Artificial Boundary Conditions in the Lattice Boltzmann Method

Dissertation

zur Erlangung
des akademischen Grades
eines
Doktors der Naturwissenschaften (Dr. rer. nat.)

der
Fakultät für Mathematik und Naturwissenschaften
der Bergischen Universität Wuppertal
vorgelegt von

Daniel Heubes

geboren am 08.01.1985 in Haan

betreut von: Prof. Dr. Matthias Ehrhardt
Dr. Andreas Bartel

Februar 2017

Die Dissertation kann wie folgt zitiert werden:

urn:nbn:de:hbz:468-20170207-102047-8
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz%3A468-20170207-102047-8]

Die vorliegende Arbeit wurde von der Fakultät für Mathematik und Naturwissenschaften der Bergischen Universität Wuppertal (BUW) als Dissertation zur Erlangung des Grades eines Doktors der Naturwissenschaften genehmigt.

Prüfungskommission:

Vorsitzender:	Prof. Dr. Michael Günther (BUW)
1. Gutachter:	Prof. Dr. Matthias Ehrhardt (BUW)
2. Gutachter:	Prof. Dr. Wim Vanroose (Universiteit Antwerpen)

Tag der Einreichung:	21. August 2015
Tag der Disputation:	25. Mai 2016

Abstract

The lattice Boltzmann method (LBM) was originally developed as a numerical scheme in the field of computational fluid dynamics. In this dissertation the LBM is considered and the topic of this work lies on the construction of artificial boundary conditions (ABCs) – well suited boundary conditions for artificial boundaries. Artificial boundaries are introduced when a spatial domain is confined to a smaller computational domain. A boundary condition (BC) not tailored for artificial boundaries, like a fixed pressure or velocity condition, behaves in an unphysical manner and generates spurious reflections. Ideally, a BC does not interact with the fluid. Thus, any fluctuation from the interior should leave the computational domain without generating a reflection when impinging on the artificial boundary.

At the beginning of this dissertation a theoretical embedding of the LBM is given. As an example, the D3Q19 model is analytically derived with a numerical integration approach. Moreover the connection of the LBM to macroscopic fluid models (Navier-Stokes equations) is shown, and it is demonstrated that the application of the LBM is not limited to simulate fluid flows.

Based on the theoretical foundation, the features of an ideal boundary condition for artificial boundaries are elucidated. Approaches from literature for the treatment of artificial boundaries are presented. Here a focus lies on perfectly matched layer (PML) approaches and LODI-based characteristic boundary conditions (CBCs). Then the LODI-based CBCs are extended and thus novel ABCs for one- to three-dimensional problems are developed. CBCs are fully described on a macroscopic level, thus differently to the fluid description the LBM is based on. It is explained that from the perspective of the LBM any CBC eventually describes a Dirichlet BC for macroscopic quantities, whose implementation introduces errors.

Afterwards, in the main part of this dissertation the construction of ABCs, which are completely described on the discrete level of the LBM, is pursued. Firstly, a novel procedure based digraphs is introduced for understanding the (temporal) evolution of the discrete quantities within the LBM. With this new understanding a theoretical basis for the formulation of novel general discrete ABCs is constructed by analytically deriving an exact BC for artificial boundaries in 1D. A consistency condition satisfied by this exact discrete ABC is deduced. The one-dimensional exact discrete ABC is approximated, whereby a new parameter is introduced which controls the accuracy of the approximation. The approximated discrete ABC is interpreted as a separate lattice Boltzmann simulation, and this interpretation is generalized. Hence, a discrete ABC for general lattice Boltzmann simulations (not restricted to models recovering Navier-Stokes equations) in higher dimensions is formulated. Error sources of the discrete ABCs, details for their efficient implementation as well as their computational costs are discussed.

All novel ABCs are implemented and applied for a variety of one- to three-dimensional test scenarios. Their performance is compared to selected ABCs from literature. One numerical simulation explains the working principle of the discrete ABCs visually.

Finally, the results of this dissertation are summarized and additionally possible future research tasks are pointed out.

Zusammenfassung

Die Lattice Boltzmann Methode (LBM) ist ursprünglich als ein numerisches Verfahren im Bereich der numerischen Strömungsmechanik entwickelt worden. Diese Arbeit befasst sich mit der LBM und entwickelt für diese künstliche Randbedingungen (kRBen) – speziell geeignete Randbedingungen für künstliche Ränder. Man stößt auf künstliche Ränder, wenn man ein gegebenes räumliches Gebiet für eine Simulation auf ein kleineres Rechengebiet beschränkt. Randbedingungen (RBen), welche nicht für künstliche Ränder konzipiert wurden (beispielsweise RBen, die einen konstanten Druck oder konstante Geschwindigkeit umsetzen) zeigen kein physikalisches Verhalten, da diese störende Reflexionen erzeugen. Idealerweise sollte eine RB nicht das Fluid beeinflussen, das heißt alle Schwankungen aus dem Inneren sollten das Rechengebiet ohne Erzeugung von Reflexionen verlassen, sobald diese auf den Rand treffen.
Am Anfang dieser Arbeit wird die LBM theoretisch eingebettet. Als ein Beispiel wird das Model D3Q19 analytisch hergeleitet, indem Verfahren aus der numerischen Integration angewandt werden. Der Zusammenhang zwischen der LBM und makroskopischen Fluidmodellen (Navier-Stokes Gleichungen) wird dargestellt, außerdem wird aufgezeigt, dass sich die möglichen Anwendungsfälle der LBM nicht auf Simulationen von Fluidströmungen beschränken.
Aufbauend auf der theoretischen Grundlage, werden die Merkmale einer idealen RB für künstliche Ränder erläutert. Aus der Literatur bekannte Ansätze zur Behandlung künstlicher Ränder werden dargestellt. Hierbei liegt ein Schwerpunkt auf PML (perfectly matched layer) basierten Ansätzen sowie auf charakteristischen RBen, welche auf den LODI Gleichungen basieren. Diese LODI basierten charakteristischen RBen werden erweitert und dadurch neue kRBen zur Anwendung ein- bis drei-dimensionaler Problemstellungen entwickelt. Charakteristische RBen sind vollständig auf makroskopischer Ebene formuliert, und unterscheiden sich diesbezüglich von der Fluidbeschreibung innerhalb der LBM. Es wird erklärt, dass vom Standpunkt der LBM, jede charakteristische RB einer Dirichlet RB makroskopischer Größen entspricht, deren Umsetzung in der LBM mit Fehlern einhergeht.
Im Hauptteil der Arbeit wird die Entwicklung von kRBen, die vollständig auf der diskreten Ebene der LBM formuliert sind, verfolgt. Zunächst wird eine neue Vorgehensweise eingeführt, die auf gerichteten Graphen aufbaut. Diese dient dem Verständnis der (zeitlichen) Entwicklung diskreter Größen innerhalb der LBM. Mit Hilfe dieser Vorgehensweise wird analytisch eine exakte eindimensionale diskrete kRB konstruiert. Diese exakte, diskrete kRB stellt die theoretische Grundlage zur Formulierung allgemeingültiger, diskreter kRBen dar. Eine Konsistenzbedingung, welche die exakte diskrete RB erfüllt, wird hergeleitet. Die eindimensionale kRB wird approximiert, wobei ein neuer Parameter eingeführt wird, der die Genauigkeit der Approximation steuert. Die approximierte diskrete RB wird als Lösung einer eigenständigen Lattice Boltzmann Simulation interpretiert. Diese Interpretation wird generalisiert, wodurch diskrete kRBen für allgemeingültige, höherdimensionale Lattice Boltzmann Simulationen entstehen. Fehlerquellen, die Details einer effizienten Implementierung sowie Rechenaufwände der diskreten kRBen werden behandelt.
Alle neuen kRBen werden implementiert und in verschiedenen ein- bis drei-dimensionalen

Testszenarien angewandt. Die Ergebnisse werden mit ausgewählten, bereits existierenden kRBen verglichen. Anhand einer numerischen Simulation wird die Vorgehensweise der entwickelten diskreten RBen visuell dargestellt.
Zum Abschluss werden die Ergebnisse dieser Arbeit zusammengefasst und es werden mögliche weitere Forschungsaufgaben aufgezeigt.

Danksagung

Die vorliegende Arbeit wurde während meiner Tätigkeit als wissenschaftlicher Mitarbeiter in der Arbeitsgruppe *Angewandte Mathematik / Numerische Analysis* der Bergischen Universität Wuppertal erstellt. Mein besonderer Dank gilt hier Prof. Dr. Michael Günther und Prof. Dr. Matthias Ehrhardt, die mich als Mitarbeiter in der Arbeitsgruppe aufgenommen haben. Dieser Dank gilt aber auch für die langjährige, gute Zusammenarbeit und Unterstützung. Somit wurde mir überhaupt erst die Möglichkeit gegeben die vorliegende Arbeit zu verfassen.
Nicht weniger stark fällt mein Dank an meinen fachlichen Betreuer Dr. Andreas Bartel aus. Hier gilt mein Dank nicht nur für die zahlreichen fachlichen Diskussionen, sondern unter anderem auch dafür immer Zeit und ein offenes Ohr gehabt zu haben.
Auch allen Kolleginnen und Kollegen, die mich während meiner Zeit am Lehrstuhl begleitet haben, möchte ich meinen Dank gleichermaßen aussprechen. Das angenehme Arbeitsklima hat sehr zum Gelingen dieser Arbeit beigetragen. Während meiner Tätigkeit teilte ich die gefühlt längste Zeit über ein Büro mit Dmitry Shcherbakov, bei dem ich mich hiermit für die freundschaftliche Atmosphäre bedanken möchte. Gleiches gilt für meine weiteren Bürokollegen, Dr. Long Teng und Zuzana Bučková. Ich bin jedoch sicher, dass ich auch mit meinen Kollegen Christoph Hachtel, Christian Hendricks, Kai Gausling, José Pedro Silva und Michèle Wandelt als Büropartner eine genauso schöne Zeit gehabt hätte.
Ich bedanke mich nicht nur für die Hinweise beim sorgfältigen Korrekturlesen bei Dr. Long Trieu. Kennengelernt haben wir uns als Kommilitonen im gemeinsamen Mathematikstudium, seither besteht eine tiefe Freundschaft. Namentlich erwähnen möchte ich diesbezüglich auch meinen ehemaligen Kommilitonen Dr. Martin Wagner, mit dem mich ebenfalls eine gute Freundschaft verbindet. Stellvertretend für viele weitere, möchte ich mich bei beiden unter anderem für den gegenseitigen Austausch sowie die angenehme Lernatmosphäre während des Studiums bedanken. Durch das Studium wurde eine Grundlage geschaffen, auf die diese Arbeit aufbaut.
Zum Schluss danke ich meinen Eltern, die mir das Studium ermöglichten, stets an mich geglaubt haben und immer unterstützend für mich da waren. Ganz besonderer Dank gilt Anina für ihre Liebe, Geduld und Unterstützung. Die sehr liebevolle und angenehme familiäre Atmosphäre außerhalb der Universität haben einen sehr großen Anteil am Gelingen der vorliegenden Arbeit.

Contents

List of figures

Abbreviations

ABC	artificial boundary condition
BC	boundary condition
BBGKY	Bogoliubov, Born, Green, Kirkwood and Yvon
BGK	Bhatnagar, Gross and Krook
CBC	characteristic boundary condition
CFD	computational fluid dynamics
DABC	discrete artificial boundary condition
DVBE	discrete velocity (BGK-)Boltzmann equation
e.g.	for example (from the Latin *exempli gratia*)
EBC	equilibrium boundary condition
FHP	Frisch, Hasslacher and Pomeau
GPU	graphics processing unit
i.e.	that is (from the Latin *id est*)
IBC	impedance boundary condition
ITBC	ideal transparent boundary condition
LB	lattice Boltzmann
LBE	lattice Boltzmann equation
LBM	lattice Boltzmann method
LODI	local one-dimensional inviscid
MRT	multiple-relaxation-time
nEBC	non-equilibrium boundary condition
NRBC	non-reflecting boundary condition
NSE	Navier-Stokes equations
ODE	ordinary differential equation
PDE	partial differential equation
PML	perfectly matched layer
SRT	single-relaxation-time
TRT	two-relaxation-time
vCBC	viscous characteristic boundary condition

Notation

Throughout this thesis, vectors, matrices and higher order tensors are noted with boldface letters. The components of a vector are written by adding a subscript to the same (nonbold) letter. As an example the fluid velocity in a d-dimensional setting reads $\boldsymbol{u} = (u_1, \dots, u_d)^\top$, where the superscript indicates a transposition. Analogously, the entries of a matrix, say $\boldsymbol{A}$, are written as $A_{j,k}$. In this thesis we do not use Einstein notation. We intend to use a mathematical clean notation, for which we assume that a vector is always a column.

We use the common notation of spaces, in which $\mathbb{R}$ denotes the set of real numbers, as well as $\mathbb{N}$ and $\mathbb{Z}$ the set of natural and integer numbers, respectively. The inner product of two (numeric) vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is written as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$. The nabla operator $\boldsymbol{\nabla} = (\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_d})^\top$ is formally seen as a vector constructed of spatial differential operators $\frac{\partial}{\partial x_j}$. The divergence $\operatorname{div}(\boldsymbol{u})$ of a vector field is therefore equivalent to $\boldsymbol{\nabla} \cdot \boldsymbol{u}$. The application of the nabla operator to a matrix shall be understood as follows:

$$\boldsymbol{\nabla} \cdot \boldsymbol{A} := \begin{pmatrix} \operatorname{div}\left(\boldsymbol{A}_{1,-}^\top\right) \\ \vdots \\ \operatorname{div}(\boldsymbol{A}_{m,-}^\top) \end{pmatrix},$$

where $\boldsymbol{A}_{j,-}^\top$ is the vector corresponding to the j-th row of $\boldsymbol{A}$. That is, the nabla operator is applied to each row of the matrix. The Laplace operator is denoted as $\Delta = \boldsymbol{\nabla} \cdot \boldsymbol{\nabla}$.

List of symbols

Below we present an alphabetically ordered list of symbols. We also give a short description of them and state the page on which the symbols are first used.

Latin letters

$\widetilde{w}_j = \widetilde{w}_j(n_k, p_m)$ 52
a path weight for path p_m, starting in node n_k

$\bar{w}_i^B = \bar{w}_i^B(m, v)$ 61
weight for a path with $(2v-1)$ directional changes, starting in a past boundary point (exact DABC)

$\bar{w}_i^F = \bar{w}_i^F(k, m, v)$ 57
weight for a path with $2v$ directional changes, starting in a fictitious node (exact DABC for time level k)

w_i 18
weights in the discrete equilibrium distribution

$\boldsymbol{x}$ 9
space coordinate

$\boldsymbol{x}_b$ 26
a boundary point, element of $\mathcal{B}$ (or $\mathcal{B}^{\mathrm{ref}}$, $\mathcal{B}^k$)

x_I^k 73
intersection grid point, element of Γ^k

Greek letters

α 51
weight in linear equilibrium (D1Q2)

$\alpha_l = \alpha_l(m)$ 62
node weight of a past boundary population in the exact DABC

$\tilde{\alpha} = \tilde{\alpha}(m)$ 67
auxiliary node weights, closely related to $\alpha_l(m)$

β 51
weight in linear equilibrium (D1Q2)

$\beta_l = \beta_l(k)$ 62
gathered node weights for equilibrium populations at time level k in the exact DABC

$\tilde{\beta}_l = \tilde{\beta}_l(k)$ 66
gathered node weights for equilibrium populations at time level k in the approximate DABC (linear case)

Γ 29
set of open boundary points, subset of $\mathcal{B}$

Γ^k 73
set of open boundary points in the k-th subproblem, subset of $\mathcal{B}^k$

Γ^{ref} 30
set of open boundary points in reference solution, subset of $\mathcal{B}^{\mathrm{ref}}$

γ 51
weight in linear equilibrium (D1Q2)

Δt 16
time step-size

δ 51
weight in linear equilibrium (D1Q2)

κ 54
shortcut for $\kappa = \lfloor \frac{k}{2} \rfloor$

$\boldsymbol{\Lambda}_{x_\alpha}$ 38
diagonal matrix of eigenvalues

$\lambda_{x_\alpha, j}$ 38
eigenvalues

μ 9
dynamic viscosity

ν 21
kinematic viscosity

$\boldsymbol{\xi}$ 10
particle velocity

$\rho = \rho(\boldsymbol{x}, t)$ 9
mass density (Navier-Stokes equations)

$\rho = \rho(\boldsymbol{x}, t)$ 12
mass density in LBM

ρ_0 9
constant mass density of an incompressible fluid

$\rho_0 = \rho_0(x)$ 86
initial density profile

$\rho_D = \rho_D(\boldsymbol{x}_b)$ 41
Dirichlet value for mass density

$\rho^+ = \rho^+(\boldsymbol{x}_b, t)$ 31
sum of populations corresponding to $\mathcal{I}_{\mathcal{G}}^+(\boldsymbol{x}_b)$

$\rho^- = \rho^-(\boldsymbol{x}_b, t)$ 31
sum of populations corresponding to $\mathcal{I}_{\mathcal{G}}^-(\boldsymbol{x}_b)$

$\rho^0 = \rho^0(\boldsymbol{x}_b, t)$ 31
sum of populations corresponding to $\mathcal{I}_{\mathcal{G}}^0(\boldsymbol{x}_b)$

$\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{x}, t)$ 9
deviatoric stress tensor

σ_x, σ_y 32
PML absorption coefficients

$\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{x}, t)$ 12
deviatoric stress tensor in LBM

τ 14
relaxation parameter

ϕ 96
angle of incidence

χ 25
thermal conductivity

ψ_j 11
a collision invariant

Ψ 17
polynomial

Ω 25
computational domain

ω 16
relaxation time

ω_g 24
relaxation time (thermal model)

$\boldsymbol{\omega} = \boldsymbol{\omega}(\boldsymbol{x}, t)$ 96
vorticity

Chapter 1

Introduction

This thesis considers a specific numerical method in the field of *computational fluid dynamics* (CFD). This method, known as the *lattice Boltzmann method* (LBM), originates from *lattice gas automata*, and was first introduced by McNamara and Zanetti in 1988 [74]. Since its first appearance the LBM was permanently improved and is nowadays well established in industry, e.g., the widely used CFD software products of Exa Corp. are based on the LBM.
In contrast to *lattice gas automata*, which compute the evolution of single particles on a computational grid [120], the LBM focuses on the evolution of ensemble averages (so-called *populations*). By this, statistical fluctuations are eliminated, where *lattice gas automata* suffer from. However, it inherits the algorithmic advantages of its predecessor, which makes the LBM computational efficient and competitive to earlier established methods in the field of CFD. Indeed, mathematical analysis can show its relation with the *Navier-Stokes equations* (NSE), such that the LBM can be seen as an approximate explicit solver for the NSE [55]. Hence, although it is based on a *mesoscopic* formulation, its application usually focuses on the macroscopic behavior. However, the LBM also benefits from its kinetic-based formulation, which distinguishes it from conventional CFD solvers. The advantages particularly lie in an easy handling of complex and irregular geometries and in the ease of incorporating effects which are harder to describe on a macroscopic level [103]. Thus, the LBM has demonstrated its success for various applications where macroscopic solvers have more difficulty with. In particular, the simulation of flows in complex porous media [35, 58, 79], as well as multiphase and multicomponent flows [15, 40, 73].
The local particle collisions in simulations with *lattice gas automata* is transferred to *lattice Boltzmann* (LB) simulations as a local and explicit evaluation of a collision formula in terms of *populations*. Non-local operations usually consist only in the exchange of *populations* between adjacent grid points. Differently speaking, the method is fully explicit and the majority of calculations is done locally, which declares the direct possibility for a parallelization of the implementation. Therefore, it can also be implemented efficiently on conventional graphics hardware [110, 121]. To sum up, the LBM emerged as an efficient numerical method in the field of CFD, capable of solving the NSE approximately. Chapter 2 deals with the LBM, where we especially derive the method analytically and show its relation to the *Boltzmann equation* and to the NSE.

The efficiency of a simulation is not only determined by the underlying method, but also by the size of the computational domain. Usually in an application one is interested in the solution only in a bounded *domain of interest*, while the problem itself is theoretically formulated on a much larger or even unbounded domain. For instance, when simulating the flow past an object, the flow behavior nearby the object is relevant, while the situation far away from the object is not. The restriction to a bounded domain introduces *artificial boundaries*, where *a priori* no physical information is known. However, for all boundaries we have to formulate *boundary conditions* (BCs), this task holds for any simulation independent of the method used. Ideally these BCs, so-called *artificial boundary conditions*

(ABCs), are chosen such that the solution on the bounded domain matches the solution on the larger/unbounded domain. Unfortunately, using a standard BC which can implement known physical information, e.g., a prescribed pressure or velocity, leads to an unphysical behavior at the boundary. That is, the boundaries produce unphysical reflections, which influence the results in the interior of the computational domain. An intuitive way to overcome spurious reflections from the boundaries is to extend the computational domain. Not only that it would be inefficient, but also a long-term simulation would never be possible, since reflected waves eventually reach the *domain of interest* anyway. Thus, using an ABC increases the efficiency of the simulation, and also allows for long-term simulations. On the macroscopic scale, several studies on ABCs in the field of CFD were performed. Here, the pioneering work for wave equations was established by Engquist and Majda [27]. Hedstrom [44] and later Thompson [108, 109] developed non-reflecting *characteristic boundary conditions* (CBCs) for nonlinear hyperbolic equations. Kröner [65] devised approximations to exact absorbing BCs for the two-dimensional linear Euler equations. *Non-reflecting boundary conditions* (NRBCs) for the Navier-Stokes equations were presented by Poinsot and Lele [84]. A review on absorbing boundary conditions for hyperbolic systems can be found in [25].
In a fully discrete approach, Wilson [119] and later Rowley and Colonius [90] derived non-reflecting BCs for the linear Euler equations. Their fully discrete approaches, formulated directly for the chosen numerical scheme, possess the advantage that the BCs are already perfectly adapted to the interior scheme resulting in higher accuracy and better stability properties compared to the previous approaches.

For the LBM the situation is different, there exist only a few studies on ABCs. Kam et al. [57] compared first approaches in their article from 2007. In a LB simulation a BC has the task to determine unknown populations, hence any macroscopic formulation of an ABC cannot be applied directly. Therefore, if possible at all, a macroscopic ABC has to be transferred appropriately to the mesoscopic level of *populations*. Studies based on such a transfer are done by Izquierdo and Fueyo [54] and Kim et al. [59], who transfer a non-reflecting CBC based on the *local one-dimensional inviscid* (LODI) equations. Another approach on ABCs for the LBM is pursued by the works of Najafi-Yazdi and Mongeau [77], Craig and Hu [18], as well as Tekitek et al. [106]. The principle in these works is based on Bérenger's concept of *perfectly matched layer* [4], devised for the absorption of electromagnetic waves. Furthermore, Schlaffer [93] introduced recently *impedance boundary conditions* for the LBM, in which the acoustic impedance of the fluid is chosen appropriately, aiming that reflections are prevented. In Chapter 3 we go into more detail and present ABCs for LB simulations, which originate from macroscopic considerations. Here we also present our work [46], developed to improve existing non-reflecting CBCs. Moreover we give further enhancements to the non-reflecting CBCs from the literature.

The implementation of macroscopic information to the *populations* is not uniquely possible, thus errors are likely created. This problem concerns not only when using BCs derived from macroscopic considerations, but also occurs in the initialization of the method, where usually only a macroscopic information is known. Initialization of a LB simulation is also subject of research [113]. On the other hand, in the last two decades the LBM has been greatly developed. Hence, apart from being a solver for the NSE, in recent years the LBM has been shown to be applicable also for other kind of problems [67, 70, 75, 76, 78, 97, 111, 124, 126]. Thus, macroscopic BCs for NSE are not applicable.
A major goal of this thesis is to overcome these issues and to devise a novel concept for artificial BCs, which is formulated purely on the discrete level. The discrete formulation

has the advantage to be applicable in any LB simulation, including those in addition to the NSE. A discrete BC can be used even if no macroscopic formulation of a BC exists. To realize our goal we first consider the LBM with a linear collision term in Chapter 4 and derive an exact BC, formulated on the discrete level of populations. In Chapter 5 we approximate the exact discrete BC and generalize the approximation to LB simulations with arbitrary collision models. The next subsection further explains the structure of this thesis.

1.1 Outline of this thesis

In this thesis we study various approaches for artificial boundary conditions in the LBM. Next we state a brief summary of the following chapters.

Chapter 2 – Fluid dynamics and the lattice Boltzmann method The goal of this chapter is to provide a theoretical embedding of the LBM. We begin with a short presentation of fluid flow models on different levels of description. There, we continue considering the *Boltzmann equation* and show its relation to macroscopic models. An alternative approximate collision model is introduced and we derive the LBM by a discretization of the Boltzmann equation, where the discrete velocities of the so-called *D3Q19 model* are analytically derived from a numerical integration approach. Moreover, it is elucidated that the choice of the equilibrium distribution within the collision term strongly influences the equivalent macroscopic formulation. The chapter ends with the introduction of a short cut notation of a complete lattice Boltzmann simulation, used in the following chapters. Hereby we also explain the boundary problem in the perspective of a lattice Boltzmann simulation.

Chapter 3 – Non-reflecting boundary conditions In this chapter we explain the features of an *ideal boundary condition* for *artificial boundaries* in the LBM. We present several existing approaches for the treatment of artificial boundaries in the LBM, aimed at preventing spurious reflection at the artificial boundaries. In detail we consider *impedance boundary conditions*, the concept of *perfectly matched layer* and LODI-based CBCs. Then we present in detail our contribution to CBCs, where we extend the LODI-based boundary conditions. Furthermore we explain further enhancements of CBCs.

Chapter 4 – Exact discrete artificial boundary condition for linear collision model At the beginning of this chapter we explain the use of *digraphs* to understand the evolution of populations in a general lattice Boltzmann simulation. Afterwards, we restrict the consideration to a one-dimensional lattice Boltzmann model with two discrete velocities (D1Q2 model) and a linear collision operator. For this model, we apply the digraph interpretation and obtain *weighted digraphs*. With their help an *exact artificial boundary condition* is formulated purely on the discrete level. Its mathematical formulation is analytically derived in detail and we compute a consistency condition satisfied by this exact *discrete artificial boundary condition*.

Chapter 5 – Discrete artificial boundary conditions (DABCs) We begin with a derivation of an analytically founded approximation to the exact boundary condition of Chapter 4. The accuracy of this approximation is controlled by a free parameter, the *history depth*, which remains present in the general case. We give an interpretation of this approximate boundary condition as a separate lattice Boltzmann simulation. Based on this interpretation we generalize the boundary condition and achieve discrete artificial boundary conditions for general lattice Boltzmann simulations in higher dimensions. This chapter ends with an analysis of discrete artificial boundary conditions, where we elucidate their error sources, explain an efficient implementation and consider their computational costs.

Chapter 6 – Numerical results In this chapter we present the results of six numerical tests ranging from one to three space dimensions. The first example focuses on the approximate discrete artificial boundary condition for the 1D model of Chapter 4. Then we consider a one-dimensional pressure wave to test several artificial boundary conditions. At the third test case we visually explain the working principle of the general discrete boundary conditions. Afterwards, we present the results of the simulation of an isolated two-dimensional vortex. The next numerical test is academic and demonstrates how errors of boundary conditions depend on the angle of incidence, shown on the basis of a plane wave simulation. Finally we simulate the flow past an obstacle in a three-dimensional duct and present the results.

Chapter 7 – Conclusions and outlook The last chapter summarizes the results of this thesis. Additionally, we point out possible future research tasks.

1.2 Related scientific works

During the doctoral studies of the author, five scientific manuscripts have been written by the authors Heubes, Bartel and Ehrhardt. In chronological order, they are:

An Introduction to the Lattice Boltzmann Method for Coupled Problems (2013) [45]

In: M. Ehrhardt (Ed.), *Progress in Computational Physics, Volume 3: Novel Trends in Lattice-Boltzmann Methods: Reactive Flow, Physicochemical Transport and Fluid-Structure Interaction*, pages 3–30

Abstract: The first part of this introduction is devoted to the known derivation of the lattice Boltzmann method (LBM): We track two different derivations, a historical one (via lattice gas automata) and a theoretical version (via a discretization of the Boltzmann equation). Thereby the collision term is approximated with a single relaxation time model (BGK) and we motivate the introduction of this common approximation. By applying a multiscale expansion (Chapman-Enskog), the solution of the numerical method is verified as a meaningful approximation of the solution of the Navier-Stokes equations. To state a well posed problem, common boundary conditions are introduced and their realization within a LBM is discussed.
In the second part, the LBM is extended to handle coupled problems. Four cases are investigated: (i) multiphase and multicomponent flow, (ii) additional forces, (iii) the coupling

to heat transport, (iv) coupling of electric circuits with power dissipation (as heat) and heat transport.

Characteristic boundary conditions in the lattice Boltzmann method for fluid and gas dynamics (2014) [46]

Journal of Computational and Applied Mathematics, 262: 51–61

Abstract: For numerically solving fluid dynamics problems efficiently one is often facing the problem of having to confine the computational domain to a small domain of interest introducing so-called non-reflecting boundary conditions (NRBCs).
In this work we address the problem of supplying NRBCs in fluid simulations in two space dimensions using the lattice Boltzmann method (LBM): so-called characteristic boundary conditions are revisited and transferred to the framework of lattice Boltzmann simulations.
Numerical tests show clearly that the unwanted unphysical reflections can be reduced significantly by applying our newly developed methods. Hereby the key idea is to transfer and generalize Thompson's boundary conditions originally developed for the nonlinear Euler equations of gas dynamics to the setting of lattice Boltzmann methods. Finally, we give strong numerical evidence that the proposed methods possess a long-time stability property.

Exact Artificial Boundary Conditions for a Lattice Boltzmann Method (2014) [47]

Computers & Mathematics with Applications, 67(11): 2041–2054

Abstract: When using a lattice Boltzmann method on an unbounded (or very large) domain one has to confine this spatial domain to a computational domain. This is realized by introducing so-called artificial boundary conditions. Until recently, characteristic boundary conditions for the Euler equations were considered and adapted to the lattice Boltzmann method.
In this work we propose novel discrete artificial boundary conditions which are derived directly for the chosen lattice Boltzmann model, i.e., on the discrete level. They represent the first exact artificial boundary conditions for lattice Boltzmann methods. Doing so, we avoid any detour of considering continuous equations and obtain boundary conditions that are perfectly adapted to the chosen numerical scheme. We illustrate the idea for a one dimensional, two velocity (D1Q2) lattice Boltzmann method and show how the computational efficiency can be increased by a finite memory approach. Analytical investigations and numerical results finally demonstrate the advantages of our new boundary condition compared to previously used artificial boundary conditions.

Concept for a one-dimensional discrete artificial boundary condition for the lattice Boltzmann method (2015) [48]

Computers & Mathematics with Applications, 70(10): 2316–2330

Abstract: This article deals with artificial boundaries which you encounter when a large spatial domain is confined to a smaller computational domain. Such an artificial boundary condition should not preferably interact with the fluid at all. Standard boundary conditions, e.g., a pressure or velocity condition, result in unphysical reflections. So far,

existing artificial boundary conditions for the lattice Boltzmann method (LBM) are transferred from macroscopic formulations.
In this work we propose novel discrete artificial boundary conditions (ABCs) which are tailored on the LBM's mesoscopic level. They are derived directly for the chosen LBM with the aim of higher accuracy. We describe the idea of discrete ABCs in a three velocity (D1Q3) model governing the Navier-Stokes equations in one dimension. Numerical results finally demonstrate the superiority of our new boundary condition in terms of accuracy compared to previously used ABCs.

Discrete artificial boundary conditions for the lattice Boltzmann method in 2D (2015) [49]

ESAIM: Proceedings and Surveys, 52: 47–65

Abstract: To confine a spatial domain to a smaller computational domain, one needs artificial boundaries. This work considers the lattice Boltzmann method and deals with boundary conditions for these open boundaries. Ideally, such a condition does not interact with the fluid at all. We present novel two-dimensional discrete artificial boundary conditions to pursue that goal and we discuss four different versions. This type of condition is formulated on the discrete lattice Boltzmann level and does not require a PDE formulation of the fluid. We set a special focus on the D2Q9 model. Our numerical results compare the novel discrete artificial boundary conditions to simulations using the existing non-reflecting characteristic boundary condition and an exit boundary condition.

Chapter 2

2 Fluid dynamics and the lattice Boltzmann method

A *fluid* refers to substances, which will continually deform under an application of shear stress. The deformation of the fluid is commonly called the *fluid flow*. The shape and size of fluids can change in time, due to compressibility and application of forces. Fluids normally refer to liquids or gases, where liquids only adapt their shape to the container in which they are filled in, whereas gases additionally change their size impressively. Gases fill the entire available space. For additional general information about fluids and their dynamics, we refer to the books [3, 85, 100].

The goal of *computational fluid dynamics* (CFD) is to numerically simulate the fluid flow. To this end, the fluid flow has to be described by a mathematical model. This model, usually a system of differential equations, is solved numerically. Doing so, an approximate solution to the mathematical model and thus the fluid's motion is found [28, 118]. Moreover, there is no unique model describing a fluid flow, but several approaches exist.

2.1 Modeling fluid flows

We briefly present some models for describing a fluid's motion, which are here sorted according to the level of description. The choice which level can be used for a simulation is mainly based on the given problem, especially it is based on the attributed properties of the fluid. If the fluid is given as a highly rarefied gas, each molecule is important and a macroscopic model has to be discarded, since its underlying continuum assumption is violated [51, 99]. On the contrary, if the continuum assumption holds a simulation based on a microscopic model might become very inefficient, due to the high number of molecules. However some effects can be easier depicted and thus incorporated into a model on a microscopic level, why it can be useful to consider a microscopic model even if a continuum assumption is satisfied. A special focus lies on the mesoscopic fluid flow models, since the *lattice Boltzmann method* (LBM) is based on a formulation on this level. The mesoscopic level is settled in between a microscopic and macroscopic formulation, it combines the advantages of both approaches [10].

2.1.1 Microscopic formulation

Considering a gas consisting of N particles (molecules), the motion of this fluid can be described by simulating each particle of the gas explicitly, taking into account the interaction with other particles. Newton's rules of mechanics hold for each particle and a system

of kinematic equations of motions for all N particles can be constructed. Alternatively, the fluid flow can be expressed microscopically by the Hamilton equations of motion [2]:

$$\frac{\mathrm{d}\boldsymbol{q}_k(t)}{\mathrm{d}t} = \frac{\partial \mathcal{H}(t, \boldsymbol{q}, \tilde{\boldsymbol{p}})}{\partial \boldsymbol{p}_k(t)}, \quad \frac{\mathrm{d}\boldsymbol{p}_k(t)}{\mathrm{d}t} = -\frac{\partial \mathcal{H}(t, \boldsymbol{q}, \tilde{\boldsymbol{p}})}{\partial \boldsymbol{q}_k(t)}, \quad k = 1, \ldots, N. \tag{2.1}$$

The system of differential equations (2.1) describes the temporal change of generalized position coordinates $\boldsymbol{q}(t) = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_N)$ and generalized momenta $\tilde{\boldsymbol{p}}(t) = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N)$. The time variable is denoted by t and the Hamiltonian $\mathcal{H}(t, \boldsymbol{q}, \tilde{\boldsymbol{p}})$ expresses the energy of the system.

A microscopic simulation of an entire fluid is usually not practical, since the number of particles is then too large, $N > 10^{20}$. There might be situations where one is interested in the movement of individual particles. However, then it is usually sufficient to concentrate on a smaller fraction of the gas, having a manageable amount of particles.

2.1.2 Macroscopic formulation

In many applications, there is no need to have a detailed information about the movement of each particle. Instead, it is sufficient to know the average quantities within small volumes, e.g., the average velocity of particles in the volume. The values should be computed by averaging a sufficient amount of particles within a volume, such that they are not affected by stochastic fluctuations, especially they should not be determined by one specific particle if the volume is decreased. It requires that the gas is sufficiently dense. Then the volumes can become infinitesimal small without that the averaged quantities change, such that the average values are given pointwise. That is, the gas is described as a continuum, which means the gas is considered as if there are no empty spaces between the particles. This assumption obviously also holds for liquids. The Knudsen number Kn for gases is defined as the ratio of the mean free path l_f and a representative physical length scale L [102]:

$$\mathrm{Kn} := \frac{l_f}{L}.$$

The continuum assumption is equivalent to the mathematical requirement of having a very small Knudsen number $\mathrm{Kn} \ll 1$. In liquids the molecules are predominantly not in random motion, but in a vibrational mode. Therefore, the mean free path l_f in the definition of the Knudsen number is less reasonable and may be replaced by the intermolecular free length l_{im} for liquids [63]. As an example, for water this length is about $10^{-10}\,\mathrm{nm}$, resulting in a Knudsen number close to zero, approving the continuum assumption is valid for liquids as well.

Mathematical models for describing the fluid's motion in the continuum range are usually formulated in terms of pointwise given average quantities with respect to the fluid's particles, referred to as macroscopic dynamic quantities. These quantities are continuous and their local change can be expressed by differentials. Therefore, one obtains a model for the fluid dynamic by a system of appropriate differential equations in terms of the dynamic quantities. The resulting system often further depends on fluid dependent parameters. Typically the differential equations express conservation laws for dynamic quantities [107].

In the hydrodynamic regime $\mathrm{Kn} \leq 0.01$ the motion of an incompressible Newtonian fluid, featured by having a constant mass density $\rho_0 \in \mathbb{R}^+$, can be modeled by the incompressible *Navier-Stokes equations* (NSE) [107]:

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0, \tag{2.2a}$$

$$\rho_0 \frac{\partial \boldsymbol{u}}{\partial t} + \rho_0 \left(\boldsymbol{u} \cdot \boldsymbol{\nabla}\right) \boldsymbol{u} = -\boldsymbol{\nabla} p + \mu \Delta \boldsymbol{u}. \tag{2.2b}$$

Unknowns of this system are the fluid velocity $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t) \in \mathbb{R}^d$ and the local pressure $p = p(\boldsymbol{x}, t) \in \mathbb{R}$, where $\boldsymbol{x} \in \mathbb{R}^d$ is the space coordinate in a d-dimensional space. The first equation (2.2a) states the incompressibility of the fluid. The second equation (2.2b) describes a balance equation of momenta. Here, the fluid is driven by pressure fluctuations, and a viscous stress term $\mu \Delta \boldsymbol{u}$, with the fluid dependent dynamic viscosity $\mu \in \mathbb{R}^+$. The equations (2.2) describe appropriately the flow of a compressible fluid, if the compressibility effects are negligible. However, if they become more important the model has to be extended. So, for compressible fluids, i.e., when the mass density $\rho = \rho(\boldsymbol{x}, t) \in \mathbb{R}^+$ is not a constant, but may vary in space and time, we need to consider a compressible version of the NSE [83]:

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) = 0, \tag{2.3a}$$

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot \left(\rho \boldsymbol{u} \boldsymbol{u}^\top\right) = -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}. \tag{2.3b}$$

Equation (2.3a), known as the continuity equation, describes the conservation of mass. For a constant density this equation reduces to (2.2a). The second equation (2.3b) again is a balance equation for momenta, i.e., it describes the conservation of momenta. In the second term on the right hand side $\boldsymbol{\sigma}$ is the deviatoric stress tensor, e.g., it can have the form

$$\boldsymbol{\sigma} = \mu \left[\boldsymbol{\nabla} \boldsymbol{u} + \boldsymbol{\nabla} \boldsymbol{u}^\top - \frac{2}{3} (\boldsymbol{\nabla} \cdot \boldsymbol{u}) \boldsymbol{I}\right],$$

where $\boldsymbol{I} \in \mathbb{R}^{d \times d}$ denotes the identity. For a discussion and other choices see, e.g., [19]. Normally the system (2.3) is expanded by an energy conservation equation, but we omit this equation here, because we consider the LBM for athermal flows. For the system of conservation equations for compressible flows (here (2.3) without energy conservation) a closure relation is required. This becomes directly obvious by inspecting the number of unknowns and the number of equations. If (2.3) is compared to the system (2.2), we have the same amount of equations, but since ρ is a dynamic quantity, one additional unknown. Usually, the system of the compressible NSE is closed by an equation of state, which models the pressure in dependence of the mass density [28].

If the given fluid becomes inviscid, meaning viscous effects are negligible, the Navier-Stokes system transforms into the Euler equations (here for the compressible version (2.3)):

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) = 0, \tag{2.4a}$$

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot \left(\rho \boldsymbol{u} \boldsymbol{u}^\top\right) + \boldsymbol{\nabla} p = 0. \tag{2.4b}$$

The Euler equations can be formally seen as the NSE with viscosity set to zero. From

a mathematical point of view the systems (2.3) and (2.4) differ not only by the viscous stress term. But, the system of NSE (2.3) is parabolic, whereas the Euler equations (2.4) are hyperbolic.

The macroscopic fluid models are nowadays the commonly most used ones. Analytical solutions of the NSE are known only for few problems, e.g., [23, 89]. Especially the nonlinearity in the convective term causes mathematical difficulties. So generally the NSE (or Euler equations) are solved numerically. Several techniques exist, details are given for example in [28, 29, 71, 118].

2.1.3 Mesoscopic formulation

A fluid flow can also be described on a level between a microscopic (Section 2.1.1) and a macroscopic formulation (Section 2.1.2). This is done commonly by the Boltzmann equation, which uses a mesoscopic description of the fluid. The Boltzmann equation can be derived, e.g., from a microscopic formulation by some approximations. The derivation is sketched here, a full (and also different) derivation can be found in [10, 102]. Instead of beginning the derivation with Hamilton equations of motion (2.1) one considers an N-particle distribution function P_N. Here,

$$P_N(\boldsymbol{q}_1, \boldsymbol{p}_1, \boldsymbol{q}_2, \boldsymbol{p}_2, \ldots, \boldsymbol{q}_N, \boldsymbol{p}_N, t)\, \mathrm{d}\boldsymbol{q}_1 \ldots \mathrm{d}\boldsymbol{q}_N\, \mathrm{d}\boldsymbol{p}_1 \ldots \mathrm{d}\boldsymbol{p}_N$$

gives the probability to find this N-particle ensemble with the i-th particle in an infinitesimal small space volume $\mathrm{d}\boldsymbol{q}_i$ around $\boldsymbol{q}_i$ and with momentum in the infinitesimal small volume $\mathrm{d}\boldsymbol{p}_i$ around $\boldsymbol{p}_i$ at time t. Then the Liouville equation, which describes the temporal evolution of P_N, is equivalently written as the so-called BBGKY-hierarchy [2, 9]. The well-known abbreviation is due to the initials of Bogoliubov [6, 7], Born & Green [8], Kirkwood [61, 62] and Yvon [122]. The BBGKY-hierarchy is a system of coupled equations, the s-th equation is an evolution equation for the s-particle distribution function

$$\begin{aligned} &P_N^{(s)}(\boldsymbol{q}_1, \boldsymbol{p}_1, \boldsymbol{q}_2, \boldsymbol{p}_2, \ldots, \boldsymbol{q}_s, \boldsymbol{p}_s, t) \\ &\qquad = \int P_N(\boldsymbol{q}_1, \boldsymbol{p}_1, \boldsymbol{q}_2, \boldsymbol{p}_2, \ldots, \boldsymbol{q}_N, \boldsymbol{p}_N, t)\, \mathrm{d}\boldsymbol{q}_{s+1} \ldots \mathrm{d}\boldsymbol{q}_N\, \mathrm{d}\boldsymbol{p}_{s+1} \ldots \mathrm{d}\boldsymbol{p}_N, \end{aligned}$$

and has an direct coupling only to the $(s+1)$-th distribution function. The integration is done with respect to the coordinates and momenta of $N - s$ molecules. For more details we refer for example to [10]. Thus, the first equation of the equivalent BBGKY-hierarchy is an equation for a single particle distribution function and it is coupled to higher particle distribution functions by an integral describing collisions of particles. To obtain the Boltzmann equation, only two particles are assumed to be involved in a collision and it is assumed that the involved particles are uncorrelated before collision. We end up with the Boltzmann equation for the single particle distribution $f(\boldsymbol{\xi}, \boldsymbol{x}, t)$ $\big($which is proportional to $P_N^{(1)}\big)$ [10]:

$$\frac{\partial f(\boldsymbol{\xi}, \boldsymbol{x}, t)}{\partial t} + \boldsymbol{\xi} \cdot \boldsymbol{\nabla} f(\boldsymbol{\xi}, \boldsymbol{x}, t) = Q(f). \tag{2.5}$$

Here, $f(\boldsymbol{\xi}, \boldsymbol{x}, t)\, \mathrm{d}\boldsymbol{\xi}\, \mathrm{d}\boldsymbol{x}$ (for infinitesimal small $\mathrm{d}\boldsymbol{\xi}$ and $\mathrm{d}\boldsymbol{x}$) gives the number density, multiplied by particle mass m, of a particle having the velocity $\boldsymbol{\xi} \in \mathbb{R}^d$ at position $\boldsymbol{x} \in \mathbb{R}^d$ and time $t \in \mathbb{R}$. The $Q(f)$ on the right hand side gives the contribution from collisions

of particles under the assumption stated above. In equation (2.5) the particle velocity $\boldsymbol{\xi} \in \mathbb{R}^d$ appears as a parameter, hence the Boltzmann equation (2.5) is formally an infinite system of equations. The collision is modeled as an elastic collision of two particles, such that kinetic energy and momenta are conserved. The collision integral due to Boltzmann reads in three-dimensional space ($d = 3$)

$$Q(f) = \int_{\mathbb{R}^3} \int_{\mathbb{S}^2} \sigma(\Omega) \left|\boldsymbol{\xi} - \boldsymbol{\xi}_2\right| \left[f(\bar{\boldsymbol{\xi}}, \boldsymbol{x}, t) f(\bar{\boldsymbol{\xi}}_2, \boldsymbol{x}, t) - f(\boldsymbol{\xi}, \boldsymbol{x}, t) f(\boldsymbol{\xi}_2, \boldsymbol{x}, t)\right] \mathrm{d}\Omega \, \mathrm{d}\boldsymbol{\xi}_2, \tag{2.6}$$

with pre-collision velocities $\boldsymbol{\xi}$, $\boldsymbol{\xi}_2$. The post-collision velocities $\bar{\boldsymbol{\xi}}$, $\bar{\boldsymbol{\xi}}_2$ can be computed depending on the pre-collision velocities and the impact angle [11]. Moreover in (2.6), $\sigma(\Omega)$ denotes the differential collision cross section and the inner integration is done over all possible solid angles $\Omega \in \mathbb{S}^2$. For (2.6) there exist five elementary collision invariants [10]

$$\psi_1 = 1, \quad (\psi_2, \psi_3, \psi_4) = \boldsymbol{\xi} \quad \text{and} \quad \psi_5 = |\boldsymbol{\xi}|^2, \tag{2.7a}$$

which are classified by vanishing integrals as follows:

$$\int_{\mathbb{R}^d} \psi_k Q(f) \, \mathrm{d}\boldsymbol{\xi} = 0, \quad k = 1, \ldots, 5. \tag{2.7b}$$

As for the macroscopic formulation, several techniques for directly solving the Boltzmann equation exist [1, 11, 64]. It is important to understand that the single particle distribution contains all information which on a macroscopic level are expressed with several dynamic quantities. Therefore, the sole knowledge of only a few macroscopic dynamic quantities does not correspond to a unique value of the single particle distribution. Conversely, the information of $f(\cdot, \boldsymbol{x}, t)$ (for all $\boldsymbol{\xi}$) is sufficient to uniquely determine a pointwise (at $\boldsymbol{x}$) corresponding macroscopic quantity at time t. We explain this connection in more detail in what follows next.

2.2 Boltzmann equation and its macroscopic limit

The system of NSE is derived by conservation considerations of physical quantities. The same holds for the Euler equations. They both are widely accepted descriptions for fluid flows. In the following, we link the Boltzmann equation and the NSE [92].

2.2.1 Moments of the single particle distribution

Given the Boltzmann equation (2.5), we consider the single particle distribution $f(\boldsymbol{\xi}, \boldsymbol{x}, t)$. Based on physical interpretation we obtain the fluid's (local) mass density by the integral of the single particle distribution with respect to all possible velocities [51, 92]:

$$\rho(\boldsymbol{x}, t) = \int_{\mathbb{R}^d} f(\boldsymbol{\xi}, \boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{\xi}. \tag{2.8}$$

This corresponds to the zeroth moment of the single particle distribution. Furthermore, the (local) fluid velocity is given as an average of particle velocities

$$\boldsymbol{u}(\boldsymbol{x},t) = \frac{1}{\rho(\boldsymbol{x},t)} \int_{\mathbb{R}^d} \boldsymbol{\xi} f(\boldsymbol{\xi},\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{\xi}. \tag{2.9}$$

In fact, the first moment of f, integral in (2.9), is required to determine the fluid velocity. Higher moments can also be interpreted physically. Here only the moments which are used in this work are presented, remaining moments can be found, e.g., in [51, 102]. The internal energy of the fluid is derived from the mesoscopic level by

$$e(\boldsymbol{x},t) = \frac{1}{\rho(\boldsymbol{x},t)} \int_{\mathbb{R}^d} \frac{|\boldsymbol{\xi} - \boldsymbol{u}(\boldsymbol{x},t)|^2}{2} f(\boldsymbol{\xi},\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{\xi}, \tag{2.10}$$

and the total stress tensor is given as

$$\boldsymbol{p}(\boldsymbol{x},t) = \int_{\mathbb{R}^d} (\boldsymbol{\xi} - \boldsymbol{u})\,(\boldsymbol{\xi} - \boldsymbol{u})^\top f(\boldsymbol{\xi},\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{\xi}. \tag{2.11}$$

A common way to identify the pressure $p(\boldsymbol{x},t)$ is to take the isotropic part of the total stress $\boldsymbol{p}$, i.e., the trace of $\boldsymbol{p}$ divided by dimension d:

$$p(\boldsymbol{x},t) = \frac{1}{d} \operatorname{tr}\big(\boldsymbol{p}(\boldsymbol{x},t)\big) = \frac{1}{d} \int_{\mathbb{R}^d} |\boldsymbol{\xi} - \boldsymbol{u}(\boldsymbol{x},t)|^2 f(\boldsymbol{\xi},\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{\xi}. \tag{2.12}$$

With the pressure p given, the deviatoric stress tensor $\boldsymbol{\sigma}$ can be computed by

$$\boldsymbol{\sigma}(\boldsymbol{x},t) = p(\boldsymbol{x},t)\boldsymbol{I} - \boldsymbol{p}(\boldsymbol{x},t). \tag{2.13}$$

2.2.2 Moments of the Boltzmann equation

So far, all dynamic quantities of the NSE (2.3) are defined in terms of the single particle distribution (2.8)–(2.13). From the Boltzmann equation (2.5) we have the information of the evolution of the single particle distribution. This then implies an evolution for each macroscopic quantity (2.8)–(2.13). We achieve the corresponding macroscopic evolution equations by computing moments of the Boltzmann equation (2.5). The zeroth and first moment bring the macroscopic fluid density (2.8), velocity (2.9) and the stress tensor (2.11) into a system of differential equations [92]:

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot \rho\boldsymbol{u} = \int_{\mathbb{R}^d} Q(f)\,\mathrm{d}\boldsymbol{\xi}, \tag{2.14a}$$

$$\frac{\partial \rho\boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot \left(\rho\boldsymbol{u}\boldsymbol{u}^\top\right) + \boldsymbol{\nabla} \cdot \boldsymbol{p} = \int_{\mathbb{R}^d} \boldsymbol{\xi} Q(f)\,\mathrm{d}\boldsymbol{\xi}. \tag{2.14b}$$

The system (2.14) resembles the compressible NSE (2.3), and due to (2.7) both systems even match. The equation of state for the system (2.14) is achieved by combining (2.10)

and (2.12) to

$$p(\boldsymbol{x},t) = \frac{2}{d}\rho(\boldsymbol{x},t)e(\boldsymbol{x},t). \tag{2.15}$$

We point out that properties of the collision term (2.6) are crucial for the system (2.14) to agree with the NSE (2.3). A Boltzmann-like equation, where the collision term $Q(f)$ is replaced by a different model $\tilde{Q}(f)$, is called a *kinetic model*. So any kinetic model can lead to the NSE provided the required properties of $Q(f)$ remain valid also for $\tilde{Q}(f)$. In detail we require the conservation conditions of the collision term (2.6) as

$$\int_{\mathbb{R}^d} \psi_k Q(f)\,\mathrm{d}\boldsymbol{\xi} = 0, \quad k = 1,\dots,4. \tag{2.16}$$

Two other important properties which are satisfied by the collision integral (2.6) are related to Maxwellian distributions [10]

$$M(\boldsymbol{\xi};\rho,\boldsymbol{u},T) := \rho\left(\frac{m}{2\pi k_B T}\right)^{d/2} \exp\left(\frac{-m}{2k_B T}\left|\boldsymbol{\xi}-\boldsymbol{u}\right|^2\right). \tag{2.17}$$

In (2.17) k_B denotes the Boltzmann constant and T is the fluid temperature. A Maxwellian describes an equilibrium distribution for the Boltzmann equation, where an equilibrium is defined as a homogeneous steady state. On the one hand, the collision integral vanishes for any Maxwellian distribution, i.e.,

$$Q(M) = 0. \tag{2.18}$$

On the other hand, we have that the collision term satisfies

$$\int_{\mathbb{R}^d} \log\big(f(\boldsymbol{\xi},\boldsymbol{x},t)\big)Q(f)\,\mathrm{d}\boldsymbol{\xi} \le 0, \tag{2.19}$$

with equality holding if, and only if, f is Maxwellian. The latter equation is essential for Boltzmann's famous H-theorem, which is in accordance with the second law of thermodynamics [10]. Thus for any kinetic model an analogue to Boltzmann's H-theorem has to be valid. Next, we present one collision alternative for $Q(f)$ satisfying all required properties.

