Masters Thesis

Investigation of Electrolytic Two-Phase Flow Systems using the Lattice Boltzmann Method

Alexander Reinauer

April 30, 2021

Examiner:	Prof. Dr. Christian Holm Institute for Computational Physics
Co-Examiner:	Prof. Dr. Siegfried Dietrich Max-Planck-Institute for Intelligent Systems

Eigenständigkeitserklärung

Ich erkläre mit meiner Unterschrift, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen dieser Arbeit, die dem Wortlaut, dem Sinn oder der Argumentation nach anderen Werken entnommen sind (einschließlich des World Wide Web und anderer elektronischer Text- und Datensammlungen), habe ich unter Angabe der Quellen vollständig kenntlich gemacht.

30.04.2021, Stuttgart
Datum, Ort

Unterschrift

Zusammenfassung

Das Verstehen und Vorhersagen des Verhaltens von Fluiden ist ein sehr komplexes Thema, für das noch immer fundamentale Fragen ungelöst sind. Die beschreibenden partiellen Differentialgleichungen sind bekannt als die Navier-Stokes Gleichungen, deren mathematische Formulierung abhängig von den Eigenschaften des Fluids leicht abweichen können. Seit über einem Jahrhundert schon sind diese Gleichungen bekannt, jedoch sind sie notorisch schwer zu lösen, weshalb nur wenige analytische Lösungen für sehr spezielle Systeme bekannt sind. Deshalb sind in den letzen Jahrzehnten numerische Löser populär geworden, welche verschiedene Lösungstechniken verwenden und somit für verschiedene Längenskalen geeignet sind. In den vergangenen Jahrzenten wurden verschiedene Modelle entwickelt, um diese numerischen Löser zu erweitern, insbesondere ist die Erweiterung auf mehrphasige/mehrkomponentige Fluidsysteme von Relevanz für das Verständnis von derartigen Systemen. Zusätzlich zu den mehrkomponentigen Systemen können gelöste Ionen z.B. Salz, das Verhalten der Fluide beeinflussen. Die Modellierung dieser gelösten Ionen kann auf verschiedenen Längenskalen geschehen und ist abhängig von der zu untersuchenden Längenskala, welche von der Simulation einzelner Ionen bis hin zu kontinuierlichen Dichtebeschreibungen reichen kann. Beispiele, die den Einfluss der Salzkonzentration auf das Fluidverhalten zeigen sind Öltropfen in Salzwasser, welche durch elektrische Felder getrieben werden oder die Veränderung des Benetzungsverhalten von Öltropfen durch das gelöste Salz. Das letztere Beispiel tritt insbesondere bei der Extraktion von Öl aus Ölfeldern auf. Hierbei werden verschiedene Methoden unter dem Namen „Enhanced Oil Recovery“ zusammengefasst, welche das Ziel der Erhöhung der extrahierten Ölmenge vereinen. Einer dieser Methoden ist unter dem Namen Low-Salinity Flooding [1] bekannt, bei der die Ölfelder mit Salzwasser mit geringer Salzkonzentration geflutet werden um das Rohöl zu extrahieren. Die auftretenden physikalischen Mechanismen bei dieser Methode sind nicht vollständig verstanden, weshalb verschiedene Laborexperimente und Simulationen durchgeführt wurden. Eines dieser Laborexperimente ist von Mahani et al. [2], welche einen Öltropfen auf einer Tonablagerung in umgebender Salzlösung verschiedener Salzkonzentration untersuchen. Hierbei stellten sie fest, dass das Benetzungsverhalten und damit auch der Kontaktwinkel des Öltropfens mit fallender Salzkonzentration abnimmt und der Tropfen sich nach mehreren Tagen der Beobachtung von der Tonablagerung ablöst.

Ein numerischer Löser für die Navier-Stokes Gleichung ist die Gitter-Boltzmann Methode, welche ihre Wurzeln in der Boltzmann-Gleichung hat. Diese Methode kann durch verschiedene Modell für mehrkompoentige Fluide erweitert werden, dazu zählen das Shan-Chen-Modell, das Free-Energy oder das jüngere Color-Gradient Gitter Boltzmann Modell. In dieser Arbeit wird das Shan-Chen-Modell verwendet und an

einen diskreten Gitter Elektrokinetik Löser gekoppelt, welcher die Nernst-Planck Gleichung für die Ionendichten löst. Für die Diskretisierung der Nernst-Planck Gleichung auf einem kubischen Gitter wird ein finites Volumen-Verfahren verwendet. Die Kopplung dieser beiden Methoden beinhaltet die Reibungskopplung der Fluide an den Ionenstrom, die Advektion der Ionen durch das Fluid und die chemischen Eigenschaften der Fluidkomponenten durch das chemische Potential.

In dieser Arbeit wird das beschriebene Modell für das mehrkomponentige Fluidmodell in einem Code-Generations Framework namens pystencils [3] implementiert, welches in der Lage ist hochoptimierten Quellcode für gitterbasierte Algorithmen aus mathematischen Beschreibungen zu erzeugen. Für die Gitter-Boltzmann Methode existiert bereits eine eigene Erweiterung names lbmpy [4], welche speziell für die Erzeugung von Quellcode für Gitter-Boltzmann Algorithmen entwickelt wurde und ebenfalls pystencils als Grundlage verwendet. Basierend auf dem gleichen Framework wird ebenfalls der Gitter Elektrokinetik Algorithmus implementiert, welcher einen Löser für die elektrostatische Poisson-Gleichung als auch die Nernst-Planck Gleichung beinhaltet. Diese Implementation wird an verschiedenen physikalischen Systemen auf ein korrektes physikalisches Verhalten getestet und anschließend dazu verwendet, den Einfluss der Salzkonzentration und verschiedenen anderen Parametern auf die Oberflächenspannung eines Öltropfens in Wasser zu untersuchen.

Contents

1. Introduction

Studying the flow of fluids is an important topic in physics and engineering and is often crucial in understanding physical phenomena. The partial differential equations describing fluid motion are the Navier-Stokes equations. They have different mathematical formulations depending on the properties of the fluid and are valid for a wide range of length scales. Despite the fact that the equations have been known for over a century they are difficult to solve and analytical solutions are known only for a few problems. This is why numerical solvers have become popular in the past decades where different methods have been developed to resolve different length scales. One of these solvers is the Lattice Boltzmann (LB) Method which is used in this work. Especially of scientific interest is currently the computation of multiphase/multicomponent fluid systems which can include phase transitions or mixtures of different fluid components. These systems can also be simulated with the LB Method by multiphase extensions, e.g. the Shan-Chen, Color-Gradient, or Free-Energy LB models.

In most real-world experiments the ion concentration in the fluid phases is non-zero and can alter the flow of the fluid. To include ion effects in the simulation the length scale of interest is important which can range from the observation of individual ions with can be simulated with particle based Molecular Dynamics to ion density descriptions which are formulated by solving the Nernst-Planck equation.

One fluid system of interest related to this work are oil droplets surrounded by brines of different salinity in a porous environment. These systems are especially relevant for the oil extraction from oil fields where different methods are grouped under the name Enhanced Oil Recovery. All of these methods have the common goal to increase the yield of oil extracted from the fields. One of such a technique is the Low-Salinity Flooding [1] where the oil reservoir is flooded with low salinity water. The exact mechanisms used during this enhanced recovery method are not well understood. Therefore several experiments have been performed under lab conditions, one of them the work by Mahani et al. [2] where the contact angle of oil droplets on clay patches is measured after exposure to brines of different salinities. In their work, they observe that the change in salinity also changes the contact angle, which in their set-up, causes the oil droplets to detach from the clay patches on the timescale of days.

In this work, a mesoscopic multicomponent fluid model including salt ions is implemented and tested for several physical systems. The multicomponent nature of the fluids is modeled with the Shan-Chen model. The salt ions that are simulated with a grid-based discretization scheme of the Nernst-Planck equation that is coupled to the Shan-Chen LB model, known as the lattice electrokinetics method. The coupling between the two models is realized via a friction coupling between the ion fluxes, the

advection of the ions through the movement of the solvents, as well as the the solvent chemical potential that describes the chemical nature of the solvents. This model is then used for a first numerical investigation of the effects of the salt on the surface tension of an oil-brine system.

2. Theory

In this section a short introduction to the low-salinity water flooding is provided. We moreover present the mathematical fundamentals to the modeling techniques and methods, where we focus on providing a full description of the equations and highlight the discretization techniques involved in the solution process.

2.1. Introduction to Low-Salinity Water Flooding

To increase the amounts of extracted oil from oil reservoirs is economically of great interest, and one would like to gain a basic understanding of the underlying physical mechanisms. The technical extraction process consists of several steps, the first primary oil recovery step uses the pressure inside the oil field which is strong enough to push the oil inside the wellbore from where it can be pumped to the surface mechanically. This step on its own leaves a portion of the oil in the field which can be extracted using enhanced oil recovery techniques that are only applied if they are economically reasonable and therefore depend on a wide variety of variables included local availability of resources and the current oil price. There exist different techniques for enhanced oil recovery: thermal treatment of oil which is aiming for a reduced viscosity of oil or injecting fluids which are supposed to influence the porous media-oil interactions where polymer flooding or water flooding are examples for [1, 5]. The efficiency of the techniques applied depend strongly on the local properties of the oil field and also on the local accessibility of the required resources. The latter is the reason that the water flooding technique is applied because of the high availability and the low cost. However, for this technique the exact mechanisms are not exactly understood, and are subject to ongoing theoretical and experimental works. One of the observed experimental results is the dependency of the yield of oil on the salt concentration present in the water used for the flooding, showing that high salinity brine is less effective compared to low salinity brine [5] which is why the technique is called low-salinity water flooding.

The laboratory scale experiments focus on single aspects which can elucidate the influence of the brine in the oil recovery process. One of such experiments was performed by Mahani et al. [2] and investigates the change in the wetting state of oil droplets when exposing them to brines of different salinities. In the experiment they investigated the contact angle of an oil droplet on a clay patch over time while diluting the surrounding salinity of the brine from high (HS) to low salinity (LS). For the experiment a glass plate was used on which clay patches are put on. According to the authors the clay patches were used to mimic conditions in sandstone oil fields. On top of the clay patches oil droplets were placed and the whole set-up was exposed

to high salinity brine. After a specific observation time the brine was replaced with low salinity brine and the relaxation process was observed again until the droplet detached from the clay patch. The droplet was observed with a camera and the contact angle was measured from the video footage. A series of snapshots of the experimental set-up is shown in Figure 2.1. The resulting contact angle over the time curve is shown in Figure 2.2.

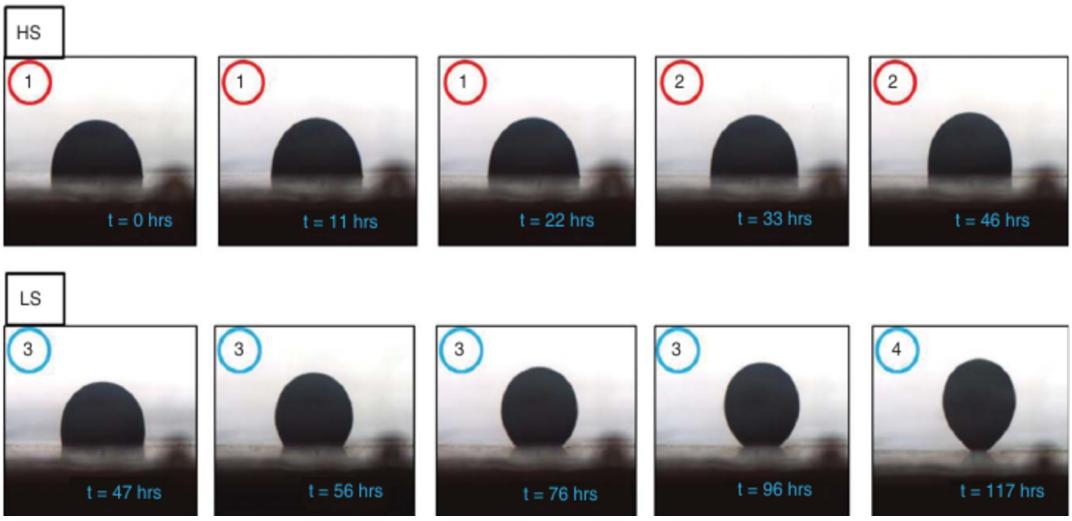
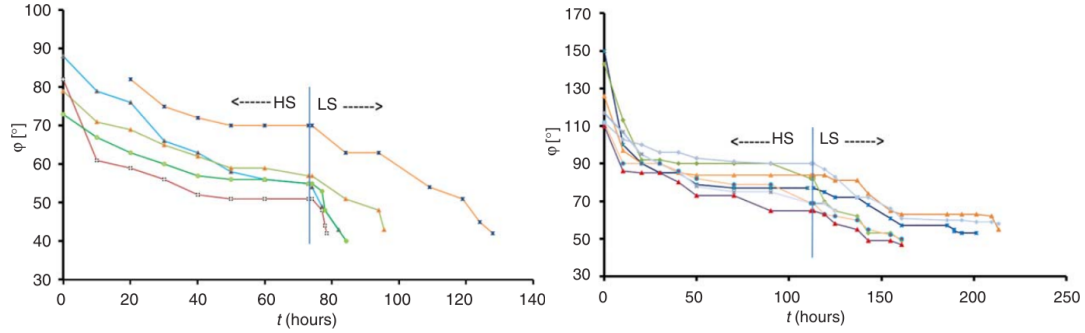


Figure 2.1.: Images of experimental set-up for contact angle measurements of oil droplets on clay patches exposed to high salinity (HS) brine and low salinity (LS) brine. The colors correspond to the salinity and the numbers refer to stages of the detachment process which were defined in the corresponding paper [2].

In all cases in the experiment the oil droplet detached from the clay patches which is the end of the contact angle measurement lines.



(a) (b)

Figure 2.2.: Measured contact angle curves over time starting at high salinity brine and exchanged with low salinity brine at the vertical blue line. The dilution factor from HS to LS brine is four for (a) and eight for (b). The different colors correspond to different samples taken and the individual curves end with the detachment of the droplets. [2]

2.2. Fluid Dynamics with the Lattice Boltzmann Method

When describing the behavior of a fluid in simulations there can be different length scales of interest depending on the quantity of interest. An illustration of the different possible length scales is shown in Figure 2.3.

The smallest scale possible is the microscale which is suitable for problems requiring the trajectories of individual atoms or molecules but is limited to small systems and time scales due to the increasing number of atoms which have to be treated when the system size increases. In these systems Newton's equation of motion are solved individually for each particle. The equation reads

$$m\ddot{\vec{x}} = -\vec{\nabla}V(\vec{x}) \tag{2.2.1}$$

which relates the acceleration $\ddot{\vec{x}}$ to the applied forces given by the gradient of the potential $V(\vec{x})$, where m is the mass of the particle. Systems on these length scales are typically solved using Molecular Dynamics, using an integration scheme in time. [7]

The other end of possible length scales is formed by the macroscale. On this length scale the fluid is described as a continuous medium using density and velocity fields. This can be done because the atomistic nature of the individual particles is negligible on these large length scales. The quantities of interest on this scale are the density or velocity which are connected by the Navier-Stokes Equation which reads for an incompressible Newtonian isotropic fluid

$$\rho\left(\frac{\partial\vec{v}}{\partial t} + (\vec{v}\cdot\vec{\nabla})\vec{v}\right) = -\vec{\nabla}p + \mu\Delta\vec{v} + \vec{f} \tag{2.2.2}$$

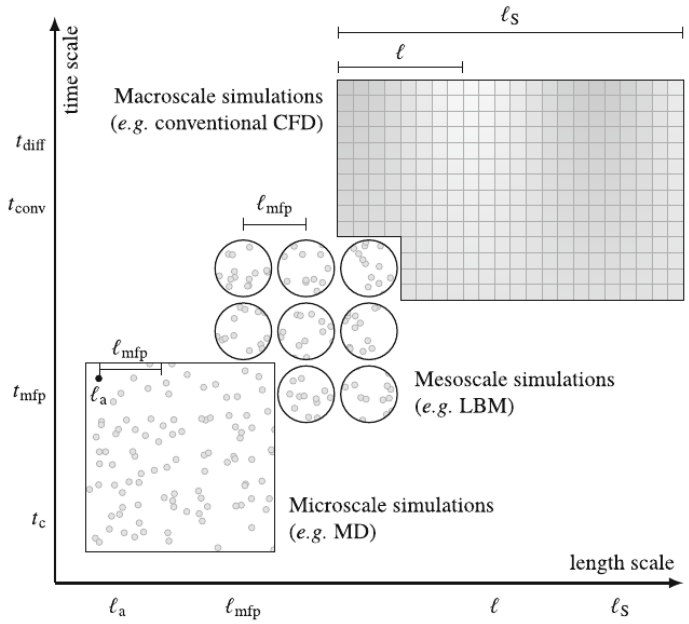


Figure 2.3.: The length and time scales typically present in fluid dynamics simulations, reaching from the description of individual atoms to a continuum description using density fields. [6, p. 12]

where ϱ is the density, $\vec{v}$ the fluid velocity, p the pressure and $\vec{f}$ the external force density.

The length scale between the two extremes is described as the mesoscale which describes the transition regime. On this scale the individual trajectories of particles is neglected but the atomistic nature is not fully negligible. This is why the description is performed statistically which is further discussed in the following section.

2.2.1. Introduction to Kinetic Theory of Gases

The kinetic theory of gases describes the behavior of molecules in a statistical way by introducing the distribution of particles in a specific volume. This is especially well suited for systems of interest in the previously discussed mesoscopic length scale.

The key variable of the kinetic theory of gases is the particle distribution function $f(\vec{x}, \vec{\xi}, t)$ and it describes the density of particle at the position $\vec{x}$ at the time t with the velocity $\vec{\xi}$. This quantity contains only the statistical behavior of molecules or atoms and is connected to physical quantities mass density ϱ, momentum density given by the product of the mass density and the velocity $\vec{v}$ and the total energy density E in the following way:

$$\varrho(\vec{x},t) = \int f(\vec{x},\vec{\xi},t)\mathrm{d}\vec{\xi} \tag{2.2.3a}$$

$$\varrho(\vec{x},t)\vec{v}(\vec{x},t) = \int \vec{\xi} f(\vec{x},\vec{\xi},t)\mathrm{d}\vec{\xi} \tag{2.2.3b}$$

$$\varrho(\vec{x},t)E(\vec{x},t) = \int |\vec{\xi}|^2 f(\vec{x},\vec{\xi},t)\mathrm{d}\vec{\xi}. \tag{2.2.3c}$$

The governing evolution equation of the distribution function is the Boltzmann Equation which is given by the total derivative of the particle distribution function

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \vec{\xi}\cdot\frac{\partial f}{\partial \vec{x}} + \frac{\vec{F}}{\varrho}\cdot\frac{\partial f}{\partial \vec{\xi}} = \Omega(f), \tag{2.2.4}$$

where Ω is known as the collision operator. This operator describes the local changes of the particle distributions due to particle collisions. It relaxes the distributions towards an equilibrium distribution which is dependent on the system of interest, which is typically the Maxwell-Boltzmann distribution. This collision operator has to fulfill physically motivated restrictions on the conservation of the previously shown moments in Equation (2.2.3) which are known as the conservation of mass, momentum and energy. Only a few collision operators are known analytically: the collision operator for non-interacting particles is trivially vanishing since the particles do not interact. Another analytical solution is known if the particles interact as hard spheres and only collide in pairs. The assumption of pairwise collisions is reasonable in the case of a dilute monoatomic gas [6]. In general the collision operator cannot be expressed exactly which is why approximations have to be made to be able to work with the Boltzmann equation.

2.2.2. Introduction to Lattice Boltzmann

The Lattice Boltzmann Method is rooted in the previously discussed kinetic theory of gases in Section 2.2.1 but is not a direct solver for the Boltzmann equation (2.2.4). The statistical description of the particle distributions can be used to recover the solution to the Navier-Stokes Equation which can be shown with the Chapman-Enskog expansion.

In the previous Section 2.2.1 the fundamental population function $f(\vec{x},\vec{\xi},t)$ was discussed which describes the number of particles at the position $\vec{x}$ at the time t which are moving with the velocity $\vec{\xi}$. This quantity is discretized in the Lattice Boltzmann Method on a grid on all three arguments which means that not only space and time but also the velocity is discretized. The resulting fundamental quantity is therefore the *discrete-velocity distribution function* $f_i(\vec{x},t)$. The velocity argument has been moved to the index i which is convenient to do and corresponds to the discrete velocity vector $\vec{c}_i$. The moments of the distribution function in Equation (2.2.3) can also be

written for the discretized population function by adapting to the discretization of the velocities by transforming the integral into a discrete sum

$$\varrho(\vec{x},t) = \sum_i f_i(\vec{x},t) \tag{2.2.5a}$$

$$\varrho(\vec{x},t)\vec{v}(\vec{x},t) = \sum_i \vec{c}_i f_i(\vec{x},t). \tag{2.2.5b}$$

The velocities $\vec{c}_i$ are grouped in a set of velocities $\{\vec{c}_i\}$ which contains all discrete velocities. These velocities are chosen in such a way that each velocity $\vec{c}_i$ exactly reaches the next lattice cell within one unit of time Δt. Therefore, the unit of the velocities is space over time and reads for a cubic grid $\frac{\Delta \mathrm{x}}{\Delta \mathrm{t}}$. This also means that formally multiplying the vector $\vec{c}_i$ with a timestep Δt results in a vector that is the positional offset from one lattice cell to the neighboring one. They are referred to with the symbols $\vec{e}_i$ and define the unit of length Δx. These offset vectors also appear in discretizations like the finite difference discretization and there these vectors are called *stencil* which is the reason this term is sometimes also used to describe the set of velocities. A visual illustration of the vectors is shown in Figure 2.4. The last remaining unit in the set of units is the unit of mass Δm, these units are called lattice units and are convenient to use in the context of Lattice Boltzmann simulations.

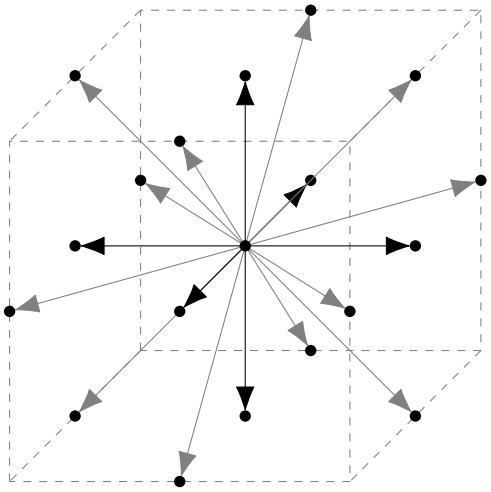


Figure 2.4.: Illustration of the discrete velocities used in the Lattice Boltzmann Method. In this case 19 velocities are used which form the 'D3Q19'-stencil. The velocities pointing to the center of the volume faces are of unit length and drawn in black and the velocities pointing to the edge of the volume faces are drawn in gray. All vectors, including the vector of zero length, are part of the set $\{\vec{c}_i\}$.

The size of the velocity set is dependent on the dimensionality and the accuracy of the discretization. It typically includes velocities to the direct and the diagonal

neighbors. These sets are named with the following scheme DdQq where d is the dimensionality and q is the number of discrete velocities. The sums in this chapter which are indexed with i always run over all discrete velocities which corresponds to the number q in the set of velocities used. Commonly sets used are D2Q9, D3Q19 or D3Q27 where a higher number of velocities typically corresponds to a higher accuracy of the discretization but comes with higher computational cost. The mathematical formulation of the vectors is shown as an example for the set D3Q19 which can be written as

$$\vec{c}_i = \frac{\Delta x}{\Delta t} \begin{cases} (0,0,0) & i = 0 \\ (\pm 1,0,0),(0,\pm 1,0),(0,0,\pm 1) & i = 1,\ldots,6 \\ (\pm 1,\pm 1,0),(\pm 1,0,\pm 1),(0,\pm 1,\pm 1) & i = 7,\ldots,18 \end{cases} . \tag{2.2.6}$$

Each of the velocity sets is associated with a speed of sound c_s that is in the most common cases $c_\mathrm{s} = \frac{1}{\sqrt{3}} \frac{\Delta \mathrm{x}}{\Delta \mathrm{t}}$. This speed of sound is related to the pressure in the fluid with the equation of state

$$p(\vec{x}) = c_\mathrm{s}^2 \varrho(\vec{x}). \tag{2.2.7}$$

The discretization of the Boltzmann equation on the lattice, the *Lattice Boltzmann equation*, reads

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i(\vec{x}, t) + \hat{\Omega}_i(\vec{x}, t). \tag{2.2.8}$$

This equation consists of two parts which can be interpreted individually. The first part of the right hand side of Equation (2.2.8) is known as the *streaming*-step. It describes the movement of the populations f_i in one timestep Δt. The population is propagated from one grid point to the next one where the target points are given by the corresponding velocity $\vec{c}_i$ multiplied with the timestep Δt. An illustration of this operator for a single grid point in two dimensions is shown in image 2.5. This streaming step is applied for each and every grid cell for each timestep.

The second part of Equation (2.2.8) is called the *collision operator* $\hat{\Omega}_i$. As already described in Section 2.2.1 it redistributes the populations f_i of the discrete velocities $\vec{c}_i$ and is supposed to mimic the collision of particles. An illustration of the redistribution is shown in Figure 2.6.

