^*\vec{u}^* + \frac{1}{2c_s^2}(\vec{c}_i \cdot \overrightarrow{\nabla}^*)(\boldsymbol{Q}_i : \rho\vec{u}^*\vec{u}^*)\right)\right). \end{aligned} \tag{4.3}$$

It is concluded from this equation that different contributions to the particle distribution functions scale at a different rate under a change of δ_x and δ_t. To compare the results from two LB simulations that are run on different grids, it is therefore necessary to split the f_i into their components, and to rescale each component individually. It is possible, using the values contained locally on a lattice cell, to compute ρ and $\vec{u}$, and thus to compute the various components of f^{eq}, and to separate the equilibrium contributions from the off-equilibrium part. It is however not possible to split the off-equilibrium part into its components, because not all velocity gradients are known locally on a lattice cell: only the symmetric part of the Jacobian matrix can be computed by means of Eq. (2.66). The skew symmetric part needs to be evaluated by some other, non-local method.

It is possible to rescale the particle distribution functions by purely local means if the $\mathcal{O}(u^2)$ terms in the off-equilibrium distribution functions are neglected. In that case, all components scale in a known way: the density ρ is invariant, the velocity $\vec{u}$ scales like δ_t/δ_x, and the off-equilibrium distributions $f_i^{(1)}$ scale like δ_t. This approach is chosen in Ref. [31] to formulate a grid-refinement algorithm for the LB method. It must be pointed out that this is one of the rare papers on a related topic that explicitly tackles the problems of scale invariance between two grids during a grid refinement procedure.

A different approach is presented here that is not based on an approximation of the distribution function. The idea behind this method is to replace the BGK model by a new model, the regularized LB (RLB) model, which has less degrees of freedom. This new model depends only on the value of ρ, $\vec{u}$ and $\boldsymbol{\Pi}^{(1)}$. Each of them can be rescaled individually, as the tensor $\boldsymbol{\Pi}^{(1)}$ is related to the tensor S of Eq. (1.6) through Eq. (2.66). The details of the regularized method are explained

in Section 4.2. It is found that additionally to the advantages mentioned previously, the regularized LB model is exceptionally accurate and numerically stable. This point is verified in Section 4.4 on chosen numerical examples.

In the subsequent chapters, various typical problems of computational fluid dynamics are listed in which the variables of a lattice Boltzmann simulation need to be rescaled. The regularized LB model is proposed as a solution in each of those cases, and the change of scales is handled in a straightforward way via a dimensionless formulation of the variables ρ, $\vec{u}$ and $\mathbf{\Pi}^{(1)}$. The described procedures are actually also applicable to the BGK model, in which the tensor $\mathbf{\Pi}^{(1)}$ can be computed locally. The obtained results are however slightly less accurate, because by doing so, the $\mathcal{O}(u^2)$ terms in the off-equilibrium parts of the distribution functions are not taken into account.

4.2 Regularization procedure

4.2.1 Regularized particle distribution functions

The particle distribution functions f_i are expanded by means of a multi-scale analysis in Section 2.3. In particular, the value of the first order contribution $f_i^{(1)}$ is explicited in Eq. (2.65) in terms of the macroscopic variables ρ and $\vec{u}$. This term is however of limited interest, because it does not explicitly appear in the macroscopic PDE's that describe the hydrodynamic behavior of the model, Eqs. (2.35,2.71). Instead, Eq. (2.71) refers to the tensor $\Pi^{(1)}$, which is the projection of $f_i^{(1)}$ on the second order base vectors $\mathcal{C}^2_{\alpha\beta}$ shown in Eq. (2.66). During this projection, many terms cancel out, and only those components of $f_i^{(1)}$ that are expanded on $\mathcal{C}^2_{\alpha\beta}$ are kept. The idea of the regularized LB method is to simplify the value of $f_i^{(1)}$ accordingly, and to keep only those terms that are relevant to the computation of $\mathbf{\Pi}^{(1)}$.

It is useful at this point to take up again the expression for $f_i^{(1)}$ from Eq. (2.65) and to split it into contributions that are relevant (R_i) resp. irrelevant (I_i) to the computation of $\Pi^{(1)}$:

$$f_i^{(1)} = R_i + I_i, \quad \text{where} \tag{4.4}$$

$$R_i = t_i \boldsymbol{Q}_i : \left(-\frac{1}{c_s^2\,\omega}\,\rho\,\overrightarrow{\nabla}_1\vec{u} - \frac{1}{4\,c_s^4}\left(\vec{F}\vec{u} + \vec{u}\vec{F}\right) + \frac{c}{2\,c_s^2\,\omega}\delta\mathbf{\Pi}\right) \quad \text{and} \tag{4.5}$$

$$I_i = \frac{t_i}{c_s^2\,\omega}\left(\vec{c}_i\overrightarrow{\nabla}_1 : \rho\vec{u}\vec{u} - \frac{1}{2\,c_s^2}\left(\vec{c}_i\cdot\overrightarrow{\nabla}_1\right)(\boldsymbol{Q}_i : \rho\vec{u}\vec{u})\right) - \frac{1}{2}\frac{t_i}{c_s^2}\vec{c}_i\cdot\vec{F}. \tag{4.6}$$

The term R_i is of the form $R_i = t_i\,\boldsymbol{Q}_i : \boldsymbol{T}$, where the tensor $\boldsymbol{T}$ stands for the whole content of the parentheses on the right hand side of Eq. (4.5). The tensor $\mathbf{\Pi}^{(1)}$ was calculated in Eq. (2.66) by projecting R_i on the vectors $\mathcal{C}^2_{\alpha\beta}$ as follows:

$$\mathbf{\Pi}^{(1)}_{\alpha\beta} = \sum_i t_i\,Q_{i\alpha\beta}Q_{i\gamma\delta}\,T_{\gamma\delta} = c_s^4\,(T_{\alpha\beta} + T_{\beta\alpha}) \quad \text{by Eq. (2.12)}. \tag{4.7}$$

The key to the regularization procedure is the observation that the evaluation of $\mathbf{\Pi}^{(1)}$ in Eq. (4.7) can be reversed. Once the tensor $\mathbf{\Pi}^{(1)}$ is known, the tensor $\boldsymbol{T}$ can

be evaluated in its turn. This is done by contracting $\mathbf{\Pi}^{(1)}$ with the tensor $\boldsymbol{Q}$:

$$\boldsymbol{Q}_i : \mathbf{\Pi}^{(1)} = c_s^4 \, \boldsymbol{Q}_i : (\boldsymbol{T} + \boldsymbol{T}^T) = 2\, c_s^4 \, \boldsymbol{Q}_i : \boldsymbol{T} \quad \text{by symmetry of the tensor } \boldsymbol{Q}. \tag{4.8}$$

From this, a regularized substitute $\bar{f}_i^{(1)}$ for $f_i^{(1)}$ is derived:

$$\bar{f}_i^{(1)} = R_i = \frac{t_i}{2\, c_s^4} \boldsymbol{Q}_i : \mathbf{\Pi}^{(1)}. \tag{4.9}$$

4.2.2 Dimensionless formulation

It is interesting to remind that up to the accuracy of the hydrodynamic terms, the tensor $\mathbf{\Pi}^{(1)}$ can be related to the strain rate tensor $\boldsymbol{S}$. Indeed, using Eq. (2.66) on page 25 yields, when the force and the correction term are not considered:

$$\bar{f}_i^{(1)} \approx -\frac{\rho\, t_i}{c_s^2\, \omega} \boldsymbol{Q}_i : \boldsymbol{S}. \tag{4.10}$$

The tensor $\boldsymbol{S}$, defined by Eq. (1.6), is a macroscopic quantity that depends only on the gradients of the velocity field. To conclude this discussion, it is pointed out that the set of lattice kinetic variables defined by the regularized particle distribution functions $\bar{f}_i = f_i^{eq} + \bar{f}_i^{(1)}$ is equivalent to a set of macroscopic variables defined by the scalar ρ, the vector $\vec{u}$ and the tensor $\mathbf{\Pi}^{(1)}$. Those variables, and thus the state of the simulation, can be represented in a dimensionless system of units:

$$\begin{aligned} \rho^* &= \rho, \\ \vec{u}^* &= \frac{\delta_t}{\delta_x} \vec{u} \quad \text{and} \\ \mathbf{\Pi}^{(1)*} &= \delta_t \, \mathbf{\Pi}^{(1)}. \end{aligned} \tag{4.11}$$

4.2.3 Algorithm

To summarize, the regularized LB method replaces the usual BGK method in Eqs. (1.14,2.52) by the following procedure:

1. Compute the equilibrium distribution $f_i^{(eq)}$ from the particle distribution functions by means of Eq. (1.18).

2. Determine the off-equilibrium distribution functions $f_i^{(neq)} = f_i - f_i^{(eq)}$ and the related tensor $\mathbf{\Pi}^{(neq)} = \sum_i \boldsymbol{Q}_i f_i^{(neq)}$.

3. Derive the first order regularized distribution functions $\bar{f}_i^{(1)}$ from $\mathbf{\Pi}^{(neq)}$ through the approximation $\mathbf{\Pi}^{(1)} \approx \mathbf{\Pi}^{(neq)}$ in Eq. (4.9).

4. Replace the distribution functions f_i by their regularized counterpart $\bar{f}_i = f_i^{(eq)} + \bar{f}_i^{(1)}$.

5. Execute the BGK collision and the streaming step.

In order to clarify this algorithm, the original BGK LB model (Eqs. (1.14,2.52)) is first reformulated as follows:

$$\begin{aligned} f_i(\vec{r}+\vec{c}_i,t+1) &= f_i(\vec{r},t) - \omega\left(f_i(\vec{r},t) - f_i^{(eq)}(\rho,\vec{u})\right) + FT_i + CT_i \\ &= f_i^{(eq)}(\rho,\vec{u}) + (1-\omega)\, f_i^{(neq)}(\vec{r},t) + FT_i + CT_i. \end{aligned} \tag{4.12}$$

By analogy to this equation, the regularized model can be summarized as follows:

$$\begin{aligned} f_i(\vec{r}+\vec{c}_i,t+1) &= f_i^{(eq)}(\rho,\vec{u}) + (1-\omega)\, \bar{f}_i^{(1)}(\vec{r},t) + FT_i + CT_i \\ &= f_i^{(eq)}(\rho,\vec{u}) + \frac{(1-\omega)\, t_i}{2\, c_s^4}\, \boldsymbol{Q}_i : \boldsymbol{\Pi}^{(1)}(\vec{r},t) + FT_i + CT_i. \end{aligned} \tag{4.13}$$

4.2.4 Related models

It is emphasized that the regularization procedure described in this section principally applies to all LB models of the BGK family. The idea is throughout the same: the first-order term $f_i^{(1)}$ is projected on the vectors $\mathcal{C}_{\alpha\beta}^2$ corresponding to the first non-conserved moment, and the resulting regularized term $\bar{f}_i^{(1)}$ is directly computed from the value of this non-conserved moment. For the purpose of illustration, the advection-diffusion model described in Section 2.2 is now regularized. When the error and correction terms of this model cancel, for example by means of the procedure described in Section 3.1, the first-order contribution to the model reads

$$f_i^{(1)} = -\frac{\lambda}{c_s^2} t_i\, \boldsymbol{Q}_i : \overrightarrow{\nabla}_1(\rho\vec{u}) - \lambda\, t_i\, \vec{c}_i \cdot \overrightarrow{\nabla}_1 \rho. \tag{4.14}$$

The first non-conserved moment of this model is the momentum $\vec{j}$, and the regularized version of Eq. (4.14) takes therefore the following form:

$$\bar{f}_i^{(1)} = -\lambda\, t_i\, \vec{c}_i \cdot \overrightarrow{\nabla}_1 \rho. \tag{4.15}$$

Indeed, the momentum computed from those two expressions is the same:

$$\vec{j} = -\lambda\, c_s^2 \overrightarrow{\nabla}_1 \rho, \tag{4.16}$$

from which the regularized term $\bar{f}_i^{(1)}$ is computed as follows:

$$\bar{f}_i^{(1)} = \frac{t_i}{c_s^2} \vec{c}_i \cdot \vec{j}. \tag{4.17}$$

The LB model introduced here under the name of RLB has previously been described in the literature, where it was derived from different principles and applied in different settings. In Ref. [32], this model is used for the simulation of particle simulations, whereas in Ref. [33], it is introduced as a model for the simulation of high Knudsen number flows. The derivation of RLB from the Chapman-Enskog expansion of a BGK model however is novel. Furthermore, the particular properties of RLB, such as the enhanced stability and accuracy, as well as the adequacy of the model to situations with varying time and space scales, have not been pointed out before.

Note 4.1: Memory savings *The state of a RLB simulation is fully described by the three quantities ρ, $\vec{u}$ and $\mathbf{\Pi}^{(1)}$. It is therefore possible to write a program that holds only those variables in memory instead of the particle distribution functions. This leads to memory savings because it requires a number of $1 + d + d(d+1)/2$ variables to be stored, less than the usual q particle distribution functions. On the other hand, the resulting code is quite different from a typical LB code and may be more difficult to work with. Furthermore, it depends on the particular software and hardware platform whether this alternative formulation of RLB leads to performance benefits or not.*

4.3 RLB and MRT

The collision operator of the RLB model is linear and acts only on the off-equilibrium parts of the particle distribution functions. This fact appears clearly when Eq. (4.13) is formulated as follows:

$$f_i(\vec{r}+\vec{c}_i, t+1) = f_i^{(eq)}(\rho, \vec{u}) + (1-\omega)\sum_j \frac{t_i}{2\,c_s^4}\,\boldsymbol{Q} : \vec{c}_j\vec{c}_j\,\left(f_j - f_j^{eq}\right), \tag{4.18}$$

or, to emphasize the analogy with a general collision operator as in Eq. (3.27) on page 39:

$$f_i^{out} = \left(\sum_j \frac{1-\omega}{2\,c_s^4}\,\boldsymbol{Q}_i : \vec{c}_j\vec{c}_j - \delta_{ij}\right)\left(f_j - f_j^{eq}\right). \tag{4.19}$$

The RLB model can thus be reinterpreted as a LB model with multiple relaxation times, by using the theory introduced in Section 3.3.1. As an example, the D2Q9 lattice is now considered. Using the same notation as in Eq. (3.37), and after some algebra, the collision operator of Eq. (4.19) is written as follows:

$$\Omega_i = -t_i\left(\omega\,\frac{1}{2\,c_s^4}\mathbf{\Pi}^{neq} : \boldsymbol{Q} + h_i\left(\frac{1}{4}N + \frac{3}{8}\vec{c}_i\cdot\vec{J}\right)\right). \tag{4.20}$$

This means that all modes of $\mathbf{\Pi}^{neq}$ are relaxed with a parameter ω, as in the BGK model, and all other modes are relaxed with a parameter 1.

4.4 Numerical verification

We now turn to numerical verifications of the regularized model on two 2D flows using a D2Q9 lattice. The first test case, known under the name of *Kovasznay flow*, approximates a stationary 2D flow behind a regular grid. An analytical solution for this flow, proposed in [34], is written as follows in dimensionless units:

$$\begin{aligned} u_x^* &= 1 - \exp(\lambda\,x^*)\cdot\cos(2\pi y^*) \quad \text{and} \\ u_y^* &= \frac{\lambda}{2\pi}\exp(\lambda\,x^*)\cdot\sin(2\pi y^*), \end{aligned} \tag{4.21}$$

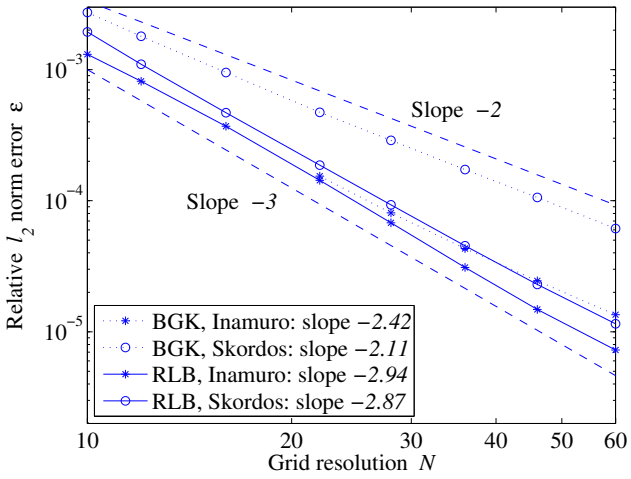


Figure 4.1: Relative error of the numerical result on a Kovasznay flow in the wake region. Both traditional BGK and the regularized model are tested with two commonly used boundary conditions.

with

$$\lambda = Re/2 - \sqrt{4\pi^2 + Re^2/4}. \tag{4.22}$$

The Reynolds number $Re = u_\infty L/\nu$ is defined as a function of the asymptotic velocity u_∞ of the velocity in the wake, far from the grid, the spacing L between two grid nodes and the dynamic viscosity ν of the fluid. The simulations are executed in the wake of the grid, in the intervals $x \in [L/2, 2L]$ and $y \in [-L/2, 3L/2]$, with $Re = 10$. The space step δ_x is varied linearly, while the ratio between the discrete time and space step is fixed to a value $\delta_t/\delta_x = 0.01$. As it follows from the discussion in Section 2.4, keeping the ratio δ_t/δ_x constant amounts to fixing the Mach number $Ma = u/c_s$. This value is chosen small enough to mimic an incompressible flow. Given that the flow is periodic in the y-direction, the upper and the lower boundary of the simulation can be chosen to be periodic, whereas the Kovasznay solution in Eq. (4.21) is imposed through Dirichlet boundary conditions on the left and right boundary. After the simulation has reached a time independent state, the numerical result is compared with the solution in Eq. (4.21) through an l_2 norm on each grid point, and then averaged over space. The result is shown in Fig. 4.1, on two commonly used implementations of the boundary conditions. One of them has been described by T. Inamuro *e.a.* in Ref. [35] and the other one by P. Skordos in Ref. [36]. The accuracy of the simulation with respect to the grid resolution is of order 2 to 2.5 when the BGK model is used, whereas the regularized model is almost third-order accurate. On the BGK simulations with the Inamuro boundary condition, data points for small grids are missing because numerical instabilities make them impossible, whereas the regularized model has no such stability deficiencies.

The second test case implements a flow in a 2D square cavity whose top-wall moves with a uniform velocity. Reference values for the definition of the Reynolds number are defined by the side length L of the box and the top-wall velocity $\vec{u_0}$. Both standard BGK and the regularized model are first compared with the reference

solution of Ghia *e.a.* [37], on a lattice size of $N \times N$ with $N = 129$, at $Re = 100$ and a Mach number, in lattice units, defined as $u_0 = \delta_t/\delta_x = 0.02$. This time, another commonly used boundary condition is applied, which is described by Q. Zou and X. He in Ref. [38]. The reference solution [37] proposes a set of accurate numerical values for some x- and some y-components of the velocity on chosen space points. A l_1 norm error with respect to these reference points is averaged over all available points and normalized with respect to u_0. For the BGK model, this yields an error of $\epsilon = 3.71{\cdot}10^{-3}$, and for the regularized method, of $\epsilon = 2.40{\cdot}10^{-3}$. Thus, both methods solve the problem with satisfying accuracy. This fact is underlined graphically on Fig. 4.3, on which the numerical results of the RLB simulation and the solution by Ghia *e.a.* are compared. The regularized model is however found to be substantially more stable. To make this statement more quantitative, a series of simulations is run, on which the velocity is again fixed to a value of $u_0 = 0.02$. For several chosen grid sizes N, the maximal Reynolds number Re^{max} at which the simulation remains stable (*i.e.* delivers finite numerical values) is determined. Figure 4.2 shows that, although both methods exhibit a linear relationship between Re^{max} and N, the observed increase rate is 7.7 times higher for the regularized method than for BGK.