2.2.3 Alternative collision model

If the collision term (2.6) is replaced by another (maybe simpler) model $\tilde{Q}(f)$, the conservation conditions (2.16) and properties (2.18) and (2.19) have to be retained. The second property (2.19) expresses that in a closed system any initial single particle distribution $f(\boldsymbol{\xi},\boldsymbol{x},t)$ describing the state of a fluid tends locally to a Maxwellian distribution [10]. This tendency towards a Maxwellian can be easily taken into account. The effect of collision has to change any particle distribution f by an amount which is proportional to the discrepancy of f from a Maxwellian. The collision model of Bhatnagar, Gross and Krook (BGK) [5] realizes this principle by

$$\tilde{Q}(f) := -\frac{1}{\tau}\big[f(\boldsymbol{\xi},\boldsymbol{x},t) - M(\boldsymbol{\xi};\rho,\boldsymbol{u},T)\big]. \tag{2.20}$$

Here τ is a constant, called the relaxation parameter of the collision model. The BGK model is a *single-relaxation-time* (SRT) model. We refer to [10] for a validation that $\tilde{Q}(f)$ satisfies (2.19). In order that the BGK collision model satisfies the conservation conditions (2.16) the three dynamic quantities of the Maxwellian in (2.20) have to be fixed by moments of f, see (2.8) to (2.10). Note that the temperature T can be computed from the energy. For a monatomic gas we have the relation

$$e = \frac{d}{2} RT, \tag{2.21}$$

where $R = \frac{k_b}{m}$ is the specific gas constant [10]. Obviously the property (2.18) is satisfied by the BGK collision model. The full kinetic model with the BGK collision model then reads

$$\frac{\partial f(\boldsymbol{\xi}, \boldsymbol{x}, t)}{\partial t} + \boldsymbol{\xi} \cdot \boldsymbol{\nabla} f(\boldsymbol{\xi}, \boldsymbol{x}, t) = -\frac{1}{\tau} \left[f(\boldsymbol{\xi}, \boldsymbol{x}, t) - M(\boldsymbol{\xi}; \rho, \boldsymbol{u}, T) \right]. \tag{2.22}$$

2.3 Lattice Boltzmann method

The LBM is nowadays seen as a certain discretization of the Boltzmann equation or of another kinetic model. However, historically, it originates from cellular automata, and specifically lattice gas automata. A cellular automaton is a discrete model introduced by von Neumann (see [117]), which has simple rules but allows for a more complex behavior [66]. Lattice gas automata were derived from classical cellular automata with the intention of an application to physical processes [120]. The first proposed lattice gas automaton of Hardy et al. [38] still could not be used to correctly simulate fluid flows. Whereas the famous FHP-model of Frisch, Hasslacher and Pomeau [30] introduced in 1986, could overcome the difficulties of previous lattice gas automata. Hence, starting with the introduction of the FHP-model, several lattice gas automata governing the NSE were proposed for different kind of problems [123]. As only one example we mention a lattice gas automaton for a simulation of flows through porous media [14]. Very soon after the FHP-model was proposed, in 1988, a modification to lattice gas automata was published from McNamara and Zanetti [74]. Hereby the boolean variables of lattice gas automata were changed into real variables representing averaged particle distributions on a mesoscopic level. This work is nowadays considered as the first article on the LBM. Four years later, in 1992, a BGK-based collision model for the LBM was proposed to recover the Navier-Stokes macroscopic equations [13, 86].

An interpretation as a discretization of the Boltzmann equation was found five years later, thus not until after the introduction of the method [43]. For a formal derivation of the LBM we consider here the kinetic model (2.22), that is the Boltzmann equation with a BGK collision model (2.20) instead of the original collision term. We refer to this equation (not quite correctly) as the *BGK-Boltzmann equation*. It is integrated along its characteristic curves and the outcome is approximated. Afterwards the still continuous velocity space is discretized, which introduces so-called *lattice vectors*. We here present the derivation of the D3Q19 velocity model by numerical integration conditions (Section 2.3.3), which (to our best knowledge) has been worked out for the first time in literature in our work [45]. Before the integration of (2.22) is performed, we give a general summary of characteristics for hyperbolic differential equations.

2.3.1 Characteristic curves

The description of characteristics follows the one-dimensional approach of [69]. At first, we consider the Cauchy problem of a general advection equation:

$$\frac{\partial z}{\partial t} + \boldsymbol{a} \cdot \boldsymbol{\nabla} z = 0, \qquad z(\boldsymbol{x}, 0) = z_0(\boldsymbol{x}). \tag{2.23}$$

Here we seek $z = z(\boldsymbol{x}, t) : \mathbb{R}^d \times \mathbb{R}^+ \to \mathbb{R}$, where $z_0 : \mathbb{R}^d \to \mathbb{R}$ denotes the initial profile and $\boldsymbol{a} \in \mathbb{R}^d$ is a given advection velocity vector. The unique solution of (2.23) for $t \geq 0$ is given as

$$z(\boldsymbol{x}, t) = z_0(\boldsymbol{x} - \boldsymbol{a}t),$$

thus the initial data propagates in a direction defined by $\boldsymbol{a}$, while the norm of $\boldsymbol{a}$ gives the propagation speed. If the solution is sketched in a t-$\boldsymbol{x}$-plane, we would see that the solution is constant along each line $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{a}t$, for any $\boldsymbol{x}_0 \in \mathbb{R}^d$. These rays are called the *characteristics* (or *characteristic curves*) of the equation (2.23). The characteristics are solutions to the ordinary differential equations (ODEs)

$$\frac{\partial \boldsymbol{x}}{\partial t} = \boldsymbol{a}, \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0. \tag{2.24}$$

For a space dependent $\boldsymbol{a} = \boldsymbol{a}(\boldsymbol{x})$ in (2.23), the characteristics are not rays, but curves, which explains the naming of *characteristic curves*. Along them the solution z satisfies an ODE which can be solved easily.

Now, let $\boldsymbol{z}$ be a vector valued function $\boldsymbol{z} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^n$. We consider the linear system

$$\frac{\partial \boldsymbol{z}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{z}}{\partial x} = 0, \qquad \boldsymbol{z}(x, t) = \boldsymbol{z}_0(x), \tag{2.25}$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is a constant diagonalizable matrix with real eigenvalues $\lambda_1, \ldots, \lambda_n$. In this case (2.25) is a hyperbolic system and we can find a matrix $\boldsymbol{B}$, such that

$$\boldsymbol{B}\boldsymbol{A} = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)\boldsymbol{B}.$$

By a (left-)multiplication with $\boldsymbol{B}$ the system (2.25) is decoupled. Introducing so-called characteristic variables $\bar{\boldsymbol{z}} = \boldsymbol{B}\boldsymbol{z}$, we get

$$\frac{\partial \bar{\boldsymbol{z}}}{\partial t} + \operatorname{diag}(\lambda_1, \ldots, \lambda_n)\frac{\partial \bar{\boldsymbol{z}}}{\partial x} = 0. \tag{2.26}$$

This is a decoupled system of n equations of type (2.23) (for $d = 1$), where the directions of advection are determined by the eigenvalues of $\boldsymbol{A}$.

2.3.2 Integration of the kinetic model along its characteristics

The BGK-Boltzmann equation (2.22) is of the same form as (2.23), apart from the non-zero right hand side. Along its characteristic curves the BGK-Boltzmann equation satisfies

the ODE

$$\frac{\mathrm{d}}{\mathrm{d}t} f(\boldsymbol{\xi}, \boldsymbol{x}, t) + \frac{1}{\tau} f(\boldsymbol{\xi}, \boldsymbol{x}, t) = \frac{1}{\tau} M(\boldsymbol{\xi}; \rho, \boldsymbol{u}, T), \tag{2.27}$$

with the advective differential operator $\frac{\mathrm{d}}{\mathrm{d}t} = \frac{\partial}{\partial t} + \boldsymbol{\xi} \cdot \boldsymbol{\nabla}$. Note that (2.22), as well as the Boltzmann equation (2.5) and any kinetic model, is an infinite set of equations, one for each $\boldsymbol{\xi} \in \mathbb{R}^d$. Hence, the ODE (2.27) is an equation only for one fixed velocity $\boldsymbol{\xi}$, which still enters as a parameter.

For the integration of (2.27), we follow the strategy from He and Luo [43]. The Maxwellian on the right hand side of (2.27) depends on space and time only via the macroscopic dynamic quantities ρ, $\boldsymbol{u}$ and T. To underline this space and time dependency we rewrite the Maxwellian by introducing an equilibrium distribution f^{eq} as

$$f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t) := M\big(\boldsymbol{\xi}; \rho(\boldsymbol{x}, t), \boldsymbol{u}(\boldsymbol{x}, t), T(\boldsymbol{x}, t)\big). \tag{2.28}$$

Formally, the solution of (2.27) at $t + \Delta t$ for an arbitrary time step $\Delta t \geq 0$ reads

$$\begin{aligned} &f(\boldsymbol{\xi}, \boldsymbol{x} + \boldsymbol{\xi}\Delta t, t + \Delta t) \\ &\quad = \frac{1}{\tau} \exp\left(-\frac{\Delta t}{\tau}\right) \int_0^{\Delta t} \exp\left(\frac{s}{\tau}\right) f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x} + \boldsymbol{\xi} s, t + s)\, \mathrm{d}s + \exp\left(-\frac{\Delta t}{\tau}\right) f(\boldsymbol{\xi}, \boldsymbol{x}, t). \end{aligned} \tag{2.29}$$

Next, the integrand is approximately substituted by $f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t)$, which arises from a Taylor expansion (around $s = 0$) and neglect of terms of order $\mathcal{O}(s)$:

$$\exp\left(\frac{s}{\tau}\right) f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x} + \boldsymbol{\xi} s, t + s) = f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t) + \mathcal{O}(s).$$

The other two exponentials appearing in (2.29) are substituted by their Taylor series. Neglecting all terms of order $\mathcal{O}(\Delta t^2)$ yields the following approximation of the BGK-Boltzmann equation as a time discrete formulation:

$$f(\boldsymbol{\xi}, \boldsymbol{x} + \boldsymbol{\xi}\Delta t, t + \Delta t) - f(\boldsymbol{\xi}, \boldsymbol{x}, t) = -\frac{\Delta t}{\tau} \big[f(\boldsymbol{\xi}, \boldsymbol{x}, t) - f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t) \big]. \tag{2.30}$$

For the approximations which led to (2.30), the Maxwellian is assumed to be locally smooth enough in space and time. Also the time step size Δt is assumed to be chosen sufficiently small. Note that in (2.30) space $\boldsymbol{x}$ and velocity $\boldsymbol{\xi}$ are still continuous. Another very common notation, which is also used in the sequel, replaces the right hand side coefficient by a relaxation time $\omega = \frac{\Delta t}{\tau}$.

2.3.3 Analytical derivation of the D3Q19 velocity set

In the following we derive a discretization of the velocity space for (2.30). A velocity space discretization implies a spatial discretization and we end up with a fully discretized equation, which is the core equation of the LBM. Here, we derive the discrete velocities used in the D3Q19 model, that is a 19 velocity model in three space dimensions. In general, the common notation DdQq by Qian et al. [86] represents a d-dimensional model with q discrete velocities. For the derivation we follow the procedure of He and Luo [43], in which the D3Q27 model is derived. In the derivation of D3Q27 the three space dimensions are

separated and can be considered independently of each other. Since such a separation is not possible for D3Q19, this makes the derivation here more complex.

In the context of the LBM the collision model and especially the equilibrium distribution used in the BGK model is essential. In (2.30) the equilibrium (2.28) is a Maxwellian, which we approximate by a truncated Taylor series up to order $\mathcal{O}(\boldsymbol{u}^2)$:

$$f^{\mathrm{eq}} \approx \rho \left(\frac{1}{2\pi RT}\right)^{d/2} \exp\left(-\frac{|\boldsymbol{\xi}|^2}{2RT}\right) \left[1 + \frac{\langle \boldsymbol{\xi}, \boldsymbol{u}\rangle}{RT} + \frac{\langle \boldsymbol{\xi}, \boldsymbol{u}\rangle^2}{2(RT)^2} - \frac{|\boldsymbol{u}|^2}{2RT}\right], \tag{2.31}$$

with the inner product $\langle \boldsymbol{\xi}, \boldsymbol{u}\rangle$. We made the substitution $R = \frac{k_b}{m}$ in contrast to the definition of a Maxwellian in (2.17). It is easily possible to incorporate higher order terms in the previous so-called *low-Mach-number approximation*. In what follows this truncated expansion shall be used as the equilibrium distribution instead of the full Maxwellian (2.17).

The derivation of the discrete velocity space is based on a numerical consideration of integrals in the form

$$I(\Psi) = \int_{\mathbb{R}^d} \exp\left(-\frac{|\boldsymbol{\xi}|^2}{2RT}\right) \Psi(\boldsymbol{\xi}) \,\mathrm{d}\boldsymbol{\xi}, \tag{2.32}$$

where the exponential function is coming from the equilibrium distribution. The nodes and weights of a quadrature rule for a numerical computation of these integrals are derived from the condition that polynomials $\Psi(\boldsymbol{\xi})$ up to a certain maximal degree d_p shall be integrated exactly. The term within in the square brackets of the equilibrium distribution (2.31) is a polynomial of second degree in $\boldsymbol{\xi}$. Thus, it follows $d_p \geq 2$, in order to integrate the equilibrium distribution exactly. As the ψ_k in the required conservation conditions (2.16) are themselves polynomials up to order one, we even have $d_p \geq 3$. Besides moments of the equilibrium distribution also moments (up to the second order) of the unknown single particle distribution f are required to obtain the macroscopic conservation equations (2.14), which further increases d_p as can be seen below. The problem that the single particle distribution is unknown is addressed, e.g., by using a Chapman-Enskog expansion [12]

$$f(\boldsymbol{\xi}, \boldsymbol{x}, t) = f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t) + \mathrm{Kn}\, f^{(1)}(\boldsymbol{\xi}, \boldsymbol{x}, t) + \mathrm{Kn}^2 f^{(2)}(\boldsymbol{\xi}, \boldsymbol{x}, t) + \mathcal{O}(\mathrm{Kn}^3).$$

The first order term in the Chapman-Enskog analysis is approximately given by

$$f^{(1)}(\boldsymbol{\xi}, \boldsymbol{x}, t) \approx -\tau \left(\frac{\partial^{(1)}}{\partial t} + \boldsymbol{\xi} \cdot \boldsymbol{\nabla}^{(1)}\right) f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t).$$

Also higher order terms ($f^{(k)},\ k \geq 2$), can be expressed approximately in terms of the equilibrium distribution. However the conservation equations (2.14) can be derived by only taking f^{eq} and $f^{(1)}$ into account. Hereby, the first order term $f^{(1)}$ is necessary to obtain the viscous terms in the conservation equations (2.14). A more detailed view is achieved when performing a Chapman-Enskog analysis, see, e.g., [15].

Quadrature rule for D3Q19 With the aforementioned explanations, the computation of a second moment of the unknown single particle distribution is equivalent to the computation of a third moment of the equilibrium distribution. In total, the condition to a quadrature rule for the integrals (2.32) reads to be exact for polynomials Ψ of degree at least up to order 5, i.e., $d_p \geq 5 = 3 + 2$. Hereby 3 is the order of the highest required moment and 2 results from the low-Mach-number approximation. In other words, as the integral is a linear operator, we can consider integrals

$$\begin{aligned} I_{m,n,p} &= \left(\frac{1}{2\pi RT}\right)^{3/2} \int\limits_{\mathbb{R}^3} \exp\left(-\frac{|\boldsymbol{\xi}|^2}{2RT}\right) \xi_\alpha^m \xi_\beta^n \xi_\gamma^p \,\mathrm{d}\boldsymbol{\xi} \\ &= \pi^{-3/2} (2RT)^{\frac{m+n+p}{2}} \int\limits_{\mathbb{R}^3} \exp\left(-|\boldsymbol{\zeta}|^2\right) \zeta_\alpha^m \zeta_\beta^n \zeta_\gamma^p \,\mathrm{d}\boldsymbol{\zeta}, \end{aligned}$$

for $m, n, p \in \mathbb{N}_0^+$, where equality of a quadrature rule is required for all choices satisfying $m + n + p \leq 5$. Note that the first factor arises from the equilibrium distribution and we have explicitly set $d = 3$. We take the following ansatz for the quadrature rule $Q_{m,n,p}$

$$I_{m,n,p} \approx Q_{m,n,p} := \pi^{-3/2} (2RT)^{\frac{m+n+p}{2}} \sum_{(j,k,l)\in\mathcal{J}} w_{j,k,l}\, \zeta_j^m\, \zeta_k^n\, \zeta_l^p,$$

with weights $w_{j,k,l}$ and nodes ζ_i. To obtain the velocity set of D3Q19, the index set $\mathcal{J}$ shall be given by

$$\mathcal{J} = \Big\{(j,k,l) \in \{-1,0,1\}^3 \mid j = 0 \vee k = 0 \vee l = 0\Big\}. \tag{2.33}$$

The or-operator ($\vee$) in the definition of $\mathcal{J}$ is seen to be non-exclusive. The solution for $Q_{m,n,p}$ under the given requirements has nodes of a Gauß-Hermite quadrature [87]

$$\zeta_{-1} = -\sqrt{3/2}, \qquad \zeta_0 = 0, \qquad \zeta_1 = \sqrt{3/2},$$

and corresponding weights as follows

$$w_{j,k,l} = \begin{cases} \frac{\pi^{3/2}}{3} & \text{for } (j,k,l) = (0,0,0), \\ \frac{\pi^{3/2}}{18} & \text{for } (j,k,l) \in \Big\{(\pm1,0,0),(0,\pm1,0),(0,0,\pm1)\Big\}, \\ \frac{\pi^{3/2}}{36} & \text{otherwise.} \end{cases}$$

The quadrature rule for the integrals (2.32) is thereby fully given, and we can proceed with deriving the discrete velocities for the D3Q19 model.

Lattice vectors of D3Q19 A discrete set of velocities

$$\mathcal{V} = \big\{\boldsymbol{c}_i \in \mathbb{R}^3 \mid i = 0, \ldots, q-1\big\} \tag{2.34}$$

is obtained from the quadrature rule by constructing vectors $\boldsymbol{c}_i = \sqrt{2RT}(\zeta_j, \zeta_k, \zeta_l)^\top$. Each index i corresponds to a combination of indices j, k, l, i.e., to an element of index set $\mathcal{J}$. Also new weights are constructed by the same mapping $w_i = \frac{1}{\pi^{3/2}} w_{j,k,l}$. Note that $\mathcal{J}$ is a set of 19 vectors, therefore $q = 18$ and the discrete velocity set $\mathcal{V}$ of D3Q19 is therefore

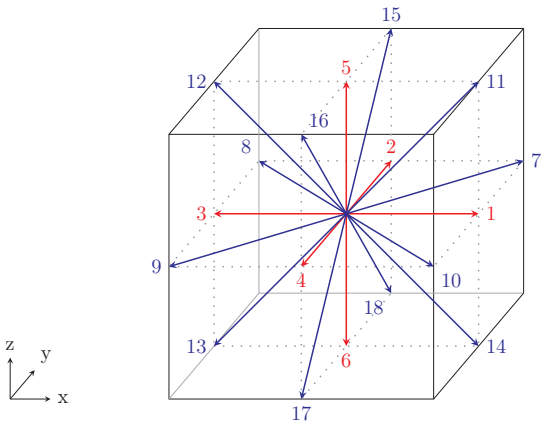


Figure 2.1: *Lattice vectors of D3Q19* – Illustration of the lattice vectors (2.36) of the D3Q19 model. Vectors shown in red are unitary, while vectors shown in blue satisfy $\|\boldsymbol{c}_j\| = \sqrt{2}c$.

defined. By introducing

$$c = \sqrt{3RT} \tag{2.35}$$

these discrete velocities, also called *lattice vectors*, in D3Q19 are

$$\begin{aligned} \boldsymbol{c}_0 &= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \qquad \boldsymbol{c}_{1-6} = \left\{ \begin{pmatrix} \pm c \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \pm c \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ \pm c \end{pmatrix} \right\}, \\ \boldsymbol{c}_{7-18} &= \left\{ \begin{pmatrix} \pm c \\ \pm c \\ 0 \end{pmatrix}, \begin{pmatrix} \pm c \\ 0 \\ \pm c \end{pmatrix}, \begin{pmatrix} 0 \\ \pm c \\ \pm c \end{pmatrix} \right\}. \end{aligned} \tag{2.36}$$

See also Fig. 2.1 for a visual depiction. For the chosen assignment from $\mathcal{J}$ to an index i the corresponding weights w_i read

$$w_0 = \frac{1}{3}, \qquad w_{1-6} = \frac{1}{18} \qquad \text{and} \qquad w_{7-18} = \frac{1}{36}.$$

Thus with the discrete velocities (2.36), any moment of the equilibrium distribution is approximately given as

$$\int_{\mathbb{R}^3} \psi(\boldsymbol{\xi}) f^{\mathrm{eq}}(\boldsymbol{\xi}, \boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{\xi} \approx \sum_{j=0}^{18} \psi(\boldsymbol{c}_j) \rho w_j \left[1 + \frac{3\langle \boldsymbol{c}_j, \boldsymbol{u} \rangle}{c^2} + \frac{9\langle \boldsymbol{c}_j, \boldsymbol{u} \rangle^2}{2c^4} - \frac{3\,|\boldsymbol{u}|^2}{2c^2} \right]. \tag{2.37}$$

By construction, the quadrature rule is exact for polynomials ψ up to third degree.

The *lattice Boltzmann* (LB) equation can be formulated now by combining the time discrete BGK-Boltzmann equation (2.30) and the discrete lattice velocities. A short presentation of other very common discrete velocity sets can be found in Appendix A.

2.3.4 Lattice Boltzmann equation

The velocity $\boldsymbol{\xi}$ in the time discrete BGK-Boltzmann equation (2.30) is treated like a free parameter. In particular, (2.30) represents an individual equation for each fixed $\boldsymbol{\xi} \in \mathbb{R}^d$, thus (2.30) describes an infinite system. We use the discrete velocities to obtain a finite number of equations and a fully discretized scheme, since a space discretization is implied:

$$\Delta \boldsymbol{x} = \boldsymbol{c}_i \Delta t. \tag{2.38}$$

It is important to note, that an expression like (2.37) also holds for other velocity models (see Appendix A). From (2.37) it is obvious to define a discrete equilibrium distribution as follows

$$f_i^{\mathrm{eq}}(\boldsymbol{x}, t) = \rho w_i \left[1 + \frac{3\langle \boldsymbol{c}_i,\, \boldsymbol{u} \rangle}{c^2} + \frac{9\langle \boldsymbol{c}_i,\, \boldsymbol{u} \rangle^2}{2c^4} - \frac{3\,|\boldsymbol{u}|^2}{2c^2} \right], \tag{2.39}$$

where the weights for common discrete velocity models are given in Table 2.1 [103]. With help of the low-Mach-number approximation (2.31) we can conclude (see also [42])

$$f_i^{\mathrm{eq}}(\boldsymbol{x}, t) = (2\pi RT)^{d/2} \exp\left(\frac{|\boldsymbol{c}_i|^2}{2RT} \right) w_i f^{\mathrm{eq}}(\boldsymbol{c}_i, \boldsymbol{x}, t) + \mathcal{O}(\boldsymbol{u}^3).$$

Similarly a discrete analogy of the single particle distribution, so-called *populations*, are defined by

$$f_i(\boldsymbol{x}, t) := (2\pi RT)^{d/2} \exp\left(\frac{|\boldsymbol{c}_i|^2}{2RT} \right) w_i f(\boldsymbol{c}_i, \boldsymbol{x}, t).$$

Using the discrete distributions, the BGK-Boltzmann equation (2.30) reduces to a finite system of q equations as follows

$$f_i(\boldsymbol{x} + \boldsymbol{c}_i \Delta t, t + \Delta t) = \tilde{f}_i(\boldsymbol{x}, t) := f_i(\boldsymbol{x}, t) + \mathrm{C}_{\mathrm{BGK},i}, \tag{2.40}$$

with the discrete BGK collision model

$$\mathrm{C}_{\mathrm{BGK},i} := -\omega \big[f_i(\boldsymbol{x}, t) - f_i^{\mathrm{eq}}(\boldsymbol{x}, t) \big]. \tag{2.41}$$

Equation (2.40) is known as the *lattice Boltzmann equation* (LBE). Often the term *lattice BGK equation* is used, to emphasize the collision model. In the discrete variant the macroscopic dynamic quantities, originally defined in (2.8) and (2.9) by moments of the single particle distribution, are computed via

$$\rho(\boldsymbol{x}, t) = \sum_{j=0}^{q-1} f_j(\boldsymbol{x}, t), \qquad \boldsymbol{u}(\boldsymbol{x}, t) = \frac{1}{\rho(\boldsymbol{x}, t)} \sum_{j=0}^{q-1} \boldsymbol{c}_j f_j(\boldsymbol{x}, t). \tag{2.42}$$

We note that c is fixed in the LBM, meaning the temperature T does not have to be computed. Thus, the LBM simulates isothermal flows of fluids. For later use we note that the local computation of $\tilde{f}_i(\boldsymbol{x}, t)$ in (2.40) is commonly known as the *collision step*. The assignment to the left hand side, i.e., the movement to adjacent grid points and the temporal progress, is called *streaming step* (also *transport step*).

Table 2.1: *Weights of the discrete equilibrium distribution* – Values of the weights w_j in the discrete equilibrium distribution (2.39) for several discretization models.

	weights for			
	$\|\boldsymbol{c}_j\| = 0$	$\|\boldsymbol{c}_j\| = c$	$\|\boldsymbol{c}_j\| = \sqrt{2}c$	$\|\boldsymbol{c}_j\| = \sqrt{3}c$
D1Q3	$\frac{2}{3}$	$\frac{1}{6}$	-	-
D2Q7	$\frac{1}{2}$	$\frac{1}{12}$	-	-
D2Q9	$\frac{4}{9}$	$\frac{1}{9}$	$\frac{1}{36}$	-
D3Q15	$\frac{2}{9}$	$\frac{1}{9}$	-	$\frac{1}{72}$
D3Q19	$\frac{1}{3}$	$\frac{1}{18}$	$\frac{1}{36}$	-
D3Q27	$\frac{8}{27}$	$\frac{2}{27}$	$\frac{1}{54}$	$\frac{1}{216}$

Although first order approximations have been used above, the LBM is second order in space and time [101], since the discretization error can be absorbed into the viscosity term, such that it reads:

$$\nu = \frac{\Delta t}{3}\left(\frac{1}{\omega} - \frac{1}{2}\right)c^2. \tag{2.43}$$

An asymptotic analysis, like a Chapman-Enskog expansion [12], shows that the fluid quantities (2.42) evolve according to the compressible NSE. Moreover we remark that the collision model principally determines the macroscopic behavior. So, for example the incompressible NSE can be recovered when the equilibrium distribution in the BGK model (2.41) is slightly changed, see, e.g., [41]. Another example can be found below in Section 2.4.1. Without going into details, we also point out the existence of other collision models than the BGK operator. Popular alternatives are the *multiple-relaxation-time* (MRT) model [21] and the *two-relaxation-time* (TRT) model [32]. For discussion of these models, their advantages and other alternatives, we refer to literature. A nice first overview of several further enhancements is given in [104], it contains, e.g., the incorporation of forces and the simulation of multiple phases/components [36, 95, 96].

2.4 Lattice Boltzmann beyond Navier-Stokes equations

In previous sections the LBM was considered solely as a scheme for solving the NSE. The aim of the current section is to demonstrate that the LBM allows for more applications than classic fluid flows described by NSE. Above we have hinted at the influence of the equilibrium distribution to the macroscopic expression. Regarding this, we go into more detail in Section 2.4.1 and demonstrate that a simple one-dimensional lattice Boltzmann model can be constructed to achieve mainly an advection equation. This model will later be the central component of the considerations in Chapter 4. In Section 2.4.2 we further demonstrate that the LBM is not limited to isothermal NSE by discussing thermodynamical flows. Moreover we list some further applications of the LBM, showing that the method has a wide range of applications.

2.4.1 Advection equation with the D1Q2 model

We consider the lattice BGK equation (2.40) for the one-dimensional discretization $c_1 = -c$ and $c_2 = c$ (D1Q2). This model is a restriction of the D1Q3 discretization, where the rest velocity of D1Q3 has been neglected. Hence, to simplify the comparison between both models (in upcoming chapters) we enumerate the discrete velocities starting with index 1 for D1Q2. For the sake of simplicity we assume $c = 1$ in the following. We remark that based on D1Q2 no model can be constructed, which would recover the NSE. Here, in the lattice BGK equation we take a simple discrete equilibrium distribution as follows

$$f_i^{\mathrm{eq}}(x,t) = \frac{1}{2}\rho(x,t)\left(1 + ac_i\right), \tag{2.44}$$

with a parameter $a \in (-1,1)$ and the local quantity ρ defined by the zeroth moment:

$$\rho = f_1 + f_2. \tag{2.45}$$

Next, we use a Chapman-Enskog analysis [12] to obtain the macroscopic evolution equation for the given model. The grid and time spacing shall be given by $h > 0$.

Chapman Enskog expansion for D1Q2 For the Chapman-Enskog analysis the populations are expanded

$$f_i = f_i^{(0)} + hf_i^{(1)} + h^2 f_i^{(2)} + h^3 f_i^{(3)} + \mathcal{O}(h^4), \tag{2.46}$$

where $f_i^{(0)} = f_i^{\mathrm{eq}}$ is the local equilibrium distribution (2.44). We omit the arguments of the populations throughout the Chapman-Enskog analysis. The operators are formally expanded as

$$\frac{\partial}{\partial t} = \frac{\partial^{(0)}}{\partial t} + h\frac{\partial^{(1)}}{\partial t} + h^2\frac{\partial^{(2)}}{\partial t}, \qquad \frac{\partial}{\partial x} = \frac{\partial^{(0)}}{\partial x}. \tag{2.47}$$

The Chapman-Enskog analysis begins with a Taylor series expansion of the left hand side of (2.40) ($\Delta t = h$), resulting in

$$\begin{aligned} h\left(\frac{\partial}{\partial t} + c_i\frac{\partial}{\partial x}\right)f_i + \frac{h^2}{2}\left(\frac{\partial}{\partial t} + c_i\frac{\partial}{\partial x}\right)^2 f_i \\ + \frac{h^3}{6}\left(\frac{\partial}{\partial t} + c_i\frac{\partial}{\partial x}\right)^3 f_i + \mathcal{O}(h^4) = -\omega\left(f_i - f_i^{(0)}\right). \end{aligned} \tag{2.48}$$

For later use we introduce the abbreviations

$$\mathrm{D}_t = \frac{\partial}{\partial t} + c_i\frac{\partial}{\partial x} \qquad \text{and} \qquad \mathrm{D}_t^{(0)} = \frac{\partial^{(0)}}{\partial t} + c_i\frac{\partial^{(0)}}{\partial x}$$

for the substantial derivative, where the formal expansion of $\mathrm{D}_t^{(0)}$ is explained by (2.47). The expanded expressions (2.46) and (2.47) are inserted in the Taylor expanded lattice

BGK equation (2.48). The outcome is split with respect to powers of h:

$$
\begin{aligned}
h^1: &\qquad \mathrm{D}_t^{(0)} f_i^{(0)} = -\omega f_i^{(1)},\\
h^2: &\qquad \Big(1-\frac{\omega}{2}\Big)\,\mathrm{D}_t^{(0)} f_i^{(1)} + \frac{\partial^{(1)}}{\partial t} f_i^{(0)} = -\omega f_i^{(2)},\\
h^3: &\quad \frac{\partial^{(2)}}{\partial t} f_i^{(0)} + \frac{\partial^{(1)}}{\partial t} f_i^{(1)} + \mathrm{D}_t^{(0)} f_i^{(2)} - \omega \frac{\partial^{(1)}}{\partial t} f_i^{(1)} + \left(\frac{1}{2}-\frac{\omega}{6}\right)\left(\mathrm{D}_t^{(0)}\right)^2 f_i^{(1)} = -\omega f_i^{(3)}.
\end{aligned}
$$

In the Chapman-Enskog analysis, macroscopic equations are achieved by computing moments of this expanded equation. For our purpose, it is sufficient to compute the zeroth moment, which is done below in the split formulation, i.e., we compute the moment for each order of h separately. Although we are only interested in zeroth moments, we also have to compute some first moments, because a first moment of the h^k-terms is required to express the zeroth moment of h^{k+1}-terms. That means, we have to compute the zeroth and the first moment of h^1- and h^2-terms, as well as the zeroth moment of h^3-terms. All moments of $f_i^{(k)}$, $k \geq 1$, can be reduced to a calculation of higher moments of the local equilibrium (2.44). The latter can be stated explicitly. In fact, zeroth moments of $f_i^{(k)}$ vanish for all $k \geq 1$. Furthermore, the zeroth moment of f_i is exclusively defined by the discrete equilibrium. In total we get the following zeroth moments:

$$
\begin{aligned}
h^1: &\qquad \frac{\partial^{(0)}}{\partial t}\rho + a\frac{\partial^{(0)}}{\partial x}\rho = 0,\\
h^2: &\qquad \frac{\partial^{(1)}}{\partial t}\rho = \eta\frac{\partial^{(0)}}{\partial x}\frac{\partial^{(0)}}{\partial x}\rho,\\
h^3: &\qquad \frac{\partial^{(2)}}{\partial t}\rho = \lambda\frac{\partial^{(0)}}{\partial x}\frac{\partial^{(0)}}{\partial x}\frac{\partial^{(0)}}{\partial x}\rho,
\end{aligned}
$$

with

$$
\eta = \left(\frac{1}{\omega}-\frac{1}{2}\right)(1-a^2), \qquad \lambda = 2a\Big(\frac{1}{\omega^2}-\frac{1}{\omega}+\frac{1}{6}\Big)\Big(1-a^2\Big). \tag{2.49}
$$

All terms are recomposed to a single equation, this yields

$$
\frac{\partial \rho}{\partial t} + a\frac{\partial \rho}{\partial x} = h\eta\frac{\partial^2 \rho}{\partial x^2} + h^2\lambda\frac{\partial^3 \rho}{\partial x^3} + \mathcal{O}(h^3).
$$

Thus, for h small the dominant behavior of the D1Q2 model with equilibrium (2.44) is given by an advection with constant velocity defined by a. Additionally, on a lower scale numerical diffusion appears. The same result is obtained by Junk and Rheinländer [56] with their more general asymptotic expansion.

2.4.2 Thermodynamical flows and other applications

The previous section gave already a detailed example for a LBM with another application than the NSE. As an explicit numerical scheme with only local operations, the LBM is naturally attractive for all kind of problems. It has the advantage of being easy to implement on recent computer architectures focusing on parallel computations. And it is especially suitable for computations on graphics processing units (GPUs), e.g., [110, 121]. In this section we merely list some *partial differential equations* (PDEs) which can be

numerically solved with a suitable LBM. By this we aim to demonstrate that the LBM is by now a more universal scheme than originally developed.

The model of the previous section has shown to recover mainly a one-dimensional advection equation. The approach can be extended so the LBM can also be used to solve higher dimensional (nonlinear) convection-diffusion equations [97, 111]. A LBM can be constructed for solving the Korteweg-de Vries equations [124] and the shallow water equations [70, 126]. It is noteworthy that the shallow water equations can be derived from the NSE and therefore do not represent a completely different class of problems. However, LBMs for PDEs which have a clear distance from the NSE exist. They comprise the Kohn-Sham equations [76], Maxwell equations [75], (second order) Benjamin-Ono equations [67] and quantum LB simulations for Bose–Einstein condensates (Gross-Pitaevskii equation) [78].

Moreover let us go into more detail for a method used for thermodynamical processes, such as the Rayleigh-Bénard convection [31]. For thermodynamical flows the NSE (2.3) are often considered alongside with an energy conservation equation. Firstly, we recapitulate that the basic LBM simulates isothermal flows of fluids. Secondly, for the derivation of the D3Q19 velocities the conservation condition (2.16) was used, which implied the requirement $d_p \geq 5$ (see Section 2.3.3 for details). To recover an energy equation with the LBM the conservation condition (2.16) has to be extended by

$$\int_{\mathbb{R}^d} \psi_5 Q(f)\,\mathrm{d}\boldsymbol{\xi} = 0,$$

with $\psi_5 = |\boldsymbol{\xi}|^2$, see (2.7). This implies $d_p \geq 6$ and it can be shown that the D3Q19 is not capable to satisfy the resulting requirement. Therefore to obtain an energy conservation equation the lattice vectors have to be adapted, e.g., by using multi-speed models. A review of multi-speed models for thermodynamical simulations is given in [34]. Using a multi-speed approach with the BGK approximation has the limitation to have a fixed Prandtl number. This is caused by having only one free parameter. The problem can be overcome by using a collision operator having more free parameters or by using a double distribution function approach instead [34]. In double distribution function approaches the energy equation is obtained by a separate LBM. It is also conceivable to construct a hybrid method, where the NSE (2.3) are solved by a classical solver and the energy equation by a LB scheme, as presented here. A hybrid method the other way round is proposed in [68]. We here consider only the energy LBM, given by the LBE for the energy populations, say g_i:

$$g_i(\boldsymbol{x} + \boldsymbol{c}_i\Delta t, t + \Delta t) - g_i(\boldsymbol{x}, t) = \mathrm{C}_i(\boldsymbol{x}, t).$$

The flow variables, i.e., density and fluid velocity are assumed to be known already, such that the energy variations may only have a negligible effect on the real fluid's motion. In the model of He et al. [39] the right hand side is chosen by a BGK approximation plus a source term. In this way, their model is able to incorporate viscous dissipation as well as compression work. Neglecting additional source terms, i.e., when having a simple BGK like collision term

$$\mathrm{C}_i(\boldsymbol{x}, t) = -\omega_g \left[g_i(\boldsymbol{x}, t) - g_i^{\mathrm{eq}}(\boldsymbol{x}, t)\right],$$

the model is not considering viscous dissipation and compression work. Thus it satisfies

an advection-diffusion-equation [80, 81]

$$\frac{\partial(\rho e)}{\partial t} + \boldsymbol{\nabla} \cdot (\rho e \boldsymbol{u}) = \boldsymbol{\nabla} \cdot (\rho \chi \boldsymbol{\nabla} e)\,,$$

provided the following d-dimensional discrete energy equilibrium is used in the collision term

$$g_i^{\mathrm{eq}}(\boldsymbol{x}, t) = w_i \rho e \left[\frac{3\,|\boldsymbol{c}_i^2|}{dc^2} + \left(\frac{9\,|\boldsymbol{c}_i|^2}{dc^2} - \frac{6}{d} \right) \frac{\langle \boldsymbol{c}_i,\, \boldsymbol{u} \rangle}{c^2} + \frac{9}{2} \frac{\langle \boldsymbol{c}_i,\, \boldsymbol{u}^2 \rangle}{c^4} - \frac{3}{2} \frac{|\boldsymbol{u}|^2}{c^2} \right],$$

with model dependent weights w_i as in Table 2.1. The relaxation time ω_g is related to the thermal conductivity χ, and the zeroth moment of g_i computes the energy density [39, 80, 81]

$$\rho(\boldsymbol{x}, t) e(\boldsymbol{x}, t) = \sum_{j=0}^{q-1} g_j(\boldsymbol{x}, t).$$

This model again emphasizes that the equilibrium distribution is essential for the macroscopic evolution.

2.5 Problem formulation

In this section we summarize a complete LB simulation solving an arbitrary physical problem. For later use, we introduce the notation

$$P := \mathtt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}, \mathrm{C}, f_j, I_j, B_j),$$

where $\mathtt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}, \mathrm{C}, f_j, I_j, B_j)$ represents a complete LB setup as explained below, shortly referred to as P. Next, the arguments of $\mathtt{LB}()$ are explained in detail by summarizing the procedure described in the sections above.

Discrete spaces Let $\Omega \subset \mathbb{R}^d$ be the $d-$dimensional computational domain on which the problem is formulated. The LB simulation is working on a computational grid $\mathcal{G}$, which is a discrete subset of Ω:

$$\mathcal{G} \subseteq \Omega \cap \varrho \mathbb{Z}^d, \quad \varrho \in \mathbb{R}.$$

A set of discrete velocities, sometimes called *lattice vectors*, is essential for the simulation:

$$\mathcal{V} = \{\boldsymbol{c}_j \in \mathbb{R}^d \mid j \in \mathcal{I}\}.$$

Hereby $\mathcal{I}$ is the set of feasible indices for discrete velocities, populations and any derived quantities. It depends on the chosen discretization model. Using the temporal step size Δt and an initial time t_0 we obtain a set of $(N_t + 1)$ discrete time levels

$$\mathcal{T} = \{t_k \in \mathbb{R} \mid t_k = t_0 + k\Delta t,\ k = 0, \ldots, N_t\}.$$

The LBM computes the evolution of *populations*

$$f_j : \mathcal{G} \times \mathcal{T} \to \mathbb{R}, \quad j \in \mathcal{I}.$$

The evolution process is split into a *streaming* step and a *collision* step. Hereby, the local collision is described by a term C.

Type of grid points We can distinguish two categories, $\mathcal{B}$ and $\mathcal{F}$, for a *lattice point* $\boldsymbol{x} \in \mathcal{G} = \mathcal{B} \cup \mathcal{F}$. First, a boundary point $\boldsymbol{x} \in \mathcal{B}$ is given when there is a lack of adjacent points. That is, $\boldsymbol{x}_b \in \mathcal{B}$ is called a *boundary point* if there is at least one lattice vector $\boldsymbol{c}_j \in \mathcal{V}$, such that $\boldsymbol{x}_b - \boldsymbol{c}_j \Delta t \notin \mathcal{G}$. The reason for writing the last statement in this form (i.e., with a minus sign) is, because it is this lack of an adjacent lattice point which leads to an undetermined population $f_j(\boldsymbol{x}_b, t)$. The first category $\mathcal{B} \subset \mathcal{G}$ is formally given by

$$\mathcal{B} := \{\boldsymbol{x} \in \mathcal{G} \mid \exists j \in \mathcal{I} : \ \boldsymbol{x} - \boldsymbol{c}_j \Delta t \notin \mathcal{G}, \ \boldsymbol{c}_j \in \mathcal{V}\}. \tag{2.50}$$

The second category $\mathcal{F} := \mathcal{G} \setminus \mathcal{B}$ is the set of all ordinary lattice points, for which the LBE is used to compute new populations.

Initialization For the algorithm to work, obviously, populations have to be initialized at $t = t_0$, that is done formally by $f_j(\boldsymbol{x}, t_0) = I_j(\boldsymbol{x})$, $\boldsymbol{x} \in \mathcal{G}$, with functions

$$I_j : \mathcal{G} \to \mathbb{R}, \quad j \in \mathcal{I}.$$

Boundary problem For each boundary lattice point $\boldsymbol{x}_b \in \mathcal{B}$, as defined via (2.50), we can split the index set $\mathcal{I}$ into three disjunctive subsets

$$\mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b) := \{j \in \mathcal{I} \mid \boldsymbol{x}_b - \boldsymbol{c}_j \Delta t \notin \mathcal{G}\}, \tag{2.51a}$$

$$\mathcal{I}_{\mathcal{G}}^{+}(\boldsymbol{x}_b) := \left\{j \in \mathcal{I} \setminus \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b) \mid \boldsymbol{c}_k = -\boldsymbol{c}_j, \ k \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)\right\}, \tag{2.51b}$$

$$\mathcal{I}_{\mathcal{G}}^{0}(\boldsymbol{x}_b) := \mathcal{I} \setminus \left(\mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b) \cup \mathcal{I}_{\mathcal{G}}^{+}(\boldsymbol{x}_b)\right). \tag{2.51c}$$

For each boundary point $\boldsymbol{x}_b \in \mathcal{B}$, the populations $f_j(\boldsymbol{x}_b, t)$, $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$, are not explained by the LBE, due to the lack of a required adjacent lattice point. To ensure having all populations in all grid points $\boldsymbol{x} \in \mathcal{G}$ at any time, an additional equation is necessary to compute the undetermined populations $f_j(\boldsymbol{x}_b, \cdot)$, $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$. This is exactly the minimal task of a *boundary condition* (BC) within the LBM. Therefore, the general formulation of a LB simulation ends with imposing BCs for these undetermined populations:

$$f_j(\boldsymbol{x}_b, t) = B_j(\boldsymbol{x}_b, t), \quad \boldsymbol{x}_b \in \mathcal{B}, \ j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b), \ t \in \tilde{\mathcal{T}} := \mathcal{T} \setminus \{t_0\}. \tag{2.52}$$

Each B_j is seen as a function $B_j : \mathcal{B} \times \tilde{\mathcal{T}} \to \mathbb{R}$, where $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$. We note that some BCs also recompute (some) already known populations at a boundary point (e.g., [98]), then $j \in \mathcal{I}$. At a first glance, the choice of B_j is in some sense arbitrary, although a derivation from a physical condition is more reasonable. For example one can compute the unknown populations in such a way that a fluid pressure or velocity at the boundary is specified [127]. If a boundary of the computational domain is aligned with a physical boundary often a physical condition can be chosen. Exemplarily, for a boundary which physically agrees with a wall, often no-slip BCs are applicable [52]. However, for open boundaries, which

are not aligned with any physical boundary, a condition has to be found, which behaves still physically correct. The following chapters focus on this and aim finding conditions for open boundaries.

3 Chapter 3
Non-reflecting boundary conditions

We consider open boundaries, i.e., boundaries not aligned with a physical boundary. Applying for instance a fixed pressure condition here will generate some unphysical, spurious reflections. For open boundaries the BC should compute the incoming populations such that the boundary preferably has no interaction with the fluid. In this chapter we first discuss such an *ideal transparent boundary condition* (ITBC) for open boundaries (Section 3.1). This ITBC is of importance for measuring errors of other BCs. The second section (Section 3.2) gives a short literature survey of existing concepts of open boundaries for the LBM for fluid flows. Hereby, we go into more detail for so-called *impedance boundary conditions* (IBCs) [93] and *perfectly matched layer* (PML) approaches [18, 77, 106], since they are used to draw comparisons with our own BCs, which are found in this and in upcoming chapters. One of our BCs is an extension to *characteristic boundary conditions* (CBCs) [53, 59]. The original version and our extension is presented in Section 3.3, in Section 3.4 one finds further enhancements of CBCs. The upcoming two chapters deal with remaining BCs developed by us, which unlike the BCs of this chapter are fully described on the discrete level.

For what follows, let all populations in interior grid points $\boldsymbol{x} \in \mathcal{F}$ be given at all time levels up to time $t = t_{s+1}$. This assumption implies that all populations in a boundary point $\boldsymbol{x}_b \in \mathcal{B}$ up to time $t = t_s$ are known, as well as $f_j(\boldsymbol{x}_b, t_{s+1})$, $j \in \mathcal{I} \setminus \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$. Especially, also the macroscopic quantities (2.42) at time level $t = t_s$ are known in a boundary point. Hence, we concentrate on open BCs for the time level $t = t_{s+1}$, having the task to compute the unknown populations for $\boldsymbol{x}_b \in \Gamma$ at time $t = t_{s+1}$. Here $\Gamma \subseteq \mathcal{B}$ is the set of all boundary points which correspond to an open boundary. The boundaries considered here are further assumed to be aligned with the main lattice direction. In other words, we are considering a LB setup P (see also Section 2.5)

$$P := \mathtt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}, \mathrm{C}, f_j, I_j, B_j) \tag{3.1}$$

and are looking for the values of $B_j(\boldsymbol{x}_b, t_{s+1})$, $\boldsymbol{x}_b \in \Gamma$, $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$. For $\boldsymbol{x}_b \in \mathcal{B} \setminus \Gamma$ the boundary populations $B_j(\boldsymbol{x}_b, t_{s+1})$ shall be given, e.g., derived from physical information at these boundaries.

3.1 An ideal transparent boundary condition

Let be given the setup P of (3.1), i.e., a LB simulation on a computational grid $\mathcal{G}$ with initial data $I_j(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{G}$. Furthermore, we consider

$$P^{\mathrm{ref}} := \mathtt{LB}(\mathcal{G}^{\mathrm{ref}}, \mathcal{T}, \mathcal{V}, \mathrm{C}, f_j^{\mathrm{ref}}, I_j^{\mathrm{ref}}, B_j^{\mathrm{ref}}),$$

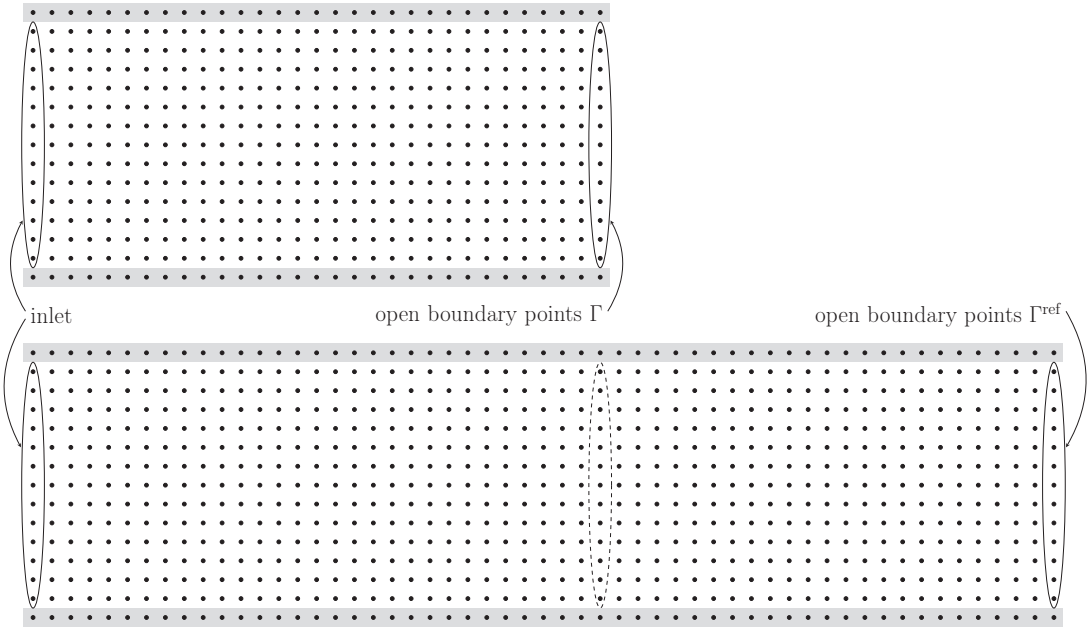


Figure 3.1: *Extension of a computational grid for a two-dimensional channel flow* – On top the computational grid $\mathcal{G}$ for a two-dimensional channel flow is visualized, while its extended grid $\mathcal{G}^{\mathrm{ref}}$ is shown at the bottom. Within the extension the open boundary points of $\mathcal{G}$ become usual interior fluid points.

this is another, *equally discretized* LB simulation on a *sufficiently larger* computational grid $\mathcal{G}^{\mathrm{ref}} \supset \mathcal{G}$. Let $\Gamma^{\mathrm{ref}} \subseteq \mathcal{B}^{\mathrm{ref}}$ denote the set of lattice points corresponding to open boundaries of $\mathcal{B}^{\mathrm{ref}}$. The reference grid $\mathcal{G}^{\mathrm{ref}}$ is said to be *sufficiently larger* if the populations $f_j^{\mathrm{ref}}(\boldsymbol{x}_b, \cdot)$, $\boldsymbol{x}_b \in \Gamma \subset \mathcal{G} \subset \mathcal{G}^{\mathrm{ref}}$, are independent of $B_j^{\mathrm{ref}}(\boldsymbol{x}, t)$, for all $\boldsymbol{x} \in \Gamma^{\mathrm{ref}}$ and $t \in \mathcal{T}$. That is, the boundary populations B_j in (3.3) are independent of open boundary values $B_j^{\mathrm{ref}}(\boldsymbol{x}, \cdot)$, $\boldsymbol{x} \in \Gamma^{\mathrm{ref}}$. In particular for $\mathcal{G}^{\mathrm{ref}}$ this implies an extension specially at the open boundaries, that is $\Gamma \cap \Gamma^{\mathrm{ref}} = \varnothing$. We illustrate an extended grid for a two-dimensional channel flow in Fig. 3.1. The formulation *equally discretized* used above refers to the equal choice of the velocity discretization model $\mathcal{V}$ as well as using same spatial and temporal step sizes. The superscript in the setup P^{ref} indicates that the populations f_j^{ref} serve as reference values. For these reference populations we require a consistent initialization in the joint lattice points $\mathcal{G}^{\mathrm{ref}} \cap \mathcal{G} = \mathcal{G}$:

$$I_j^{\mathrm{ref}}(\boldsymbol{x}) = I_j(\boldsymbol{x}) \qquad \text{for all } \boldsymbol{x} \in \mathcal{G}. \tag{3.2}$$

Eventually, an ITBC for the open boundary is equivalent to compute the unknown populations of P by

$$f_j(\boldsymbol{x}_b, t_{s+1}) = B_j(\boldsymbol{x}_b, t_{s+1}) := f_j^{\mathrm{ref}}(\boldsymbol{x}_b, t_{s+1}), \qquad \boldsymbol{x}_b \in \Gamma,\ j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b). \tag{3.3}$$

Clearly, the ideal populations depend not only on the initial values of (3.2), but also on the initial populations $I_j^{\mathrm{ref}}(\boldsymbol{x})$ in the points $\boldsymbol{x} \in \mathcal{G}^{\mathrm{ref}} \setminus \mathcal{G}$. Hence, an ITBC takes exterior information into account (from the view of P). Unless differently mentioned, the exterior populations are assumed to be chosen homogeneously in an equilibrium state. The equilibrium is computed with appropriate values of ρ and $\boldsymbol{u}$, such that no inflow is expected to $\mathcal{G}$.

After the BC (3.3) has been applied all populations are known at a point $\boldsymbol{x}_b$, and new values $\rho(\boldsymbol{x}_b, t_{s+1})$ and $\boldsymbol{u}(\boldsymbol{x}_b, t_{s+1})$ can be computed, see (2.42). Therefore some corresponding values for macroscopic quantities are implied by an ITBC, but it is noteworthy that the knowledge of these values (i.e., $\rho(\boldsymbol{x}_b, t_{s+1})$ and $\boldsymbol{u}(\boldsymbol{x}_b, t_{s+1})$) would not be sufficient to compute the ideal populations (3.3). More precisely, any boundary condition $B_j(\boldsymbol{x}_b, t_{s+1})$, $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$, implies corresponding macroscopic quantities and we emphasize that these quantities are linked to each other: There is a commonly known connection between the density and the normal velocity, see, e.g., [127]. For the explanation we consider a boundary node $\boldsymbol{x}_b \in \mathcal{B}$ of the computational grid (no edge or corner). The unknown populations are f_j with $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$, whereas the populations corresponding to the opposite directions, i.e., f_j with $j \in \mathcal{I}_{\mathcal{G}}^{+}(\boldsymbol{x}_b)$, are known. Let their sum be denoted by $\rho^+ = \sum_{j \in \mathcal{I}_{\mathcal{G}}^{+}(\boldsymbol{x}_b)} f_j$ and the sum of the remaining (also known) populations by $\rho^0 = \sum_{j \in \mathcal{I}_{\mathcal{G}}^{0}(\boldsymbol{x}_b)} f_j$. The latter is the sum over all populations corresponding to directions tangential to the boundary and the rest population. The implied density $\rho(\boldsymbol{x}_b, t)$ and normal velocity component $u_n(\boldsymbol{x}_b, t) = \langle \boldsymbol{n}, \boldsymbol{u}(\boldsymbol{x}_b, t) \rangle$ satisfy the following equations:

$$\begin{aligned} \rho(\boldsymbol{x}_b, t) &= \rho^+(\boldsymbol{x}_b, t) + \rho^-(\boldsymbol{x}_b, t) + \rho^0(\boldsymbol{x}_b, t), \\ \rho(\boldsymbol{x}_b, t) u_n(\boldsymbol{x}_b, t) &= \rho^+(\boldsymbol{x}_b, t) - \rho^-(\boldsymbol{x}_b, t). \end{aligned}$$

Analogously, it is $\rho^- = \sum_{j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)} f_j$. Both previous expressions are an immediate consequence of the definitions (2.42). The unit normal vector $\boldsymbol{n}$ of the boundary is pointing outside the computational domain. Both equations can be combined, such that no unknown populations occur anymore:

$$\rho(\boldsymbol{x}_b, t) = \frac{2\rho^+(\boldsymbol{x}_b, t) + \rho^0(\boldsymbol{x}_b, t)}{1 + u_n(\boldsymbol{x}_b, t)}. \tag{3.4}$$

This relation is in some sense fundamental, since it is valid for any BC and not only for an ITBC. As a direct consequence, there is no actual BC, which implements arbitrary values of the mass density and the normal velocity at the same time, but only those values which satisfy (3.4). However arbitrary values can be implemented when also actually known populations in the boundary point are recomputed.