Most of the approximations in this method have to be made for this operator due to the difficulties with the analytical solution which were already discussed in Section 2.2.1.

Many different collision-operators can be found in the literature which typically become more and more complex by providing increasing degrees of freedom which can be tuned individually on different physical parameters. Furthermore they typically also increase the stability of the algorithm. All of the collision operators have in common that they conserve the local density ϱ and the local momentum $\varrho\vec{v}$. In this

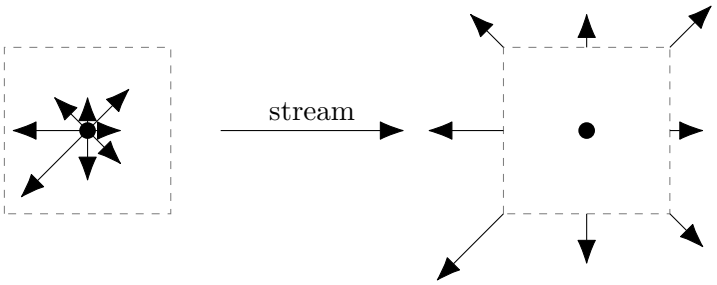


Figure 2.5.: Image showing the streaming-step for a single grid point with the D2Q9 velocity set. The density of the populations are shown as the magnitudes of the vectors and the directions are given by the discretized velocities. The streaming step is moving the populations to the neighboring grid points which is shown on the right.

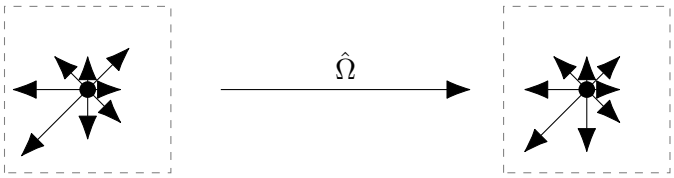


Figure 2.6.: Image showing the collision-step for a single grid point with the D2Q9 velocity set. The magnitudes of the vectors illustrate the density of the corresponding population. The collision operator redistributes the local populations towards the equilibrium while preserving the macroscopic quantities formulated in Equation (2.2.5a) and (2.2.5b).

work we use the most simple collision-operator which is known as the Bhatnagar-Gross-Krook (BGK) operator. It relaxes the local populations towards an equilibrium f_i at a fixed rate given by the *relaxation time* τ. The equation reads [8]

$$\Omega_i(\vec{f}) = -\frac{f_i - f_i^{\mathrm{eq}}}{\tau}\Delta t \tag{2.2.9}$$

and the equilibrium distribution is given by

$$f_i^{\mathrm{eq}}(\vec{x},t) = \omega_i \varrho \left(1 + \frac{\vec{v}\cdot\vec{c}_i}{c_\mathrm{s}^2} + \frac{(\vec{v}\cdot\vec{c}_i)^2}{2c_\mathrm{s}^4} - \frac{\vec{v}\cdot\vec{v}}{2c_\mathrm{s}^2}\right), \tag{2.2.10}$$

where the positional and time dependency of the velocity and density is not expressed explicitly.
The connection between the Navier-Stokes Equation and the Lattice Boltzmann equation can be made by the *Chapman-Enskog expansion* [9]. The expansion itself is tedious and well established in the literature, a well written example can be found in [6, Section 4]. The expansion can be performed for each collision operator and reveals the relations between input parameters of the collision operator and physical,

macroscopic quantities in the Navier-Stokes Equation. The BGK collision operator only has one free parameter, the relaxation time τ, which is related to the kinematic shear viscosity ν

$$\nu = c_\mathrm{s}^2 \left(\tau - \frac{\Delta t}{2} \right). \tag{2.2.11}$$

More complex collision operators with more parameters typically have more degrees of freedom which can be tuned individually to match physical quantities.

The propagation of the populations f_i is performed with the two introduced operators which are applied in succession. First the populations are relaxed with the collision operator $\hat{\Omega}_i$. This application of the operator can be written as

$$f_i^*(\vec{x}, t) = f_i(\vec{x}, t) \left(1 - \frac{\Delta t}{\tau} \right) + f_i^\mathrm{eq}(\vec{x}, t) \frac{\Delta t}{\tau}, \tag{2.2.12}$$

where $f_i^*(\vec{x}, t)$ are the populations after the application of the collision operator. Afterwards the streaming step is performed which can be expressed with the updated populations as

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i^*(\vec{x}, t). \tag{2.2.13}$$

The application of the two operators are repeated, one application of both operators is what is known as a *time step*. Each of the introduced operators is only dependent on local values which is trivial to see for the streaming operator because only neighboring populations need to be known. For the collision operator this can be seen from the definition of the equilibrium distribution (2.2.10) which only depends on the local density ϱ (2.2.5a) and the local velocity $\vec{v}$ (2.2.5b) which both only depend on the local populations f_i. This locality feature is the reason that the Lattice Boltzmann Method is well suited for parallelized computing.

2.2.3. Forces in the Lattice Boltzmann Method

The previous derivation of the Lattice Boltzmann equation in Section 2.2.2 is the force-free Lattice Boltzmann equation. There are different ways known to include external body forces to the Lattice Boltzmann equation, in this section one possible way to include external forces is shown which is used in this work.

The chosen scheme to include forces is the *Guo forcing* scheme [10] which is shown for the previously introduced BGK collision operator (2.2.9). It is a forcing scheme with a second-order accuracy for both space and time that also has been developed for more complex collision operators like the Multi-Relaxation-Time collision operator, the derivation for it can be found in [11].
The inclusion of the force is done with a source term in the Lattice Boltzmann equation which is added to the collision operator

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) - f_i(\vec{x}, t) = \hat{\Omega}_i(\vec{x}, t) + \hat{F}_i(\vec{x}, t) \Delta t \tag{2.2.14}$$

where the force operator $\hat{F}_i(\vec{x}, t)$ is defined as

$$\hat{F}_i = \left(1 - \frac{1}{2\tau}\right) \omega_i \left[\frac{\vec{c}_i - \vec{v}}{c_\mathrm{s}^2} + \frac{(\vec{c}_i \cdot \vec{v})}{c_\mathrm{s}^4}\vec{c}_i\right] \cdot \vec{F}. \tag{2.2.15}$$

In this equation ω_i are weights associated with the lattice velocities $\vec{c}_i$. These weights are chosen in such a way that the lattice velocity obeys a sufficient amount of rotational lattice isotropy. This can be formulated with the following equations [6, p. 85, Equation 3.60]

$$\begin{aligned} \sum_i \omega_i &= 1 & \sum_i \omega_i c_{i\alpha} &= 0 \\ \sum_i \omega_i c_{i\alpha} c_{i\beta} &= c_\mathrm{s}^2 \delta_{\alpha\beta} & \sum_i \omega_i c_{i\alpha} c_{i\beta} c_{i\gamma} &= 0 \\ \sum_i \omega_i c_{i\alpha} c_{i\beta} c_{i\gamma} c_{i\delta} &= c_\mathrm{s}^4 \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right) & \sum_i \omega_i c_{i\alpha} c_{i\beta} c_{i\gamma} c_{i\delta} c_{i\epsilon} &= 0. \end{aligned} \tag{2.2.16}$$

The solution for the weights can be found in [6, p. 88, Table 3.1] which are all given for the most common speed of sound of $c_\mathrm{s} = \frac{1}{\sqrt{3}}\frac{\Delta \mathrm{x}}{\Delta \mathrm{t}}$.
In the forcing scheme also the fluid velocity $\vec{v}$ for both the equilibrium populations f_i^{eq} in Equation (2.2.10) as well as the macroscopic velocity is extended by the acting body force and reads

$$\vec{v} = \frac{1}{\varrho}\sum_i \vec{c}_i f_i + \frac{\vec{F}\Delta t}{2\varrho}. \tag{2.2.17}$$

This redefinition ensures the previously mentioned second order time and space accuracy, which arises due to the way the streaming and collision operations are performed. The mathematical form of the force operator $\hat{F}$ is determined by a power series expansion of the force in the particle velocity [10].
The calculations of this additional source operator in the algorithm is typically included in the calculations of the collision step, which was introduced in Section 2.2.2.

2.2.4. Boundary Conditions

Boundary conditions have a crucial influence on the solution of differential equations which is also the reason of the importance to the simulations. There are a wide variety of boundary conditions known in the literature, in this work we will only discuss two of the most used ones.

Periodic Boundary Condition

An often used boundary condition are the periodic boundary conditions. With this boundary condition the simulation domain is periodically repeated in the desired directions. The fluid flow leaving the simulation domain on one end enters again on

the opposite side of the domain. This is especially useful to simulate bulk systems which can be approximated by repeating a small part of the simulation domain periodically in one direction, which is a common scenario in the fluid simulation. The term "approximated" is used in this case because only in the mathematic description a system can be infinitely large, when comparing to laboratory experiments systems are always of finite size. Nevertheless this approximation is often accurate enough to capture the desired effects which have to appear on length scales smaller than the system size. Another reason for the frequent use of this boundary condition is that it allows to reduce the size of the simulation domain which significantly decreases the computation time because of the lower amount of operations which have to be performed for each timestep. Technically the periodic boundary conditions are typically implemented with ghost layers which are additional cells on the edges of the simulation domain. These additional cells can be used to copy the first layer of the periodic images to these cells which allows the "implementation" of periodic boundary conditions with a simple copy operation and does not need a change in the streaming/collision kernels.

No-Slip Boundary Condition

The *No-Slip* boundary condition is applied when the fluid on the surface is supposed to have zero velocity. Mathematically this translates to a Dirichlet boundary condition of zero velocity at the wall:

$$\vec{v}|_{\vec{x}\in\{\text{wall}\}} = \vec{0}. \tag{2.2.18}$$

The approach taken in this work to fulfill this zero surface velocity condition is known as the *half-way bounce back* method. In this method the populations f_i which correspond to lattice velocities $\vec{c}_i$ pointing into the wall are reflected back to the lattice node they came from during the streaming step with the lattice velocity -$\vec{c}_i$. An illustration of this process is shown in Figure 2.7.

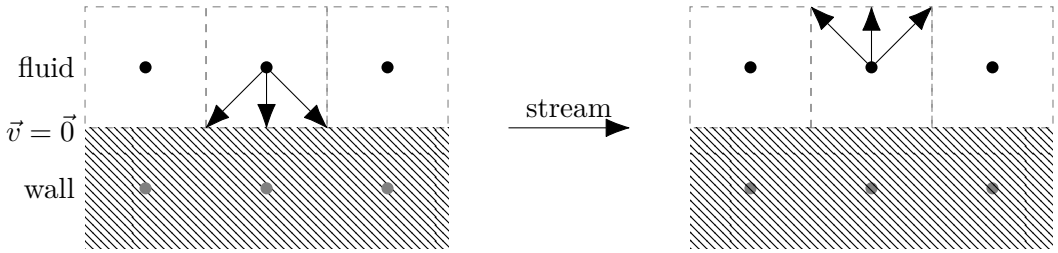


Figure 2.7.: Schematic illustration of the half-way bounce back implementation for a no-slip boundary condition. On the left the pre-streaming situation is shown and on the right the post-streaming situation. The populations of lattice velocities $\vec{c}_i$ that are pointing into the wall are reflected back during the streaming step.

This approach of reflecting the populations pointing into the wall formally results in the zero-velocity halfway in between the boundary lattice point and the fluid lattice point as it is also illustrated in Figure 2.7. Due to the reflecting nature of the boundary handling the total density in the system stays conserved since the value of the population is only transferred but the total momentum of the fluid is not conserved because the direction of the velocity changes. The missing momentum in the fluid is absorbed by the boundary cell, which is fixed in space in the simulations causing the loss of momentum.

2.3. Multiphase Fluid Model: Shan-Chen

The previous introduction to the Lattice Boltzmann Method in Section 2.2.2 is only capable of simulating fluids of a single phase. To simulate fluids of multiple phases or components the method has to be extended. In this work the Shan-Chen method [12, 13, 14] is used which was proposed in 1993. This model introduces a pseudopotential resulting in an external force on the fluids. Two variations of the model can be used which are known as the *multiphase* and the *multicomponent* variations which differ in the amount of distinct fluid phases which have to be simulated.

2.3.1. Introduction

The basis of the Shan-Chen multiphase models is the definition of the pseudopotential, which has the same mathematical form for both the mentioned versions. The definition of the force reads

$$\vec{F}_{\sigma,\sigma'}(\vec{x},\vec{x}') = -G_{\sigma\sigma'}(\vec{x},\vec{x}')\psi_\sigma(\vec{x})\psi_{\sigma'}(\vec{x}')\left(\vec{x}'-\vec{x}\right). \tag{2.3.1}$$

where the subscript σ denotes the fluid component, $G_{\sigma\sigma'}(\vec{x},\vec{x}')$ is an interaction function, sometimes also called Green's function in the literature, and $\psi_\sigma(\vec{x})$ defines the effective fluid density of component σ at the position $\vec{x}$ which will be discussed later, for the moment it can be seen as the fluid density. In the literature the interaction function $G_{\sigma\sigma'}$ is typically defined as

$$G_{\sigma\sigma'}(\vec{x},\vec{x}') := G_{\sigma\sigma'}(\left|\vec{x}-\vec{x}'\right|) = \begin{cases} G_{\sigma\sigma'}\omega_i c_\mathrm{s}^2 & |\vec{x}-\vec{x}'| = \vec{e}_i \\ 0 & \text{else} \end{cases}, \tag{2.3.2}$$

which incorporates only the nearest-neighbors into the potential calculation which are described by the lattice vectors $\vec{e}_i$ and the corresponding weights ω_i. The number of neighbors included in the calculation is in general a trade off between computational cost and discretization accuracy which also influences the strength of model artifacts. The reason for including only the nearest-neighbors is to keep the locality feature of the Lattice Boltzmann Method which was introduced in Section 2.2.2.

The definition of the interaction function shown in Equation (2.3.2) leaves an interaction constant $G_{\sigma\sigma'}$ which is dependent on the interacting fluid components and can be either positive or negative for a repulsive or attractive fluid interaction,

respectively. The total force on fluid component σ at the lattice position $\vec{x}$ is then given by the summation over all lattice positions $\vec{x}'$ and all fluid components σ'. By substituting in Equation (2.3.2) the total force reads

$$\vec{F}_\sigma(\vec{x}) = -\psi_\sigma(\vec{x}) \sum_{\sigma'} G_{\sigma\sigma'} c_\mathrm{s}^2 \sum_i \omega_i \psi_{\sigma'}(\vec{x} + \vec{e}_i)\vec{e}_i. \tag{2.3.3}$$

A sharp look at the discretization from Equation (2.3.3) reveals that the formula can be seen as a Taylor-expansion of the term

$$\vec{F}_\sigma(\vec{x}) \approx -\sum_{\sigma'} G_{\sigma\sigma'} c_\mathrm{s}^2 \psi_\sigma(\vec{x}) \vec{\nabla}_{\vec{x}'} \psi_{\sigma'}(\vec{x}') \tag{2.3.4}$$

including the first layer of neighbors, which corresponds to a second order expansion. This observation is used later to extend the original model (Section 2.3.5) and reduce some of the drawbacks which come with this model. The presence of the pseudopotential is especially relevant at fluid-fluid interfaces which causes a constant force on the fluids even when in equilibrium. This additional force also enters the Chapman-Enskog expansion and alters the equation of state of the system which now reads [15]

$$p(\vec{x}) = \sum_\sigma c_\mathrm{s}^2 \varrho_\sigma(\vec{x}) + \frac{c_\mathrm{s}^2}{2} \sum_{\sigma,\sigma'} G_{\sigma\sigma'} \psi_\sigma(\vec{x}) \psi_{\sigma'}(\vec{x}). \tag{2.3.5}$$

The first term is the ideal-gas contribution which can also be seen in the equation of state of the classical Lattice Boltzmann Method (Section 2.2.2) and the second contribution is the result of the pseudopotential.

In the case of using the multi-component version of the Shan-Chen method, to keep the second order accuracy in space and time from the Guo forcing scheme, which was introduced in Section 2.2.3, the velocity of the fluid has to be changed to the *barycentric velocity* for both the equilibrium velocity as well as the velocity solving the Navier-Stokes Equation [6, 15]. It is given by

$$\begin{aligned} \vec{v}_\mathrm{b}(\vec{x}, t) &= \frac{1}{\varrho(\vec{x}, t)} \sum_\sigma \left(\sum_i f_{\sigma,i}(\vec{x}, t)\vec{c}_i + \frac{\vec{F}_\sigma(\vec{x}, t)\Delta t}{2} \right) \\ &= \frac{1}{\varrho(\vec{x}, t)} \sum_\sigma \left(\varrho_\sigma(\vec{x}, t)\vec{v}_\sigma(\vec{x}, t) + \frac{\vec{F}_\sigma(\vec{x}, t)\Delta t}{2} \right), \end{aligned} \tag{2.3.6}$$

where f_σ are the populations of the component σ and the density $\varrho_\sigma(\vec{x}, t)$, the density without the subset $\varrho(\vec{x}, t)$ is the total density given by

$$\varrho(\vec{x}, t) = \sum_\sigma \varrho_\sigma(\vec{x}, t). \tag{2.3.7}$$

The force $\vec{F}_\sigma(\vec{x}, t)$ includes all forces on the fluid, the Shan-Chen interaction force as well as external forces. This formula expresses the total fluid velocity as the individual

fluid velocities (2.2.17) corrected by the force acting on them and weighted by their densities.

This force correction of the fluid velocity in the multicomponent model also comes with a pitfall for external forces. Because the total velocity is calculated by the density based average of the fluid velocities, the external forces on the fluids have to also respect this averaging to correctly include the external forces. This means that every external volume force has to be distributed to each fluid phase $\vec{F}_\sigma^{\text{ext}}(\vec{x},t)$ based on the local density. For an external force $\vec{F}^{\text{ext}}(\vec{x},t)$ this can be expressed as

$$\vec{F}_\sigma^{\text{ext}}(\vec{x},t) = \frac{\varrho_\sigma(\vec{x},t)}{\varrho(\vec{x},t)}\vec{F}^{\text{ext}}(\vec{x},t), \tag{2.3.8}$$

where the total density $\varrho(\vec{x},t)$ is again given by Equation (2.3.7). The correctness of this equation can trivially be shown by plugging the definition (2.3.8) in the definition of the barycentric velocity in Equation (2.3.6). The summation of the force contribution over the component σ can be evaluated and the density prefactor of Equation (2.3.8) exactly cancels with the summation by using the definition of the total density (2.3.7). The resulting force correction of the barycentric velocity $\vec{v}_{\text{b}}$ is exactly the applied external force.

2.3.2. Effective Density

The effective density $\psi(\vec{x})$ is fundamentally a function of the density ϱ and has to have the units of a density which are $\frac{\Delta\text{m}}{\Delta\text{x}^3}$ where Δm is the unit of the mass and Δx the distance between lattice nodes. The reason to use this function is that the potential is a pseudopotential which tries to include the immiscible nature of the fluid components. For this purpose the density is a natural choice but the form of the function is not obvious. Therefore there are different choices in the literature, in this work only the two mostly used functions are introduced:

linear function

$$\psi_\sigma(\vec{x}) = \varrho_\sigma(\vec{x}) \tag{2.3.9}$$

The simplest choice for the effective density function is the linear function which is motivated by the simplicity but has the drawback that density fluctuations are directly translated to force fluctuations that can cause stability issues especially when dealing with large density ratios.

exponential function

$$\psi_\sigma(\vec{x}) = \varrho_{0,\sigma}\left(1 - \exp\left(-\frac{\varrho_\sigma(\vec{x})}{\varrho_{0,\sigma}}\right)\right). \tag{2.3.10}$$

The most common choice for the density function is the exponential function which exponentially decays for increasing densities. The value $\varrho_{0,\sigma}$ is a parameter of the model for each fluid component σ. The parameter is typically chosen to be the initial

density of the fluid component. This function is also used in this work and has the advantage over the linear choice to allow larger densities and larger densities ratios to be used. The reason is the saturation of the effective density for large density towards the constant $\varrho_{0,\sigma}$ which also causes the pseudopotential and the forces to saturate which stabilizes the simulations. An illustration of both of the functions is shown in Figure 2.8.

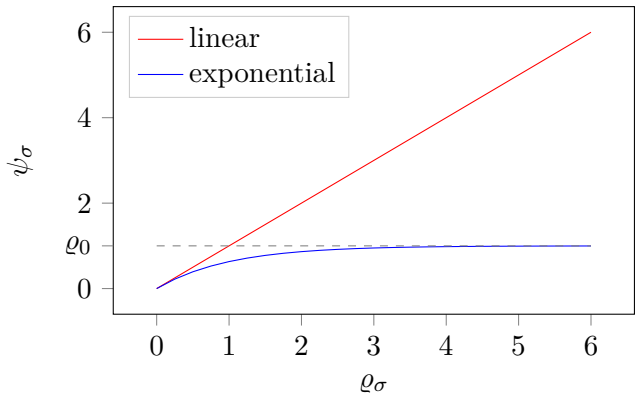


Figure 2.8.: Illustration of the two introduced effective density functions showing the saturation effect of the exponential function which can be helpful in stabilizing the simulation.

In the literature more complex choices for the effective density $\psi(\vec{x})$ are used, especially if specific equations of state are desired. Most of them can be included in the model by using a specific formula for the effective densities. The idea for the derivation of this formula is to take the equation of state (2.3.5) and isolate the effective density function $\psi(\vec{x})$. From this point many equations of state can be achieved by substituting the desired equation of state into the pressure p term. The exact procedure is described and has been analyzed in [16].

2.3.3. Boundary Condition

As the Shan-Chen multicomponent method is a simulation of separate Lattice Boltzmann fluids on the same lattice which are coupled through forces almost all boundary conditions of regular Lattice Boltzmann fluids can be applied. In theory every implementation of a boundary condition should work as long as it does not rely on the specific form of the equation of state of the original Lattice Boltzmann method in Equation (2.2.7) because it is altered in the Shan-Chen model.
In multiphase fluid simulations a popular system is to look at droplets, especially if the interaction of the fluids with the wall is of interest. These systems appear in contact angle simulations where a fluid droplet is placed onto a wall which then forms a stable configuration with a specific contact angle. This angle is often dependent on simulation parameters and therefore can be tuned for different contact angles. The method chosen in this work to simulate specific contact angles follows the same

scheme as the fluid-fluid interaction in the plain Shan-Chen method [17]. To put this into an equation it reads

$$\vec{F}_\sigma(\vec{x}) = -\psi_\sigma(\vec{x}) G_{\sigma\mathrm{W}} \sum_i s(\vec{x} + \vec{e}_i)\vec{e}_i, \tag{2.3.11}$$

where $s(\vec{x})$ is an indicator function which is defined as

$$s(\vec{x}) = \begin{cases} 1, & \vec{x} \in \{\text{wall}\} \\ 0, & \text{else} \end{cases}. \tag{2.3.12}$$

It is 1 whenever the position is defined to be a wall and 0 everywhere else. $G_{\sigma\mathrm{W}}$ is the interaction constant which defines the interaction strength between the fluid component σ and the wall W in the same way as the interaction $G_{\sigma\sigma'}$ between two fluid components. In this definition the sum runs over all interaction neighbors which are conveniently chosen to be the same as for the fluid-fluid interactions of Equation (2.3.3). The interaction constants $G_{\sigma\mathrm{W}}$ define whether one fluid phase becomes more wetting than the others, or all components behave equally if all interaction constants $G_{\sigma\mathrm{W}}$ are equal.

Another identical interpretation of the wall interaction is that we introduce a virtual fluid component which is only living in the cells defined as a wall. This virtual fluid is not propagated but has a density which interacts with the fluid components in the domain. This density is not related to the physical density and is just used as a parameter for the simulation. In the most general case this virtual density does not have to be uniform but in this work the spatial variation of the virtual density was not used.