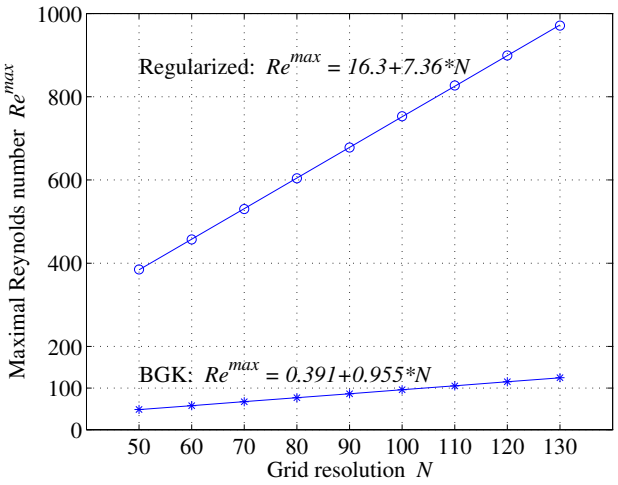


Figure 4.2: Simulation of 2D cavity flow for fixed Mach number. ∘, ∗: maximal stable Reynolds number, numerically determined; solid line: least-square linear fit of the data points (parameters of the fit are indicated on the graph).

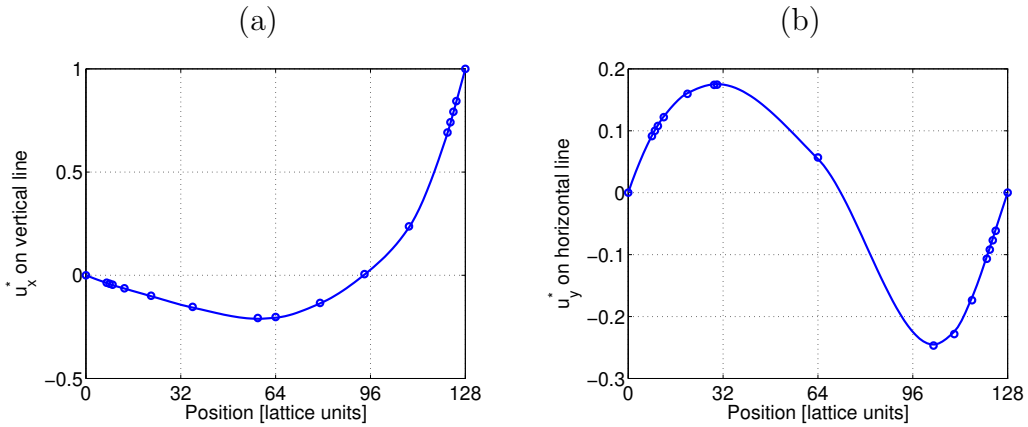


Figure 4.3: Comparison of computed RLB results with reference values from the literature for the 2D lid-driven cavity flow at $Re = 100$. The solid line represents the reference values, which have been interpolated to guide the eye, and the dots stem from numerical results. (a) x-component of the velocity along a vertical line passing through the center of the cavity. (b) y-component of the velocity along a horizontal line passing through the center of the cavity.

Chapter 5

Boundary and initial conditions

5.1 Introduction

The Navier-Stokes equation possesses a unique solution when adequate initial and boundary values for a given problem are specified. The problems of fluid dynamics are therefore *initial and boundary value problems*. It is common to specify the initial value for a problem by imposing the velocity field at the initial time $t = 0$. Several approaches exist for specifying the boundaries of a flow. It is possible to implement a so-called *Dirichlet boundary condition* by specifying a velocity profile along the domain boundaries of the problem. In other commonly adopted approaches, the value of the velocity gradient in the direction of the boundary normal is specified, or the value of the pressure on the boundary.

Whatever approach is adopted, one major difficulty needs to be addressed in LB simulations: the conversion from macroscopic variables, which are taken from the initial or boundary condition, to the particle distribution functions of a LB simulation. When the regularized LB model is used, it is natural to implement this conversion by means of the approach described in Chapter 4. In this approach, the tensor $\mathbf{\Pi}^{(1)}$ is computed additionally to the pressure p and the velocity $\vec{u}$. By virtue of Eq. (4.9) on page 46, this set of variables fully specifies the state of the LB simulation.

During the implementation of an initial condition, the tensor $\mathbf{\Pi}^{(1)}$ can be related to the strain rate tensor $\boldsymbol{S}$, as defined in Eq. (1.6) on page 7. The velocity gradients contained in this tensor can in their turn be computed either analytically, when the initial velocity profile is defined through an analytical expression, or otherwise numerically, by applying for example a finite difference scheme. It is actually shown in Section 5.3 that it is more difficult to properly initialize the pressure field in the initial condition. In this section, a time-dependent benchmark problem is introduced whose results depend heavily, among others, on the initial condition of the problem. The initialization procedure described here, inspired from a macroscopic vision of the problems of CFD, is tested numerically. It is compared with another approach taken from the literature that rather adopts a microscopic view and is implemented fully in terms of LB variables.

When the boundary condition is implemented, the velocity gradients cannot be evaluated analytically, even when the velocity profile along the boundary is known in terms of an analytical formula. Indeed, the gradients normal to the boundary

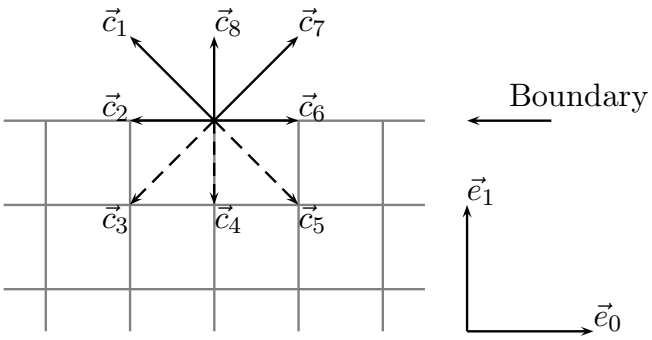


Figure 5.1: Orientation of a 2D boundary for the example developed in the text. Dashed arrows label directions that originate from locations outside the lattice.

depend on the results of the simulation and are *a priori* unknown. It is possible to evaluate those gradients by means of a finite difference scheme, and close the system of equations on the boundary in this way. Although this approach is feasible, it violates partly the spirit of the LB methods, in which all operations, except for the streaming step, tend to be purely local. In Section 5.2, another approach is therefore shown, in which the elements of $\boldsymbol{S}$ are computed from the known particle distribution functions on the wall.

5.2 Boundary condition

5.2.1 Overview

This section focuses on the implementation of Dirichlet boundary conditions for the velocity field. Following the approach discussed in the introduction, implementing the boundary condition amounts to computing the values for the missing variables ρ and $\boldsymbol{\Pi}^{(1)}$. In this discussion, only straight boundaries are considered that are parallel to the lattice edges and pass through lattice nodes. Other types of boundaries, such as corner nodes, can be implemented via the same approach. The discussion of this topic is however omitted here because it is quite technical. The interested reader is invited to directly refer to the publicly available and well documented code of the `OpenLB` project in Ref. [13]. The orientation of the boundary is taken to be orthogonal to the unitary vector $\vec{e}_1$. This means that for a lattice location $\vec{r}$ that lies on the boundary, the location $\vec{r} + \delta_x \vec{e}_1$ lies outside the computational domain, and the location $\vec{r} - \delta_x \vec{e}_1$ lies inside, as shown on Fig. 5.1. As a consequence, all neighbors of a boundary location $\vec{r}$, placed at a position $\vec{r} - \vec{c}_k$, are unreachable when $c_{k1} = -1$. However, particle streams parallel to the boundary, along directions $\vec{c}_l$ for which $c_{l1} = 0$ participate in the LB dynamics of the simulation. This approach is based on a point of view advocated in Ref. [39], according to which boundary nodes are located inside the fluid, but infinitesimally close to the wall. The problem with the computation of ρ and $\boldsymbol{\Pi}^{(1)}$ resides in the fact that among all distribution functions f_i on the boundary those with an index $i \in \{k \,|\, c_{k1} > 0\}$ are unknown and must be evaluated on ground of some additional closure relations.

5.2.2 Implementation of the pressure

The first step consists in computing the variable ρ on the wall. In an incompressible fluid, Eq. (1.8) on page 7 shows that this is equivalent to computing the pressure. A fairly standard procedure for doing so is described for example in Refs. [38, 35]. It is based on the definition of three separate contributions to the density:

$$\rho_1 = \sum_{\{k\,|\,c_{k1}=1\}} f_k, \tag{5.1a}$$

$$\rho_0 = \sum_{\{k\,|\,c_{k1}=0\}} f_k, \quad \text{and} \tag{5.1b}$$

$$\rho_{-1} = \sum_{\{k\,|\,c_{k1}=-1\}} f_k, \tag{5.1c}$$

of which the component ρ_1 is unknown on the boundary. From Eqs. (1.15) and (1.16) on page 10, one finds that

$$\rho = \rho_1 + \rho_0 + \rho_{-1} \quad \text{and} \quad \rho u_1 = \rho_1 - \rho_{-1}, \tag{5.2}$$

and ρ can be formulated in terms of known quantities:

$$\rho = \frac{1}{1-u_1}(2\rho_{-1} + \rho_0). \tag{5.3}$$

Note 5.1: Pressure boundaries *Although only Dirichlet velocity boundaries are discussed in this text, other boundary types are defined in a very similar way. Equation (5.3) can for example be inverted to compute the normal velocity component u_1 from the density ρ. This leads to the definition of a so-called* pressure boundary*, on which the density ρ and the tangential velocity u_0 are specified, and the normal velocity u_1 is free.*

5.2.3 Regularization of on-wall distribution functions

Now that the density ρ and the velocity $\vec{u}$ are known, a last closure relation for the evaluation of the stress tensor $\mathbf{\Pi^{(1)}}$ is required, before the regularized distribution functions can be calculated via Eq. (4.9) on page 46. An alternative approach consists in computing the macroscopic strain rate tensor $\boldsymbol{S}$ instead of $\mathbf{\Pi}^{(1)}$ and by evaluating the regularized particle distribution functions via Eq. (4.10) on page 46. In that case, the velocity gradients of the strain rate tensor are found with an asymmetric second-order accurate finite difference scheme on the boundary. This computation involves values residing on nearest and a next-to-nearest neighbors of the boundary node. The resulting BC is similar to the one presented in Ref. [36] in which the values of the distribution functions on the boundary are fitted according to their zeroth- and first-order terms of the Chapman-Enskog expansion. It is however not typical for simple LB methods to make use of non-local finite difference schemes, as one prefers to implement a collision step depending only on values defined locally on a lattice site, both for conceptual and technical simplicity. A purely local means

to computing $\mathbf{\Pi}^{(1)}$ is therefore presented, which is based on knowledge of the particle distribution functions f_i that are known on the boundary. The dynamics of the wall is implemented in two steps:

Compuation of $\mathbf{\Pi}^{(1)}$. To all missing distribution functions f_i on the wall, attribute a value $f_i \equiv f_i^{eq}(\rho, \vec{u}) + f_j^{neq}$, where j is the index of the velocity opposite to $\vec{c}_i$: $\vec{c}_i = -\vec{c}_j$. This procedure is commonly applied and has been called "bounce-back of off-equilibrium parts" in Ref. [38]. It is motivated by symmetries in the regularization formula Eq. (4.9), from which it is easily deduced that the off-equilibrium part for two distribution functions $\bar{f}_i$ attributed to opposite directions is equal. In conclusion, the obtained distribution functions possess values that are compatible with the order $\mathcal{O}(\epsilon)$ development of the BGK dynamics. They are however not yet fully satisfying, as they lead to slightly erroneous values for the density ρ and the velocity $\vec{u}$. For this reason, they are only used to compute the value of the stress tensor $\mathbf{\Pi}^{(1)} = \sum_i f_i - c_s^2 \rho \boldsymbol{I} - \rho \vec{u}\vec{u}$ on the wall.

Regularization. Replace all distribution functions on the wall by their value obtained from the variables ρ, $\vec{u}$ and $\mathbf{\Pi}^{(1)}$ via Eq. (4.9).

Note 5.2: Conservation laws *In the two-step implementation of a boundary condition presented in this paragraph, it can seem appropriate to simply skip step (2). Indeed, the missing particle distribution functions on the wall are initialized in step (1) to a reasonable value that respects the conclusions of the Chapman-Enskog analysis. The problem is that the exact conservation laws for the mass and the momentum (see Note 2.4 on page 21) would not be respected by this procedure. The distribution functions computed in step (1) do for example not respect the relation $\sum_i f_i^{neq} = 0$, which is required for exact mass conservation. It must be noted that exact conservation laws are important both for theoretical and practical reasons, as they ensure for example a good numerical stability of LB models. The regularization step (2) restores symmetries of the distribution functions and leads to exact conservation laws. The boundary condition described in Ref. [38] chooses to apply the "bounce-back of off-equilibrium parts"-rule of step (1) only to one distribution function which is orthogonal to the wall, and it uses the remaining degrees of freedom to explicitly enforce the conservation laws.*

One particular strength of the described model is that it can be implemented on just any lattice, independently on the number of dimensions and the number of distribution functions. This is a striking feature that distinguishes the regularized boundary condition from other approaches. It is for example reflected in the LB source code of Ref. [13] in which a generic implementation for the lattice boundaries exists. An application programmer can for example chose to define a new lattice structure and immediately apply it to a given problem, as both the LB dynamics and the boundary conditions are generic and adapt to the new lattice. Nevertheless, to help with the implementation of typical codes, explicit formulas for the tensor $\mathbf{\Pi}^{(1)}$ are listed for the D2Q9 and the D3Q19 lattice structures (those two structures

are specified in Section 1.3.3 on page 11). On a D2Q9 lattice, the following value are obtained:

$$\begin{aligned}
\mathbf{\Pi}^{(1)}_{00} &= f_2 + f_6 + 2(f_1 + f_7) - \\
&\quad \rho\left(\frac{u_1}{3} + u_0^2 + \frac{1}{3}\right), && \text{(5.4a)}\\
\mathbf{\Pi}^{(1)}_{11} &= 2\left(f_1 + f_7 + f_8\right) - \rho\left(u_1 + u_1^2 + \frac{1}{3}\right), && \text{(5.4b)}\\
\mathbf{\Pi}^{(1)}_{01} &= 2\left(f_7 - f_1\right) - \rho\left(\frac{u_0}{3} + u_0 u_1\right). && \text{(5.4c)}
\end{aligned}$$

The value of $\mathbf{\Pi}^{(1)}$ on a D3Q19 lattice is:

$$\begin{aligned}
\mathbf{\Pi}^{(1)}_{00} &= f_1 + f_2 + f_3 + f_5 + f_6 + f_7 + f_8 + \\
&\quad 2(f_{10} + f_{12}) - \rho\left(\frac{u_1}{3} + u_0^2 + \frac{1}{3}\right), && \text{(5.5a)}\\
\mathbf{\Pi}^{(1)}_{11} &= 2(f_9 + f_{10} + f_{11} + f_{12} + f_{13}) - \\
&\quad \rho\left(u_1 + u_1^2 + \frac{1}{3}\right), && \text{(5.5b)}\\
\mathbf{\Pi}^{(1)}_{22} &= f_2 + f_4 + f_5 + f_6 + f_7 + f_8 + 2(f_{11} + f_{13}) - \\
&\quad \rho\left(\frac{u_1}{3} + u_2^2 + \frac{1}{3}\right), && \text{(5.5c)}\\
\mathbf{\Pi}^{(1)}_{01} &= 2(f_{10} - f_{12}) - \rho\left(\frac{u_0}{3} + u_1 u_0\right), && \text{(5.5d)}\\
\mathbf{\Pi}^{(1)}_{12} &= 2(f_{11} - f_{13}) - \rho\left(\frac{u_2}{3} + u_1 u_2\right), && \text{(5.5e)}\\
\mathbf{\Pi}^{(1)}_{02} &= f_5 - f_6 + f_7 - f_8 - \rho\, u_0 u_2. && \text{(5.5f)}
\end{aligned}$$

5.2.4 Numerical verification

The quality of the regularized boundary condition is tested on a numerical implementation of the 2D Kovasznay flow described in Section 4.4. Four different types of boundary conditions are implemented for this flow:

Local regularized: The regularized boundary condition, in which the tensor $\mathbf{\Pi}^{(1)}$ is computed from local values on the boundary node.

Non-local regularized: The regularized boundary condition computed from the strain rate $\boldsymbol{S}$, for which a finite difference scheme on neighboring nodes is applied.

Zou-He condition: A commonly used local boundary condition described in Ref. [38].

Inamuro condition: A commonly used local boundary condition described in Ref. [35].

The Reynolds number in all simulations is fixed to a value $Re = 20$, and the grid resolution is progressively refined to observe how the accuracy scales with the discrete time step δ_x. Taking into account the observations of Section 2.4, the time step

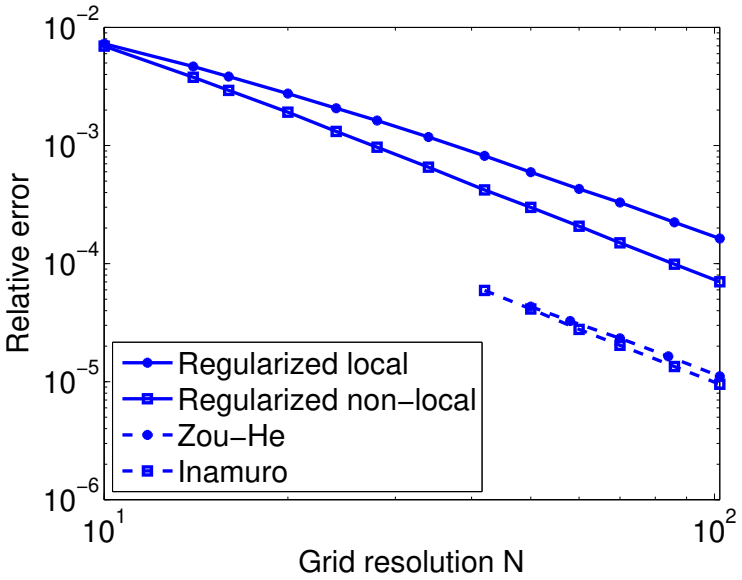


Figure 5.2: Accuracy of a 2D Kovasznay flow, implemented with four different boundary conditions. The curves corresponding to the Zou-He and the Inamuro methods are incomplete, because those methods produce numerical instabilities on coarse grids.