3.2 Artificial boundary conditions

One has to deal with an artificial boundary if the computational domain is confined, although the physical problem is easily formulated on a much larger or even an unbounded domain. Since these open boundaries are not aligned with a physical boundary, finding a BC describing the open boundary physically correct (see ITBC above) is a challenging task. Using a standard BC at an open boundary would create unphysical reflections [54]. For example, let us assume we would prescribe a fixed velocity at an open boundary, which fits to the (unperturbed) background flow of the problem. Any pressure wave traveling has a fluid velocity component deviating from this background flow velocity. Now, the only way that the fixed velocity BC can be satisfied when the pressure wave hits the open boundary is by reflection, so that the deviating fluid velocities of the original and the reflected wave cancel. To obtain an ideal open boundary, it follows that the velocity has to be chosen dynamically. For open boundaries in the LBM there exist a couple of techniques and we list some selected approaches.

In an *exit boundary condition* proposed by Chikatamarla et al. [16] the inward populations at a boundary point $\boldsymbol{x}_b$ are computed by evaluating a non-equilibrium distribution based on Grad's approximation [33]:

$$\begin{aligned} B_j(\boldsymbol{x}_b, t) &= f_j^{\mathrm{G}}\big(\rho(\boldsymbol{x}_b, t-\Delta t), \boldsymbol{u}(\boldsymbol{x}_b, t-\Delta t), \boldsymbol{P}(\boldsymbol{x}_b, t-\Delta t)\big) \\ &:= \rho w_j \left(1 + 3\frac{\langle \boldsymbol{c}_j,\, \boldsymbol{u}\rangle}{c^2} + \frac{9}{2}\frac{\left[\boldsymbol{P} - c_s^2\rho\boldsymbol{I}\right] : \left[\boldsymbol{c}_j\boldsymbol{c}_j^\top - c_s^2\boldsymbol{I}\right]}{c^4}\right), \end{aligned}$$

with values for ρ, $\boldsymbol{u}$ and $\boldsymbol{P}$ from the previous time level. Here $\boldsymbol{A} : \boldsymbol{B} := \mathrm{tr}(\boldsymbol{A}\boldsymbol{B}^\top)$ is the *Frobenius inner product* as well as $\boldsymbol{I} \in \mathbb{R}^{d\times d}$ the identity and $\boldsymbol{P}$ the pressure tensor

$$\boldsymbol{P}(\boldsymbol{x}, t) = \sum_{j=0}^{q-1} f_j(\boldsymbol{x}, t)\boldsymbol{c}_j\boldsymbol{c}_j^\top.$$

Vergnault et al. [115] extend the computational domain by some additional layers. In this added zone the viscosity is artificially increased by varying the relaxation parameter. Due to the monotonically increasing viscosity all waves entering this *absorbing layers* are aimed to be damped, so that smaller reflections are observed at the boundary of the extended domain. Similarly, a technique for damping of acoustic waves is described by Viggen in [116]. In a strict sense, this concept does not describe a boundary condition, but an *absorbing layer treatment*, which still needs a BC to close the layer.

Kam et al. [57] discuss additional BCs for artificial boundaries in their article. In the following two subsections we present further approaches for artificial boundaries in more detail, including both a BC and an absorbing layer.

3.2.1 Perfectly matched layers

The concept of a *perfectly matched layer* (PML) was originally introduced by Bérenger for absorbing electromagnetic waves of the Maxwell equations [4]. In this approach the computational grid is extended by a PML region, such that all waves entering this region are damped or absorbed. To be more precise, the PML technique is not a BC in a strict sense. It introduces a damping zone in the (extended) computational domain, for which a BC is still required. The equations in the PML region are constructed, such that, at least theoretically, waves are not reflected at the interface of the actual computational domain and the PML region. To explain the PML technique we consider an auxiliary two-dimensional system of equations

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{U}}{\partial x_1} + \boldsymbol{B}\frac{\partial \boldsymbol{U}}{\partial x_2} = 0, \tag{3.5}$$

where $\boldsymbol{U} = \boldsymbol{U}_1 + \boldsymbol{U}_2$ is split. Next, absorption coefficients σ_j are introduced by considering a decomposed formulation for the above auxiliary system:

$$\frac{\partial \boldsymbol{U}_1}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{U}}{\partial x_1} = -\sigma_x\boldsymbol{U}_1, \qquad \frac{\partial \boldsymbol{U}_2}{\partial t} + \boldsymbol{B}\frac{\partial \boldsymbol{U}}{\partial x_2} = -\sigma_y\boldsymbol{U}_2.$$

This system for the split variables $\boldsymbol{U}_j$ can be recast into a system for the original variable $\boldsymbol{U}$ by a transformation to the frequency domain, an ensuing analytical combination and finally a transformation back to the time domain. Details can be found, e.g., in [50], and the result reads

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{U}}{\partial x_1} + \boldsymbol{B}\frac{\partial \boldsymbol{U}}{\partial x_2} = -(\sigma_x + \sigma_y)\boldsymbol{U} - \sigma_x\sigma_y\boldsymbol{Q} - \sigma_y\boldsymbol{A}\frac{\partial \boldsymbol{Q}}{\partial x_1} - \sigma_x\boldsymbol{B}\frac{\partial \boldsymbol{Q}}{\partial x_2}, \tag{3.6a}$$

$$\frac{\partial \boldsymbol{Q}}{\partial t} = \boldsymbol{U}. \tag{3.6b}$$

The concept of a PML for the LBM was proposed by Najafi-Yazdi and Mongeau [77] as well as Tekitek et al. [106]. In the work of Tekitek et al. an MRT collision model is modified, such that in the asymptotic limit the method recovers the PML formulation of the macroscopic conservation equations. We do not further examine this approach here. Another approach is followed by Najafi-Yazdi and Mongeau [77], which incorporates the PML technique into the LBM via the mesoscopic level of the *discrete velocity BGK-Boltzmann equation* (DVBE) (see, e.g., [82] for more information about the DVBE). Similarly, in the work of Craig and Hu [18] another PML formulation is given for the DVBE. Although the focus in [18] lies on finite difference methods for the DVBE, the PML formulation can be used for the LBM as well. Next, we further explain the PML approaches of Najafi-Yazdi and Mongeau [77] as well as of Craig and Hu [18].

The PML of Najafi-Yazdi and Mongeau To implement the concept of PMLs to the LBM Najafi-Yazdi and Mongeau [77] consider the DVBE [82]

$$\frac{\partial f_i}{\partial t} + \boldsymbol{c}_i \cdot \boldsymbol{\nabla} f_i = \mathrm{C}_{\mathrm{BGK},i}, \tag{3.7}$$

where $f_i(\boldsymbol{x}, t) : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$ is the distribution related to a given particle velocity $\boldsymbol{c}_i$. The DVBE is equivalently written in a vector notation

$$\frac{\partial \boldsymbol{f}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{f}}{\partial x_1} + \boldsymbol{B}\frac{\partial \boldsymbol{f}}{\partial x_2} = \mathrm{C}_{\mathrm{BGK}},$$

where $\boldsymbol{A} = \operatorname{diag}(c_{0,1}, \ldots, c_{q-1,1})$ and $\boldsymbol{B} = \operatorname{diag}(c_{0,2}, \ldots, c_{q-1,2})$ are diagonal matrices using $\boldsymbol{c}_i = (c_{i,1}, c_{i,2})^\top$, and $\boldsymbol{f} = (f_j)_{j=0,\ldots,q-1}$. For an application of the PML technique in [77] the distribution functions are decomposed as

$$\boldsymbol{f} = \bar{\boldsymbol{f}}^{\mathrm{eq}} + \tilde{\boldsymbol{f}}^{\mathrm{eq}} + \boldsymbol{f}^{\mathrm{neq}}, \qquad \boldsymbol{f}^{\mathrm{eq}} = \bar{\boldsymbol{f}}^{\mathrm{eq}} + \tilde{\boldsymbol{f}}^{\mathrm{eq}}, \qquad \boldsymbol{f}^{\mathrm{neq}} = \boldsymbol{f} - \boldsymbol{f}^{\mathrm{eq}}, \tag{3.8}$$

where the bar and tilde denote the mean and perturbed equilibrium portions, respectively, and $\boldsymbol{f}^{\mathrm{neq}}$ are the non-equilibrium portions. It is concluded that the DVBE for $\tilde{\boldsymbol{f}}^{\mathrm{eq}}$ reads

$$\frac{\partial \tilde{\boldsymbol{f}}^{\mathrm{eq}}}{\partial t} + \boldsymbol{A}\frac{\partial \tilde{\boldsymbol{f}}^{\mathrm{eq}}}{\partial x_1} + \boldsymbol{B}\frac{\partial \tilde{\boldsymbol{f}}^{\mathrm{eq}}}{\partial x_2} = 0,$$

and therefore is of the form (3.5), such that the damped PML formulation (3.6) can be applied to $\tilde{\boldsymbol{f}}^{\mathrm{eq}}$. The authors of [77] propose to use $\sigma_x = \sigma_y = \sigma$ to overcome numerical instabilities. After combining all equations for the portions in (3.8), one obtains the PML-

DVBE to be solved in the PML region:

$$\frac{\partial \boldsymbol{f}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{f}}{\partial x_1} + \boldsymbol{B}\frac{\partial \boldsymbol{f}}{\partial x_2} = \mathrm{C}_{\mathrm{BGK}} + \mathrm{C}_{\mathrm{PML}}, \tag{3.9}$$

with

$$\mathrm{C}_{\mathrm{PML}} = -\sigma\left[\boldsymbol{A}\frac{\partial \boldsymbol{Q}}{\partial x_1} + \boldsymbol{B}\frac{\partial \boldsymbol{Q}}{\partial x_2} + 2\tilde{\boldsymbol{f}}^{\mathrm{eq}} + \sigma\boldsymbol{Q}\right], \qquad \frac{\partial \boldsymbol{Q}}{\partial t} = \tilde{\boldsymbol{f}}^{\mathrm{eq}}.$$

A spatial and temporal discretization of the DVBE (3.7) yields the usual LBE, see, e.g., [120]. Together with a discretization of $\mathrm{C}_{\mathrm{PML}}$ we obtain the LBM with a PML damping zone. The three-dimensional formulation can be found in [77].

The PML of Craig and Hu Another PML formulation for the DVBE is given by Craig and Hu in [18]. In contrast to the previous formulation it is based on a decomposition

$$\boldsymbol{f} = \bar{\boldsymbol{f}}^{\mathrm{eq}} + \boldsymbol{f}',$$

with $\bar{\boldsymbol{f}}^{\mathrm{eq}}$ being the mean flow equilibrium distribution as above in (3.8). The PML equation is derived for the perturbed portion $\boldsymbol{f}'$, and thus also non-equilibrium portions are considered. Moreover a space-time transformation is applied for stability reasons. For a mean flow in x_1-direction with velocity u_0 it reads

$$\bar{t} = t + \beta x_1, \qquad \beta = -\frac{u_0}{c_s^2 - u_0^2}.$$

Then, the PML formulation is given by (3.9), but with

$$\begin{aligned}\mathrm{C}_{\mathrm{PML}} = {} & -\sigma_y \boldsymbol{A}\frac{\partial \boldsymbol{Q}}{\partial x_1} - \sigma_x \boldsymbol{B}\frac{\partial \boldsymbol{Q}}{\partial x_2} - (\sigma_x + \sigma_y)\boldsymbol{f}' - \sigma_x\sigma_y\boldsymbol{Q} - \sigma_x\beta\boldsymbol{A}\left[\boldsymbol{f}' + \sigma_y\boldsymbol{Q}\right] \\ & - \sigma_y\beta\boldsymbol{A}\boldsymbol{f}' - \omega\left[(\sigma_x + \sigma_y)\boldsymbol{r}_1 + \sigma_x\sigma_y\boldsymbol{r}_2\right], \\ \frac{\partial \boldsymbol{Q}}{\partial t} = {} & \boldsymbol{f}', \quad \frac{\partial \boldsymbol{r}_1}{\partial t} = \boldsymbol{f}^{\mathrm{neq}}, \quad \frac{\partial \boldsymbol{r}_2}{\partial t} = \boldsymbol{r}_1.\end{aligned}$$

The terms in the second line of $\mathrm{C}_{\mathrm{PML}}$ result from taking non-equilibrium portions into account. The parameter ω is explained by the BGK collision model. In [18] also an alternative PML formulation is achieved, called a split formulation. It mainly differs from (3.6) by not combining the split equations for $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ in the frequency domain in the derivation leading to (3.6). For details we refer to [18], the outcome reads as follows:

$$\mathrm{C}_{\mathrm{PML}} = -\sigma_x\boldsymbol{q}_1 - \sigma_y\boldsymbol{q}_2 - \sigma_x\beta\boldsymbol{A}\boldsymbol{f}',$$

where the auxiliary variables $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ satisfy

$$\begin{aligned}\frac{\partial \boldsymbol{q}_1}{\partial t} + \sigma_x\boldsymbol{q}_1 + \sigma_x\beta\boldsymbol{A}\boldsymbol{f}' + \boldsymbol{A}\frac{\partial \boldsymbol{f}'}{\partial x_1} = 0, \\ \frac{\partial \boldsymbol{q}_2}{\partial t} + \sigma_y\boldsymbol{q}_2 + \boldsymbol{B}\frac{\partial \boldsymbol{f}'}{\partial x_2} = 0.\end{aligned}$$

Like the previous PML approach of Najafi-Yazdi and Mongeau, both PML formulations are implemented in the LBM in the same way. That is, in the PML region a discretized variant of $\mathrm{C}_{\mathrm{PML}}$ is added to the usual lattice BGK equation.

3.2.2 Impedance boundary condition

The *specific acoustic impedance* z defined as the ratio of pressure to the flow per unit area is a property of a medium. In acoustics it is well known that a pressure wave is partially reflected at an interface of two media, when the impedances of both media are not equal [60]. The *impedance boundary condition* (IBC) for the LBM developed by Schlaffer [93] is derived under the assumption that any pressure variation propagates with the speed of sound c_s. For the IBC of Schlaffer [93] the velocity at the boundary is derived from a balance equation for the incoming momentum flux and the rate of momentum change in a control volume, such that both terms cancel out. By this the impedance of the medium is set appropriately. For a pressure wave with normal incidence the density and normal velocity u_n should satisfy

$$\begin{aligned}\rho(\boldsymbol{x}_b, t_{s+1})\left[u_n(\boldsymbol{x}_b, t_{s+1}) - u_n(\boldsymbol{x}_b, t_s)\right]\left(c_s \pm \frac{1}{2}\left[u_n(\boldsymbol{x}_b, t_{s+1}) - u_n(\boldsymbol{x}_b, t_s)\right]\right) \\ \pm \left[\rho(\boldsymbol{x}_b, t_{s+1}) - \rho(\boldsymbol{x}_b, t_s)\right] c_s^2 = 0,\end{aligned} \tag{3.10}$$

where the plus and minus sign are chosen according to the boundary location. This expression can be solved explicitly for u_n by the use of (3.4), such that the normal velocity is the only unknown. This gives (for the minus sign equation)

$$\begin{aligned}u_n(\boldsymbol{x}_b, t_{s+1}) = u_n(\boldsymbol{x}_b, t_s) + c_s^2 \rho_z(\boldsymbol{x}_b) \\ + c_s\left(1 - \sqrt{\left(c_s\rho_z(\boldsymbol{x}_b) + 1\right)^2 + 2\rho_z(\boldsymbol{x}_b)\left[1 + u_n(\boldsymbol{x}_b, t_s)\right]} - 2\right),\end{aligned}$$

where

$$\rho_z(\boldsymbol{x}_b) = \frac{\rho(\boldsymbol{x}_b, t_s)}{\rho^0(\boldsymbol{x}_b, t_{s+1}) + 2\rho^+(\boldsymbol{x}_b, t_{s+1})}.$$

With the normal velocity given, the density $\rho(\boldsymbol{x}_b, t_{s+1})$ can be computed via (3.4). In higher dimensions each parallel velocity can be chosen freely, however in [93] it is computed with a *non-equilibrium bounce-back condition* (for further details see [93]). Also the inward populations at a boundary point are computed by evaluating the equilibrium distribution (2.39) with the dynamic quantities found in addition to a bounce-back rule for the non-equilibrium portions. Schlaffer also derived further adaptations of the IBC, such as an *isotropic IBC*, advantageous for non-orthogonal angles of incidence. Hereby, (3.10) is substituted by another non-linear expression, for which no explicit solution is available, thus, as a drawback of the isotropic IBC a numerical solution is required (e.g., by Newton's method). For further details and other adaptations, such as those for fixed reference levels of density and velocity, or an IBC for an incompressible model, see [93]. Lastly we remark, although populations enter the above computation, the IBC is not derived from discrete considerations.

3.3 Characteristic boundary conditions

Already before the LBM was known Hedstrom [44] and later Thompson [108] developed the concept of a *characteristic boundary condition* (CBC) for nonlinear hyperbolic equations. Recall that characteristic curves are explained only for hyperbolic equations, see

Section 2.3.1. The starting point of a CBC is a system (*basic system*) of hyperbolic equations (or an equivalent formulation) for the dynamic quantities where the reflection is aimed to be eliminated. The CBC is then formulated by considering the *characteristic form* of the *basic system* and by performing some analytical steps, which are described in more detail below. Intuitively, it is best if the *basic system* agrees with the system describing the evolution in the interior of the computational domain.

Thus, for an application in CFD, if the fluid in the interior of the computational domain is modeled by the Euler equations (2.4), the CBC should also select the Euler equations as the *basic system* and performs the characteristic analysis on them. Let us remind that the Euler equations are hyperbolic, while the Navier-Stokes equations represent a singularly perturbed parabolic system. Therefore, the concept of CBCs is not directly applicable to Navier-Stokes equations and an appropriate *basic system* for the characteristic analysis has to be selected. Below we focus on the application of the CBCs in the LBM.

The CBCs for the LBM presented in the current section below (i.e., [46, 53, 59]) are constructed with the aim to be perfectly non-reflective. With perfectly non-reflecting boundaries, we have no control of the pressure at the boundary after the waves have left the computational domain. Contrary, if we specify a fixed pressure at the boundary, the boundary loses the non-reflective feature. Similarly, a velocity BC is reflecting, while a perfectly *non-reflecting boundary condition* (NRBC) does not give control of the mean boundary velocity. Modifications shown in Section 3.4 are a compromise of both features, i.e., the specification of values at the boundary and a non-reflective feature. Non-reflecting CBCs for the LBM were first introduced by Izquierdo & Fueyo [53] and Kim et al. [59]. In both works the *local one-dimensional inviscid* (LODI) equations are taken as the *basic system* in the derivation of the BC. As an improvement of the LODI based CBCs we developed a CBC which is based on an extended system [46]. Details are given below for each the LODI based and our CBC. In all three publications [46, 53, 59] the CBCs focus on two-dimensional problems, whereas also three dimensional problems are considered below.

3.3.1 Basic systems of characteristic boundary conditions

The macroscopic evolution in the LBM is described by (2.14). This system can easily be written in the form of Navier-Stokes equations (2.3) by substituting the total stress tensor, see (2.13), and by using the conservation conditions (2.16). The equation of state (2.15) is transformed to

$$p(\boldsymbol{x},t) = \frac{c^2}{3}\rho(\boldsymbol{x},t), \tag{3.11}$$

by help of (2.21) and (2.35), so the speed of sound c_s can directly be derived to

$$c_s = \frac{c}{\sqrt{3}}. \tag{3.12}$$

Therefore the rewritten system (2.14) reads

$$\frac{\partial\rho}{\partial t} + \boldsymbol{\nabla}\cdot\rho\boldsymbol{u} = 0, \tag{3.13a}$$

$$\frac{\partial\rho\boldsymbol{u}}{\partial t} + \boldsymbol{\nabla}\cdot\left(\rho\boldsymbol{u}\boldsymbol{u}^\top\right) = -c_s^2\boldsymbol{\nabla}\rho + \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}, \tag{3.13b}$$

and due to being not hyperbolic it is not able to provide as the *basic system* for a CBC. The authors in [53, 59] take the LODI equations for two-dimensional problems as the *basic system* for their CBCs. Here we state the LODI equations for an open boundary normal to the x_1-direction, which we call the x_1-LODI equations. Unlike [53, 59], we directly neglect the energy conservation equation and use the equation of state (3.11) of the LBM:

$$\frac{\partial \rho}{\partial t} + u_1 \frac{\partial \rho}{\partial x_1} + \rho \frac{\partial u_1}{\partial x_1} = 0, \tag{3.14a}$$

$$\frac{\partial u_1}{\partial t} + u_1 \frac{\partial u_1}{\partial x_1} + \frac{c_s^2}{\rho} \frac{\partial \rho}{\partial x_1} = 0, \tag{3.14b}$$

$$\frac{\partial u_2}{\partial t} + u_1 \frac{\partial u_2}{\partial x_1} = 0. \tag{3.14c}$$

The LODI system (3.14) can be derived from (3.13) by the following approximations: In (3.13b) the deviatoric stress term is neglected and all tangential derivatives are suppressed. The drop of the deviatoric stress term is necessary to obtain hyperbolic equations. Contrary, the *basic system* in our CBC [46] is obtained also by neglecting the deviatoric stress term, but we keep the parallel derivatives. The resulting system reads in vector notation

$$\frac{\partial \boldsymbol{U}_2}{\partial t} + \boldsymbol{A}_{2,x_1} \frac{\partial \boldsymbol{U}_2}{\partial x_1} + \boldsymbol{A}_{2,x_2} \frac{\partial \boldsymbol{U}_2}{\partial x_2} = \boldsymbol{0}, \tag{3.15}$$

with vector of characteristic variables $\boldsymbol{U}_2^\top = \big(\rho,\, u_1,\, u_2\big)$. The coefficient matrices are

$$\boldsymbol{A}_{2,x_1} = \boldsymbol{A}_{2,x_1}(\boldsymbol{U}_2) = \begin{pmatrix} u_1 & \rho & 0 \\ \frac{c_s^2}{\rho} & u_1 & 0 \\ 0 & 0 & u_1 \end{pmatrix}, \quad \boldsymbol{A}_{2,x_2} = \boldsymbol{A}_{2,x_2}(\boldsymbol{U}_2) = \begin{pmatrix} u_2 & 0 & \rho \\ 0 & u_2 & 0 \\ \frac{c_s^2}{\rho} & 0 & u_2 \end{pmatrix}. \tag{3.16}$$

While the LODI system in the formulation (3.14) is valid only for boundaries normal to the x_1-direction, the system (3.15) is not limited to a certain boundary orientation. The systems (3.14) and (3.15) coincide when setting $\boldsymbol{A}_{2,x_2}$ to zero. Hence, to easily obtain the x_2-LODI equations for a boundary normal to the x_2-direction, we use (3.15) with $\boldsymbol{A}_{2,x_2}$ from (3.16), but $\boldsymbol{A}_{2,x_1}$ set to zero. The general d-dimensional *basic system* for the vector of unknowns $\boldsymbol{U}_d^\top = (\rho,\, u_1,\, \ldots,\, u_d)$ reads

$$\frac{\partial \boldsymbol{U}_d}{\partial t} + \sum_{\alpha=1}^{d} \boldsymbol{A}_{d,x_\alpha} \frac{\partial \boldsymbol{U}_d}{\partial x_\alpha} = \boldsymbol{0}, \quad d = 1, 2, 3. \tag{3.17}$$

In the one-dimensional case when no tangential derivatives are present, clearly the LODI based approach and ours coincide. For three-dimensional problems, the LODI based CBC requires the two coefficient matrices corresponding to parallel derivatives to vanish.

3.3.2 Characteristic analysis

Continuing with (3.17), we omit all subscripts d for a better readability. All statements comprise also LODI based CBCs by neglecting all terms involving parallel derivatives. The coefficient matrices in (3.17) are (real) diagonalizable, i.e., it holds

$$\boldsymbol{R}_{x_\alpha} \boldsymbol{A}_{x_\alpha} \boldsymbol{R}_{x_\alpha}^{-1} = \boldsymbol{\Lambda}_{x_\alpha}, \tag{3.18}$$

with diagonal matrices $\boldsymbol{\Lambda}_{x_\alpha} = \mathrm{diag}(\lambda_{x_\alpha,1}, \ldots, \lambda_{x_\alpha,d+1})$ of eigenvalues

$$\lambda_{x_\alpha,1} = u_\alpha - c_s, \qquad \lambda_{x_\alpha,2} = u_\alpha + c_s, \qquad \lambda_{x_\alpha,j} = u_\alpha, \qquad j = 3, \ldots, d+1.$$

Note that the dimension d has no effect to the first two eigenvalues, but only determines the algebraic and geometric multiplicity of the eigenvalue $\lambda = u_\alpha$. Each matrix $\boldsymbol{R}_{x_\alpha}$ is constructed from the corresponding eigenvectors, e.g., for $\boldsymbol{R}_{x_1}$ we get (here $d = 2$)

$$\boldsymbol{R}_{x_1} = \begin{pmatrix} c_s^2 & -c_s\rho & 0 \\ c_s^2 & c_s\rho & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad \boldsymbol{R}_{x_1}^{-1} = \begin{pmatrix} \frac{1}{2c_s^2} & \frac{1}{2c^2} & 0 \\ -\frac{1}{2c_s\rho} & \frac{1}{2c_s\rho} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The matrices for other dimensions and directions can be found in Appendix B.

Depending on the location of a boundary point, we identify a number of N_n spatial derivatives as orthogonal, and the other $N_p = d - N_n$ derivatives as parallel. For a usual boundary point there is exactly one orthogonal derivative ($N_n = 1$), and there are $N_p = d - 1$ parallel ones. If the boundary point is located at a corner of the computational domain, then all spatial derivatives are assumed to be orthogonal ($N_n = d$). Moreover, in three dimensions a boundary point may be located on an edge. In this case there shall be two orthogonal ($N_n = 2$) and only one parallel derivative ($N_p = 1$). We denote the orthogonal derivative(s) with $\frac{\partial}{\partial n_j}$ (normal), with $j = 1, \ldots, N_n$. In an analogue manner, the parallel derivatives are written as $\frac{\partial}{\partial p_j}$, for $j = 1, \ldots, N_p$. For instance let us consider a boundary, for which the x_1-direction is normal, then $n_1 = x_1$. On the other hand, for a boundary point (in three dimensions) lying on an edge which is parallel to the x_2-axis, we have two orthogonal derivatives, $n_1 = x_1$ and $n_2 = x_3$, as well as one parallel derivative $p_1 = x_2$. Written in this notation the *basic system* (3.17) reads

$$\frac{\partial \boldsymbol{U}}{\partial t} + \sum_{j=1}^{N_n} \boldsymbol{A}_{n_j} \frac{\partial \boldsymbol{U}}{\partial n_j} + \sum_{j=1}^{N_p} \boldsymbol{A}_{p_j} \frac{\partial \boldsymbol{U}}{\partial p_j} = \boldsymbol{0}.$$

The information (3.18) is sufficient to construct a characteristic system, analogue to (2.26):

$$\begin{aligned} \boldsymbol{R}_{n_1} \frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{\Lambda}_{n_1} \boldsymbol{R}_{n_1} \frac{\partial \boldsymbol{U}}{\partial n_1} \\ + \boldsymbol{R}_{n_1} \left[\sum_{j=2}^{N_n} \boldsymbol{A}_{n_j} \boldsymbol{R}_{n_1}^{-1} \boldsymbol{R}_{n_1} \frac{\partial \boldsymbol{U}}{\partial n_j} + \sum_{j=1}^{N_p} \boldsymbol{A}_{p_j} \boldsymbol{R}_{n_1}^{-1} \boldsymbol{R}_{n_1} \frac{\partial \boldsymbol{U}}{\partial p_j} \right] = \boldsymbol{0}. \end{aligned} \tag{3.19}$$

Hereby, only one of the orthogonal derivatives is written in characteristic form. In- and outgoing waves with respect to n_1-direction can be separated by inspecting the sign of the eigenvalues (entries of $\boldsymbol{\Lambda}_{n_1}$), see Section 2.3.1. The outgoing waves can be expressed in terms of the known interior information. However, it is not possible to use the interior information to compute the incoming waves. If available, an exterior solution can be used at this. Hedstrom [44] and Thompson [108] proposed for NRBCs that incoming waves are annihilated. That is, the system (3.19) is modified by substituting $\boldsymbol{\Lambda}_{n_1}$ with a diagonal matrix $\widetilde{\boldsymbol{\Lambda}}_{n_1}$. In $\widetilde{\boldsymbol{\Lambda}}_{n_1}$ all eigenvalues (entries of $\boldsymbol{\Lambda}_{n_1}$) corresponding to an incoming wave are replaced by zero. The modified system is (left-)multiplied with $\boldsymbol{R}_{n_1}^{-1}$ to get a system

in the original variables:

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{R}_{n_1}^{-1}\widetilde{\boldsymbol{\Lambda}}_{n_1}\boldsymbol{R}_{n_1}\frac{\partial \boldsymbol{U}}{\partial n_1} + \sum_{j=2}^{N_n}\boldsymbol{A}_{n_j}\frac{\partial \boldsymbol{U}}{\partial n_j} + \sum_{j=1}^{N_p}\boldsymbol{A}_{p_j}\frac{\partial \boldsymbol{U}}{\partial p_j} = \boldsymbol{0}.$$

Based on this system, a characteristic analysis is successively repeated for the remaining orthogonal derivatives. Eventually, we get the *CBC system*

$$\frac{\partial \boldsymbol{U}}{\partial t} + \sum_{j=1}^{N_n}\widetilde{\boldsymbol{A}}_{n_j}\frac{\partial \boldsymbol{U}}{\partial n_j} + \sum_{j=1}^{N_p}\boldsymbol{A}_{p_j}\frac{\partial \boldsymbol{U}}{\partial p_j} = \boldsymbol{0}, \qquad \widetilde{\boldsymbol{A}}_{n_j} = \boldsymbol{R}_{n_j}^{-1}\widetilde{\boldsymbol{\Lambda}}_{n_j}\boldsymbol{R}_{n_j}, \tag{3.20}$$

which formulates a BC on the macroscopic level. The non-diagonal matrices $\widetilde{\boldsymbol{A}}_{n_j}$ are stated in Appendix B. Unlike (3.17), the *CBC system* (3.20) can be solved with interior information. We note that all coefficient matrices are time dependent, since mass density and velocity components are dynamic. We refer to (3.20) as the *perfectly non-reflecting CBC*, since it follows the aim to neglect any reflections. However, numerical experiments will show that in fact there are small reflections, for details see Chapter 6.

3.3.3 Solution of the CBC system

For an application to the LBM, the system which finally formulates a CBC on a macroscopic level (such as (3.20) in the case above), has to be solved in each boundary lattice point $\boldsymbol{x}_b$. There is no restriction on the procedure how to solve this system numerically, so the explanation below is only one possible realization.

In the *CBC system* different types of derivatives appear: We have a time derivative, at least one spatial derivative orthogonal to the boundary and possibly parallel spatial derivatives. Each type is treated differently. In a first step all spatial derivatives are discretized using finite differences, this gives the so-called *method of lines* approach. Doing so, the PDE system turns into a system of ODEs. For the parallel derivatives a centered finite difference stencil can be applied, such as (see, e.g., [88])

$$\frac{\partial z(\boldsymbol{x}_b, t)}{\partial p} \approx \frac{1}{2}\Big(z(\boldsymbol{x}_b + c\Delta t\boldsymbol{e}_p, t) - z(\boldsymbol{x}_b - c\Delta t\boldsymbol{e}_p, t)\Big). \tag{3.21}$$

Here z is an arbitrary quantity and p is a parallel direction (i.e., x_1, x_2 or x_3, depending on the boundary orientation). The vector $\boldsymbol{e}_p$ is the unit vector in positive p-direction. We note that there is a lattice vector $\boldsymbol{c}_j = c\boldsymbol{e}_p$, such that $\boldsymbol{x}_b \pm c\Delta t\boldsymbol{e}_p$ refer to the nearest lattice points in positive and negative p-direction, respectively. However, for the orthogonal derivatives with respect to n ($n \in \{x_1, x_2, x_3\}$) a centered difference stencil cannot be applied, due to a lack of adjacent lattice points. Instead we have to use a one-sided finite differences. As above $\boldsymbol{e}_n$ is the unit vector in (positive) n-direction. Let $\boldsymbol{n}$ be the corresponding unitary normal vector of the boundary, pointing outside the computational domain. Then the orthogonal derivatives of the *CBC system* can be discretized with a second order one-sided finite difference [88]:

$$\frac{\partial z(\boldsymbol{x}_b, t)}{\partial n} \approx \frac{\langle \boldsymbol{n},\, \boldsymbol{e}_n\rangle}{2}\Big(-3z(\boldsymbol{x}_b, t) + 4z\big(\boldsymbol{x}_b - \tilde{\boldsymbol{e}}_n, t\big) - z\big(\boldsymbol{x}_b - 2\tilde{\boldsymbol{e}}_n, t\big)\Big) \tag{3.22}$$

with $\tilde{\boldsymbol{e}}_n = c\langle \boldsymbol{n}, \boldsymbol{e}_n\rangle \Delta t \boldsymbol{e}_n$. The inner product $\langle \boldsymbol{n}, \boldsymbol{e}_n\rangle$ of $\boldsymbol{n}$ and $\boldsymbol{e}_n$ is either -1 or 1. As above, z is an arbitrary quantity. At edges or corners the unitary normal vector $\boldsymbol{n}$ is not uniquely defined. In (3.22) the only role of $\boldsymbol{n}$ is to specify a sign, in order to approximate the derivatives correctly. Therefore, we can easily assume to have several unit normal vectors, one for each orthogonal derivative. For instance, at a corner node in two-dimensional problems the $\frac{\partial}{\partial x_1}$-terms would take $\boldsymbol{n} = (\pm 1, 0)^\top$, while the $\frac{\partial}{\partial x_2}$-terms would take $\boldsymbol{n} = (0, \pm 1)^\top$ instead. This vector is always chosen, such that it points outside the computational domain.

With the finite differences (3.21) and (3.22) applied to the spatial derivatives of the *CBC system*, the latter turns into a system of ODEs

$$\frac{\partial \boldsymbol{U}}{\partial t}(\boldsymbol{x}_b, t) = \boldsymbol{h}\left(\boldsymbol{U}, t\right), \tag{3.23}$$

one ODE for each boundary lattice point. They have to be integrated from time level $t = t_s$ to level $t = t_{s+1} = t_s + \Delta t$. The right hand side function $\boldsymbol{h}$ depends on the solution itself, as well as on quantities at adjacent lattice points. On the one hand, the latter involve interior lattice points, resulting from finite difference stencils for orthogonal derivatives. On the other hand, for $d \geq 2$ the function $\boldsymbol{h}$ in general depends also on adjacent boundary lattice points, which enter via the finite differences of parallel derivatives. The latter couples the systems for each boundary points. It is also noteworthy that macroscopic quantities at interior lattice points are not continuously given, but only discrete in time. If evaluations of these are required for intermediate time points $t \in (t_s, t_{s+1})$, the values could be approximated by interpolation. Lastly, we note that $\boldsymbol{U}(\boldsymbol{x}_b, t_s)$ is known and is used as initial condition for (3.23). The ODEs (3.23) for all boundary points $\boldsymbol{x}_b$ are solved, e.g., by using one *explicit Euler* step [37]

$$\boldsymbol{U}(t_{s+1}) = \boldsymbol{U}(t_s) + \Delta t \boldsymbol{h}\left(\boldsymbol{U}(t_s), t_s\right). \tag{3.24}$$

Using an explicit Euler method, no interpolation of interior macroscopic quantities is needed. Alternatively, other methods can be used, such as Runge-Kutta methods. For integration of (3.23) Thompson [108] proposed the *explicit second order Runge-Kutta scheme* corresponding to a *Butcher tableau* as follows:

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 1/4 & 1/4 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \\ \hline & 0 & 0 & 0 & 1 \end{array} \tag{3.25}$$

Eventually, the integration of (3.23) yields a Dirichlet BC which has to be transferred to the LBM, i.e., to the boundary populations.

3.3.4 Determination of boundary populations

By the steps above, the complete CBC is reduced to the final task of computing the populations, such that a Dirichlet BC is applied. More precisely, this is a Dirichlet condition for the mass density and all velocity components. We recall that only if the Dirichlet

values satisfy the relation (3.4), the condition can be implemented by computing only the unknown populations. The transfer of a general Dirichlet condition for both the mass density and normal velocity requires to recompute also given populations. In more detail, the numerical solution of (3.23) yields Dirichlet values denoted by ρ_D and $\boldsymbol{u}_D$. The simplest strategy to transfer the condition is by evaluating the equilibrium distribution

$$\begin{aligned} B_i(\boldsymbol{x}_b, t_{s+1}) &:= f_i^{\mathrm{eq}}(\boldsymbol{x}_b, t_{s+1}) \\ &= \rho_D(\boldsymbol{x}_b) w_i \left[1 + \frac{3\langle \boldsymbol{c}_i,\, \boldsymbol{u}_D(\boldsymbol{x}_b)\rangle}{c^2} + \frac{9\langle \boldsymbol{c}_i,\, \boldsymbol{u}_D(\boldsymbol{x}_b)\rangle^2}{2c^4} - \frac{3\,|\boldsymbol{u}_D(\boldsymbol{x}_b)|^2}{2c^2} \right], \end{aligned} \tag{3.26}$$

for all $i = 0, \ldots, q-1$. We refer to this procedure as *equilibrium boundary condition* (EBC). It is the equilibrium portion of the populations which determines the macroscopic quantities (2.42). Thus, compared to any other approach which transfers a Dirichlet condition the populations will differ only by their non-equilibrium part

$$f_i^{\mathrm{neq}}(\boldsymbol{x}, t) = f_i(\boldsymbol{x}, t) - f_i^{\mathrm{eq}}(\boldsymbol{x}, t).$$

Several possibilities arise for computing the f_i^{neq} at the boundary nodes. Following the idea in [125] the non-equilibrium part at the boundary node can be determined by extrapolation. Thus, we obtain

$$B_i(\boldsymbol{x}_b, t_{s+1}) = f_i^{\mathrm{eq}}(\boldsymbol{x}_b, t_{s+1}) + \alpha f_i^{\mathrm{neq}}(\boldsymbol{x}_b - \tilde{\boldsymbol{e}}_n, t_{s+1}) + \beta f_i^{\mathrm{neq}}(\boldsymbol{x}_b - 2\tilde{\boldsymbol{e}}_n, t_{s+1}), \tag{3.27}$$

with $\tilde{\boldsymbol{e}}_n$ as in (3.22). We have a linear extrapolation by the choice $\alpha = 2$ and $\beta = -1$. By $\alpha = 1$ and $\beta = 0$ we obtain a constant extrapolation of the non-equilibrium part. In both cases the extrapolation is performed normal to the boundary. Here, for a corner lattice node or an edge node we assume the normal direction $\boldsymbol{n}$ (used in $\tilde{\boldsymbol{e}}_n$) to be the diagonal direction. Note that the non-equilibrium parts of interior lattice points can be computed at the new time level, since the necessary information is known. We denote the BC (3.27) as *non-equilibrium boundary condition* (nEBC) in the sequel.

3.4 Enhanced characteristic boundary conditions

Compared to literature, the novelty of the CBC presentation in the previous section lies on the three-dimensional generalization, each for the LODI based CBC and also of our CBC. In this section we further present novel enhancements for CBCs in the LBM. Except for the last one they can be combined with each other.

3.4.1 Incorporation of viscous terms

In what follows we consider the CBC, whose *basic* and *CBC system* is given by (3.17) and (3.20), respectively. Recall that for an implementation of the CBC in the LBM the BC formulated as (3.20) is transferred to the discrete populations in several steps. The *CBC system* is first solved numerically and the resulting Dirichlet condition is used to determine the boundary populations for the LB simulation. In this complete procedure several error sources can be identified. Errors are created in the numerical solution of the *CBC system*, however they can be tackled by using other procedures (e.g., implicit methods). Also errors

in the determination of populations, i.e., in implementing the Dirichlet condition are done. They can be reduced by techniques computing non-equilibrium parts, e.g., (3.27), but even more by use of advanced techniques [112, 114]. Furthermore, the *basic system* of a CBC and the *CBC system* already differ by terms

$$\sum_{i=1}^{N_n} \left(\boldsymbol{A}_{d,n_i} - \widetilde{\boldsymbol{A}}_{d,n_i} \right) \frac{\partial \boldsymbol{U}_d}{\partial n_i}.$$

The *basic system* (3.17) fits nicely to the Euler equations, however the interior fluid model for the LB simulation is given by (3.13). Thus, another error source of the CBC lies in neglecting the deviatoric stress terms

$$\boldsymbol{\nabla} \cdot \rho\nu \left[\boldsymbol{\nabla}\boldsymbol{u} + \boldsymbol{\nabla}\boldsymbol{u}^\top - \frac{2}{3}(\boldsymbol{\nabla} \cdot \boldsymbol{u})\boldsymbol{I} \right]$$

in the momentum equations, where ν is the kinematic viscosity (2.43). This term has to be neglected in the *basic system*, since otherwise the equations would not be hyperbolic and the characteristic analysis could not be performed. Following the idea of [84], we reincorporate viscous terms after the characteristic analysis is done. More precisely, the *CBC system* (3.20) formulating the CBC turns into the adapted version

$$\frac{\partial \boldsymbol{U}}{\partial t} + \sum_{j=1}^{N_n} \widetilde{\boldsymbol{A}}_{n_j} \frac{\partial \boldsymbol{U}}{\partial n_j} + \sum_{j=1}^{N_p} \boldsymbol{A}_{p_j} \frac{\partial \boldsymbol{U}}{\partial p_j} = \boldsymbol{\tau}, \tag{3.28}$$

with

$$\boldsymbol{\tau} = \frac{\Delta t}{3} \left(\frac{1}{\omega} - \frac{1}{2} \right) c^2 \begin{pmatrix} 0 \\ \boldsymbol{\nabla} \cdot \left[\boldsymbol{\nabla}\boldsymbol{u} + \boldsymbol{\nabla}\boldsymbol{u}^\top - \frac{2}{3}(\boldsymbol{\nabla} \cdot \boldsymbol{u})\boldsymbol{I} \right] \end{pmatrix}.$$

Like (3.20), the extended CBC formulation (3.28) is solved numerically and corresponding populations are calculated with its solution, by implementing a Dirichlet condition as described above. We call this extended version the *viscous CBC* (vCBC).

3.4.2 Inlet and outlet characteristic boundary conditions

This section deals with two issues, both related to the work of Poinsot and Lele [84] for direct Navier-Stokes solvers. First, the explanation below states a strategy for the computation of the pressure or normal velocity when the other is specified (paragraph b). Therefore we need to recompute known populations in boundary lattice points, meaning the relation (3.4) can be disregarded. Let us recall, that unless known populations are recomputed, the relation only allows to choose parallel velocity components, when a pressure or a normal velocity Dirichlet condition in the LBM is applied. Secondly, we present a modification which combines the possibility to specify an average pressure and to have non-reflective properties at the boundary (paragraph c).

a) Introduction of wave amplitude variations

For the sake of simplicity we consider standard boundaries having only one orthogonal derivative, i.e., $N_n = 1$. Let x_k be the only orthogonal direction, meaning $n_1 = x_k$ holds. The following explanation is based on so-called *wave amplitude variations* $\mathcal{L}_{x_k,j}$ (also simply called *amplitudes* in the following) defined in vector notation by

$$\boldsymbol{\mathcal{L}}_{x_k} = \begin{pmatrix} \mathcal{L}_{x_k,1} \\ \vdots \\ \mathcal{L}_{x_k,d+1} \end{pmatrix} = \boldsymbol{\Lambda}_{x_k} \boldsymbol{R}_{x_k} \frac{\partial \boldsymbol{U}}{\partial x_k}.$$

Note the relation to the *basic system* as follows $\boldsymbol{R}_{x_k}^{-1} \boldsymbol{\mathcal{L}}_{x_k} = \boldsymbol{A}_{x_k} \frac{\partial \boldsymbol{U}}{\partial x_k}$. The amplitudes are independent of the dimension d and given by:

$$\begin{aligned}
\mathcal{L}_{x_k,1} &= (u_k - c_s) \left(c_s^2 \frac{\partial \rho}{\partial x_k} - c_s \rho \frac{\partial u_k}{\partial x_k} \right), \\
\mathcal{L}_{x_k,2} &= (u_k + c_s) \left(c_s^2 \frac{\partial \rho}{\partial x_k} + c_s \rho \frac{\partial u_k}{\partial x_k} \right), \\
\mathcal{L}_{x_k,j+2} &= u_k \frac{\partial u_{p_j}}{\partial x_k}, \qquad\qquad j = 1, \ldots, d-1.
\end{aligned}$$

Slightly different in notation, here the indices $p_j \in \{1, \ldots, d\} \setminus \{k\}$ refer to the parallel components. Each amplitude is linked to a direction via its corresponding eigenvalue. Only outward amplitudes can be computed from interior information, see also the previous Section 3.3. The perfectly non-reflecting CBC (3.20) is equivalent to set inward wave amplitude variations to zero. A tilde denotes the modified wave amplitude variations, which for the perfectly non-reflecting CBC exhibits the form

$$\widetilde{\boldsymbol{\mathcal{L}}}_{x_k} = \begin{pmatrix} \widetilde{\mathcal{L}}_{x_k,1} \\ \vdots \\ \widetilde{\mathcal{L}}_{x_k,d+1} \end{pmatrix}, \qquad \widetilde{\mathcal{L}}_{x_k,i} = \begin{cases} \mathcal{L}_{x_k,i} & \text{outgoing} \\ 0 & \text{incoming} \end{cases}. \tag{3.29}$$

The perfectly non-reflecting CBC (3.20) for $N_n = 1$ and $n_1 = x_k$ is equivalently written with amplitudes by

$$\frac{\partial \boldsymbol{U}}{\partial t} + \sum_{j=1}^{d-1} \boldsymbol{A}_{p_j} \frac{\partial \boldsymbol{U}}{\partial p_j} = -\boldsymbol{R}_{x_k}^{-1} \widetilde{\boldsymbol{\mathcal{L}}}_{x_k}. \tag{3.30}$$

b) Relations of wave amplitude variations

Instead of setting inward amplitudes to zero, as done in (3.29), we are looking for other reasonable strategies. To this end we consider the x_k-LODI equations, which serve for finding relations of inward and outward wave amplitude variations. The x_k-LODI equations in terms of wave amplitude variations read

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{R}_{x_k}^{-1} \boldsymbol{\mathcal{L}}_{x_k} = 0,$$

see also (3.14) for an alternative formulation. This system represents the scalar equations:

$$\frac{\partial \rho}{\partial t} + \frac{1}{2c_s^2}\big(\mathcal{L}_{x_k,1} + \mathcal{L}_{x_k,2}\big) = 0, \tag{3.31a}$$

$$\frac{\partial u_k}{\partial t} + \frac{1}{2c_s\rho}\big(\mathcal{L}_{x_k,2} - \mathcal{L}_{x_k,1}\big) = 0, \tag{3.31b}$$

$$\frac{\partial u_{p_j}}{\partial t} + \mathcal{L}_{x_k,j+2} = 0, \qquad j = 1, \dots, d-1. \tag{3.31c}$$

Both (3.31a) and (3.31b) contain an inward and an outward wave amplitude variation. Whenever either the density (respectively the pressure) or normal velocity is prescribed at the boundary, one of this two equations can be used to explicitly write the inward wave amplitude variation. As an example let us consider the situation with a given normal velocity for a boundary for which x_k is the inner normal, then $\mathcal{L}_{x_k,2}$ is an inward amplitude. Instead of setting $\widetilde{\mathcal{L}}_{x_k,2}$ to zero, as proposed by (3.29), we compute the inward amplitude by

$$\widetilde{\mathcal{L}}_{x_k,2} = \mathcal{L}_{x_k,1} - 2c_s\rho\frac{\partial u_k}{\partial t}. \tag{3.32}$$

Moreover, in the resulting system we can omit all equations which correspond to prescribed quantities. The remaining equations are solved with the amplitudes computed as in (3.32). The amplitudes $\mathcal{L}_{x_k,3}$ to $\mathcal{L}_{x_k,d+1}$ for $d > 1$ need some further discussion: At outlets these amplitudes can be computed from interior information. For inlets, the procedure above requires the corresponding parallel velocity to be known such that (3.31c) can be used to estimate the amplitude. However, when the parallel velocity is prescribed, there is no need of the corresponding wave amplitude variation, since in (3.30) this amplitude enters only the equation corresponding to this velocity component and would be omitted anyway. Therefore, we can only either prescribe the parallel velocity or we set the corresponding amplitude to zero, which is analogue to the perfectly non-reflecting CBC.

c) Amplitudes for average quantities

The approach of the previous paragraph leading to modified amplitudes (3.32) requires knowledge of either the density or the normal velocity at the boundary. It turned out that in case of parallel velocities only a fixed velocity or a zero wave amplitude variation was possible. We now focus on the situation where no macroscopic information at the boundary is given and aim at finding modified amplitudes other than (3.29). The only condition we pose is to have an average density of ρ_∞ at the outlet. The inward wave amplitude is computed as a relaxation towards this value:

$$\widetilde{\mathcal{L}}_{x_k,\{1,2\}} = Kc_s^2\big(\rho - \rho_\infty\big), \tag{3.33}$$

where ρ is the best known value of the current density at the boundary. Note only one of $\widetilde{\mathcal{L}}_{x_k,1}$ and $\widetilde{\mathcal{L}}_{x_k,2}$ is computed, which is the inward amplitude. The value of K (unit s^{-1}) has a strong influence on the simulation results, for details see, e.g., [84]. It was suggested by Rudy and Strikwerda [91] to scale K as follows

$$K = \sigma\big(1 - \mathrm{Ma}^2\big)\frac{c_s}{L}$$

and in [94] a test case dependent range for the constant $\sigma \in [0.2, \pi]$ is proposed. Here L is the domain size and Ma the maximal Mach number of the flow. As above, (3.30) is solved with the amplitude (3.33) together with the unmodified outgoing amplitudes.

4 Chapter 4
Exact discrete artificial boundary condition for linear collision model

In this chapter we develop an exact BC for a LBM with a linear equilibrium distribution. We focus on the D1Q2 model, i.e., a model with discrete velocity space $\mathcal{V}_{\mathrm{D1Q2}} = \{-c, +c\}$. Thus, we consider a setup of the form

$$P_{\mathrm{lin}} := \mathtt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}_{\mathrm{D1Q2}}, \mathrm{C}_{\mathrm{lin}}, f_j, I_j, B_j^{\mathrm{ex}}), \tag{4.1}$$

with

$$\mathrm{C}_{\mathrm{lin}} = \frac{\omega}{2} \begin{pmatrix} -(a+1) & -(a-1) \\ a+1 & a-1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}, \quad (\omega, a) \in (0,\, 2) \times (-1,\, 1). \tag{4.2}$$

This collision term $\mathrm{C}_{\mathrm{lin}}$ is equivalent to a BGK collision with the linear equilibrium (2.44). The developed BC of this chapter is given in a fully discrete formulation, i.e., all computations are done on the discrete level of the given LB simulation. In contrast to many common BCs for the LBM, neither computations on the macroscopic level nor macroscopic quantities are required at all. Our derived BC is exact in the sense of an ITBC, see Section 3.1. However, an increasing computational effort when time evolves, makes a real application of the BC non-practical. The main focus of the development therefore does not lie in the formulation of a practical BC, but in the preparation for an approximate *discrete artificial boundary condition* (DABC). This approximate DABC will be the topic of the subsequent chapter. Thereby, the restriction to linear collision models will be relaxed.

In Section 4.1 we explain the use of digraphs for describing the evolution of populations. Afterwards, the second section applies this novel perspective to P_{lin}, which results in weighted digraphs. The weighted digraphs are used in Sections 4.3 and 4.4, in which we first discuss the construction idea of the exact DABC and afterwards present its outcome. Sections 4.2 to 4.4 mainly follow our work [47], whereas the main part of Section 4.1 is based on our work [48]. In addition to the content of the articles, we here provide complete proofs of all lemmas.

4.1 Evolution represented with digraphs

Here we introduce digraphs to express any population in dependence on past information. Thereby, the past information need not be given necessarily as a population from the previous time level. This general approach is used in the subsequent section for the linear collision model of the problem setup (4.1). In the sequel, a *node* $n = (\boldsymbol{x}, t)$ refers to a pair consisting of a lattice point $\boldsymbol{x} \in \mathcal{G}$ and a time $t \in \mathcal{T}$. Nodes are denoted by n throughout, possibly equipped with super- and subscripts.

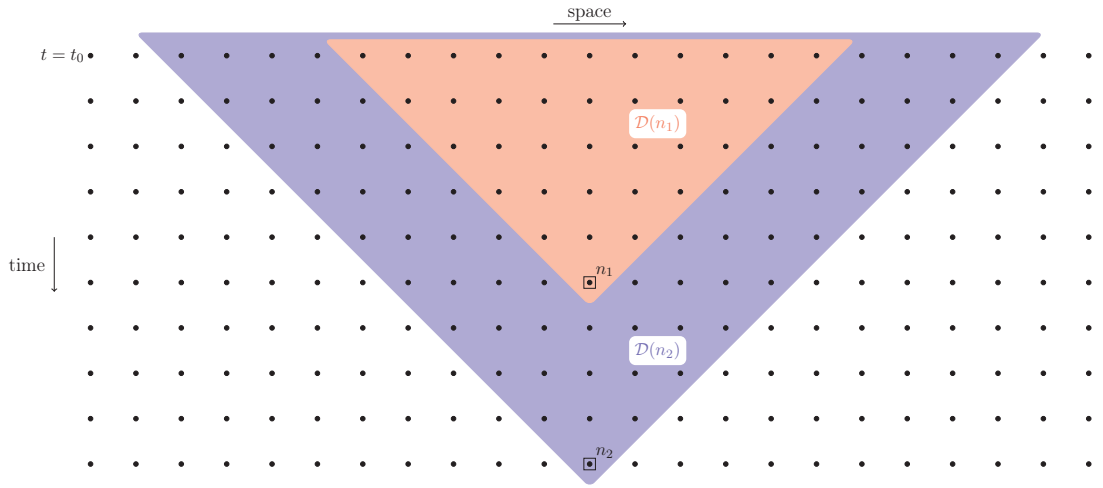


Figure 4.1: *Domains of dependence for two nodes in a space-time diagram* – The space-time diagram shows the domains of dependence $\mathcal{D}(n_1)$ and $\mathcal{D}(n_2)$ for two nodes n_1 and n_2, respectively.