2.3.4. Phase Field

A commonly used quantity used in two-phase flow systems is the phase field $\chi(\vec{x})$. It is used as a continuous indicator function showing the type of fluid present at a specific position $\vec{x}$. This is especially useful in the case of the two-component simulation because it includes the local densities of both fluid components. Naming the fluid components A and B the phase field is defined as

$$\begin{aligned} \chi(\vec{x}) &= \frac{\frac{\varrho_\mathrm{A}(\vec{x})}{\varrho_{\mathrm{A}0}} - \frac{\varrho_\mathrm{B}(\vec{x})}{\varrho_{\mathrm{B}0}}}{\frac{\varrho_\mathrm{A}(\vec{x})}{\varrho_{\mathrm{A}0}} + \frac{\varrho_\mathrm{B}(\vec{x})}{\varrho_{\mathrm{B}0}}} \\ &= \frac{\varrho_\mathrm{A}(\vec{x})\varrho_{\mathrm{B}0} - \varrho_\mathrm{B}(\vec{x})\varrho_{\mathrm{A}0}}{\varrho_\mathrm{A}(\vec{x})\varrho_{\mathrm{B}0} + \varrho_\mathrm{B}(\vec{x})\varrho_{\mathrm{A}0}} \end{aligned} \tag{2.3.13}$$

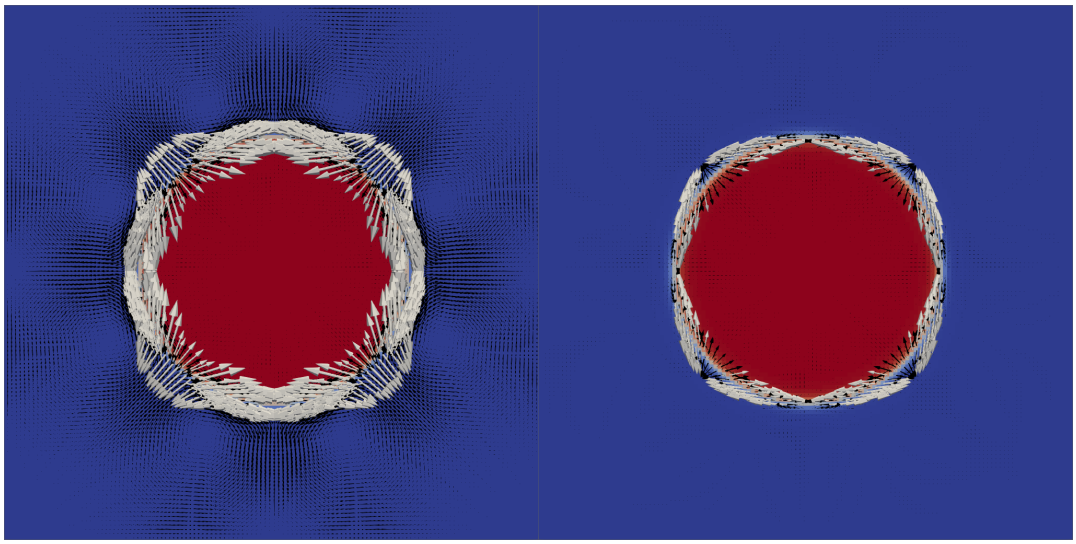
where the subset 0 denotes is the initial density of the fluid phase. In the case where the two fluid phases have the same density the equation reduces to

$$\chi(\vec{x}) = \frac{\varrho_\mathrm{A}(\vec{x}) - \varrho_\mathrm{B}(\vec{x})}{\varrho_\mathrm{A}(\vec{x}) + \varrho_\mathrm{B}(\vec{x})} \tag{2.3.14}$$

which is a function with the range of values between -1 and 1. A value of -1 indicates that only fluid component b is present at that position and a value of 1 the opposite. In the case of Shan-Chen two component fluids these extreme values are not reached due to model limitations but the sign of the phase field can be used as indication of which fluid density is more dominant at a specific position. This is a common way of coloring the simulation domain with the specific fluid component, since it independent on the density values used.

2.3.5. Limitations and Extensions

In the literature the Shan-Chen model is typically described as the most simple model for multiphase flows which allows the inclusion of the model into already existing Lattice Boltzmann implementations. In this section some weaknesses and extensions to the Shan-Chen model are shown, more of them can be found in [14]. All the multiphase models for Lattice Boltzmann mentioned in the literature have their strengths and weaknesses, for this model the most dominant weakness is the magnitude of the *spurious currents*. Spurious currents are nonphysical, model related, fluid currents which appear at the surface of curved fluid interfaces for which an example is shown in Figure 2.9a.
In the case of the Shan-Chen model the reason for these spurious currents can be tracked down to the discretization of the gradient for the pseudopotential interaction force which was shown in Equation (2.3.4). The gradient is discretized with the weights ω_i to posses the highest degree of rotational isotropy possible and is highly dependent on the amount of neighbors included in the interaction. For the inclusion of the first layer of neighboring cells the order of isotropy is four [18] which is illustrated in two dimensions as the orange points in Figure 2.10. Increasing the order of isotropy by including more neighboring cells reduces the magnitude of the spurious currents. An example is shown in Figure 2.10 where the next nearest neighboring points are visualized in two dimensions which are used for the gradient discretization. The result of this discretization is an eighth order isotropic discretization. The corresponding weights can be found in [18] or a more general derivation in [19] and are provided in Table 2.1.
The previously shown set-up for the spurious currents is repeated with the higher order isotropy discretization and can be seen in Figure 2.9b. The image shows a significant reduction of the strength of the spurious currents. It has to be noted that this model artifact of spurious currents is not exclusive to the Shan-Chen model, most of the models found in the literature have this artifact but it is typically less dominant in the other models compared to the Shan-Chen model.



(a) (b)

Figure 2.9.: Simulation snapshots showing the phase field of a droplet as the background color and the velocities as arrows. The system should be free of any fluid movement which makes all shown velocities model artifacts that are known as spurious currents. The magnitude of the white colored arrows are larger than a threshold value of $10^{-3}\,\frac{\Delta x}{\Delta t}$ and the black ones are smaller. The identical set-up is calculated for the nearest-neighbor interaction in (a) as well as for the next-nearest neighbor interaction in (b). The increased number of interacting neighbors significantly reduces the magnitude of the spurious currents in the system.

Table 2.1.: Table of weights to include multiple layers of neighboring cells for the gradient discretization. The argument in the brackets denotes the squared norm of the vector. Different orders of isotropy can be achieved by including different numbers of neighbors.

	isotropy-order	$\omega(1)$	$\omega(2)$	$\omega(3)$	$\omega(4)$	$\omega(5)$	$\omega(6)$	$\omega(8)$
2D	4	1/3	1/12					
	6	4/15	1/10		1/120			
	8	4/21	4/45		1/60	2/315		1/5040
3D	4	1/6	1/12					
	6	2/15	1/15	1/60	1/120			
	8	4/45	1/21	2/105	5/504	1/315	1/630	1/5040

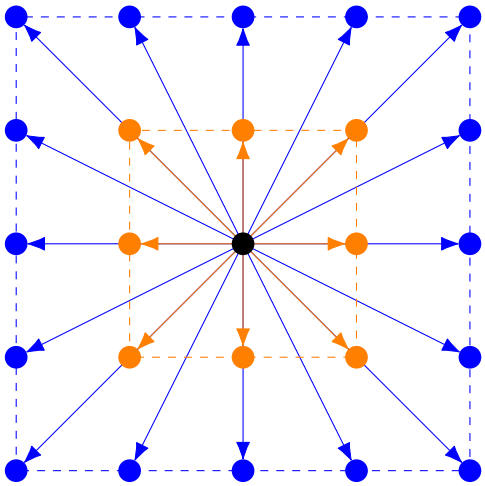


Figure 2.10.: Schematic illustration of the neighbors (orange) and next nearest neighbors (blue) for a two dimensional rectangular grid. The arrows are interpreted as the offsets to the central point. These neighbors are used for the discretization of the gradients for different order of isotropy.

2.4. Electrokinetics

The dynamics of ions can be described on different length and time scales which follow the same pattern as shown in Figure 2.3 ranging from the description of individual ions to the continuum description which is done with concentration and flux fields.

2.4.1. Introduction

In this work the focus is on the continuum level description of ions, which is done with the *Nernst-Planck equation* which extends the *Fick's law of diffusion* [20] to charged particles with the inclusion of electrostatic forces.

Considering a chemical species k which is dissolved in a solvent, the concentration of these species can be described with a scalar field $n_k(\vec{x}, t)$ and the fluxes with a vector field $\vec{j}_k(\vec{x}, t)$. Both fields are connected by the mass conservation equation

$$\partial_t n_k(\vec{x}, t) = -\vec{\nabla} \cdot \vec{j}_k(\vec{x}, t). \tag{2.4.1}$$

The equation relates the change of the concentration in time to the positional change of the fluxes, which can be thought of as the change of the concentration in a specified volume in time is determined by the fluxes passing through the surfaces of the volume. In the case of the ions the fluxes $\vec{j}_k$ have two distinct contributions: the diffusive and the advective flux.

$$\vec{j}_k(\vec{x}, t) = \vec{j}_k^{\text{dif}}(\vec{x}, t) + \vec{j}_k^{\text{adv}}(\vec{x}, t) \tag{2.4.2}$$

The first contribution to the fluxes discussed is the diffusive flux. In the case of uncharged particles, where electrostatics is negligible, this term is typically dominated by the concentration gradient. For charged particles this term also includes the electrostatic forces which are given by the well known gradient of the electrostatic potential $-\vec{\nabla}\Phi(\vec{x}, t)$

$$\vec{j}_k^{\text{dif}}(\vec{x}, t) = -D_k \vec{\nabla} n_k(\vec{x}, t) - \frac{D_k}{k_{\text{B}}T} q_k n_k \vec{\nabla}\Phi(\vec{x}, t), \tag{2.4.3}$$

where D_k is the diffusion coefficient of the species k, $k_{\text{B}}T$ is the thermal energy of the system which is given by the product of the Boltzmann factor and the temperature and q_k is the charge of the ions. The derivation of this equation can be found in [21] which finds an expression for the chemical potential and calculates the gradient of it that results in the thermodynamic driving force given by Equation (2.4.3). This procedure can also be expanded to include different types of forces. For this work the qualitative picture that the diffusive flux drives the ions into their thermodynamic equilibrium is sufficient.

The second contribution to the fluxes is the advective flux contribution, which is the contribution of the solvent of the ions moving with velocity $\vec{v}$ which causes the ions to be dragged along the direction of the solvent velocity. This can be written as

$$\vec{j}_k^{\text{adv}}(\vec{x}, t) = n_k(\vec{x}, t)\vec{v}(\vec{x}, t), \tag{2.4.4}$$

which is the instantaneous displacement of the ion concentration by the solvent velocity. This description of the instantaneous displacement neglects the inertia of the ions which is a reasonable assumption in the limit of the continuous scale because the time scale is significantly larger than the time scale of the ballistic regime of individual particles, which means that the inertia of the ions is negligible.

In the case when at least one chemical species k has a net charge, electrostatics play a significant role in the system. Therefore the well known *Poisson equation* for the electrostatic potential has to be solved, which can be derived from Maxwell's equations. The reason which allows to solve electrostatics instead of electrodynamics is that the contribution of the dynamic effects for the electric and magnetic fields requires ion velocities to be comparable to the speed of light to have a significant contribution. Such velocities can not be captured by this model, which reduces the computational effort for solving the electrostatic potential $\Phi(\vec{x}, t)$ significantly.

$$\vec{\nabla} \cdot \Big(\varepsilon(\vec{x}, t) \vec{\nabla} \Phi(\vec{x}, t) \Big) = -\frac{q(\vec{x}, t)}{\varepsilon_0} \tag{2.4.5}$$

In the most general form the Poisson equation allows for a spatial varying dielectric permittivity $\varepsilon(\vec{x}, t)$, which increases the complexity of solving this equation, ε_0 is the vacuum permittivity and the $q(\vec{x}, t)$ is the charge present at the location $\vec{x}$ at the time t, which is given by the sum of all individual ions present

$$q(\vec{x}, t) = \sum_k q_k n_k(\vec{x}, t). \tag{2.4.6}$$

The more common form of the Poisson equation includes the assumption of a spatial invariant dielectric permittivity ε which simplifies the equation to

$$\Delta \Phi(\vec{x}, t) = -\frac{q(\vec{x}, t)}{\varepsilon \varepsilon_0}. \tag{2.4.7}$$

2.4.2. Limitations

This continuum level description of the ions takes significantly less effort for the computation than solving the equations of motion for each ion individually due to the reduced number of equations which have to be solved but this reduced complexity comes at the cost of assumptions which have to be made. In the following the most important assumptions of the model are mentioned.

First of all the ion model assumes non-interacting ions which means excluded volume effects are neglected. This restricts the possible ion concentration to moderate ones. Furthermore this also restricts the valency to monovalent ions. The reason for this is that with large ion concentrations or higher valency ions correlation effects play a significant role and layering of ions occurs, which will not happen in this mean-field model.

Another limitation is the constant diffusion coefficient D_k and the constant temperature T in the system. The model assumes a constant diffusion coefficient for

the whole domain, which restricts the application of the model and excludes systems with varying diffusion coefficients e.g. multicomponent fluid systems of components with different viscosities.

2.4.3. Lattice Electrokinetics

In the previous Section 2.4.1 the continuous mathematical description of the ion dynamics known as the Nernst-Planck equation was presented. In this section the discretization of the continuous equation will be discussed, which will be shown for the specific electrokinetics equations. The discretization of the ions is chosen to live on the same grid as the fluids discussed in Section 2.2.

The mass conservation Equation (2.4.1) is discretized using the Finite Volume Discretization technique. In this case this it is superior to other discretization methods like the Finite Elements Discretization Method because it respects conservation laws by construction. With this approach each point on the discretization grid is assigned a volume and the equation, which is supposed to be discretized, is integrated over this volume. The divergence terms are then converted to surface integrals with the help of Gauss' divergence theorem. In the case of the ion dynamics this results in the equation

$$\begin{aligned}\partial_t \int_V n_k(\vec{x},t)\mathrm{d}V &= -\int_V \vec{\nabla}\cdot\vec{j}_k(\vec{x},t)\mathrm{d}V \\ &= -\int_{\partial V}\vec{j}_k(\vec{x},t)\cdot\vec{n}\mathrm{d}A, \end{aligned} \tag{2.4.8}$$

where $\vec{n}$ is the normal vector of the surfaces, V is the volume to be integrated over, ∂V denotes the surface of the volume V and $\mathrm{d}A$ is the infinitesimal area element of the surfaces. With this transformation the fluxes $\vec{j}_k$ now have to be evaluated at the surfaces of the volumes. The right hand side of the equation can typically be calculated analytically, which is especially trivial for a cubic volume which is used in this work. To use this equation it has to be discretized also in time which is done with a discrete time step Δt and a simple finite difference expansion of the partial time derivative. The integrated concentration, the number of particles within the volume element, is defined as $N_k(\vec{x},t) = \int_V n_k(\vec{x},t)\mathrm{d}V$

$$\frac{N_k(\vec{x},t+\Delta t) - N_k(\vec{x},t)}{\Delta t} = -\sum_i A_i\vec{e}_i\cdot\vec{j}_k(\vec{x}+\frac{\vec{e}_i}{2},t) \tag{2.4.9}$$

where A_i is the area associated to one flux and depends on the amount of fluxes used for the discretization. In the case the number of fluxes per cell corresponds to the number of dimensions this is simply the surface area of the grid cell, for the remaining cases these areas are chosen to recover the correct diffusion coefficient D_k when calculating the mean-squared displacement of the concentration [22]. The vectors $\vec{e}_i$ are the lattice vectors to the neighboring cells which also denote the direction of the flux vectors associated with the index i. The number of lattice vectors used for

the discretization is again a trade-of between accuracy and computational cost which was already discussed for a similar problem in Section 2.2.

One crucial assumption for this model is that the particles are distributed homogeneously within the volume element. This assumption is necessary to allow for the discretization of the fluxes on the surfaces. Furthermore this allows to calculate the number of particles within a volume element by a simple multiplication $N_k(\vec{x}, t) = n_k(\vec{x}, t)\Delta x^3$. Substituting this into the discretization in Equation (2.4.9) results in the expression

$$n_k(\vec{x}, t + \Delta t) = n_k(\vec{x}, t) - \frac{\Delta t}{\Delta x^3} \sum_i A_i \vec{e}_i \cdot \vec{j}_k(\vec{x} + \frac{\vec{e}_i}{2}, t). \tag{2.4.10}$$

In Equation (2.4.8) the evaluation of the fluxes on the surfaces of the volumes are introduced which show up in the discretized equation (2.4.9) and (2.4.10) by the positional half shift $\frac{\vec{e}_i}{2}$ by the lattice vectors which is also known as the *staggered* access to a grid. These shifts are illustrated in Figure 2.11 for a two dimensional square grid where the black dots are the centers of the cells which are the positions the density is defined at and the red crosses are the positions of the fluxes on the surfaces.

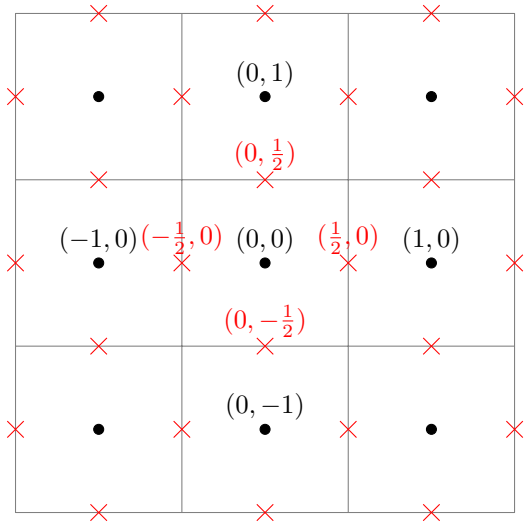


Figure 2.11.: Two dimensional square grid where the density fields are defined at the center of the cells which are shown as black dots. The red crosses are showing the shifted positions on the surfaces of the cells for the fluxes according to the finite volume discretization scheme.

The fluxes on the surfaces are typically not trivially accessible because the densities are only known at center of the cells which enforces the use an interpolation scheme. In this work the interpolation of the scalar fields and the discretization of the gradients on the surfaces is done with a finite difference approach where the hat denotes the normalized vector.

$$n(\vec{x}+\frac{\vec{e}_i}{2}) = \frac{n(\vec{x}+\vec{e}_i)+n(\vec{x})}{2} \tag{2.4.11}$$

$$(\vec{\nabla}n)(\vec{x}+\frac{\vec{e}_i}{2}) = \frac{n(\vec{x}+\vec{e}_i)-n(\vec{x})}{|\vec{e}_i|}\hat{\vec{e}}_i \tag{2.4.12}$$

These transformations are applied to the diffusive flux equation (2.4.3) for all gradients and the interpolation for the density field n.

$$\vec{j}_k^{\text{dif}}(\vec{x},t) = -D_k\vec{\nabla}n_k(\vec{x},t) - \frac{D_k}{k_\text{B}T}q_k(\vec{x},t)n_k(\vec{x},t)\vec{\nabla}\Phi(\vec{x},t)$$

The transformed equation therefore reads

$$\begin{aligned}\vec{j}_k^{\text{dif}}(\vec{x}+\frac{\vec{e}_i}{2},t) =& D_k\frac{n_k(\vec{x}+\vec{e}_i,t)-n_k(\vec{x},t)}{|\vec{e}_i|}\hat{\vec{e}}_i \\ &+ \frac{D_k}{k_\text{B}T}q_k\frac{n_k(\vec{x}+\vec{e}_i,t)+n_k(\vec{x},t)}{2}\frac{\Phi(\vec{x}+\vec{e}_i,t)-\Phi(\vec{x},t)}{|\vec{e}_i|}\hat{\vec{e}}_i.\end{aligned} \tag{2.4.13}$$

The discretization of the advective flux shown in Equation (2.4.4) is done with a method called *Volume of Fluid* [23]. In this method the volume element is virtually displaced by the velocity vector $\vec{v}$ as it is shown in Figure 2.12.

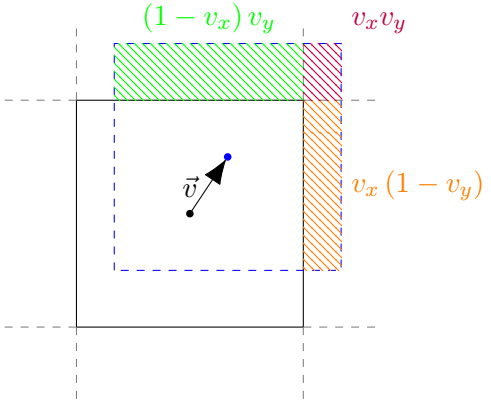


Figure 2.12.: Schematic illustration of the volume of fluid discretization in two dimensions which is used for the discretization of the advection. The volume cell is displaced by the velocity vector $\vec{v}$. The overlapping areas, shown with diagonal color stripes, to neighboring volume cells are added to the corresponding fluxes.

Because of the virtual displacement the volume element overlaps with neighboring volumes. The fluxes pointing to the volume elements which have an overlapping volume are extended by an additional contribution given by the product of the local species concentration n_k and the overlapping volume. Using the definitions

$$v_x(\vec{x}) := \vec{v}(\vec{x}) \cdot \vec{e}_x \frac{\Delta t}{\Delta x} \qquad v_y(\vec{x}) := \vec{v}(\vec{x}) \cdot \vec{e}_y \frac{\Delta t}{\Delta x} \tag{2.4.14}$$

the corresponding mathematical expressions for the two dimensional case are given by

$$\begin{aligned}
\vec{j}_k^{\text{adv}}(\vec{x} \pm \frac{\vec{e}_x}{2}) &= n_k(\vec{x}) v_x(\vec{x}) \left(1 - v_y(\vec{x})\right) \hat{\vec{e}}_x \frac{\Delta x}{\Delta t} \begin{cases} 1 & v_x(\vec{x}) \gtrless 0 \\ 0 & \text{else} \end{cases} \\
\vec{j}_k^{\text{adv}}(\vec{x} \pm \frac{\vec{e}_y}{2}) &= n_k(\vec{x}) v_y(\vec{x}) \left(1 - v_x(\vec{x})\right) \hat{\vec{e}}_y \frac{\Delta x}{\Delta t} \begin{cases} 1 & v_y(\vec{x}) \gtrless 0 \\ 0 & \text{else} \end{cases} \\
\vec{j}_k^{\text{adv}}(\vec{x} + \frac{\vec{e}_x}{2} \pm \frac{\vec{e}_y}{2}) &= n_k(\vec{x}) \left|v_x(\vec{x}) v_y(\vec{x})\right| \frac{\hat{\vec{e}}_x \pm \hat{\vec{e}}_y}{\sqrt{2}} \frac{\Delta x}{\Delta t} \begin{cases} 1 & v_x(\vec{x}) > 0 \wedge v_y(\vec{x}) \gtrless 0 \\ 0 & \text{else} \end{cases} \\
\vec{j}_k^{\text{adv}}(\vec{x} - \frac{\vec{e}_x}{2} \pm \frac{\vec{e}_y}{2}) &= n_k(\vec{x}) \left|v_x(\vec{x}) v_y(\vec{x})\right| \frac{-\hat{\vec{e}}_x \pm \hat{\vec{e}}_y}{\sqrt{2}} \frac{\Delta x}{\Delta t} \begin{cases} 1 & v_x(\vec{x}) < 0 \wedge v_y(\vec{x}) \gtrless 0 \\ 0 & \text{else} \end{cases}.
\end{aligned} \tag{2.4.15}$$

The extension towards three dimensions can easily be done with some writing effort due to the additional factors according to the overlaps in the z-direction.

The discretization approach taken has the benefit compared to the finite element discretization that it is mass and momentum conserving by construction. The reason for this is that the flux leaving one volume element is the incoming flux to an adjacent volume element, which insures that no mass can be lost. The downside is that the fluxes $\vec{j}_k$ have to be known on the surfaces of the volume elements for which it is typically necessary to use an interpolation scheme.

2.4.4. Boundary Conditions

For the lattice electrokinetics based algorithm two boundary conditions for the density and one for the flux are used. The periodic boundary condition was already described in Section 2.2.4 and is the most used boundary condition in this work. Another boundary condition is the constant density boundary condition which can trivially be implemented by forcing the density of a specific cell $\vec{x}_0$ to a specific value $N_0 \in \mathbb{R}$ in each iteration

$$n_k(\vec{x}_0) = N_0. \tag{2.4.16}$$

For the flux the only boundary condition necessary is the No-Flux boundary condition which is used whenever a cell is not allowed to have fluxes leaving or entering the cell. This boundary condition is especially useful when there is a wall inside the domain where no particle densities are allowed to enter the cell. In the discretization this

can be achieved by forcing a specific flux pointing into and out of the cell value to vanish. Assuming $\vec{x}_0$ is the boundary cell in the domain this can be written as

$$\vec{j}_k(\vec{x}_0 \pm \frac{\vec{e}_i}{2}) = \vec{0}. \tag{2.4.17}$$

2.4.5. Electrostatic Solver

In Section 2.4 the ion dynamics has been introduced where also the charged nature of the ions has been mentioned and the contribution of the gradient of the electrostatic potential to the diffusive flux of the ions has been discussed. The respective Poisson equation for the electrostatic potential was shown in Equation (2.4.5), in this section the focus is on how the discretization of this equation is done and what techniques for solving are used. For simplicity the time dependency of the electrostatic potential is not expressed explicitly.