δ_t is rescaled so as to keep constant the ratio $\delta_t/\delta_x^2 = 0.42$. According to Note 2.7 on page 29, this amounts to fixing the relaxation parameter to $\omega = 1.7762$. The results of the benchmark are presented on Figure 5.2 on simulations with a varying grid size. It is good to remember at this point that the value $N = 1/\delta_x$ does not correspond to the resolution of the complete simulated domain, but rather to the resolution of a reference distance, corresponding to the spacing between to points of the grid (the physical grid represented in the Kovasznay flow, not the numerical grid). The simulations are implemented with a BGK dynamics instead of RLB in order to respect the spirit in which the Zou-He and the Inamuro boundary conditions were initially developed. As it is expected from the discussion of the accuracy of LB methods in Section 2.4, the accuracy of all simulations scales at a second order rate with the grid resolution. This confirms on the one hand the choice of how to rescale the time step δ_t properly, and on the other hand the fact that all boundary conditions exhibit an accuracy that is compatible with the accuracy of the LB dynamics. Although the convergence rate is the same, the Zou-He and the Inamuro boundary conditions are somewhat more accurate than the regularized one. This difference can be understood by the fact that the benchmark simulations implement a BGK dynamics, whereas the regularized boundary condition cuts off all higher order modes of the dynamics. This difference between the boundary conditions vanishes when they are applied to a regularized LB simulation. It is furthermore remarked that the Zou-He and Inamuro boundary conditions are substantially less stable than the regularized one. Simulations using those boundary conditions are subject to numerical instabilities, whenever the grid is too coarse or the Reynolds number too high.

The regularized boundary condition is hence found to be an excellent alternative to boundary conditions traditionally used in the LB method. It exhibits the desired second order accuracy, it is numerically more stable and, a point that must be particularly lined out, it is generic and works with all types of lattice structures. The accuracy of the simulation is somewhat better when a non-local version of the regularized boundary is used. The local version delivers however good results too, and it is often preferred for conceptual reasons.

5.3 Initial condition

5.3.1 Introduction

The impact of initial conditions on the evolution of a time-dependent flow can be just as crucial as the choice of an appropriate boundary condition. This issue is illustrated in the present section with the help of an interesting 2D benchmark problem, which is taken from the recent literature and describes the turbulent interaction between two vortexes and a no-slip wall. This problem was originally devised as a benchmark case to test the quality of boundary conditions in CFD software. It is however shown that when using the LB method, the real difficulty stems from an appropriate initialization of the particle distribution functions. It is especially crucial to compute a correct initial value for the pressure and to initialize the equilibrium contribution to the distribution functions properly. The numerical experiment is explained in some detail, and the proposed macroscopic approach to the implementation of the initial condition is compared with an approach described in the literature, based on lattice kinetic considerations. The benchmark application is also used to show the limits of the LB method for incompressible flows due to constraints on the discrete time step.

In fluid turbulence, the interaction between vortexes and no-slip walls is known to have an important influence on the evolution of a flow. Numerous numerical studies have shown that the wall acts like a source for small-scale structures, which on their hand interact with the fluid and affect its overall characteristics. In the numerical experiment used in this section, a complex of two counter rotating vortexes is set up and used as an initial condition for the simulation. The flow is incompressible, and it is confined within a quadratic box with no-slip walls. The rotation of the vortexes induces a resulting forward momentum, and the dipole is self-propelled towards one of the walls. During the rebound that follows, various additional vortexes are generated and detach from the wall. They interact on their turn with the original vortexes and form, among others, systems of secondary dipoles.

5.3.2 The benchmark

The benchmark is based on an incompressible fluid, described by the Navier-Stokes equation in Eq. (1.10) on page 8, together with a continuity condition in Eq.(1.9). The flow is confined in a box of dimension $[-1, 1] \times [-1, 1]$ with no-slip walls. The initial condition is chosen so as to define two counter-rotating monopoles, one with positive core vorticity at the position (x_1, y_1) and one with negative core vorticity

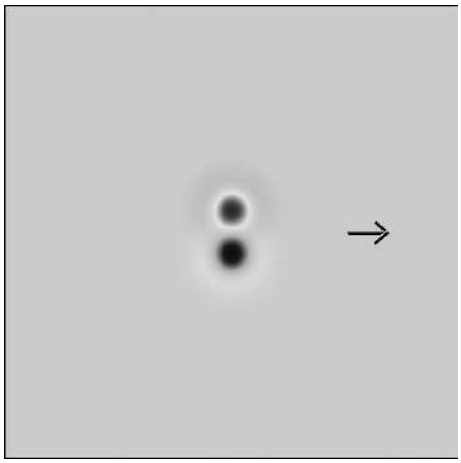


Figure 5.3: Vorticity in the initial configuration. The system is self-propelling and is going to move towards the right wall, as it is suggested by the arrow.

at (x_2, y_2). This is achieved with the following velocity field $\vec{u}_0 = (u_0, v_0)$:

$$u_0 = -\frac{1}{2}|\omega_e|(y-y_1)\exp\left(-(r_1/r_0)^2\right) + \frac{1}{2}|\omega_e|(y-y_2)\exp\left(-(r_2/r_0)^2\right), \quad (5.6)$$

$$v_0 = +\frac{1}{2}|\omega_e|(x-x_1)\exp\left(-(r_1/r_0)^2\right) - \frac{1}{2}|\omega_e|(x-x_2)\exp\left(-(r_2/r_0)^2\right), \quad (5.7)$$

where $r_i = \sqrt{(x-x_i)^2 + (y-y_i)^2}$, the parameter r_0 labels the width of a monopole and ω_e its core vorticity.

The average kinetic energy of this system at a given time is defined by the expression

$$\bar{E}(t) = \frac{1}{2}\int_{-1}^{1}\int_{-1}^{1}|\vec{u}|^2(\vec{x},t)d^2x, \quad (5.8)$$

and the average enstrophy by

$$\bar{\Omega}(t) = \frac{1}{2}\int_{-1}^{1}\int_{-1}^{1}\omega^2(\vec{x},t)d^2x, \quad (5.9)$$

where $\omega = \partial_x v - \partial_y u$ is the flow vorticity.

In all presented simulations, the initial kinetic energy is fixed to $\bar{E} = 2$, by requiring $\omega_e = 299.5286$. Furthermore, the Reynolds number and the monopole radius are set to $Re = 625$ and $r_0 = 0.1$. The monopoles are aligned symmetrically with the box, in such a way that the dipole-wall collision is frontal and takes place in the middle of a wall. The position of the monopole centers is $(x_1, y_1) = (0, 0.1)$ and $(x_2, y_2) = (0, -0.1)$.

The enstrophy field for the initial configuration is shown on Fig. 5.3. The upper monopole has a positive, and the lower one a negative core vorticity.

The flow is simulated by means of the RLB method, implemented on a D2Q9 lattice.

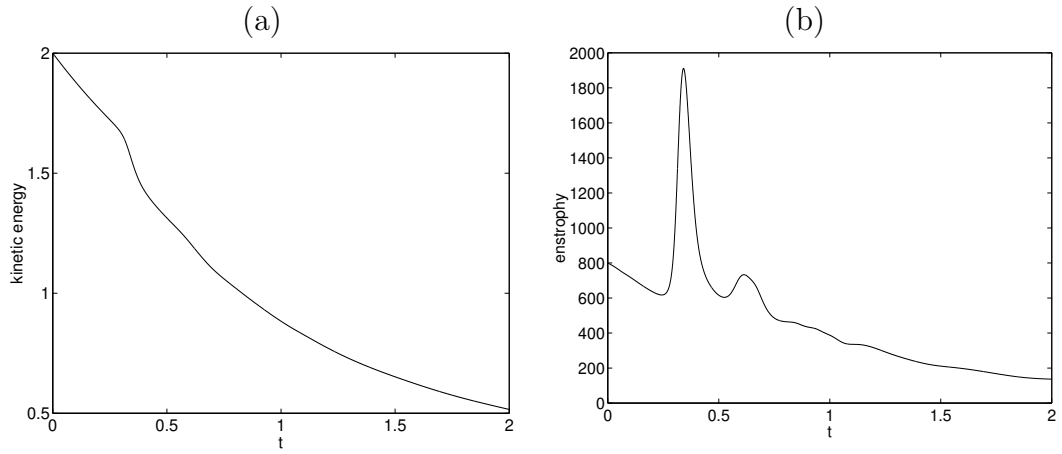


Figure 5.4: Time evolution of the kinetic energy (a) and the enstrophy (b)

5.4 Numerical results

5.4.1 Time evolution of the flow

Figure 5.4 displays the time evolution of the energy and the enstrophy of the system, with the numerical parameters defined in the previous section. The first rebound of the dipole with the wall happens approximately at a time $t = 0.37$. This event is represented by a peak in the enstrophy curve, due to the generation of numerous small-scale vortexes. At the same time, the slope of the energy curve becomes steeper, as the energy dissipation rate is directly dependent on the enstrophy. Two smaller peaks in the enstrophy curve indicate successive rebounds of the dipole with the wall. The value of those maxima, and the time at which they occur, are used as benchmark values to test the accuracy of the simulation. A chosen number of results have been simulated by a finite difference and a spectral method and reported in Ref. [40].

In order to illustrate the physical processes occurring during the rebound of the dipole on the wall, Figure 5.5 shows a contour plot of the vorticity at four chosen time steps. At $t = 0.30$, the lower monopole is observed to be approaching the wall. At $t = 0.34$, small scale structures are generated in the boundary layer close to the wall. At $t = 0.38$, a vortex with positive core vorticity detaches from the wall and creates a secondary dipole with the monopole that initially collided with the wall. At $t = 0.38$, this secondary dipole turns over and prepares for a second collision with the wall.

5.4.2 Benchmark values

In order to complete the discussion of this benchmark problem, the obtained numerical results are shortly presented. The chosen approach for the implementation of the LB simulation is summarized in the next two sections.

The choice of a sufficiently small grid spacing depends on the width of the boundary layer, as the smallest relevant structures are located within this layer. The boundary layer scales like the inverse square root of the Reynolds number, and so does consequently the size of a grid cell. The resolution required to obtain grid

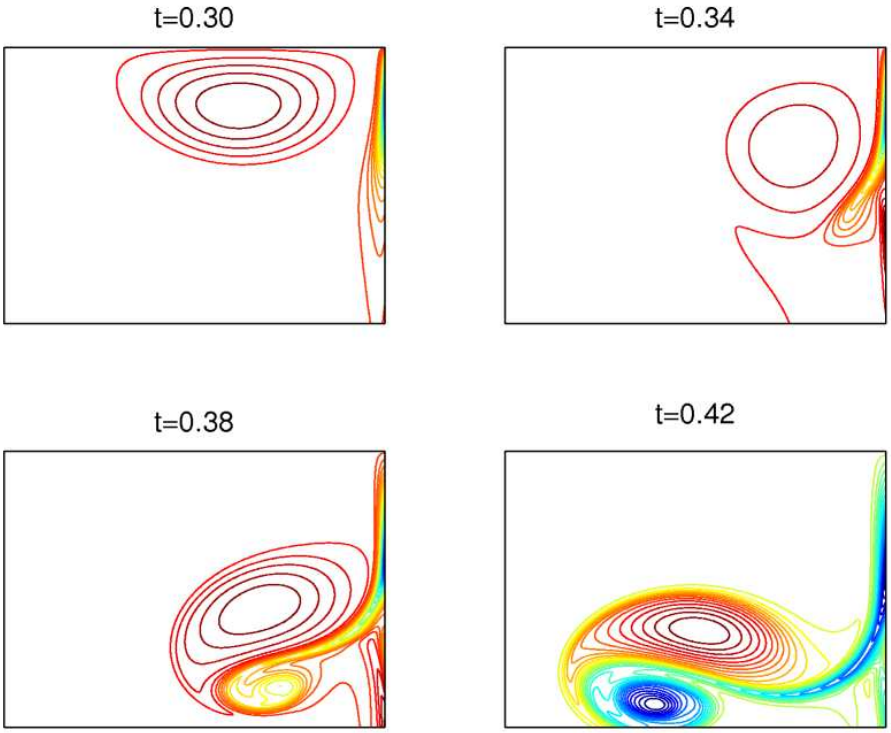


Figure 5.5: Vorticity contours at four chosen time steps for the collision of the lower monopole with the right wall. The shown system is an extract of the full simulated space.

convergence with the LB method is practically identical with the one described in Ref. [40] on a FD scheme. At the given number of $Re = 625$, the system size is for example 700×700. This is surprising, because the LB model is much more straightforward than the FD model, which makes use of a multi-grid technique. Additionally, the LB model is based on a simple bounce-back model for the simulation of the no-slip wall, which contrasts with the sophisticated boundary condition used in Ref. [40]. The following table shows the computed position and value of the first enstrophy peak with the LB, FD and spectral methods:

	Lattice Boltzmann	Finite Difference	Spectral Method
Position (time)	0.371	0.371	0.371
Value (enstrophy)	933.8	932.8	933.6

5.5 Setup of the initial condition

The initial value for the velocity field is described by Eqs. (5.6) and (5.7). It is not straightforward to set up such an initial condition in a LB simulation. This is because the simulated degrees of freedom in this method are not the macroscopic variables p and $\vec{u}$, but rather the so-called particle distribution functions f_i, with $i = 0..8$. The particle distribution functions can however be related to the macroscopic fields and their space derivatives throught the formula of the regularized LB method, Eq. 4.10:

$$f_i = f_i^{(0)}(p, u_\alpha) + f_i^{(1)}(p, \partial_\alpha u_\beta). \tag{5.10}$$

Given that the flow is considered in its incompressible limit, the pressure p and the density ρ are identified through the ideal gas law, Eq. (1.8) on page 7.

Two approaches to setting up an appropriate initial condition are listed in the following. The first approach has been cited in the literature [41] and shares the philosophy of LB methods. It proposes to obtain the initial condition iteratively by implementing a common LB dynamics. During those iterations, the local fluid velocity $\vec{u}$ is however not recomputed on ground of the simulated particle distribution functions, but it is rather kept at the value one wants it to be in the initial condition. In this way, only the pressure p and the first-order contributions $f^{(1)}$ to the particle distribution functions are free variables. They correspondingly converge to their appropriate value for the initial condition.

The second approach, inspired from a macroscopic approach to fluid dynamics, consists in computing the pressure p and the strain rate $\boldsymbol{S}$ directly. The velocity gradients are *e.g.* computed using a finite difference scheme, and the pressure by solving an iterative Gauss-Seidel scheme for the Poisson equation, Eq. (1.11).

Both approaches to setting up the initial condition require an iterative algorithm to be implemented. Their efficiency is compared in Fig. 5.6 (a). In this comparison, the "LB approach" was implemented in terms of BGK iterations, and the "macroscopic approach" in terms of a successive over-relaxation (SOR) scheme. It is obvious that the SOR scheme converges at a much faster rate, with a gain in efficiency that implies several orders of magnitude. For the sake of completeness, it must be pointed out that the scheme suggested in the Ref. [41] is based on a MRT approach to LB, in which some relaxation parameters are fine-tuned so as to speed up the convergence. It is nevertheless clear that a dedicated method to solving the

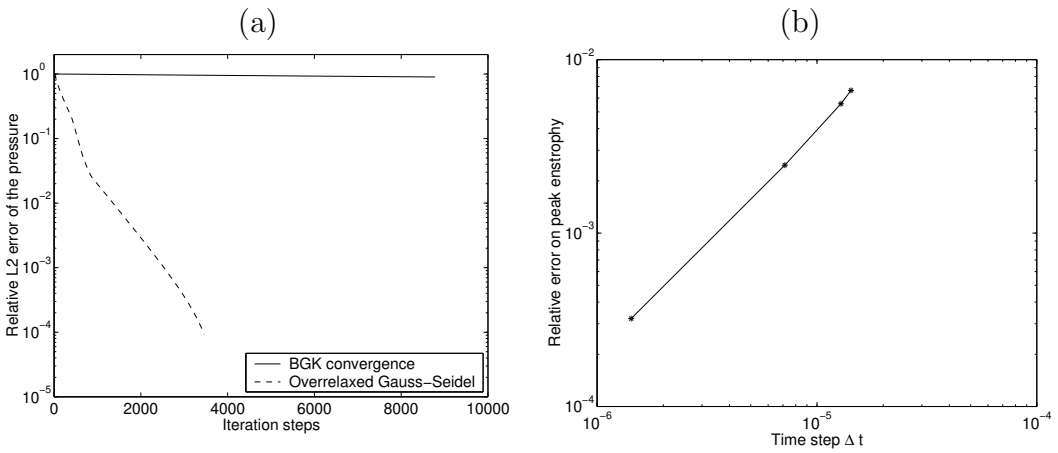


Figure 5.6: (a) Comparison of two iterative schemes for the setup of the initial condition: BGK iterations and SOR scheme (log-log plot). The rate of convergence of the SOR scheme exceeds the one of the BGK approach by several orders of magnitude. (b) Relative error on the value of the enstrophy at the first enstrophy peak. The simulated values are compared with the numerical reference data obtained with a spectral method, and plotted for various choices of the discrete time step.

pressure equation is qualitatively and quantitatively faster than the resolution of a lattice-kinetic scheme.

5.6 Discussion: choice of the time step

It is argued in Section 2.4 that the time step δ_t of a LB simulation cannot be chosen freely because it is coupled to the Mach number. This is a serious drawback of the LB method, whose influence is clearly visible in the present benchmark application. In order for the LB fluid to mimic an incompressible fluid, the Mach number, and consequently the time step, must be chosen very small. The actual time step required in the LB method is typically one order of magnitude smaller than the one used in the FD method.

Figure 5.6 (b) shows how the error on the position of the first enstrophy-peak decreases as a function of the discrete time step. To achieve the accuracy of three significant digits, a time step as small as $\delta t = 10^{-6}$ must be chosen. For comparison, it can be noted that the time step adopted in Ref. [40] in a FD approach takes the value $\delta t = 6.25 \cdot 10^{-5}$. One is therefore tempted to conclude that the error plotted in Fig. 5.6 (b) is dominated by contributions due to fluid compressibility.

Chapter 6

Adaptive space and time steps

6.1 Introduction

In the LB method, the discrete space and time steps δ_x and δ_t are constants that can vary from one simulation to another but are fixed within a simulation. The grid spacing is for example the same in all space directions, and it does not vary from one position of the discrete space to another. The same can be said for the time step δ_t, which does not change during the time evolution of the system. Those assumption were implicitly taken for granted in the theoretical analysis of the LB method in Chapter 2, and they must hold in order for the simulation to asymptotically solve the associated PDE. Some workarounds to this have been proposed in the literature, where a mapping between a regular and an irregular mesh is obtained via interpolation and extrapolation schemes. These techniques will however not be considered further, as they represent hybrid constructs rather than pure LB models according to the line of thought of this thesis.

Other types of CFD methods are not subject to such restrictions on the regularity of the time and space discretization. Finite difference methods, presented *e.g.* in Ref. [4], are principally executed on regular grids with fixed spacing as well. The grid can however be anisotropic, that is, the value of the grid spacing can differ between from a space direction to another. Furthermore, the discrete time step can change during the evolution of the system. Finite volume methods [5] and finite element methods [6, 5] are even more general and can be implemented on unstructured grids: the grid points can be situated on arbitrary space positions, and it is not necessarily possible to represent the relative position of grid points to each other by a matrix data structure.