4.1.1 Domain of dependence

In general, each population $f_j(\boldsymbol{x}, t)$ in a node $n = (\boldsymbol{x}, t)$ takes into account information from previous nodes up to the initial time. The *domain of dependence* $\mathcal{D}(n)$, for a certain node n comprises all nodes from previous time levels, which affects the populations at node n. $\mathcal{D}(n)$ is increasing with time, in the sense that the number of contributing nodes from initial time $t = t_0$ is increasing with every time step which is simulated, see the discrete space-time diagram shown in Fig. 4.1. Here the domains of dependence for two nodes n_1 and n_2 having the same location $\boldsymbol{x}$, but different time levels are shown. It is important to clarify that such a dependence of nodes from previous time levels in particular also holds for a boundary point. Considering the case where the unknown populations are computed with an ITBC shows that besides interior nodes up to the initial time also exterior information should be considered.

The update rule of the LBM specifies a formula for a population in terms of populations from the directly previous time level. Nonetheless, due to its iterative application, populations depend on all information from the domain of dependence. In other words, any population which can be computed with (2.40) generally contains also information from all time levels prior to the previous one.

4.1.2 Lattice Boltzmann equation with reversed perspective

We rewrite the LBE, see also (2.40), in a goal oriented version:

$$f_j(\boldsymbol{x}, t) = f_j(\boldsymbol{x} - \boldsymbol{c}_j \Delta t, t - \Delta t) + \mathrm{C}\big(\boldsymbol{f}(\boldsymbol{x} - \boldsymbol{c}_j \Delta t, t - \Delta t)\big), \quad j = 0, \ldots, q-1, \tag{4.3}$$

where q gives the number of lattice vectors. Here $\mathrm{C}(\boldsymbol{f})$ is an arbitrary collision model and $\boldsymbol{f}(\boldsymbol{x}, t) = \big(f_j(\boldsymbol{x}, t)\big)_{j=0,\ldots,q-1}$ is the vector of all populations in a given node. Recapitulate that the equilibrium distribution (2.39) used above in the BGK collision model can be

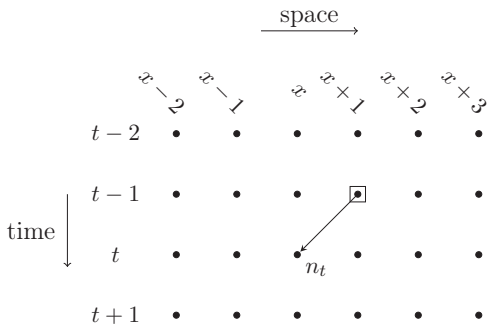


Figure 4.2: *Population symbolized by an arrow* – The arrow shown in the space-time diagram symbolizes the population $f_1(x,t)$. It depends on information from the square node.

written purely in populations, since the fluid quantities (2.42) are functions of populations:

$$\rho(\boldsymbol{x},t) = \rho\big(\boldsymbol{f}(\boldsymbol{x},t)\big), \qquad \boldsymbol{u}(\boldsymbol{x},t) = \boldsymbol{u}\big(\boldsymbol{f}(\boldsymbol{x},t)\big).$$

The complete right hand side of (4.3) therefore depends solely on $\boldsymbol{f}$. By introducing post-collision populations g_i,

$$g_i\big(\boldsymbol{f}(\boldsymbol{x},t)\big) := f_i(\boldsymbol{x},t) + \mathrm{C}\big(\boldsymbol{f}(\boldsymbol{x},t)\big), \tag{4.4}$$

each population determined by (4.3) is given by

$$f_i(\boldsymbol{x},t) = g_i\big(\boldsymbol{f}(\boldsymbol{x}-\boldsymbol{c}_i\Delta t, t-\Delta t)\big) \tag{4.5a}$$
$$= g_i\big(f_0(\boldsymbol{x}-\boldsymbol{c}_i\Delta t, t-\Delta t), \ldots, f_{q-1}(\boldsymbol{x}-\boldsymbol{c}_i\Delta t, t-\Delta t)\big). \tag{4.5b}$$

This equation represents a generalized LBE. Obviously, it gives a dependence from the previous time level. However, for constructing a BC it would be helpful to have an alternative, which helps to better understand the contribution of specific populations from the domain of dependence, i.e., from arbitrary past time levels.

4.1.3 Visualization by digraphs

We use digraphs in a space-time diagram to track the influence of certain populations. The statements and explanations in the following of this section hold for any dimension, any discretization model and any collision model. Note, in contrast to other models for D1Q2 the enumeration of velocities begins with index 1. For the sake of a better clarity the visualized space-time diagrams however only correspond to the one-dimensional LB models D1Q2 and D1Q3, respectively. Exemplarily, we visualize the computation of a population f_1 according to (4.5) for a one-dimensional model by an arrow in a space-time diagram, shown in Fig. 4.2. The arrow symbolizes the transport of the post-collision population in an intuitive way. For a better readability in all figures, the labels are written under the assumption $\Delta t = 1$ and $c = 1$. We interpret the arrow as a digraph and use the common terminology of graph theory as given next. For more details on graph theory see, e.g., [22].

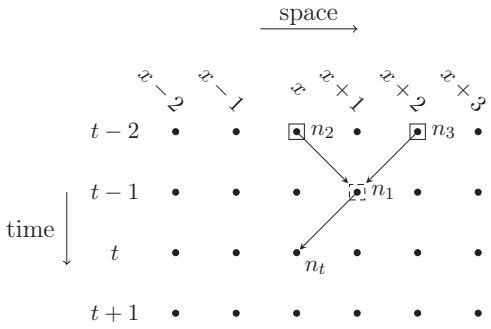


Figure 4.3: *Population represented with a digraph* – Each edge of the digraph visualizes the transport of a population. The population $f_1(x,t)$ in the terminal node n_t depends on information from contributing nodes (squares). Nodes n_2 and n_3 are contributing nodes for both D1Q2 and D1Q3, whereas the dashed square node n_1 is a contributing node only for D1Q3.

Terminology An arrow as shown in Fig. 4.2 is called an *edge*. The node at its tip (node $n_t = (x,t)$ in Fig. 4.2) is called *head* and the node at its source (node $n = (x+1, t-1)$ in Fig. 4.2) is called *tail*. A *path* of length m is a sequence of edges e_1 to e_m, for which the tail of e_j equals the head of e_{j-1}, $j = 2, \ldots, m$. The path starts in the tail of e_1 and ends in the head of e_m. The *in-degree* of a node is given as the number of edges which end in this node. Analogously, the *out-degree* is the number of edges which begin in the node. If there is a path from a node A to a node B, then the node B is called a *successor* of node A. Vice versa, node A is a *predecessor* of node B. Additionally, we call the final node, which has in-degree 1 and out-degree 0, the *terminal node*, in the following always denoted as n_t. Furthermore, all predecessors of the terminal node, which have an in-degree less than q are called *contributing nodes*. The populations corresponding to missing edges in contributing nodes are called *contributing populations*. Moreover for D1Q2 directional terms can be defined: An edge is said to be *leftward*, if the space coordinates of its head x_H and its tail x_T satisfy $x_H = x_T - c\Delta t = x_T - 1$. Analogously, the edge is *rightward* if the equation $x_H = x_T + c\Delta t = x_T + 1$ holds.

Populations represented with digraphs In our interpretation, an edge in a space-time diagram symbolizes a pre-collision population in its head. Consequently, the maximal in-degree of a node is equal to the number of discrete velocities q. Each edge visualizes the transport of a post-collision population from its tail to its head. Note that after a transport step, a post-collision population turns into a pre-collision population. Thus, for each edge a local collision at its tail is implied. For contributing nodes, each missing inward edge shall be equivalent to the knowledge of the corresponding population, i.e., of the contributing population. Therefore, if a node has in-degree zero, we assume all populations are known in this node. To illustrate this further, we consider the digraph in Fig. 4.3 for both the D1Q2 and the D1Q3 model. In both models the populations f_1 and f_2 at node $n_1 = (x + c\Delta t, t - \Delta t) = (x+1, t-1)$ are computed from $\boldsymbol{f}(x, t-2)$ and $\boldsymbol{f}(x+2, t-2)$, respectively. Thus, for D1Q2 all populations are unknown at node n_1. In contrast, for D1Q3 n_1 is a contributing node, where the missing edge from node $(x+1, t-2)$ to n_1 means that the population $f_0(x+1, t-1)$ is known, hence it is a contributing population. Note, such an edge would not be reasonable for the D1Q2 model. Moreover, all populations in nodes $n_2 = (x, t-2)$ and $n_3 = (x+2, t-2)$ are

known.
Certainly, not every digraph is reasonable. For each edge the head has to be on the very next time level compared to its tail. Also an edge is only reasonable if there is a lattice vector connecting its tail's with its head's space coordinate.

Computation of the population in the terminal node Given a digraph, the final population at the terminal node n_t can be computed with information (contributing populations) from all contributing nodes. The corresponding formula is a composition of functions (4.5b), which is obtained by traversing the digraph backwards when starting in the terminal node. The inward edge at the terminal node n_t corresponds to a population $f_j(n_t)$, expressed as a function g_j evaluated at an adjacent node, see (4.5b). Each argument of this function g_j is either a population f_m, taken if it is known, or otherwise, it is a function g_m itself. The arguments of this function itself are again chosen under consideration of the next edge. This approach terminates in all cases, at latest when nodes from initial time $t = t_0$ are reached (here the in-degree is always zero). As an example, the digraph of Fig. 4.3 would result in the formula (for D1Q3)

$$f_1(x,t) = g_1\big(f_0(x + c\Delta t, t - \Delta t), g_1\big(\boldsymbol{f}(x + 2c\Delta t, t - 2\Delta t)\big), g_2\big(\boldsymbol{f}(x, t - 2\Delta t)\big)\big). \quad (4.6)$$

4.2 Weighted digraphs for linear collision models

The general digraph interpretation leads to compositions of post-collision functions, e.g., (4.6), which in general are expensive to evaluate. This is mainly due to the nonlinearity of the collision model and hence of the post-collision functions g_j. To compensate this difficulty in the remainder of the current chapter, we consider the linear collision model $\mathrm{C}_{\mathrm{lin}}$ (4.2) for the D1Q2 discretization corresponding to (4.1).

For the setup at hand the post-collision populations (4.4) are

$$g_1(x,t) = (1+\alpha) f_1(x + c\Delta t, t - \Delta t) + \beta f_2(x + c\Delta t, t - \Delta t), \quad (4.7a)$$
$$g_2(x,t) = \gamma f_1(x - c\Delta t, t - \Delta t) + (1+\delta) f_2(x - c\Delta t, t - \Delta t), \quad (4.7b)$$

where the four weights follow from the linear equilibrium (2.44):

$$\alpha = -\frac{1}{2}\omega(1+a), \qquad \beta = \frac{1}{2}\omega(1-a), \qquad \gamma = \frac{1}{2}\omega(1+a), \qquad \delta = -\frac{1}{2}\omega(1-a). \quad (4.8)$$

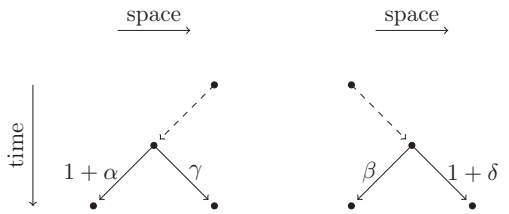
Node weights Due to the linearity of (4.7) any composition of post-collision functions remains a linear function, such that a population in the terminal node is expressed by

$$f_i(x,t) = \sum_{n_k \in \mathcal{N}} \big(W_1(n_k) f_1(n_k) + W_2(n_k) f_2(n_k)\big), \quad (4.9)$$

where $\mathcal{N}$ is the set of all contributing nodes. A *node weight* $W_j(n_k)$ is zero, if $f_j(n_k)$ is not a contributing population, i.e., if the digraph has an inward edge at n_k corresponding to f_j. Applying the strategy explained in the previous section, it follows that the non-zero node weights $W_j(n_k)$ depend on every path p_m^k, $m = 1, \ldots, \ell_k$, from n_k to the terminal

previous \ current	leftward	rightward
leftward	$1+\alpha$	γ
rightward	β	$1+\delta$

(a) Table defining the factors



(b) (c)

Figure 4.4: *Edge weights for D1Q2 with linear collision model* – The edge weights for the weighted digraphs used to express a population in the D1Q2 discretization with linear collision operator. The weights depend on the direction of the current edge as well as on the direction of the previous edge. Weights β and γ occur when the path has a directional change, while $(1+\alpha)$ and $(1+\delta)$ define the edge weight if there is no change in direction.

node n_t, where ℓ_k is the number all paths from n_k to n_t. Then the non-zero node weights are given as the sum of *path weights* $\widetilde{w}_j$:

$$W_j(n_k) = \sum_{m=1}^{\ell_k} \widetilde{w}_j(n_k, p_m^k). \tag{4.10}$$

Weighted digraphs Obviously as a composition of functions (4.7) each path weight in (4.10) is a product of $(1+\alpha)$, β, γ and $(1+\delta)$, where each edge of the path contributes exactly one factor, a so-called *edge weight*. To explain the choice of a weight for a certain edge (the *current edge*) we use the directional terms *leftward* and *rightward*, as explained above. We recapitulate that a leftward edge visualizes the transport of a g_1 population, while a rightward edge shows a moving g_2 population. As g_1, see (4.7a), only contains the coefficients $(1+\alpha)$ and β, they are the only possible edge weights if the current edge is leftward. We further conclude that $(1+\alpha)$ is used as coefficient for the f_1 population, i.e., if a f_1 population leaves a node as a population of same direction. In the perspective of a path: when both the current edge and its previous edge are leftward. Contrary, the edge weight β appears if the previous edge was rightward, that is, the path has a directional change. Analogously, a rightward edge has edge weight $(1+\delta)$, if the previous edge was also rightward, whereas γ is used, when the path changes its direction from left to right. The first edge of a path has no real previous edge, but we can add a fictitious one where its direction is determined whether a node weight for $j=1$ or for $j=2$ is computed. The complete situation is summarized in Fig. 4.4. The dashed edge is the previous (possibly fictitious) edge. If we equip each edge with a weight, then the digraph turns into a *weighted digraph*. An example of a thus obtained weighted digraph is shown in Fig. 4.5. It is important to see that we obtain weighted digraphs for each j in (4.10), however, they differ only by the first edge weight.

Computation of path weights Given a weighted digraph, the path weights $\widetilde{w}_j$ are computed by multiplying all edge weights of the path. The following corollary summarizes their computation.

Corollary 1. *Let be given one path p^k from a contributing node n_k to the terminal node n_t. The path shall have the property $\bar{\boldsymbol{p}} = (\bar{c}_l, \bar{c}_r, \bar{s}_l, \bar{s}_r)$, where*

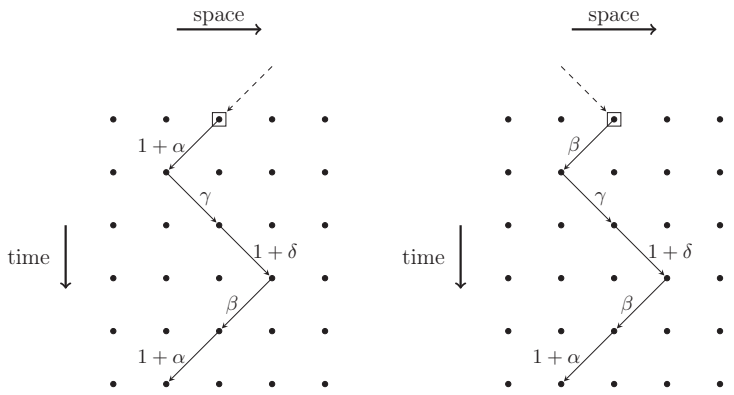


Figure 4.5: *Two weighted digraphs* – The fictitious previous edge (dashed arrow) of the starting edge is assumed to be leftward in the weighted digraph on the left hand side. This assumption is done when computing a path weight $\widetilde{w}_1$. In the weighted digraph on the right hand side, the fictitious previous edge is rightward, as necessary in the computation of a path weight $\widetilde{w}_2$.

- $\bar{c}_l$ *gives the number of changes from right to left,*
- $\bar{c}_r$ *gives the number of changes from left to right,*
- $\bar{s}_l$ *is the number of leftward edges, for which the previous edge was also leftward,*
- $\bar{s}_r$ *is the number of rightward edges, for which the previous edge was also rightward.*

The fictitious previous edge of the first edge is chosen according to the following rule: It is leftward in the computation of $\widetilde{w}_1$*, and it is rightward for* $\widetilde{w}_2$*. Then, the path weight corresponding to a contributing population* $f_j(n_k)$ *is given by:*

$$\widetilde{w}_j(n_k, p^k) = \bar{w}_j(\bar{\boldsymbol{p}}) := (1+\alpha)^{\bar{s}_l} \cdot \beta^{\bar{c}_l} \cdot \gamma^{\bar{c}_r} \cdot (1+\delta)^{\bar{s}_r}. \tag{4.11}$$

If we apply rule (4.11) to the digraph of Fig. 4.5 we get

$$\widetilde{w}_1 = (1+\alpha)^2 \cdot \beta \cdot \gamma \cdot (1+\delta), \qquad \widetilde{w}_2 = (1+\alpha) \cdot \beta^2 \cdot \gamma \cdot (1+\delta).$$

Eventually, when considering all paths from a certain node the node weights (4.10) can be computed.

4.3 Construction principle of the exact discrete artificial boundary condition

The (weighted) digraph view is now used to illustrate the construction of an exact BC. To this end we consider the boundary lattice point x_b. The task of a BC is to compute the inward directed population $f_j(x_b, t_k)$ for all time levels $t_k \in \widetilde{\mathcal{T}}$. We like to compute the unknown populations, such that they equal the ones of an ITBC. For the ITBC the information of $f_j(x_b, t_k)$ comes from an adjacent lattice point, which is not present in our case. We add fictitious lattice nodes in the exterior of the computational domain to

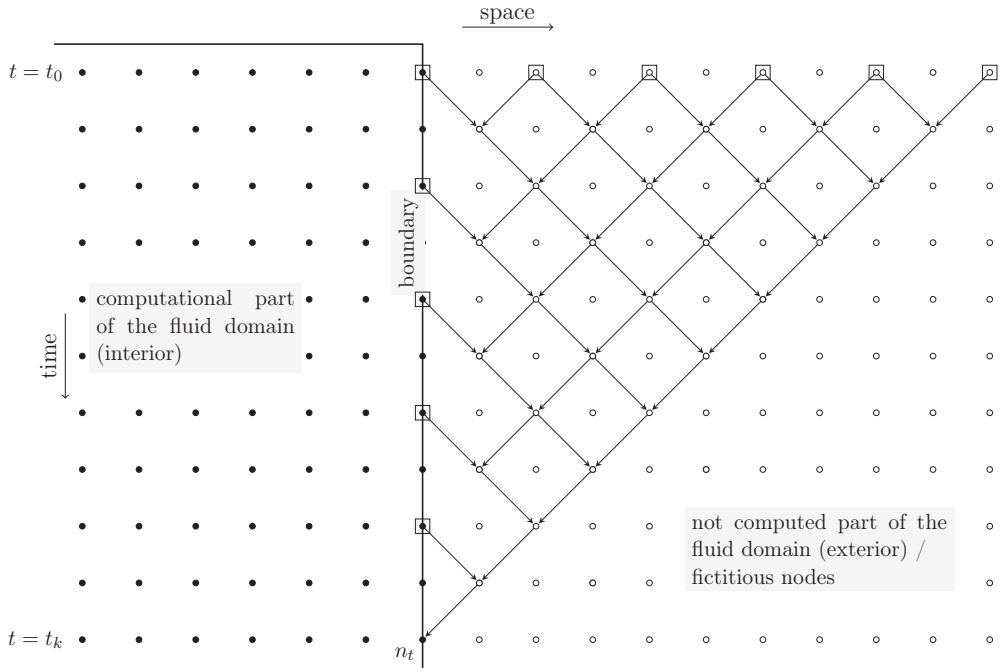


Figure 4.6: *Full weighted digraph for constructing an exact boundary condition* – The digraph shown is used to compute the inward population f_1 in the terminal node $n_t = (x_b, t_k)$. Nodes in the exterior of the computational domain (unfilled circles) are only fictitious. We distinguish two type of contributing nodes (squares), past boundary nodes and fictitious nodes from initial time level.

compensate the lack of missing adjacent lattice points theoretically. This idea is visualized for a right boundary in Fig. 4.6. The unfilled circles shown in the space-time diagram are fictitious nodes. The *contributing fictitious nodes* of the digraph are located at initial time $t = t_0$. Therefore it would be sufficient to assume that populations in the fictitious lattice points are known only at initial time. In fact we assume that the populations are homogeneously initialized in an equilibrium state E_i, that is for all fictitious locations x_E:

$$f_i(x_E, t_0) = E_i, \tag{4.12}$$

which agrees with the assumption of ITBCs, see Section 3.1. Under the assumption (4.12) the digraph can be shortened without loss of information, since all populations at nodes within the triangle in Fig. 4.7 retain the initial equilibrium state. The reduced digraph is shown in Fig. 4.7. For the use below, we abbreviate

$$\kappa = \left\lfloor \tfrac{k}{2} \right\rfloor, \tag{4.13}$$

where $\lfloor \cdot \rfloor$ denotes the *floor function*. All populations at contributing fictitious nodes, see (4.15) below, are known by the equilibrium used in (4.12). In fact, there are contributing nodes, n_1^k to n_κ^k, located at the boundary:

$$n_m^k = (x_b, t_k - 2m\Delta t), \qquad m = 1, \ldots, \kappa. \tag{4.14}$$

The superscript is used to emphasize that the nodes are used to compute the unknown population at time level $t = t_k$. All other remaining contributing nodes ($n_{\kappa+1}^k$ to $n_{\kappa+\kappa}^k$)

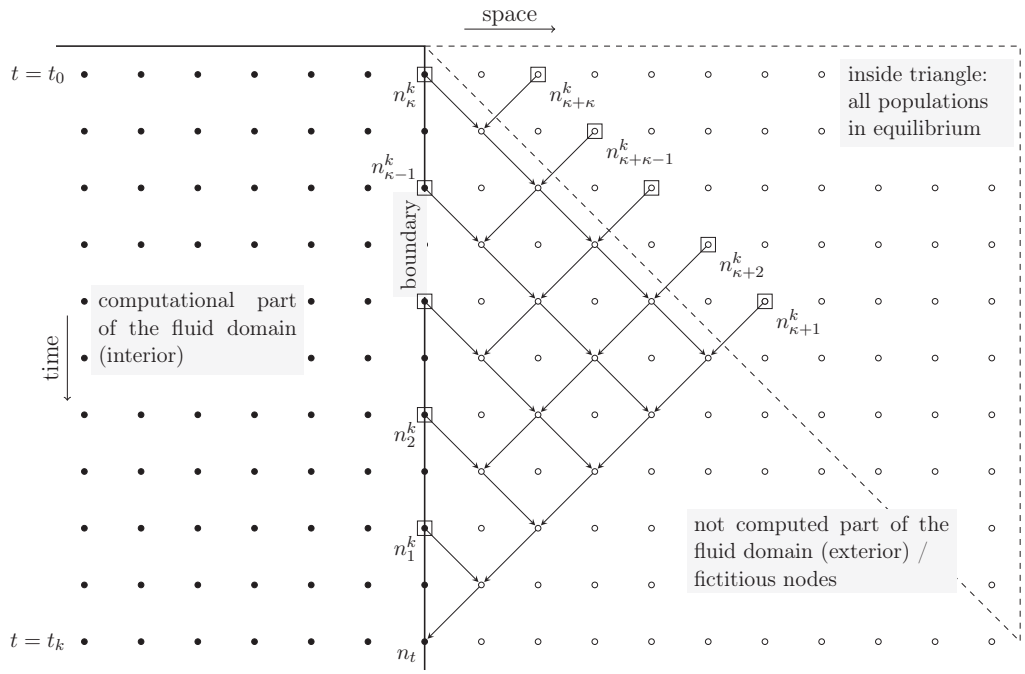


Figure 4.7: *Adapted weighted digraph for constructing an exact boundary condition* – The digraph is used to compute the inward population in the terminal node n_t. The nodes in the exterior of the computational domain (unfilled circles) are fictitious. When these nodes are assumed to be initialized in a homogeneous equilibrium, then all nodes within the dashed triangle are also in this equilibrium state and the digraph can be shorten.

are fictitious and are located in the dashed triangle of Fig. 4.7:

$$n^k_{\kappa+m}(k) = \Big(x_b \pm \big(\kappa + 2 - m\big)c\Delta t,\, t_0 + \Delta t\big(b(k) + \kappa - m\big)\Big), \quad m = 1, \ldots, \kappa. \tag{4.15}$$

The plus sign in the spatial component is used if x_b is at the right boundary of $\mathcal{G}$ and the minus sign for x_b lying at the left boundary. Moreover, $b(k) = \lceil \frac{k}{2} \rceil - \lfloor \frac{k}{2} \rfloor$ is a binary state function, which is zero/one for k being even/odd, respectively. The digraph in the space-time diagram of Fig. 4.7 suggests that we can compute the inward population $f_j(x_b, t_k)$, in terms of E_i and previous boundary information $\boldsymbol{f}(x_b, t_\ell)$ with $0 \leq \ell < k$. Thus, we can conclude the following structure for the exact DABC

$$f_j(x_b, t_k) = B^{\mathrm{ex}}_j(x_b, t_k) := \sum_{m=1}^{\kappa} \sum_{l=1}^{2} \Big(W_l(n^k_m) f_l(x_b, t_k - 2m\Delta t) + W_l(n^k_{\kappa+m}) E_l \Big). \tag{4.16}$$

It remains to explicitly state calculation rules for the node weights W_l. This is done in the following section.

4.4 Exact discrete artificial boundary condition

To determine the node weights in (4.16) we need to investigate each path connecting a contributing node with the terminal node $n^k_t = (x_b, t_k)$. Note that due to our setup all

node weights are non-zero, because there are no contributing nodes, where populations are only partially known.

The node weights are defined according to (4.10) where the required path weights are calculated by the rules given in Corollary 1. Obviously, the path weights are equal for all paths from a certain node n_j when having the same property $\bar{\boldsymbol{p}}$. Hence, given the number of paths $P = P(n_j, \bar{\boldsymbol{p}})$ from a node n_j with property $\bar{\boldsymbol{p}}$, we can compute the node weights equivalently to (4.10) by

$$W_l(n_j) = \sum_{\bar{\boldsymbol{p}} \in \mathcal{P}} P(n_j, \bar{\boldsymbol{p}}) \cdot \bar{w}_l(\bar{\boldsymbol{p}}), \tag{4.17}$$

where the sum is taken over all appearing properties $\mathcal{P}$.

We split our further explanation into two parts, which are separately investigated in the following two subsections. The first part investigates the contributing nodes (4.15), and the second part the nodes (4.14). For convenience, below we use superscripts F and B, respectively, to distinguish the two parts. Thereby, F denotes fictitious nodes, and B previous boundary nodes. For both parts we state the number of paths P and their corresponding path weights. Doing so, we determine the required node weights (4.17). In both cases we will explicitly use, that under consideration of the location of a node, the *total number of directional changes* is the essential variable to obtain the property $\bar{\boldsymbol{p}}$.

4.4.1 Node weights for contributing fictitious nodes

Here we compute the node weights (4.17) for contributing fictitious nodes

$$n_\ell^k = n_{\kappa+m}^k, \quad m = 1, \dots, \kappa,$$

given by (4.15), and obviously it holds $\ell = \kappa + m$. We consider an arbitrary path from a node n_ℓ^k to the terminal node n_t. By a graphical observation, see Fig. 4.7, we can directly conclude that the length of the path is equal to $\ell = \kappa + m$, thus the length depends on the time level k. Furthermore, we see for a right (left) boundary that $\kappa + 1$ and $m - 1$ give the number of leftward (rightward) and rightward (leftward) edges, respectively. The first and last edge are always of the same direction, which implies that the number of all directional changes has to be even and the number of left-right and right-left directional changes has to be equal, provided the first and its (fictitious) previous edge have same direction. Obviously, the number of directional changes has to be smaller than $2(m-1)$, otherwise the path could not end in n_t. As the following corollary shows, by consideration of the location of n_ℓ^k and the number of all directional changes, we achieve the required property $\bar{\boldsymbol{p}}$ to calculate the path weights.

Corollary 2. *Let be given a path from node n_ℓ^k to the terminal node n_t, which has $2v$ directional changes in total and we assume the first and its (fictitious) previous edge have same direction, i.e., no additional directional change in n_ℓ^k. Then, the path has property*

$$\bar{\boldsymbol{p}} = (v, v, m-1-v, \kappa+1-v) \quad \textit{or} \quad \bar{\boldsymbol{p}} = (v, v, \kappa+1-v, m-1-v).$$

for a left or right boundary, respectively.

Proof. We only show the validity for a right boundary, the left boundary follows analogously.

i) The number of leftward edges is $\bar{s}_l + \bar{c}_l = \kappa + 1$.

ii) The number of rightward edges is $m - 1$, i.e.: $\bar{s}_r + \bar{c}_r = m - 1$.

iii) There is an equal number of left-right and right-left directional changes: $\bar{c}_l - \bar{c}_r = 0$.

iv) The total amount of directional changes is given by $2v$, in detail: $\bar{c}_l + \bar{c}_r = 2v$.

The solution of the corresponding linear system leads to the stated values in $\bar{\boldsymbol{p}}$. □

As a consequence, all computations of weights for a given node are reduced to depend only on one variable, that is the total number of directional changes. We denote by $\bar{w}_i^F(k, m, v)$ the path weight of a path from a node $n_\ell^k = n_{\kappa+m}^k$ to the terminal node $n_t^k = (x_b, t_k)$, which has $2v$ directional changes in total. Again the amount of directional changes should not take into account a possible change in the first node.

Lemma 3. *The path weights $\bar{w}_i^F(k, m, v)$, are given for a right boundary by*

$$\bar{w}_1^F(k, m, v) = (1 + \alpha)^{\kappa+1-v} \cdot \beta^v \cdot \gamma^v \cdot (1 + \delta)^{m-v-1}, \tag{4.18a}$$

$$\bar{w}_2^F(k, m, v) = \frac{\beta}{1 + \alpha} \bar{w}_1^F(k, m, v). \tag{4.18b}$$

For a left boundary they read:

$$\bar{w}_2^F(k, m, v) = (1 + \alpha)^{m-v-1} \cdot \beta^v \cdot \gamma^v \cdot (1 + \delta)^{\kappa+1-v}, \tag{4.19a}$$

$$\bar{w}_1^F(k, m, v) = \frac{\gamma}{1 + \delta} \bar{w}_2^F(k, m, v). \tag{4.19b}$$

Proof. The path weights (4.18a) and (4.19a) follow directly from Corollary 2. Since Corollary 2 assumed there is no directional change in n_ℓ^k, the edge weight of the first edge in case of a right boundary is $(1+\alpha)$. This assumption is violated for a right boundary when computing $\bar{w}_2^F(k, m, v)$, and the first edge weight is β instead, which implies the property $\bar{\boldsymbol{p}} = (v + 1, v, \kappa - v, m - 1 - v)$ and thus (4.18b). Analogously, we can argue the validity of (4.19b). □

Lemma 3 gives the path weight for a certain path, however in order to state the node weights (4.17) we need their amount. To this end, let $P^F(k, m, v)$ give the number of paths from the node $n_\ell^k = n_{\kappa+m}^k$ to the terminal node $n_t^k = (x_b, t_k)$ having $2v$ directional changes in total, counted as above.

Lemma 4. *For $P^F(k, m, v)$, we have:*

a) The possible arguments of $P^F(k, m, v)$ are:

$$k \in \mathbb{N}^+, \quad m = 1, \ldots, \kappa, \quad v = 0, \ldots, m - 1.$$

b) If k is even, then holds for all possible choices of m and v:

$$P^F(k + 1, m, v) = P^F(k, m, v).$$

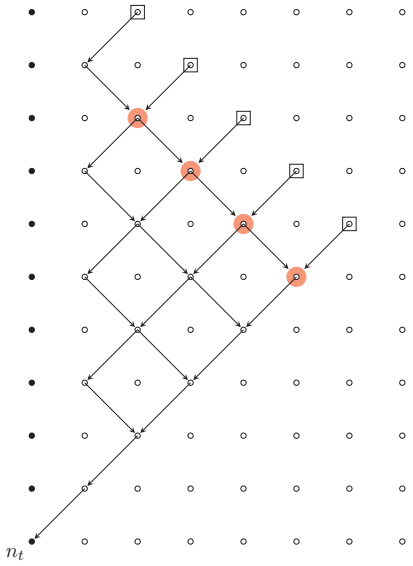


Figure 4.8: *Digraph restricted to contributing fictitious nodes* – The digraph here is a subgraph of the digraph from Fig. 4.7, limited to the contributing fictitious nodes. This view is sufficient for the proof of Lemma 4. The colored nodes show the nodes (4.22).

c) For all $k \in \mathbb{N}^+$: If $m = 1$ then there is only one path with no directional change:

$$P^F(k, m, 0) = 1.$$

If $m > 1$ then there are no paths having no directional change:

$$P^F(k, m, 0) = 0.$$

d) The remaining non-zero values for $k \geq 4$ can be computed as

$$P^F(k, m, v) = \left(1 - \frac{m-1}{\kappa}\right) \cdot \binom{\kappa}{v} \cdot \binom{m-2}{v-1}, \tag{4.20}$$

with $m = 2, \ldots, \kappa$ and $v = 1, \ldots, m-1$.

Proof. a) The restriction of m follows directly from (4.15), whereas the one for v has been argued at the beginning of the current subsection.

b) If k is even then at the next time level $t = t_{k+1}$ the amount of nodes (4.15) remains the same. All contributing nodes are displaced by one level positive in time, i.e., $b(k)$ in (4.15) switches from 0 to 1. The structures of the digraphs for k and $k+1$ are the same.

c)+d) The unique path for $m = 1$ (from the node $n^k_{\kappa+1}$) has no directional change (i.e., $v = 0$). The paths from other nodes must have at least two directional changes. Therefore, it immediately follows $P^F(k, 1, 0) = 1$ and $P^F(k, m, 0) = 0$ for $m > 1$. Moreover it follows that in (4.20) v has to be positive, i.e., $v = 1, \dots, m-1$. The validity of (4.20) for $k = 4$ is easily clarified by a graphical consideration. For $k \geq 4$ we argue the validity of the following recursion and eventually verify that (4.20) satisfies the recursion:

$$P^F(k, m, v) = P^F(k-2, m, v) + \sum_{p=1}^{m-1} P^F(k-2, p, v-1), \quad m = 1, \dots, \kappa-1, \tag{4.21a}$$

$$P^F(k, \kappa, v) = \sum_{p=1}^{m-1} P^F(k-2, p, v-1). \tag{4.21b}$$

The following argumentation is done for a right (left) boundary. Let $k \geq 4$, we consider the digraph of Fig. 4.8, which shows only the relevant edges of the general digraph in Fig. 4.7. The colored nodes shown in the plot visualize the nodes

$$\tilde{n}_p = \big(x_b \pm \big(\kappa + 1 - p\big)c\Delta t,\, t_0 + \Delta t\big(b(k) + \kappa + 1 - p\big)\big), \tag{4.22}$$

with $p = 1, \dots, \kappa - 1$. Their relative location to n^k_t is the same as the nodes $n^{k-2}_\cdot$ of (4.15) have to the terminal node n^{k-2}_t of two time levels ago. Hence, the number of paths starting in a colored node $\tilde{n}_p$, for which the first edge is leftward (rightward), is given by an appropriate evaluation of $P^F(k-2, p, \cdot)$. We investigate $P^F(k, m, v)$ and consider three cases, depending on the choice of m:

Case 1) $m = 2, \dots, \kappa - 1$: $P^F(k, m, v)$ is the number of paths starting in a contributing node. The first edge of each path from a contributing node is leftward (rightward). Obviously, the head of the first edge is a colored node, which is arrived without a change of direction. The number of all paths, whose second edge continues leftward (rightward) is therefore $P^F(k-2, m, v)$. That is the first term of (4.21a). If the second edge however is rightward (leftward) a directional change occurs. Without an additional change of direction only other colored nodes can be reached. The second directional change is therefore done in one of the colored nodes. After this change is done the number of paths is given by $P^F(k-2, p, v-1)$, for one $p = 1, \dots, m-1$ according to where the second directional change occurs. Since we want to count all paths, we have to sum those $P^F(k-2, p, v-1)$ over all possible p. Thus the second term of (4.21a) is explained. In total this argues the validity of (4.21a) for $m = 2, \dots, \kappa - 1$.

Case 2) $m = 1$: The number of paths $P^F(k, 1, v)$ is equal to $P^F(k-2, 1, v)$, since the second edge has to continue leftward (rightward). The sum in (4.21a) is zero, hence the recursion is also valid for $m = 1$.

Case 3) $m = \kappa$: Here all paths have a directional change after its first edge. Therefore the first term in (4.21a) has to vanish and we obtain (4.21b).

The proof is finished by showing that (4.20) satisfies the recursion (4.21). This is done by straightforward analysis, which is presented in Appendix C. □

The node weights for contributing fictitious nodes (4.15) can now be computed with (4.17)

as follows:

$$W_l(n^k_{\kappa+m}) = \sum_{v=0}^{m-1} P^F(k,m,v) \cdot \bar{w}^F_l(k,m,v), \tag{4.23}$$

where Lemmas 3 and 4 define the right hand side terms.

4.4.2 Node weights for past boundary nodes

Here, we compute the node weights for the contributing nodes of past boundary nodes (4.14). If we consider the node n^k_m, then all paths from here to n^k_t have length $2m$. The directions of the first and last edge of each path are fixed and are opposite to each other, implying an odd number, say $(2v-1)$, of total directional changes. Moreover at least one directional change has to occur. If we omit the first and last edge (with fixed directions) each, then each reduced path defines a *Dyck-path* of *semilength* $m-1$ [20]. Transferred to the terminology of Dyck-paths, the total amount of $(2v-1)$ directional changes corresponds to a number of v *peaks* and $v-1$ *valleys*, for more details see [20].

The number of Dyck-paths of semilength $m-1$ with v peaks are [20]

$$D(m-1,v) = \frac{1}{m-v}\binom{m-1}{v}\binom{m-2}{v-1}, \quad v = 1,\dots,m-1. \tag{4.24}$$

The amount of contributing nodes in (4.14) is influenced by the time level k, however the relative location of each node n^k_m to the corresponding terminal node n^k_t is independent of k. By this, neither the number of paths nor the corresponding path weights depend on k. Analogue to the previous subsection, let $P^B(m,v)$ denote the total number of paths starting in the contributing node $n^k_m = (x_b, t_k - 2m)$, ending in n^k_t and having $(2v-1)$ changes of directions.

Lemma 5. *For the number of paths $P^B(m,v)$ it holds:*

i) If $m = 1$ there is only one path, and this path has exactly one directional change. Thus:

$$P^B(m,v) = P^B(1,1) = 1. \tag{4.25}$$

ii) For $m \geq 2$ the non-zero values are computed by

$$P^B(m,v) = \frac{1}{m-v}\binom{m-1}{v}\binom{m-2}{v-1}, \tag{4.26}$$

where $v = 1,\dots,m-1$.

Proof. i) For $m = 1$ the length of a path from node n^k_m to n^k_t is equal to 2, where the first and last edge have different directions. Hence, $P^B(1,1) = 1$ is obvious. Since a path cannot have more directional changes than its length, no other values for v are possible.
ii) The result (4.26) follows from the Dyck-path enumeration (4.24). □

By the previous lemma we know the number of paths, but in order to eventually state the node weights (4.17) we need to have the corresponding path weights $\bar{w}_i^B$. Let $\bar{w}_i^B(m,v)$ denote the path weight according to a path enumerated by $P^B(m,v)$. We have the following result:

Lemma 6. *The path weights $\bar{w}_i^B(m,v)$ for a right boundary are given by*

$$\bar{w}_2^B(m,v) = (1+\alpha)^{m-v} \cdot \beta^v \cdot \gamma^{v-1} \cdot (1+\delta)^{m-v+1}, \tag{4.27a}$$

$$\bar{w}_1^B(m,v) = \frac{\gamma}{1+\delta}\bar{w}_2^B(m,v). \tag{4.27b}$$

For a left boundary they read:

$$\bar{w}_1^B(m,v) = (1+\alpha)^{m-v+1} \cdot \beta^{v-1} \cdot \gamma^v \cdot (1+\delta)^{m-v}, \tag{4.28a}$$

$$\bar{w}_2^B(m,v) = \frac{\beta}{1+\alpha}\bar{w}_1^B(m,v). \tag{4.28b}$$

Proof. We only show the derivation of weights for a right boundary, the left boundary follows in an analogue manner. We use the general form for the path weight $\bar{w}_2^B$ implied by Corollary 1:

$$\bar{w}_2^B(m,v) = (1+\alpha)^{\bar{s}_l} \cdot \beta^{\bar{c}_l} \cdot \gamma^{\bar{c}_r} \cdot (1+\delta)^{\bar{s}_r}.$$

There are four conditions to the exponents:

i) The number of leftward edges is determined exclusively by m: $\bar{s}_l + \bar{c}_l = m$.

ii) The first condition holds also for rightward edges: $\bar{s}_r + \bar{c}_r = m$.

iii) There is one right/left directional change more than left/right changes: $\bar{c}_l - \bar{c}_r = 1$.

iv) The total amount of directional changes depends on v, in detail: $\bar{c}_l + \bar{c}_r = 2v - 1$.

The unique solution of the corresponding linear system for the exponents leads to (4.27a). In the computation for $\bar{w}_1^B(m,v)$ the edge weight of the first edge is γ instead of $(1+\delta)$. This implies the relation (4.27b). □

Lemmas 5 and 6 are used to state the node weights for contributing nodes (4.14):

$$W_l(n_1^k) = \bar{w}_l^B(1,1), \tag{4.29a}$$

$$W_l(n_m^k) = \sum_{v=1}^{m-1} P^B(m,v) \cdot \bar{w}_l^B(m,v), \quad m = 2,\dots,\kappa. \tag{4.29b}$$

4.4.3 Exact discrete artificial boundary condition

We take the previously computed node weights (4.23) and (4.29), to express the inward population at a boundary lattice point $x_b \in \Gamma \subseteq \mathcal{B}$ by (4.16). So, under the assumption that the exterior domain was initialized in equilibrium, see (4.12), it follows an exact

DABC:

$$B_j^{\mathrm{ex}}(x_b, t_1) = E_j, \tag{4.30a}$$

$$B_j^{\mathrm{ex}}(x_b, t_k) = \sum_{l=1}^{2} \left(\sum_{m=1}^{\kappa} \alpha_l(m) f_l(x_b, t_k - 2m\Delta t) + \beta_l(k) E_l \right), \qquad k \geq 2, \tag{4.30b}$$

where the node weights α_l for past boundary nodes are directly available by (4.29)

$$\alpha_l(1) = \bar{w}_l^B(1,1), \tag{4.31a}$$

$$\alpha_l(m) = \sum_{v=1}^{m-1} P^B(m,v) \cdot \bar{w}_l^B(m,v), \qquad m = 2, \ldots, \kappa. \tag{4.31b}$$

Due to (4.12) all populations in contributing fictitious nodes are equal, thus we can write the BC with one node weight for the equilibrium E_l, see (4.12):

$$\beta_l(k) = \sum_{m=1}^{\kappa} \sum_{v=0}^{m-1} P^F(k,m,v) \cdot \bar{w}_l^F(k,m,v). \tag{4.32}$$

When evolving in time the node weights α_l remain unchanged, however additional node weights have to be computed. Contrary, the node weights β_l depend on the time level, however they change only after two time steps, since for even k holds:

$$\beta_l(k+1) = \beta_l(k).$$

This follows directly as k only appears in κ, see (4.13). Next we show that the β_l can be determined alternatively, only in dependence of the node weights α_l.

Alternative computation of the node weights First we assume that also all interior populations are homogeneously initialized with the same equilibrium values E_i, i.e., I_i in (4.1) shall satisfy

$$I_i(x) = E_i, \quad \text{for all } x \in \mathcal{G}. \tag{4.33}$$

Then, as an exact BC all inward populations computed by (4.30) are equally given by E_j. Therefore, equation (4.30) exhibits the form

$$B_j^{\mathrm{ex}}(x_b, t_k) = \left[S_{\alpha_1}^{\kappa} + \frac{1+a}{1-a} S_{\alpha_2}^{\kappa} + \beta_1(k) + \frac{1+a}{1-a} \beta_2(k) \right] E_1 \tag{4.34a}$$

$$= \left[\frac{1-a}{1+a} S_{\alpha_1}^{\kappa} + S_{\alpha_2}^{\kappa} + \frac{1-a}{1+a} \beta_1(k) + \beta_2(k) \right] E_2. \tag{4.34b}$$

In the derivation of (4.34), we used the general relation

$$E_2 = \frac{1+a}{1-a} E_1,$$

which holds for any equilibrium populations in the present D1Q2 model. Moreover, we used the short-hand notation

$$S^q_{\alpha_j} := \sum_{m=1}^{q} \alpha_j(m). \tag{4.35}$$

We conclude that the corresponding coefficient of E_j (square bracket term) in (4.34) has to be equal to one. If α_l and β_l are computed by (4.31) and (4.32), respectively, this requirement is naturally satisfied. Hence, the node weights β_l, see (4.32), can also be determined uniquely by solving the linear system

$$(1-a)\beta_1(k) + (1+a)\beta_2(k) = (1 \pm a) - (1-a)S^\kappa_{\alpha_1} - (1+a)S^\kappa_{\alpha_2}, \tag{4.36}$$

in combination with one of the following relations implied by (4.18b) or (4.19b), respectively:

$$\beta_2(k) = \frac{\beta}{1+\alpha}\beta_1(k) \qquad \text{or} \qquad \beta_2(k) = \frac{1+\delta}{\gamma}\beta_1(k). \tag{4.37}$$

The first equation of (4.37) has to be combined with the minus sign in (4.36) for a right boundary. Having a left boundary the plus sign in (4.36) is taken in combination with the second equation of (4.37). We remark that a relation analogue to (4.37) holds also for the node weights α_1 and α_2, and thus also for their sums (4.35).

Chapter 5
5 Discrete artificial boundary conditions

The previous chapter provided an exact DABC for a one-dimensional LB simulation with a linear collision model. Its computational effort is too high to serve as a practical BC itself. The main goal of the present chapter is to formulate a practical BC, which is derived from the exact DABC. Thereby it is also formulated purely on the discrete level. We emphasize that we do not confine ourselves to one-dimensional problems, but consider also higher spatial dimensions.

To accomplish our goal, we pursue the following steps. First we devise an approximation to the exact DABC for the one-dimensional linear collision model, used in the previous chapter (Section 5.1). Hereby we introduce a so-called *history depth*, which serves as a key parameter for the DABCs. In a second step (Section 5.2), we interpret this approximate DABC as the solution of a separate problem, a so-called *subproblem*. These are individual LB simulations, where the history depth is its most important parameter. Based on this view, in Section 5.3, the idea of this approximate DABCs is transferred to a one-dimensional nonlinear collision model. In Section 5.4 it is generalized to higher dimensions, any collision models, and arbitrary discretization models, especially including the nonlinear models used to recover the Navier-Stokes equations. Lastly, in Section 5.5, we investigate DABCs in terms of error sources, relations between their history depth and their subproblem's initialization, as well as on an efficient implementation and its computational cost.

This chapter combines results of our works. Section 5.1 is based on [47], while Sections 5.2, 5.3 and 5.5 rests upon [48]. The content of [49] is found in Section 5.4.

5.1 One-dimensional discrete artificial boundary conditions for a linear collision model

The basis for deriving a practical (approximate) DABC B_j for a setup in the form of (4.1), i.e.,

$$P_{\text{lin}} = \texttt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}_{\text{D1Q2}}, \text{C}_{\text{lin}}, f_j, I_j, B_j), \tag{5.1}$$

is the exact DABC (4.30). We recall that its computational effort consists in the determination of additional node weights α_i necessary when time is evolving. Also these costs do increase with advancing time, since the sum in (4.31b) contains more and more summands. To obtain the approximate DABC we aim to reduce the computational effort at the expense of accuracy. This is achieved by truncating the sum in (4.31b). The idea of truncation is motivated by the result of the following subsection.

5.1.1 Decreasing node weights

Let us consider the exact DABC (4.30) for the special case $\omega = 1$ (see (4.2)) of the setup (5.1), then the weights (4.8) satisfy

$$1 + \alpha = \beta, \qquad 1 + \delta = \gamma.$$

It follows that the path weights defined in Lemma 6 are independent of the boundary location

$$\bar{w}_j^B(m, v) = \left[\frac{1}{4}(1 - a^2)\right]^m \leq 4^{-m}, \quad j = 1, 2,$$

where the upper bound holds due to $a \in (-1, 1)$. If one uses this estimate, the node weights (4.31) have an upper bound, which tends to zero for m going to infinity:

$$0 \leq \alpha_i(m) \leq 4^{-m} \sum_{v=1}^{m-1} \frac{1}{m - v} \binom{m-1}{v} \binom{m-2}{v-1} \leq \frac{1}{4m\sqrt{\pi(m-1)}}.$$

This estimate holds, since the sum represents the *Catalan number* C_{m-1} with known bounds [24, 72]

$$C_{m-1} = \frac{1}{m} \binom{2m-2}{m-1} \leq \frac{4^{m-1}}{m\sqrt{\pi(m-1)}}.$$

We conclude the node weights $\alpha_i(m)$ are decreasing for increasing m. This is a motivation to disregard the populations corresponding to large m in the exact DABC (4.30) to obtain an efficient approximation.

5.1.2 Approximate discrete artificial boundary condition

We will show in Section 5.1.3 that the decay of $\alpha_i(m)$ is not limited to the case $\omega = 1$. So, in the exact DABC (4.30), we neglect populations corresponding to large m to obtain a suitable approximation. To this end, in (4.30b) the inner sum is truncated at some $\tilde{h}(k) \in \mathbb{N}$:

$$B_j(x_b, t_k) = \sum_{l=1}^{2} \left(\sum_{m=1}^{\tilde{h}(k)} \alpha_l(m) f_l(x_b, t_k - 2m\Delta t) + \tilde{\beta}_l(k) E_l \right), \quad k \geq 2. \tag{5.2}$$

The function $\tilde{h}(k)$ controlling the truncation should be bounded above $\tilde{h}(k) \leq \tilde{H}$ for all $k \in \mathbb{N}$, where $\tilde{H} \in \mathbb{N}$ is fixed. This avoids the high computational costs of the exact DABC, i.e., the approximation (5.2) becomes efficient. Also the first sum in (4.32) has an increasing computational effort, which is compensated by substituting the β_l by new node weights $\tilde{\beta}_l$, which are solutions to

$$(1 - a)\tilde{\beta}_1(k) + (1 + a)\tilde{\beta}_2(k) = (1 \pm a) - (1 - a)S_{\alpha_1}^{\tilde{h}(k)} - (1 + a)S_{\alpha_2}^{\tilde{h}(k)},$$

and

$$\tilde{\beta}_2(k) = \frac{\beta}{1+\alpha}\tilde{\beta}_1(k) \qquad \text{or} \qquad \tilde{\beta}_2(k) = \frac{1+\delta}{\gamma}\tilde{\beta}_1(k).$$

Such a system was derived for the original node weights above, see (4.36) and (4.37). We can conclude that the new node weights are equivalent to

$$\tilde{\beta}_l(k) = \sum_{m=1}^{\tilde{h}(k)} \sum_{v=0}^{m-1} P^F(k,m,v) \cdot \bar{w}_l^F(k,m,v).$$

This choice of $\tilde{\beta}_l$ ensures that a global equilibrium is retained by the approximate DABC (5.2).

History depth The boundary population computed by (5.2) depends on populations from the past time levels

$$t = t_k - m\Delta t \quad \text{with} \quad m = 1, \ldots, 2\tilde{h}(k).$$

The truncated formulation with $\tilde{h}(k) \leq \tilde{H}$ exhibits a finite memory, which gives rise to call $h(k) = 2\tilde{h}(k)$ the *history depth* of time level $t = t_k$. Analogously, $H = 2\tilde{H}$ is denoted as the *maximal history depth*. The history depth of the exact DABC is $h(k) = 2\kappa$, and thus not bounded above. We suggest to choose the truncation parameter such that the corresponding history depth satisfies

$$h(k) = \min\{2\kappa,\ H\}. \tag{5.3}$$

Thus, the approximate DABC (5.2) with (5.3) is exact at the beginning of the simulation as long as $2\kappa \leq H$ holds.

5.1.3 Proof of node weights' general decrease

We end this section by showing the convergence of node weights $\alpha_i = \alpha_i(m)$ (for $m \to \infty$) for any $(\omega, a) \in F := (0,\, 2) \times (-1,\, 1)$, where F is the full parameter domain. This makes the truncation reasonable, furthermore it yields an analytical basis for the history depth being an accuracy parameter. We consider the auxiliary node weights

$$\tilde{\alpha}(m) = \sum_{v=1}^{m-1} \frac{1}{m-v} \binom{m-1}{v} \binom{m-2}{v-1} (1+\alpha)^{m-v}\ \beta^v\ \gamma^v\ (1+\delta)^{m-v}. \tag{5.4}$$

The actual weights (4.31) are obtained by $\alpha_i(m) = C \cdot \tilde{\alpha}(m)$ with an appropriate constant C, which can be omitted in the convergence investigation:

$$\lim_{m\to\infty} |\alpha_i(m)| = 0 \quad \Leftrightarrow \quad \lim_{m\to\infty} |\tilde{\alpha}(m)| = 0. \tag{5.5}$$

Before we prove the convergence (5.5) (see Lemma 9 below) we gather some properties of the edge weights:

Lemma 7. *It holds:*

a) The weights β and γ are positive for all $(\omega, a) \in F$.

b) The weights $(1+\alpha)$ and $(1+\delta)$ are non-negative for $(\omega, a) \in D_1 \subset F$, where

$$D_1 := \Big\{(\omega, a) \in F \mid \omega(1+a) \leq 2 \ \wedge \ \omega(1-a) \leq 2\Big\}. \tag{5.6}$$

For $(\omega, a) \in F \setminus D_1$ either $(1+\alpha)$ or $(1+\delta)$ is negative, but they are not negative simultaneously.