The discretization of the Poisson equation is done on the same cubic grid as with all the previous models, the gradients are discretized with the well known central finite difference scheme with the direct neighbors which corresponds to a second order Taylor expansion. The discretization of the derivatives for a scalar field $n(x)$ with equidistant points Δx apart reads

$$\begin{aligned} \partial_x n(x) &\approx \frac{n(x+\Delta x) - n(x-\Delta x)}{2\Delta x} \\ \partial_x^2 n(x) &\approx \frac{n(x+\Delta x) + n(x-\Delta x) - 2n(x)}{\Delta x^2}. \end{aligned} \tag{2.4.18}$$

The continuous equation for spatial invariant dielectric permittivity reads

$$\Delta\Phi(\vec{x}) = -\frac{q(\vec{x})}{\varepsilon\varepsilon_0} \tag{2.4.19}$$

which is discretized using the notation $\vec{e}_i$ to indicate the vector pointing to the next neighbor in the i-direction. The equation then reads

$$\sum_{\vec{e}_i \in \{\vec{e}_x, \vec{e}_y, \vec{e}_z\}} \frac{2\Phi(\vec{x}) - \Phi(\vec{x}+\vec{e}_i) - \Phi(\vec{x}-\vec{e}_i)}{|\vec{e}_i|^2} = \frac{q(\vec{x})}{\varepsilon\varepsilon_0}. \tag{2.4.20}$$

Fourier Transformation based Poisson Solver

The first technique for solving the discretized Poisson equation is based on the Fast Fourier Transformation (FFT). The discretized equation is transformed to the Fourier space where shifts in real space are transformed to a multiplication with an exponential factor in Fourier space.

$$\begin{aligned} \mathcal{F}\{n(x)\} &= \hat{n}(k) \\ \mathcal{F}\{n(x+e_i)\} &= e^{-ie_i k}\hat{n}(k) \end{aligned} \tag{2.4.21}$$

This relation can be used to solve the discretized version of the Poisson equation (2.4.20) with the spatial invariant permittivity directly in Fourier space. The exact derivation of the necessary Greens function in Fourier space for a cubic lattice can be found in [21]. The function evaluations of $\Phi(\vec{x}+\vec{e}_i)$ results in pairs of exponential factors in Fourier space, which can be combined using the exponential definition of the cosine function

$$\cos(x) = \frac{e^{ix} + e^{-ix}}{2}. \tag{2.4.22}$$

From the discrete Fourier transformation it is known that the basis in Fourier space is orthogonal which means that the Fourier transformed discretized relation (2.4.20) has to be true not just for the total sum but for each term on its own, which can be written as

$$\begin{aligned} \hat{\Phi}(\vec{k}) &= \frac{-\hat{q}(\vec{k})}{2\varepsilon\varepsilon_0 \left(\cos(\frac{2\pi k_x}{N_x}) + \cos(\frac{2\pi k_y}{N_y}) + \cos(\frac{2\pi k_z}{N_z}) - 3\right)}. \\ \hat{\Phi}(\vec{0}) &= 0 \end{aligned} \tag{2.4.23}$$

The vanishing electrostatic potential at $\vec{k} = \vec{0}$ enforces charge neutrality in the system which is necessary because otherwise the electrostatic potential would diverge because of the periodic boundary conditions. Excluding the Fourier transformed charge density from Equation (2.4.23) the remaining factor is a constant which only depends on the systems dimensions and therefore can be precomputed and reused for every iteration. The overall algorithm therefore only involves the Fourier transformation of the charges, a simple pointwise multiplication, and the inverse Fourier transformation. In total this produces a scaling of $\mathcal{O}\left(N(2\log(N)+1)\right)$ for the computation, where N is the number of lattice cells.

Successive Over-Relaxation based Poisson Solver

The second technique used in this work is the successive over-relaxation method [24], which is an iterative scheme for solving linear systems of equations, and is closely related to the Gauss-Seidel method. The description of the method is again done with the discretized Poisson equation (2.4.19). The idea is to start with an initial guess for the solution and iteratively update the guess towards a solution of the problem. In the case of the Poisson equation one can define a residual of the discretization (2.4.20) which reads for a cubic grid with side length Δx

$$R^i(\vec{x}) = -\Phi^i(\vec{x}) + \frac{1}{6}\left(\sum_{\vec{e}_i \in \{\vec{e}_x, \vec{e}_y, \vec{e}_z\}} \Phi^i(\vec{x}+\vec{e}_i, t) + \Phi^i(\vec{x}-\vec{e}_i, t) + \frac{q(\vec{x})\Delta x^2}{\varepsilon\varepsilon_0}\right). \tag{2.4.24}$$

With the residual definition the initial guess of the electrostatic potential is iteratively updated with the calculated residual which is weighted using the factor $\omega \in (0, 2)$.

This additional weighting typically allows for a faster convergence when choosing values $\omega > 1$ or a slower convergence for $\omega < 1$ to prevent stability issues in the simulation. Typical values used in this work are $\omega \approx 1.9$ for a fast convergence. The equation reads

$$\Phi^{i+1}(\vec{x}) = \Phi^{i}(\vec{x}) + \omega R^{i}(\vec{x}) \tag{2.4.25}$$

$$= \Phi^{i}(\vec{x})\,(1-\omega) + \frac{\omega}{6}\left(\sum_{\vec{e}_i \in \{\vec{e}_x, \vec{e}_y, \vec{e}_z\}} \Phi^{i}(\vec{x}+\vec{e}_i) + \Phi^{i}(\vec{x}-\vec{e}_i) + \frac{q(\vec{x})\Delta x^2}{\varepsilon\varepsilon_0}\right). \tag{2.4.26}$$

where the superscript i denotes the iteration and for the starting point the last calculated electrostatic potential $\Phi^0(\vec{x}) = \Phi(\vec{x}, t - \Delta t)$ is used. The potential is updated using a red/black coloring of the domain which is shown in Figure 2.13.

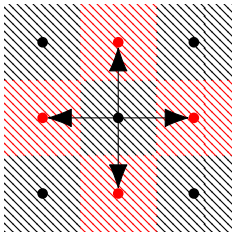


Figure 2.13.: Red and black coloring of the domain cells which are used as indicators for the iterations in the SOR method for solving the electrostatic Poisson equation.

This approach can be taken because the calculation of the residual (2.4.24) only includes the value for the electrostatic potential of the direct neighbors, which is shown with the arrows which are all pointing from each black cell to only red cells and vice versa. Therefore it is possible to update all the black colored nodes in parallel. For the red cells the already updated potential from the black colored nodes can be used which can again be done in parallel. Therefore when starting with a black iteration the following red iteration can use the already updated values from the black iteration which accelerates the convergence of the algorithm. Due to the iterative nature of the algorithm a convergence criterion has to be used. Different convergence criteria are possible, in this work the absolute sum of all residuals has to be below some threshold value which can be written as

$$\sum_{\vec{x}} \left| R^{i}(\vec{x}) \right| < \text{max error}. \tag{2.4.27}$$

The iterative approach has the possibility to include all kinds of different boundary conditions to the algorithm with relatively small effort. Most of the boundary conditions have to be enforced with every iteration of the algorithm, for example a

Dirichlet boundary condition which forces the electrostatic potential Φ to a specific value. This can be done at any specific point on the grid by fixing the value of the desired cell to a specific value which was not possible with the previously discussed Fourier transformation based method in Section 2.4.5. This flexibility comes with a diminished scaling of the computation time of $\mathcal{O}(N^{\frac{3}{2}})$ compared to the Fourier transformation method. For this the reason the Fourier transformation method is used whenever possible.

Non-uniform dielectric permittivity

The Poisson equation (2.4.19) is only valid for a spatially uniform dielectric permittivity, which is not necessarily the case in the context of multicomponent fluids where different fluid components can have a different dielectric permittivity which can be expressed as a spatially varying dielectric permittivity $\varepsilon(\vec{x})$. The corresponding Poisson equation reads

$$\begin{aligned} \vec{\nabla} \cdot \left(\varepsilon(\vec{x}) \vec{\nabla} \Phi(\vec{x}) \right) &= -\frac{q(\vec{x})}{\varepsilon_0} \\ \Delta \Phi(\vec{x}) &= -\frac{q(\vec{x})}{\varepsilon(\vec{x})\varepsilon_0} - \frac{\vec{\nabla}\varepsilon(\vec{x}) \cdot \vec{\nabla}\Phi(\vec{x})}{\varepsilon(\vec{x})}. \end{aligned} \tag{2.4.28}$$

This more complex equation can not be solved trivially using the described methods which is why the additional term on the right hand side is treated as a polarization charge and solved iteratively by using the calculated electrostatic potential from the last iteration [25]. In this way the previously discussed solvers can be reused but have to solve the equation multiple times for a single timestep. The additional term is discretized as described previously by using the second order central finite differences scheme which results in

$$\frac{\vec{\nabla}\varepsilon(\vec{x}) \cdot \vec{\nabla}\Phi(\vec{x})}{\varepsilon(\vec{x})} \approx \sum_{\vec{e}_i \in \{\vec{e}_x, \vec{e}_y, \vec{e}_z\}} \frac{\left(\varepsilon(\vec{x} + \vec{e}_i) - \varepsilon(\vec{x} - \vec{e}_i)\right) \left(\Phi(\vec{x} + \vec{e}_i) - \Phi(\vec{x} - \vec{e}_i)\right)}{4\varepsilon(\vec{x})\Delta x^2}. \tag{2.4.29}$$

The iterative update of the polarization charge is stopped with the same criteria as described in equation (2.4.27). This approach for solving the generalized Poisson equation is similar to approaches taken in Molecular Dynamics simulations [26] and allows the reuse of previous implementations of regular Poisson solvers. The downside of the approach is the computational cost of the repeated solving of the regular Poisson equation which is the most costly operation in the context of this work anyways.

2.5. Couplings between Fluid and Ions

The models for the fluid (Section 2.2.2) and the ions (Section 2.4.3) are both discretized on the very same cubic grid. The discretization is performed on the same grid to reduce the interpolations necessary if couplings are introduced between the methods which are discussed in this section.

Advective Coupling

The first coupling was implicitly mentioned in Section 2.4.3 when discussing the discretization of the advection term for the ions. In this discretization the volume cell is virtually displaced by the distance the fluid would move in a single timestep given by the product of the velocity vector $\vec{v}$ and the timestep Δt. The resulting overlapping volumes to the neighboring cells determines the fluxes to the respective volumes. This velocity vector is calculated for each volume cell individually by solving the Navier-Stokes Equation using the Lattice Boltzmann Method. This describes the first coupling between the methods because of the advection.

Friction Coupling

The second coupling between the ions and the fluid and is induced by the ion movement. The movement of the ions, which are expressed as fluxes, drags along the fluid they are solvated in due to the friction between the solute and the solvent. The friction applies only for velocity differences between the solute and the solvent and causes the velocities between the two to become the same. In Equation (2.4.2) the two contributions to the ion fluxes are shown where the advective contribution consists of ions moving with the solvent velocity. The advective flux therefore does not contribute to the friction coupling and only the diffusive flux couples to the fluid, which reads

$$\begin{aligned} \vec{f}(\vec{x}) &= \gamma \left(\vec{j}_{\text{ions}}(\vec{x}) - \varrho_{\text{ions}} \vec{v}_{\text{fluid}}(\vec{x}) \right) \\ &= \gamma \vec{j}^{\text{dif}}_{\text{ions}}(\vec{x}), \end{aligned} \tag{2.5.1}$$

where the friction coefficient is defined as $\gamma = \frac{k_{\text{B}}T}{D}$, $\vec{v}_{\text{fluid}}$ is the fluid velocity and $\vec{j}_{\text{ions}}$ is the ion flux. In the case where multiple ion species are present it is possible that the individual ion species k have different rates of diffusion D_k and therefore different friction coupling constants. The friction coupling force density in this case can be expressed as

$$\vec{f}(\vec{x}) = k_{\text{B}}T \sum_k \frac{\vec{j}^{\text{dif}}_k(\vec{x})}{D_k}. \tag{2.5.2}$$

In Section 2.4.3 the discretization of the fluxes on the surface of the volume elements is discussed. Since the friction coupling forces have to be known in the center of

the volume elements there are multiple approaches which can be taken. The first approach, which is also the more correct one, is to discretize the friction coupling force on the center of the volume element. This approach calculates the diffusive ion fluxes once for the surfaces and once for the center of the volume elements. The second approach is to interpolate the already evaluated fluxes from the surfaces back onto the center. The equation for this interpolation reads

$$\vec{f}(\vec{x}) = k_{\mathrm{B}}T \sum_{k} \sum_{i} \frac{\vec{j}_k^{\mathrm{dif}}(\vec{x} + \frac{\vec{e}_i}{2}) + \vec{j}_k^{\mathrm{dif}}(\vec{x} - \frac{\vec{e}_i}{2})}{2D_k}. \tag{2.5.3}$$

This approach is less accurate but computationally more efficient because the calculated diffusive fluxes can be reused. This is the default discretization in this work except when explicitly stated differently.

Multicomponent Coupling

All mentioned couplings deal with the single-component nature of the solvent fluid. In Section 2.3 a multicomponent fluid model was discussed which offers more coupling possibilities to the system. In particular one focus is on the chemical nature of the different fluid components which can be interpreted as different excess chemical potentials which are present in the fluid components. For the case of two fluid components there are two different excess chemical potentials $\mu_{a,b}^{\mathrm{ex}}$ present. The inclusion of the chemical potentials into the fluid model and the model for the I ion species can be done with forces. The respective equation is the *Gibbs-Duhem relation* which reads

$$0 = S\mathrm{d}T - V\mathrm{d}p + \sum_{k=1}^{I} N_k \mathrm{d}\mu_k. \tag{2.5.4}$$

With the assumption of an isothermal system ($\mathrm{d}T = 0$) Equation (2.5.4) can be simplified and reads

$$\vec{f}(\vec{x}) = -\vec{\nabla}p(\vec{x}) = -\sum_{k} \frac{N_k}{V} \vec{\nabla}\mu_k(\vec{x}) \tag{2.5.5}$$

$$\vec{f}_k(\vec{x}) = -n_k \vec{\nabla}\mu_k(\vec{x}), \tag{2.5.6}$$

which links the pressure gradient to the gradient of the chemical potential and the number density of the different species k. In the context of the previously mentioned models the chemical potential μ_k has not been included and is therefore not known. Furthermore it is not clear on how the chemical potential should change in the transition region between the fluid components.

In this work the only change in the chemical potential between the fluid components is due to the change in the solvation energy. The difference of the chemical potentials between two fluid components A and B can be written as

$$\Delta\mu_k = \mu_k^{\text{solv}}(\text{fluid A}) - \mu_k^{\text{solv}}(\text{fluid B}), \tag{2.5.7}$$

which is called *Gibbs transfer energy* in the literature and describes the change in free energy when moving a particle of species k from fluid A to fluid B. This value is an input parameter to the models and the solvation chemical potential μ_k^{solv} is approximated. Fundamentally the solvation chemical potential is given by the chemical nature of the solvent which part of it are the hydrogen bonds formed with the solvated particle species. In the mesoscopic simulation scale this length scale can not be resolved, an approximative approach is taken which describes the solvation chemical potential using the fluid density as described in [27]. The equation for a linear dependency on the fluid density of only one of the two fluid components reads

$$\mu_k^{\text{solv}} = \frac{\Delta\mu_k}{\Delta\varrho_{\text{fluid}}^{\text{A}}}\varrho_{\text{fluid}}^{\text{A}}, \tag{2.5.8}$$

where $\Delta\mu_k$ is an input parameter and $\Delta\varrho_{\text{fluid}}^{\text{A}}$ is the maximum density difference of the density of fluid component A. The resulting force is given by plugging the definition of the solvation chemical potential (2.5.8) into the simplified Gibbs-Duhem relation (2.5.6) which results in

$$\vec{f}_k(\vec{x}) = -n_k(\vec{x})\frac{\Delta\mu_k}{\Delta\varrho_{\text{fluid}}^{\text{A}}}\vec{\nabla}\varrho_{\text{fluid}}^{\text{A}}(\vec{x}). \tag{2.5.9}$$

This additional force density is included as an additional contribution to the diffusive flux which was defined in Equation (2.4.3). The inclusion is done in the same manner as with the electrostatic force $-q_k n_k \vec{\nabla}\Phi$. The resulting addition to the diffusive flux reads

$$\vec{j}_k^{\text{solv}}(\vec{x},t) = -\frac{D_k}{k_{\text{B}}T}n_k(\vec{x},t)\frac{\Delta\mu_k}{\Delta\varrho_{\text{fluid}}^{\text{A}}}\vec{\nabla}\varrho_{\text{fluid}}^{\text{A}}(\vec{x},t) \tag{2.5.10}$$

and is added to the already discussed diffusive flux from Equation (2.4.3) and is therefore also part of the friction coupling in Equation (2.5.2). For the momentum to be conserved in the system the force density also needs a counter-force which in this case acts on the fluid. This can be done by switching the role of the fluid density $\varrho_{\text{fluid}}^{\text{A}}$ and the number density n_k in Equation (2.5.9) and exerting it on the fluid. The resulting force density reads

$$\vec{f}_{\text{A}}(\vec{x},t) = -\varrho_{\text{fluid}}^{\text{A}}(\vec{x},t)\sum_k \frac{\Delta\mu_k}{\Delta\varrho_{\text{fluid}}^{\text{A}}}\vec{\nabla}n_k(\vec{x},t). \tag{2.5.11}$$

The diffusive flux contribution for the ion species k is discretized in the same way as the other contributions to the diffusive flux, which apply the equations (2.4.11) and (2.4.12). Therefore the resulting discretized flux reads

$$\vec{j}_k^{\text{solv}}(\vec{x}+\frac{\vec{e}_i}{2},t) = -\frac{D_k}{k_\text{B}T}\frac{\Delta\mu_k}{\Delta\varrho^\text{A}_\text{fluid}}\frac{n_k(\vec{x}+\vec{e}_i,t)+n_k(\vec{x},t)}{2}\frac{\varrho^\text{A}_\text{fluid}(\vec{x}+\vec{e}_i,t)-\varrho^\text{A}_\text{fluid}(\vec{x},t)}{|\vec{e}_i|}. \tag{2.5.12}$$

For the force on the fluid the equation is discretized in the same way as the Shan-Chen interaction force is calculated, which is described in Section 2.3.1. The reason this discretization works comes from the observation in Equation (2.3.4) which shows the continuous equation of the Shan-Chen interaction which is of the same structure as Equation (2.5.9). The discretization for this term reads

$$\vec{f}_\text{A}(\vec{x},t) = -\varrho^\text{A}_\text{fluid}(\vec{x},t)\sum_k \frac{\Delta\mu_k}{\Delta\varrho^\text{A}_\text{fluid}\Delta t^2}\sum_i \omega_i n_k(\vec{x}+\vec{e}_i,t)\vec{e}_i. \tag{2.5.13}$$

The model of the solvation chemical potential in Equation (2.5.8) based on the fluid density has two disadvantages. First of all it requires the knowledge of the density differences of the fluid $\Delta\varrho^\text{A}_\text{fluid}$ and it is only dependent on the density of one fluid component and therefore neglects the other fluid component. Furthermore this formulation also suffers from the density fluctuations due to pressure variations because of the ideal gas contribution in the equation of state (2.2.7) of the Lattice Boltzmann Method. An alternative formulation for this model is using the previously introduced phase field χ in Equation (2.3.13). This quantity typically is less influenced from density fluctuations and can simply replace the density formulation in Equation (2.5.8) which reads

$$\mu_k^\text{solv} = \frac{\Delta\mu_k}{2}\chi(\vec{x}), \tag{2.5.14}$$

where the factor $\frac{1}{2}$ comes from the range of values of the phase field. It includes both fluid densities and does not require the knowledge of the fluid densities values inside the fluid phases when using fluid components with equal bulk densities. The discretization of the forces is analogue to the ones described above by replacing the fluid density with the phase field and $\Delta\varrho^\text{A}_\text{fluid}$ with the factor $\frac{1}{2}$.

2.6. Summary of Models

In the previous sections the models for the multicomponent fluids as solvents as well as the ions as solutes have been discussed together with the changes to the original models. In this section a summary of the models is provided for a quick overview. In all equations the time dependency of the quantities is omitted.

Shan-Chen Multicomponent Fluids

The multicomponent fluids are simulated using the Shan-Chen multicomponent model. The necessary Lattice Boltzmann fluids are simulated on a cubic lattice using the

BGK collision operator. The forces on the fluids include the Shan-Chen interaction forces, the friction coupling due to the solute movement as well as the forces stemming from the differences in the chemical potentials given by the Gibbs transfer energy. The continuous formulation of the forces reads

$$\begin{aligned}\vec{f}_\sigma(\vec{x}) =& -\psi_\sigma(\vec{x})\sum_{\sigma'} G_{\sigma\sigma'} c_\mathrm{s}^2 \vec{\nabla}\psi_{\sigma'}(\vec{x}) - \delta_{\sigma\mathrm{A}}\varrho^\mathrm{A}_\mathrm{fluid}(\vec{x})\sum_k \frac{\Delta\mu_k}{\Delta\varrho^\mathrm{A}_\mathrm{fluid}}\vec{\nabla} n_k(\vec{x}) \\ &- \frac{\varrho_\sigma}{\varrho}\left(k_\mathrm{B}T\sum_k \vec{\nabla} n_k(\vec{x}) + \sum_k q_k n_k(\vec{x})\vec{\nabla}\Phi(\vec{x}) + \sum_k \frac{\Delta\mu_k}{\Delta\varrho^\mathrm{A}_\mathrm{fluid}} n_k(\vec{x})\vec{\nabla}\varrho^\mathrm{A}_\mathrm{fluid}(\vec{x})\right)\end{aligned} \tag{2.6.1}$$

which is discretized on the grid with the back-interpolation of the friction coupling described in Equation (2.5.3)

$$\begin{aligned}\vec{f}_\sigma(\vec{x}) =& -\psi_\sigma(\vec{x})\sum_{\sigma'} G_{\sigma\sigma'} c_\mathrm{s}^2 \sum_i \omega_i \psi_{\sigma'}(\vec{x}+\vec{e}_i)\vec{e}_i \\ &+ \frac{\varrho_\sigma k_\mathrm{B}T}{\varrho}\sum_k\sum_i \frac{\vec{j}^\mathrm{dif}_k(\vec{x}+\frac{\vec{e}_i}{2}) + \vec{j}^\mathrm{dif}_k(\vec{x}-\frac{\vec{e}_i}{2})}{2D_k} \\ &- \varrho^\mathrm{A}_\mathrm{fluid}(\vec{x})\sum_k \frac{\Delta\mu_k}{\Delta\varrho^\mathrm{A}_\mathrm{fluid}\Delta t^2}\sum_i \omega_i n_k(\vec{x}+\vec{e}_i)\vec{e}_i\end{aligned} \tag{2.6.2}$$

Lattice Electrokinetics

The dynamics of the ion distributions given by the Nernst-Planck equation are propagated using a finite volume discretization method. The ion fluxes have a diffusive and an advective contribution. The diffusive flux includes the concentration gradient, the coupling to the electrostatic potential as well as the contribution from the chemical potential formulated with the Gibbs transfer energy. The continuous formulation reads

$$\vec{j}^\mathrm{dif}_k(\vec{x}) = -D_k\vec{\nabla} n_k(\vec{x}) - \frac{D_k}{k_\mathrm{B}T}q_k n_k(\vec{x})\vec{\nabla}\Phi(\vec{x}) - \frac{D_k}{k_\mathrm{B}T}n_k(\vec{x})\frac{\Delta\mu_k}{\Delta\varrho^\mathrm{A}_\mathrm{fluid}}\vec{\nabla}\varrho^\mathrm{A}_\mathrm{fluid}(\vec{x}) \tag{2.6.3}$$

which is discretized on the surfaces of the volume elements according to the finite volume discretization

$$\begin{aligned}\vec{j}^\mathrm{dif}_k(\vec{x}+\frac{\vec{e}_i}{2}) =& D_k\frac{n_k(\vec{x}+\vec{e}_i)-n_k(\vec{x})}{|\vec{e}_i|}\hat{\vec{e}}_i \\ &+ \frac{D_k}{k_\mathrm{B}T}q_k\frac{n_k(\vec{x}+\vec{e}_i)+n_k(\vec{x})}{2}\frac{\Phi(\vec{x}+\vec{e}_i)-\Phi(\vec{x})}{|\vec{e}_i|}\hat{\vec{e}}_i \\ &+ \frac{D_k\Delta\mu_k}{k_\mathrm{B}T\Delta\varrho^\mathrm{A}_\mathrm{fluid}}\frac{n_k(\vec{x}+\vec{e}_i)+n_k(x)}{2}\frac{\varrho^\mathrm{A}_\mathrm{fluid}(\vec{x}+\vec{e}_i)-\varrho^\mathrm{A}_\mathrm{fluid}(\vec{x})}{|\vec{e}_i|}\hat{\vec{e}}_i.\end{aligned} \tag{2.6.4}$$

The advective flux is discretized using the volume of fluid method as an additional contribution to the fluxes and reads

$$\vec{j}_k^{\text{adv}}(\vec{x}) = n_k(\vec{x})\vec{v}(\vec{x}), \tag{2.6.5}$$

with the discretization given in Equation (2.4.15).
The Poisson equation for the electrostatic potential reads in the most general form

$$\vec{\nabla} \cdot \Big(\varepsilon(\vec{x},t)\vec{\nabla}\Phi(\vec{x},t)\Big) = -\frac{q(\vec{x},t)}{\varepsilon_0} \tag{2.6.6}$$

which includes a spatial dependent permittivity. In the case of spatially independent permittivity the equation reduces to

$$\Delta\Phi(\vec{x}) = -\frac{q(\vec{x})}{\varepsilon\varepsilon_0}, \tag{2.6.7}$$

which is solved based on the fast Fourier transformation (Section 2.4.5) for fully periodic systems or the successive overrelaxation method (Section 2.4.5) if more complex boundary conditions are required.

3. Implementation

Grid based algorithms like the Lattice Boltzmann Method are prime examples where highly optimized and parallelized code can make a huge difference in the computation time of the algorithms. In the scientific programming community the requirements for production code has evolved in the past decade. The focus for implementations has shifted from purely performant code towards higher quality code which includes readability, testability and maintainability. The reason for this shift is the overall increase in performance of the hardware which allows to write less optimized code without loosing all of the performance but allow others to read and understand the implementations.