The use of inhomogeneous grids makes sense when a problem with an inhomogeneous geometry is being simulated. Some parts of the simulated domain require a higher grid resolution than others in order to reach the required level of accuracy. The grid resolution needs for example to be increased close to an obstacle with complicated shape to ensure that the discretized version of the obstacle resembles the original one sufficiently well. In other cases, the resolution needs to be increased because the fluid flow exhibits small-scale patterns, such as the small vortexes generated close to the wall in the numerical experiment of Section 5.3. In simulations with fixed-sized grids, the overall resolution needs to be adapted to the maximal required value, by which much computational effort is wasted. It should be pointed

out that approaches to calculate a local *a priori* estimate of the numerical error exist in many numerical methods, as those described in Ref. [6]. With this estimate, the grid can be dynamically adapted during a simulation to reach the desired trade-off between accuracy and efficiency. The LB approach to CFD has however been developed quite recently, and no formal framework exists currently that would lead to such an error estimate in LB simulations.

An adaptive time interval is likewise useful to adjust the time resolution, depending on how rapidly the flow structures vary in the simulation. In some methods such as the one presented in Ref. [4], the time step must furthermore be chosen on ground of well known stability criteria. Again, experimental evidence shows that LB simulations are also numerically unstable when the discrete time step is too large, but a theoretical framework is lacking from which exact stability criteria could be deduced.

In order to overcome the limitation of a fixed grid spacing in the LB method and to adapt the simulation at least roughly to the geometry of a problem, a so-called *multi block technique* is often used. The idea is to partition the domain of interest into rectangular subdomains, for each of which a different grid is used with its own parameters δ_x and δ_t. It is obviously necessary to transfer data from one of those grids to another during the execution of a simulation. Two difficulties need to be overcome in order to do so. First, the data on the interface between two grids needs to be interpolated, because the different grids overlap only partially, as well in space as in time. Second, the data needs to be rescaled to account for the changing value of δ_x and δ_t. As a matter of fact, it was shown in Section 2.4 that the LB variables are not dimensionless with respect to a macroscopic system of units and thus depend on the particular value of the discretization parameters. When the RLB model is used, it is however easy to rescale the variables of interest. They can be reduced to a set of variables ρ, $\vec{u}$ and $\mathbf{\Pi}^{(1)}$ which are cast into a dimensionless representation according to Eq. (4.11) on page 46. An application of this approach to multi block LB modeling is presented in Section 6.2. The numerical example presented in this section illustrates how one can compute the drag force acting on an obstacle immersed in a fluid with infinite extent. A system of successively refined, nested grids is used to account for the need of a high resolution close to the obstacle. By combining this technique with a novel approach for the implementation of open boundaries, which has recently been described in the literature, benchmark results of an exceptionally good quality are obtained.

In Section 6.3, the RLB model with adaptive time stepping is introduced. It is used to simulate the values of an incompressible stationary flow. A large time step is chosen initially, so as to rapidly converge to the stationary state, and a smaller time step in a later stage, so as to decrease the numerical error due to the compressible nature of the fluid model.

6.2 Grid refinement

6.2.1 Introduction

The multi block grid refinement technique of the regularized LB model discussed in the previous section is now tested on a numerically challenging task: simulating a stationary incompressible fluid flow past a rigid obstacle. The fluid is assumed to fill the whole 2D space, the obstacle is placed at the origin of the coordinate system, and the fluid velocity is asymptotically constant far from the obstacle. Thus, the problem consists in the resolution of the stationary Navier-Stokes equation for the velocity field $\vec{u}(\vec{r})$ with boundary condition

$$\lim_{|\vec{r}|\to\infty} \vec{u}(\vec{r}) = \vec{u}_0. \tag{6.1}$$

Apart from the grid refinement procedure, a major difficulty stems from the necessity to truncate the infinite domain for numerical purposes, and to find an artificial boundary condition for the boundaries of the truncated domain. A straightforward approach consists in using the asymptotic condition $\vec{u} = \vec{u}_0$ on the numerical domain boundaries. Although this method is easy to implement, it appears to be quite inappropriate for the needs of numerical modeling, as it requires the use of excessively large domains. Indeed, it will be shown in the present section that the structure of the flow is strongly influenced by the shape of the obstacle even far from the center. Other approaches to this problem use extrapolation schemes on the boundaries so as to ensure a vanishing gradient perpendicular to the boundaries, for the velocity or other physically relevant quantities. The drawback of those approaches is that they are unable to properly tune the asymptotic velocity $\vec{u}_0$ in the fluid and can therefore not be used on all boundaries. Furthermore, they make it difficult to ensure conservation of mass and momentum across domain boundaries.

For those reasons, an alternative technique is introduced here that has been recently described in the literature [42, 43, 44]. In this method, an explicit vector field is proposed that can be used to implement a Dirichlet boundary condition for the fluid velocity in a region reasonably far from the domain center. The expression for this vector field is obtained from a truncated asymptotic expansion of a solution to the stationary Navier-Stokes equation and approximates the structure of the flow with considerably higher precision than the constant approximation $\vec{u} = \vec{u}_0$. The drawback of this method is that it depends on the drag and lift coefficients of the obstacle which are *a priori* unknown. Therefore, the solution process involves a series of iteration steps during which the formula of the boundary condition is updated on ground of the drag coefficients of the obstacle measured at this state of the simulation. A brief overview of the method is found in the next section, Section 6.2.2.

The RLB method, executed on a set of progressively refined grids, is used to solve this problem. The simulations are run at very low Mach numbers in order to approach the nature of an incompressible fluid with sufficient accuracy. The following sections present a case study for the numerical evaluation of a drag coefficient, and serve three main purposes. First, they present an introduction to the boundary condition of Refs. [42, 43, 44] and demonstrate its efficiency and simplicity in the context of LB simulations. Second, it is shown that this method can be coupled

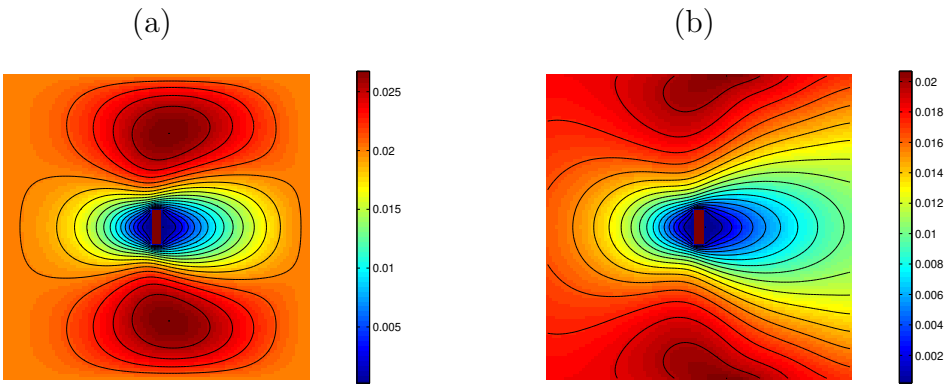


Figure 6.1: (a) Flow profile close to the obstacle when a zeroth-order boundary condition $\vec{u} = \vec{u}_0$ is used. (b) Velocity profile with the first-order boundary condition introduced in this section.

with a numerical technique based on iterative grid refinement. The grid refinement approach of RLB, in which all variables are cast into a dimensionless representation according to Eq. (4.11) is observed to produce excellent results. Finally, it is argued that although the boundary condition of Refs. [42, 43, 44] has been developed for incompressible flows, it also proves useful for simulations of compressible flows at low Mach numbers. For further precision, the influence of the fluid compressibility on the computation of a drag force is analyzed.

6.2.2 Analytical profile on the boundaries

In Ref. [42], the solution to the 2D incompressible Navier-Stokes equation is expanded in a finite series, as a function of formal parameters depending on the drag and lift coefficients of the obstacle. The corresponding theory for the 3D case is presented in Ref. [43]. It is recognized that at a certain distance from the center, the structure of the flow doesn't depend for the specific details of the obstacle geometry, but only on 2 (2D) or 3 (3D) distinct parameters. These considerations result in the prescription of an explicit vector field that can be used as a boundary condition on the numerical domain boundaries. The zeroth- and first-order terms of the expansion for a 2D flow on the velocity field $\vec{u} = (u, v)$ at a position $\vec{r} = (x, y)$ read

$$\begin{aligned} u(x,y) &= u_\infty \left(1 + l\frac{d}{\pi}\frac{x}{x^2+y^2} + l\frac{b}{\pi}\frac{y}{x^2+y^2} - \theta(x)\sqrt{l}\frac{d}{\sqrt{\pi}}\frac{1}{\sqrt{x}}e^{-\frac{y^2}{4lx}} \right), \\ v(x,y) &= u_\infty \left(l\frac{d}{\pi}\frac{y}{x^2+y^2} - l\frac{b}{\pi}\frac{x}{x^2+y^2} - \theta(x)\sqrt{l}\frac{d}{2\sqrt{\pi}}\frac{y}{x^{3/2}}e^{-\frac{y^2}{4lx}} \right) \end{aligned} \tag{6.2}$$

where $d = F_x/(2\,\rho\,l\,u_\infty^2)$ and $b = F_y/(2\,\rho\,l\,u_\infty^2)$ are the drag and the lift coefficient, and $l = \nu/u_\infty$ is the viscous length, dependent on the dynamic fluid viscosity ν. The formula also uses the Heaviside function $\theta(x)$, defined as $\theta(x) = 1$ for $x > 0$ and $\theta(x) = 0$ for $x < 0$. Without loss of generality, the asymptotic velocity $\vec{u}_\infty = (u_\infty, 0)$ has been taken to be parallel to the x-axis. This boundary condition is implicit in

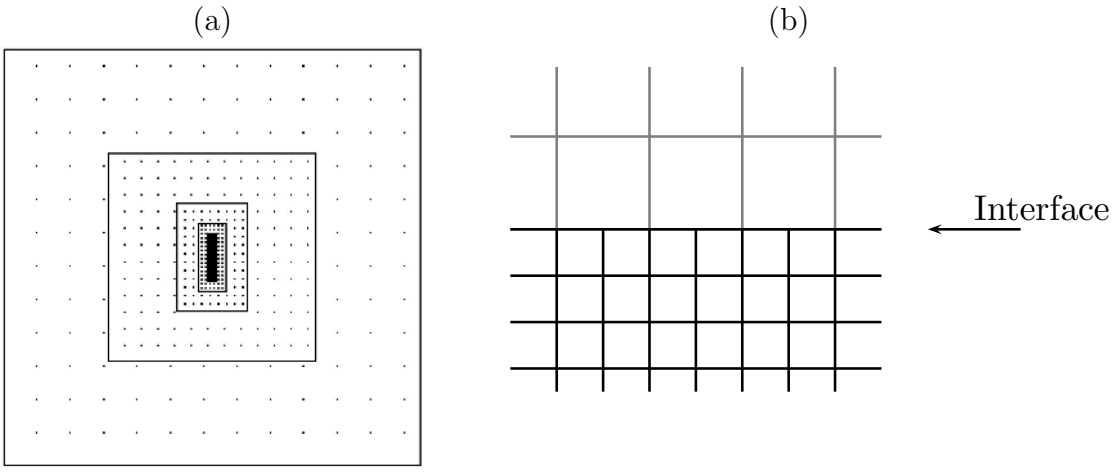


Figure 6.2: (a) Structure of the numerical grid close to a rectangular obstacle. One dot on the figure represents a square of 8×8 grid nodes. (b) Schematic representation of the interface between two adjacent grids. In the actual application, the grids overlap by additionally one node to allow for accurate interpolations.

the sense that it depends on two constants, d and b that are in general unknown before the execution of the simulation.

The computed velocity profile close to the obstacle, depending on whether the zeroth-order boundary condition $\vec{u} = \vec{u}_0$ or the boundary condition defined by Eq. (6.2) is used, is displayed on Fig. 6.1. The zeroth-order boundary condition on Fig. 6.1 (a) produces quite unexpected results, as it imposes an unphysical zero-flux condition through the domain boundaries. The first-order boundary condition on Fig. 6.1 (b) is free from such unphysical effects and produces, at least graphically, the illusion of an infinitely extended domain.

6.2.3 Numerical implementation

The RLB simulation uses a grid with high resolution close to the center, given that in this region, the fluid is subject to sharp pressure and velocity variations. A grid refinement technique is therefore applied with a hierarchy of nested grids that have a successively finer resolution as they approach the system center. This hierarchy of nested grids is schematically represented on Fig. 6.2 (a). At each level of grid refinement, both the space step δ_x and the time step δ_t are divided by two. This choice was taken for a particular reason, because one purpose of this simulation was to analyze the effect of fluid compressibility on the accuracy of the result. For this to be possible, the Mach number, which is proportional to the ratio δ_t/δ_x, needs to be the same from one grid to another. It should be mentioned however that the most efficient choice for the simulation of incompressible fluids consists in keeping the ratio δ_x^2/δ_t constant. This topic is discussed extensively in Section 2.4. During the streaming steps, values are transferred from one grid to another on the common interface. To do this, they are first cast into a dimensionless representation according to Eq. 4.11 in order to be independent of the parameters δ_x and δ_t. Then, the values are interpolated in space (because one grid has twice as many nodes as

the other) and in time (because one grid iterates twice while the other one executes only one iteration). The procedure adopted for those interpolations follows closely the method proposed in Ref. [31]. In this reference however, the particle distribution functions f_i are directly interpolated, while in the present work, the interpolation is applied to the dimensionless macroscopic variables ρ^*, $\vec{u}^*$ and $\mathbf{\Pi}^{(1)*}$. As an additional difference, it is remarked that Ref. [31] makes use of a first order accurate time interpolation scheme. Although the accuracy of the time interpolation is not crucial here, as a stationary flow is being simulated, it is generally recommended to use second order accurate time interpolations for consistency with the accuracy of the LB method discussed in Section 2.4. Numerous other grid refinement techniques have been suggested in the literature that mostly adopt a microscopic point of view to justify their approach to rescaling the particle distribution functions. Some of those techniques can be found in Refs. [45, 46, 47, 48, 49].

The nested grids are deployed progressively in such a way as to speed up the convergence of the simulation towards a stationary state. In a first stage, the simulation is run on a small domain close to the obstacle. Then, at chosen time intervals, the size of the physical domain represented in the simulation is enlarged by implementing a new, coarser grid. This is a convenient way of doing to efficiently converge towards a stationary state. Another approach to reach a rapid convergence would consist in refining the time step δ_t as shown in Section 6.3.

As it has been presented so far, the utilized numerical approach contains two separate iterative processes, one for the implementation of the boundary condition, as explained in Section 6.2.2, and one for the convergence to the stationary state, as explained in the current section. It is observed that both processes take place at comparable time scales, and can therefore be coupled in a simple manner. In the present simulations, the drag and lift coefficients needed for the implementation of the boundary condition are updated each time the grid is enlarged.

The no-slip boundaries of the obstacle are implemented via a so-called bounce-back boundary condition, introduced *e.g.* in Ref. [11]. The value of the force acting on the obstacle is evaluated by computing the momentum which is exchanged on its surface. It is easy to evaluate this momentum exchange on bounce-back boundaries, on which particle distribution functions entering a node from the bulk are simply reflected. The exchanged momentum amounts to twice the value of the momentum carried by the reflected distribution functions.

6.2.4 Simulation results

For illustration purposes, the simulation results of a 2D flow across a rectangular obstacle are presented. The ratio between the width and the height of the obstacle is $5:1$, as it is shown in Fig. 6.2. The simulations are run on quadratic domains of varying size, up to three orders of magnitude larger than the obstacle. The Reynolds number $Re = A/l$, defined with respect to the height A of the obstacle, is fixed at $Re = 1$, and the Mach number at $Ma = 0.02 \cdot \sqrt{3}$. On the domain boundaries, both the constant boundary condition $\vec{u} = \vec{u}_0$, and the boundary condition described in Section 6.2.2 are tested. The measured drag coefficient d is plotted on Fig. 6.3 (a) as a function of the domain size, compared to a reference solution $d = -5.0268$ that was computed on a substantially larger domain. This figure shows that the accuracy

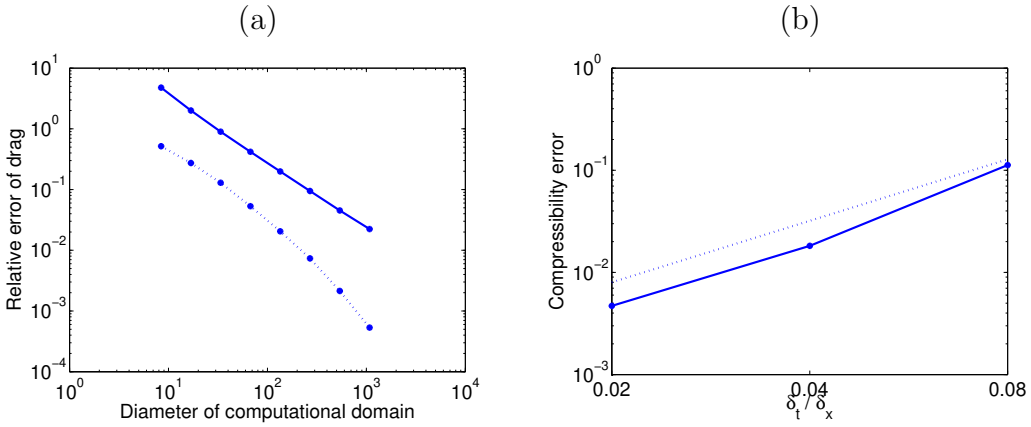


Figure 6.3: (a) Drag coefficient as a function of the system size. The x-axis represents the ratio between the system size and the height of the obstacle. Solid line: asymptotic boundary condition $\vec{u} = \vec{u}_0$. Dotted line: first-order accurate boundary condition described by Eq. (6.2). (b) Solid line: error on the drag coefficient as a function of the Mach number. Dotted line: reference curve, representing a power law with slope -2.

of the drag coefficient converges approximately at the same rate in the presence of either boundary condition. However, using the boundary condition described by Eq. (6.2) gains at least an order of magnitude on the accuracy of the drag coefficient, independently of the system size.

As a conclusion, the combination of a nested grid technique with an appropriate boundary condition results in an efficient numerical scheme. Indeed, the computation of drag forces as those shown on Fig. (6.3) (a) were performed within one day on a common desktop computer. In comparison, the same simulation would take about one month to be executed on a homogeneous, small sized grid. It must also be noted that at the high level of accuracy that is applied here, compressibility effects play an important role. This statement is quantified by a plot on Fig. 6.3 (b), which shows the difference between the drag force exerted by a compressible fluid and a reference solution for an incompressible fluid found in Ref. [44]. As it is expected, this difference decreases at roughly a second order rate with respect to the Mach number. It is however not possible to implement fluids at an arbitrary small Mach number in the LB BGK model used here, as the choice of this parameter is limited by CPU time constraints.