The domain (5.6) is visualized in Fig. 5.1.

Proof. a) This follows directly from the definitions of β and γ, see (4.8).

b) An equivalent condition for $(1+\alpha)$ being negative is given as follows:

$$(1+\alpha) < 0 \Leftrightarrow 1 - \frac{1}{2}\omega(1+a) < 0 \Leftrightarrow \omega(1+a) > 2$$

Analogously, we obtain $(1+\delta) < 0 \Leftrightarrow \omega(1-a) > 2$. Both conditions taken together imply the non-negativity on D_1, since for $(\omega,\, a) \in D_1$ neither $\omega(1+a) > 2$ nor $\omega(1-a) > 2$ is satisfied.

It remains to show $\omega(1+a) > 2 \Rightarrow \omega(1-a) \leq 2$, such that the negativity of $(1+\alpha)$ implies the non-negativity of $(1+\delta)$: It obviously holds $(1+a) \in (0,\, 2)$ and $\omega \in (0,\, 2)$. Therefore the validity of $\omega(1+a) > 2$ implies $\omega > 1$ and $a \in (0,\, 1)$. With $(\omega, a) \in (1,2) \times (0,1)$ it directly follows $0 \leq \omega(1-a) \leq 2$. Analogously, we can show the implication $\omega(1-a) > 2 \Rightarrow 0 \leq \omega(1+a) \leq 2$. □

One part of the proof of the convergence (5.5) is based on an estimate presented in the following lemma.

Lemma 8. *Let us consider*

$$S_{p_1,p_2,q_1,q_2}(m) := \sum_{v=1}^{m-1} \frac{1}{m-v} \binom{m-1}{v} \binom{m-2}{v-1} p_1^{m-v} p_2^{m-v} q_1^v q_2^v, \tag{5.7}$$

for $m \geq 2$ and $p_i,\, q_i \in \mathbb{R}$. If both p_i and both q_i are non-negative, then there exists an upper bound:

$$S_{p_1,p_2,q_1,q_2}(m) \leq \frac{[(p_1+q_1)(p_2+q_2)]^m}{m}.$$

Proof. The statement is shown by using the rather rough estimate

$$\sum_{k=1}^{n} a_k b_k \leq \left(\sum_{k=1}^{n} a_k\right)\left(\sum_{k=1}^{n} b_k\right),$$

which holds if all a_k and b_k are non-negative. The function $S_{p_1,p_2,q_1,q_2}(m)$ is estimated as

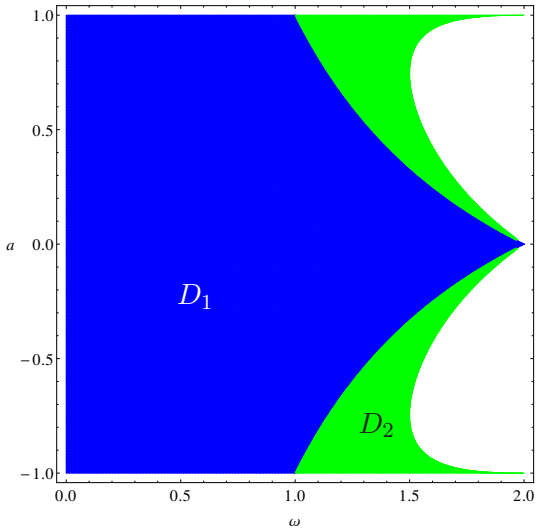


Figure 5.1: *Visualization of the subdomains D_1 and D_2* – The rectangle describes the full parameter domain F. The blue subdomain visualizes the domain D_1 (5.6), on which the weights (4.8) satisfy certain properties (see Lemma 7) and on which (5.8) is proven in Lemma 9 with application of (5.7). Furthermore, the green subdomain D_2 gives the extension of D_1 for which Lemma 8 could be used at most to prove (5.8).

follows:

$$\begin{aligned}
S_{p_1,p_2,q_1,q_2}(m) &= \frac{1}{m}\sum_{v=1}^{m-1}\binom{m}{v}\binom{m-2}{v-1}p_1^{m-v}p_2^{m-v}q_1^v q_2^v \\
&\leq \frac{1}{m}\left[\left(\sum_{v=1}^{m-1}\binom{m}{v}p_1^{m-v}q_1^v\right)\cdot\left(\sum_{v=1}^{m-1}\binom{m-2}{v-1}p_2^{m-v}q_2^v\right)\right] \\
&\leq \frac{p_2q_2}{m}\left[\left(\sum_{v=0}^{m}\binom{m}{v}p^v q^{m-v}\right)\cdot\left(\sum_{v=0}^{m-2}\binom{m-2}{v}p_2^{m-2-v}q_2^v\right)\right] \\
&= \frac{p_2q_2}{m(p_2+q_2)^2}\left[(p_1+q_1)(p_2+q_2)\right]^m \\
&\leq \frac{\left[(p_1+q_1)(p_2+q_2)\right]^m}{m}.
\end{aligned}$$

□

All required preparations to show the convergence (5.5) are now settled and the following lemma corroborates the statement analytically.

Lemma 9. *For all $(\omega,a) \in F := (0,2)\times(-1,1)$ the node weights (5.4) satisfy the convergence (5.5), i.e., it holds*

$$\lim_{m\to\infty}\left|\sum_{v=1}^{m-1}\frac{1}{m-v}\binom{m-1}{v}\binom{m-2}{v-1}(1+\alpha)^{m-v}\ \beta^v\ \gamma^v\ (1+\delta)^{m-v}\right| = 0. \tag{5.8}$$

Proof. i) In the first part, we show (5.8) for $(\omega, a) \in D_1$, where the domain is defined in (5.6): For $(\omega, a) \in D_1$ Lemma 8 can be applied directly with

$$p_1 = 1 + \alpha = 1 - \gamma, \qquad p_2 = 1 + \delta = 1 - \beta,$$
$$q_1 = \gamma, \qquad q_2 = \beta.$$

Then the convergence (5.8) follows immediately.

ii) In the second part we prove the convergence for all $(\omega, a) \in F \setminus D_1$: To this end, we rewrite (5.4) as

$$\tilde{\alpha}(m) = [(1+\alpha)(1+\delta)]^m \, N_{m-1}(q), \tag{5.9a}$$
$$N_{m-1}(q) = \sum_{v=1}^{m-1} \frac{1}{m-1} \binom{m-1}{v} \binom{m-1}{v-1} q^v, \tag{5.9b}$$

where $N_{m-1}(q)$ is a *Narayana polynomial* with argument $q = \frac{\beta\gamma}{(1+\alpha)(1+\delta)}$, see [72]. Note that the identity

$$\frac{1}{m-v} \binom{m-2}{v-1} = \frac{1}{m-1} \binom{m-1}{v-1}$$

was used to obtain this result. For the present case the argument q is negative due to Lemma 7. Now we use an equivalent expression of a Narayana polynomial [72]:

$$N_{m-1}(q) = (q-1)^m \int_0^{q/(q-1)} P_{m-1}(2x-1) \, \mathrm{d}x = \frac{(q-1)^m}{2} \int_{-1}^{b} P_{m-1}(z) \, \mathrm{d}z, \tag{5.10}$$

with P_{m-1} being the $(m-1)$-th *Legendre polynomial* and the upper integration bound $b = \frac{q+1}{q-1}$ satisfying $-1 < b < 1$. We use the following estimate for Legendre polynomials [105]

$$|P_n(x)| < \frac{\sqrt{2}}{\sqrt{n\pi}(1-x^2)^{1/4}}, \quad x \in (-1,1), \; n \in \mathbb{N},$$

to estimate the Narayana polynomial (5.10):

$$\begin{aligned}
|N_{m-1}(q)| &\le \frac{|q-1|^m}{2} \int_{-1}^{b} |P_{m-1}(z)| \, \mathrm{d}z \le \frac{|q-1|^m}{2} \int_{-1}^{1} |P_{m-1}(z)| \, \mathrm{d}z \\
&\le \frac{|q-1|^m}{\sqrt{2(m-1)\pi}} \int_{-1}^{1} \frac{1}{(1-z^2)^{1/4}} \, \mathrm{d}z \\
&= \frac{|q-1|^m}{\sqrt{2(m-1)\pi}} \int_{-\pi/2}^{\pi/2} \sqrt{\cos(t)} \, \mathrm{d}t \\
&\le \frac{|q-1|^m \sqrt{\pi}}{\sqrt{2(m-1)}}.
\end{aligned}$$

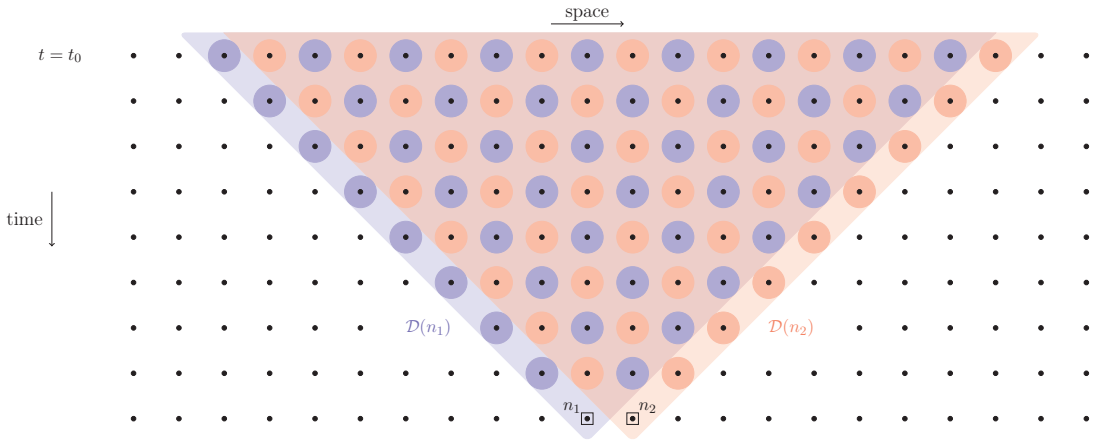


Figure 5.2: *Domain of dependence for two adjacent nodes in D1Q2* – The space-time diagram shows the domains of dependence $\mathcal{D}(n_1)$ and $\mathcal{D}(n_2)$ for two adjacent nodes n_1 and n_2, respectively. Within each domain of dependence the nodes which influence the information at node n_1 and n_2 are marked, respectively.

In total we obtain for (5.9)

$$|\tilde{\alpha}(m)| \leq |(1+\alpha)(1+\delta)(q-1)|^m \frac{\sqrt{\pi}}{\sqrt{2(m-1)}},$$

and the convergence (5.8) follows due to

$$|(1+\alpha)(1+\delta)(q-1)| = |\omega - 1| < 1.$$

This completes the proof. □

We end this section with a remark: Based on the result of Lemma 8 the convergence can be shown only for some $(\omega, a) \in F \setminus D_1$. Since negative edge weights occur in this case we need to consider absolute values when intending to apply Lemma 8. By choosing optimal values for p_i and q_i the convergence can additionally be shown at most for the green domain $D_2 \subset F \setminus D_1$ visualized in Fig. 5.1. For details we refer to Appendix D. Thus, the alternative approach used in part (ii) of the proof was necessary. Vice versa, the approach followed in part (ii) is not applicable for $(\omega, a) \in D_1$, either.

5.2 Interpretation as subproblems

In preparation for generalizing the approximate DABC (5.2) we explain a certain interpretation of the BC in the current section. However, let us first remark, that in D1Q2 only every second node in space and time has a contribution to the terminal node, as one can see in Fig. 5.2 and also in previous figures (Figs. 4.3 to 4.8). In Fig. 5.2 we show the domains of dependence $\mathcal{D}(n_1)$ and $\mathcal{D}(n_2)$ for two adjacent nodes n_1 and n_2. Clearly the D1Q2 model computes the evolution of two separate groups, where information is never exchanged between them. That is the reason why the approximate DABC (5.2) depends

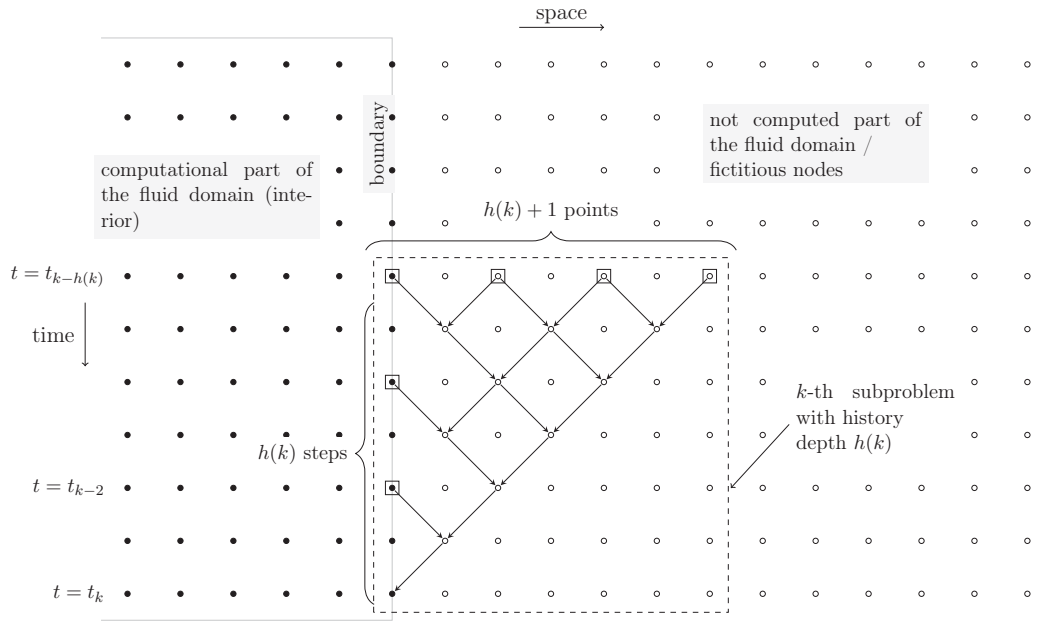


Figure 5.3: *Digraph corresponding to an approximate discrete boundary condition* – For a given history depth $h(k)$ the digraph visualizing the exact discrete boundary condition (Fig. 4.6) is cut at time level $t = t_{k-h(k)}$ and new contributing fictitious nodes are introduced. The dashed box containing the digraph symbolizes the k-th subproblem (5.14).

not on all past boundary nodes

$$n_m = (x_b, t_k - m\Delta t), \quad m = 1, \ldots, h(k), \tag{5.11}$$

but only on those where m is even. However, an equivalent formulation formally taking all nodes (5.11) into account is easily written down:

$$B_j(x_b, t_k) = \sum_{l=1}^{2} \left(\sum_{m=1}^{h(k)} \alpha'_l(m) f_l(x_b, t_k - m\Delta t) + \tilde{\beta}_l(k) E_l \right), \quad k \geq 2, \tag{5.12}$$

with node weights $\alpha'_l(m) = \alpha_l(m/2)$ for even m and $\alpha'_l(m) = 0$ otherwise.

A graphical interpretation for the approximate DABC is given by the digraph of Fig. 5.3, if we set the populations in the nodes

$$n_m = (x_b - c_j m\Delta t, t_k - h(k)\Delta t), \quad m = 1, \ldots, h(k) \tag{5.13}$$

(for appropriate m) to E_i. If we consider this digraph, we can split the contributing nodes into two types: Past boundary nodes (5.11) and fictitious contributing nodes (5.13). For both types of nodes the history depth $h(k)$ is directly related to the number of contributing nodes each. In our interpretation the processes within the dashed rectangle are described by some separate LB setup S^k. This means that the boundary population $B_j(x_b, t_k)$ of (5.12) is equivalently given as the solution of a separate LB simulation, which is called the *k-th subproblem*. It should be emphasized that for each time level a new subproblem is considered.

Next, we describe the setup of the *k-th subproblem*

$$S^k = \mathtt{LB}(\mathcal{G}^k, \mathcal{T}^k, \mathcal{V}_{\mathrm{D1Q2}}, \mathrm{C}_{\mathrm{lin}}, h_i^k, I_j^k, B_j^k), \tag{5.14}$$

in more detail, where we use the notation introduced in Section 2.5. Remark that the original problem at hand is given by P_{lin}, see (5.1). We observe that the k-th subproblem inherits the velocity discretization $\mathcal{V}_{\mathrm{D1Q2}}$ and the collision model $\mathrm{C}_{\mathrm{lin}}$ from P_{lin}. The computational grid of S^k, i.e.,

$$\mathcal{G}^k = \{x_1^k, x_2^k, \ldots, x_{h(k)+1}^k\}$$

consists of $h(k)+1$ grid points, where $h(k) = 2\tilde{h}(k)$ is the chosen history depth of time level $t = t_k \in \mathcal{T}$. We identify the boundary point of $\mathcal{G}^k$ for which c_j points outwards of the computational grid as a so-called *intersection grid point* denoted by $x_I^k \in \Gamma^k \subset \mathcal{G}^k$. In this context, j is the index of the boundary population B_j, which is needed in the original problem. The name stems from the fact, that $x_I^k = x_b \in \mathcal{G}^k \cap \mathcal{G}$ holds. Exemplarily, if the BC (5.12) is applied for the right boundary of the original computational grid $\mathcal{G}$, then the intersection grid point x_I^k lies on the left boundary of $\mathcal{G}^k$. The corresponding populations of S^k, denoted by h_i^k, shall be initialized according to

$$I_i^k(x_I^k) = f_i(x_b, t_k - h(k)\Delta t), \tag{5.15}$$

and for other grid points by

$$I_i^k(x_m^k) = E_i, \quad x_m^k \in \mathcal{G}^k \setminus \{x_I^k\}. \tag{5.16}$$

The inward boundary populations at the intersection grid point are set by

$$B_i^k(x_I^k, t_0^k + j\Delta t) = f_i\big(x_b, t_k - (h(k)-j)\Delta t\big), \quad j = 1, \ldots, h(k)-1, \tag{5.17}$$

whereas the populations at the opposite boundary of $\mathcal{G}^k$ can be chosen arbitrarily, since they are not relevant for our purpose. The initial time of the k-th subproblem is $t_0^k = t_k - h(k)\Delta t$, and $h(k)$ time steps have to be simulated, thus we have

$$\mathcal{T}^k = \{t_0^k, t_0^k + \Delta t, t_0^k + 2\Delta t, \ldots, t_0^k + h(k)\Delta t\}. \tag{5.18}$$

The boundary population $B_j(x_b, t_k)$ of the original problem is given, equivalently to (5.12), at the latest time level of the k-th subproblem:

$$B_j\big(x_b, t_k\big) = h_j^k\big(x_I^k, t_0^k + h(k)\Delta t\big). \tag{5.19}$$

The DABC, formulated by (5.19) as the solution of a separate LB simulation (S^k), can be generalized to arbitrary collision models. We focus on this issue in the upcoming sections.

5.3 One-dimensional discrete artificial boundary conditions for a nonlinear collision model

The aim of this section is to perform the first of two steps to generalize the DABC (5.19) to arbitrary (nonlinear) collision models in any dimension. In this first step, we formulate

a DABC for a nonlinear collision model, but remain in one space dimension. The second step, done in the subsequent section, transfers the idea to higher dimensions, eventually leading to the most general formulation of our DABCs.

Here, we consider the usual LBE (2.40) in one space dimension with a BGK collision operator and three velocities $c_0 = 0$, $c_1 = -c$ and $c_2 = c$ (D1Q3). In contrast to the previous sections we now focus on

$$P_{\text{non-lin}} = \mathtt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}_{\text{D1Q3}}, \mathrm{C}_{\text{non-lin}}, f_j, I_j, B_j), \tag{5.20}$$

with nonlinear $\mathrm{C}_{\text{non-lin}}$ given by (2.39) and (2.41). In this case the post-collision populations (4.4) read

$$g_0(\boldsymbol{f}) = g_0(f_0, f_1, f_2) = f_0 - \frac{\omega}{3}\frac{{f_0}^2 + {f_1}^2 + {f_2}^2 - f_0(f_1 + f_2) - 10 f_1 f_2}{f_0 + f_1 + f_2}, \tag{5.21a}$$

$$g_{1,2}(\boldsymbol{f}) = g_{1,2}(f_0, f_1, f_2) = f_{1,2} + \frac{\omega}{6}\frac{{f_0}^2 + {f_1}^2 + {f_2}^2 - f_0(f_1 + f_2) - 10 f_1 f_2}{f_0 + f_1 + f_2}, \tag{5.21b}$$

and are clearly nonlinear. Noteworthy, the use of a BGK collision term is done solely for simplicity, but it does not represent a limitation of the derived BC. Also the equilibrium distribution used in the BGK model can be taken differently.

For $P_{\text{non-lin}}$ we can formulate, at least theoretically, an exact BC using digraphs as explained in Section 4.1. Unfortunately, due to the nonlinearity such an exact BC cannot be written as an efficient formula. Therefore, a DABC for a nonlinear collision model cannot be derived by an analytical approximation of this theoretical exact DABC. The expedient is to incorporate the nonlinearity into the approximate DABC for the linear collision model. To this end, we use the subproblem interpretation (5.19) of the BC, but solve a subproblem having a nonlinear collision model as well. More precisely, the efficient approximate DABC has the history depth as a free parameter. In order to compute the inward boundary population $B_j\big(x_b, t_k\big)$ at time level $t = t_k$ we solve the k-th subproblem S^k; it reads for the nonlinear case:

$$S^k = \mathtt{LB}(\mathcal{G}^k, \mathcal{T}^k, \mathcal{V}_{\text{D1Q3}}, \mathrm{C}_{\text{non-lin}}, h_j^k, I_j^k, B_j^k). \tag{5.22}$$

As in the linear case, the subproblem's grid $\mathcal{G}^k$ has $h(k)$ grid points, which are initialized by (5.15) and (5.16). Again, the physical location of the grid points give a logical extension of $\mathcal{G}$, i.e., $\mathcal{G}^k \subseteq \mathcal{G}^{\text{ref}}$. Past boundary nodes of the original problem serve as BC for the subproblem, see (5.17). For the simulation of S^k we use the same nonlinear model $\mathrm{C}_{\text{non-lin}}$ as for the original problem (5.20). So the unknown population given by (5.19) depends nonlinearly on past boundary populations and the initial data (5.16). Figs. 5.4 and 5.5 summarize the procedure of the DABC.

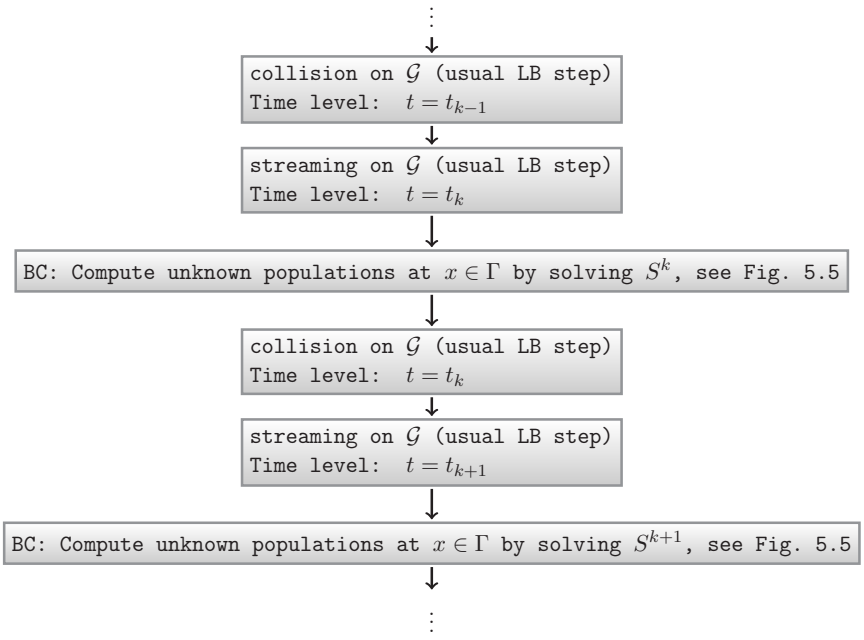


Figure 5.4: *Flow chart explaining the application of the DABC* – After an usual collision step we advance in time by the usual streaming step of the algorithm. At the new time level, say $t = t_k$, the inward populations are computed by solving the k-th subproblem S^k. The flow chart of Fig. 5.5 gives more details for the subproblem.

5.4 General discrete artificial boundary conditions in any dimension

The DABCs considered so far dealt only with one-dimensional problems. In principle the idea of the DABC can be directly transferred also to higher dimensional problems

$$P_{\text{gen}} = \texttt{LB}(\mathcal{G}, \mathcal{T}, \mathcal{V}, \text{C}, f_j, I_j, B_j),$$

without any restrictions to the velocity model $\mathcal{V}$ and the collision model C. As in the one-dimensional case we solve subproblems to achieve the missing boundary populations. However in higher dimensions the choice of a subproblem's (here k-th subproblem) computational grid $\mathcal{G}^k$ needs special consideration. Another issue arises from dealing with corners and edges. In the current section we discuss these issues.

5.4.1 Computational grid

In the one-dimensional problems considered above there are at most two open boundary points. These points were considered independently since they are on the left and right boundary of the computational grid and thus are not related to each other. There, for each boundary point exactly one population has to be determined and each population is found by solving a separate subproblem. In the general (higher dimensional) problem P_{gen} the open boundary points $\Gamma \subseteq \mathcal{B}$ are generally not separated from each other. As a beneficial

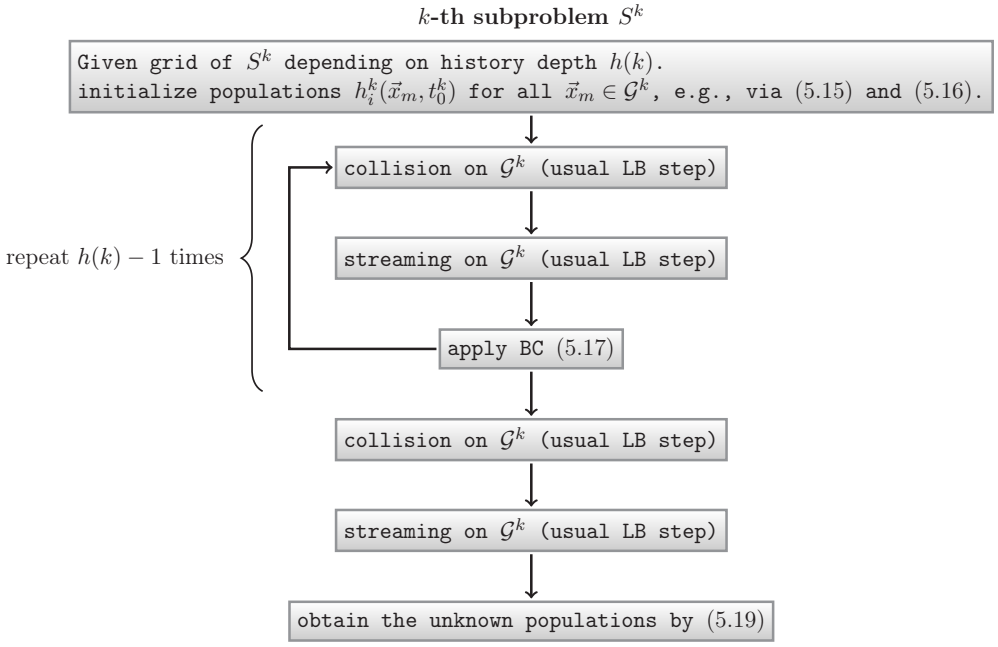


Figure 5.5: *Flow chart explaining the procedure of a subproblem* – The k-th subproblem S^k is solved to compute inward populations at time level $t = t_k$, see Fig. 5.4. It is first initialized and then evolves according to the usual approach of a LB algorithm. After all iterations have been computed, the inward populations of the original problem are given by outward populations of S^k.

consequence we do not need to consider one individual subproblem for each $\boldsymbol{x}_b \in \Gamma$, but one subproblem for all $h(k)$-*connected* open boundary points. In this context, two boundary points $\boldsymbol{x}_{b,1}$ and $\boldsymbol{x}_{b,2}$ are called $h(k)$-*connected* if one can construct a reasonable path out of lattice vectors $\boldsymbol{c}_i \in \mathcal{V}$ from $\boldsymbol{x}_{b,1}$ to $\boldsymbol{x}_{b,2}$ of length at most $h(k)$ on the grid $\mathcal{G}^{\text{ref}} \setminus \mathcal{G} \cup \{\boldsymbol{x}_{b,1}, \boldsymbol{x}_{b,2}\}$. Otherwise we call the nodes $h(k)$-*disconnected*. We give two theoretical examples in Figs. 5.6 and 5.7, one for the connected case and one for the disconnected case.

Analogue to the one-dimensional explanations above, $\Gamma \subseteq \mathcal{B}$ denotes the set of open boundary points in the original problem P_{gen}. Let the set Γ be split into as many as possible disjunctive non-empty subsets

$$\Gamma = \Gamma_1 \cup \ldots \cup \Gamma_N, \tag{5.23}$$

such that for a given history depth the splitting satisfies the following condition:

- There are no $h(k)$-connected points $\boldsymbol{x}_{b,1} \in \Gamma_l$ and $\boldsymbol{x}_{b,2} \in \Gamma_m$ if $l \neq m$.

Then, the boundary values $B_j(\boldsymbol{x}_b)$ for all $\boldsymbol{x}_b \in \Gamma_m$ (m fixed) are computed by the same subproblem. In fact, whether the boundary values $B_j(\boldsymbol{x}_{b,1})$ and $B_j(\boldsymbol{x}_{b,2})$ for two arbitrary boundary points $\boldsymbol{x}_{b,1}, \boldsymbol{x}_{b,2} \in \Gamma$ can be obtained by considering the same subproblem depends on the reference grid $\mathcal{G}^{\text{ref}}$ and the chosen history depth $h(k)$. A decision based only on the location of the boundary point is not possible, as one can also see in Fig. 5.6. In the following we describe the DABC for one set Γ_m, but omit the subscript. To get the

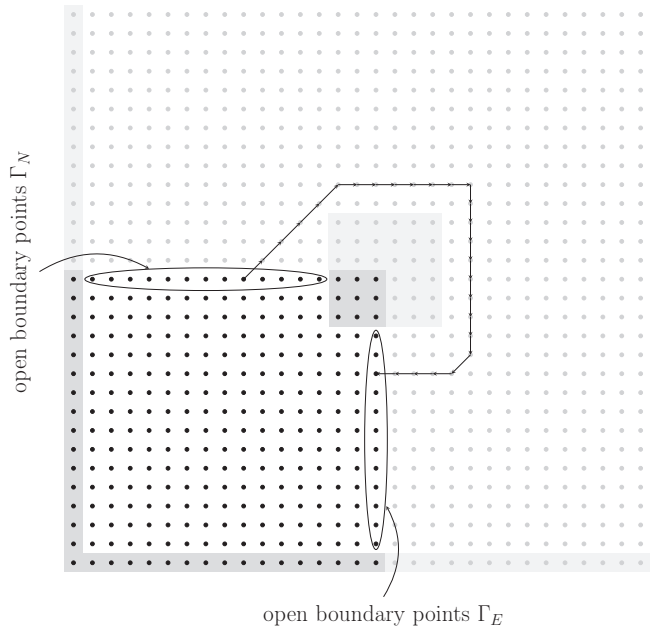


Figure 5.6: *Theoretical example of $h(k)$-connected boundaries* – Let the history depth be $h(k) \geq 19$. This theoretical example shows open boundary points Γ_N and Γ_E, where the points $x \in \Gamma_N$ and $y \in \Gamma_E$ are $h(k)$-connected, one path is shown exemplary. Therefore the splitting (5.23) is given by only one subset $\Gamma_1 = \Gamma_N \cup \Gamma_E$. (Note, history depths $h(k) < 19$ yield a splitting $\Gamma_1 = \Gamma_N$ and $\Gamma_2 = \Gamma_E$ instead, i.e., for $h(k) < 19$ Γ_N and Γ_E are $h(k)$-disconnected.)

boundary values $B_j(\boldsymbol{x}_b, t_k)$, $\boldsymbol{x}_b \in \Gamma \subseteq \mathcal{B} \subset \mathcal{G}$, $j \in \mathcal{I}_{\mathcal{G}}^{-}(\boldsymbol{x}_b)$, we consider one subproblem

$$S^k = \mathtt{LB}(\mathcal{G}^k, \mathcal{T}^k, \mathcal{V}, \mathrm{C}, h_j^k, I_j^k, B_j^k).$$

The set of discrete time points $\mathcal{T}^k$ is chosen exactly as in the one-dimensional case explained above, meaning by (5.18), where $h(k)$ is the chosen history depth and $t_0^k = t_k - h(k)\Delta t$. Also the size of the computational grid $\mathcal{G}^k$ is determined by the history depth. Unfortunately, there is no general choice of $\mathcal{G}^k$, which is correct for all possible cases, but $\mathcal{G}^k$ depends strongly on the given problem and the boundary arrangement. Next, we explain the individual choice for $\mathcal{G}^k$ based on its theoretical requirements and we give a construction principle.

Theoretical requirements of the computational grid For the computational grid $\mathcal{G}^k$ we select points belonging to a sufficiently larger lattice extension of $\mathcal{G}$, meaning $\mathcal{G}^k \subseteq \mathcal{G}^{\mathrm{ref}}$. The computational grid $\mathcal{G}^k$ is chosen such that the open boundary points of P_{gen} lie in the intersection $\mathcal{H}^k = \mathcal{G} \cap \mathcal{G}^k$, i.e., $\Gamma \subseteq \mathcal{H}^k$. Moreover, the intersection shall not contain regular lattice points: $\mathcal{F} \cap \mathcal{G}^k = \varnothing$. We emphasize that, however, we possibly have to add some further boundary points $\boldsymbol{x} \in \mathcal{B} \setminus \Gamma$ to the computational grid $\mathcal{G}^k$. By this inclusion we avoid having boundary points $\boldsymbol{x}_b \in \mathcal{B}^k$ in the problem S^k, which do neither belong to $\mathcal{B}^{\mathrm{ref}}$ nor to $\mathcal{H}^k$. This is important because otherwise there could be a lack of missing populations in these points when solving S^k. Note that these additional boundary points are equipped with regular BCs.

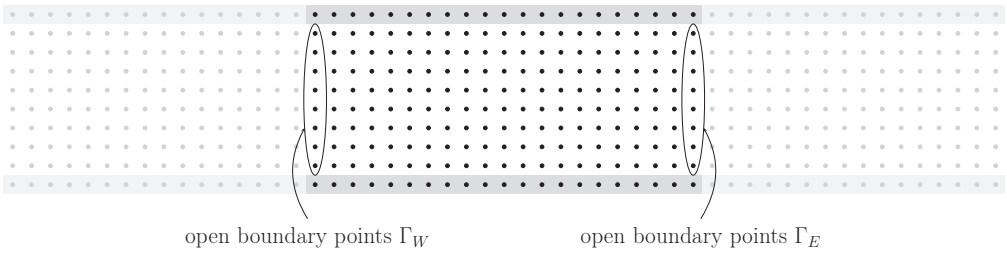


Figure 5.7: *Theoretical example of disconnected boundaries* – The set of open boundary points Γ_W and Γ_E are separated, such that $x \in \Gamma_W$ and $y \in \Gamma_E$ are $h(k)$-disconnected for all history depths. Therefore the splitting (5.23) is given by $\Gamma_1 = \Gamma_W$ and $\Gamma_2 = \Gamma_E$.

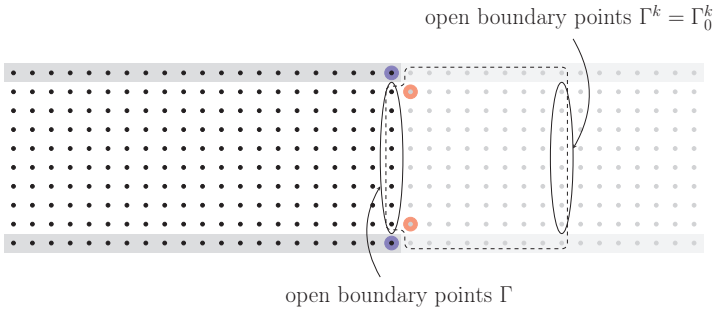


Figure 5.8: *Construction of the computational grid of a subproblem* – The dashed container describes the set $\mathcal{G}_0^k$. The red nodes are boundary points of $\mathcal{G}_0^k$ without any prescribed information, such that the blue points have to be added. This union gives the computational grid $\mathcal{G}^k$ of the k-th subproblem.

Construction of the computational grid We consider the example of a two-dimensional channel flow in Fig. 5.8 to better understand the construction of the computational grid. We describe an approach yielding $\mathcal{G}^k$ for a given history depth $h(k)$: Firstly, we get a preliminary version $\tilde{\mathcal{G}}_0^k \subset \mathcal{G}^k$ by considering the reference grid $\mathcal{G}^{\text{ref}}$, and then disregard all points of $\mathcal{G}$ except for Γ, i.e.,

$$\tilde{\mathcal{G}}_0^k = \Gamma \cup (\mathcal{G}^{\text{ref}} \setminus \mathcal{G}).$$

Next, we also disregard all points $\boldsymbol{x}_1 \in \mathcal{G}^{\text{ref}} \setminus \mathcal{G}$, which are $h(k)$-disconnected to each point $\boldsymbol{x}_2 \in \Gamma$. By this we get rid of irrelevant grid points, the outcome is denoted as $\mathcal{G}_0^k \subset \tilde{\mathcal{G}}_0^k$ and in Fig. 5.8 it is shown by all points surrounded by the dashed box. If the thus obtained grid has boundary points $\boldsymbol{x}_b \in \mathcal{G}_0^k \setminus \Gamma_0^k$, for which no physical values are prescribed (i.e., $\boldsymbol{x}_b \notin \mathcal{B}^{\text{ref}}$), we add the corresponding missing adjacent points. These points are always elements of $\mathcal{B} \setminus \Gamma$. After this addition, the computational grid $\mathcal{G}^k$ is found. In the illustrated example, this means the blue points have to be added, since otherwise the red points would be boundary points for which no condition is given. Thus for the given example, the grid $\mathcal{G}^k$ consists of $\mathcal{G}_0^k$ and the two blue points.

5.4.2 Solving the subproblem

Given the computational grid $\mathcal{G}^k$ the k-th subproblem can be solved, provided initial and boundary values I_i^k and B_j^k are present, respectively. Basically, the same rules as above hold also for the general problem P_{gen}. The relevant boundary values B_j^k for S^k are either given by information from the problem P_{gen} (if $\boldsymbol{x} \in \mathcal{H}^k$) or by known physical conditions (if $\boldsymbol{x} \in \mathcal{B}^k \setminus \mathcal{H}^k$). Here we state the BC for the k-th subproblem only for the intersection points:

$$B_j^k(\boldsymbol{x}, t) = f_j\big(\boldsymbol{x}, t\big), \quad \boldsymbol{x} \in \mathcal{H}^k,\ j \in \mathcal{I}_{\mathcal{G}^k}^-(\boldsymbol{x}),\ t \in \mathcal{T}^k.$$

The initial populations are chosen consistently in the intersection points, i.e.,

$$I_i^k(\boldsymbol{x}) = f_i\big(\boldsymbol{x}, t_0^k\big), \quad \boldsymbol{x} \in \mathcal{H}^k,\ t_0^k \in \mathcal{T}^k.$$

Lastly, all other points are initialized following the procedure of one-dimensional problems by

$$I_i^k(\boldsymbol{x}) = f_i^{\text{ref}}(\boldsymbol{x}, t_0), \quad \boldsymbol{x} \in \mathcal{G}^k \setminus \mathcal{H}^k. \tag{5.24}$$

Specially, this means the interior points of the subproblem are initialized in the same way as the reference populations would have been initialized at the very beginning at $t = t_0$. In the next section, we end this chapter with further investigations of DABCs.

5.5 Analysis of the discrete artificial boundary conditions

The investigations of the DABCs are valid for all problems P_{gen}. However they are explained in an intelligible manner assuming one-dimensional problems. We first discuss the error sources of DABCs, afterwards we explain the relation of history depth and initialization. In the third section we show how DABCs are efficiently implemented and finally, the last section discusses their computational costs.

5.5.1 Error sources

To measure the accuracy of the DABC we refer to the ITBC of Section 3.1. Let us first assume that all interior populations up to time level $t = t_k$ are exactly given in the sense of an ITBC, i.e.,

$$f_i(x, t_s) = f_i^{\text{ref}}(x, t_s), \quad \forall x \in \mathcal{G},\ t_0 \le t_s < t_k. \tag{5.25}$$

This assumption is true at the beginning of the simulation, if the history depth satisfies

$$h(k) = \min\{k,\ H\}. \tag{5.26}$$

We discuss the error $\delta_j(x_b, t_k)$ of the inward population $f_j(x_b, t_k)$ at the current time level $t = t_k$. It is given as the deviation from the ideal value:

$$\delta_j(x_b, t_k) := \left| f_j(x_b, t_k) - f_j^{\text{ref}}(x_b, t_k) \right|.$$

Clearly, computing the inward population by the DABC with history depth $h(k) = k$ yields $\delta_j(x_b, t_k) = 0$. Generally, for a smaller history depth $h(k) < k$, solving the k-th subproblem initialized with (5.15) and (5.16) leads to non-zero errors. However, if the initial interior populations of the k-th subproblem were chosen, instead of (5.16), by the (theoretical) values

$$I_i^k(x_m^k) = f_i^{\mathrm{ref}}\big(x_m^k, t_k - h(k)\Delta t\big) \quad \forall x_m^k \in \mathcal{G}^k \setminus \{x_I^k\}, \tag{5.27}$$

we would obtain a vanishing error as well. It is important to see that this statement holds for any history depth $h(k) > 0$. The statement can be validated, since using the initialization (5.27) in the k-th subproblem, the subproblem would only describe the relevant processes in the first $(h(k) + 1)$ grid points of $\mathcal{G}^{\mathrm{ref}}$ not lying in $\mathcal{G}$ anymore.

As soon as an error $\delta_j(x_b, t_k) > 0$ is done for the first time, due to an inexact initialization, the assumption (5.25) is violated. Then, these (inexact) boundary populations from time level $t = t_k$ are used in the BC (5.17) of subproblems for later time levels. Thus, even if the exact initialization (5.27) is used for later time levels, an error is created by inexact boundary populations in the subproblems.

To sum up, any error of the DABC is introduced by assigning inexact data at the initialization of the subproblems. We can classify two sources for the error $\delta_j(x_b, t_k)$, both related to subproblems' initialization:

E1 The use of inexact initial values in the k-th subproblem gives an error directly linked with the initialization.

E2 An indirect error results from inexact initial values in preceding subproblems (s-th subproblem, with $s < k$). They lead to inexact populations in earlier time levels, which enter the k-th subproblem via the BC, see (5.17).

We point out, that under assumption (5.25) no errors of type E2 exist. However, an error of type E1 causes the assumption to be violated, such that an indirect error E2 follows later.

5.5.2 On history depth and initialization

As we have seen above, the errors of the DABC are related to the initialization of the subproblems. However, it is intuitively clear that a large history depth close to the exact value $h(k) = k$ yields smaller errors. This intuitive view will be confirmed by numerical experiments, for which we refer to Chapter 6. The aim of the current section is to work out the link between the history depth and the quality of initialization. We concentrate on the directly linked error E1, that is assuming (5.25) holds.

Let us consider the k-th subproblem (5.22) with an enlarged history depth $h(k)+s$, $s \geq 1$, called $\tilde{S}^k$:

$$\tilde{S}^k = \mathtt{LB}\left(\tilde{\mathcal{G}}^k_{h(k)+s}, \tilde{\mathcal{T}}^k, \mathcal{V}_{\mathrm{D1Q3}}, \mathrm{C}_{\text{non-lin}}, \tilde{h}_j^k, \tilde{I}_j^k, \tilde{B}_j^k\right).$$

We recall that its computational grid $\tilde{\mathcal{G}}^k_{h(k)+s}$ consists of an intersection point x_I^k and a number of $h(k) + s$ adjacent grid points. The subscript emphasizes the dependence on

the history depth. The eventually required population $\tilde{h}_j^k\big(x_I^k, t_k\big)$ (see also equation 5.19) depends on information from all nodes

$$n = \big(x_m^k, t_k - (h(k)+s)\big), \quad x_m^k \in \tilde{\mathcal{G}}_{h(k)+s}^k.$$

After each time step of $\tilde{S}^k$ the number of relevant grid points decreases, such that after s steps only information from nodes

$$n = (x_m^k, t_k - h(k)\Delta t), \quad x_m^k \in \left\{ x \in \tilde{\mathcal{G}}_{h(k)+s}^k \;\middle|\; x = x_I^k - c_j r \Delta t,\ r = 0, 1, \ldots, h(k) \right\}$$

is relevant. Obviously this set is equivalent to $\mathcal{G}_{h(k)}^k$, that is the computational grid of the usual k-th subproblem S^k. Therefore, the boundary population obtained by solving $\tilde{S}^k$ is equivalent to solving S^k when it is initialized as follows:

$$I_i^k(x_m^k) = \tilde{h}_i^k\big(x_m^k, t_k - h(k)\Delta t\big), \quad x_m^k \in \mathcal{G}^k. \tag{5.28}$$

Thus, simulating the problem $\tilde{S}^k$ for s time steps, yields an alternative initialization for S^k, which intuitively leads to smaller errors. Also for a later subproblem, that is a $(k+J)$-th subproblem, $J \geq 1$, an initialization according to (5.28) can be reasonable, if J is not too large. Provided the corresponding history depth satisfies $h(k+J) \leq h(k)$ we can initialize the subproblem by

$$I_i^{k+J}(x_m^{k+J}) = \tilde{h}_i^k\Big(x_m^{k+J}, t_k - h(k)\Delta t\Big), \quad x_m^{k+J} \in \mathcal{G}^{k+J}. \tag{5.29}$$

Note that $h(k+J) \leq h(k)$ implies $\mathcal{G}^{k+J} \subseteq \mathcal{G}_{h(k)}^k$, and so all populations on the right hand side are given.

We may conclude that the use of a larger history depth is related to the goal of obtaining a better initialization of a subproblem. As we have seen before, ideally, the initialization is done with reference populations, see (5.27). Thus any populations raising the claim to be close to the unknown reference populations seem to serve as a promising initialization. For example one might think of an initialization as

$$I_i^k(x_m^k) = f_i^{\mathrm{eq}}\big(x_I^k, t_k - h(k)\Delta t\big), \quad x_m^k \in \mathcal{G}^k, \tag{5.30}$$

which describes a constant extrapolation of a past boundary population.

5.5.3 Implementation

The solution of the k-th subproblem is required at time level $t = t_k$. However, it is efficient to start the simulation of the k-th subproblem not just when it is needed, i.e., when time has reached level $t = t_k$, but as soon as necessary data is available. That is, the simulation of the k-th subproblem should start at time level $t = t_0^k = t_{k-h(k)}$. In general a LB simulation is structured as

$$\ldots \to C \to S \to BC \to C \to S \to BC \to C \to S \to BC \to \ldots,$$

where we have the steps collision (C), streaming (S) and boundary condition (BC). Recall that the streaming step gives the time stepping. Also note that the information from

intermediate time levels, $t_0^k < t < t_k$, is entering a subproblem only in the boundary condition step via (5.17). In fact, all information exchange between the original problem and a subproblem is done either in the BC step, see (5.17) and (5.19), or for the initialization of a subproblem, see (5.15). Fig. 5.9 visualizes the time range of several subproblems, using (5.26) with maximal history depth $H = 5$. A black box indicates that a certain subproblem (ordinate) gives the inward boundary population of the original problem at the corresponding time level (abscissa). Furthermore, the figure shows labeled boxes in the row of a k-th subproblem. A box with label `j` states that information from time level $t = t_{k-j}$ is entering the computation of the k-th subproblem.

Inspecting a fixed time level in Fig. 5.9, e.g., $t = t_{15}$, we see that there are multiple subproblems using data from that time level. We focus on the boundary condition steps (BC) and briefly demonstrate that there is no lack of information, when we compute all these subproblems simultaneously. After the streaming step the inward population of the original problem is given by the solution of the corresponding subproblem. The information entering the subproblems which were initialized in previous time levels is given from the original problem. Also the same information is used for all subproblems which have to be initialized at the current time level. After the subsequent collision and streaming step the situation remains the same. We remark that no information is exchanged between subproblems directly.

In an efficient implementation like explained above, one has to handle several subproblems simultaneously, but then the collision and streaming steps of all subproblems can be done in alignment with those of the original problem. Hence one inherits the computational benefits of the LBM, e.g., the possibility of parallel computing.

5.5.4 Computational costs

The computational effort of the DABC chiefly depends on three quantities. In particular each subproblem solved is a LB problem related to the original problem P_{gen}, hence the chosen velocity discretization $\mathcal{V}$ and the collision term C (e.g., SRT or MRT) are crucial. Due to this dependence, the computational costs of the DABC cannot be stated by a generally valid number of required arithmetic operations. But, the computational costs can be measured in terms of corresponding grid points.

If the history depth is chosen according to (5.26), then the computational grids $\mathcal{G}^j$ and $\mathcal{G}^m$ of a j-th and m-th subproblem, $j, m \geq H$ are equal. So for simplicity, we may assume that all subproblems have the same number of grid points J. The implementation strategy described in Section 5.5.3 considers H subproblems simultaneously, where the collision and streaming steps of the subproblems and the original problem are computed simultaneously. Therefore one does not need to distinguish whether a grid point belongs to a subproblem or to the original problem. It follows, the total computational costs of the DABC is equivalent to a lattice enlargement by $H \cdot J$ nodes. Indeed it is even less, since with each time step a certain number of grid points of a subproblem becomes irrelevant and thus could be omitted. Therefore, besides $\mathcal{V}$ and C the third key quantity determining the computational effort is the maximal history depth H.

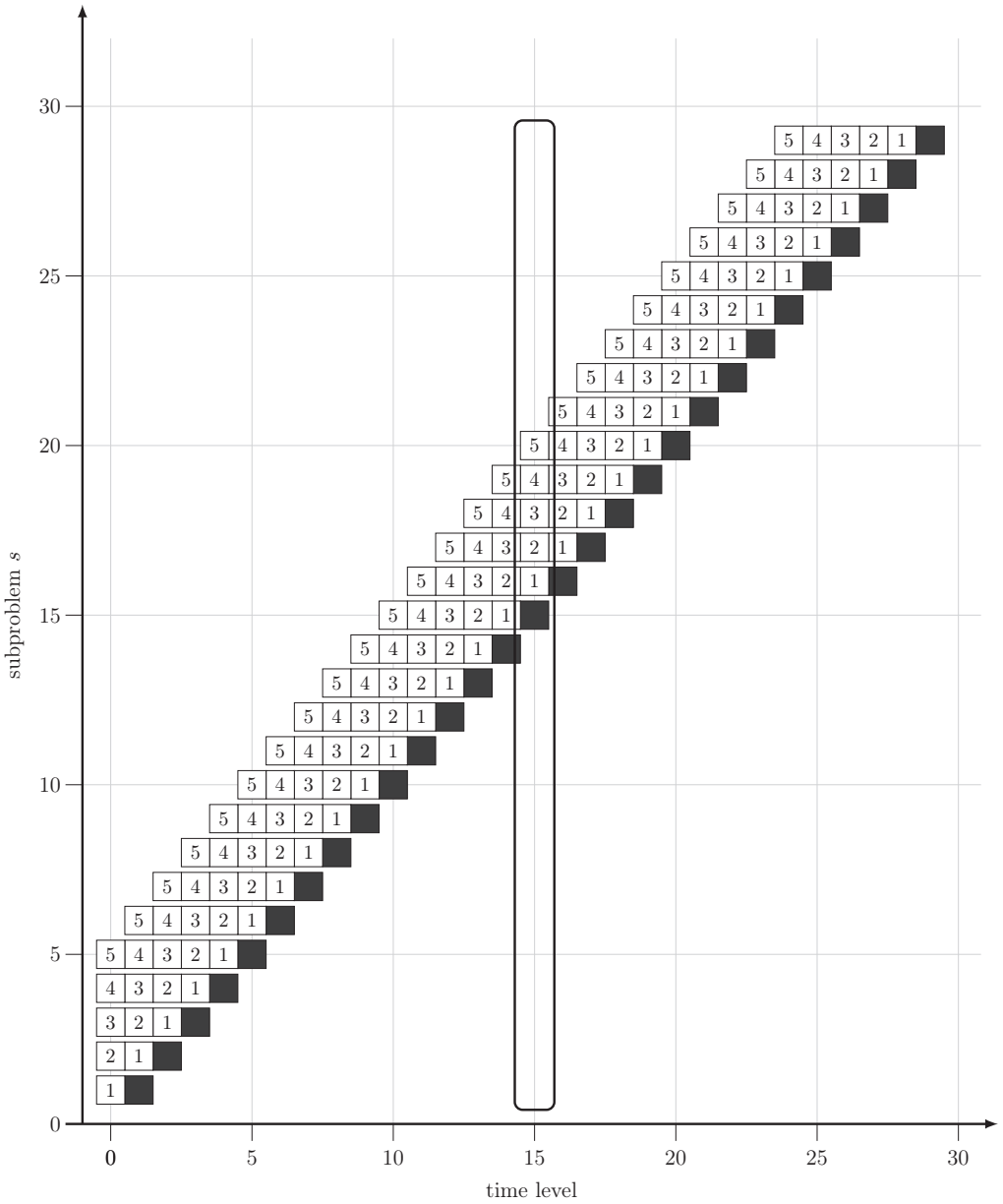


Figure 5.9: *Subproblems' data dependence from different time levels* – The illustration assumes a maximal history depth of 5 and shows subproblems on the ordinate and time level on the abscissa. A black box indicates when a subproblem determines the inward population at a certain time level. Other boxes indicate that information from the corresponding time level enters the corresponding subproblem. The presence of several subproblem in a vertical line (e.g., within the rectangle) shows the potential to compute several subproblems simultaneously.

6 Chapter 6
Numerical results

In this chapter we present the results of several numerical simulations ranging from one to three dimensions. In Section 6.1, we first give a condense overview of all BCs, which are used in numerical simulations. The simulations are ordered with respect to their spatial dimension. Section 6.2 considers only the (approximate) one-dimensional DABC for the linear problem (4.1). A test case with the same initialization is considered in Section 6.3, but for the nonlinear case resulting in a one-dimensional pressure pulse, whose evolution is described by the Navier-Stokes equations. Thereby we present results for all our BCs in comparison to chosen benchmark BCs. The first problem considered in two dimensions, in Section 6.4, is used to visualize the working principle of the DABC at the example of a concentric pressure wave. We simulate an isolated vorticity wave as a benchmark problem for two-dimensional BCs in Section 6.5. The second two-dimensional test case for the artificial BCs is based on a theoretical example presented in Section 6.6. Here we simulate plane waves and investigate the errors of the BCs with respect to the angle of incidence. Finally, we show the results of a three-dimensional simulation in Section 6.7, where we consider the flow past an obstacle within a square duct.