In this work the implementation of the grid based algorithms is done using a python module named *pystencils* [3]. It is capable of generating highly optimized C++ or Cuda code from symbolic math expressions which can be used for image processing as well as for numerical simulations which are performed on a grid. Using the code generation feature one can write the expressions in an abstract mathematical way which allows for good readability and maintainability of the code.

pystencils

The *pystencils* module itself extends the well known *sympy* symbolic math library [28] by stencil operations which are performed on each grid cell. These operations typically include the values of neighboring grid cells which are expressed as offsets to the current cell. Internally each access to an array is represented by a symbol. This representation allows to use the *sympy* symbolic math optimizations which are capable of grouping common sub-expressions and reducing the overall amount of operations necessary for the evaluation. This symbolic-math grouping is one ingredient for highly-performant code which can directly be translated to lower level programming languages like C++ or Cuda which is the process referred to as code generation. The code can be extracted and used together with waLBerla [29] or used with the build-in array handlers. The first one is a high-performance simulation framework written in C++ that is designed for block-structured multiphysics simulations. It is focused on highly parallelized Lattice Boltzmann implementations including regular parallelization with MPI or hybrid-parallelization with MPI+OpenMP. The latter mentioned build-in array handler provides a just-in-time compilation of the code as well as the organization of the arrays which allow for calculations performed on the CPU using *numpy* [30] arrays as well as calculations on GPUs with *pyCUDA* or *openCL*. The array-handling also includes a functionality which allows to impose

Listing 3.1: Example code for the code generation of a simple diffusion term using the finite-volume discretization.

```
flux_expression = - diffusion_coefficient * sympy.Matrix(
            [pystencils.fd.diff(density_field, i)
                for i in range(density_field.spatial_dimensions)])

fvm = pystencils.fd.FVM1stOrder(density_field,
                                flux=expression)

flux_calc = fvm.discrete_flux(flux_field)
density_update = fvm.discrete_continuity(flux_field)

flux_kernel = pystencils.create_staggered_kernel(flux_calc).compile()
density_kernel = pystencils.create_kernel(density_update).compile()
```

boundary conditions to the arrays and can therefore be used as a framework to solve differential equations which are discretized on grids.

The representation of stencil operations with the symbols using *sympy* also has the advantage that derivatives can directly be expressed using *sympy* built-in tools. These derivatives can then easily be replaced with finite difference derivative expressions which is done with an automatic discretization tool build into *pystencils*. This tool is also able to discretize equations with the finite-volume discretization based on staggered accesses to fields. An example on how this discretization looks like for a simple diffusion term using the finite-volume discretization is shown in Listing 3.1. The calculation of the fluxes and the following density update is done with two separate kernels.

lbmpy

The *pystencils* module is primarily designed to generate code which is focused on finite difference and finite volume discretizations. The code for the Lattice Boltzmann Method is slightly more complex which is why we there is an extension-module to *pystencils* in development named *lbmpy* [4] that is designed to generate Lattice Boltzmann kernels for all different kind of flavors possible. This includes different dimensionality, velocity sets, collision models and force models. It uses the *pystencils* framework to generate highly optimized streaming and collision kernels for the respective set of chosen options which can also include the output of macroscopic quantities. An example on how to generate the collision and streaming kernels is shown in Listing 3.2.

This extension is also capable of using *sympy* to perform the Chapman-Enskog expansion on the specific Lattice Boltzmann model. At the point of writing this expansion does not respect forces which are added to the Lattice Boltzmann models, which are for example present in the Shan-Chen model.

Listing 3.2: Example code for the code generation of simple collision and streaming kernels.

```
collision_operation = create_lb_update_rule(
            method="srt",
            stencil=get_stencil("D2Q9"),
            relaxation_rates=(1.0,),
            compressible=True,
            force_model="guo",
            force=force_field,
            kernel_type="collide_only")

collision_kernel = pystencils.create_kernel(collision_operation).compile()

stream_operation = create_stream_pull_with_output_kernel(
            lb_method=collision_operation.method,
            src_field=populations_field_src,
            dst_field=populations_field_dst,
            output={
                "density": density_field,
                "velocity": velocity_field
            })

stream_kernel = pystencils.create_kernel(stream_operation).compile()
```

3.1. Model Implementations

The remaining part of this chapter deals with implementation details. The implementation of the models is done in a python module which offers the discretization using the *pystencils* and *lbmpy* modules for the code generation. This way all computations in this work are performed with the highly optimized code generated by these modules. The implementations of the individual models are independent of the couplings between them. The couplings are introduced with forces which are evaluated in a prescribed order which is given by a time-loop functionality.

3.1.1. Lattice Boltzmann Implementation

The kernels for the LB implementation are generated using the *lbmpy* built-in discretization functions. In this work the Guo-forcing scheme to include forces together with the BGK collision operator as described in Section 2.2 is used. This choice is not fixed and can be swapped without additional effort. The discretization functions generate kernels for the collision operator as well as for the streaming operator which in this case also include the calculation and output of macroscopic quantities like the density and the velocity. These calculated quantities are reused in the collision operator as well as the forces which are included with an additional field. The reason for using a field for the forces is that the forces can depend on the fluid densities as it is the case in the Shan-Chen model which is a problem if the density is not double-buffered and directly calculated in the streaming kernel. With the specific mentioned set-up the velocity correction with the force uses already updated fluid

densities which causes a violation to the momentum conservation. Storing the forces in a field also has the benefit that the calculated forces can be reused in the velocity calculation. The collision and streaming operators are called consecutively with the boundary handling functions between the calls, a schematic representation of the timeloop is shown in Figure 3.1.

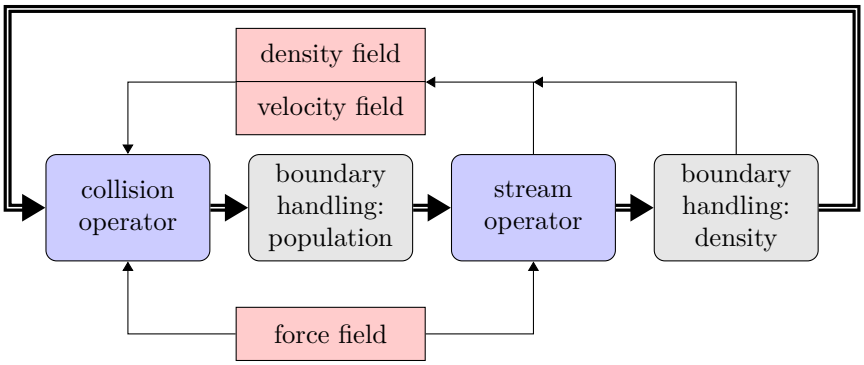


Figure 3.1.: Sketch of the used calling-order of the kernels in the Lattice Boltzmann implementation.

3.1.2. Shan-Chen Implementation

The more interesting implementation is the Shan-Chen multi-component model for which two separate LB fluids are used which each have separate fields including the force fields. These force fields are populated using a force calculation kernel which calculates the Shan-Chen-interaction between the fluids using the density fields which are the output of the streaming kernels. This kernel can also include other force expressions. The collision kernels of the two LB fluids are reused and merged to a single kernel with a modified velocity input according to Equation (2.3.6). The two streaming kernels are also merged to a single kernel and the output of the barycentric velocity according to Equation (2.3.6) is added. Optionally the calculation of the phase field can also be included in the streaming kernel for which a boundary handling is also available. The timeloop for this implementation is shown in Figure 3.2. It starts with the calculation of the forces using the force kernel which uses the density fields as input and writes the results to a force field. These forces are then used together with the barycentric equilibrium velocity and the densities in the collision operator which operates on both fluid components. Afterwards the boundary handling for the populations is executed which is followed by the streaming operators which reuse the forces in the force fields for the velocity correction and output the macroscopic velocity field as well as the fluid densities and optionally other fields. The execution cycle finishes with the boundary handling for the densities and the velocities.

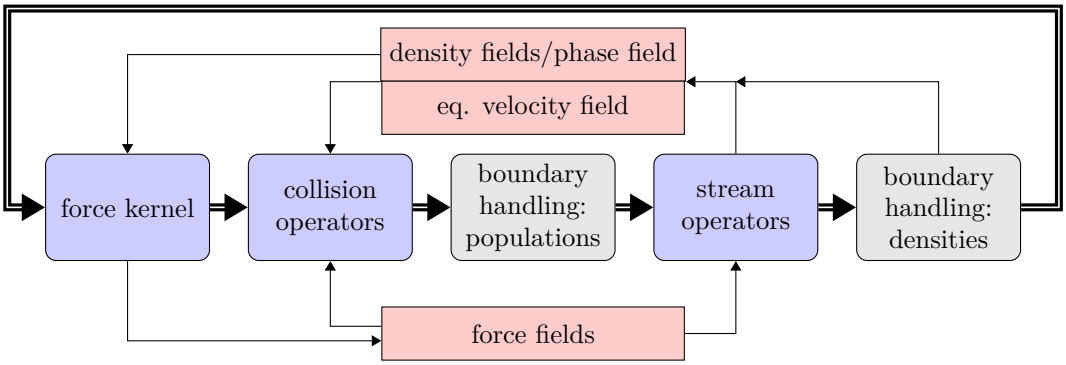


Figure 3.2.: Sketch of the used calling-order of the kernels in the Shan-Chen implementation.

3.1.3. Lattice Electrokinetics Implementation

For the discretization of the Nernst-Planck equation the automatic finite-volume discretization of *pystencils* is used. For that only the number of discrete fluxes used and the symbolic expression for the diffusive flux has to be provided using *sympy* symbols. The symbols corresponding to field accesses are automatically interpolated on the surfaces and the generated kernel stores the fluxes in a staggered field. The second generated kernel updates the densities according to the fluxes provided. The advective flux is discretized using the automatic volume-of-fluid discretization function which generates a kernel that uses a velocity field as an input and adds the corresponding fluxes to the diffusive fluxes. A boundary handling for the fluxes as well as for the densities is included. The possibility of propagating charged species is provided with a child class that includes the calculation of the charge density and uses a Poisson Solver to calculate the electrostatic potential. The timeloop for this child class is shown in Figure 3.3, the variation without electrostatics is implemented in the parent class and extended with the electrostatic contribution. It starts with the electrostatics which includes the calculation of the charge density using the ion density fields that is written to a field including the boundary handling as well as the actual solving process of the Poisson equation where the electrostatic potential is written to a separate field. Afterwards the ion fluxes are calculated from the electrostatic potential and the ion densities and are outputted to the flux field where also the boundary handling is performed on. The ion densities in the density field are updated using the calculated fluxes in the update kernel. Lastly the boundary handling for the density is executed which is the final call in a single time step.

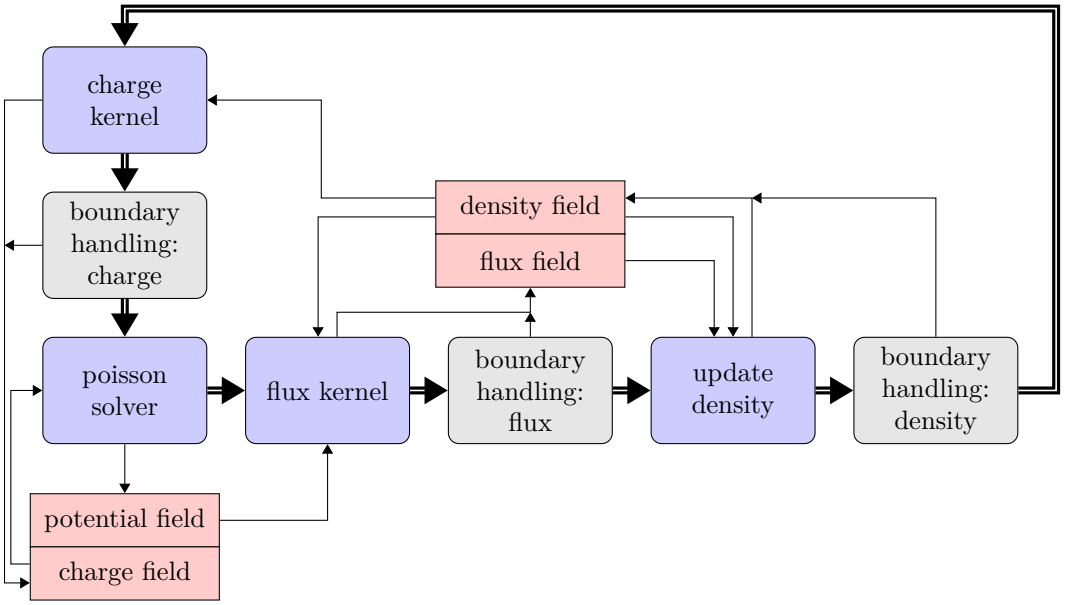


Figure 3.3.: Sketch of the used calling-order of the kernels in the lattice electrokinetics implementation.

3.1.4. Two Phase Fluid with Charged Ion Species

The previously described implementations are coupled together for the two phase fluid together with the charged ion species. The additional coupling contributions are provided to the models using the described mechanisms. The only additional code needed is the timeloop which is structured like shown in Figure 3.4. The simulation starts with zero-initialized force-fields for the fluids which are updated using the force-calculation kernel including Shan-Chen interactions and other external forces which are not coupling to the diffusive flux. The two LB fluids are propagated using the regular timeloop for the fluids including collision and streaming with the corresponding boundary handlings. Afterwards the diffusive flux for the ion species is calculated and the boundary handling is applied. This flux is then used to calculate the friction coupling force for the fluid which is applied in the next timestep. The advective flux is added to the diffusive flux with an additional boundary handling call for the fluxes. Finally the ion densities are updated with the calculated fluxes and the boundary handling for the density is applied.

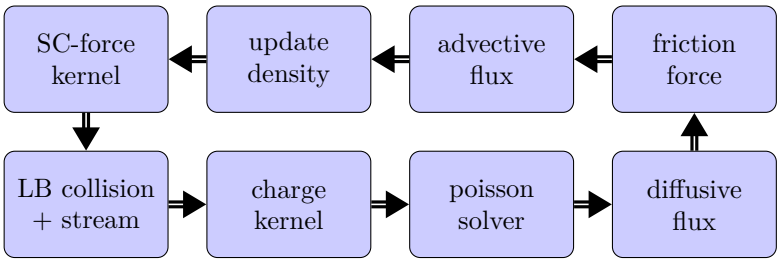


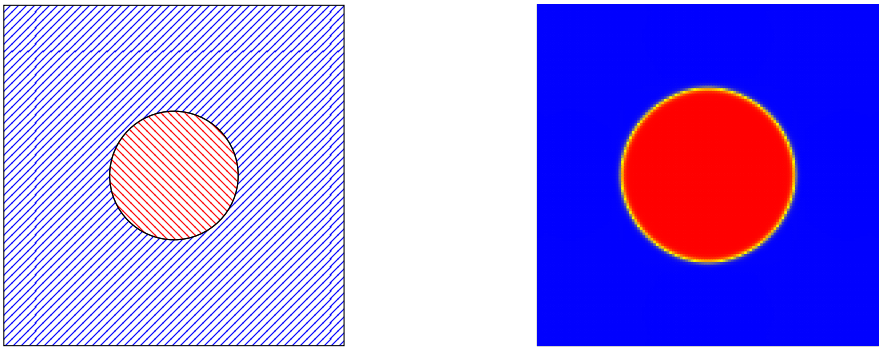
Figure 3.4.: Sketch of the used calling-order of the kernels in the full implementation of the Shan-Chen two phase model as well as the lattice electrokinetics. The steps for the boundary handlings as well as the illustration of the fields are omitted.

4. Testcases

In the previous section some details of the implementation were discussed, in this section the tests for the implementation are presented which are used to verify the correctness of the implementation. First the individual models are tested on their own starting with the Shan-Chen two component implementation followed by the lattice electrokinetics implementation. The last testcases shown are using the full coupling of both models.

4.1. Density Separation and Young-Laplace Surface Tension

In the Lattice Boltzmann Method the pressure in the fluid is given by the equation of state (2.2.7) which was introduced in Section 2.2.2. It shows that the pressure in the fluid is related to the local density. The same is true for the Shan-Chen multiphase model where the equation of state is altered by the potential as shown in Equation (2.3.5) that is also dependent on the local fluid density. This is especially relevant in systems where the pressure inside the two fluid components is not equal which is the case for systems with curved fluid interfaces. This relation is shown in the following sections for a fluid droplet embedded in another fluid component, a sketch of the system is shown in Figure 4.1.



(a) (b)

Figure 4.1.: Sketch of a droplet simulation set-up (a) in a fully periodic domain. The fluid droplet is shown in red and named fluid component A, the outer fluid is named component B and is shown in blue. A snapshot of the phase field of the simulation is shown in (b). The simulation domain has a side length of $128\Delta x$ and the radius of the droplet is $30\Delta x$.

4.1.1. Density Separation

In the first testcase the interaction constant $G_{\sigma\sigma'}$ is varied and the average densities inside the droplet are measured. The definition of what is accounted to be inside the droplet and what is outside is done with the phase field $\chi(\vec{x})$ defined in Equation (2.3.13). The threshold value for the interface is chosen to be $\chi(\vec{x}) = 0$ and therefore $\chi(\vec{x}) > 0$ is accounted to be inside the droplet. The density curves are shown in Figure 4.2, the simulation parameters are listed in Table 4.1.

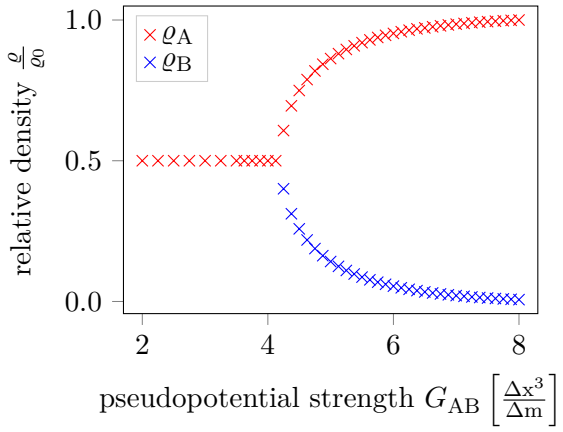


Figure 4.2.: Average fluid density inside of a fluid droplet for both fluid components as a function of the interaction constant in a two component Shan-Chen fluid set-up. For low interaction constants the fluid components do not separate, while by increasing the interaction constant the separation of the average fluid density increases and the immiscible fluid component is displaced.

In this figure it can be deduced that the separation of the fluid components starts after a specific threshold value for the interaction constant is reached. It can also be seen that the fluid component B is still present with a strongly diminished density in the fluid component A even after the droplet has stabilized. This density decreases with increasing the interaction constant $G_{\sigma\sigma'}$ but does not completely vanish in the shown parameter range. Therefore, a property of the model is, that in the simulation of immiscible fluid components there is still a small portion of fluid density left in the immiscible component.

Table 4.1.: Simulation parameters used for the calculation of the phase separation.

quantity	symbol	value	unit
domain size		(100, 100)	Δx
LB fluid densities	$\varrho_{A/B}$	1	$\frac{\Delta m^3}{\Delta x}$
LB relaxation times	$\tau_{A/B}$	1	Δt

4.1.2. Young Laplace Surface Tension

A popular testcase for multiphase fluid implementations is the Young-Laplace equation. The equation relates the curvature of a fluid interface to the pressure difference between the two phases forming the interface with the proportionality constant as the surface tension σ between the phases. The equation reads

$$\Delta p = p^{\text{inside}} - p^{\text{outside}} = \begin{cases} \sigma \frac{1}{R} & \text{in 2D} \\ \sigma \frac{2}{R} & \text{in 3D} \end{cases}, \tag{4.1.1}$$

where σ is the surface tension between the two fluid components and R is the principal radius of the interface. This relation can be tested on the previously described set-up of the fluid droplet which is sketched in Figure 4.1.

The simulation is set-up for varying initial radii of the droplet and different interaction coefficients $G_{\sigma\sigma'}$. In Section 4.1 it was briefly mentioned that the Lattice Boltzmann fluid density determines the pressure which can be seen in the equation of state (2.3.5). This relation between the pressure and the fluid density causes the radius of the fluid droplets to slightly change during the simulation which is why the radius of the droplet is measured from the simulation. For this measurement the fluid-fluid interface is used which is defined with the phase field at $\chi(\vec{x}) = 0$. These points on the interface are then used to determine the radius of a circle with a least-squares fit.

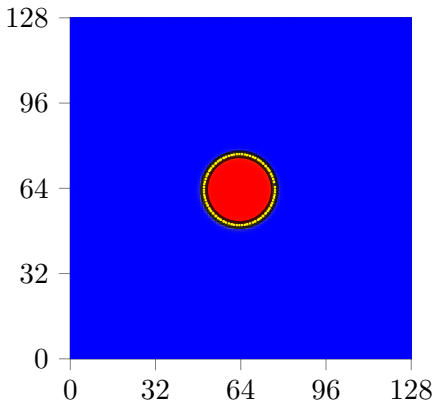


Figure 4.3.: Measurement of the droplet radius from the phase field. The values of the phase field are shown as background colors, the points on the contour line as yellow marks and the fitted circle is shown as a black line.

The pressure value inside the center of the droplet is estimated using the arithmetic mean of the four most central points in the droplet. For the outside pressure the arithmetic mean of the pressure in the corner points is used. The measured pressure difference is plotted over the inverse droplet radius and is shown in Figure 4.4. For each dataset corresponding to a specific interaction constant G_{AB} a linear function is interpolated with a least-squares fit with the only degree of freedom being the slope.

The fitted parameter is the surface tension σ for this specific interaction constant according to Equation (4.1.1). The simulation values used are shown in Table 4.2.

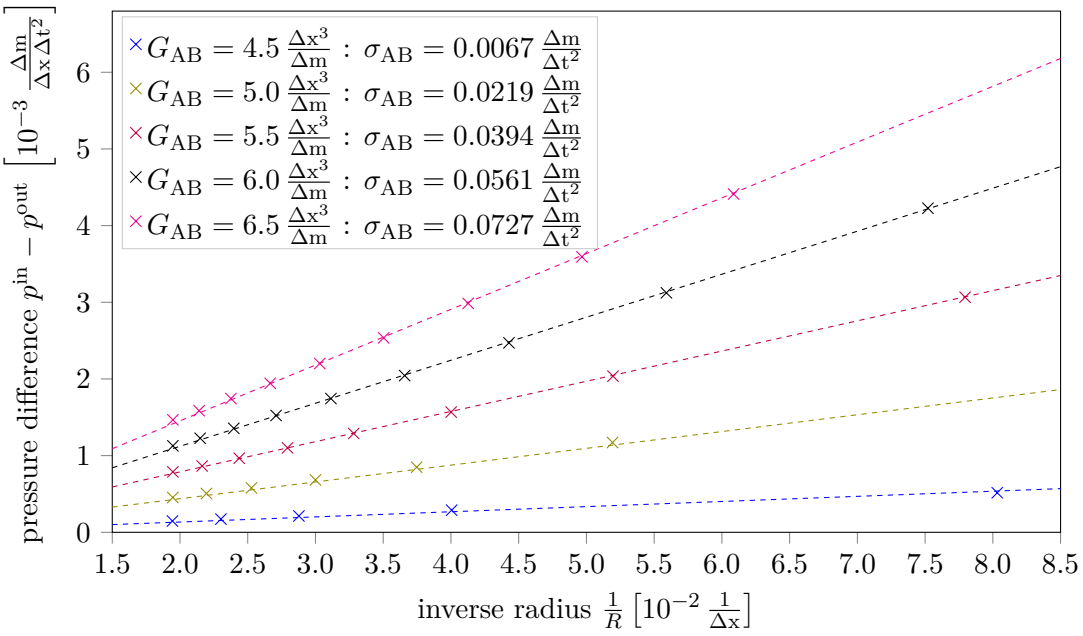


Figure 4.4.: Young-Laplace surface tension relation showing the pressure difference inside to outside the droplet above the inverse radius of the droplets. The resulting relation is linear with the proportionality constant as the surface tension σ which is determined by a linear fit for each interaction constant separately. A sketch of the system is shown in Figure 4.1.

Table 4.2.: Simulation parameters used for the calculation of the surface tension.

quantity	symbol	value	unit
domain size		(128, 128)	Δx
LB fluid densities	$\varrho_{A/B}$	1	$\frac{\Delta m^3}{\Delta x}$
LB relaxation times	$\tau_{A/B}$	1	Δt

From the figure it can be seen that the Young-Laplace equation (4.1.1) shows a good agreement with the parameter fitted to the data points for each interaction constant. As previously mentioned this fitting procedure is the most convenient way of measuring the surface tension for a specific set of model parameters because of the unknown dependency of the surface tension on the model parameters. The measured surface tensions provided in the labels of the figure are also shown over the pseudopotential fluid-fluid interaction strength in Figure 4.5.

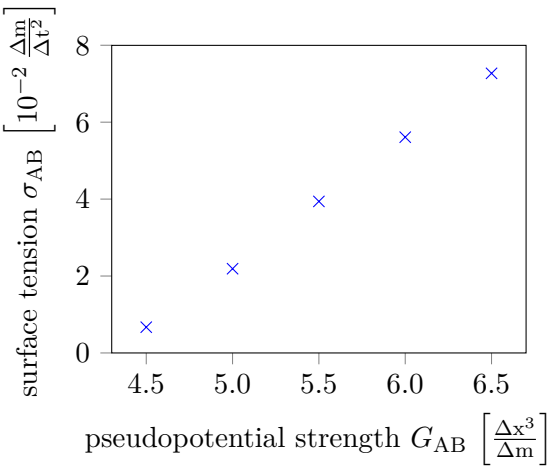


Figure 4.5.: Measured surface tension from the linear interpolation in Figure 4.4 over the pseudo-potential fluid-fluid interaction strength. The surface tension seems to be approximately linearly increasing with the interaction strength G_{AB} although there is no analytical equation known that predicts this behavior.