6.3 Adaptive time

6.3.1 Introduction

It is relatively simple to implement adaptive time stepping in the regularized LB method. At a time step t_n, the variables can be cast into their dimensionless representation using the previous time interval $\delta_{t1} = t_n - t_{n-1}$. After this, lattice units are recovered again, based on the new time interval $\delta_{t2} = t_{n+1} - t_n$. For the velocity, one computes for example the dimensionless value $\vec{u}^* = \delta_x/\delta_{t1}\,\vec{u}$ and the rescaled value

$\vec{u}' = \delta_{t2}/\delta_x \vec{u}^*$. Combining those two expressions yields $\vec{u}' = \delta_{t2}/\delta_{t1}\vec{u}$. An adaptive time interval can be used only to simulate incompressible flows, for which the Mach number is phyiscally irrelevant. Indeed, the Mach number of a simulation is related to the lattice parameters through the expression $Ma \sim \delta_t/\delta_x$ and thus varies with the time step.

Compared to other numerical techniques, the LB method is not very sensitive, from the point of view of numerical stability, to the choice of δ_t. Numerically stable simulations are observed at impressively large values of the discrete time step. This parameter is therefore rather chosen as a criteria related to numerical accuracy and the limitation of compressibility effects. In this section, an adaptive time interval is used to find the solution to stationary flow problems as fast as possible. The initial time step is chosen quite large in order to rapidly drive the system from its initial condition to the stationary state. During the evolution of the system, the time step is however progressively decreased so as to obtain accurate results. A similar approach to accelerating LB simulations has been previously described in the literature under the name of *Mach number annealing*. This technique is for example described in Ref. [50]. Compared to this method, the novelty of the approach described in the following sections consists in a proper rescaling of the particle distribution functions at each modification of the time step. This is obtained through a regularization procedure explained in Section 4 and ensures an accurate representation of the time-dependent dynamics.

6.3.2 Numerical experiment

The stationary flow in a lid-driven cavity introduced in Section 4.4 is used again as a benchmark application to verify the efficiency of an approach based on an adaptive time interval. Again, the accuracy of the numerical simulations is evaluated through a comparison with high resolution reference results reported in the literature in Ref. [37]. For this, the value of the velocity field is compared with the reference value at some chosen space positions along a horizontal and a vertical line passing through the cavity center, as it is shown on Fig. 4.3 on page 51.

Note 6.1: Stationary flows and LB *In the present section, as well as in Section 5.2, incompressible stationary flow problems are considered. Those problems are described by a time-independent version of the Navier-Stokes equation (1.10) on page 8, in which the term containing a time-derivative is simply skipped. To solve such a problem with the LB method, a time-dependent system is simulated which progressively approaches a steady state. When the macroscopic variables of the system do not change any more from one time step to the next, the system is considered to have reached the time-independent solution to the Navier-Stokes equation. This is usually not an efficient way of doing. Other iterative procedures exist that find solutions to stationary flow problems substantially faster. In those methods, the intermediate steps of the iteration do not necessarily represent a physically meaningful flow, by which shortcuts can be taken to reach the desired final result. This limitation of the LB method can be partially overcome with*

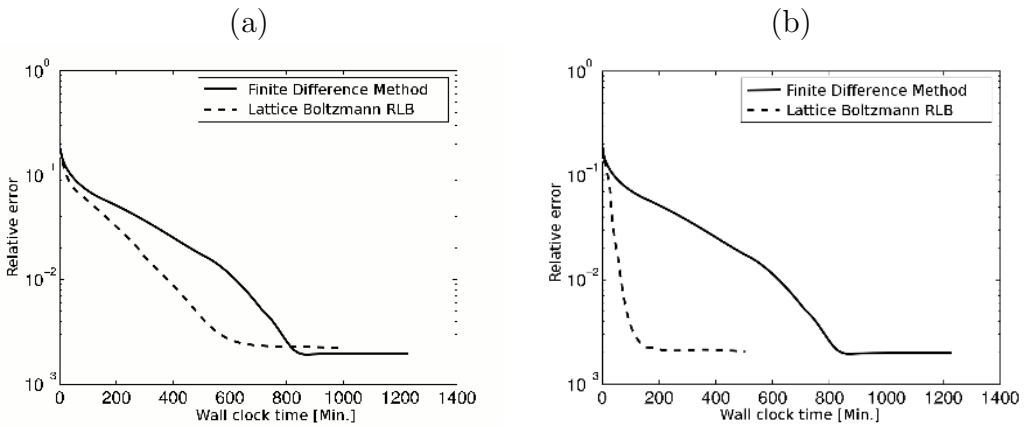


Figure 6.4: Convergence of a 2D lid-driven cavity flow towards the steady state, with a lattice Boltzmann RLB method and a finite difference method. (a) Fixed time step $\delta_t = 1/2\,\delta_x^2$ in FD, fixed time step $\delta_t = 5/2\,\delta_x^2$ in RLB. (b) Fixed time step $\delta_t = 1/2\,\delta_x^2$ in FD; variable time step from $\delta_t = 25\,\delta_x^2$ to $\delta_t = 5/2\,\delta_x^2$ in RLB.

the help of the techniques described in this thesis. In Section 5.2, the problem is first solved on a domain with limited extent, surrounding the region of interest. This gives a first rough approximation to the solution, which in the following is improved by progressively enhancing the extent of the simulation. In the present section, a first rough approximation is obtained via a simulation with a large time step, and the accuracy is increased by a progressive decrease of the time step.

In order to appreciate the efficiency of the LB model, the flow problem is also solved with the help of a finite difference model which is taken from Ref. [4] and described in more detail in Chapter 7. The FD solver has similar properties as the LB model, as it also makes use of a fixed-width regular grid. With both the LB and the FD model, the accuracy of the solution is monitored during the evolution of the simulation. Given that the emphasis is put on rapidly obtaining a stationary solution, rather than analyzing the time evolution of the flow, the real wall-clock time instead of the simulated physical time is measured. Figure 6.4 displays results of a simulation run on a common desktop computer, with a Reynolds number of $Re = 100$ and on a grid of size 129×129. When the time step δ_t is fixed, as on Fig. 6.4 (a), both simulations converge at a comparable speed. The time step of the FD method is fixed by stability criteria, whereas in the LB method, it is chosen so as to obtain results of equivalent accuracy in both methods. On Fig. 6.4 (b), the RLB simulation makes use of an adaptive time interval, starting out at a value of $\delta_t = 25\,\delta_x^2$. This means that in the beginning of the simulation, the velocity of the lid adopts a value of $u_0 = \delta_t/\delta_x \approx 0.2$. In the following, δ_t is decreased at each iteration step, until the final value $\delta_t = 5/2\,\delta_x^2$ is reached. With this approach, the convergence is accelerated by roughly an order of magnitude. Those comparisons must of course be appreciated with some care, because the speed of a simulation depends among others on the quality of its numerical implementation. One can

however consider the FD and LB implementations used here to be of equivalent quality, as they were both written by the same programmer in the *C* language, and comparable efforts were made to optimize the execution speed of either code. As a conclusion, it can be said that the speed up obtained by applying an adaptive time interval is considerable, and this technique is obviously of high interest in the practical work with the LB method.

Chapter 7

Coupling with other tools of CFD

7.1 Introduction

The topic of computational fluid dynamics is introduced in Section 1.2. The overview presented there shows that many different models for the numerical analysis of fluid flows exist. On the one hand there are solvers based on discrete representations of the Navier-Stokes equation, most notably finite difference, finite element or finite volume methods. On the other hand, a new category of solvers has emerged over the past decades which describe the fluid at a molecular level, such as Cellular Automata, or at an intermediate level, such as LB methods.

In this section, the possibility of solving separate spatial regions of a simulation with a different solver is investigated. In this approach, a finite difference (FD) solver is coupled with a LB method. The motivation for developing such a type of coupling is that, depending on the geometry of the flow, one technique can be more efficient, less memory consuming, or physically more appropriate than the other in some regions, whereas the converse is true for other parts of the same system. One can also imagine that a given system solved, say by FD, can be augmented in some spatial regions with a new physical process that is better treated by a LB model. With the approach shown here, only the concerned region is modified, without altering the rest of the computation.

In order to couple a LB with a FD model, it is crucial to understand how the LB set of variables is related to the FD set and vice versa. As has often been the case in previous chapters, the transformation from LB variables to macroscopic variables is easy to perform, whereas the reverse transformation from macroscopic variables to the particle distribution functions requires special attention. Also, as many times before, it seems most natural to solve this problem via the regularized LB method, in which the state of the LB system is expressed in terms of macroscopic variables and their derivatives. In order to obtain the desired coupling, it only remains to find appropriate interpolations for the concerned variables on the interface between a LB and a FD region.

The procedure is of course also applicable when the BGK method is used. In that case, the formula of the regularized LB method, Eq. (4.10) on page 46, is used to convert from macroscopic variables to the particle distribution functions, after which the BGK collision step is executed. The numerical application presented in Section 7.6 makes actually use of the BGK method. At the time this simulation was

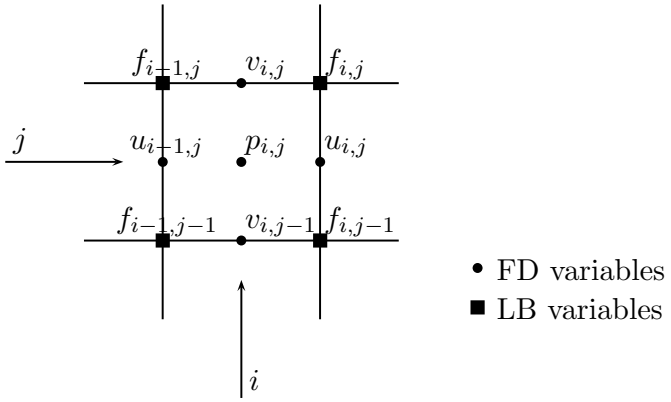


Figure 7.1: Choice of indexes for FD and LB variables on a chosen grid cell (i, j).

prepared, the benefits of using the regularized LB on the full domain was not yet understood, and this method was used only on boundary nodes.

7.2 FD model and computational grids

The FD model used in the present section is taken from Ref. [4]. Its scheme is explicit for the velocities and implicit for the pressure. In order to be sufficiently stable, the method makes use of a staggered grid, that is, a grid on which all macroscopic variables are defined on different space locations. The relative arrangement of those variables, as well as their space position with respect to the LB set of variables, is discussed in the following.

The full computational domain is considered a two-dimensional, rectangular region

$$\Omega = [0, l] \times [0, h] \in \mathbb{R}^2,$$

on which a regular grid is defined. This grid is divided into $i_{\max}$ cells of equal size in the x-direction and $j_{\max}$ cells in the y-direction, resulting in grid lines spaced at a distance

$$\delta_x = l/i_{\max} \quad \text{and} \quad \delta_y = h/j_{\max}.$$

Although the FD model is potentially anisotropic and can handle cases in which δ_x is different from δ_y, this is not true for the LB method. Therefore, those two parameters are always taken equal in the following and are simply labelled δ_x.

The FD model is based on three quantities that are defined at each cell position: the pressure (p), the x-component (u) and the y-component (v) of the velocity. They are however placed on a staggered grid. A given index (i, j) of the cell is assigned to the pressure at the cell center, to the x-component of the velocity at the right edge and the y-component at the upper edge (*cf.* Figure 7.1). The reason for this staggered arrangement is that it prevents possible pressure oscillations which could occur had all three variables u, v and p be evaluated at the same grid points.

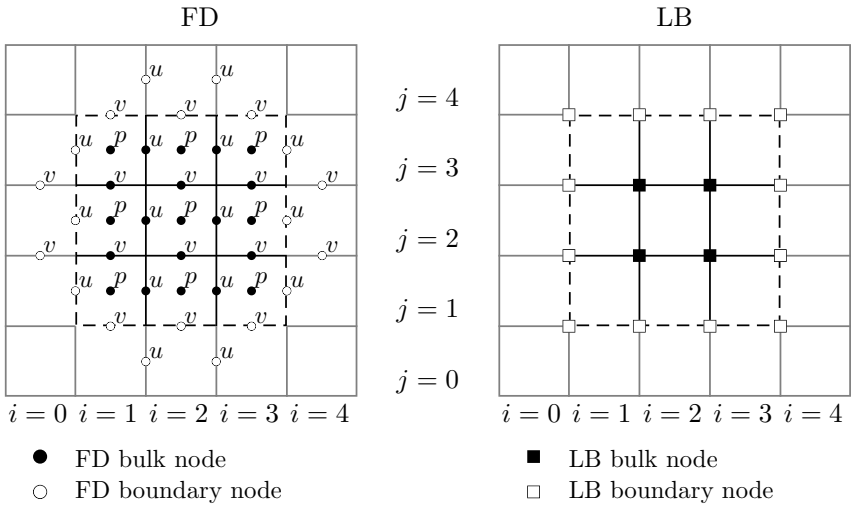


Figure 7.2: Computational grid representing a domain Ω of size $(i_{\max} \cdot \delta_x) \times (j_{\max} \cdot \delta_x)$ with $i_{\max} = j_{\max} = 3$. The left hand side depicts the staggered arrangement of the variables over the grid when the domain is resolved by a FD scheme. In the case of a LB solver, all variables are located on cell edges, as shown on the right hand side. The location of the boundary strip is indicated by a dashed line.

The LB model uses 9 variables $f_k, k = 0 \cdots 8$ which are all evaluated at the same location of a cell. In the present coupling model, the LB variables are situated in the upper right corner. This choice is of course arbitrary. It is however reasonable to situate the f's on cell corners, to ensure that the LB and the FD model have a compatible interpretation of the location of the domain boundary $\partial\Omega$. Indeed, this boundary is defined on a cell edge in the FD model. Considering that most implementations of LB boundary conditions set the domain boundary on top of a LB node, this leads to placing the LB node on the intersection of two cell edges.

The situation is depicted on Figure 7.2 for a system of extent $i_{\max} = j_{\max} = 3$. It shows also that as a result of the staggered arrangement, not all extremal grid points of the FD set of variables come to lie on the domain boundary. For this reason, an extra boundary strip of grid cells is introduced, so that the boundary conditions may be found by linear interpolations between the nearest grid points on either side.

The FD model is based on a discretization of the incompressible Navier-Stokes equation (1.10) and the continuity equation (1.9). The computation of successive iterations $\left(u^{(t)}, v^{(t)}, p^{(t)}\right) \Rightarrow \left(u^{(t+1)}, v^{(t+1)}, p^{(t+1)}\right)$ contains two distinct steps:

1. Resolution of the Poisson equation (Eq. (1.11) on page 8) to obtain the new pressure field. This computation utilizes the values of the pressure and the velocity at the time t: $\left(u^{(t)}, v^{(t)}, p^{(t)}\right) \Rightarrow \left(p^{(t+1)}\right)$. In presence of Dirichlet boundary conditions, this procedure has a unique solution (except for an integration constant). In particular, there is no need to know the value of the pressure on the boundary.

2. Computation of the new velocity field according to a finite difference scheme. It uses the pressure field at step $t+1$: $\left(u^{(t)}, v^{(t)}, p^{(t+1)}\right) \Rightarrow \left(u^{(t+1)}, v^{(t+1)}\right)$.

7.3 Choice of units

The LB method as presented so far makes use of a system of lattice units, in which the distance between two adjacent grid points, as well as the time interval between two successive iteration steps, is unitary. The FD method on the other hand uses the dimensionless system introduced in Section 1.2.3, in which the simulated variables are independent of δ_x and δ_t. In order to easily formulate the coupling between the two methods, the variables of the LB method are also cast into dimensionless units. The spacing between adjacent grid points thus takes the value δ_x, and the interval between successive iterations the value δ_t. The lattice constants $\vec{c}_i$ are replaced by the lattice velocities $\vec{v}_i = \delta_x/\delta_t \vec{c}_i$, and the LB dynamics in Eq. (1.14) on page 10 is reformulated as follows:

$$f_i(\vec{r} + \delta_t\, \vec{v}_i, t + \delta_t) - f_i(\vec{r}, t) = \Omega_i. \tag{7.1}$$

Whenever some data is transferred from one grid to another, the dimensionless form ρ, $\vec{u}^*$ and $\mathbf{\Pi}^{(1)*}$ of the variables in the RLB model is used. The stars $*$ are however omitted to simplify the notation.

7.4 The FD-LB interface

The full domain Ω is now split into two subdomains Ω_1 and Ω_2 such that $\Omega = \Omega_1 \bigcup \Omega_2$. In domain Ω_1 the FD method is active and in Ω_2 the LB method is used. For simplicity, the subdomains are taken to be rectangular, Ω_1 occupying the left hand side and Ω_2 the right hand side of the domain. This procedure can however be extended with few changes to a general boundary.

Figure 7.2 displays the position of extremal grid points, drawn as white circles and squares, on which a boundary condition needs to be implemented to achieve a coupling between the two grids. On the interface between Ω_1 and Ω_2, those boundary values are taken from the boundaries of the opposite domain. As a consequence of the staggered arrangement of the LB values with respect to the FD values, there is need for an overlap between Ω_1 and Ω_2. There are several ways the coupling can be implemented. In the approach chosen here, the two grids overlap by roughly one lattice site. The position of this site is indexed by $i = i_{\text{int}}$ (see Figure 7.3).

As a conclusion, the FD domain requires knowledge of the values of $u_{i,j}$ for $i = i_{\text{int}}$ and $j = 1 \cdots j_{\max}$, and the values of $v_{i,j}$ for $i = i_{\text{int}} + 1$ and $j = 0 \cdots j_{\max}$. Those values are easily obtained from the LB field, on which the macroscopic variables are computed by means of Eqs. (1.15) and (1.16) on page 10.

The LB domain on the other hand requires the knowledge of the 9 values $f_{k;i,j}$ at $i = i_{\text{int}} - 1$ and $j = 0 \cdots j_{\max}$. To obtain those, the regularization formula from Eq. (4.10) on page 46 is used, which connects the pressure, the velocity and the the velocity derivatives to the value of the particle distribution functions.

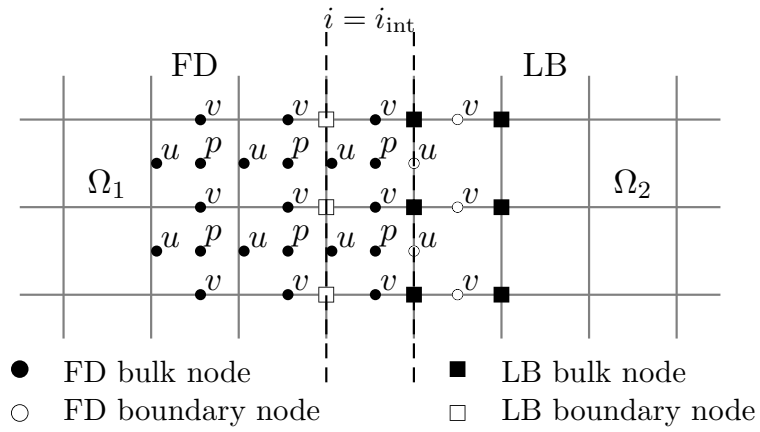


Figure 7.3: Subdivision of the computational domain Ω into a FD subdomain (Ω_1) and a LB subdomain (Ω_2). One lattice cell on the interface between Ω_1 and Ω_2, at the position $i = i_{\text{int}}$, is computed by both methods. The boundary nodes of the subdomains are represented by white symbols. They must be implemented by means of a coupling term between the two methods.