6.1 Terminology

Reference solution For all simulations presented in the sequel, we always compute a *reference solution* by applying an *ideal transparent boundary condition* (ITBC) to all open boundaries of the computational domain. The ITBC has been described in Section 3.1. All quantities corresponding to the reference solution are marked with the superscript $\cdot^{\mathrm{ref}}$ and curves in the plots are named `ITBC`.

Error measurement Using this reference solution, we measure the effects of other BCs by computing the *maximal absolute error*

$$\mathrm{Err}_z(t) := \max_{\boldsymbol{x}\in\mathcal{G}} \left| z(\boldsymbol{x},t) - z^{\mathrm{ref}}(\boldsymbol{x},t) \right|, \tag{6.1}$$

where z stands for any possible quantity. Moreover, we consider an ℓ^2*-error*

$$\ell^2_z(t) := \|z(\boldsymbol{x},t) - z^{\mathrm{ref}}(\boldsymbol{x},t)\|_{\ell^2} = \sqrt{\sum_{\boldsymbol{x}\in\mathcal{G}} \big(z(\boldsymbol{x},t) - z^{\mathrm{ref}}(\boldsymbol{x},t)\big)^2}. \tag{6.2}$$

and a *maximal* ℓ^2*-error*

$$L_z := \|z(\boldsymbol{x},t) - z^{\mathrm{ref}}(\boldsymbol{x},t)\|_{\ell^2,\infty} = \max_{t\in\mathcal{T}} \ell^2_z(t). \tag{6.3}$$

Discrete artificial boundary conditions We distinguish two DABCs in our simulations. On the one hand, the DABCs (5.2) for the problem (5.1) with the linear collision model (4.2), are denoted by `linDABC`. On the other hand, the general DABCs (for nonlinear collision models) introduced in Sections 5.3 and 5.4 are denoted by `DABC`. The `DABC` are initialized according to (5.24).

Characteristic boundary conditions We distinguish CBCs (Sections 3.3 and 3.4) with respect to their basic system, see Section 3.3.1 for details. If they are based on the LODI equations, we refer to them as `LODI`, whereas `CBC` denotes a boundary condition based on the general basic system (3.17). Unless no name affix is used, the characteristic systems are solved with an explicit Euler step (3.24) and the resulting Dirichlet condition is transferred by (3.26). The following name affixes are possible:

`-Neq` the Dirichlet values are transferred with linear extrapolation of non-equilibrium portions, see (3.27).

`-RK` the characteristic system is solved with the Runge-Kutta scheme (3.25).

`vis-` the extended characteristic system (3.28) is considered.

Impedance boundary conditions The IBC as described in Section 3.2.2 is denoted by `ImpBC`. Moreover, we consider Schlaffer's isotropic adaptation (`isoImpBC`), we refer to [93] for a detailed description. The implementation of `isoImpBC` is based on the source code given in [93]. Both BCs cannot be applied at corners or edges, therefore all results denoted by `ImpBC` and `isoImpBC` use a LODI based BC at corners and edges.

Perfectly Matched Layers In Section 3.2 we presented different kind of PMLs for the LBM. For the numerical tests we focus on the PML formulation of Najafi-Yazdi and Mongeau [77] denoted by `PML`. The absorption coefficient σ is linearly increasing from zero to $\sigma_{\max} = 0.05$ in the PML zone, which has a thickness of 30 grid points. At the boundary of the PML zone (grid points $\boldsymbol{x}_b$), we set the inward populations by $f_j(\boldsymbol{x}_b, t) = I_j^{\mathrm{ref}}(\boldsymbol{x}_b)$ throughout.

6.2 Approximate discrete artificial boundary conditions for linear collision models

In this test case for `linDABC`, the mass density ρ, see (2.45), is initialized at $t = 0$ as [56]

$$\rho_0(x) = \begin{cases} 1 & \text{for } x \le 0.3 \text{ or } x \ge 0.7, \\ 1 + 0.4 \cdot \exp\left(\frac{-15^{-2}}{(x-0.3)^2}\right) \cdot \exp\left(\frac{-15^{-2}}{(x-0.7)^2}\right) & \text{for } 0.3 < x < 0.7, \end{cases} \tag{6.4}$$

such that $\rho_0 \in C^\infty(\mathbb{R})$. The corresponding populations for the LB simulation are initialized by equilibrium distributions (2.44) evaluated with (6.4). The computational grid $\mathcal{G}$ corresponds to an equidistant discretization of the interval $[0, 1]$ with step size $h = 0.005$. The free parameters in the simulation are the advection velocity $a \in (-1, 1)$, the collision parameter $\omega \in (0, 2)$ and the maximal history depth H.

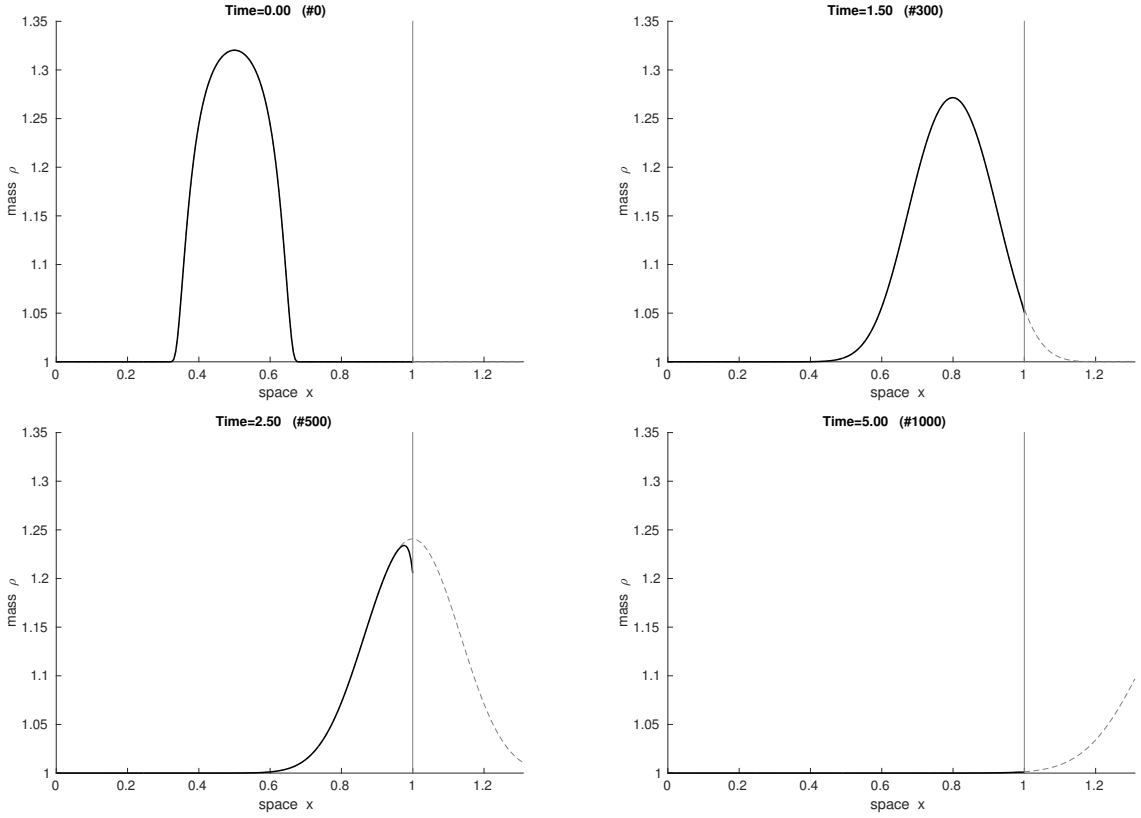


Figure 6.1: *Mass density evolution in a LB simulation with linear collision operator* – The four plots show the mass density at initial time, as well as after 300, 500 and 1000 time steps, respectively. The dashed lines show the reference signal modeling an ITBC at $x = 1$. The solid lines refer to the density profiles for a simulation with `linDABC` ($a = 0.2$, $H = 5$ and $\omega = 1.1$).

To begin with, we select $a = 0.2$, $H = 5$ and $\omega = 1.1$ and plot the mass density for different time levels in Fig. 6.1, where `linDABC` is used at the right boundary. Here, also the reference solution is plotted, given by the dashed line. Note that we have plotted the reference curve also in the exterior of the computational domain, for a better readability. As one can clearly see `linDABC` leads to errors near the boundary, which are obviously transported out of the computational domain. To measure the error in dependence of the free parameters, we compute the maximal absolute error (6.1), here Err_ρ, for different simulations, where always two of the three free parameters are fixed and only one varies. In Fig. 6.2 the varying parameter is the maximal history depth H and we see that the error is decreasing when we take a larger value for H. This result confirms impressively the analytical investigations of Section 5.1. In Fig. 6.3, we kept all parameters except for the advection velocity a and the relaxation parameter ω, respectively. We observe that the error is smaller the faster the initial peak comes upon the artificial boundary. Furthermore, the larger the relaxation parameter ω is chosen the smaller are the maximal absolute errors. The coefficient of numerical diffusion, see (2.49), is becoming small when ω is increased. We presume that the higher numerical diffusion for small ω causes the larger errors.

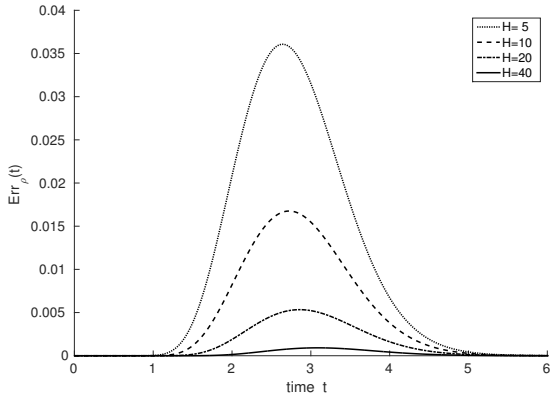


Figure 6.2: *Errors of the approximate DABC with varying history depths* – The plot shows the maximal absolute errors $\mathrm{Err}_\rho(t)$ for the approximate `linDABC`. The maximal history depth varies, while the advection velocity and the collision parameter are set to $a = 0.2$ and $\omega = 1.1$, respectively. Errors decrease for larger maximal history depths.

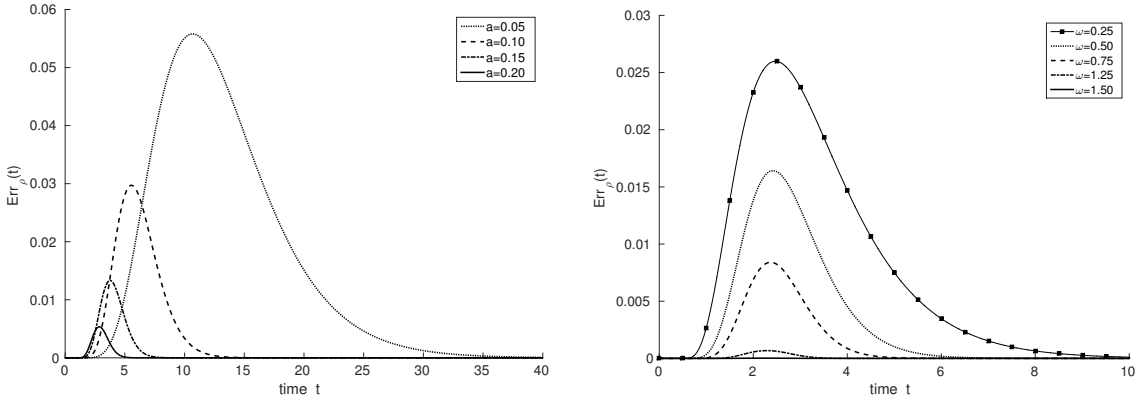


Figure 6.3: *The DABC with varying advection velocity and collision parameter* – Both plots show the maximal absolute errors $\mathrm{Err}_\rho(t)$ for the approximate `linDABC` with a maximal history depth of $H = 20$. The plot on the left gives the errors when the advection velocity a is varied, whereas the collision parameter is set to $\omega = 1.1$. Vice versa, on the right plot the collision parameter varies, while we set $a = 0.25$.

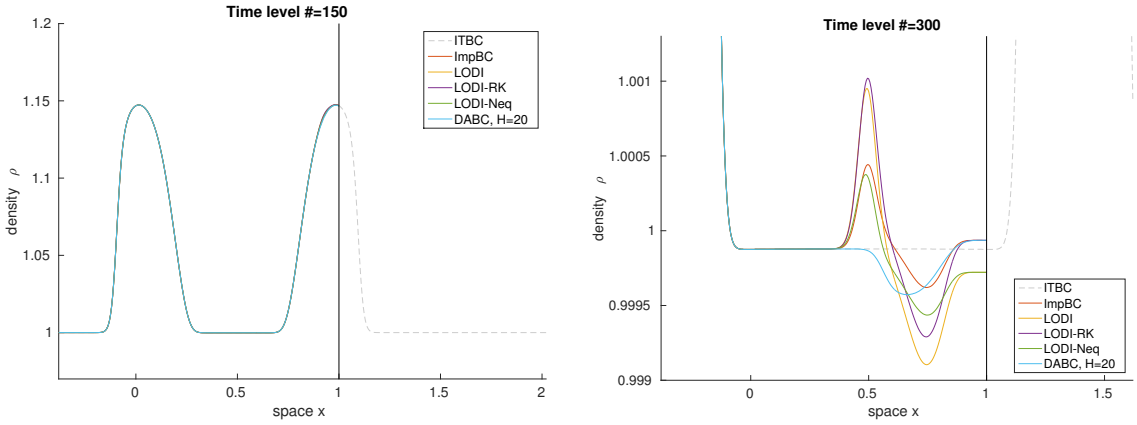


Figure 6.4: *Mass density in a 1D LB simulation with nonlinear collision operator* – The two plots show the mass density for several BCs after 150 and 300 iterations, respectively. The simulation is done with the settings $u_0 = 0$ and $\tau = 0.8$. The deflections visible in the range $0 \leq x \leq 1$ on the right plot are reflected waves generated at the boundary at $x = 1$.

6.3 One-dimensional pressure wave

For the next test, the mass profile (6.4) is used in a usual LB simulation, i.e., where the collision model is given by (2.41) with (2.39), and thus the macroscopic quantities evolve according to the Navier-Stokes equations. Analogously, the populations are initialized by an evaluation of the equilibrium distribution (2.39), where the fluid velocity u is homogeneously given by $u = u_0$. We test different choices for u_0, including negative and positive values. We consider the interval $[-1, 1]$, where an ITBC is applied to the left boundary, such that the right boundary is used to test different BCs. In Fig. 6.4 the density is plotted for a simulation with $u_0 = 0$ and $\tau = 0.8$ after 150 and 300 time steps, respectively. We see that the initial pressure pulse is split into two pressure waves, where one is going left and the other going right. In the right plot, corresponding to time level 300, we see some unphysical waves, which were generated at the boundary at $x = 1$ and are traveling back into the computational domain. All BCs fail to reach the ideal level of the signal (density given by dashed line) after the pulse has left the computational domain. We also see that the boundary conditions `LODI` have different errors, depending on how they are implemented, especially, a consideration of the non-equilibrium portions for `LODI` is advantageous here.

Recall that our CBC coincides with `LODI` in the one-dimensional case, therefore the novel BCs in this simulation are `DABC`. To test the BC in more detail, we compute the maximal ℓ^2-error (6.3) for simulations with different values for the parameters u_0 and τ. In Figs. 6.5 and 6.6 we see this kind of error for varying velocities u_0 and relaxation times τ, respectively. The error is only shown for the density, the error curves for velocity are omitted, since they look very similar (except for scaling). The `DABC` is considered for maximal history depths $H = 10, 20, 40$ in Figs. 6.5 and 6.6, and we can clearly see that errors decrease for larger maximal history depths. In Fig. 6.5 it is noteworthy, that `vis-LODI` on the left plot ($\tau = 0.8$) has the largest errors, while for $\tau = 1$ this BC behaves much better. It is not shown on the third plot ($\tau = 1.5$), since the error of `vis-LODI` is several magnitudes higher compared to errors of the other BCs. For $\tau = 1$, `LODI` and `LODI-Neq` are clearly

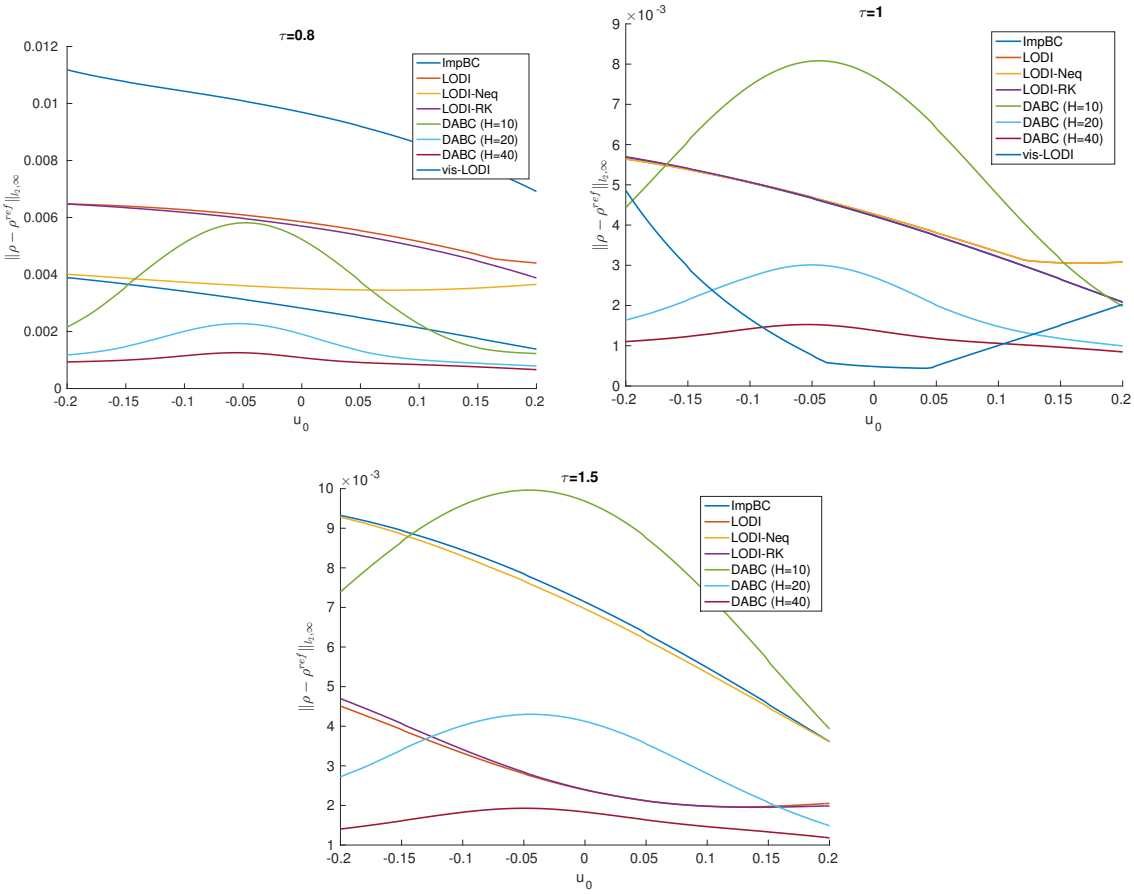


Figure 6.5: *Errors of boundary conditions with respect to velocity* – On the plots the maximal ℓ^2-error of the mass density is illustrated for several BCs. The simulation refers to a one-dimensional pressure wave. We have selected three relaxation times $\tau = 0.8$, 1, 1.5.

equal, but also `ImpBC` and `LODI-RK` are visually matching. Strikingly, for $\tau = 1.5$ `LODI` has smaller errors than `LODI-Neq`, a feature which cannot be observed for the first two plots. The properties derived from Fig. 6.5 are also visible in the plots of Fig. 6.6. That is, we can see that the error of `vis-LODI` is strongly depending on τ, where smallest errors are obtained for a relaxation time about $\tau = 1$. When comparing `LODI` with `LODI-Neq` the former has smaller errors for $\tau > 1$. The curves of `ImpBC` and `LODI-RK` intersect around $\tau = 1$, while `LODI-RK` and `LODI` are very similar throughout.

As a final result, Fig. 6.7 visualizes errors similar to Fig. 6.6, but only for DABCs with different history depths. Comparing only history depths of one category (even or odd), a larger value decreases the error, as expected. However, we can clearly see a different behavior for even and odd values, and taking odd history depths seems to be advantageous. We came to the same conclusion in one of our previous works after some more tests, see [48].

We conclude that our DABCs can be tuned (higher history depth, preferring an odd value) to outperform existing BCs with respect to accuracy. For an extensive investigation of the DABCs with respect to an initialization according to (5.29) we refer to our article [48]. Moreover, in [48] an additional test case (an acoustic sinusoidal signal) can be found.

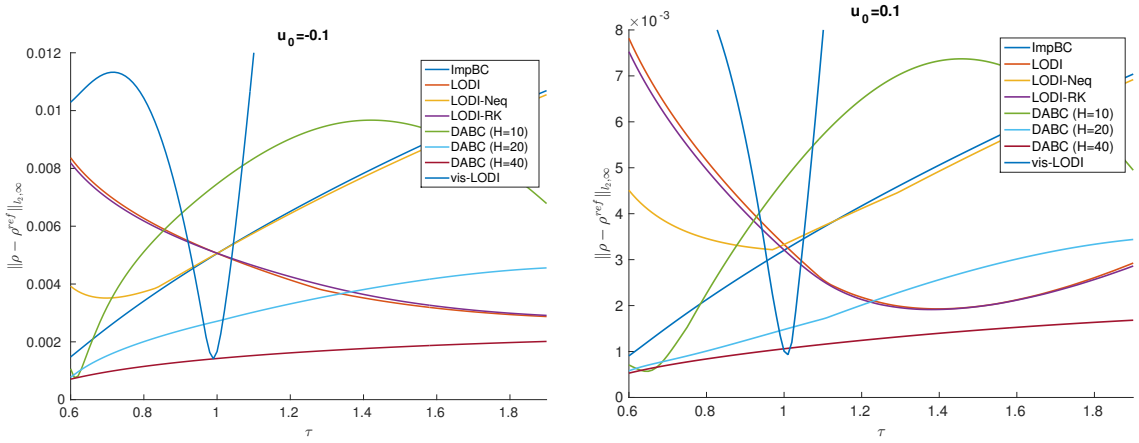


Figure 6.6: *Errors of boundary conditions with respect to relaxation time* – On the plots the maximal ℓ^2-error of the mass density is illustrated for several BCs. The simulation refers to a one-dimensional pressure wave. Two velocities have been selected. The curve related to `vis-LODI` continues in the exterior of the chosen range.

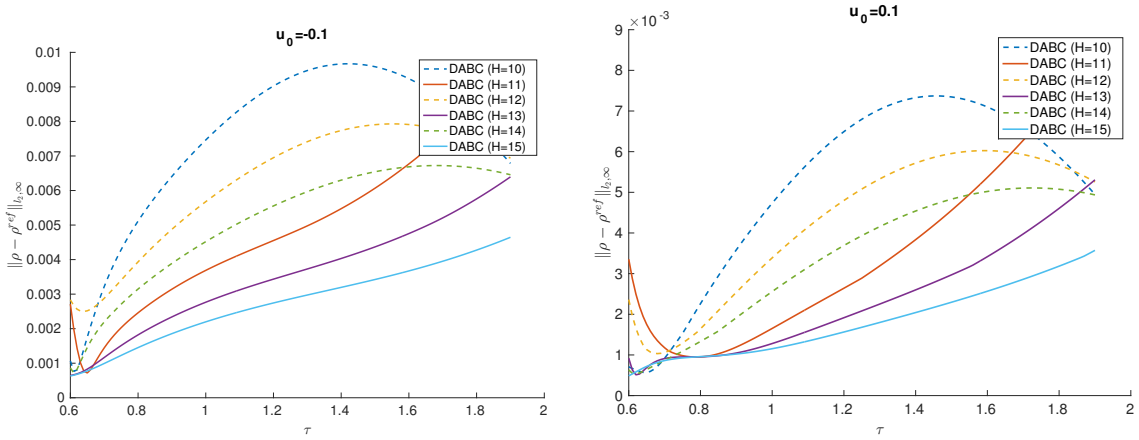


Figure 6.7: *Errors of DABCs for different maximal history depths* – Situation as in Fig. 6.6, but here we focus on the errors for `DABC`, when the maximal history depths is varied. The dashed curves are related to an even maximal history depth, while the solid curves refer to errors of an odd maximal history depth.

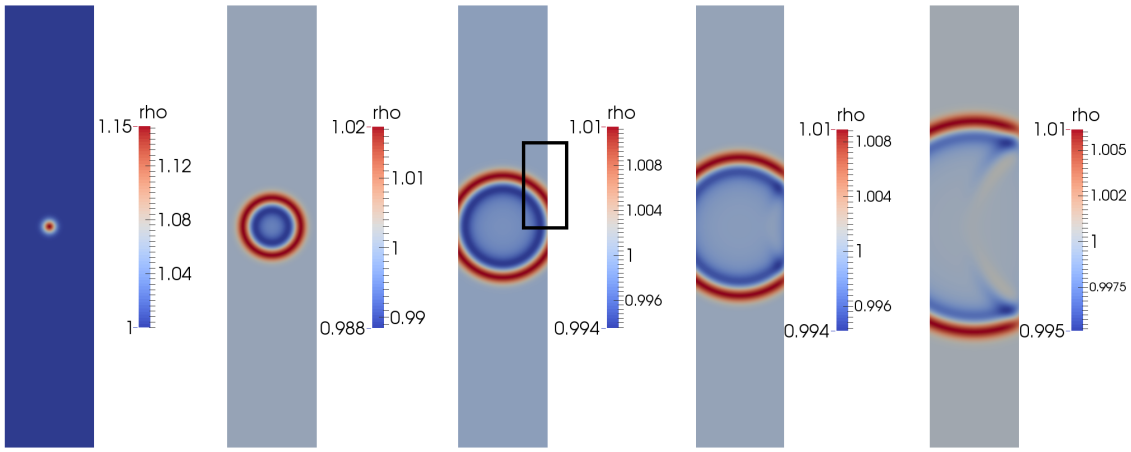


Figure 6.8: *Pressure pulse in a two-dimensional simulation* – The temporal evolution of a Gaussian pressure pulse is illustrated. Snapshots show the corresponding mass density at the time levels $t \in \{t_0, t_{100}, t_{185}, t_{255}, t_{400}\}$ (from left to right). At the left boundary an ITBC is applied. On the right boundary we use `DABC`, here, a reflected wave can be observed traveling back into the interior. The square rectangle on the third snapshot gives the domain considered in Fig. 6.9.

6.4 Visual interpretation of discrete artificial boundary conditions

The first two-dimensional simulation focuses on a visualization of the working principle of our DABC. We consider a rectangular computational domain $[-1, 1] \times [-5, 5]$ with a Gaussian pressure pulse:

$$p(x, y) = p_0 + (p_{\max} - p_0) \exp\left(\frac{-(x^2 + y^2)}{2s^2}\right),$$

where pressure values are related to the density by (3.11). We set $s = 0.1$ and the pressures p_0 and $p_{\max}$ according to $\rho_0 = 1$ and $\rho_{\max} = 1.15$, respectively. The fluid velocity is homogeneously set to zero, the maximal history depth to $H = 40$ and $\tau = 1$. We simulate the pressure pulse on a computational grid of dimension 201×1001, representing the computational domain with an equidistant spatial step-size of $h = 0.01$. We apply periodic BCs at the top and bottom boundary, an ITBC at the left boundary and the DABC on the right boundary. In Fig. 6.8 the density profile is plotted for different time levels. We can clearly see a reflection generated at the right boundary, traveling leftwards. To explain the working principle of the DABC visually, we consider the situation at time $t = t_{185}$. In fact, we only focus on the marked section in the third snapshot of Fig. 6.8. In Fig. 6.9 the mass density profile of the original problem is shown for time $t = t_{185}$. Moreover, a cut-out (cropped in x_2 direction) of the mass density of the 185-th subproblem is shown, for selected time levels $t \in \mathcal{T}^{185}$ during the simulation of the subproblem, while the profile of the original problem is frozen. The first plot shows the subproblem's density profile at its initialization. The intermediate time levels are not relevant for finding the unknown populations of the original problem, but only the last plot (final time) is relevant. We can see that at final time the profiles of the original problem and of the subproblem seem to match. However, in fact there is a small mismatch, which causes the reflection visible in

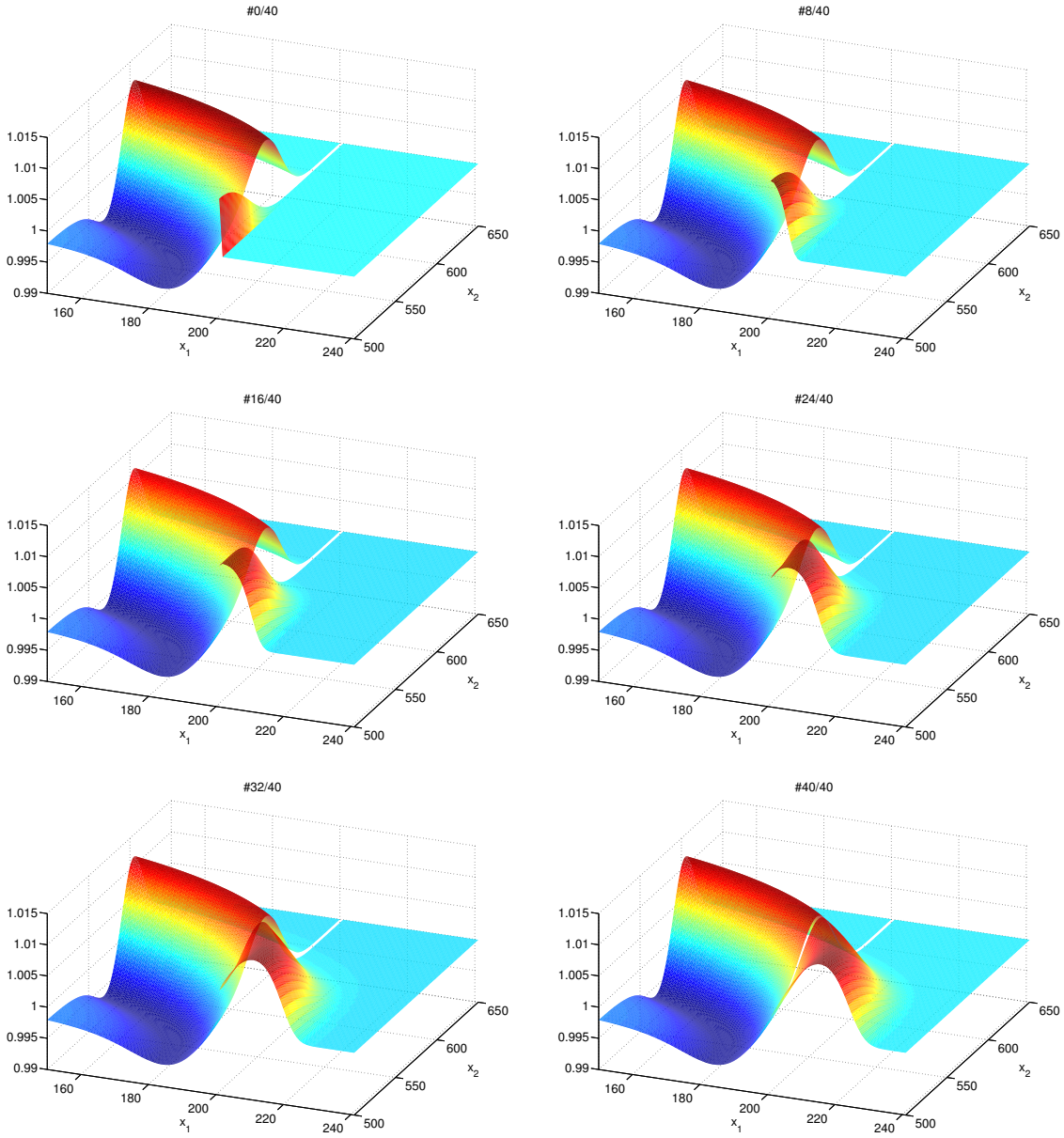


Figure 6.9: *Density profile evolution in a subproblem* – The plots show the temporal evolution of the 185-th subproblem ($x_1 \in [201, 241]$) in the domain marked in Fig. 6.8. The profile of the original problem ($x_1 \leq 200$) is shown for fixed time $t = t_{185}$. From the perspective of a BC for the original problem only the last plot (after iteration 40) is relevant.

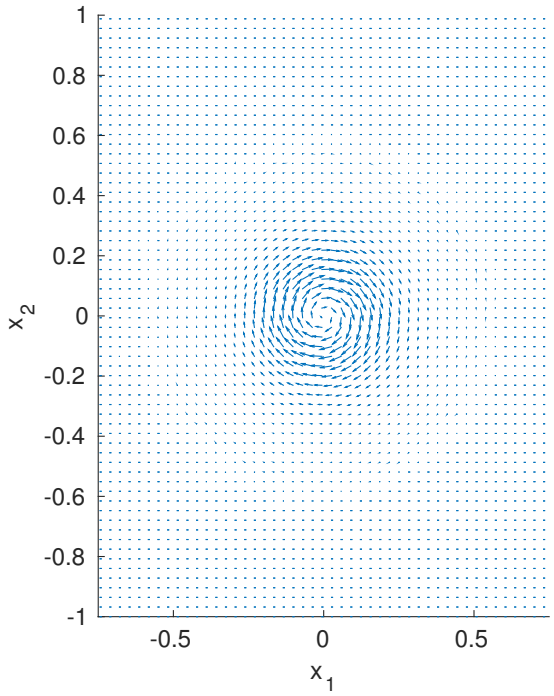


Figure 6.10: *An isolated vortex* – The plot shows the vector field of the velocity $\boldsymbol{v}(\boldsymbol{x})$, which is used to model an isolated vortex.

Fig. 6.8. Thus, the DABC constructs a suitable extension of the computational domain in fictitious points, which provides then the unknown populations. This interpretation also clarifies that the initialization of a subproblem determines the accuracy of the DABC.

6.5 An isolated vorticity wave

As the first test of BCs in two-dimensional applications we consider a vorticity wave with the following initial conditions inspired by an example in [18]:

$$\rho_0(\boldsymbol{x}) = 1, \qquad \boldsymbol{u}_0(\boldsymbol{x}) = \boldsymbol{a}(\boldsymbol{x}) + \begin{cases} \boldsymbol{0} & \text{for } \boldsymbol{x} \notin \mathcal{B}_{0.7}(\boldsymbol{0}), \\ \boldsymbol{v}(\boldsymbol{x}) & \text{for } \boldsymbol{x} \in \mathcal{B}_{0.7}(\boldsymbol{0}) \end{cases},$$

where $\mathcal{B}_r(\boldsymbol{c}) = \{\boldsymbol{x} \in \mathbb{R}^2 \mid \|\boldsymbol{x} - \boldsymbol{c}\|_2 < r^2\}$ is the open disc around $\boldsymbol{c} \in \mathbb{R}^2$ with radius $r > 0$ and

$$\boldsymbol{a}(\boldsymbol{x}) = \begin{pmatrix} 0.2c_s \\ 0 \end{pmatrix}, \qquad \boldsymbol{v}(\boldsymbol{x}) = \begin{pmatrix} \frac{1}{2}x_2 \exp\left(-(\ln 2)\frac{x_1^2+x_2^2}{b^2}\right) \\ -\frac{1}{2}x_1 \exp\left(-(\ln 2)\frac{x_1^2+x_2^2}{b^2}\right) \end{pmatrix}$$

with $b = 0.15$. Hence, the vortex (shown in Fig. 6.10), modeled by $\boldsymbol{v}(\boldsymbol{x})$ is traveling in positive x_1-direction due to $\boldsymbol{a}(\boldsymbol{x})$. The restriction to $\mathcal{B}_{0.7}(\boldsymbol{0})$ causes a negligible discontinuity at the boundary of the disc, meaning a jump of the velocity magnitude less than 10^{-11}. Here we introduced the compact support of the vortex to ensure the vortex is not interacting with the boundary of the computational grid at initial time. The grid is given by a discretization of $[-\frac{3}{4}, \frac{3}{4}] \times [-3, 3]$ with $h = \frac{1}{250}$. We select periodic BCs in

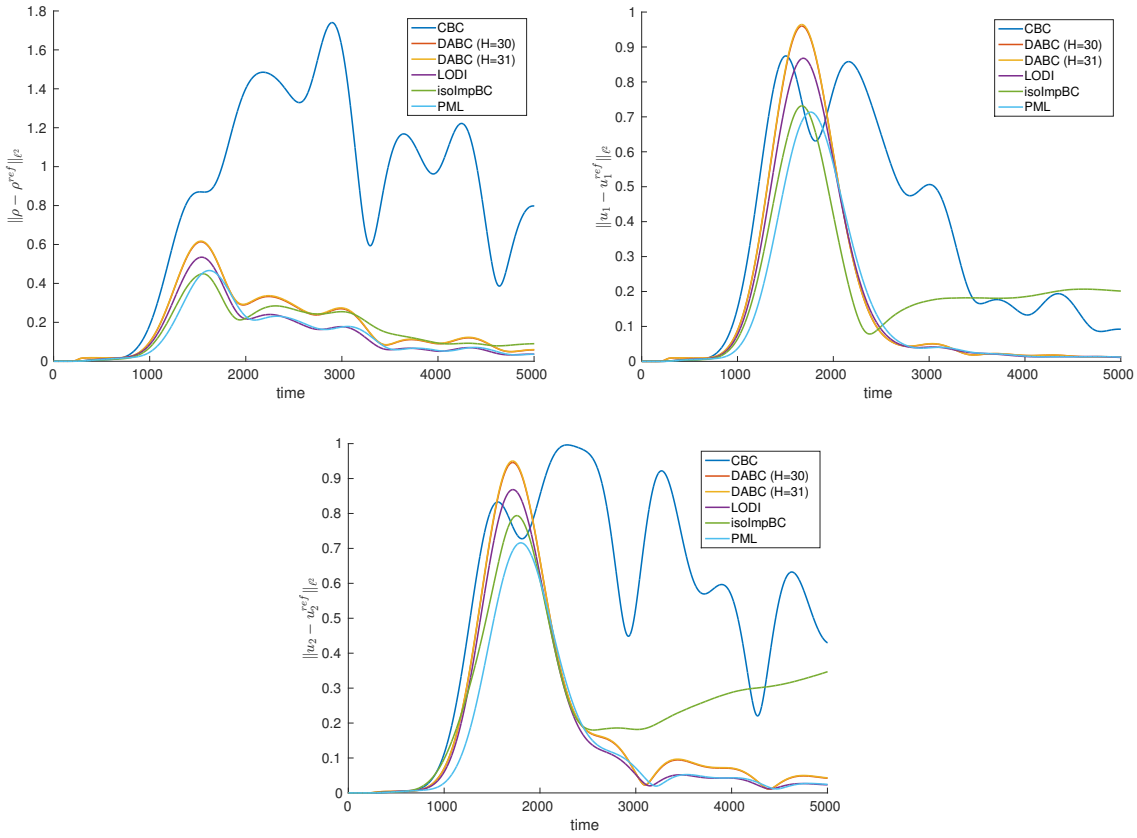


Figure 6.11: *Errors in the simulation of an isolated vorticity wave* – The plots show the ℓ^2-errors in the mass density and both velocity components for a simulation of an isolated vorticity wave. With an ITBC the vortex leaves the computational domain after around 3200 time steps. Errors of `DABC` ($H = 30$) are barely visible smaller than errors of `DABC` ($H = 31$).

x_2-direction and an ITBC at the left boundary. Therefore, the right boundary is used to test different BCs. For the simulations we have taken a relaxation parameter of $\tau = 0.8$. Having an ITBC at the right boundary the vortex would leave the computational domain completely after around 3200 time steps. Fig. 6.11 shows the time dependent ℓ^2-error (6.2) for all macroscopic quantities. In contrast to the one-dimensional pressure pulse, no significant difference between the DABCs with even/odd maximal history depth of $H = 30$ and $H = 31$, respectively, can be observed. Two further striking observations can be done. First, the errors of `CBC` are unmistakably larger, hence for the given test case the incorporation of non-orthogonal derivatives has a negative impact on the accuracy. This result is striking, since larger errors of `CBC` compared to `LODI` have not been observed in previous simulations [46]. Second, the errors in the velocity (second and third plot) for `isoImpBC` become worse after around 2500 iterations, whereas the density error does not show this behavior.

For the simulation with `DABC` at the right boundary we visualize the transverse velocity at selected time levels by contour plots in Fig. 6.12. Already at the second plot (time level $t = t_{1000}$) one can see that the contour lines are squeezed together in an unphysical way. For later time levels we clearly see a deformation which results from a non-ideal behavior of the BC. In Appendix E we also show contour plots for simulations with other BCs. For

a better comparability all contour levels are chosen equally across all figures. The levels are chosen equidistantly in the range given by the color bar.

Moreover, we consider the vorticity during the simulation. The vorticity of a three-dimensional flow is defined as the curl of the velocity [3]:

$$\boldsymbol{\omega}(\boldsymbol{x},t) = \mathrm{rot}(\boldsymbol{u}(\boldsymbol{x},t)) = \boldsymbol{\nabla} \times \boldsymbol{u}(\boldsymbol{x},t).$$

In the two-dimensional problem at hand, where the velocity does not depend on the z component of the velocity, it can be simply expressed by a scalar valued function

$$\omega_3(\boldsymbol{x},t) = \frac{\partial u_2(\boldsymbol{x},t)}{\partial x_1} - \frac{\partial u_1(\boldsymbol{x},t)}{\partial x_2}, \tag{6.5}$$

since the first and second component of $\boldsymbol{\omega}$ are always zero. In Fig. 6.13 we show the vorticity (6.5) for the simulation with `DABC`, whereas results for other BCs are given in Appendix E. Again all contour levels are chosen equally across all figures, except for the last contour plot corresponding to time level $t = t_{2500}$. Obviously, the DABC leads to unphysical effects, however the effects are small and seem to be confined to the region near the boundary.

6.6 Plane waves with different angles of incidence

In this academic example we consider a plane wave impinging on the boundary with different angles of incidence $\phi > 0$. With this test we compare the magnitude of reflection with respect to ϕ for several BCs. The corresponding LB simulation is initialized with equilibrium populations corresponding to zero velocity and a density according to

$$\rho_0(x_1,x_2) = 1 + \frac{1}{10}\exp\left(\frac{-\hat{x}_1(x_1,x_2)^2}{2s^2}\right),$$

with $s = \frac{1}{50}$ and transformed locations

$$\begin{pmatrix}\hat{x}_1(x_1,x_2)\\ \hat{x}_2(x_1,x_2)\end{pmatrix} = \begin{pmatrix}\cos(\phi\frac{\pi}{180}) & \sin(\phi\frac{\pi}{180})\\ -\sin(\phi\frac{\pi}{180}) & \cos(\phi\frac{\pi}{180})\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix}.$$

The computational grid is a discretization of a rectangular domain with step-size $h = \frac{1}{200}$, its size chosen such that the density is maximal in the bottom right corner, as also shown in Fig. 6.14. The dimension of the domain depends on the angle of incidence ϕ. Due to initialization the density profile splits into two spreading plane waves. In the simulation we use ITBCs for the left, bottom and upper boundary. Moreover, for the first 250 iterations of the simulation also the right boundary is equipped with an ITBC. Considering the maxima, one plane wave is traveling away from the bottom right corner along the bottom boundary, while the other is traveling along the right boundary, see also Fig. 6.14. The size of the computational domain is chosen, such that after 250 iterations the plane wave traveling along the right boundary has reached one third of its total height. From the 251^{st} iteration on, we change the BC at the right boundary, and measure the maximal

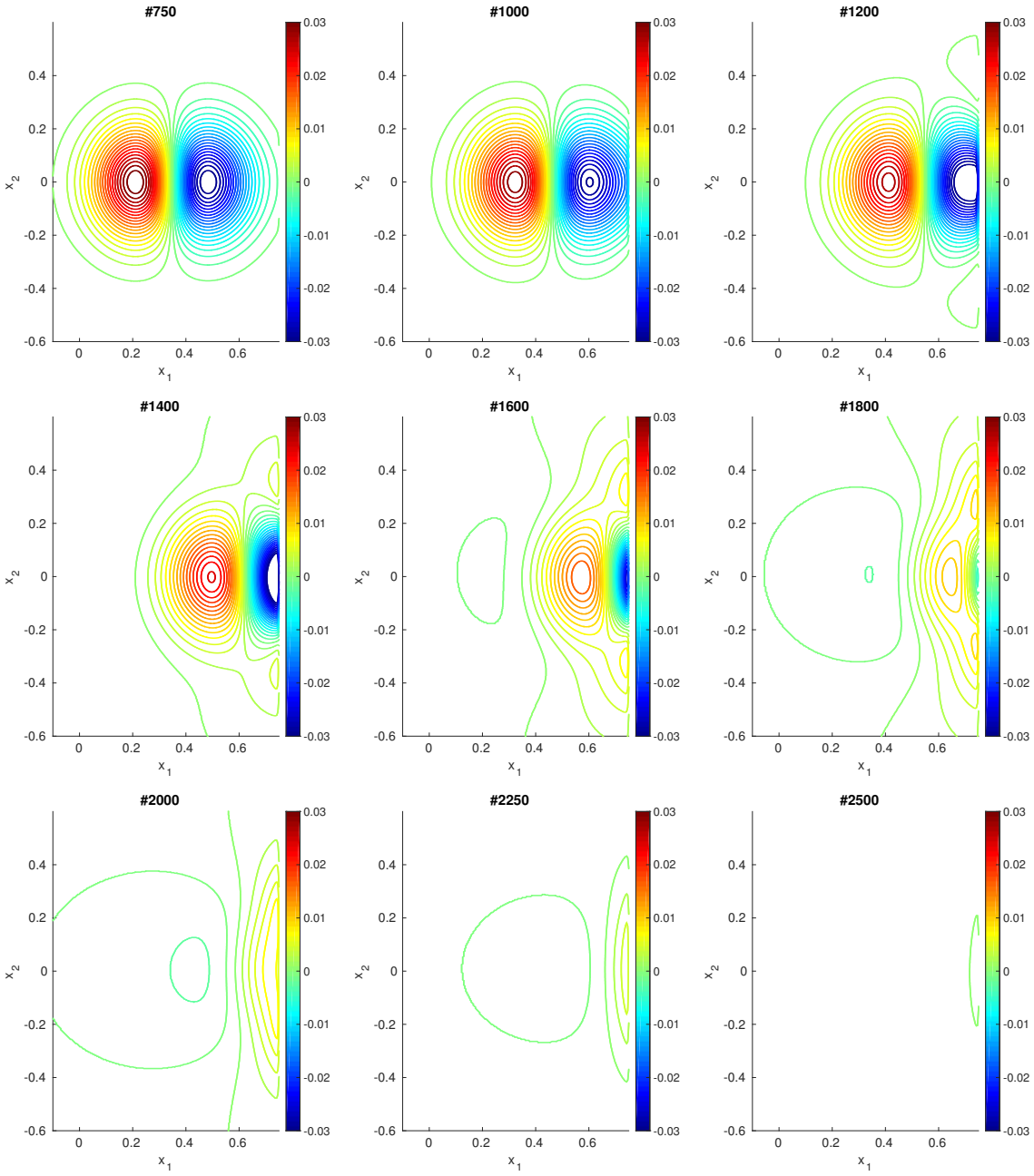


Figure 6.12: *Iso-transverse-velocity contours for an isolated vortex using* `DABC` – The plots show the evolution of contour lines of the non-orthogonal velocity component u_2 in the simulation of an isolated vorticity wave using a DABC at the right boundary. The maximal history depth for the DABC is $H = 30$.

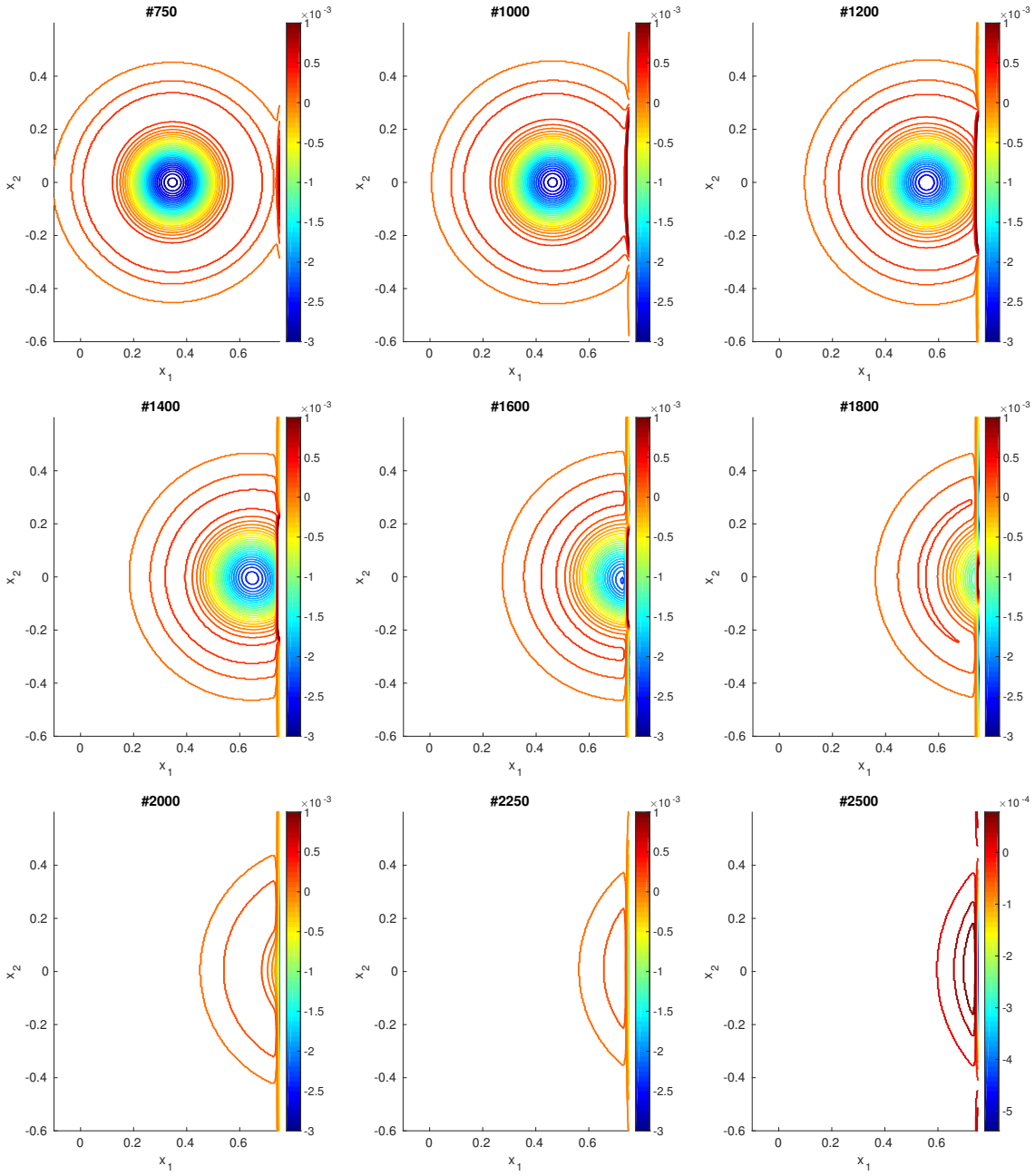


Figure 6.13: *Iso-vorticity contours for an isolated vortex using* `DABC` – The contour plots show the evolution of the vorticity (6.5) using a DABC at the right boundary. The maximal history depth for the DABC is $H = 30$. Except for the last plot, the contour levels are the same for all time levels.

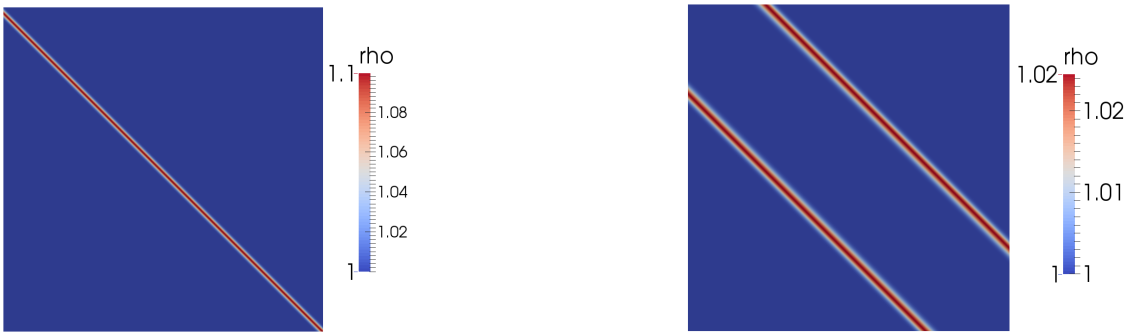


Figure 6.14: *Density profiles for plane wave simulation* – The left plot shows the density initialization ($t = t_0$). The initial setting generates two spreading plane waves. After 250 iterations (computed with ITBCs) the situation on the right picture is reached.