4.2. Poiseuille Flow

The Navier-Stokes Equation (2.2.2) is difficult to solve and only a few analytic results results are known. To validate the correctness of the implementation of the numeric solvers testcases are needed. Especially useful and robust are testcases with an analytical solution. For the case of fluid simulations the Poiseuille flow is such a possible set-up. There are different geometric shapes for which the analytical solution is known, the one discussed here is the Poiseuille flow between two parallel plates with the distance h apart which is depicted in Figure 4.6.

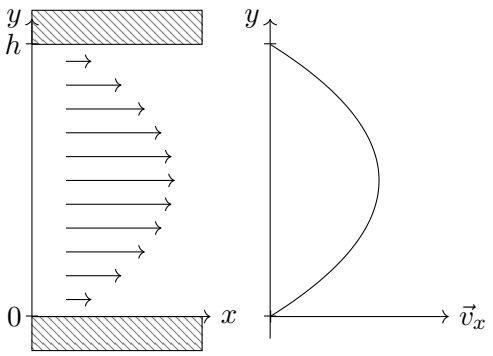


Figure 4.6.: Sketch of a planar Poiseuille flow set-up in two dimensions on the left with velocity vectors shown in between the two plates. On the right the corresponding velocity curve is shown.

Because the set-up of the problem of the infinite parallel plates requires the solution to be invariant to shifts in the direction parallel to the plates, and the solution should be the stationary state upon applying the constant body force density $\vec{f} = f\vec{e}_x$, the problem is significantly simplified. The solution can only depend on the relative position orthogonal to the plates and, with the boundary conditions of a vanishing fluid velocity at the surfaces, also the pressure is constant in the system. Therefore a constant pressure in the system is assumed and the resulting partial differential equation to be solved reads

$$\partial_y^2 v_x(y) = -\frac{f}{\mu}, \tag{4.2.1}$$

with the two boundary conditions

$$v_x(0) = 0 \qquad\qquad v_x(h) = 0. \tag{4.2.2}$$

The solution to this equation can be trivially calculated by integrating Equation (4.2.1) twice with respect to the y-coordinate and resolving the two integration constants with the boundary conditions from Equation (4.2.2):

$$v_x(y) = -\frac{f}{2\mu} y\,(y-h)\,. \tag{4.2.3}$$

This is a standard test for single phase Lattice Boltzmann implementations and is shown here for the sake of completeness. The result for a single parameter set is shown in Figure 4.7 with the parameters in Table 4.3.

Table 4.3.: Simulation parameters used for the calculation of the single phase Poiseuille flow.

quantity	symbol	value	unit
domain size		(5, 50)	Δx
LB fluid density	ϱ	1	$\frac{\Delta \mathrm{m}^3}{\Delta \mathrm{x}}$
LB relaxation time	τ	1	Δt
external force	f_x^{ext}	10^{-5}	$\frac{\Delta \mathrm{m}}{\Delta \mathrm{x}^2\,\Delta \mathrm{t}^2}$

The more interesting testcase for the implementation is the extension to the two phase Poiseuille flow, similar to the work in [31] although the used model is different. The governing equation in the fluid components is the same as for the one-phase testcase in Equation (4.2.1) with the only difference that the dynamic viscosity is not necessarily equal between the two components. This means that in each fluid component Equation (4.2.1) has to be solved with the respective dynamic viscosity $\mu_{\mathrm{A/B}}$:

$$\begin{cases} \partial_y^2 v_x^{\mathrm{A}}(y) = -\frac{f}{\mu_{\mathrm{A}}} & \text{for fluid A} \\ \partial_y^2 v_x^{\mathrm{B}}(y) = -\frac{f}{\mu_{\mathrm{B}}} & \text{for fluid B.} \end{cases} \tag{4.2.4}$$

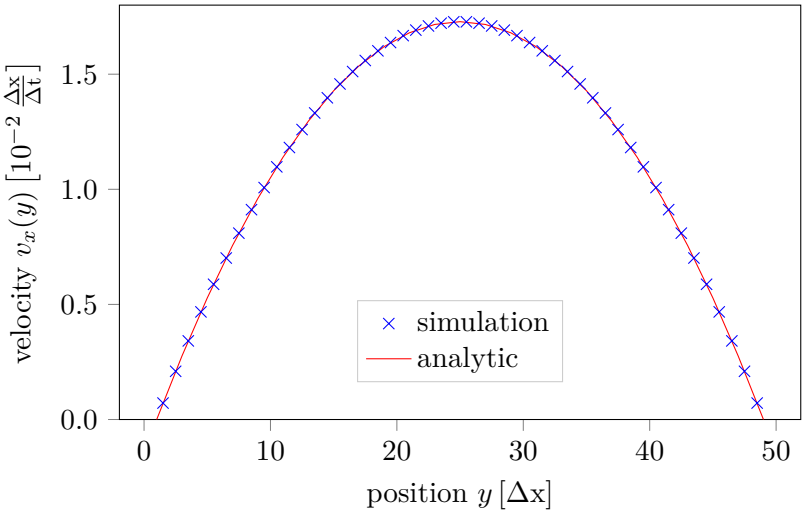


Figure 4.7.: Comparison between the lattice Boltzmann simulation for a Poiseuille flow to the analytic solution given by Equation (4.2.3).

The boundary conditions for the velocities from Equation (4.2.3) are still valid but additional boundary conditions for the fluid interfaces are introduced. The shear stresses and the velocities of the fluids at the fluid interfaces have to be continuous. This has to be preserved for every fluid interface. If the fluid interface is positioned at $y = h_0$ the conditions can be written as

$$-\mu_{\mathrm{A}}\partial_y v_x^{\mathrm{A}}(y)\big|_{y=h_0} = -\mu_{\mathrm{B}}\partial_y v_x^{\mathrm{B}}(y)\big|_{y=h_0} \tag{4.2.5a}$$

$$v_x^{\mathrm{A}}(h_0) = v_x^{\mathrm{B}}(h_0). \tag{4.2.5b}$$

Two different set-ups are solved analytically in the parallel channel set-up. In the first system there is only one fluid interface in the channel which separates the two components, shown in Figure 4.8a. The second set-up is that one fluid component is embedded between another fluid component as shown in Figure 4.8b. A summary of the derivation of the corresponding velocity equations is given in Appendix A.1. The simulation results for two viscosity ratios are shown in Figure 4.9 for the set-up in Figure 4.8a and the results for set-up 4.8b in Figure 4.11.
It can be seen that the analytic expression does not match the velocity profiles of the simulation for both shown viscosity ratios. The reason for this mismatch is the remaining fluid density of the immiscible fluid component inside the opposite fluid component which was already discussed in Section 4.1.1 and can also be seen in Figure 4.2. The two fluid components have different viscosities, and because the fluid components are not completely immiscible, the effective viscosity is influenced. A priori it is not clear how the resulting viscosity quantitatively looks like. To make this point more clear, two miscible fluid components are set-up with the same viscosity

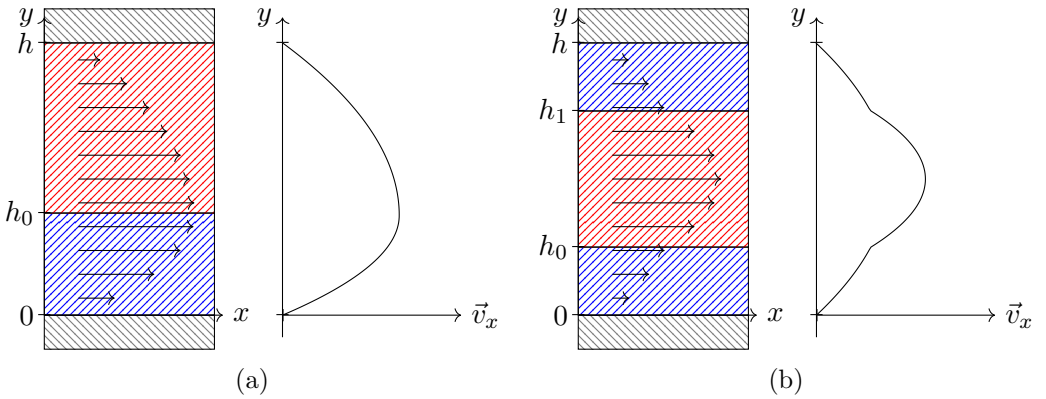


Figure 4.8.: Sketches of the two different two-phase set-ups with the fluid A marked in blue and fluid B marked in red. In (a) the set-up with only one fluid interface separating the two phases is shown. (b) is showing the set-up with the central slice of fluid B embedded in fluid A. In both shown cases the blue fluid component is of higher viscosity.

contrast but with different fluid density ratios in the same channel. The expected results of the velocity profile follow the one-phase Poiseuille flow profile as seen in Figure 4.7 but with different effective viscosities. These results for these simulations are shown in Figure 4.10 where the analytic solution uses a linear interpolation between the fluid viscosities based on their densities. It can be seen that the effective viscosity of the immiscible Shan-Chen fluid components is influenced by the remaining density of the immiscible component. Therefore the velocity profiles in Figure 4.8 are also analyzed with the fluid viscosities as fitting parameters, and the resulting profiles seem to match the simulated values very well.

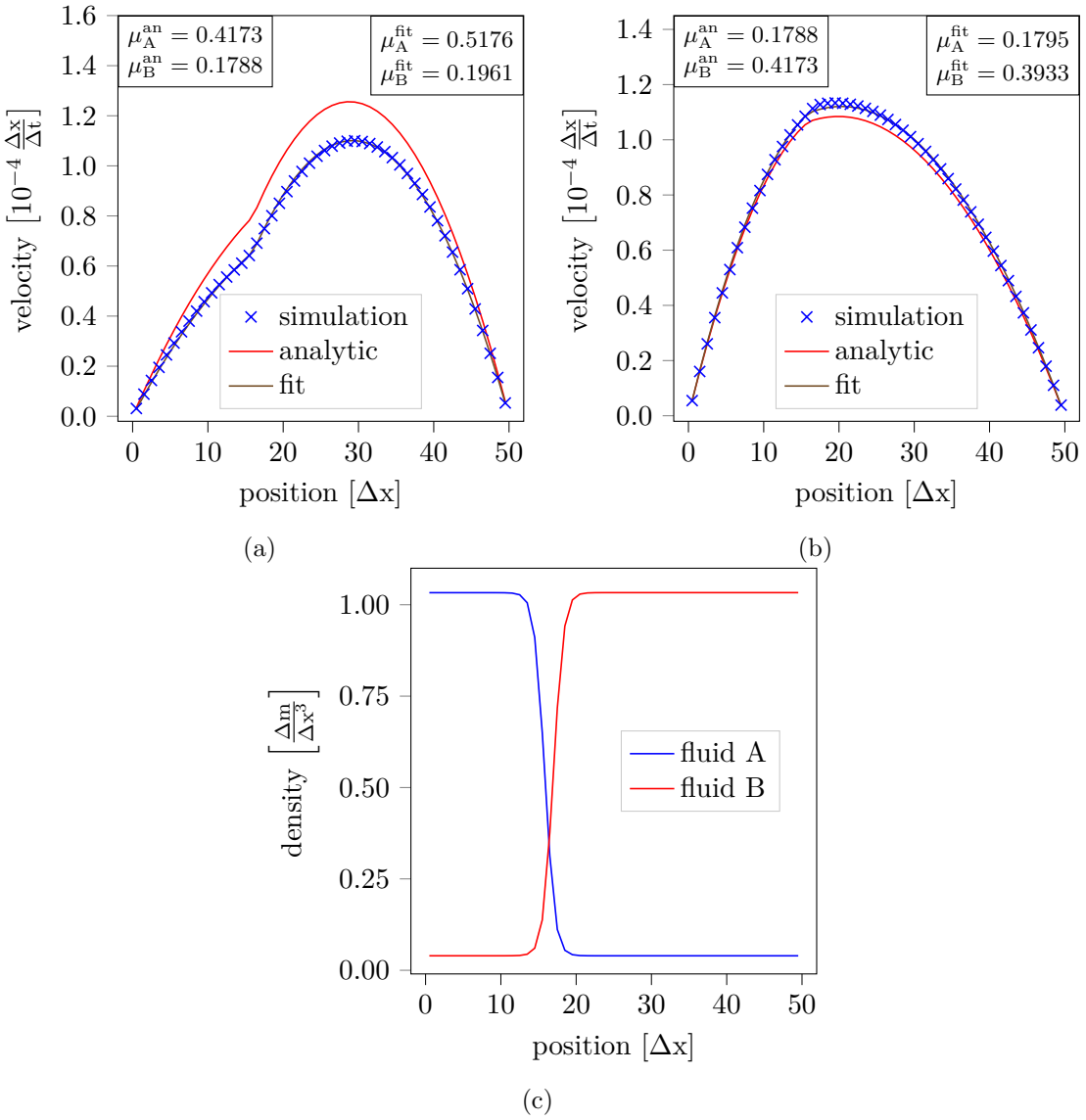


Figure 4.9.: Simulation results of immiscible two-phase Poiseuille profiles with different viscosity ratios. A sketch of the simulation set-up can be seen in Figure 4.8a. (a) shows the velocity profile for the case where the fluid on the left of the domain is of higher viscosity and (b) shows the opposite set-up. In (c) the fluid density profiles are shown which is equal for both cases of the viscosity ratios. The used kinematic viscosities are $\nu_1 = \frac{7}{18}\frac{\Delta x^2}{\Delta t}$ and $\nu_2 = \frac{1}{6}\frac{\Delta x^2}{\Delta t}$. The fitted dynamic viscosities are shown in the figures and have the units $[\mu] = \frac{\Delta m}{\Delta x\,\Delta t}$. The discrepancy between the analytical solution and the simulation is due to effective viscosity effects which are the result of the remaining fluid density of the immiscible fluid component.

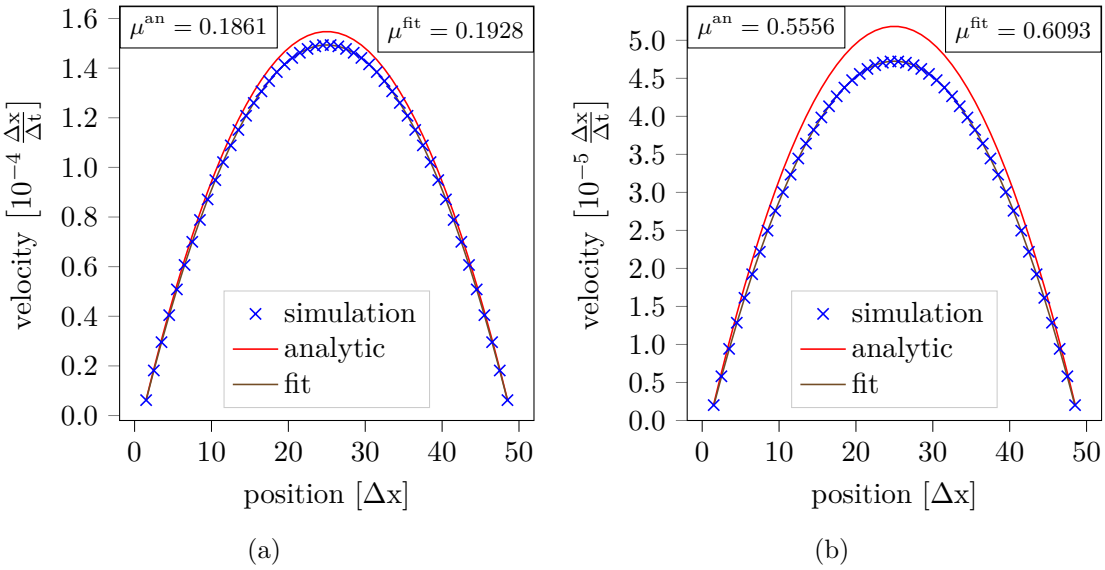


Figure 4.10.: Simulation results of miscible two-phase Poiseuille profiles with the same viscosity ratios but different density ratios. The analytical solution is calculated from a density-based linear interpolation of the viscosities. The used viscosities are $\nu_1 = \frac{7}{18}\frac{\Delta x^2}{\Delta t}$ and $\nu_2 = \frac{1}{6}\frac{\Delta x^2}{\Delta t}$. (a) shows the velocity profile for the case of the fluid densities in the ratio of $1 : 0.05$ and (b) the ratio $1 : 1$. The dynamic viscosities are shown in the figures and have the units $[\mu] = \frac{\Delta m}{\Delta x\,\Delta t}$. It can be seen that the effective viscosity is influenced by the fluid density ratio which is the reason for the discrepancy in the simulation results of Figure 4.9.

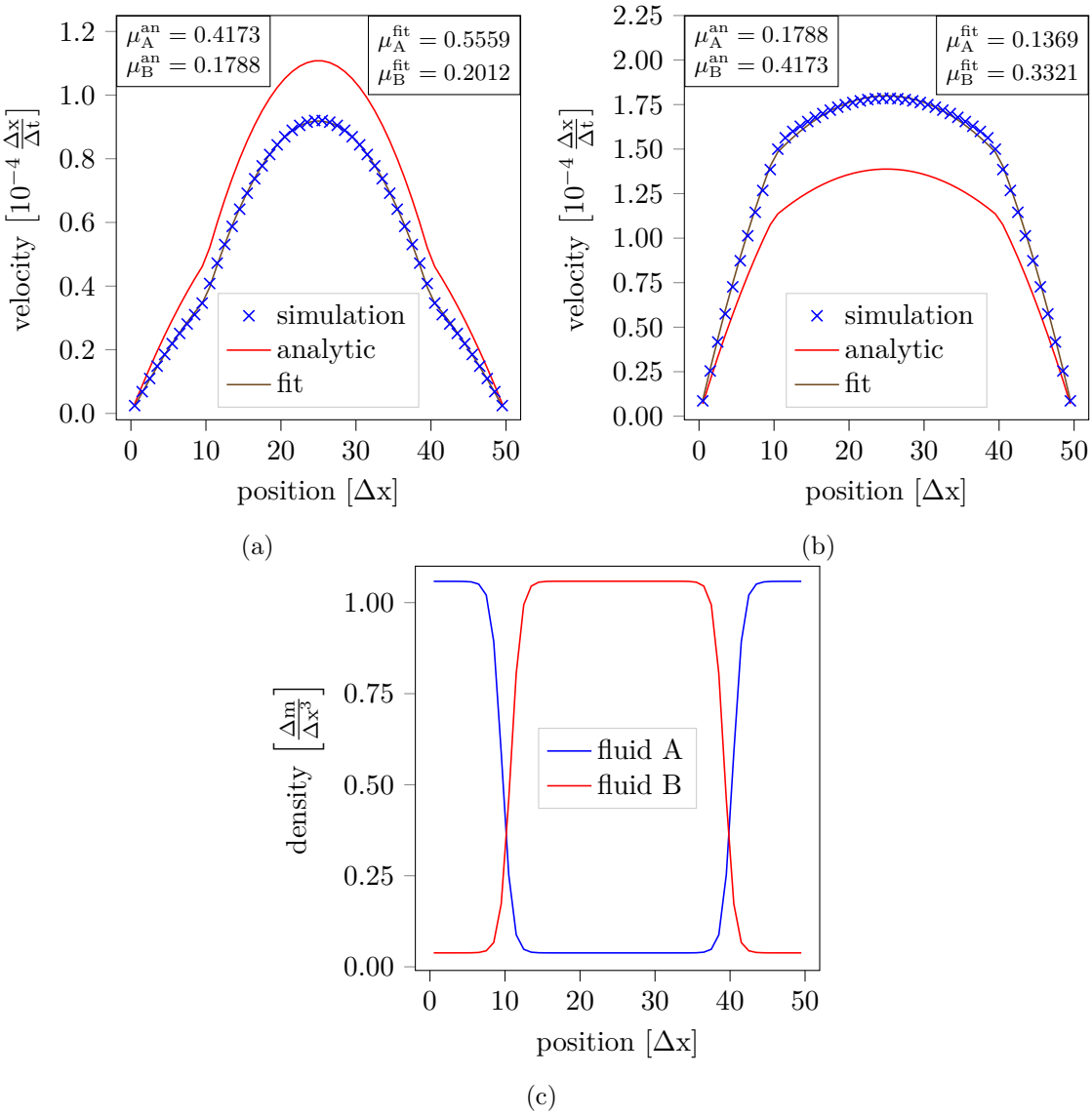


Figure 4.11.: Simulation results of immiscible two-phase Poiseuille profiles with different viscosity ratios. A sketch of the simulation set-up can be seen in Figure 4.8b. (a) shows the velocity profile for the case where the outer fluid in the domain is of higher viscosity and (b) shows the opposite set-up. In (c) the fluid density profiles are shown which is equal for both cases of the viscosity ratios. The used kinematic viscosities are $\nu_1 = \frac{7}{18}\frac{\Delta x^2}{\Delta t}$ and $\nu_2 = \frac{1}{6}\frac{\Delta x^2}{\Delta t}$. The fitted dynamic viscosities are shown in the figures and have the units $[\mu] = \frac{\Delta m}{\Delta x\,\Delta t}$. The discrepancy between the analytical solution and the simulation is due to effective viscosity effects which are the result of the remaining fluid density of the immiscible fluid component.

4.3. Shan-Chen Contact Angle

In simulations of fluid droplets it is often of interest to also include boundaries such as walls and investigate the wetting behavior of fluid droplets. In Section 2.3.3 the treatment of boundaries in the Shan-Chen model was discussed. In this testcase the ability to enforce different contact angles is shown together with the method of measuring the contact angle.

Measurement of the Contact Angle

The measurement of the contact angle of a simulation is a crucial point when discussing the wetting behavior of fluid phases. In the simulations there is no direct access to the surface tensions which is why the contact angle is measured geometrically. There are different approaches to measuring the contact angle but all have in common that the contact line has to be determined. For this the phase field is used together with the *Marching Squares/Cubes* algorithm [32] which is an algorithm of finding points on contour lines or isosurfaces within a domain. Because in this work ideal fluid droplets are simulated one can make use of the analytically known shape of the droplet which forms a sphere cap that minimizes the contact area between the fluid components while keeping the volume constant. Therefore one can make use of this shape by fitting a circle/sphere to the points on the contact line where the center and the radius are the fit parameters. With this information one can calculate the contact angle at the surface analytically by assuming a circle with radius r that intersects a horizontal line. The exemplary set-up where the center of the circle is below the horizontal line is depicted in Figure 4.12a.

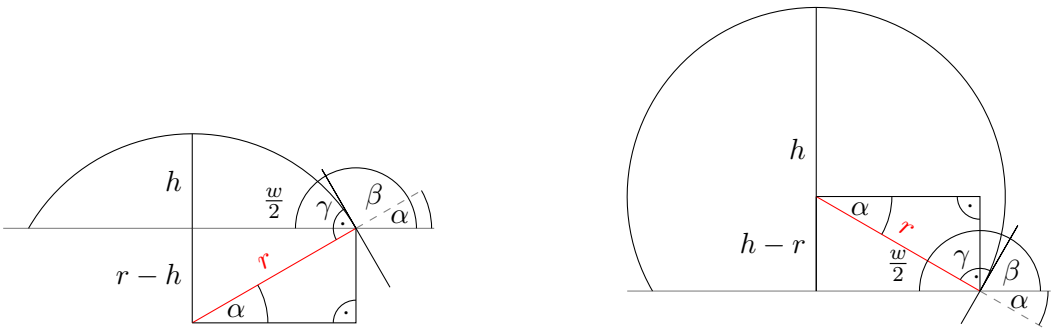


Figure 4.12.: Sketches to visualize the derivation of the contact angle in both the wetting state (a) as well as the non-wetting state (b).

A right triangle can be found from the center of the circle to the intersection point with the horizontal line where the contact angle γ is defined. This can be used to calculate the radius of the circle from the width w and the height h. The corresponding equation reads

$$r^2 = (r-h)^2 + \left(\frac{w}{2}\right)^2 \qquad r = \frac{4h^2 + w^2}{8h}. \tag{4.3.1}$$

The contact angle γ can then be simply expressed in terms of the inner angle α by using the fact that the contact angle is defined between the horizontal line and the tangent line on the circle at the intersection point. The corresponding equations are reading

$$\gamma = \frac{\pi}{2} - \alpha = \frac{\pi}{2} - \arccos\left(\frac{w}{2r}\right) = \frac{\pi}{2} - \arcsin\left(\frac{r-h}{r}\right). \tag{4.3.2}$$

The resulting parameters from the fitting procedure can trivially be used in Equation (4.3.2) to calculate the contact angle. For the case that the center of the circle is above the horizontal line, which corresponds to a droplet in the non-wetting state as shown in Figure 4.12b, the derivation of the equation for the contact angle is similar and results in the same equation. The reason for this is that the part of the circle below the horizontal line is identical to the set-up in Figure 4.12a.