7.5 The coupling algorithm

7.5.1 Coupling the pressure

Before turning to the velocity field, the treatment of the pressure in the FD simulation and the density in the LB simulation is discussed. It is useful to remember that the LB simulation is presently considered in an incompressible regime, in which the density is connected to the pressure via the ideal gas law in Eq. (1.8) on page 7.

The pressure field of an incompressible flow is not uniquely defined. It contains a constant offset which can be chosen arbitrarily. In the FD method used here, the value of the offset depends on how fast the numerical method converges, and it varies from one time step to another. It is therefore difficult to couple the pressure field of the FD and LB simulations together. To do so would require a calibration of the pressure fields by computing the average pressure on each side of the simulation.

It is fortunately possible to consistently compute the value of the pressure on each computational subdomain, without transferring values from Ω_1 to Ω_2. As a matter of fact, the value of the pressure in an incompressible flow can be directly deduced from the value of the velocity field through the Poisson equation (1.11). Practically speaking, on the side of the FD simulation, the FD-LB interface is simply considered as a domain boundary, as far as the pressure is concerned, and the Poisson equation is solved on the subdomain Ω_1. Similarly, the pressure of the LB simulation is computed by applying a usual boundary condition on the FD-LB interface as it is described in Section 5.2.2.

7.5.2 Coupling the velocity

Obtaining values for the velocity in the FD field from the LB field is a simple exercise. The two components of the velocity are computed in a straightforward

manner from Eqs. (1.15) and (1.16) on page 10. Because the FD and the LB variables are not defined at the same point of a lattice cell, we need to adjust the values by an interpolation which, for consistency with the accuracy of the overall numerical scheme, is second order accurate. This leads to the following relations:

$$u^{\mathrm{FD}}_{i_{\mathrm{int}},j} = 1/2\left(u^{\mathrm{LB}}_{i_{\mathrm{int}},j} + u^{\mathrm{LB}}_{i_{\mathrm{int}},j-1}\right) \tag{7.2}$$

$$v^{\mathrm{FD}}_{i_{\mathrm{int}}+1,j} = 1/2\left(v^{\mathrm{LB}}_{i_{\mathrm{int}}+1,j} + v^{\mathrm{LB}}_{i_{\mathrm{int}},j}\right) \tag{7.3}$$

A similar procedure is used to translate the values of the velocity from the FD field to the LB field:

$$u^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} = 1/2(u^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j} + u^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j+1}) \tag{7.4}$$

$$v^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} = 1/2(v^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j} + v^{\mathrm{FD}}_{i_{\mathrm{int}}i,j}) \tag{7.5}$$

$$\tag{7.6}$$

7.5.3 Coupling the velocity gradients

The regularization formula (4.9) refers to the tensor $\boldsymbol{S}$ which, in Eq. (1.6) on page 7, is defined in terms of velocity gradients. Figure 7.2 shows that two of those gradients can be approximated by a centered difference of half the mesh width:

$$\partial_y u^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} = 1/\delta_x(u^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j+1} - u^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j}) \tag{7.7}$$

$$\partial_x v^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} = 1/\delta_x(v^{\mathrm{FD}}_{i_{\mathrm{int}},j} - v^{\mathrm{FD}}_{i_{\mathrm{int}},j-1}) \tag{7.8}$$

One gradient needs to be calculated as a centered difference of mesh width, based on interpolated values of the velocity:

$$\partial_y v^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} = \frac{1}{4\delta_x}(v^{\mathrm{FD}}_{i_{\mathrm{int}},j+1} + v^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j+1} - (v^{\mathrm{FD}}_{i_{\mathrm{int}},j-1} + v^{\mathrm{FD}}_{i_{\mathrm{int}}-1,j-1})). \tag{7.9}$$

There are not enough grid points at hand for computing the fourth gradient at the same level of precision. Luckily enough, this value can be computed with the help of the continuity equation (1.9) on page 7:

$$\partial_x u^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} = -\partial_y v^{\mathrm{LB}}_{i_{\mathrm{int}}-1,j} \tag{7.10}$$

7.5.4 Summary

Now that all missing variables have been computed, it is useful to take a step back and discuss the overall algorithm of a LB iteration step. For the purpose of this discussion, the BGK dynamics is split into two steps. The first, the collision step, handles the computation of the equilibrium distribution and maps the "incoming particle stream" $f_i^{(\mathrm{in})}$ onto the "outgoing particle stream" $f_i^{(\mathrm{out})}$. It is followed by a streaming step that transports the particles by a value of δ_t in time and $\vec{v}_k\delta_t$ in space. The details of an iteration step are as follows:

1. On bulk nodes, $\rho(t)$ and $\vec{u}(t)$ are computed from the incoming particle densities $f_k^{(\mathrm{in})}(t)$. On boundary nodes, all the values $\rho(t)$, $\vec{u}(t)$ and $f_k^{(\mathrm{in})}(t)$ are obtained from the variables of the FD field at time t.

2. All nodes, bulk and boundaries, perform the collision step: $f_k^{(\mathrm{out})}(\vec{r},t) = (1-\omega) f_k^{(\mathrm{in})}(\vec{r},t) + \omega f_k^{(\mathrm{eq})}(\vec{r},t)$.

3. The bulk nodes perform the streaming step: $f_k^{(\mathrm{in})}(\vec{r}+\vec{v}_k\delta_t, t+\delta_t) = f_k^{(\mathrm{out})}(\vec{r},t)$, for all $\vec{r}$ such that $\vec{r}+v_k\delta_t$ lies on a bulk node.

Alternatively, it is possible to extend the streaming step to boundary nodes for those values of $f_k^{(\mathrm{in})}$ that are incoming from the bulk of the LB simulation. They are kept unchanged, unlike the remaining set of $f_k^{(\mathrm{in})}$ and the macroscopic variables ρ and $\vec{u}$ that are provided by the FD field. In the simulations, this procedure appears to produce results equivalent to those of the proposed algorithm.

7.6 Numerical validation

The coupling algorithm is validated by the simulation of a Poiseuille flow. This is a stationary flow in a channel of infinite length with no-slip boundaries. An analytical solution for this flow is shown in Eq. (3.7) on page 33. In lattice units, the boundaries extend horizontally at a height $y = 0$ and $y = L$. The fluid velocity is strictly horizontal and does not depend on the x position: $v = 0; \quad \partial_x u = 0$. In the present example, the fluid is driven by a body force. The left and the right borders implement periodic boundary conditions in order to simulate a channel of infinite length. Special care must be taken when implementing this boundary in the FD model. Indeed, during the simulation it tends to build up a pressure gradient that must be eliminated by imposing a constant value of the pressure on the left and right boundary.

The simulation is performed on a grid of size $i_{\max} = 3$ and $j_{\max} = 50$. The physical channel width is set to $L = 1$ and the body force has the value $F_x = 0.01$. The numerical values u_{sim} are compared to the analytical solution by means of the overall error:

$$\epsilon = \frac{\sqrt{\sum_{j=0}^{j_{\max}} (u_{\mathrm{sim}}(\cdot,j) - u_{\mathrm{ana}}(\cdot,j))^2}}{\sqrt{\sum_{j=0}^{j_{\max}} u_{\mathrm{ana}}(\cdot,j)^2}} \tag{7.11}$$

Three simulations have been conducted that can be compared to each other. The first simulation implements a pure LB scheme with the BGK model, the second simulation a pure FD model, and the third simulation is a FD–LB hybrid. In the first case, the no-slip property of the walls is implemented by a boundary condition known under the name of bounce-back (see *e.g.* [8]). In the third case, the top and bottom strips of size $j'_{\max} = 3$ are computed by the FD model and the bulk domain by the LB model (see Figure 7.4).

A remarkable result of the simulations is that the FD-only model reaches the analytical solution at the machine level of precision (10^{-15}). Although there exist LB boundary conditions which obtain the same result, as for example the one in Ref. ([35]), their implementation is less natural and straightforward than the one of the FD model. It is further remarked that the LB model converges faster (in terms of iteration steps) than the FD model. The stationary velocity field of the LB simulation is however distinct from the analytical prediction, due to the limited

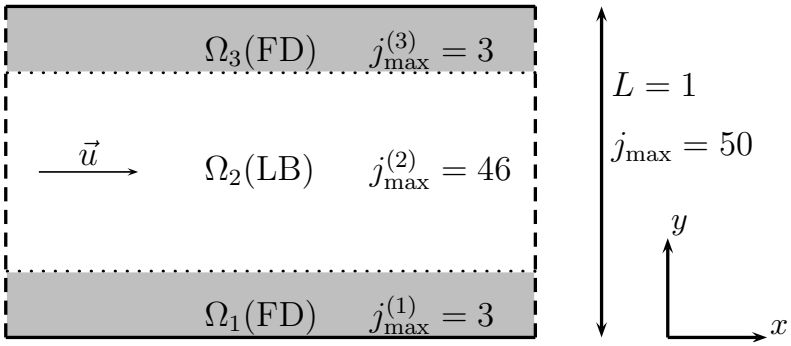


Figure 7.4: The computational grid for the simulation of a Poiseuille flow is partitioned into three subdomains Ω_1, Ω_2 and Ω_3. The FD scheme is used on the boundary domains Ω_1 and Ω_3, and the LB scheme on the bulk domain Ω_2.

precision of the boundary condition. The hybrid simulation shows the expected convergence speed of the LB model and an error due to the limited precision of the coupling. However, the error is two orders of magnitude smaller than the one due to the bounce-back boundaries of the LB-only simulation. The results of the simulations are shown on Figure 7.5.

The order of precision of the coupling can be estimated by varying the grid resolution ($j_{\max}$) while keeping the physical quantities(L, F_x) constant. Figure 7.6 plots the error of the stationary velocity field as a function of the grid resolution. It appears clearly that the coupling acts like a second-order boundary condition for the velocity field. No conclusion can be taken concerning the implementation of the pressure field, because the latter is constant in a Poiseuille flow.

7.7 Conclusion

In this chapter, a LB scheme for 2D incompressible fluid flows is spatially coupled to a FD scheme on a computational domain partitioned in two regions. The methods of the regularized LB scheme are used to relate the LB distribution functions f_i with the classical physical quantities and their derivatives. A specific FD schemes has been introduced, for which a complete coupling algorithm has been explicited. At the interface, the LB and FD are connected so as to preserve continuity of the physical quantities. The connection between the f_i variables and the standard macroscopic physical quantities is obtained through a dimensionless representation of the LB variables via the regularized LB method: the equilibrium part of f_k is related to the macroscopic quantities and the non-equilibrium part to the gradients thereof. A validation was performed by simulating a Poiseuille flow with FD boundary strips and LB bulk and comparing it with an analytical solution. The simulation shows that in this case, the coupling of the velocity field is of second order in the grid resolution.

It would be interesting to push this work further and test the results of the coupling on more interesting geometries. It would also be interesting to extend the

Figure 7.5: Simulation of a body-force driven Poiseuille flow with (1) a LB model, (2) a FD model and (3) a FD-LB hybrid. The curves show the time-evolution of the error, compared to the analytical solution of the Poiseuille flow.

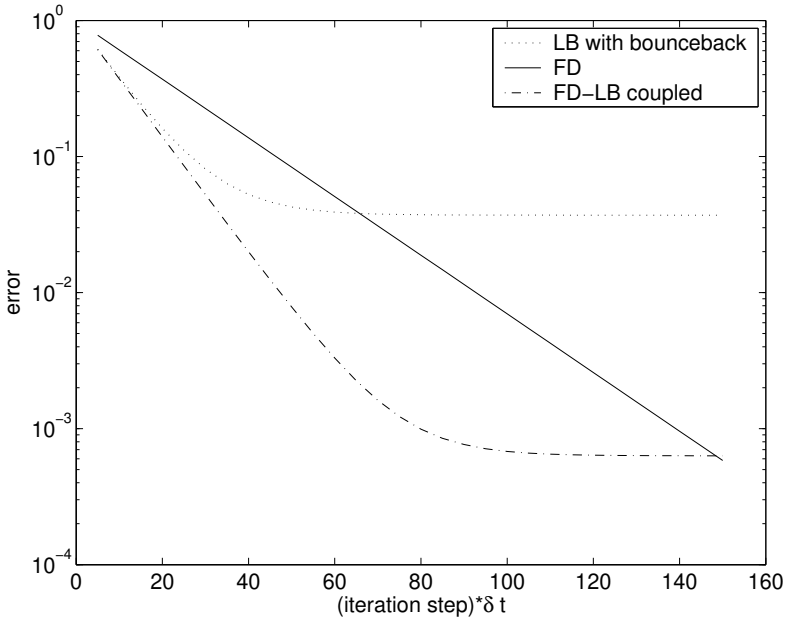
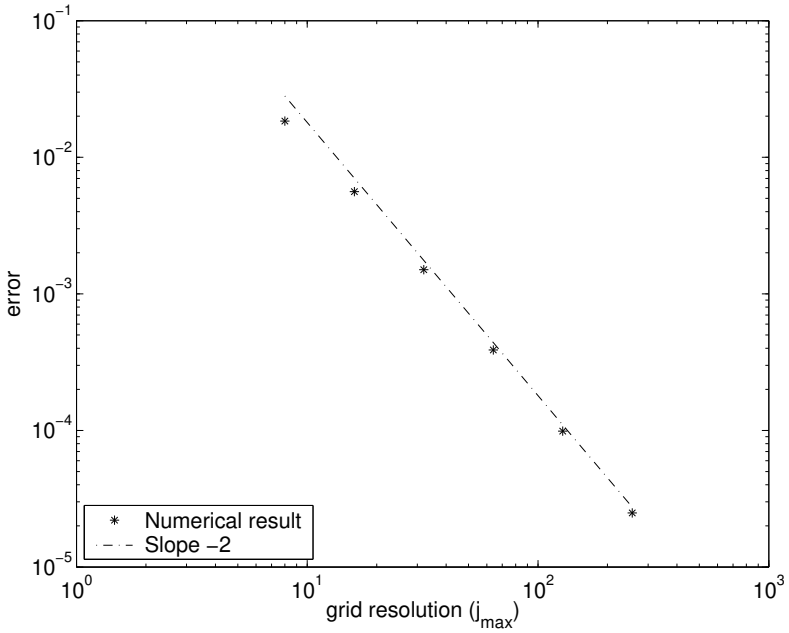


Figure 7.6: Error of a FD-LB hybrid Poiseuille flow simulation as a function of the grid resolution. A log-log plot shows the coupling of the velocity field to be of second order.



work presented here to other methods in CFD, such as the finite element method. One key issue in this procedure would be to relate the values from the irregular grid of typical finite element methods to the regular grid of the LB method. Such questions are at the heart of the `OpenLB` project in Ref. [13], in which this topic is approached from a software engineering point of view. The developed LB code is encapsulated in object-oriented data structures that are very similar to those used in a related finite element package, which encourages the use of those two tools on common problems.

Chapter 8

A model for fluid turbulence based on kinetic variables

8.1 Turbulence modeling

8.1.1 Overview

In all numerical experiments conducted in the previous chapters, the simulated flows have a laminar structure. At all time steps, it is possible to describe the velocity field exhaustively via a restricted number of *streamlines*. Those lines are defined as integral curves to the flow field: on every point of the curve, the flow velocity is tangential to the curve. In a laminar flow, the streamlines vary smoothly in space and time, and they are often used to illustrate the flow visually. In high Reynolds number flows, a *turbulent regime* is observed to take place in which such a description is not useful any more. Turbulent flows are characterized by the formation of structures at many length scales. A description that focuses on a specific length scale, such as a flow visualization through streamlines, captures only a partial aspect of its nature and neglects other, physically relevant components to an exhaustive description of its state.

The number of degrees of freedom required for the numerical simulation of flows is substantially higher in presence of turbulent flows than with laminar flows. In order to obtain an accurate representation of the flow evolution, all structures down to the smallest ones must be representable on the computational grid. With currently available computing resources, it is actually impossible to reach any of the Reynolds numbers that are of interest in most engineering applications. To face this problem, simulations are often simply run on an underresolved grid. They are known as *large eddy simulations*, because they simulate the effect of the large eddies in the flow but do not take into account the small ones. The effect of the small eddies is however not neglected. Instead, a new, modified dynamics is executed on the computational grid, which in the following is called the *coarse grid*. This dynamics replaces the effect of the small eddies, had the simulation been executed on a fully resolved *fine grid*, by an analytical or numerical prediction. This prediction is made on ground of variables simulated on the coarse grid and is of course only an approximation to the exact, fully resolved dynamics. The most common of those *subgrid models* include the notion of an *effective viscosity*: they predict that the adapted dynamics on the coarse grid implements the Navier-Stokes equation with a time and space dependent

viscosity. A rough approximation, the so-called *Smagorinsky model*, relates the effective viscosity to the value of the local strain rate tensor $\boldsymbol{S}$. In more advanced models, a new family of observables is introduced, for which a PDE is solved, and to which the effective viscosity is related. It must be emphasized that all those efforts are required only for the simulation of turbulent flows. When laminar flows are simulated on an underresolved grid, an approximation to the exact solution is obtained without the need for subgrid modeling. This is for example visible in the numerical experiments conducted in Sections 2.4, 3.1 and 4.4, in which the accuracy of the numerical result is observed to scale with the grid resolution. If on the other hand subgrid modeling is omitted in turbulent flows, the numerical results differ fundamentally from the ones of a fully resolved simulation, and, as it is most often the case, numerical instabilities appear.

The aim of this chapter is to analyze whether the closure equations required for the computation of the subgrid viscosity can be simulated at low cost in the LB method, by exploiting the high number of degrees of freedom in typical LB lattice structures. This new focus contrasts fundamentally with the point of view taken so far in this document with respect to the LB method. In the regularized LB model it is argued that, in order to solve the Navier-Stokes equation, it is sufficient to simulate the dynamics of some macroscopic variables, the density ρ, the velocity $\vec{u}$ and the strain rate tensor $\boldsymbol{S}$. The set of q particle distribution functions f_i is thus shown to be overdetermined, and it is consequently reduced in the regularized model. This is no longer the case in the present chapter, in which the original BGK model is used, and the enhanced number of degrees of freedom is explicitly exploited for the needs of subgrid modeling.

The next section contains a brief overview of large eddy models in classical CFD tools. This is followed by the introduction of a new turbulence model, which is directly based on the simulated variables of a LB simulation, the particle distribution functions. The validity of this model is then tested with the help of numerical data obtained from a fully resolved simulation of a turbulent flow.

8.1.2 Large eddy simulations in 3D incompressible fluids

At high Reynold numbers, a fluid enters a turbulent regime in which it is unsteady and non-periodic. Turbulent flows are characterized by the occurrence of eddies whose size may vary over a large range. The larger eddies contain the main portion of the flow energy. This energy is successively transferred to the smaller eddies, and is eliminated by viscous dissipation in the smallest ones. This process is described by the theory of Kolmogorov [51]. It predicts that the size of the smallest eddies is proportional to $Re^{-3/4}$.