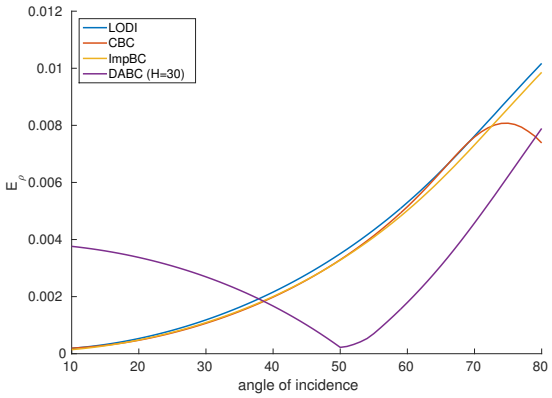


Figure 6.15: *Maximal errors for plane waves with respect to the angle of incidence* – The maximal error E_ρ, (6.6), is plotted for different angles of incidence in the range from 10° to 80°.

error

$$E_z := \max_{t\in\mathcal{T}} \max_{\boldsymbol{x}\in\mathcal{F}_b} \left| z(\boldsymbol{x},t) - z^{\mathrm{ref}}(\boldsymbol{x},t) \right|, \quad \mathcal{F}_b = \left\{ \boldsymbol{x}\in\mathcal{F} \,\middle|\, \boldsymbol{x} + \begin{pmatrix} c \\ 0 \end{pmatrix} \Delta t \in \mathcal{B} \right\} \tag{6.6}$$

along a vertical line at the boundary. More precisely, the line runs through the column of the last ordinary grid points $\mathcal{F}_b$, so that errors on the boundary are irrelevant. We compute 600 iterations after the BC is changed, thus it holds $\mathcal{T} = \{t_0, \ldots, t_{850}\}$. The measured errors of simulations with different BCs and different angles of incidence are visualized in Fig. 6.15. The errors for `LODI` and `ImpBC` are monotonically increasing with the angle of incidence. This behavior coincides with results of Schlaffer [93]. The error for `CBC` shows an atypical decrease for angles of incidence $\phi > 70°$. Even more remarkable are the errors of `DABC`. For `DABC` we illustrate the error $\rho(\boldsymbol{x}, t_{850}) - \rho^{\mathrm{ref}}(\boldsymbol{x}, t_{850})$ for simulations $\phi = 40°$ and $\phi = 60°$ in Fig. 6.16. Although the maximal deviation is comparable in an absolute sense, the direction of deviation is different. A positive deviation can be seen for angles of incidence $\phi < 50°$, while a negative is seen for $\phi > 50°$, this explains the behavior of the maximal error in Fig. 6.15. Analogue to Fig. 6.15, we plotted the errors E_{u_1} and E_{u_2} in Fig. E.12, see Appendix E. We did not consider `isoImpBC` in this example,

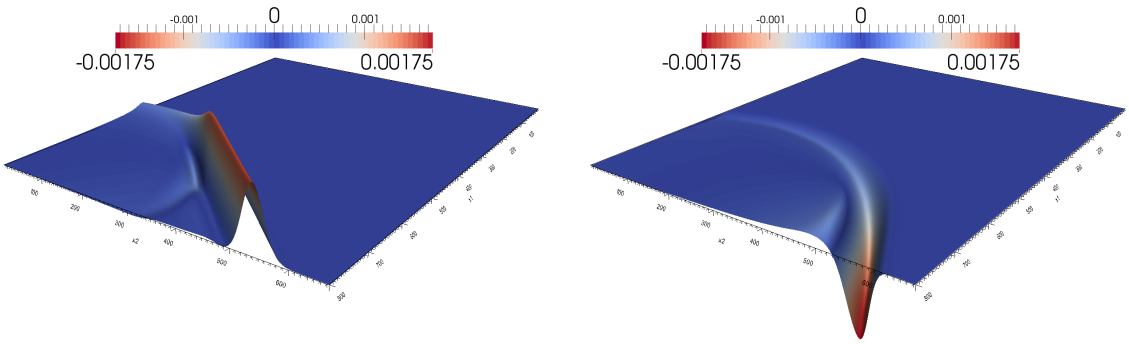


Figure 6.16: *Errors created by DABCs for two angles of incidence* – The left surface with a positive deviation shows the error in a simulation with `DABC`, when the plane wave has an angle of incidence $\phi = 40°$. The surface on the right hand side corresponds to a simulation with angle of incidence $\phi = 60°$, the deviation is mainly negative. The x_3-axis (not shown) has same scaling for both visualizations.

due to an oscillating behavior of this BC, see also Fig. E.11 (appendix). The reason for the presence of this oscillation is unknown, but it might be an effect of the switch from `ITBC` to `isoImpBC` after 250 iterations.

6.7 3D square duct flow past an obstacle

The final test case consists of a three-dimensional simulation of a duct flow past an obstacle at $\mathrm{Re} \cong 150$. The duct has a square cross-section of 90×90 inner grid points, whereas its length in flow direction is 200 grid points. The obstacle is a thin asymmetric plate located in the duct at one quarter of its total length, an illustration is given in Fig. 6.17, where one side wall was removed to reveal the obstacle. There is no slit between the obstacle and the side walls. In an earlier simulation we first computed the stationary solution for the flow with a given maximal velocity, but without having an obstacle in the duct. This stationary solution is used to initialized the duct flow past an obstacle. At the inlet we impose the parabolic-like velocity profile of the stationary solution, and at the outlet we test several artificial BCs. In total we compute 8000 time levels, in Fig. 6.18 we see the velocity at final time using `ITBC` for the outlet. We can see that the velocity profile at the outlet is far from showing a parabolic-like behavior, where the velocity would be maximal in the center. However, the subproblems in the DABC are initialized with such a parabolic-like velocity profiles, furthermore we set the maximal history depth to $H = 30$. Hence, if we consider the `DABC` simulation and investigate the error

$$\left\| \boldsymbol{u}(\boldsymbol{x}, t_{8000}) - \boldsymbol{u}^{\mathrm{ref}}(\boldsymbol{x}, t_{8000}) \right\|_2 \tag{6.7}$$

at final time, visualized in Fig. 6.19 (left), we can see the largest error in the center at the outlet. We also show the analogue error for the simulation with `LODI` (right), here the maximal error is also located in the center at the outlet. However, the error of the DABC is smaller by almost a factor of two. Finally, we consider a time-dependent error by considering the ℓ^2-error (6.2). In Fig. 6.20 we can see this error for the velocity component

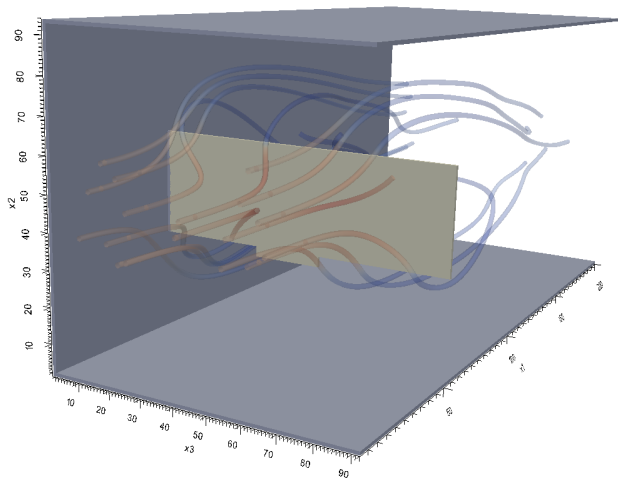


Figure 6.17: *Thin plate located in a duct* – An obstacle (thin plate, asymmetric) in a three-dimensional duct is visualized. The right side wall of the duct is not shown. Streamlines are delineated, showing the flow past the obstacle.

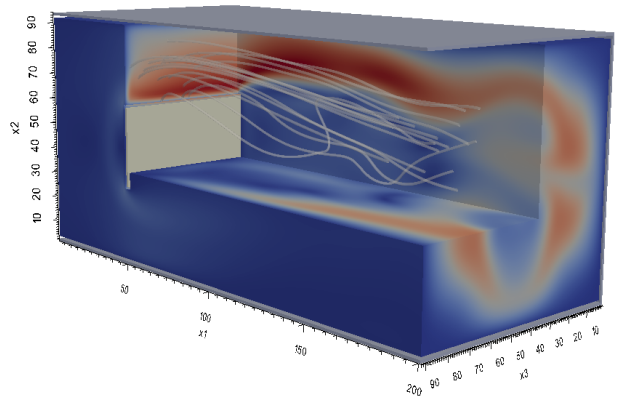


Figure 6.18: *Velocity profile of flow past an obstacle* – The velocity (magnitude) profile of a simulation past a thin asymmetric plate is shown. One side wall of the duct is not shown. At the outlet ($x_1 = 200$) an ITBC is applied.

in flow direction and for a non-orthogonal velocity component. Here we also see, that the errors in the `DABC` simulation are smaller than the ones of the `LODI` simulation.

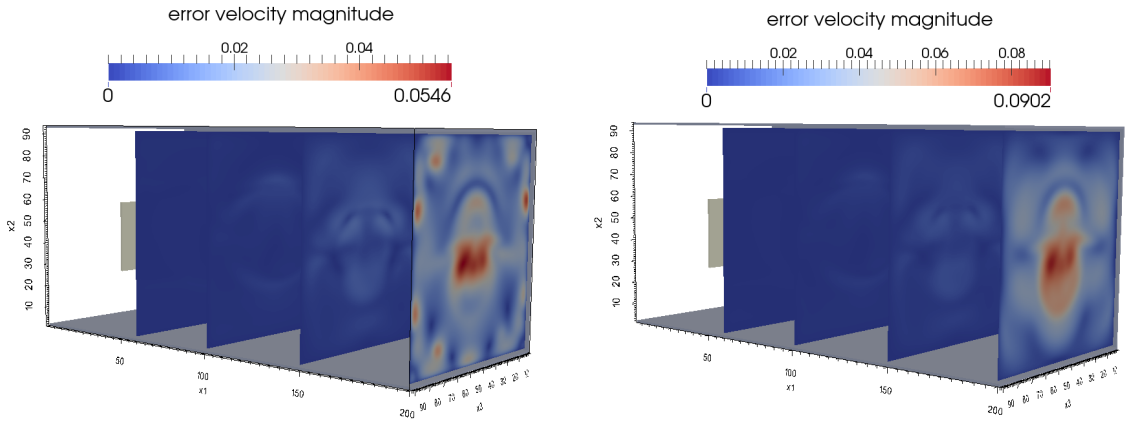


Figure 6.19: *Errors of* `DABC` *and* `LODI` *for a flow past an obstacle* – The left picture shows the error (6.7) of our `DABC`, while the right picture does for `LODI`. The four slices are located at the outlet ($x_1 = 200$), at $x_1 = 100, 150$, as well as shortly behind the obstacle at $x_1 = 60$. Note the different color ranges in the left and right picture.

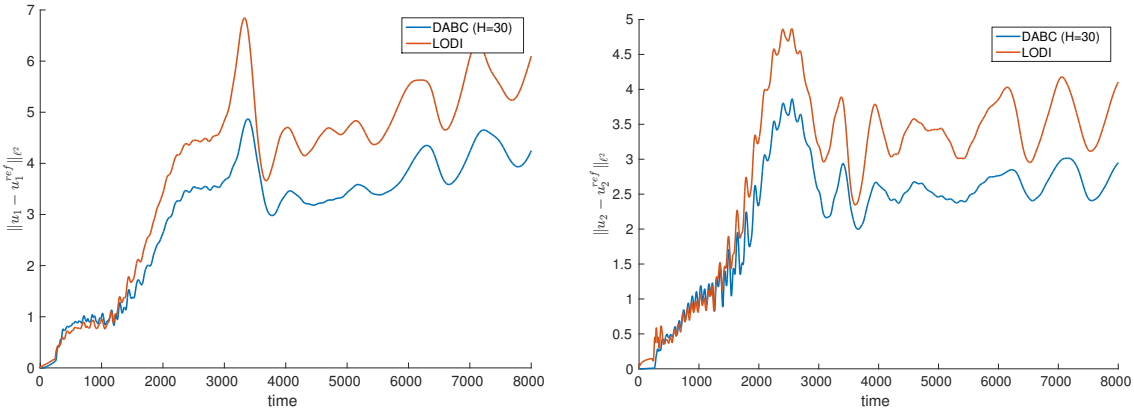


Figure 6.20: *Errors in velocities in 3D duct flow simulation* – The left plot shows the ℓ^2-error (6.2) for the velocity component in direction of the flow, while the plot on the right hand side gives the same error for a non-orthogonal velocity component.

7 Chapter 7

Conclusions and outlook

This thesis dealt with artificial boundary conditions for open boundaries in the lattice Boltzmann method. With an ideal artificial boundary condition no spurious reflections are generated at the open boundaries. Therefore, the use of an artificial boundary condition allows to confine the computational domain to the most important domain of interest. The content of this work can basically be split in three parts.

The first part (Chapter 2) of the thesis focused on establishing a theoretical basis for the development of artificial boundary conditions. We derived the three-dimensional lattice Boltzmann model D3Q19 as the analytical result of a numerical integration. Moreover, we explained in detail the influence of the collision operator to the equivalent macroscopic model.

In the second part of this thesis (Chapter 3) we began with a short overview of existing approaches, designed to minimize the reflections at open boundaries. We explained that an ideal artificial boundary condition has to depend on processes in the exterior of the computational domain. Consequently, any boundary condition which solely depends on interior information can be ideal, only if it depends on interior information up to initial time and at the same time uses a consistent assumption for the initial exterior domain. Existing approaches for open boundaries can be grouped in two categories.
The first one consists of approaches which extend the computational domain by a region in which any fluctuations are damped out, such that the boundary at the end of the extension creates only a negligible reflection. The principal representative in this category are *perfectly matched layers*, which we summarized in Section 3.2.1.
The second category, which we have addressed more extensively in this work, consists of boundary conditions in its actual sense. We have considered the recent *impedance boundary conditions* in Section 3.2.2. As a noteworthy drawback, their principle cannot be applied for edges and corners. This impedance boundary condition is featured for avoiding the necessity of computing neither spatial nor temporal derivatives, as well as assuming that any pressure changes are propagating with the speed of sound. Another common representative in the category of boundary conditions is a characteristic based boundary condition. Existing *characteristic boundary conditions* for the lattice Boltzmann method are based on the LODI equations, which are then analytically treated, such that they can be solved stably with interior information only. The LODI equations represent an approximation to the macroscopic model of the interior, hence their solution yields macroscopic values, which still have to be transferred to the lattice Boltzmann method. In this work we have extended the existing characteristic boundary conditions (Sections 3.3 and 3.4). Our characteristic boundary conditions are not based on the LODI equations, but on an extension, which does not suppress non-orthogonal derivatives. In Section 3.4.1 we also describe an idea for a further enhancement, where in contrast to LODI based and our characteristic boundary conditions also viscous terms are incorporated.
The theoretical considerations of characteristic boundary conditions are supported by results of numerical examples (Chapter 6). The results demonstrate that their way of

implementation has a non-negligible effect. Our approach to incorporate non-equilibrium portions by an extrapolation seemed to be advantageous for values of the relaxation time $\tau \leq 1$, while for $\tau > 1$ this procedure enlarged the errors. Since a more detailed study of the relation between the accuracy of characteristic boundary conditions and their implementation has not been addressed in this thesis, this issue provides as subject for further research. Also the incorporation of viscous terms could drastically reduce the errors when $\tau \approx 1$, we recognize a possible potential for this *viscous characteristic boundary condition*. They should be subject for possible future investigations.
From the perspective of the discrete lattice Boltzmann method all characteristic based boundary conditions are simply a Dirichlet condition, where the Dirichlet values are computed appropriately on the macroscopic scale. The macroscopic calculations do not obey the relation (3.4), which relates the normal velocity and the mass density at the boundary. However, the corresponding macroscopic quantities of an ideal artificial boundary condition satisfy this relation. Therefore, incorporating this relation appropriately into characteristic based boundary conditions might result in an improvement. If we supplement the characteristic system (3.20) with an algebraic equation corresponding to the relation (3.4) at final time, the solvability of the characteristic system is in generally violated. Hence an incorporation requires more advanced techniques and we see an interesting research task here.

The final part of this thesis handled discrete artificial boundary conditions (Chapters 4 and 5). Here we developed a novel concept of treating artificial boundaries in the lattice Boltzmann method. We began with deriving an exact artificial boundary condition for a one-dimensional lattice Boltzmann method with a linear collision operator. The exact artificial boundary condition was already formulated solely on the discrete level, however it was not efficient due to its computational costs. The exact discrete boundary condition assumed that the exterior domain was initialized in a known homogeneous equilibrium state. We investigated the boundary condition analytically and thereby could find an efficient approximation, which is based on a truncation of a sum in the formulation of the exact discrete boundary condition. The parameter controlling the truncation is related to the *history depth* of the later derived general discrete boundary conditions. In order to retain an equilibrium state when using the efficient approximate discrete boundary condition, we developed a consistency condition and incorporated it to the approximate discrete boundary condition. The derivation of discrete artificial boundary conditions for the more general case (not limited to the linear collision term), was based on a digraph interpretation of the discrete artificial boundary condition. We explained the use digraphs in detail. With their help we could show that at every time level the discrete artificial boundary condition is equivalent to the result of an appropriate lattice Boltzmann simulation, a so-called *subproblem*. We gave details for an efficient implementation of the subproblems, this efficiency is a feature of our novel concept. The truncation parameter of the linear case remains as a free parameter (history depth) in the general case. We theoretically elucidated that the initialization of the subproblems represents a key role, since all errors of our discrete artificial boundary condition are generated here. This theoretical result was confirmed by our numerical tests. For this task of the initialization we presented one natural choice, which in the numerical simulations turned out to lead to a competitive boundary condition. However, the initialization of subproblems represents the most obvious subject of future subsequent research tasks. Especially if we consider Fig. 6.9, we see that profiles in the subproblem's domain are shaped during the iterations of the subproblem. Using an appropriately pre-shaped profile, obtained for instance from a coarser grid might be a possible idea. Alternatively tracking the effective mass flow

across the boundary could help to dynamically adjust the level of the subproblem's initial profile. The numerical experiments also demonstrated a significant difference in the use of even and odd history depths, respectively. We have not studied the reason for this different behavior, hence it naturally yields a starting point for future research.

A Appendix A

Alternative discrete velocity models

In Section 2.3 we have derived the discrete velocities of the D3Q19 model. This is not the only possible realization of a discrete velocity space (2.34). Below we list some further models.

Contrary to D3Q19, the amount of velocities in three space dimensions can be reduced to 15. This yields the D3Q15 model, where $\mathcal{V}$ is given by the discrete velocities

$$\boldsymbol{c}_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \qquad \boldsymbol{c}_{1-6} = \left\{ \begin{pmatrix} \pm c \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \pm c \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ \pm c \end{pmatrix} \right\}, \qquad \boldsymbol{c}_{7-14} = \begin{pmatrix} \pm c \\ \pm c \\ \pm c \end{pmatrix}.$$

The D3Q27 model is also very common in three space dimensions, that is a a discretization with 27 velocities, consisting of the D3Q19 vectors plus

$$\boldsymbol{c}_{19-26} = \begin{pmatrix} \pm c \\ \pm c \\ \pm c \end{pmatrix}.$$

The three-dimensional models stated here are illustrated in Fig. A.1. The projection of the D3Q27 vectors into the two-dimensional space yields the D2Q9 model, with lattice vectors given by

$$\boldsymbol{c}_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \boldsymbol{c}_{1-4} = \left\{ \begin{pmatrix} \pm c \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \pm c \end{pmatrix} \right\}, \qquad \boldsymbol{c}_{5-8} = \begin{pmatrix} \pm c \\ \pm c \end{pmatrix}.$$

The D2Q7 model is inherited from the lattice gas automata of Frisch, Hasslacher and Pomeau (FHP model) [30]. It is based on a hexagonal lattice, with discrete velocities

$$\boldsymbol{c}_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \boldsymbol{c}_j = c \begin{pmatrix} \cos(\frac{\pi}{3} j) \\ \sin(\frac{\pi}{3} j) \end{pmatrix}, \quad j = 1, \ldots, 6.$$

In an one-dimensional space the projection of the D2Q9 model is a possible realization, that is the D1Q3 model, where the discrete velocities read

$$c_0 = 0, \qquad c_1 = -c \qquad c_2 = c. \tag{A.1}$$

All models stated here satisfy the macroscopic equations (2.14) in its d-dimensional formulation. However they do not represent a complete list of discretizations satisfying the macroscopic equations (2.14). For other models, for instance multi-speed models, we only refer to literature, e.g., [17, 34].

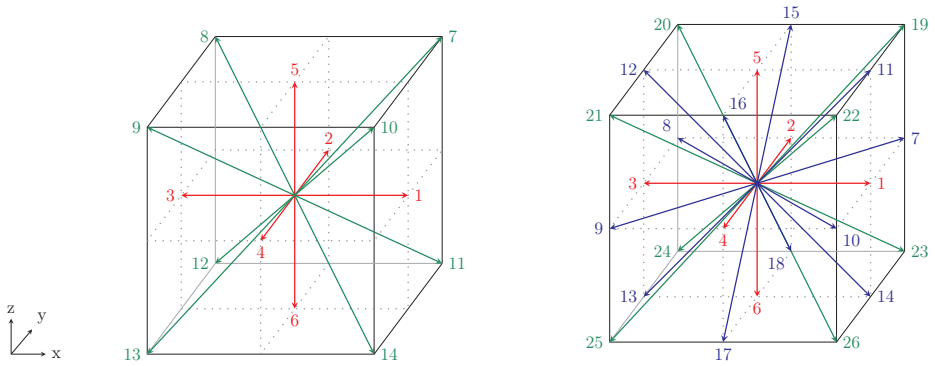


Figure A.1: *Lattice vectors of D3Q15 and D3Q27* – Illustration of the lattice vectors of the D3Q15 and D3Q27 model, respectively. Vectors shown in red are unitary, vectors shown in blue satisfy $\|\boldsymbol{c}_j\| = \sqrt{2}c$, while green vectors satisfy $\|\boldsymbol{c}_j\| = \sqrt{3}c$.

B Appendix B

Matrices for characteristic boundary conditions

B.1 Matrices of the basic system

The characteristic boundary conditions of Section 3.3 are based on the basic system (3.17). Here we explicitly give the coefficient matrices $\boldsymbol{A}_{d,x_\alpha}$ and a possible choice for the product (3.18).

In 1D the coefficient matrix reads

$$\boldsymbol{A}_{1,x_1} = \boldsymbol{A}_{1,x_1}(\rho, u_1) = \begin{pmatrix} u_1 & \rho \\ \frac{c_s^2}{\rho} & u_1 \end{pmatrix},$$

and the product (3.18) can be realized with

$$\boldsymbol{R}_{1,x_1}^{-1} = \begin{pmatrix} \frac{1}{2c_s^2} & \frac{1}{2c_s^2} \\ -\frac{1}{2c_s\rho} & \frac{1}{2c_s\rho} \end{pmatrix}, \qquad \boldsymbol{R}_{1,x_1} = \begin{pmatrix} c_s^2 & -c_s\rho \\ c_s^2 & c_s\rho \end{pmatrix}.$$

The matrices for two-dimensional problems are given already in Section 3.3. In three-dimensions the coefficient matrices $\boldsymbol{A}_{d,x_\alpha} = \boldsymbol{A}_{3,x_\alpha}(\rho, u_1, u_2, u_3)$ read

$$\boldsymbol{A}_{3,x_1} = \begin{pmatrix} u_1 & \rho & 0 & 0 \\ \frac{c_s^2}{\rho} & u_1 & 0 & 0 \\ 0 & 0 & u_1 & 0 \\ 0 & 0 & 0 & u_1 \end{pmatrix},$$

$$\boldsymbol{A}_{3,x_2} = \begin{pmatrix} u_2 & 0 & \rho & 0 \\ 0 & u_2 & 0 & 0 \\ \frac{c_s^2}{\rho} & 0 & u_2 & 0 \\ 0 & 0 & 0 & u_2 \end{pmatrix}, \qquad \boldsymbol{A}_{3,x_3} = \begin{pmatrix} u_3 & 0 & 0 & \rho \\ 0 & u_3 & 0 & 0 \\ 0 & 0 & u_3 & 0 \\ \frac{c_s^2}{\rho} & 0 & 0 & u_3 \end{pmatrix}.$$

The eigenvectors of $\boldsymbol{A}_{d,x_\alpha}$ are used to construct the matrices $\boldsymbol{R}_{d,x_\alpha}^{-1}$ for (3.18). A possible choice, for $d = 3$, is given as follows:

$$\boldsymbol{R}_{3,x_1}^{-1} = \begin{pmatrix} \frac{1}{2c_s^2} & \frac{1}{2c_s^2} & 0 & 0 \\ -\frac{1}{2c_s\rho} & \frac{1}{2c_s\rho} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \qquad \boldsymbol{R}_{3,x_1} = \begin{pmatrix} c_s^2 & -c_s\rho & 0 & 0 \\ c_s^2 & c_s\rho & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\boldsymbol{R}_{3,x_2}^{-1} = \begin{pmatrix} \frac{1}{2c_s^2} & \frac{1}{2c_s^2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{1}{2c_s\rho} & \frac{1}{2c_s\rho} & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \qquad \boldsymbol{R}_{3,x_2} = \begin{pmatrix} c_s^2 & 0 & -c_s\rho & 0 \\ c_s^2 & 0 & c_s\rho & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\boldsymbol{R}_{3,x_3}^{-1} = \begin{pmatrix} \frac{1}{2c_s^2} & \frac{1}{2c_s^2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{1}{2c_s\rho} & \frac{1}{2c_s\rho} & 0 & 0 \end{pmatrix}, \qquad \boldsymbol{R}_{3,x_3} = \begin{pmatrix} c_s^2 & 0 & 0 & -c_s\rho \\ c_s^2 & 0 & 0 & c_s\rho \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

B.2 Coefficient matrices of the characteristic system

Moreover we consider the perfectly non-reflecting CBC (3.20) and give results for the coefficient matrices $\widetilde{\boldsymbol{A}}_{d,x_\alpha}$. Let $d = 1$, we have to distinguish two cases. First $\widetilde{\boldsymbol{\Lambda}}_{1,x_1}$ contains only non-negative entries, then we get $\widetilde{\boldsymbol{A}}^{+}_{1,x_1}$. Second, if in $\widetilde{\boldsymbol{\Lambda}}_{1,x_1}$ all positive eigenvalues are set to zero, then we obtain $\widetilde{\boldsymbol{A}}^{-}_{1,x_1}$:

$$\widetilde{\boldsymbol{A}}^{+}_{1,x_1} := \begin{pmatrix} \frac{c_s+u_1}{2} & \frac{\rho(c_s+u_1)}{2c_s} \\ \frac{c_s(c_s+u_1)}{2\rho} & \frac{c_s+u_1}{2} \end{pmatrix}, \qquad \widetilde{\boldsymbol{A}}^{-}_{1,x_1} := \begin{pmatrix} \frac{-c_s+u_1}{2} & \frac{\rho(c_s-u_1)}{2c_s} \\ \frac{c_s(c_s-u_1)}{2\rho} & \frac{-c_s+u_1}{2} \end{pmatrix}.$$

Next, we consider $d = 2$, where four cases have to be considered:

i) $u_{x_\alpha} > 0$ and all positive eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{2,x_\alpha}$. Then the modified coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{2,x_1} := \begin{pmatrix} \frac{u_1-c_s}{2} & \frac{(c_s-u_1)\rho}{2c_s} & 0 \\ \frac{c_s(c_s-u_1)}{2\rho} & \frac{u_1-c_s}{2} & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \widetilde{\boldsymbol{A}}_{2,x_2} := \begin{pmatrix} \frac{u_2-c_s}{2} & 0 & \frac{(c_s-u_2)\rho}{2c_s} \\ 0 & 0 & 0 \\ \frac{c_s(c_s-u_2)}{2\rho} & 0 & \frac{u_2-c_s}{2} \end{pmatrix}.$$

ii) $u_{x_\alpha} < 0$ and all positive eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{2,x_\alpha}$. Then the modified coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{2,x_1} := \begin{pmatrix} \frac{u_1-c_s}{2} & \frac{(c_s-u_1)\rho}{2c_s} & 0 \\ \frac{c_s(c_s-u_1)}{2\rho} & \frac{u_1-c_s}{2} & 0 \\ 0 & 0 & u_1 \end{pmatrix}, \qquad \widetilde{\boldsymbol{A}}_{2,x_2} := \begin{pmatrix} \frac{u_2-c_s}{2} & 0 & \frac{(c_s-u_2)\rho}{2c_s} \\ 0 & u_2 & 0 \\ \frac{c_s(c_s-u_2)}{2\rho} & 0 & \frac{u_2-c_s}{2} \end{pmatrix}.$$

iii) $u_{x_\alpha} > 0$ and all negative eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{2,x_\alpha}$. Then the modified

coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{2,x_1} := \begin{pmatrix} \frac{c_s+u_1}{2} & \frac{(c_s+u_1)\rho}{2c_s} & 0 \\ \frac{c_s(c_s+u_1)}{2\rho} & \frac{c_s+u_1}{2} & 0 \\ 0 & 0 & u_1 \end{pmatrix}, \quad \widetilde{\boldsymbol{A}}_{2,x_2} := \begin{pmatrix} \frac{c_s+u_2}{2} & 0 & \frac{(c_s+u_2)\rho}{2c_s} \\ 0 & u_2 & 0 \\ \frac{c_s(c_s+u_2)}{2\rho} & 0 & \frac{c_s+u_2}{2} \end{pmatrix}.$$

iv) $u_{x_\alpha} < 0$ and all negative eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{2,x_\alpha}$. Then the modified coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{2,x_1} := \begin{pmatrix} \frac{c_s+u_1}{2} & \frac{(c_s+u_1)\rho}{2c_s} & 0 \\ \frac{c_s(c_s+u_1)}{2\rho} & \frac{c_s+u_1}{2} & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \widetilde{\boldsymbol{A}}_{2,x_2} := \begin{pmatrix} \frac{c_s+u_2}{2} & 0 & \frac{(c_s+u_2)\rho}{2c_s} \\ 0 & 0 & 0 \\ \frac{c_s(c_s+u_2)}{2\rho} & 0 & \frac{c_s+u_2}{2} \end{pmatrix}.$$

Finally, let $d = 3$, the same four cases as above are considered

i) $u_{x_\alpha} > 0$ and all positive eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{3,x_\alpha}$. Then the modified coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{3,x_1} := \begin{pmatrix} \frac{u_1-c_s}{2} & \frac{(c_s-u_1)\rho}{2c_s} & 0 & 0 \\ \frac{c(c_s-u_1)}{2\rho} & \frac{u_1-c_s}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\widetilde{\boldsymbol{A}}_{3,x_2} := \begin{pmatrix} \frac{u_2-c_s}{2} & 0 & \frac{(c_s-u_2)\rho}{2c_s} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{c(c_s-u_2)}{2\rho} & 0 & \frac{u_2-c_s}{2} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \widetilde{\boldsymbol{A}}_{3,x_3} := \begin{pmatrix} \frac{u_3-c_s}{2} & 0 & 0 & \frac{(c_s-u_3)\rho}{2c_s} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{c(c_s-u_3)}{2\rho} & 0 & 0 & \frac{u_3-c_s}{2} \end{pmatrix}.$$

ii) $u_{x_\alpha} < 0$ and all positive eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{3,x_\alpha}$. Then the modified coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{3,x_1} := \begin{pmatrix} \frac{u_1-c_s}{2} & \frac{(c_s-u_1)\rho}{2c_s} & 0 & 0 \\ \frac{c(c_s-u_1)}{2\rho} & \frac{u_1-c_s}{2} & 0 & 0 \\ 0 & 0 & u_1 & 0 \\ 0 & 0 & 0 & u_1 \end{pmatrix},$$

$$\widetilde{\boldsymbol{A}}_{3,x_2} := \begin{pmatrix} \frac{u_2-c_s}{2} & 0 & \frac{(c_s-u_2)\rho}{2c_s} & 0 \\ 0 & u_2 & 0 & 0 \\ \frac{c(c_s-u_2)}{2\rho} & 0 & \frac{u_2-c_s}{2} & 0 \\ 0 & 0 & 0 & u_2 \end{pmatrix}, \quad \widetilde{\boldsymbol{A}}_{3,x_3} := \begin{pmatrix} \frac{u_3-c_s}{2} & 0 & 0 & \frac{(c_s-u_3)\rho}{2c_s} \\ 0 & u_3 & 0 & 0 \\ 0 & 0 & u_3 & 0 \\ \frac{c(c_s-u_3)}{2\rho} & 0 & 0 & \frac{u_3-c_s}{2} \end{pmatrix}.$$

iii) $u_{x_\alpha} > 0$ and all negative eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{3,x_\alpha}$. Then the modified

coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{3,x_1} := \begin{pmatrix} \frac{c_s+u_1}{2} & \frac{(c_s+u_1)\rho}{2c_s} & 0 & 0 \\ \frac{c(c_s+u_1)}{2\rho} & \frac{c_s+u_1}{2} & 0 & 0 \\ 0 & 0 & u_1 & 0 \\ 0 & 0 & 0 & u_1 \end{pmatrix},$$

$$\widetilde{\boldsymbol{A}}_{3,x_2} := \begin{pmatrix} \frac{c_s+u_2}{2} & 0 & \frac{(c_s+u_2)\rho}{2c_s} & 0 \\ 0 & u_2 & 0 & 0 \\ \frac{c(c_s+u_2)}{2\rho} & 0 & \frac{c_s+u_2}{2} & 0 \\ 0 & 0 & 0 & u_2 \end{pmatrix}, \quad \widetilde{\boldsymbol{A}}_{3,x_3} := \begin{pmatrix} \frac{c_s+u_3}{2} & 0 & 0 & \frac{(c_s+u_3)\rho}{2c_s} \\ 0 & u_3 & 0 & 0 \\ 0 & 0 & u_3 & 0 \\ \frac{c(c_s+u_3)}{2\rho} & 0 & 0 & \frac{c_s+u_3}{2} \end{pmatrix}.$$

iv) $u_{x_\alpha} < 0$ and all negative eigenvalues are set to zero in $\widetilde{\boldsymbol{\Lambda}}_{3,x_\alpha}$. Then the modified coefficient matrices in (3.20) read:

$$\widetilde{\boldsymbol{A}}_{3,x_1} := \begin{pmatrix} \frac{c_s+u_1}{2} & \frac{(c_s+u_1)\rho}{2c_s} & 0 & 0 \\ \frac{c(c_s+u_1)}{2\rho} & \frac{c_s+u_1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\widetilde{\boldsymbol{A}}_{3,x_2} := \begin{pmatrix} \frac{c_s+u_2}{2} & 0 & \frac{(c_s+u_2)\rho}{2c_s} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{c(c_s+u_2)}{2\rho} & 0 & \frac{c_s+u_2}{2} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \widetilde{\boldsymbol{A}}_{3,x_3} := \begin{pmatrix} \frac{c_s+u_3}{2} & 0 & 0 & \frac{(c_s+u_3)\rho}{2c_s} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{c(c_s+u_3)}{2\rho} & 0 & 0 & \frac{c_s+u_3}{2} \end{pmatrix}.$$

C Appendix C

Completion of the proof of Lemma 4

The proof of Lemma 4 as given in the main document is incomplete. For part d) we proved the validity of the recursion (4.21) and it remains to show that (4.20) satisfies this recursion. This last step to complete the proof is done next.

Due to part b) of Lemma 4 we can restrict the calculation to even $k = 2j$. Then (4.20) is equivalent to

$$P^F(2j, m, v) = \frac{j-m+1}{j} \cdot \frac{j!}{(j-v)!\,v!} \cdot \frac{(m-2)!}{(m-v-1)!\,(v-1)!}. \tag{C.1}$$

We consider $m = 1, \ldots, j-1$ and investigate (4.21a). The first summand $P^F(2j-2, m, v)$ on its right hand side satisfies

$$\begin{aligned} P^F(2j-2, m, v) &= \left(1 - \frac{m-1}{j-1}\right) \binom{j-1}{v} \binom{m-2}{v-1} \\ &= \frac{j-m}{j-1} \frac{(j-1)!\,(m-2)!}{(j-v-1)!\,v!\,(m-v-1)!\,(v-1)!} \\ &= \frac{(m-2)!\,(j-2)!\,j}{(j-v)!\,v!\,(m-v-1)!\,(v-1)!\,j} \cdot (j-m)\,(j-v) \end{aligned} \tag{C.2}$$

The sum on the right hand side of (4.21a) satisfies

$$\begin{aligned} \sum_{p=1}^{m-1} P^F(2j-2, p, v-1) &= \sum_{p=1}^{m-1} \left(1 - \frac{p-1}{j-1}\right) \binom{j-1}{v-1} \binom{p-2}{v-2} \\ &= \sum_{p=v}^{m-1} \frac{j-p}{j-1} \cdot \frac{(j-1)!\,(p-2)!}{(j-v)!\,(v-1)!\,(p-v)!\,(v-2)!} \\ &= \frac{(j-1)!}{(j-1)\,(j-v)!\,(v-1)!\,(v-2)!} \cdot \sum_{p=v}^{m-1} \frac{(j-p)\,(p-2)!}{(p-v)!} \\ &= \frac{(j-1)!}{(j-1)\,(j-v)!\,(v-1)!\,(v-2)!} \cdot \frac{(m-1+jv-mv)\,(m-2)!}{(v-1)\,v\,(m-v-1)!} \\ &= \frac{(m-2)!\,(j-2)!\,j}{(j-v)!\,v!\,(m-v-1)!\,(v-1)!\,j} \cdot (m-1+jv-mv). \end{aligned} \tag{C.3}$$

The complete right hand side of (4.21a), i.e., the sum of (C.2) and (C.3) is

$$\begin{aligned}
&P^F(2j-2,m,v)+\sum_{p=1}^{m-1}P^F(2j-2,p,v-1)\\
&\quad=\frac{(m-2)!\,(j-2)!\,j}{(j-v)!\,v!\,(m-v-1)!\,(v-1)!\,j}\cdot\Big[(j-m)(j-v)+(m-1+jv-mv)\Big]\\
&\quad=\frac{(m-2)!\,(j-2)!\,j}{(j-v)!\,v!\,(m-v-1)!\,(v-1)!\,j}\cdot(j-1)(j-m+1)\\
&\quad=\frac{(m-2)!\,j!\,(j-m+1)}{(j-v)!\,v!\,(m-v-1)!\,(v-1)!\,j},
\end{aligned}$$

which is exactly (C.1).

Now, let $m=j$, we obtain for (4.20)

$$P^F(2j,j,v)=\frac{1}{j}\cdot\frac{j!}{(j-v)!\,v!}\cdot\frac{(j-2)!}{(j-v-1)!\,(v-1)!}=\frac{(j-1)!\,(j-2)!}{(j-v)!\,v!\,(j-v-1)!\,(v-1)!}.$$

We obtain the same result when we investigate (4.21b):

$$\begin{aligned}
\sum_{p=1}^{j-1}P^F(2j-2,p,v-1)&=\sum_{p=v}^{j-1}\left(1-\frac{p-1}{j-1}\right)\binom{j-1}{v-1}\binom{p-2}{v-2}\\
&=\frac{(j-1)!}{(j-1)\,(j-v)!\,(v-1)!\;(v-2)!}\cdot\sum_{p=v}^{j-1}\frac{(j-p)(p-2)!}{(p-v)!}\\
&=\frac{(j-1)!}{(j-1)\,(j-v)!\,(v-1)!\;(v-2)!}\cdot\frac{(j-1)\,(j-v)\,(j-2)!}{(v-1)\,v\,(j-v)!}\\
&=\frac{(j-1)!\,(j-2)!}{(j-v)!\,v!\,(j-v-1)!\,(v-1)!}.
\end{aligned}$$

This completes the proof of Lemma 4.

D Appendix D

Optimal selection for expanding the domain of convergence

The proof of Lemma 9 is split into two parts. The first part uses the result of Lemma 8 to prove the statement for the domain D_1, see (5.6). In the second part the remaining domain $F \setminus D_1$ is considered with another approach. For $F \setminus D_1$ the Lemma 8 cannot be used directly since the non-negativity assumption is violated. However one can consider absolute values, but as we show next the result of Lemma 8 does not have the potential to prove the convergence for the complete remaining domain $F \setminus D_1$.

Before we consider (5.4) as in Lemma 9 we investigate the unsplit auxiliary formulation

$$\hat{\alpha}(m) = \sum_{v=1}^{m-1} \frac{1}{m-v} \binom{m-1}{v} \binom{m-2}{v-1} A^{m-v}\, B^v. \tag{D.1}$$

With a factorization $A = a_1 a_2$ and $B = b_1 b_2$ it follows

$$|\hat{\alpha}(m)| \leq \sum_{v=1}^{m-1} \frac{1}{m-v} \binom{m-1}{v} \binom{m-2}{v-1} |a_1|^{m-v} |a_2|^{m-v} |b_1|^v |b_2|^v ,$$

and Lemma 8 would give

$$|\hat{\alpha}(m)| \leq \frac{\left[(|a_1| + |b_1|)(|a_2| + |b_2|)\right]^m}{m}, \tag{D.2}$$

with $p_j = |a_j|$ and $q_j = |b_j|$. In other words, the statement of Lemma 9 can be proven when the factorization is taken, such that $(|a_1| + |b_1|)(|a_2| + |b_2|) < 1$ holds. That means we have to solve

$$\begin{cases} \text{minimize} & J(a_1, a_2, b_1, b_2), \\ \text{such that} & a_1 a_2 = A, \\ & b_1 b_2 = B, \end{cases} \tag{D.3}$$

with $J(a_1, a_2, b_1, b_2) = (|a_1| + |b_1|)(|a_2| + |b_2|)$ and given A and B. Due to

$$J(a_1, a_2, b_1, b_2) = |A| + |B| + |a_1 b_2| + |a_2 b_1|$$

the problem (D.3) is equivalent to

$$\begin{cases} \text{minimize} & |a_1 b_2| + |a_2 b_1| , \\ \text{such that} & a_1 a_2 = A, \\ & b_1 b_2 = B, \end{cases} \tag{D.4}$$

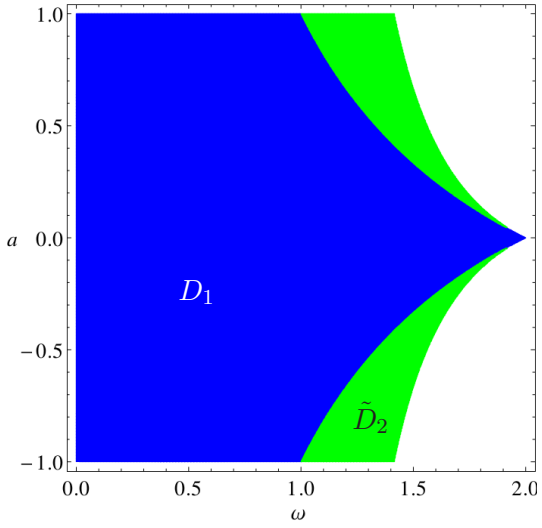


Figure D.1: *Visualization of two subdomains* – The rectangle describes the full parameter domain F. The blue subdomain visualizes the domain D_1 (5.6), on which (5.8) is proven in Lemma 9 with application of (5.7). Furthermore, in green the subdomain $\tilde{D}_2$ is shown.

We can substitute $a_2 = \frac{A}{a_1}$ and $b_1 = \frac{B}{b_2}$ and thus have to minimize $|a_1 b_2| + \frac{|AB|}{|a_1 b_2|}$. Obviously this is equivalent to

$$\min_{x \geq 0} \left(x + \frac{|AB|}{x} \right),$$

which yields $x = \sqrt{|AB|}$. The solutions of the problem (D.3) therefore satisfy

$$|a_1 b_2| = \sqrt{|AB|}, \quad a_1 a_2 = A, \quad b_1 b_2 = B.$$

Hence one solution is given by

$$a_1 = \operatorname{sgn}(A)\sqrt{|A|}, \quad a_2 = \sqrt{|A|}, \quad b_1 = \operatorname{sgn}(B)\sqrt{|B|}, \quad b_2 = \sqrt{|B|}. \tag{D.5}$$

Equation (D.1) is related to (5.4) by $A = (1+\alpha)(1+\delta)$ and $B = \beta\gamma$. From Lemma 7 it follows $\operatorname{sgn}(A) = -1$ and $\operatorname{sgn}(B) = 1$ for $F \setminus D_1$. We consider the numerator of (D.2) with the factorization (D.5):

$$\begin{aligned}
\left[(|a_1| + |b_1|)(|a_2| + |b_2|)\right]^m &= \left[\, |A| + |B| + 2\sqrt{|AB|}\right]^m \\
&= \left[\beta\gamma - (1+\alpha)(1+\delta) + 2\sqrt{-\beta\gamma(1+\alpha)(1+\delta)}\, \right]^m \\
&= \left[(\omega - 1) + 2\sqrt{\frac{\omega^3 - \omega^2}{4}(1-a^2) - \frac{\omega^4}{16}(1-a^2)^2}\, \right]^m
\end{aligned}$$

The green domain shown in Fig. 5.1 is given by

$$D_2 := \left\{ (\omega, a) \in F \setminus D_1 \;\middle|\; \left[(\omega - 1) + 2\sqrt{\frac{\omega^3 - \omega^2}{4}(1-a^2) - \frac{\omega^4}{16}(1-a^2)^2} \right] \leq 1 \right\}.$$

On D_2 the statement of Lemma 9 is proven with the above consideration. Alternatively, in Fig. D.1 we show the domain of convergence $\tilde{D}_2$ obtained with the naive factorization

$$a_1 = 1 + \alpha, \qquad a_2 = 1 + \delta, \qquad b_1 = \beta, \qquad b_2 = \gamma.$$

Noteworthy, the slightly different factorization

$$a_1 = 1 + \alpha, \qquad a_2 = 1 + \delta, \qquad b_1 = \gamma, \qquad b_2 = \beta,$$

would not even yield an expansion of the domain of convergence at all.

Appendix E

Additional numerical results

This part of the appendix assembles additional results of the numerical experiments, which are not shown in the main part of this thesis.

In addition to Fig. 6.12, which visualizes the transverse velocity when using a DABC in the simulation of an isolated vorticity wave, we present results for other BCs in Figs. E.1 to E.5. These contour plots show the transverse velocity for `ITBC`, `CBC`, `LODI`, `isoImpBC` and `PML`, respectively. Compared to the ideal evolution of Fig. E.1, all other BCs show an unphysical behavior. While for the `CBC`, `LODI`, and `PML` the contour plot of the last time level shows at most two lines, the situation is different for `isoImpBC`. Here we have a profile remaining in the given range near the boundary. Furthermore, Figs. E.6 to E.10 show the vorticity (6.5) for the same BCs. Considering the last time level, `isoImpBC` again behaves differently compared to the other BCs.

In the simulation of plane waves (Section 6.6) we did not show results of an isotropic impedance boundary condition. The reason is, that this boundary generated oscillations, as one can see in Fig. E.11. For the same test example we here show the errors, E_{u_1} and E_{u_2}, see (6.6), in Fig. E.12.

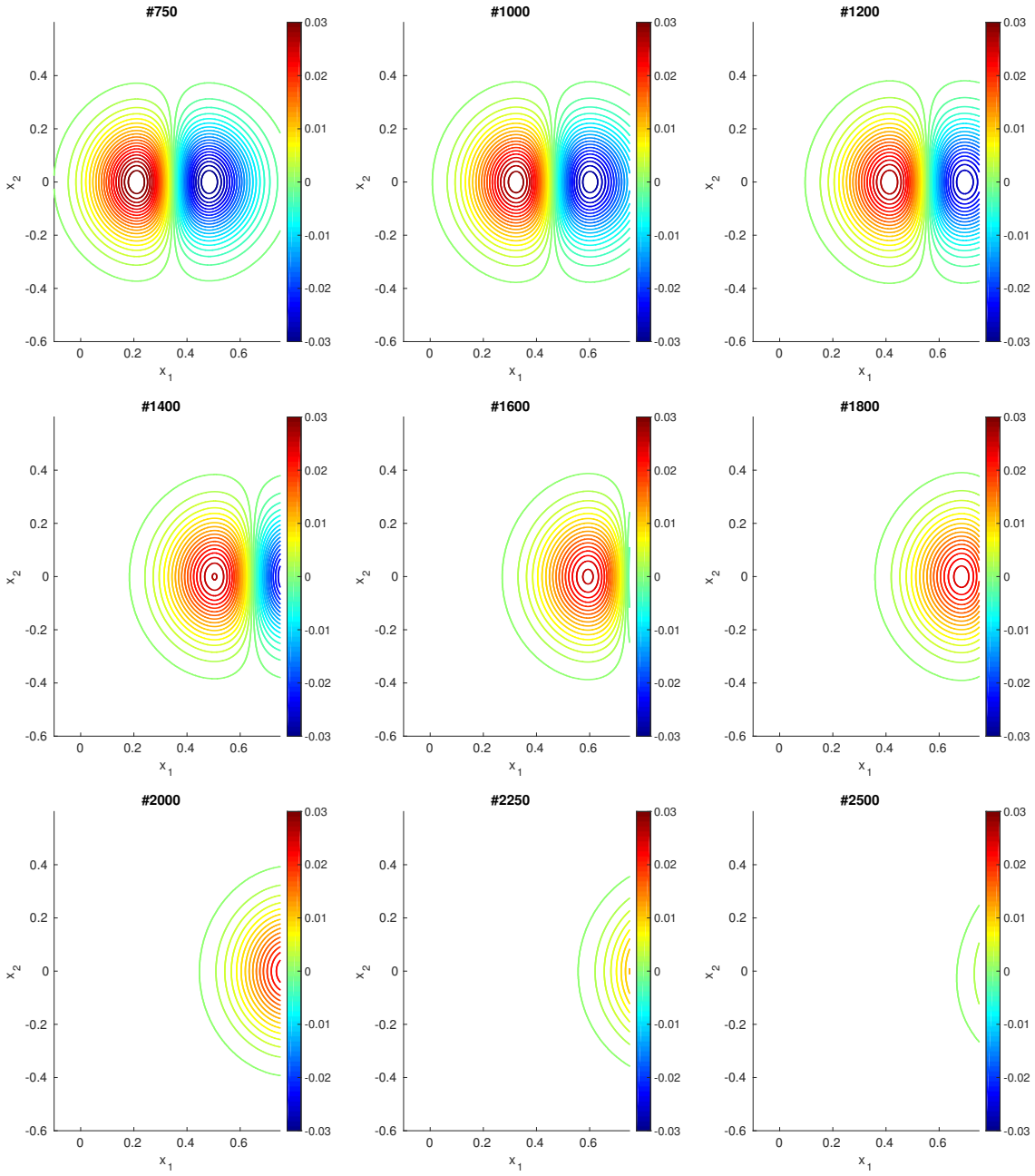


Figure E.1: *Iso-transverse-velocity contours for an isolated vortex using* `ITBC` – The plots show the evolution of contour lines of the non-orthogonal velocity component u_2 in the simulation of an isolated vorticity wave using an ITBC at the right boundary.

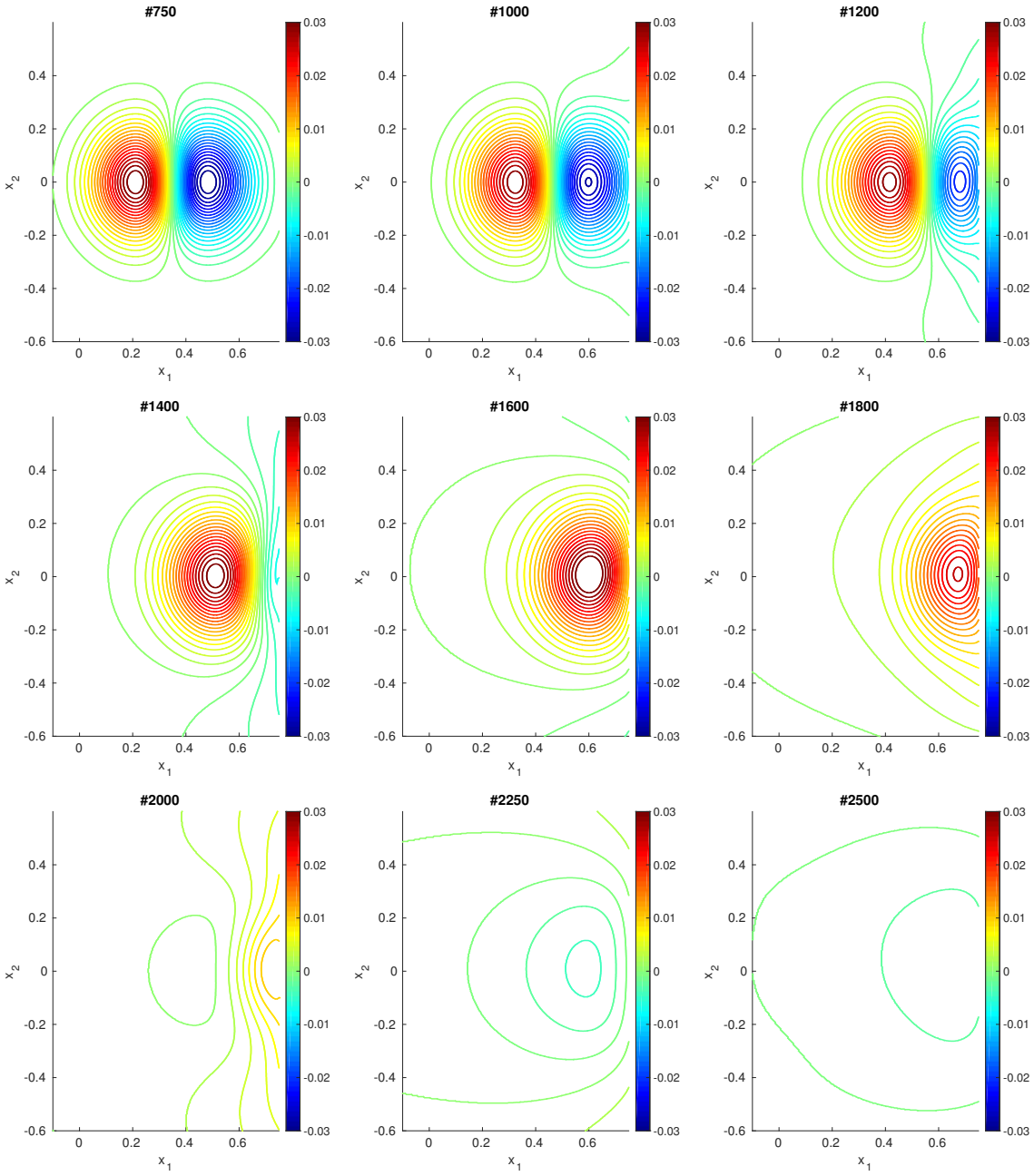


Figure E.2: *Iso-transverse-velocity contours for an isolated vortex using* `CBC` – The plots show the evolution of contour lines of the non-orthogonal velocity component u_2 in the simulation of an isolated vorticity wave using our CBC.