Simulation Results

The described measurement method is applied to a fluid droplet at a wall with different contact angles enforced by tuning the interactions of the fluid components with the wall. The simulation parameters used are listed in Table 4.4. The model used to enforce the contact angles is described in Section 2.3.3. The wall-interactions are chosen to be $G_{\mathrm{WA}} = -G_{\mathrm{WB}}$ equal in magnitude but opposite in sign. One fixed initial set-up is used with different fluid-wall interactions. This procedure is performed for two boundary conditions of the density in the wall because in the literature those are often not clearly specified. The reason for this is that this boundary condition has no physical meaning, and is just an implementation artifact because the force calculation stencil is not altered with the introduction of walls. Therefore the force calculation accesses the density of cells and they have to have some value. The most often boundary condition used in the literature is a Dirichlet boundary condition forcing the non-physical density values in the wall to a fixed value which ensures that the individual force contributions vanish. The second boundary condition is a Neumann boundary condition which is used in this work and forces a vanishing fluid density gradient normal to the wall.

Table 4.4.: Simulation parameters

quantity	symbol	value	unit
domain size		(300, 200)	$\Delta\mathrm{x}$
LB fluid density	ϱ	1	$\frac{\Delta\mathrm{m}}{\Delta\mathrm{x}^3}$
LB relaxation time	τ	1	$\Delta\mathrm{t}$
Shan-Chen interaction	G_{AB}	6	$\frac{\Delta\mathrm{x}^3}{\Delta\mathrm{m}}$

Dirichlet Boundary Condition

For this boundary condition the nonphysical fluid density in the wall is set to zero which is mathematically a Dirichlet boundary condition for the density. The fluid-fluid interaction forces are calculated according to Equation (2.3.3) and the boundary condition removes individual contributions from non-physical fluid cells, which are all wall cells. The result is a large density gradient towards the wall that can cause unwanted model artifacts. For the wetting component the force on the wall is attractive which causes some of the populations to accumulate towards the wall. Because of the bounce-back implementation for the No-Slip boundary condition these populations are reflected back but are attracted by the same force again in the next iteration. This effect causes the populations to accumulate in the first boundary cell towards the wall in the wetting phase, or to deplete in the non-wetting phase. Two snapshots of the simulation are shown in Figure 4.13 showing the fluid droplet in the wetting state 4.13a as well as in a non-wetting state 4.13b.

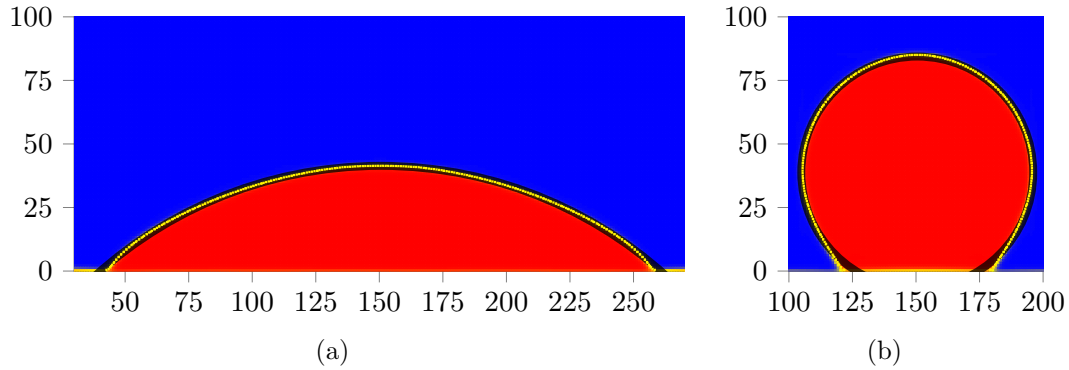


Figure 4.13.: Simulation snapshots showing a part of the simulation domain containing a droplet at different wetting states at a wall using the Dirichlet boundary condition for the fluid density in the wall. The yellow marks show the positions of the surface which are calculated with the marching squares algorithm and the black line is the corresponding fit of a circle which is used for the contact angle calculation.

In the simulations two ideal non-mixing fluids are simulated which is the reason that the expected shape of the interface contour-lines are perfect circles in two dimensions or perfect spheres in three dimensions for the contour-surface. From the Figure 4.13 it is clearly visible that in both cases the contact line does not form a circle which is most noticeable in the area close to the wall. This result is unexpected and causes the circle-fitting method of measuring the contact-angle to produce wrong results.

Neumann Boundary Condition

The Neumann boundary condition for the fluid densities is implemented by a Neumann-by-Copy approach where the density directly in front of the wall-cell is copied into

the wall-cell. This causes the gradient between the two involved cells to vanish. The motivation for using this boundary condition is to eliminate the force contributions coming from the fluid cells near the walls, and not just forcing individual terms to vanish as it is done with the Dirichlet approach. The reason for this boundary condition is that the Shan-Chen force is mathematically a discretization of a gradient as shown in Equation (2.3.4).

Again two snapshots of the simulation for the same simulation parameters as in Figure 4.13 are shown in Figure 4.14.

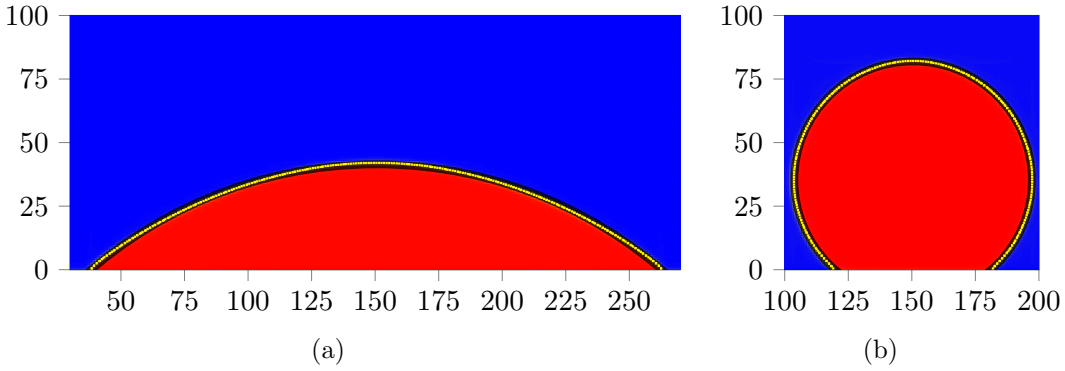


Figure 4.14.: Simulation snapshots showing a part of the simulation domain containing a droplet at different wetting states at a wall using the Neumann boundary condition for the fluid density in the wall. The yellow marks show the positions of the surface which are calculated with the marching squares algorithm and the black line is the corresponding fit of a circle which is used for the contact angle calculation.

In can be seen that the interface points match a circular shape really well as it is expected analytically. For these systems the plots for the measured contact angles above the interaction strength is shown in Figure 4.15. It can be seen that the pseudopotential interaction is capable of producing different wetting states for a fluid droplet corresponding to different contact angles at a wall by using the Neumann boundary condition for the non-physical fluid densities in the wall. Therefore the Neumann Boundary condition is used of the remainder of this thesis, and is the preferred choice for modeling wall-fluid interactions.

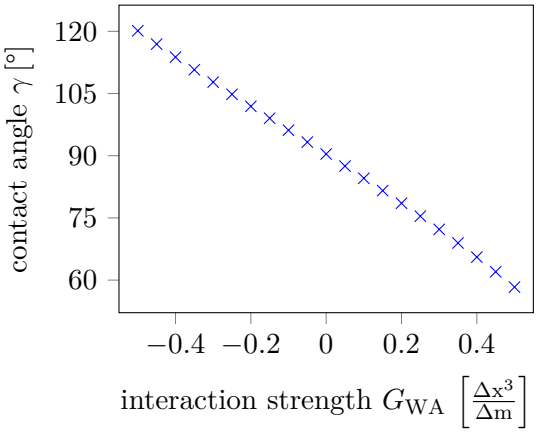


Figure 4.15.: Measured contact angles for different values of the interaction strength with the wall using the Neumann boundary condition for the fluid densities.

4.4. Advection Diffusion

The most simple case to test the lattice electrokinetics implementation for uncharged species is the diffusion of a concentration droplet for which an analytic solution is known. A concentration droplet is set-up in the origin of the domain and the time evolution including the diffusion and the advection of the concentration is described by the advection-diffusion equation in Equation (2.4.1). The equation reads

$$\partial_t n(\vec{x},t) = \vec{\nabla}\cdot\left(D\vec{\nabla}n(\vec{x},t) - \vec{v}n(\vec{x},t)\right) \tag{4.4.1}$$

and can be solved for the Dirac-delta peak at the center of the domain as the fundamental solution using a variety of techniques including Fourier transformation if the velocity is constant within the simulation domain. The solution for d dimensions reads

$$n(\vec{x},t) = \frac{n_0}{(4\pi Dt)^{\frac{d}{2}}}\exp\left(-\frac{(\vec{x}-\vec{v}t)^2}{4Dt}\right). \tag{4.4.2}$$

Because of the discretization in the simulation the fluid droplet is approximated by a single cell which is initially filled with the target concentration. The time evolution of the system is calculated and can be compared to the analytical solution, two snapshots of the simulations are shown in Figure 4.16, the simulation parameters used are listed in Table 4.5. The comparison to the analytical solution is done with a diagonal slice through the center of domain. The resulting positions are shown as offsets to the expected center of mass through the Galilei-transformation. First the data for a pure diffusive system is shown in Figure 4.17a after 100Δt and after 2000Δt in 4.17b. It can be seen that the analytical solution matches to the simulation results nicely. The second simulation is for an advective system as shown in Figure 4.16 and

the comparison after 100Δt is shown in Figure 4.17c and after 2000Δt in 4.17d. In these figures it can be seen that especially for later simulation times the position of the peak is captured very well but the density of the peaks does not match exactly. Because the pure diffusive system is not showing this mismatch it is evident that the reason has to be the discretization of the advective flux as shown in Equation (2.4.15). This artifact is already mentioned in the original paper for the discretization [23]. It is called *spurious diffusion* and is present in all sorts of discretizations for the advective flux. According to the authors of the paper this discretization method already reduces the strength of the spurious diffusion compared to other methods and is negligible for practical applications.

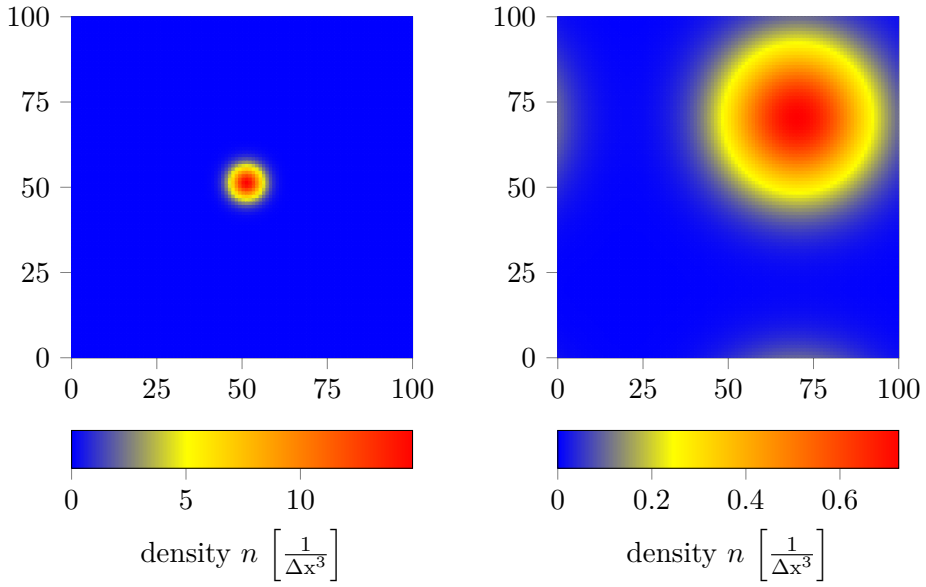


Figure 4.16.: Simulation snapshots of the concentration field for the advection-diffusion testcase after 100Δt in (a) and after 2000Δt in (b). The advective contribution is driving the droplet to the top right and the diffusive contribution is spreading out the concentration. The simulation parameters used are shown in Table 4.5.

Table 4.5.: Simulation parameters used in the advection-diffusion testcase.

quantity	symbol	value	unit
domain size		(100, 100)	Δx
diffusion coefficient	D	0.05	$\frac{\Delta x^2}{\Delta t}$
ion density	n	1000	$\frac{1}{\Delta x^3}$
velocity	$\vec{v}$	(0.01, 0.01)	$\frac{\Delta x}{\Delta t}$
fluxes per cell		4	

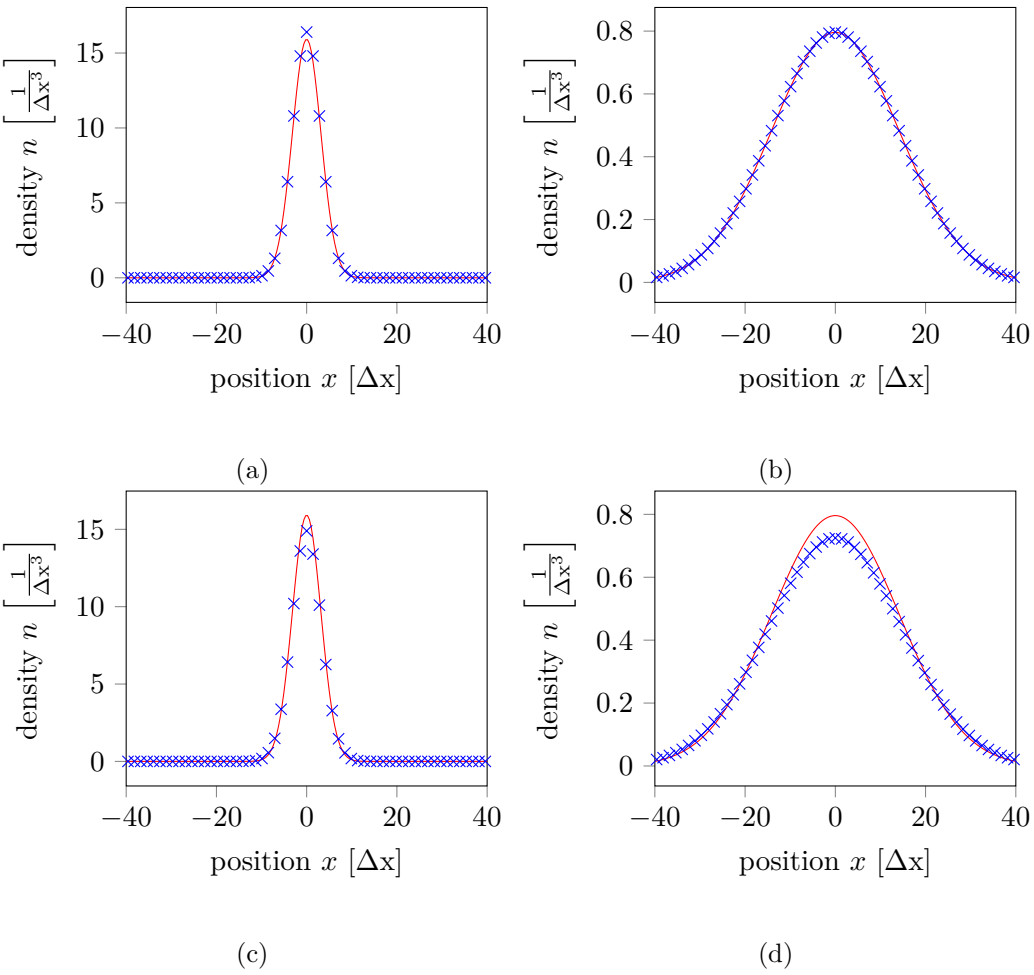


Figure 4.17.: Particle density of a pure diffusive system in (a) and (b). The simulation values are shown as blue crosses and the analytic solution as a red line. In the second row the simulation results for an advective system are shown. The positions are provided in the rest frame of the initial density peak. The mismatch of the densities in the advective system is the result of the discretization of the advective flux term.

4.5. Electroosmotic Flow

The second test set-up using the lattice electrokinetics implementation also includes the coupling of charged species to a single component Lattice Boltzmann fluid. The testcase is known as *Electroosmotic flow* which is set-up in a planar channel. Analytical solutions as testcases for the lattice electrokinetics algorithm are difficult to obtain especially including electrostatics. The corresponding equation describing the analytical electrostatic potential is known as the Poisson-Boltzmann equation which is shown in Equation (4.5.1). It is a well known equation relating the electrostatic potential of charged ion species to their local concentration which is assumed to follow the Boltzmann statistics.

$$\Delta\Phi(\vec{x}) = \frac{ez}{\varepsilon\varepsilon_0} n_0 e^{-\frac{ez\Phi(\vec{x})}{k_\mathrm{B}T}} \tag{4.5.1}$$

$$n(\vec{x}) = n_0 e^{-\frac{ez\Phi(\vec{x})}{k_\mathrm{B}T}} \tag{4.5.2}$$

In the shown formulation only one ionic species with valency z is assumed. Analytical solutions to this equation are only known for very few cases. One of those can be obtained for a system of two infinite planar plates carrying a surface charge density. In this set-up a charge neutral fluid is enclosed within two infinite parallel plates, both charged with the same fixed surface charge density. The counterions for the charged plates are in solution between the plates such that the overall system is charge neutral. A sketch of the system step is shown in Figure 4.18 where the surface charge of the planar walls is shown with the blue stripes and the ions in the solution are shown exemplary as circles but are treated continuously with a number density.

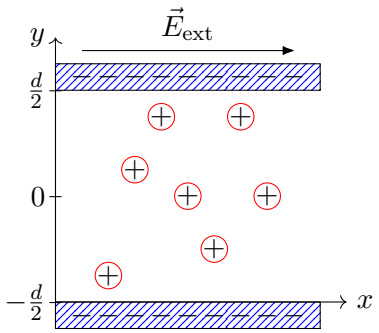


Figure 4.18.: Sketch of the planar channel simulation set-up with the fluid trapped between two charged planar walls. The counterions are shown with circular ions but are calculated with a continuous density approach.

The analytic solution for the density of the ions is determined from the electrostatic potential which is obtained by solving the shown Poisson-Boltzmann equation (4.5.1). For this this system the equation reads

$$\frac{\partial^2 \Phi(y)}{\partial y^2} = -\frac{ez}{\varepsilon\varepsilon_0} n_0 e^{-\frac{ez\Phi(y)}{k_\mathrm{B}T}}. \tag{4.5.3}$$

The reason the problem reduces to the one-dimensional case is that the system is translational invariant in all directions except for the one normal to the plates, which forces the analytical solution to also obey these invariances. Even with this reduction of the complexity the analytic solution is not trivial to obtain. The derivation of the solution can be found in [33, p. 30], in this work only the solutions are shown, which read

$$\Phi(y) = -\frac{k_\mathrm{B}T}{ez} \log\left[\frac{\varepsilon\varepsilon_0 C^2}{2n_0 k_\mathrm{B}T} \frac{1}{\cos^2\left(\frac{ezC}{2k_\mathrm{B}T}y\right)}\right], \tag{4.5.4}$$

for the electrostatic potential and the ion distribution is given by inserting the obtained solution in the definition of the density in Equation (4.5.2)

$$n(y) = \frac{\varepsilon\varepsilon_0 C^2}{2k_\mathrm{B}T} \frac{1}{\cos^2\left(\frac{ezC}{2k_\mathrm{B}T}y\right)} \qquad \frac{|zy|\, eC}{2k_\mathrm{B}T} < \frac{\pi}{2}. \tag{4.5.5}$$

The solution equations include a variable C which has to be determined using the transcendental equation

$$C \tan\left(\frac{zed}{4k_\mathrm{B}T} C\right) = -\frac{\sigma}{\varepsilon\varepsilon_0}, \tag{4.5.6}$$

which is related to the electrostatic boundary condition and has to be solved numerically for the specific set of parameters of the simulation. From this stationary state of the ions, the system can now be exposed to an external electric field which is aligned parallel to the plates. This field drives the ions in the solution, which also causes the fluid to move because of the friction coupling that drags the fluid along with the ions. The external applied electric field does not influence the previous solution of the density profile due to the invariance of the profile in the direction the field is applied. Therefore the applied force due to the friction coupling is fully determined from the external applied field and in consequence only dependent on the local charge density of the ions which is known from Equation (4.5.5). This allows the Stokes equation for the velocity profile to the written as

$$\frac{\partial^2 v_x(y)}{\partial y^2} = -\frac{q(y)E}{\mu}, \tag{4.5.7}$$

where μ is the dynamic viscosity of the fluid and $v_x(y)$ the velocity profile of the fluid. The result can be calculated by integrating Equation (4.5.7) twice and eliminating the two integration constant by enforcing the boundary conditions

$$v_x(\pm d/2) = 0. \tag{4.5.8}$$

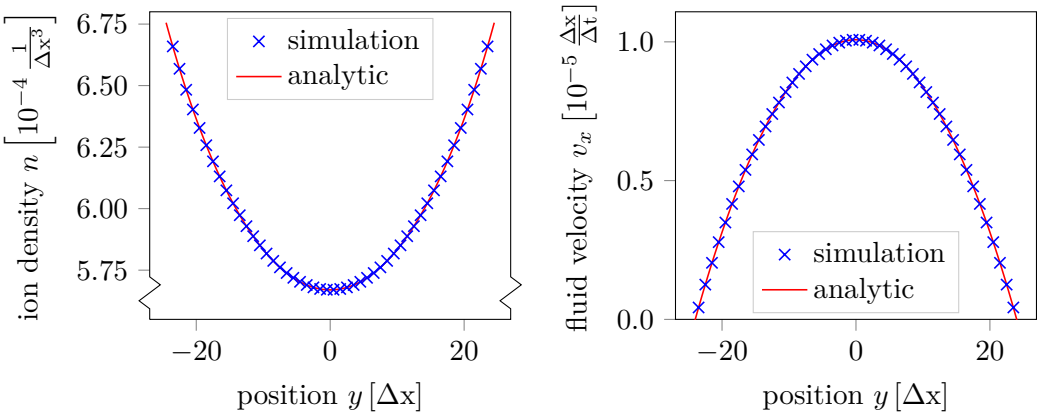
The resulting velocity profile is given by

$$v_x(y) = \frac{2E\varepsilon_0\varepsilon k_\mathrm{B}T}{\mu q} \log\left[\frac{\cos\left(\frac{qC}{2k_\mathrm{B}T}y\right)}{\cos\left(\frac{qC}{2k_\mathrm{B}T}\frac{d}{2}\right)}\right]. \tag{4.5.9}$$

This analytic solution is compared to the simulation results which is obtained using the previously described models. The system is simulated in two dimensions where one of them is periodic. For the fluid a single phase Lattice Boltzmann fluid is used which is enclosed within two walls which are prescribed using the No-Slip boundary condition. For the ions only one species is simulated which is trapped inside the simulation domain by No-Flux boundary conditions at the walls which are also impenetrable for the fluid. The surface charge is explicitly placed on the walls. For the electrostatic potential the Poisson equation is solved using a SOR-solver for which a Neumann-Boundary condition is imposed for the first layer inside the wall. The boundary condition imposed for the Poisson equation is in this case not of relevance, a constant potential boundary condition instead of the Neumann one gives the same result. The reason for this is the symmetry of the problem with respect to the channel center which requires the electrostatic potential to be equal at the boundaries and therefore a Dirichlet constant potential boundary condition is just a shift in the potential compared to the Neumann boundary condition. The result of the density profile together with the analytic solution can be seen in Figure 4.19a. The same figure set-up for the velocity profile is shown in Figure 4.19b. Both results agree very well with the calculated analytical solution. The parameters used for the simulation are provided in Table 4.6 but the agreement with the analytical curve is not restricted to the parameter set provided here.

Table 4.6.: Simulation parameters used in the electroosmotic flow testcase. The resulting density profile is shown in Figure 4.19a and the velocity profile in Figure 4.19b.

quantity	symbol	value	unit
domain size		(6, 48)	$\Delta\mathrm{x}$
diffusion coefficient	D	0.25	$\frac{\Delta\mathrm{x}^2}{\Delta\mathrm{t}}$
energy unit	$k_\mathrm{B}T$	1	$\Delta\mathrm{E}$
valency	z	1	
vacuum permittivity	ε_0	1	$\frac{e^2\,\Delta\mathrm{t}^2}{\Delta\mathrm{m}\,\Delta\mathrm{x}^3}$
relative permittivity	ε	1	
ion density	n	0.0006	$\frac{1}{\Delta\mathrm{x}^3}$
surface charge density	σ	0.0144	$\frac{e}{\Delta\mathrm{x}^2}$
external electric field	E	-10^{-5}	$\frac{\Delta\mathrm{m}\,\Delta\mathrm{x}}{\Delta\mathrm{t}^2\,e}$
LB fluid density	ϱ	1	$\frac{\Delta\mathrm{m}}{\Delta\mathrm{x}^3}$
LB relaxation time	τ	1	$\Delta\mathrm{t}$



(a) (b)

Figure 4.19.: Simulation results together with the analytical solution for the electroosmotic flow testcase. A sketch of the simulation set-up can be seen in Figure 4.18. In (a) the density profile and in (b) the velocity profile of the system is shown. The electrostatic potential is calculated with a Neumann boundary condition for the potential inside the wall. The fluid is coupled to the ions with a friction coupling and driven with an external electric field. The simulation parameters used are shown in Table 4.6.