In a numerical simulation of a turbulent flow, the smallest eddies must be resolved by the grid. Given three space dimensions, this requires $N = \mathcal{O}(Re^{9/4})$ grid points in the discretization. This straightforward approach, involving the discretization of the grid sufficiently fine for resolving all occurring eddies, is known as *direct numerical simulation* (DNS). In industrial applications such as aerodynamic investigations of automobiles, typical Reynolds numbers are found at a value of 10^6 and above. Hence, solving these types of problems properly would require over 10^{13} grid points, a number which is currently inaccessible in term of storage space or CPU performance.

For this reason, the emphasis in turbulent simulations is put on the macroscopically observed mean values rather than the microscopic detail. The physics of the smaller, subgrid scales is no longer simulated, but replaced by an appropriate model. Along this line of thought the velocity field $\vec{u}$ is split into a mean part $\vec{U} = \langle \vec{u} \rangle$ and the small variations $\vec{u}'$ known as turbulent fluctuations, such that

$$\vec{u} = \vec{U} + \vec{u}'. \tag{8.1}$$

The mean part is, for example, a componentwise statistical, temporal or spatial average. The operator $\langle\rangle$ is often called a *filter*.

The filter $\langle\rangle$ can be applied to the Navier-Stokes equation, Eq. (1.10) on page 8. The corresponding averaged equation is known under as the *Reynolds equation* for the averaged velocity $\vec{U}$:

$$\frac{\partial}{\partial t}\vec{U} + \left(\vec{U} \cdot \overrightarrow{\nabla}\right) \vec{U} = -\frac{1}{\rho_0} \overrightarrow{\nabla} P + \nu \nabla^2 \vec{U} + \overrightarrow{\nabla} \cdot \langle \vec{u}' \vec{u}' \rangle \,, \tag{8.2}$$

where $P = \langle p \rangle$ is the averaged pressure. The continuity equation, Eq. (1.9) becomes

$$\overrightarrow{\nabla} \cdot \vec{U} = 0. \tag{8.3}$$

Equation (8.2) is almost equivalent to the Navier-Stokes equation, except for an additional term depending on the *Reynolds stress tensor* $\boldsymbol{\sigma}(\vec{u}')$ defined as

$$\boldsymbol{\sigma}(\vec{u}') = - \langle \vec{u}' \vec{u}' \rangle \,. \tag{8.4}$$

A turbulence model expresses the Reynolds stress tensor in terms of the resolved mean quantities and potentially some new variables described by a set of additional equations. Widely used and easy to apply is the Smagorinsky approach [52], in which the Reynolds stress tensor is dependent only on the local strain rate. This model is extremely convenient in numerical simulations as it does not introduce new degrees of freedom and does not require the resolution of additional PDE's. It shows however good results only in two dimensions and for very fine mesh widths. An improved version is the very popular $k - \epsilon$ model [53, 54] that describes the Reynolds stress in terms of the strain rate, the turbulent kinetic energy k and the dissipation rate ϵ:

$$k = \frac{1}{2} \left\langle |\vec{u}'|^2 \right\rangle \quad \text{and} \tag{8.5}$$

$$\epsilon = \frac{\nu}{2} \left\langle |\partial_\alpha u'_\beta + \partial_\beta u'_\alpha|^2 \right\rangle . \tag{8.6}$$

The value of those additional variables is determined by two additional transport equations. The Reynolds stresses are approximated through the following term:

$$\boldsymbol{\sigma}(\vec{u}') \approx \mathcal{R}(\boldsymbol{S}, k, \epsilon) \equiv \frac{2}{3} k \, \boldsymbol{I} - 2 \, c_\mu \frac{k^2}{\epsilon} \boldsymbol{S}. \tag{8.7}$$

This expression for the Reynolds stresses is substituted into Eq. (8.2), and a renormalized viscosity ν' is introduced:

$$\nu' = \nu + \nu_T = \nu + 2 c_\mu \frac{k^2}{\epsilon}, \tag{8.8}$$

as well as a renormalized pressure:

$$P' = P + P_T = P + \frac{2}{3}\rho_0 k. \tag{8.9}$$

This leads to the following averaged momentum equation:

$$\frac{\partial}{\partial t}\vec{U} + \left(\vec{U}\cdot\overrightarrow{\nabla}\right)\vec{U} = -\frac{1}{\rho_0}\overrightarrow{\nabla}P' + \overrightarrow{\nabla}\cdot(\nu'\,\boldsymbol{S})\,. \tag{8.10}$$

This equation, as well as the continuity equation and the closure equations for k and ϵ can now be simulated on a coarse grid with reasonable requirements for computational resources. The constant c_μ, as well as some additional constants appearing in the closure equations, cannot be found from theoretical arguments and must be calibrated on numerical experiments. The $k-\epsilon$ model has also successfully been used together with the LB scheme, as has been reported in Ref. [55].

Appropriate turbulence models are difficult to implement in practice. A major difficulty stems from the treatment of the domain boundaries, along which *boundary layers* are observed, which don't obey the universal laws of the Kolmogorov theory and need to be treated separately. An overview on the basics of turbulence modeling, and the key issues for the simulation of turbulent flows, is found in Ref. [56].

8.2 Averaged LB equation

The basic idea developed in this section consists in applying the averaging procedure described in Section 8.1.2 directly to the variables of a lattice Boltzmann simulation, the particle distribution functions f_i. The following calculations are restricted to the BGK model and the the 3-dimensional D3Q19 lattice (see Section 1.3.3). The concepts are extended to other lattices in a straightforward manner. The LB dynamics can be viewed as a map that applies the family of particle distribution functions defined on the computational grid at the initial stage t_0 on their state at a time $t = t_0 + T$:

$$\mathcal{D}_T : f(t_0) \longrightarrow f(t_0 + T). \tag{8.11}$$

The dynamics of the BGK model is governed by a relaxation towards an equilibrium term, described by Eqs. (1.14) and (1.17) in Section 1.3.2. It is described as a relaxation

$$f_i(\vec{r} + v\vec{c}_i, t + \delta t) - f_i(\vec{r}, t) = -\omega(f_i(\vec{r}, t) - f_i^{eq}(\vec{u}(\vec{r}, t), \rho(\vec{r}, t))) \tag{8.12}$$

The equilibrium term is explicited in Eq. (1.18), and it is useful to repeat it here:

$$f_i^{eq}(\vec{u}(\vec{r}, t), \rho(\vec{r}, t)) = \rho\, t_i(1 + 3\,\vec{c}_i\cdot\vec{u} + \frac{9}{2}\,\boldsymbol{Q}_i : \vec{u}\vec{u}). \tag{8.13}$$

In this equation, the generic lattice constants have been replaced by their value on the D3Q19 lattice. The velocity $\vec{u}$, specified by Eq. (1.16), depends linearly on the particle flow densities f_i. Furthermore, the flow is considered incompressible, which is the case at low Mach numbers (see Section 2.4). In that case, the non-linearity of dynamics is entirely captured in the last term of equation 8.13. In this term, the

velocities are projected on the second order base vectors $C^2_{\alpha\beta}$, defined by means of the tensors $Q_{i\alpha\beta}$ specified in Eq. (1.19).

In order to formulate a turbulence model for the LB dynamics, a filter $\langle\rangle$ is introduced which acts as a linear map on the distribution functions f:

$$\langle\,\rangle : f(t) \longrightarrow F(t). \tag{8.14}$$

The dynamics of the filtered population

$$\mathcal{D}^*_T : F(\vec{r},t) \to F(\vec{r},t+T) \tag{8.15}$$

is defined as follows, using the functions composition operator $\circ$:

$$\mathcal{D}^*_T \circ \langle\rangle = \langle\rangle \circ \mathcal{D}_T. \tag{8.16}$$

Equation (8.16) has already been used in Ref. [57] with the hope that a turbulence model can be formulated from the obtained expression. An interpretation of this procedure from the point of view of kinetic theory is presented in [58], and previous work on subgrid models in the framework of kinetic theory is found in Refs. [59, 60]. The main inconvenience of an approach in which the filter is applied individually to each of the particle distribution functions stems from the fact that the obtained model depends on the index i and introduces an unwanted coupling between the distribution functions. A novel approach is therefore used here, in which the filtered equations are projected on the vectors $C^2_{\alpha\beta}$. The resulting quantities are defined as tensor in the physical space, and they replace the q-dimensional vectors in the space of particle distribution functions. This point of view is more favorable to a physical interpretation of the obtained laws. For this, a map $\mathcal{S}$ is introduced which, from a given configuration of the lattice kinetic variables f, yields the strain rate tensor, defined in Eq. (1.6) on page 7, at each space position:

$$\mathcal{S} : f(t) \longrightarrow \boldsymbol{S}(t). \tag{8.17}$$

With this definition, a weaker formulation of Eq. (8.16) is obtained as follows:

$$\mathcal{S} \circ \mathcal{D}^*_T \circ \langle\,\rangle = \mathcal{S} \circ \langle\,\rangle \circ \mathcal{D}_T, \tag{8.18}$$

Calling $\vec{U}$ the velocity associated to F, (8.18) becomes

$$\langle \boldsymbol{S}[\vec{u}(\vec{r},t)]\rangle = \boldsymbol{S}[\vec{U}(\vec{r},t)]. \tag{8.19}$$

This expression is especially useful in the context of LB, as in this case, the rate of strain can be expressed to the first order in terms of local variables on every lattice site.

Turbulence modeling consists in approximating $\mathcal{D}^*_T$ by a term depending only on the filtered variables (in some models, an additional set of variables is introduced). A very straightforward model introduces the idea of a subgrid viscosity, in which the original BGK model is implemented, with a renormalized relaxation parameter ω^*:

$$F_i(\vec{r}+v\vec{c}_i, t+\delta t) - F_i(\vec{r},t) = -\omega^*(\vec{r},t)(F_i(\vec{r},t) - f_i^{eq}(\vec{U}(\vec{r},t), \langle\rho\rangle\,(\vec{r},t))). \tag{8.20}$$

Although this model seems natural, there is no *a priori* reason to support it. There are other models, such as the $k-\epsilon$ model presented in the previous section, that make use of an additional pressure term. For the sake of completeness, it would further be useful to view turbulence models under the light of the multiple relaxation time approaches introduced in Section 3.3.1. It is indeed natural to expect that the value of the renormalized relaxation parameter is different for each of the relaxed models. For the time being, the discussion is however restricted to the case of a single relaxation time, and a numerical verification of assumption (8.20) is proposed in the next section.

It is useful to remember that the strain rate tensor is formally related to the off-equilibrium part of the tensor $\mathbf{\Pi}$. The findings of Eq. (2.66) on page 25 are reproduced by the following equation:

$$S_{\alpha\beta} \approx -\frac{3\omega}{\rho}\Pi^{\text{neq}}_{\alpha\beta}; \qquad \Pi^{\text{neq}}_{\alpha\beta}[f] = \sum_i c_{i\alpha}c_{i\beta}\left(f_i - f_i^{eq}[f]\right). \tag{8.21}$$

With assumption (8.20), equation 8.19 becomes

$$\omega\left\langle\Pi^{\text{neq}}[f]\right\rangle = \omega^*\Pi^{\text{neq}}[F], \tag{8.22}$$

which gives an estimate for the value of the renormalized relaxation parameter:

$$\frac{\omega^*}{\omega} = \frac{\left\langle\Pi^{\text{neq}}_{\alpha\beta}[f]\right\rangle}{\Pi^{\text{neq}}_{\alpha\beta}[F]} \qquad \forall\ \alpha,\beta \in 0,1,2. \tag{8.23}$$

The numerator of the right hand side is further expanded:

$$\left\langle\Pi^{\text{neq}}_{\alpha\beta}[f]\right\rangle = \sum_i c_{i\alpha}c_{i\beta}(F_i - f_i^{eq}[F] - \delta f_i^{eq}) = \Pi^{\text{neq}}_{\alpha\beta}[F] - \sum_i c_{i\alpha}c_{i\beta}\,\delta f_i^{eq}, \tag{8.24}$$

where the correction δf_i^{eq} to the equilibrium term has been defined as

$$\begin{aligned}
\delta f_i^{eq} &\equiv \langle f_i^{eq}[f]\rangle - f_i^{eq}[F] && (8.25)\\
&= \frac{9}{2}t_i\,\boldsymbol{Q}_i : \left(\langle\vec{u}\vec{u}\rangle - \vec{U}\vec{U}\right) = \frac{9}{2}\,\rho\,t_i\,\boldsymbol{Q}_i : \left(\langle\vec{u}'\vec{u}'\rangle\right) && (8.26)\\
&= -\frac{9}{2}\,\rho\,t_i\,\boldsymbol{Q}_i : \sigma. && (8.27)
\end{aligned}$$

Using the properties of the D3Q19 grid introduced in Section 2.1 ($\sum_i t_i\,Q_{i\alpha\beta} = 0$ and $\sum_i\ t_i\,Q_{i\alpha\beta}\,Q_{i\gamma\delta}\,T_{\gamma\delta} = \frac{1}{9}\left(T_{\alpha\beta} + T_{\beta\alpha}\right)$ for any second order tensor $\boldsymbol{T}$), one writes

$$\sum_i t_i\,Q_{i\alpha\beta}\,Q_{i\gamma\delta}\,\sigma_{\gamma\delta} = \frac{2}{9}\,\sigma_{\alpha\beta}, \tag{8.28}$$

and thus, using the definition of $Q_{i\alpha\beta}$,

$$\sum_i c_{i\alpha}c_{i\beta}\,\delta f_i^{eq} = \sum_i Q_{i\alpha\beta}\delta f_i^{eq} + \frac{1}{3}\delta_{\alpha\beta}\sum_i \delta f_i^{eq} = -\rho\,\sigma_{\alpha\beta} \tag{8.29}$$

This yields

$$\left\langle\Pi^{\text{neq}}_{\alpha\beta}[f]\right\rangle = \Pi^{\text{neq}}_{\alpha\beta}[F] + \rho\,\sigma_{\alpha\beta}, \tag{8.30}$$

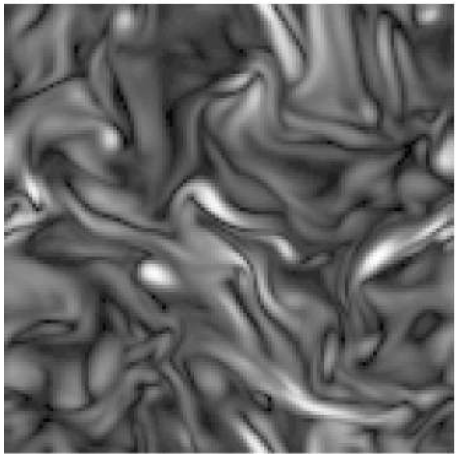


Figure 8.1: 2D cut through the vorticity field of the homogeneous isotropic turbulent flow.

and, using the definition of the tensor $\mathbf{\Pi}$:

$$\frac{\omega^*}{\omega} = 1 + \frac{\rho\,\sigma_{\alpha\beta}}{\Pi^{\mathrm{neq}}_{\alpha\beta}[F]} = 1 - 3\omega\frac{\sigma_{\alpha\beta}}{S_{\alpha\beta}}, \quad \forall\alpha\beta. \tag{8.31}$$

In Eq. 8.31, the effective viscosity is expressed in terms of a ratio between the Reynolds and the strain rate tensor. It is assumed that the various components of the stress tensor share the same effective relaxation time. This is only true for homogeneous isotropic flows, but is nonetheless consistent with the point of view taken by turbulent viscosity models (e.g. Smagorinsky and $k-\epsilon$).

8.3 Direct numerical simulation

In this section, Eq. (8.31) is verified numerically with the data of a DNS. The simulation represents a statistically homogeneous and isotropic flow. The system consists of a cubic lattice of system size $128 \times 128 \times 128$ with periodic boundary conditions. The Reynolds number, defined with respect to the RMS value of the velocity and the side length of the full cubic system, is $Re = 1513$. From this, the Kolmogorov length is found to be $\lambda = 0.5$ in lattice units. Although the Reynolds number is quite low, it is sufficient for obtaining a weakly turbulent flow. The type of forcing needs however to be chosen very carefully. The method used here is known under the name of *spectral forcing* and has been introduced in Ref. [61]. In this method, the force field is constructed in the wavenumber space. The force is constructed in such a way that it is non-zero only at the two lowest wavenumbers, in the low frequency limit. In this way, the energy is introduced at large wavelengths, and the Kolmogorov energy cascade can freely develop on the intermediate wavelengths. To obtain a statistically homogeneous and isotropic field, all components of the force field in the wavenumber space are subject to a random shift of the phase. A pseudo-random number generator of good quality is used to generate fully decorrelated values for those shifts. To end with, the force field is transformed into real space by a Fourier

transform. The method described in Ref. [61] guarantees that the obtained force field is divergenceless. In such a way, the simulation stays close to the limit of fluid incompressibility at each iteration step.

Figure 8.1 shows a two-dimensional cut through the fluid vorticity during a simulation. This picture is typical for the data observed in turbulent flows. In particular, eddies are observed at all length scales, from the size of the system down to the size of a grid cell. The average kinetic energy contained at different wavenumbers is displayed in Fig. 8.2. The curve is irregular at low wavenumbers, where the energy is injected by the method described in the previous paragraph. After that, the energy scales down along the universal range, and it is finally dissipated in the low wavelengths, close to the Kolmogorov length. As it is seen on the figure, the intermediate range agrees only partially with the prediction of the Kolmogorov theory. This is however usual for simulations at such low Reynolds numbers, in which the turbulence is not fully developed.

The model proposed in Eq. 8.31 is finally cross-checked with the data of the DNS. The filter introduced in (8.14) is interpreted as a local spatial average over a cube of size h^3. On every lattice site, ω^* is computed from (8.14). Figure 8.3 plots the measured average value for $\tau^*/\tau = \omega/\omega^*$ on off-diagonal parts of the strain rate over all lattice nodes at a given time step. The variance of the value, which is also plotted on the graph, is unfortunately larger than the average. Therefore, no definite conclusions can be taken from the data. This is most probably due to a lack of statistics, and better results can be expected to be obtained at higher Reynolds numbers. Nevertheless, the graph shows a clear increasing trend of the average value, and shows that the theory developed in the previous section finds at least a partial verification in the numerical data.

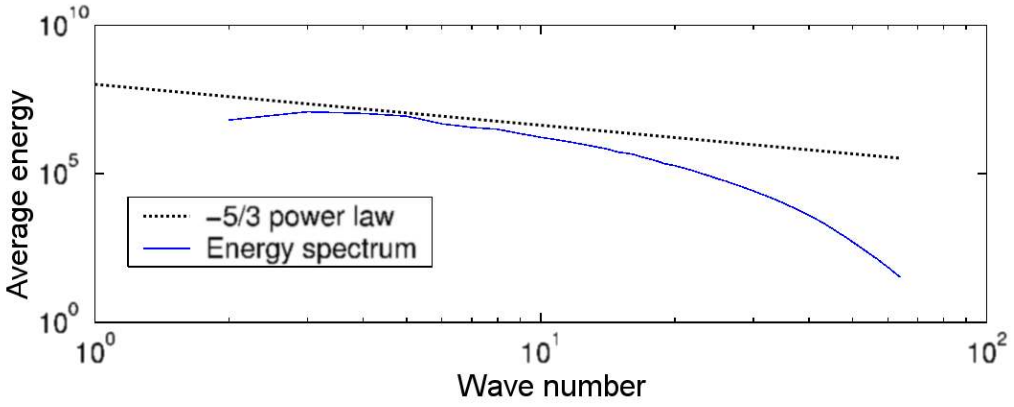


Figure 8.2: Energy spectrum of the turbulent flow.