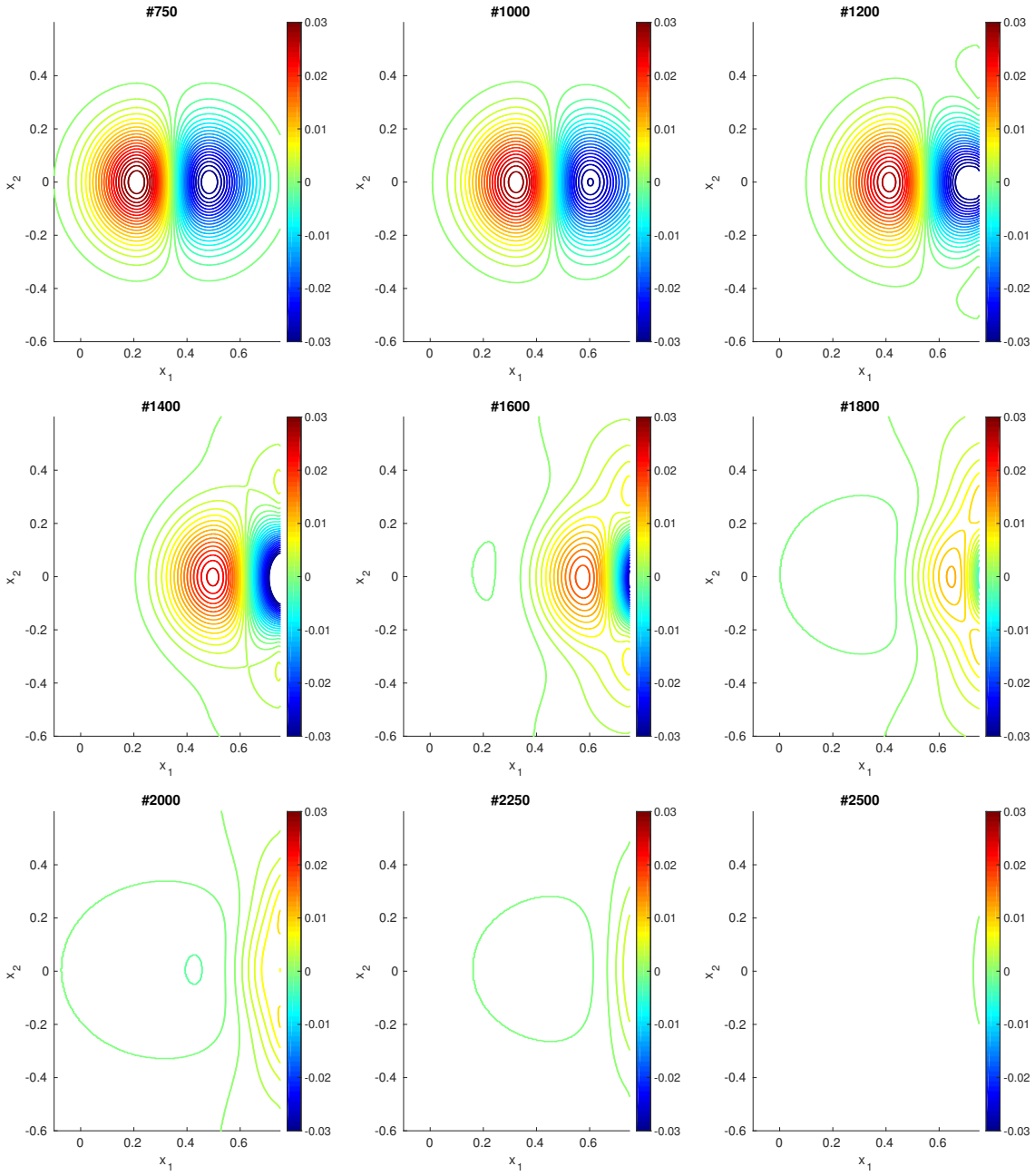


Figure E.3: *Iso-transverse-velocity contours for an isolated vortex using* `LODI` – The plots show the evolution of contour lines of the non-orthogonal velocity component u_2 in the simulation of an isolated vorticity wave using `LODI`.

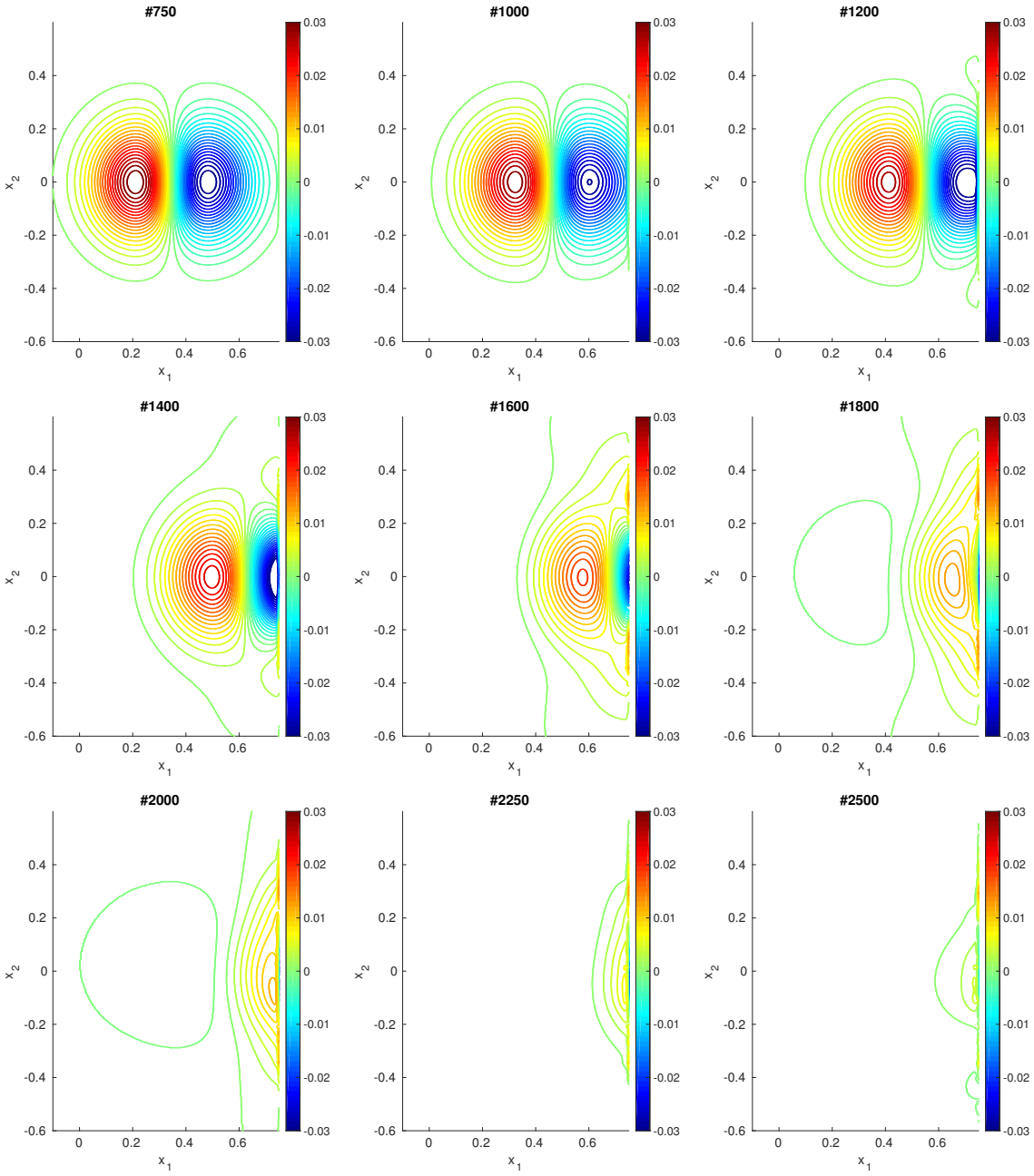


Figure E.4: *Iso-transverse-velocity contours for an isolated vortex using* `isoImpBC` – The plots show the evolution of contour lines of the non-orthogonal velocity component u_2 in the simulation of an isolated vorticity wave using the isotropic impedance boundary condition.

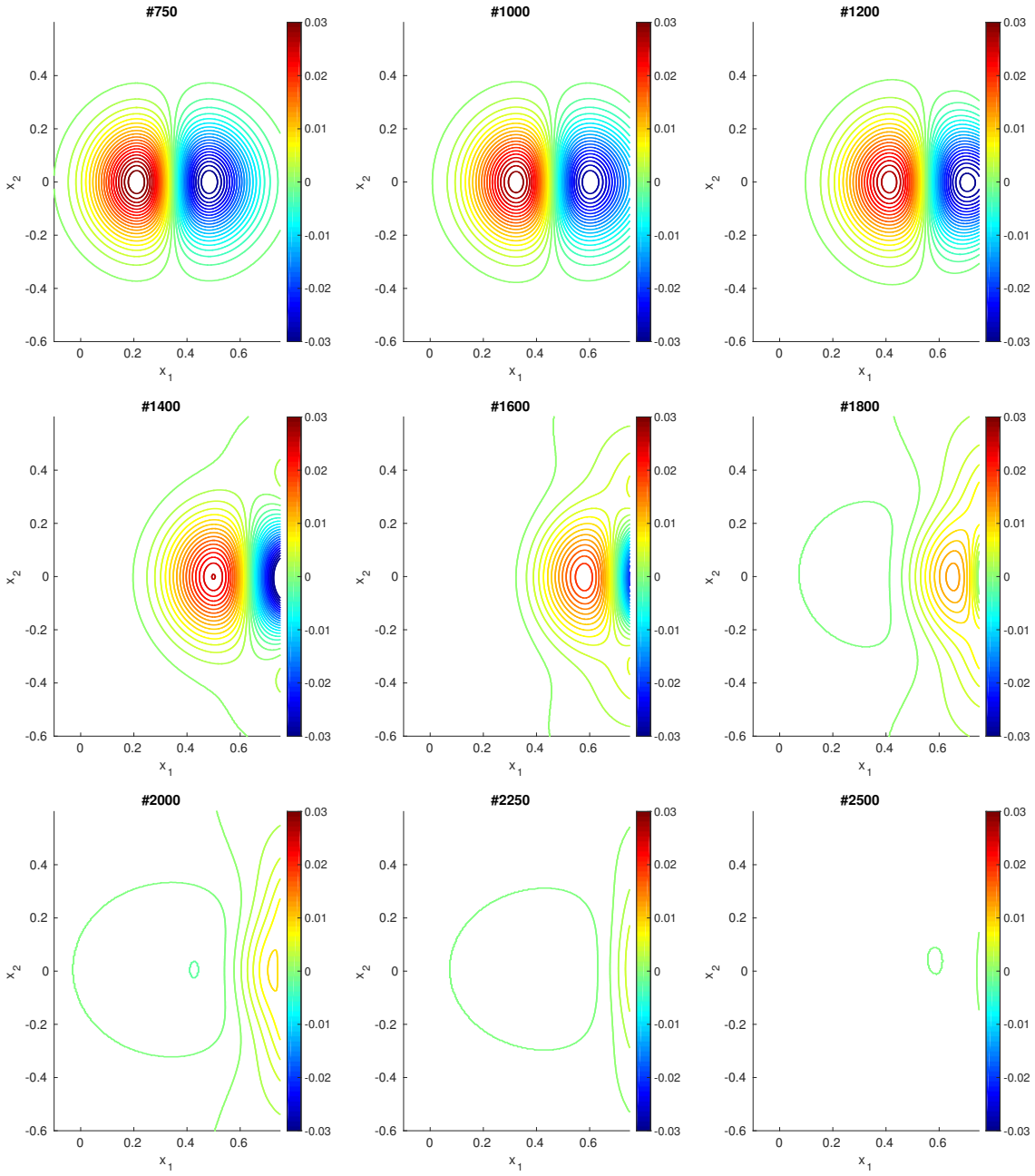


Figure E.5: *Iso-transverse-velocity contours for an isolated vortex using* `PML` – The plots show the evolution of contour lines of the non-orthogonal velocity component u_2 in the simulation of an isolated vorticity wave using a PML damping zone in the formulation of Najafi-Yazdi and Mongeau.

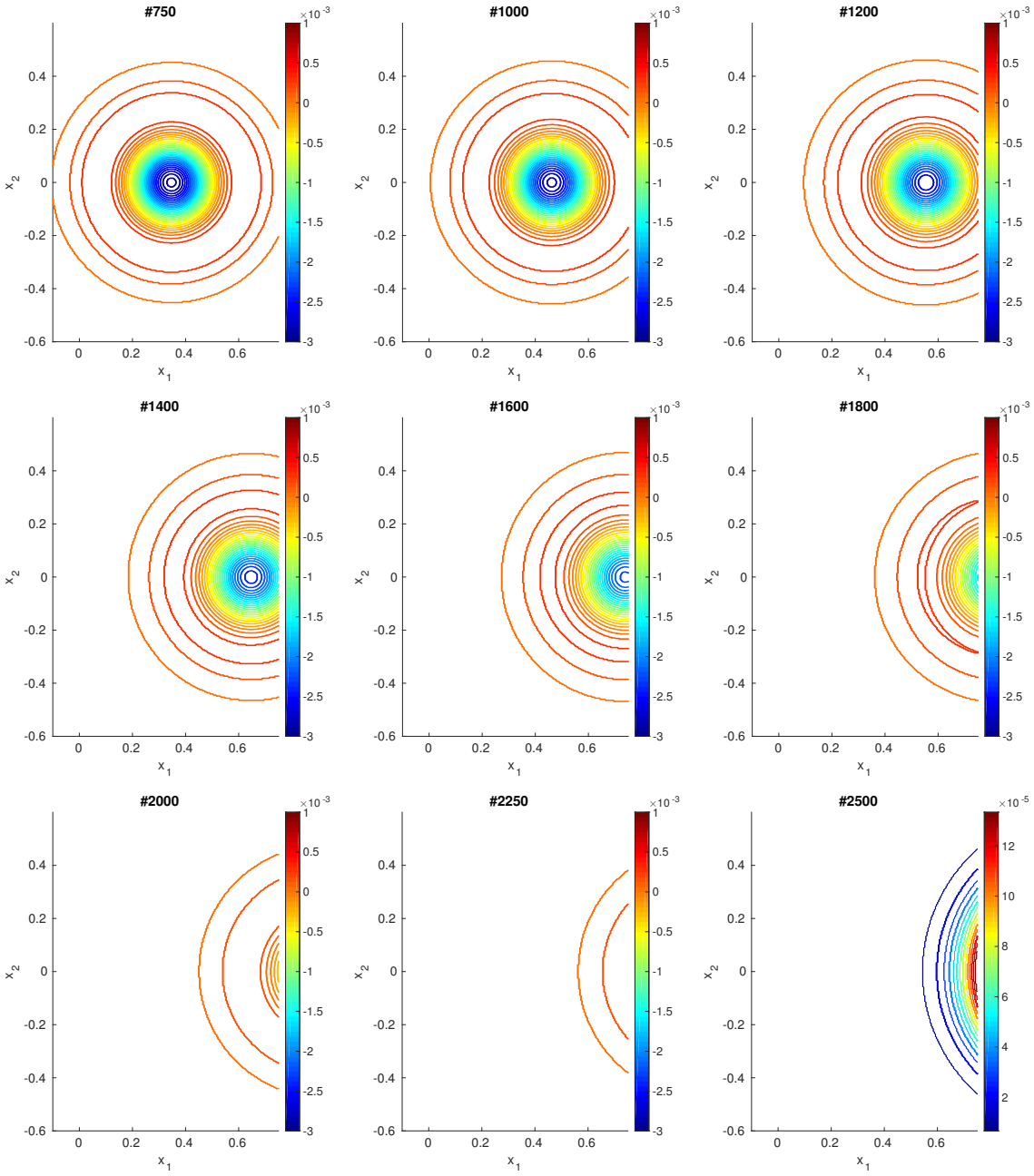


Figure E.6: *Iso-vorticity contours for an isolated vortex using* `ITBC` – The contour plots show the evolution of the vorticity (6.5) using an ITBC at the right boundary. Except for the last plot, the contour levels are the same for all time levels.

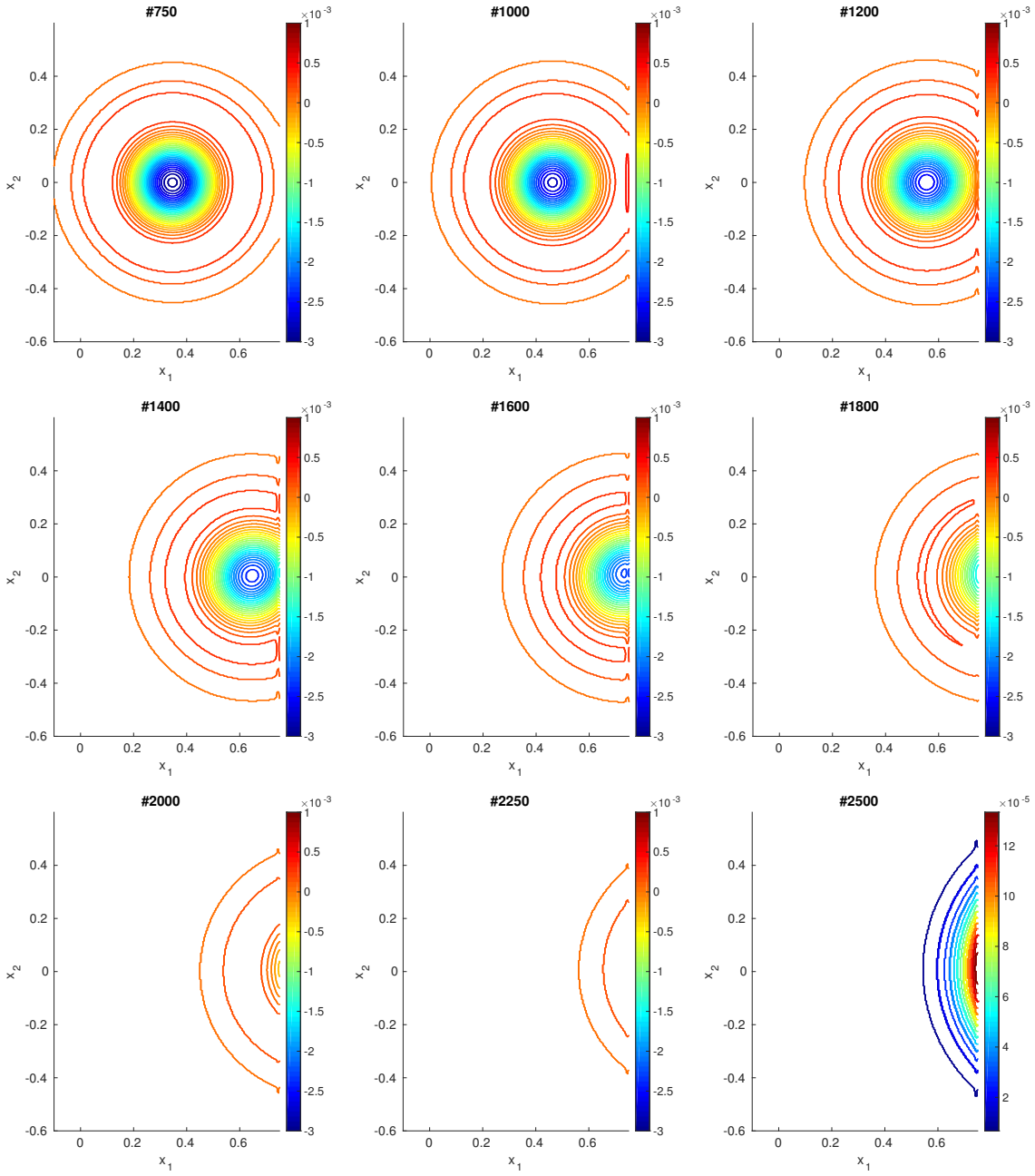


Figure E.7: *Iso-vorticity contours for an isolated vortex using* `CBC` – The contour plots show the evolution of the vorticity (6.5) using our CBC at the right boundary. Except for the last plot, the contour levels are the same for all time levels.

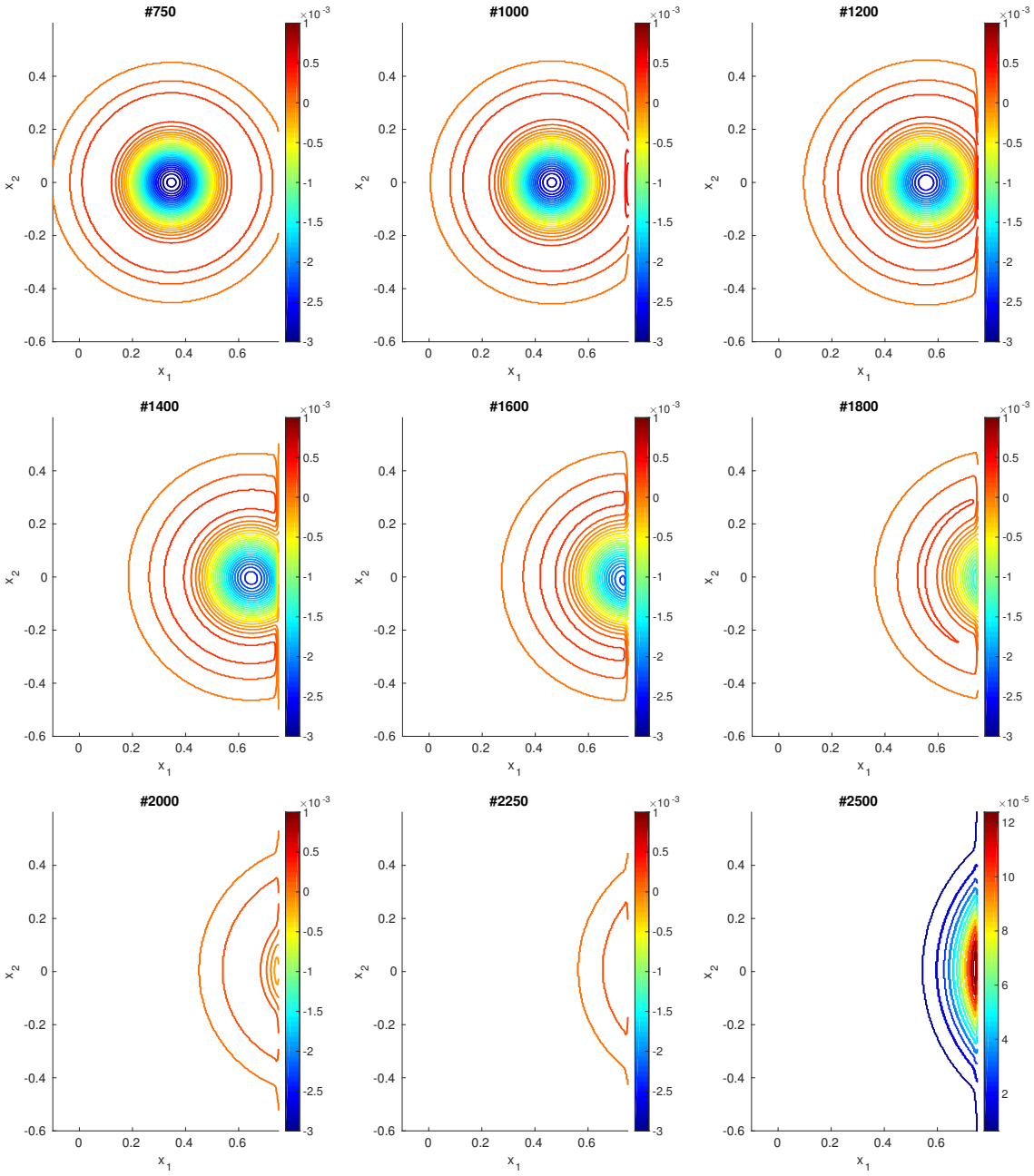


Figure E.8: *Iso-vorticity contours for an isolated vortex using* `LODI` – The contour plots show the evolution of the vorticity (6.5) using `LODI` at the right boundary. Except for the last plot, the contour levels are the same for all time levels.

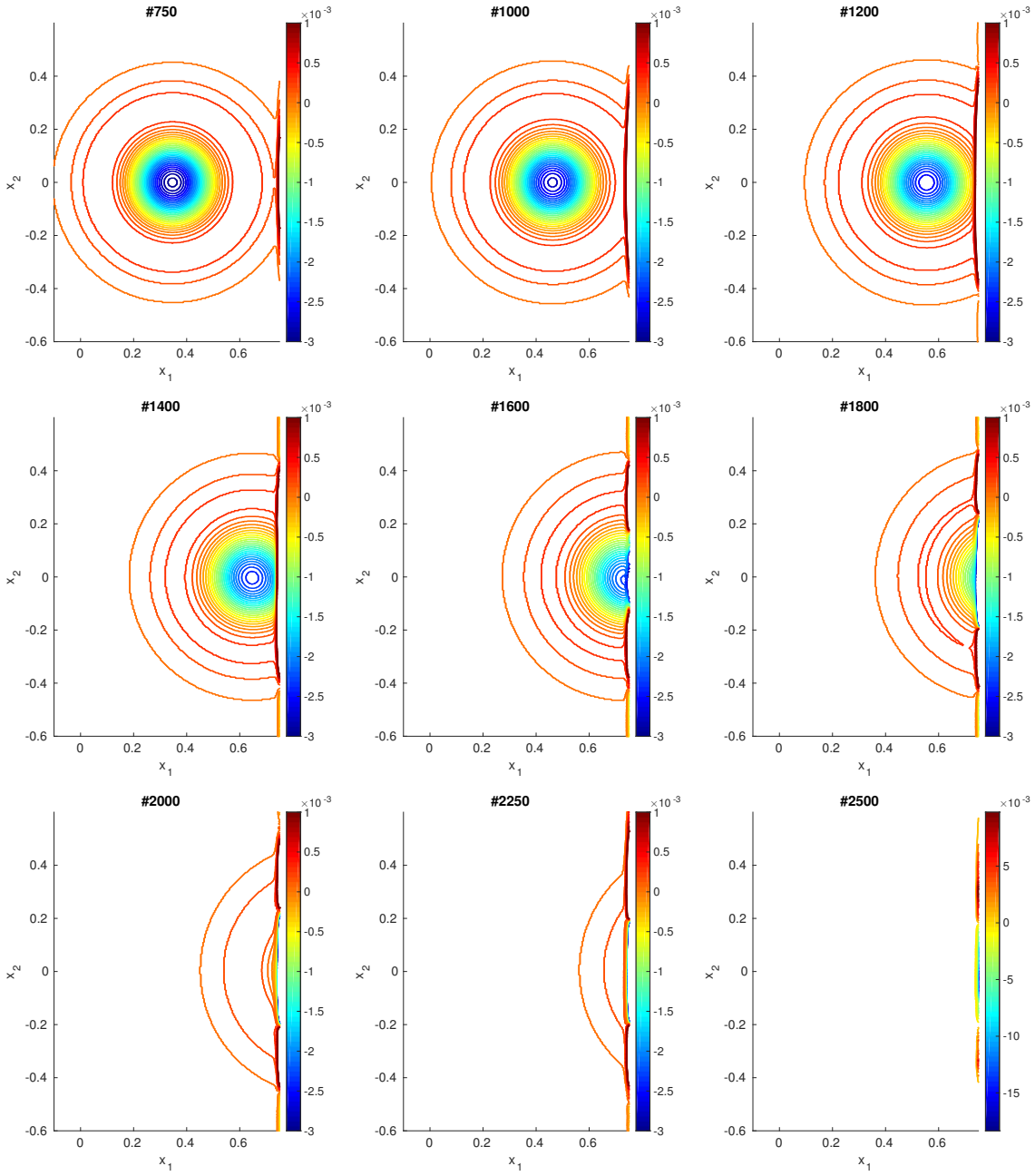


Figure E.9: *Iso-vorticity contours for an isolated vortex using* `isoImpBC` – The contour plots show the evolution of the vorticity (6.5) using the isotropic impedance boundary condition at the right boundary. Except for the last plot, the contour levels are the same for all time levels.

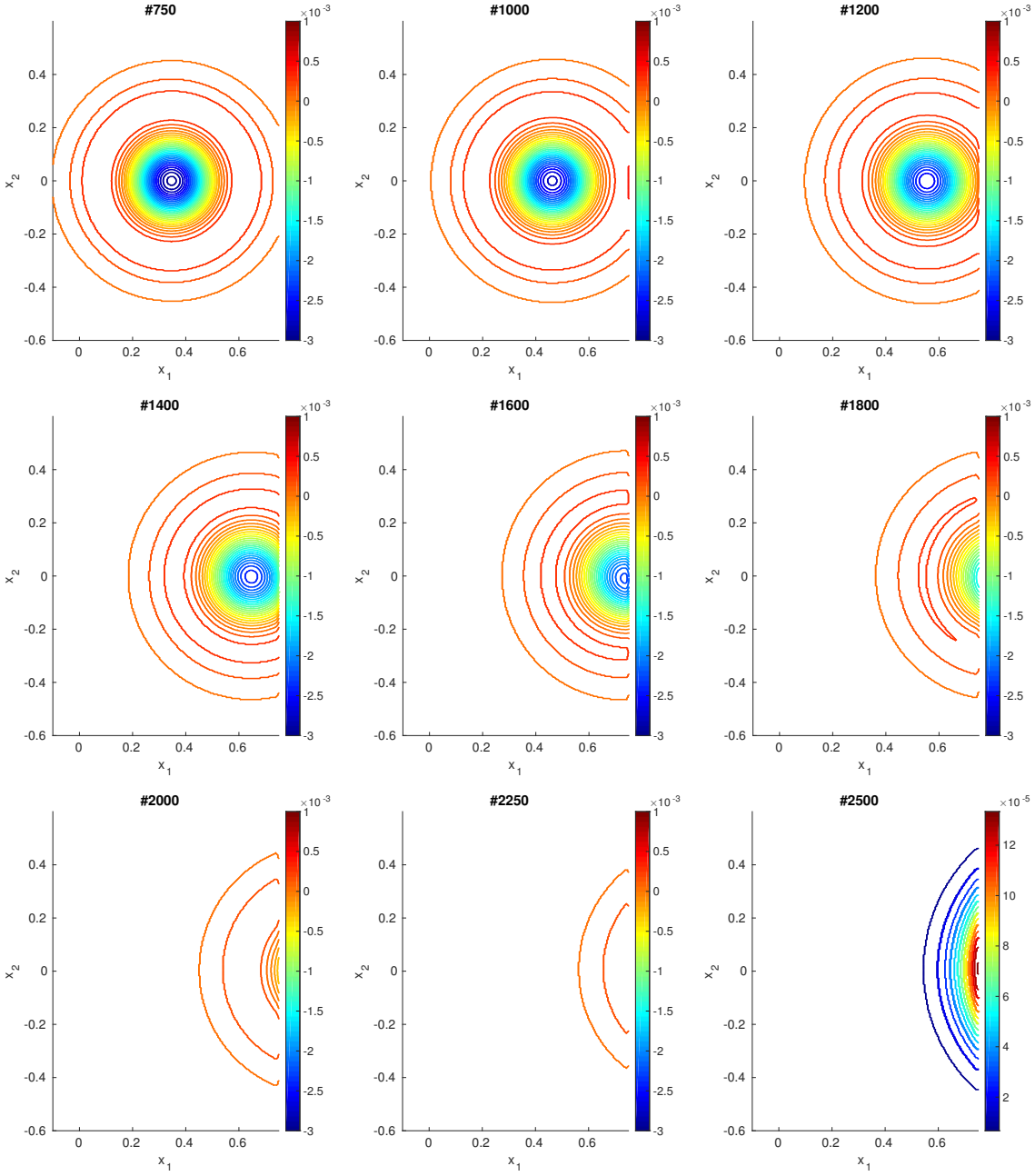


Figure E.10: *Iso-vorticity contours for an isolated vortex using* `PML` – The contour plots show the evolution of the vorticity (6.5) using a PML damping zone in the formulation of Najafi-Yazdi and Mongeau at the right boundary. Except for the last plot, the contour levels are the same for all time levels.

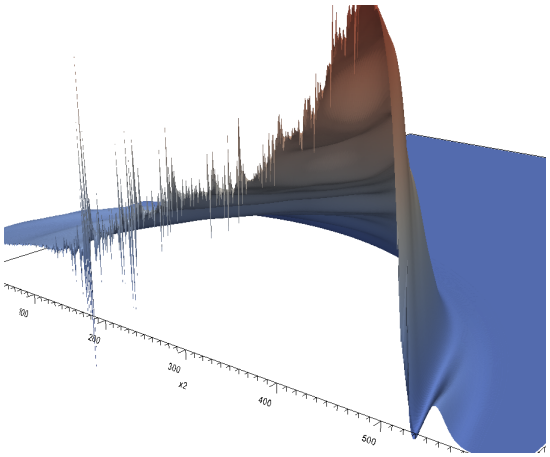


Figure E.11: *Error of an isotropic IBC in simulation of a plane wave* – This cutout shows the error in the simulation of a two-dimensional plane wave using `isoImpBC` as boundary condition. We see an oscillation at the boundary where `isoImpBC` is applied.

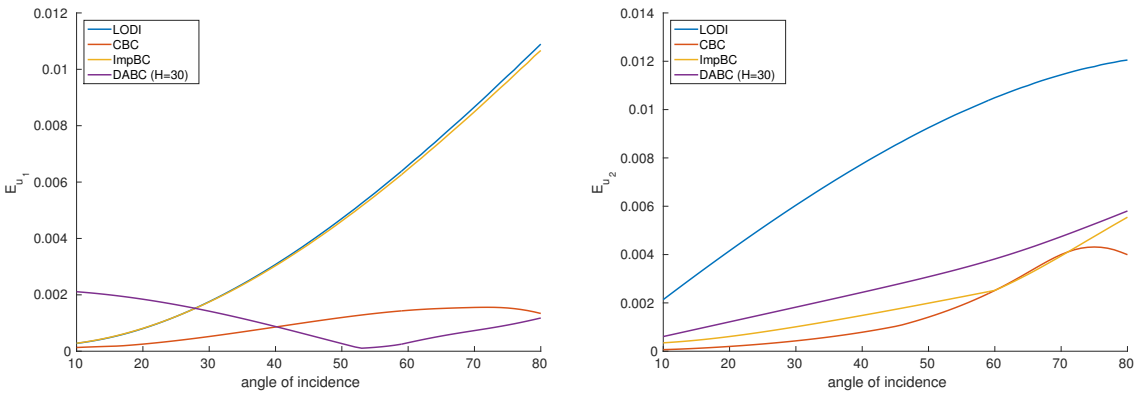


Figure E.12: *Velocity errors of plane waves with respect to the angle of incidence* – The maximal errors E_{u_1} and E_{u_2}, (6.6), are plotted for different angles of incidence in the range from 10° to 80°.

References

[1] V. V. Aristov. *Fluid Mechanics and its Applications, Volume 60: Direct Methods for Solving the Boltzmann Equation and Study of Nonequilibrium Flows*. Springer Science & Business Media, 2001.

[2] V. I. Arnold. *Graduate Texts in Mathematics, Volume 60: Mathematical Methods of Classical Mechanics*. Springer Science & Business Media, 1989.

[3] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, 2000.

[4] J.-P. Bérenger. A perfectly matched layer for the absorption of electromagnetic waves. *Journal of Computational Physics*, 114(2):185–200, 1994.

[5] P. L. Bhatnagar, E. P. Gross, and M. Krook. A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems. *Physical Review*, 94(3):511–525, 1954.

[6] N. N. Bogoliubov. Kinetic equations. *Journal of Physics USSR*, 10(3):265–274, 1946.

[7] N. N. Bogoliubov and K. P. Gurov. Kinetic equations in quantum mechanics. *Journal of Experimental and Theoretical Physics*, 17(7):614–628, 1947.

[8] M. Born and H. S. Green. A general kinetic theory of liquids. I. The molecular distribution functions. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 188(1012):10–18, 1946.

[9] C. Cercignani. On the Boltzmann equation for rigid spheres. *Transport Theory and Statistical Physics*, 2(3):211–225, 1972.

[10] C. Cercignani. *The Boltzmann Equation and Its Applications*. Springer-Verlag, 1988.

[11] C. Cercignani. *Rarefied Gas Dynamics: From Basic Concepts to Actual Calculations*. Cambridge University Press, 2000.

[12] S. Chapman and T. Cowling. *The Mathematical Theory of Non-Uniform Gases*. Cambridge University Press, 1970.

[13] H. Chen, S. Chen, and W. H. Matthaeus. Recovery of the Navier-Stokes equations using a lattice-gas Boltzmann method. *Physical Review A*, 45(8):R5339–R5342, 1992.

[14] S. Chen, K. Diemer, G. D. Doolen, K. Eggert, C. Fu, S. Gutman, and B. J. Travis. Lattice gas automata for flow through porous media. *Physica D: Nonlinear Phenomena*, 47(1–2):72–84, 1991.

[15] S. Chen and G. D. Doolen. Lattice Boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30(1):329–364, 1998.

[16] S. S. Chikatamarla, S. Ansumali, and I. V. Karlin. Grad's approximation for missing data in lattice Boltzmann simulations. *Europhysics Letters*, 74(2):215–221, 2006.

[17] S. S. Chikatamarla and I. V. Karlin. Lattices for the lattice Boltzmann method. *Physical Review E*, 79(4):046701, 2009.

[18] E. Craig and F. Q. Hu. On the perfectly matched layer for the Boltzmann-BGK equation and its application to computational aeroacoustics. *16th AIAA/CEAS Aeroacoustics Conference*, 2010–3935, 2010.

[19] P. J. Dellar. Bulk and shear viscosities in lattice Boltzmann equations. *Physical Review E*, 64:031203, 2001.

[20] E. Deutsch. Dyck path enumeration. *Discrete Mathematics*, 204(1–3):167–202, 1999.

[21] D. d'Humieres, I. Ginzburg, M. Krafczyk, P. Lallemand, and L.-S. Luo. Multiple-relaxation-time lattice Boltzmann models in three dimensions. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 360:437–451, 2002.

[22] R. Diestel. *Graph Theory.* 4$^{\text{th}}$ edition, Springer-Verlag, 2010.

[23] P. G. Drazin and N. Riley. *The Navier-Stokes Equations: A Classification of Flows and Exact Solutions.* Cambridge University Press, 2006.

[24] R. D. Dutton and R. C. Brigham. Computationally efficient bounds for the Catalan numbers. *European Journal of Combinatorics*, 7(3):211–213, 1986.

[25] M. Ehrhardt. Absorbing boundary conditions for hyperbolic systems. *Numerical Mathematics: Theory, Methods and Applications*, 3(3):295–337, 2010.

[26] M. Ehrhardt (Ed.). *Progress in Computational Physics, Volume 3: Novel Trends in Lattice-Boltzmann Methods: Reactive Flow, Physicochemical Transport and Fluid-Structure Interaction.* Bentham Science Publishers Ltd., 2013.

[27] B. Engquist and A. Majda. Absorbing boundary conditions for the numerical simulation of waves. *Mathematics of Computation*, 31:629–651, 1977.

[28] J. H. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics.* Volume 3, Springer-Verlag, 2002.

[29] C. Fletcher. *Computational Techniques for Fluid Dynamics: Fundamental and General Techniques.* Springer-Verlag, 2005.

[30] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the Navier-Stokes equation. *Physical Review Letters*, 56(14):1505–1508, 1986.

[31] A. V. Getling. *Rayleigh-Bénard Convection: Structures and Dynamics.* World Scientific, 1998 (2001, reprinted).

[32] I. Ginzburg, F. Verhaeghe, and D. d'Humieres. Two-relaxation-time lattice Boltzmann scheme: About parametrization, velocity, pressure and mixed boundary conditions. *Communications in Computational Physics*, 3(2):427–478, 2008.

[33] H. Grad. On the kinetic theory of rarefied gases. *Communications on Pure and Applied Mathematics*, 2(4):331–407, 1949.

[34] Z. Guo and C. Shu. *Lattice Boltzmann Method and its Applications in Engineering.* World Scientific, 2013.

[35] Z. Guo and T. S. Zhao. Lattice Boltzmann model for incompressible flows through porous media. *Physical Review E*, 66(3):036304, 2002.

[36] Z. Guo, C. Zheng, and B. Shi. Discrete lattice effects on the forcing term in the lattice Boltzmann method. *Physical Review E*, 65(4):046308, 2002.

[37] E. Hairer, S. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems.* 2nd edition, Springer-Verlag, 1993.

[38] J. Hardy, Y. Pomeau, and O. de Pazzis. Time evolution of a two-dimensional model system. I. Invariant states and time correlation functions. *Journal of Mathematical Physics*, 14(12):1746–1759, 1973.

[39] X. He, S. Chen, and G. D. Doolen. A novel thermal model for the lattice Boltzmann method in incompressible limit. *Journal of Computational Physics*, 146(1):282–300, 1998.

[40] X. He, S. Chen, and R. Zhang. A lattice Boltzmann scheme for incompressible multiphase flow and its application in simulation of Rayleigh–Taylor instability. *Journal of Computational Physics*, 152(2):642–663, 1999.

[41] X. He and L.-S. Luo. Lattice Boltzmann model for the incompressible Navier-Stokes equation. *Journal of Statistical Physics*, 88:927–944, 1997.

[42] X. He and L.-S. Luo. A priori derivation of the lattice Boltzmann equation. *Physical Review E*, 55(6):R6333–R6336, 1997.

[43] X. He and L.-S. Luo. Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Physical Review E*, 56(6):6811–6817, 1997.

[44] G. W. Hedstrom. Nonreflecting boundary conditions for nonlinear hyperbolic systems. *Journal of Computational Physics*, 30(2):222–237, 1979.

[45] D. Heubes, A. Bartel, and M. Ehrhardt. *An Introduction to the Lattice Boltzmann Method for Coupled Problems*, pages 3–30 in [26].

[46] D. Heubes, A. Bartel, and M. Ehrhardt. Characteristic boundary conditions in the lattice Boltzmann method for fluid and gas dynamics. *Journal of Computational and Applied Mathematics*, (262):51–61, 2014.

[47] D. Heubes, A. Bartel, and M. Ehrhardt. Exact artificial boundary conditions for a lattice Boltzmann method. *Computers & Mathematics with Applications*, 67(11):2041–2054, 2014.

[48] D. Heubes, A. Bartel, and M. Ehrhardt. Concept for a one-dimensional discrete artificial boundary condition for the lattice Boltzmann method. *Computers & Mathematics with Applications*, 70(10):2316–2330, 2015.

[49] D. Heubes, A. Bartel, and M. Ehrhardt. Discrete artificial boundary conditions for the lattice Boltzmann method in 2D. *ESAIM: Proceedings and Surveys*, 52:47–65, 2015.

[50] F. Q. Hu. A stable, perfectly matched layer for linearized Euler equations in unsplit physical variables. *Journal of Computational Physics*, 173(2):455–480, 2001.

[51] D. Hänel. *Molekulare Gasdynamik*. Springer-Verlag, 2004.

[52] T. Inamuro, M. Yoshino, and F. Ogino. A non-slip boundary condition for lattice Boltzmann simulations. *Physics of Fluids*, 7:2928–2930, 1995.

[53] S. Izquierdo and N. Fueyo. Characteristic nonreflecting boundary conditions for open boundaries in lattice Boltzmann methods. *Physical Review E*, 78:046707, 2008.

[54] S. Izquierdo, P. Martinez-Lera, and N. Fueyo. Analysis of open boundary effects in unsteady lattice Boltzmann simulations. *Computers & Mathematics with Applications*, 58:914–921, 2009.

[55] M. Junk. A finite difference interpretation of the lattice Boltzmann method. *Numerical Methods for Partial Differential Equations*, 17(4):383–402, 2001.

[56] M. Junk and M. Rheinländer. Regular and multiscale expansions of a lattice Boltzmann method. *Progress in Computational Fluid Dynamics*, 8:25–37, 2006.

[57] E. Kam, R. So, and R. Leung. Lattice Boltzmann method simulation of aeroacoustics and nonreflecting boundary conditions. *American Institute of Aeronautics and Astronautics*, 45(7):1703–1712, 2007.

[58] Q. Kang, P. Lichtner, and D. R. Janecky. Lattice Boltzmann method for reacting flows in porous media. *Advances in Applied Mathematics and Mechanics*, 2(5):545–563, 2010.

[59] D. Kim, H. M. Kim, M. S. Jhon, S. J. Vinay III, and J. Buchanan. A characteristic non-reflecting boundary treatment in lattice Boltzmann method. *Chinese Physics Letters*, 25(6):1964, 2008.

[60] L. E. Kinsler, A. R. Frey, A. B. Coppens, and J. V. Sanders. *Fundamentals of Acoustics*. John Wiley & Sons, Inc., 4^{th} edition, 2000.

[61] J. G. Kirkwood. The statistical mechanical theory of transport processes I. General theory. *The Journal of Chemical Physics*, 14(3):180–201, 1946.

[62] J. G. Kirkwood. The statistical mechanical theory of transport processes II. Transport in gases. *The Journal of Chemical Physics*, 15(1):72–76, 1947.

[63] C. Kleinstreuer. *Two-Phase Flow: Theory and Applications*. Taylor & Francis, 2003.

[64] P. Kowalczyk, A. Palczewski, G. Russo, and Z. Walenta. Numerical solutions of the Boltzmann equation: comparison of different algorithms. *European Journal of Mechanics-B/Fluids*, 27(1):62–74, 2008.

[65] D. Kröner. Absorbing boundary conditions for the linearized Euler equations in 2-D. *Mathematics of Computation*, 57(195):153–167, 1991.

[66] M. Kutrib, R. Vollmar, and T. Worsch. Introduction to the special issue on cellular automata. *Parallel Computing*, 23(11):1567–1576, 1997.

[67] H. Lai and C. Ma. The lattice Boltzmann model for the second-order Benjamin–Ono equations. *Journal of Statistical Mechanics: Theory and Experiment*, 2010:P04011, 2010.

[68] P. Lallemand and L.-S. Luo. Hybrid finite-difference thermal lattice Boltzmann equation. *International Journal of Modern Physics B*, 17(1–2):41–47, 2003.

[69] R. J. LeVeque. *Numerical Methods for Conservation Laws*. 2nd edition, Birkhäuser-Verlag, 1992.

[70] H. Liu, H. Wang, S. Liu, C. Hu, Y. Ding, and J. Zhang. Lattice Boltzmann method for the Saint–Venant equations. *Journal of Hydrology*, 524:411–416, 2015.

[71] R. Löhner. *Applied Computational Fluid Dynamics Techniques: An Introduction Based on Finite Element Methods*. John Wiley & Sons, 2008.

[72] T. Mansour and Y. Sun. Identities involving Narayana polynomials and Catalan numbers. *Discrete Mathematics*, 309(12):4079–4088, 2009.

[73] N. S. Martys and H. Chen. Simulation of multicomponent fluids in complex three-dimensional geometries by the lattice Boltzmann method. *Physical Review E*, 53(1):743–750, 1996.

[74] G. R. McNamara and G. Zanetti. Use of the Boltzmann equation to simulate lattice-gas automata. *Physical Review Letters*, 61(20):2332–2335, 1988.

[75] M. Mendoza and J. D. Munoz. Three dimensional lattice-Boltzmann model for electrodynamics. *Physical Review E*, 82:056708, 2010.

[76] M. Mendoza, S. Succi, and H. Herrmann. Kinetic formulation of the Kohn-Sham equations for ab initio electronic structure calculations. *Physical Review Letters*, 113(9):096402, 2014.

[77] A. Najafi-Yazdi and L. Mongeau. An absorbing boundary condition for the lattice Boltzmann method based on the perfectly matched layer. *Computers & Fluids*, 68:203–218, 2012.

[78] S. Palpacelli, S. Succi, and R. Spigler. Ground-state computation of Bose-Einstein condensates by an imaginary-time quantum lattice Boltzmann scheme. *Physical Review E*, 76(3):036712, 2007.

[79] C. Pan, L.-S. Luo, and C. T. Miller. An evaluation of lattice Boltzmann schemes for porous medium flow simulation. *Computers & Fluids*, 35(8):898–909, 2006.

[80] Y. Peng, C. Shu, and Y. T. Chew. A 3D incompressible thermal lattice Boltzmann model and its application to simulate natural convection in a cubic cavity. *Journal of Computational Physics*, 193(1):260–274, 2003.

[81] Y. Peng, C. Shu, and Y. T. Chew. Simplified thermal lattice Boltzmann model for incompressible thermal flows. *Physical Review E*, 68(2):026701–026709, 2003.

[82] T. Platkowski and R. Illner. Discrete velocity models of the Boltzmann equation: A survey on the mathematical aspects of the theory. *SIAM Review*, 30(2):213–255, 1988.

[83] P. Plotnikov and J. Sokołowski. *Compressible Navier-Stokes equations: Theory and Shape Optimization*. Springer Science & Business Media, 2012.

[84] T. Poinsot and S. Lele. Boundary conditions for direct simulations of compressible viscous flows. *Journal of Computational Physics*, 101(1):104–129, 1992.

[85] C. Pozrikidis. *Fluid Dynamics: Theory, Computation, and Numerical Simulation*. Springer Science & Business Media, 2009.

[86] Y. Qian, D. d'Humieres, and P. Lallemand. Lattice BGK models for Navier-Stokes equation. *Europhysics Letters*, 17(6):479–484, 1992.

[87] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical Mathematics*. 2nd edition, Springer Science & Business Media, 2007.

[88] A. Quarteroni and A. Valli. *Springer Series in Computational Mathematics, Volume 23: Numerical Approximation of Partial Differential Equations*. Springer Science & Business Media, 2008.

[89] D. F. Rogers. *Laminar Flow Analysis*. Cambridge University Press, 1992.

[90] C. W. Rowley and T. Colonius. Discretely nonreflecting boundary conditions for linear hyperbolic systems. *Journal of Computational Physics*, 157(2):500–538, 2000.

[91] D. H. Rudy and J. C. Strikwerda. A nonreflecting outflow boundary condition for subsonic Navier-Stokes calculations. *Journal of Computational Physics*, 36(1):55–70, 1980.

[92] L. Saint-Raymond. *Hydrodynamic Limits of the Boltzmann Equation*. Springer-Verlag, 2009.

[93] M. Schlaffer. *Non-reflecting Boundary Conditions for the Lattice Boltzmann Method*. PhD thesis, TU München (Germany), 2014.

[94] L. Selle, F. Nicoud, and T. J. Poinsot. Actual impedance of nonreflecting boundary conditions: Implications for computation of resonators. *AIAA Journal*, 42(5):958–964, 2004.

[95] X. Shan and H. Chen. Lattice Boltzmann model for simulating flows with multiple phases and components. *Physical Review E*, 47(3):1815–1819, 1993.

[96] X. Shan and G. Doolen. Multicomponent lattice-Boltzmann model with interparticle interaction. *Journal of Statistical Physics*, 81:379–393, 1995.

[97] B. Shi and Z. Guo. Lattice Boltzmann model for nonlinear convection-diffusion equations. *Physical Review E*, 79(1):016701, 2009.

[98] P. A. Skordos. Initial and boundary conditions for the lattice Boltzmann method. *Physical Review E*, 48(6):4823–4842, 1993.

[99] Y. Sone. *Molecular Gas Dynamics: Theory, Techniques, and Applications*. Birkhäuser-Verlag, 2007.

[100] J. Spurk and N. Aksel. *Fluid Mechanics*. 2nd edition, Springer-Verlag, 2008.

[101] J. D. Sterling and S. Chen. Stability analysis of lattice Boltzmann methods. *Journal of Computational Physics*, 123(1):196–206, 1996.

[102] H. Struchtrup. *Macroscopic Transport Equations for Rarefied Gas Flows*. Springer-Verlag, 2005.

[103] S. Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond*. Oxford University Press, 2001.

[104] M. C. Sukop and D. T. Thorne Jr. *Lattice Boltzmann Modeling: An Introduction for Geoscientists and Engineers*. Springer-Verlag, 2007.

[105] G. Szegö. *Orthogonal Polynomials*. 4th edition, American Mathematical Society, 1975.

[106] M. Tekitek, M. Bouzidi, F. Dubois, and P. Lallemand. Towards perfectly matching layers for lattice Boltzmann equation. *Computers & Mathematics with Applications*, 58(5):903–913, 2009.

[107] R. Temam. *Navier-Stokes Equations: Theory and Numerical Analysis*. AMS Chelsea Publishing, 2001.

[108] K. W. Thompson. Time dependent boundary conditions for hyperbolic systems. *Journal of Computational Physics*, 68(1):1–24, 1987.

[109] K. W. Thompson. Time-dependent boundary conditions for hyperbolic systems, II. *Journal of Computational Physics*, 89(2):439–461, 1990.

[110] J. Tölke and M. Krafczyk. TeraFLOP computing on a desktop PC with GPUs for 3D CFD. *International Journal of Computational Fluid Dynamics*, 22(7):443–456, 2008.

[111] R. van der Sman and M. Ernst. Convection-diffusion lattice Boltzmann scheme for irregular lattices. *Journal of Computational Physics*, 160(2):766–782, 2000.

[112] Y. Vanderhoydonc. *Numerical Lifting Operators for Kinetic Boltzmann Models*. PhD thesis, Universiteit Antwerpen (Belgium), 2006.

[113] Y. Vanderhoydonc and W. Vanroose. Initialization of lattice Boltzmann models with the help of the numerical Chapman–Enskog expansion. *Procedia Computer Science*, 18:1036–1045, 2013.

[114] Y. Vanderhoydonc, W. Vanroose, C. Vandekerckhove, P. Van Leemput, and D. Roose. *Numerical Lifting for Lattice Boltzmann Models*, pages 127–154 in [26].

[115] E. Vergnault, O. Malaspinas, and P. Sagaut. A lattice Boltzmann method for nonlinear disturbances around an arbitrary base flow. *Journal of Computational Physics*, 231(24):8070–8082, 2012.

[116] E. M. Viggen. Viscously damped acoustic waves with the lattice Boltzmann method. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 369(1944):2246–2254, 2011.

[117] J. von Neumann. *Theory of Self-Reproducing Automata.* University of Illinois Press, 1966.

[118] J. Wendt. *Computational Fluid Dynamics: An Introduction.* 3rd edition, Springer Science & Business Media, 2009.

[119] J. C. Wilson. Derivation of boundary conditions for the artificial boundaries associated with the solution of certain time dependent problems by Lax–Wendroff type difference schemes. *Proceedings of the Edinburgh Mathematical Society*, 25:1–18, 1982.

[120] D. Wolf-Gladrow. *Lattice-Gas Cellular Automata and Lattice Boltzmann Models.* Springer-Verlag, 2000.

[121] Q. Xiong, B. Li, J. Xu, X. Fang, X. Wang, L. Wang, X. He, and W. Ge. Efficient parallel implementation of the lattice Boltzmann method on large clusters of graphic processing units. *Chinese Science Bulletin*, 57(7):707–715, 2012.

[122] J. Yvon. La théorie statistique des fluides et l'équation d'état. *Actualités Acientifiques et Industriel*, 203, 1935.

[123] G. Zanetti. Hydrodynamics of lattice-gas automata. *Physical Review A*, 40(3):1539–1548, 1989.

[124] J. Zhang and G. Yan. A lattice Boltzmann model for the Korteweg–de Vries equation with two conservation laws. *Computer Physics Communications*, 180(7):1054–1062, 2009.

[125] G. Zhao-Li, Z. Chu-Guang, and S. Bao-Chang. Non-equilibrium extrapolation method for velocity and pressure boundary conditions in the lattice Boltzmann method. *Chinese Physics*, 11(4):366–374, 2002.

[126] J. G. Zhou. *Lattice Boltzmann Methods for Shallow Water Flows.* Springer-Verlag, 2004.

[127] Q. Zou and X. He. On pressure and velocity boundary conditions for the lattice Boltzmann BGK model. *Physics of Fluids*, 9(6):1591–1598, 1997.