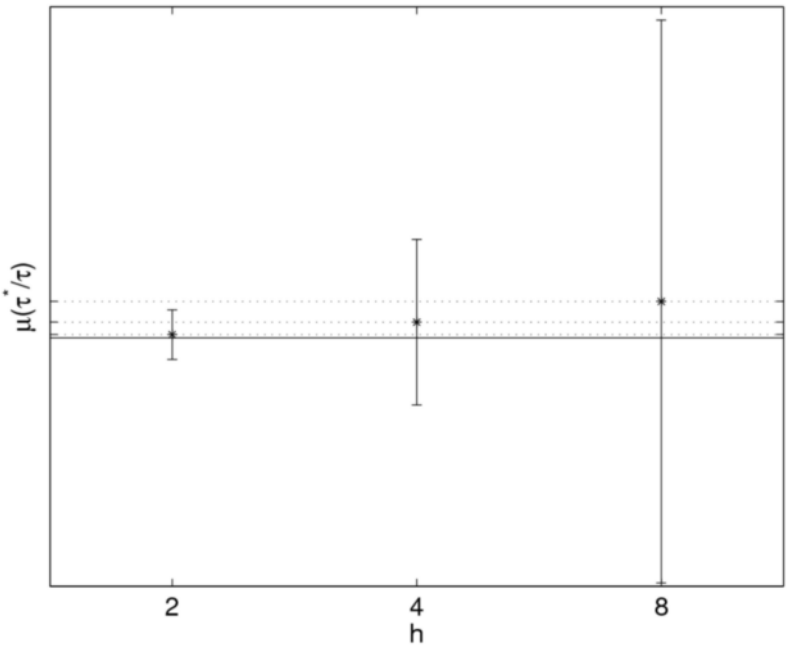


Figure 8.3: Spatial distribution of the effective relaxation time τ^*/τ based on non-diagonal components of σ and Π^{neq}. The marker $(*)$ labels the value of the effective relaxation time, averaged over space at a given time step. The standard deviation over this average is also indicated. In spite of the important standard deviation, a trend towards an increasing effective relaxation time is visible, which increases with the depth h of the average. Although the statistics is rather poor, as a low Reynolds number is used, turbulence modeling via an effective viscosity seems to be supported by numerical evidence.

Publications

Software projects

An important part of the work achieved during the time of the thesis does not appear in the present document. Most notably, important efforts were invested in developing concepts for the numerical implementation of lattice Boltzmann models. Software paradigms for the object-oriented implementation of advanced data structures, as well as programming concepts for the development of massively parallel programs are at the heart of this research topic. During the years of the thesis, three software projects were formulated, each of which is published on an online web page. On those web pages the source code of the project is available for free download, under an open source software license. All projects are furthermore accompanied by a solid documentation for software users and developers. The `OpenLB` project is particularly pointed out, as at the time of the presentation of the thesis, this project is actively maintainted and developed by several authors. The three projects are listed in the following:

Vladymir

`Vladymir` [62] is a library for the C++ programming language that defines a new, multi-functional array type. The library is specifically designed for the purpose of array-based programming - a programming style in which the use of loops over array indices is replaced by simple expressions involving the array globally. Based on this seemingly restrictive programming paradigm, it offers the user two powerful features: straightforward programming style and automatic parallelization of the code on both shared and distributed memory environments. Although it certainly proves useful in many other contexts, the library has mainly been used and tested on the implementation of scientific models such as Cellular Automata and LB schemes, thus implementing roughly nearest-neighbor dynamics.

MPTL

`MPTL` [63] stands for `MultiProcessing Template Library`, a name that hints both at the OpenMP standard and the STL. The `MPTL` proposes a programming model for writing thread-level parallel programs in C++. Within this model, the programmer can parallelize algorithms like those of the Standard Template Library (STL) in such a way that they are executed concurrently by different threads. The parallelization is transparent to the programmer, which means that he doesn't have to tackle any explicit thread-related code, nor even, in most cases, think about a strategy for how to parallelize a given algorithm. The syntax of a parallel code remains very similar to the one used in typical C++/STL programs.

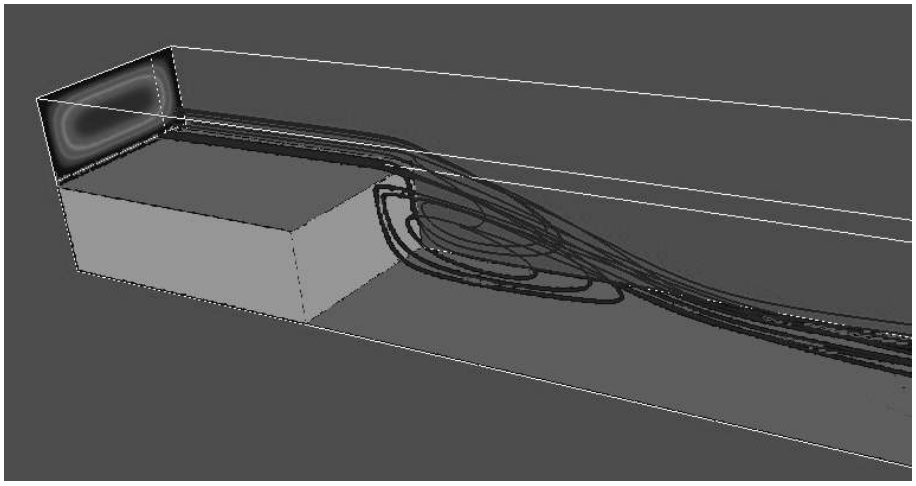


Figure 8.4: Flow behind a 3D backward facing step simulated by the `OpenLB` software.

OpenLB

The `OpenLB` project [13] provides a C++ package for the implementation of lattice Boltzmann simulations that is general enough to address a vast range of problems in computational fluid dynamics. The package is mainly intended as a programming support for researchers and engineers who simulate fluid flows by means of a lattice Boltzmann method. The source code is publicly available and constructed in a well readable, modular way. This enables for a fast implementation of both simple applications and advanced CFD problems. It is also easily extensible to include new physical content. Pre- and postprocessing tools are also available to configure the domain geometry and analyze the numerical results. Figure 8.4 shows the output produced by an example code which is shipped with the standard `OpenLB` package.

Published Articles

Several topics of this thesis have been published in scientific journals, as it can be seen from the following list:

[1] J. Latt, S. Succi B. Chopard, and F. Toschi. Numerical analysis of the averaged flow field in a turbulent lattice Boltzmann simulation. *Physica A*, 362:6–10, 2006.

[2] J. Latt and B. Chopard. Lattice Boltzmann method with regularized non-equilibrium distribution functions. *Math. Comp. Sim.*, 72:165–168, 2006.

[3] J. Latt, Y. Grillet, B. Chopard, and P. Wittwer. Simulating an exterior domain for drag force computations in the lattice Boltzmann method. *Math. Comp. Sim.*, 72:169–172, 2006.

[4] S. Orszag, H. Chen, S. Succi, J. Latt, and B. Chopard. Turbulence effects on kinetic equations. *Journal of Scientific Computing*, 28:459–466, 2006.

[5] J. Latt, G. Courbebaisse, B. Chopard, and J.-L. Falcone. Lattice Boltzmann modeling of injection moulding process. In *Cellular Automata: 6th International Conference on Cellular Automata for Research and Industry*, page 345, Amsterdam, The Netherlands, October 2004. ACRI 2004.

[6] J. Latt and B. Chopard. Vladymir – a c++ matrix library for data-parallel applications. *Future Generation Computer Systems*, 20:1023–1039, 2004.

[7] J. Latt and B. Chopard. An implicitly parallel object-oriented matrix library and its application to medical physics. *IPDPS*, 00:254a, 2003.

[8] S. Marconi, B. Chopard, and J. Latt. Reducing the compressibility of a lattice Boltzmann fluid using a repulsive force. *Int. J. Mod. Phys. C*, 14:1015–1026, 2003.

[9] L3 Collaboration. The $e^+e^- \to z\gamma\gamma \to q\bar{q}\gamma\gamma$ reaction at lep and constraints on anomalous quartic gauge boson couplings. *Phys. Lett. B*, 540:43–51, 2002.

Articles to appear

Some submitted articles have been accepted for publication, but have presently not yet appeared:

[10] J. Latt and B. Chopard. A benchmark case for lattice Boltzmann: turbulent dipole-wall collision. To appear in *Int. J. Mod. Phys. C.*

[11] L. Abrahamyan, J. Latt, A. G. Hoekstra, B. Chopard, and P. M. A. Sloot. Simulating time harmonic flows with the regularized L-BGK method. To appear in *Int. J. Mod. Phys. C.*

[12] V. Heuveline and J. Latt. The OpenLB project: an open source and object oriented implementation To appear in *Int. J. Mod. Phys. C.*

Bibliography

[1] D. Hänel. *Molekulare Gasdynamik*. Springer, 2004.

[2] A. Quarteroni and A. Valli. *Numerical approximation of partial differential equations*. Springer, 1997.

[3] http://www.mathematik.uni-dortmund.de/~kuzmin/cfdintro/cfd.html.

[4] M. Griebel, T. Dornseifer, and T. Neunhoffer. *Numerical simulation in fluid dynamics*. SIAM, 1998.

[5] T. J. Chung. *Computational fluid dynamics*. Cambridge University Press, 2002.

[6] O. C. Zienkiewicz, R. L. Taylor, and P. Nithiarasu. *Finite element method for fluid dynamics*. Elsevier, 2005.

[7] S. Chen and G.D. Doolen. Lattice Boltzmann method for fluid flows. *Annu. Rev. Fluid Mech.*, 30:329–364, 1998.

[8] B. Chopard and M. Droz. *Cellular Automata Modeling of Physical Systems*. Cambridge University Press, 1998.

[9] D. Yu, R. Mei, L.-S. Luo, and W. Shyy. Viscous flow computations with the method of lattice Boltzmann equation. *Progr. Aerosp. Sciences*, 39:329–367, 2003. http://research.nianet.org/∼luo/Reprints-luo/2003/YuDZ_PAS-2003.pdf.

[10] C. Cercignani. *Theory and application of the Boltzmann equation*. Scottisch Academic Press Ltd., 1974.

[11] S. Succi. *The lattice Boltzmann equation, for fluid dynamics and beyond*. Oxford University Press, 2001.

[12] M. C. Sukop and D. T. Thorne. *lattice Boltzmann modeling; an introduction for geoscientists and engineers*. Springer, 2006.

[13] http://www.openlb.org.

[14] D. A. Wolf-Gladrow. *Lattice-Gas Cellular Automata and Lattice Boltzmann models: an introduction*. Lecture Notes in Mathematics, 1725. Springer, 2000.

[15] D. d'Humières, M. Bouzidi, and P. Lallemand. Thirteen-velocity three-dimensional lattice Boltzmann model. *Phys. Rev. E*, 63:066702, 2001.

[16] X. He and L.-S. Luo. Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Phys. Rev. E*, 56:6811–6817, 1997.

[17] Z. Guo, B. Shi, and C. Zheng. A coupled lattice BGK model for the Boussinesq equations. *Internat. J. Numer. Methods Fluids*, 39:325–342, 2002.

[18] Z. Guo, C. Zheng, and B. Shi. Discrete lattice effects on the forcing term in the lattice Boltzmann method. *Phys. Rev. E*, 65:046308, 2002.

[19] Q. S. Zou, S. L. Hou, S. Y. Chen, and G. D. Doolen. An improved incompressible lattice Boltzmann model for time-independent flows. *J. Stat. Phys.*, 81:35–48, 1995.

[20] X. He and L.-S. Luo. Lattice Boltzmann model for the incompressible Navier-Stokes equation. *J. Stat. Phys.*, 88:927–944, 1997.

[21] P. J. Dellar. Incompressible limits of lattice Boltzmann equations using multiple relaxation times. *J. Comput. Phys.*, 190:351–370, 2003.

[22] P. J. Dellar. Bulk and shear viscosities in lattice Boltzmann equations. *Phys. Rev. E*, 64:031203, 2001.

[23] D. d'Humières. Generalized lattice-Boltzmann equations. *Prog. Astronaut. Aeronaut.*, 159:450–458, 1992.

[24] P. Lallemand and L.-S. Luo. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, galilean invariance, and stability. *Phys. Rev. E*, 61:6546–6562, 2000.

[25] R. Benzi, S. Succi, and M. Vergassola. The lattice Boltzmann equation: Theory and applications. *Phys. Reports*, 222:145–197, 1992.

[26] Y. Chen, H. Ohashi, and M. Akiyama. Prandtl number of lattice bhatnagar-gross-krook fluid. *Phys. Fluids*, 7:2280–2282, 1995.

[27] H. Chen, C. Teixeira, and K. Molvig. Digital physics approach to computational fluid dynamics: some basic theoretical features. *Int. J. Mod. Phys. C*, 8:675–684, 1997.

[28] S. Ansumali and I. V. Karlin. Stabilization of the lattice Boltzmann method by the H theorem: a numerical test. *Phys. Rev. E*, 62:7999–8003, 2000.

[29] S. S. Chikatamarla, S. Ansumali, and I. V. Karlin. Entropic lattice Boltzmann models for hydrodynamics in three dimensions. *Phys. Rev. Lett.*, 97:010201, 2006.

[30] B. M. Boghosian, P. J. Love, P. V. Coveney, I. V. Karlin, S. Succi, and J. Yepez. Galilean-invariant lattice-Boltzmann models with H theorem. *Phys. Rev. E*, 68:025103, 2003.

[31] A. Dupuis and B. Chopard. Theory and applications of an alternative lattice Boltzmann grid refinement algorithm. *Phys. Rev. E*, page 066707, 2003.

[32] A. J. C. Ladd and R. Verberg. Lattice-Boltzmann simulations of particle-fluid suspensions. *J. Stat. Phys.*, 104:1191–1251, 2001.

[33] H. Chen, R. Zhang, I. Staroselsky, and M. Jhon. Recovery of full rotational invariance in lattice Boltzmann formulations for high Knudsen number flows. *Physica A*, 362:125–131, 2006.

[34] L. Kovasznay. Laminar flow behind a two-dimensional grid. *Proc. Cambridge Philos. Soc.*, 44:58–62, 1948.

[35] T. Inamuro, M. Yoshino, and F. Ogino. A non-slip boundary condition for lattice Boltzmann simulations. *Phys. Fluids*, 7:2928–2930, 1995.

[36] P. A. Skordos. Initial and boundary conditions for the lattice Boltzmann method. *Phys. Rev. E*, 48:4823–4942, 1993.

[37] U. Ghia, N. Ghia, and C. T. Shin. High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *J. Comp. Phys.*, 48:387–411, 1982.

[38] Q. Zou and X. He. On pressure and velocity boundary conditions for the lattice Boltzmann BGK model. *Phys. Fluids*, 9:1591–1598, 1997.

[39] I. Halliday, L. Hammond, and C. Care. Enhanced closure scheme for lattice Boltzmann equation hydrodynamics. *J. Phys. A: Math. Gen.*, 35:L157–L166, 2002.

[40] H.J.H. Clercx and C.-H. Bruneau. The normal and oblique collision of a dipole with a no-slip boundary. *Comp. fluids*, 35:245–279, 2006.

[41] R. Mei, L.-S. Luo, P. Lallemand, and D. D'Humières. Consistent initial conditions for lattice Boltzmann simulations. *Comp. fluids*, 35:855–862, 2006.

[42] S. Bönisch, V. Heuveline, and P. Wittwer. Adaptive boundary conditions for exterior flow problems. *J. Math. Fluid Mech.*, 7:87–107, 2005.

[43] S. Bönisch, V. Heuveline, and P. Wittwer. Leading order down-stream asymptotics of stationary navier-stokes flows in three dimensions. *J. Math. Fluid Mech.*, 7:147–186, 2005.

[44] P. Wittwer. Second order adaptive boundary conditions for exterior flow problems: non-symmetric stationary flows in two dimensions. *J. Math. Fluid Mech.*, 8:1–26, 2006.

[45] X. He, Q. Zou, L. S. Luo, and M. Dembo. Some progress in the lattice Boltzmann method. part I, non-uniform mesh grids. *J. Comput. Phys.*, 129:357–363, 1996.

[46] O. Filippova and D. Hänel. Grid refinement for lattice-bgk models. *J. Comput. Phys.*, 147:219–228, 1998.

[47] J. Tölke, M. Krafczyk, M. Shulz, and E. Rank. Implicit discretization and nonuniform mesh refinement approaches for FD discretization of LBGK models. *Int. J. Mod. Phys. C*, 9:1143–1157, 1998.

[48] D. Kandhai, W. Soll, S. Chen, A. Hoekstra, and P. Sloot. Finite-difference lattice-BGK methods on nested grids. *Comput. Phys. Commun.*, 129:100–109, 2000.

[49] M. Bouzidi, D. d'Humières, P. Lallemand, and L. S. Luo. Lattice Boltzmann equation on a two-dimensional rectangular grid. *J. Comput. Phys.*, 172:704–717, 2001.

[50] A. M. Artoli, A. G. Hoekstra, and P. M. A. Sloot. Accelerated lattice bgk method for unsteady simulations through mach number annealing. *Internat. J. Modern Phys. C*, 14:835–845, 2003.

[51] A. Kolmogorov. The equations of turbulent motion in an incompressible fluid. *Izv. Sci. USSR Phys.*, 6:56–58, 1942.

[52] J. Smagorinsky. General circulation model of the athmosphere. *Mon. Wheather Rev.*, 91:99–164, 1963.

[53] Launder and Spalding. *Mathematical models of turbulence*. Academic Press, New York, 1975.

[54] B. Mohammadi and O. Pironneau. *Analysis of the k-epsilon turbulence model*. Wiley, 1993.

[55] C. Teixeira. Incorporating turbulence models into the lattice Boltzmann method. *Int. J. Mod. Phys. C*, 9:1159–1175, 1998.

[56] S. Pope. *Turbulent flows*. Cambridge University Press, 2000.

[57] S. Succi, O. Filippova, H. Chen, and S. Orszag. Towards a renormalized lattice Boltzmann equation for fluid turbulence. *Journ. Stat. Phys.*, 107:261–278, 2002.

[58] S. Ansumali, I. Karlin, and S. Succi. Kinetic theory of turbulence modeling: smallness parameter, scaling and derivation of Smagorinsky model. *Physica A*, 338:379–394, 2004.

[59] S. Chen, S. Hou, J. Sterling, and G. D. Doolen. A lattice subgrid model for high reynolds number flows. *Fields Inst. Comm.*, 6:151–166, 1996.

[60] H. Chen, S. Succi, and S. Orszag. Analysis of subgrid scale turbulence using the Boltzmann BGK kinetic equation. *Phys. Rev. E*, 59:2527–2530, 1999.

[61] K. Alvelius. Random forcing of three-dimensional homogeneous turbulence. *Phys. Fluids*, 11:1880–1889, 1999.

[62] http://cui.unige.ch/~latt/vladymir/index.html.

[63] http://spc.unige.ch/mptl.