4.6. Heaviside Separation

All the previously described cases only test parts of the algorithm which are either the fluid itself or one component fluids with ions but never include multiple fluid components with ions. Therefore a new testcase is developed which includes multiple fluid components as well as the couplings to the ions [27], and still can be solved analytically. The analytic system described in [34] is a one-dimensional problem which is set-up with two solvents A and B which are present in separate half spaces. If the interface is positioned at $x = 0$, the half-space $x < 0$ is fully occupied by solvent A and $x > 0$ by solvent B. Each of the solvents has its respective dielectric permittivity $\varepsilon_{\mathrm{A/B}}$. In the system monovalent anions and cations are present as solutes with the bulk densities $n_{\mathrm{A/B}}$ in the respective solvents. The solvation free energy difference for the ionic species between the two solvents is given by $\Delta\mu_{+/-}$. A schematic representation of the system is shown in Figure 4.20.

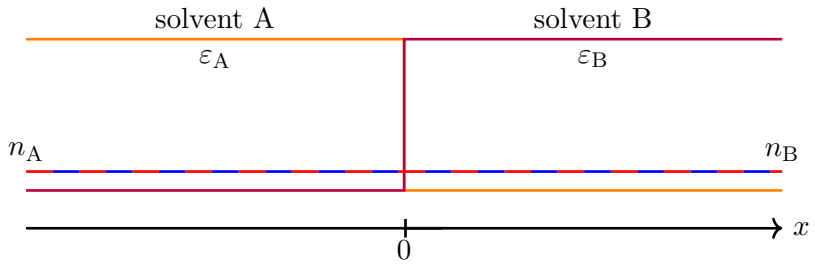


Figure 4.20.: Schematic representation of the one-dimensional system set-up. The solvent phases are shown in orange and purple and are present in separate half-planes. The ion number densities are shown in red and blue for the anions and cations, respectively.

In [34] this systems is solved analytically using a classical density functional theory approach that is fundamentally solving the very same equations as the discretization for the ionic species used in this work. For that a piecewise Debye screening factor will be defined which reads

$$\kappa^2(x) = \frac{e^2 z^2}{\varepsilon_0 k_{\mathrm{B}} T} \begin{cases} \frac{n_A}{\varepsilon_A} & x < 0 \\ \frac{n_A}{\varepsilon_B} & x \in (0, s) \\ \frac{n_B}{\varepsilon_B} & x > s \end{cases} = \begin{cases} \kappa_A^2 & x < 0 \\ \kappa_i^2 & x \in (0, s) \\ \kappa_B^2 & x > s \end{cases} \tag{4.6.1}$$

where e is the elementary charge, $z = 1$ the valency and s is a free parameter which is defining a length that can be used to capture interfacial smoothing effects. This parameter is relevant if the fluid density profile or the dielectric permittivity profile deviates from the analytical assumed Heaviside-function. From the free energy density formulation as well as the electrostatic Poisson equation the linearized Poisson-Boltzmann equation for the shifted electrostatic potential can be derived

$$\frac{\partial^2(\Phi(x)-\Phi_D\Theta(x-s))}{\partial x^2}=\kappa^2(x)(\Phi(x)-\Phi_D\Theta(x-s)) \qquad x\notin\{0,s\}\,, \tag{4.6.2}$$

where $\Theta(x)$ is the Heaviside function and $\Phi_D=\frac{\Delta\mu_+-\Delta\mu_-}{2}$ is known as the Donnan-potential. The solution to this equation with the piecewise definition of the screening factors in Equation (4.6.1) can be found in [34] and reads

$$\Phi(x)=\frac{\Phi_D}{D}\begin{cases}\exp\left(\kappa_A x\right) & x<0\\ \cosh\left(\kappa_i x\right)+\sqrt{\frac{\varepsilon_A}{\varepsilon_B}}\sinh\left(\kappa_i x\right) & x\in(0,s)\\ -\exp\left(-\kappa_B\left(x-s\right)\right)\sqrt{\frac{n_A}{n_B}}\left(\sqrt{\frac{\varepsilon_A}{\varepsilon_B}}\cosh\left(\kappa_i s\right)+\sinh\left(\kappa_i s\right)\right) & x>s\end{cases} \tag{4.6.3}$$

with $D=\left(1+\sqrt{\frac{\varepsilon_A n_A}{\varepsilon_B n_B}}\right)\cosh(\kappa_i s)+\left(\sqrt{\frac{\varepsilon_A}{\varepsilon_B}}+\sqrt{\frac{n_A}{n_B}}\right)\sinh(\kappa_i s)$.

The analytical solution is compared with two separate simulation set-ups which are shown in the following. The first system is set-up in two dimensions where half of the system is filled with fluid component A and the other half with component B. The simulation domain is rectangular with the side-length of interest chosen to be 100Δx in length which is non-periodic. The other direction has 6Δx nodes in length and is chosen to be periodic. The boundaries of the domain in the non-periodic direction is set-up with No-slip boundary conditions for the fluid components which is supposed to keep the fluid in place. The bulk values for the ion densities are chosen equal for both components and are distributed uniformly in the simulation domain. The non-periodic boundary is treated with a Dirichlet boundary condition for the ion densities which is set to the bulk-density. The electrostatic potential is solved using the SOR-method with a Neumann-boundary on the domain edges to enforce a vanishing electric field. The Gibbs transfer energy $\Delta\mu$ is chosen to be equal in magnitude but opposite in sign between the ion species and different values are simulated. All relevant simulation parameters are listed in Table 4.7.

The simulations are run for $10^6\Delta$t timesteps and the ion density profiles are plotted together with the corresponding analytic solutions. The case of equal dielectric permittivity for both solvents the solutions are shown in Figure 4.21 where for all the analytical solution $s=0$ was used.

The results for the more complex case of a spatial varying dielectric permittivity $\varepsilon(\vec{x})$ are shown in Figure 4.22 where again $s=0$ is used for the analytic solution.

This set-up is also simulated in a periodic set-up to also test the implementation of the FFT-based Poisson solver. Therefore a larger system is set-up in fully periodic boundary conditions where now the number of ions in the system is fixed. This set-up now features two of the fluid interfaces due to the period boundary conditions which have to be apart far enough to reduce the influence on each other. The parameters which differ from the previous simulation (Table 4.7) are provided in Table 4.8. From the two forming interfaces only the one in the center of the domain is observed and the results are shown in Figure 4.23.

Table 4.7.: Simulation parameters used in the Heaviside density simulation set-up with the non-periodic set-up using the SOR-solver for the Poisson equation.

quantity	symbol	value	unit
domain size		(100, 6)	$\Delta\mathrm{x}$
diffusion coefficient	D	0.0070	$\frac{\Delta\mathrm{x}^2}{\Delta\mathrm{t}}$
energy unit	$k_\mathrm{B}T$	1	$\Delta\mathrm{E}$
valency	z	1	
vacuum permittivity	ε_0	0.0014	$\frac{e^2\,\Delta\mathrm{t}^2}{\Delta\mathrm{m}\,\Delta\mathrm{x}^3}$
relative permittivity	ε	80	
ion density	n	0.0006	$\frac{1}{\Delta\mathrm{x}^3}$
LB fluid density	ϱ	20	$\frac{\Delta\mathrm{m}}{\Delta\mathrm{x}^3}$
LB relaxation time	τ	1	$\Delta\mathrm{t}$
Shan-Chen interaction	G_AB	0.3	$\frac{\Delta\mathrm{x}^3}{\Delta\mathrm{m}}$

Table 4.8.: Simulation parameters used in the Heaviside density simulation set-up with fully periodic boundary conditions which are different from the ones provided in Table 4.7. The Poisson equation is solved with the FFT-based Poisson solver.

quantity	symbol	value	unit
domain size		(500, 5)	$\Delta\mathrm{x}$

All the shown results, especially those who a calculated with the SOR-based solver, show a very good agreement with less than 5 % deviation to the analytical solution except for the interface area. In the order of $\mathcal{O}\left(\pm 3\Delta\mathrm{x}\right)$ the diffusive fluid-fluid interface is present that, together with the discretization artifacts, causes a smearing effect on the ion-density between the fluid components. Another reason for deviations is that the analytical solution only solves the linearized Poisson-Boltzmann equation whereas the simulation is solving the non-linear Poisson-Boltzmann equation. This difference is especially relevant at the fluid-fluid interface where non-linear effects are expected to have the largest influence. To counteract this contribution the ion density is chosen relatively low such that non-linear effects are less prominent but are expected to increase with larger differences in the solvation free energy as well as with larger ion densities.

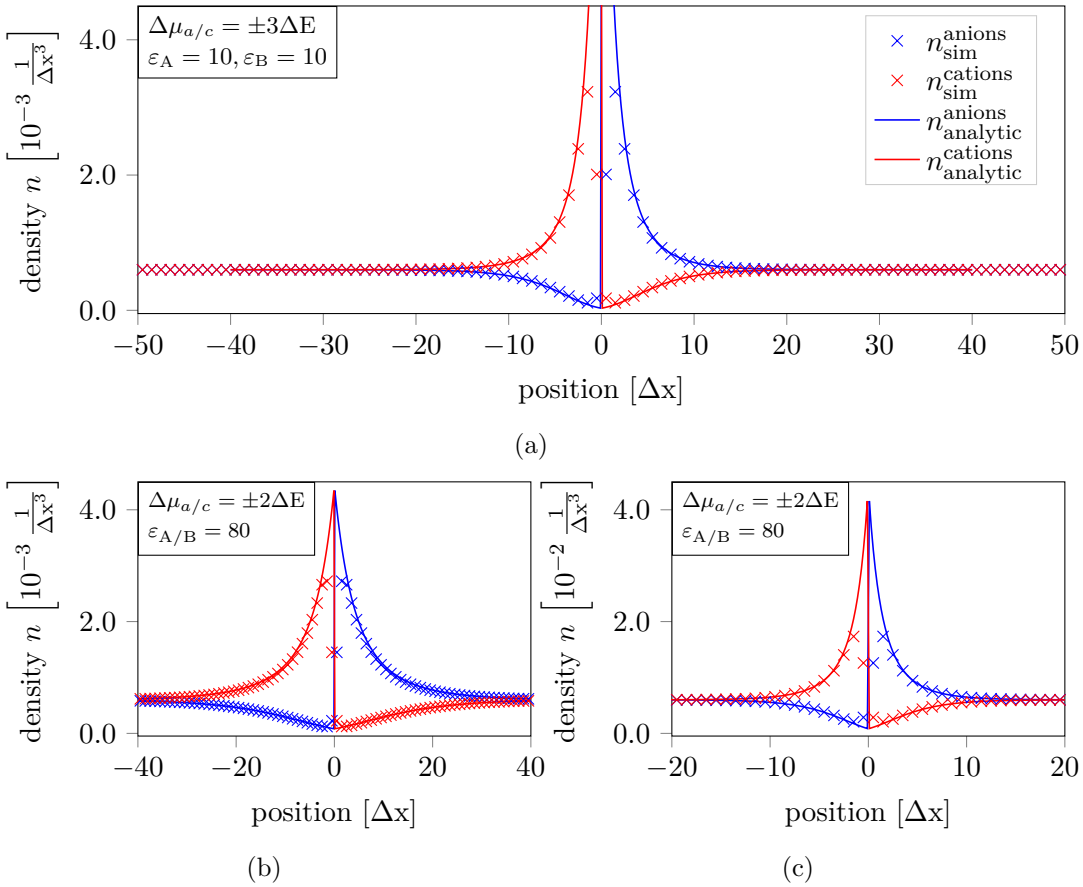


Figure 4.21.: Simulation results of the Heaviside simulation set-up with constant dielectric permittivity ε which is schematically shown in Figure 4.20. The local anion and cation density is shown for each position. In the different subfigures the simulation parameters are changed which involve the ion density n, the dielectric permittivity ε and the Gibbs-transfer energy $\Delta\mu$. The simulation values are shown as marks and the analytical solution as continuous lines with the same color-coding in all shown plots. All non-explicitly provided simulation parameters used are shown in Table 4.7.

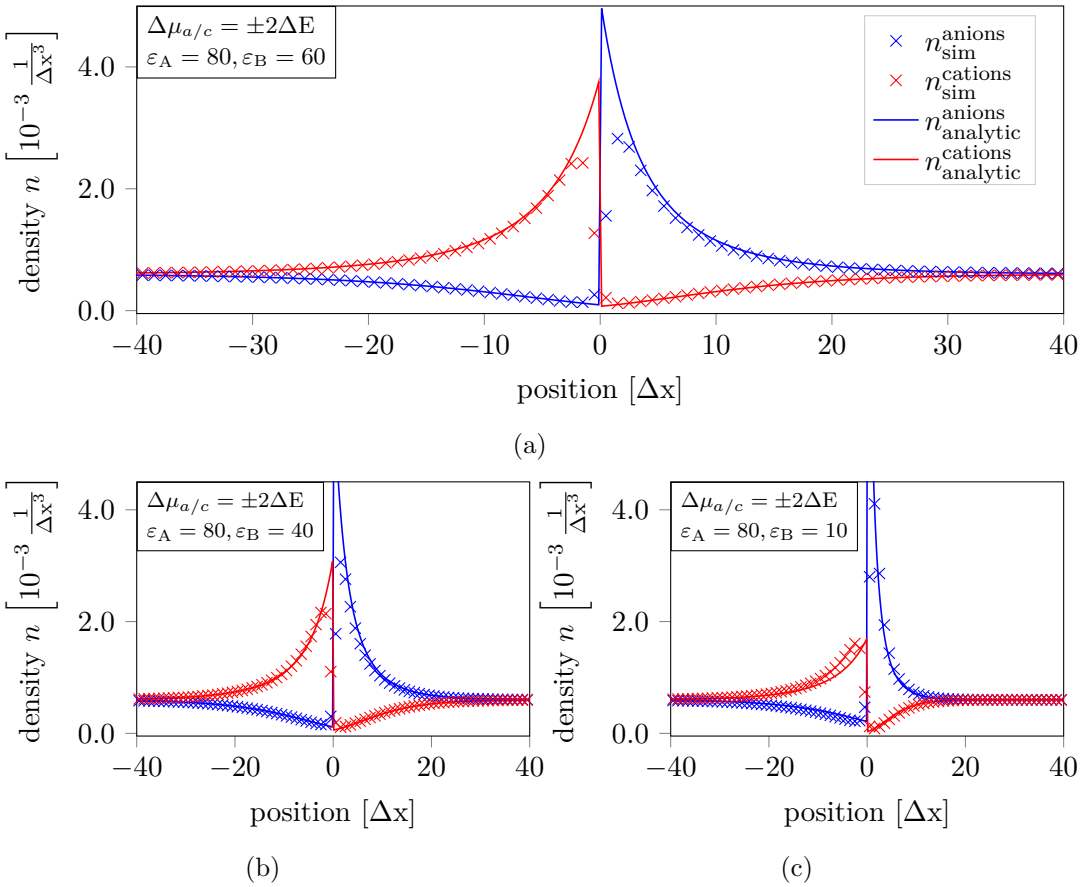


Figure 4.22.: Simulation results of the Heaviside simulation set-up with spatially varying dielectric permittivity $\varepsilon(x)$ which is schematically shown in Figure 4.20. The local anion and cation density is shown for each position. In the different subfigures the simulation parameters are changed which involve the ion density n, the dielectric permittivity ε and the Gibbs-transfer energy $\Delta\mu$. The simulation values are shown as marks and the analytical solution as continuous lines with the same color-coding in all shown plots. All non-explicitly provided simulation parameters used are shown in Table 4.7.

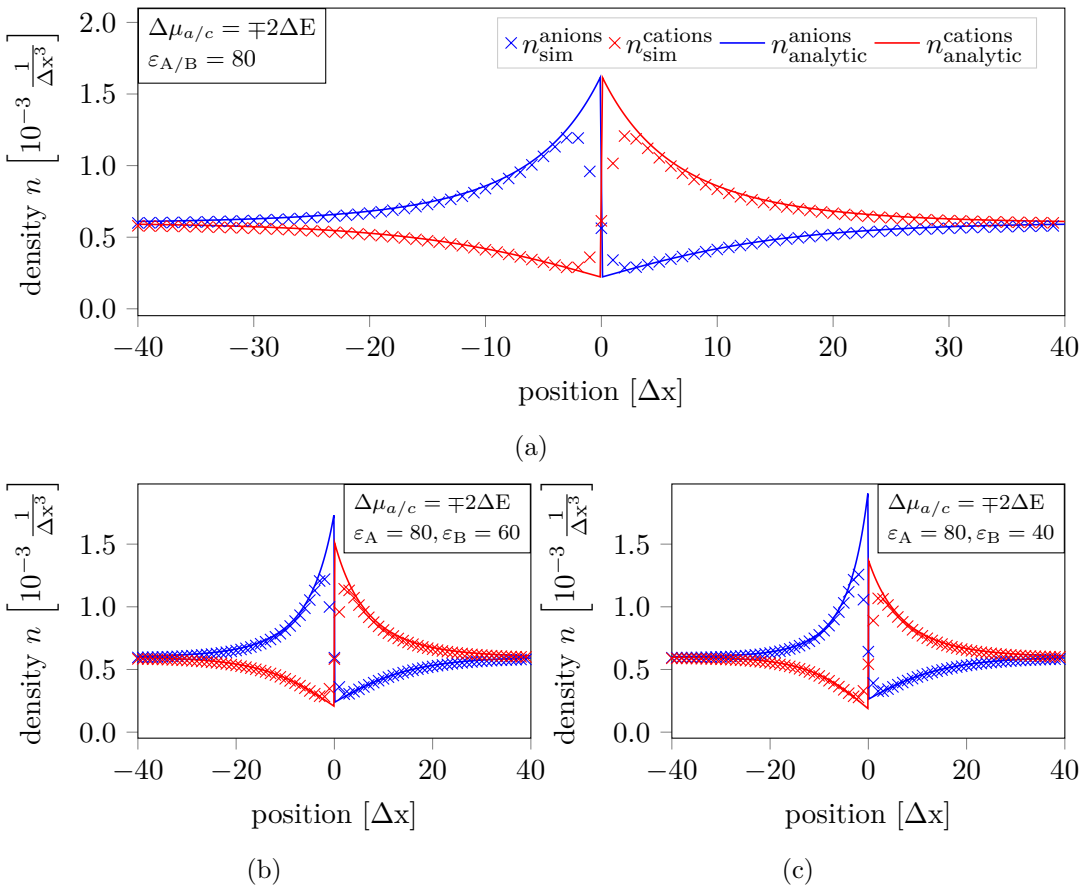


Figure 4.23.: Simulation results of the Heaviside simulation set-up with spatially varying dielectric permittivity $\varepsilon(x)$ in fully periodic boundary conditions. The set-up is schematically shown in Figure 4.20. The local anion and cation density is shown for each position at one of the two interfaces. In the different subfigures the the dielectric permittivity ε of one component is changed. The simulation values are shown as marks and the analytic solutions as continuous lines with the same color-coding in all shown plots. All non-explicitly provided simulation parameters used are shown in Table 4.8.

4.7. Electrophoretic Mobility

The final testcase of the code implemented in this work includes the full coupling of all algorithms implemented as in the previous testcase. The set-up of the system is again a fluid droplet embedded in an immiscible fluid component with a uniform ion distribution within the system. The fluid set-up is similar to the one used in testcase 4.1 with fully periodic boundary conditions. A Gibbs transfer energy $\Delta\mu_{a/c}$ is imposed for the ions in such a way, that a specific charge is trapped within the droplet. A sketch of the set-up is shown in Figure 4.24.

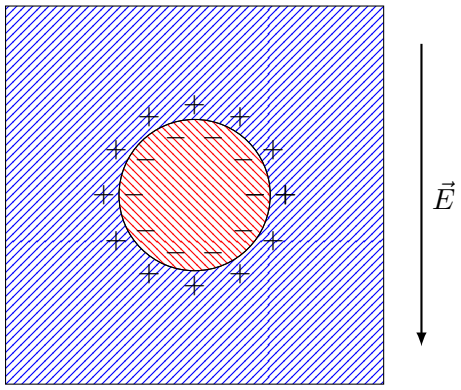


Figure 4.24.: Sketch of the system set-up for the measurement of the electrophoretic mobility with a Gibbs transfer energy imposed on the ions to effectively charge the fluid droplet. The different colors correspond to different fluid components. The droplet is then driven with an external electric field.

The simulation parameters used are listed in Table 4.9 and are chosen to correspond to a grid size of $\Delta x = 1\,\mathrm{nm}$, a salt concentration of 1 mM, a fluid density of $10^3\,\frac{\mathrm{kg}}{\mathrm{m}^3}$ and a initial droplet charge of $-1\,e$. This set-up is equilibrated and used as the starting point for simulations with different external electric fields applied to the system. Since the droplet aquires a net charge, it is accelerated through the simulation domain up to a final velocity which is determined from the balance of friction force and the acceleration force $\gamma\vec{v} = m\vec{a}$. This velocity is measured for different applied electric fields and charges of the droplet. For sufficiently small applied electric fields one expects a linear response of the terminal velocity on the applied electric field,

$$\vec{v}_{\mathrm{terminal}} = \mu\vec{E}. \tag{4.7.1}$$

The ion density profile of the equilibrated set-up without an external electric field is shown in Figure 4.25 together with the concentration curves calculated with COMSOL. COMSOL is a widely used physical simulation package solving continuity equations using the finite element method. The set-up used is slightly different and consists of a non-penetrable sphere in the center of the domain with an imposed surface charge which is comparable to the simulated system for the area outside the sphere.

The reason is that in this system the concentration curves are only influenced by electrostatic forces, which are identical for a sphere with fixed surface charge, a homogeneous volume charge density and a point charge. The ion distribution within the droplet shows two density peaks at the interface to the surrounding fluid component. The reason for these peaks are that the ions are trapped within the droplet because of the Gibbs transfer energy. Inside the droplet electrostatic forces drive the ions apart which results in the characteristic density peaks towards the surface of the droplet. The measured terminal velocity of the droplet is shown in Figure 4.26 for two charges of the droplet. From the plot it can be seen that the terminal velocities match to the proportionality curve as expected. The resulting mobilities can also be compared to measurements from the COMSOL simulation. When comparing these values one has to be aware of the differences in the simulations. The COMSOL system consists of a hard-sphere with no-slip boundary conditions for the fluid on the surface, which are different boundary conditions than for a physical fluid droplet that allows for a surface velocity. Because of this difference it is expected that the COMSOL mobilities are smaller when compared to he ones measured in Figure 4.26. The measured mobilities read $\vec{v}_{\text{terminal}}^{\text{COMSOL}} = 0.001\,39\,\frac{\Delta\text{t}\,e}{\Delta\text{m}}$ for $q = -1\,e$ and $\vec{v}_{\text{terminal}}^{\text{COMSOL}} = 0.002\,50\,\frac{\Delta\text{t}\,e}{\Delta\text{m}}$ for $q = -2\,e$. As expected, both of these values are slightly smaller.

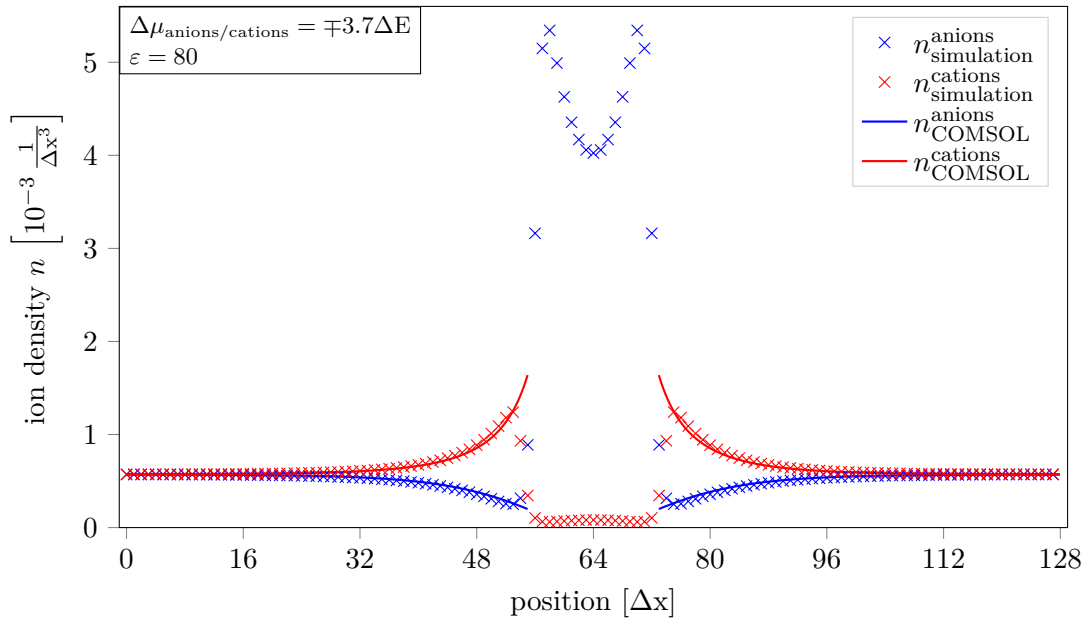


Figure 4.25.: Ion density profiles of a slice through the center of the droplet without external electric field. The continuous lines are calculated using COMSOL with a fixed surface charge density $q_{\text{surface}} = 0.0176\,\frac{e}{\text{nm}^2}$ on a sphere of radius $R = 9\,\text{nm}$ in the center of the domain.

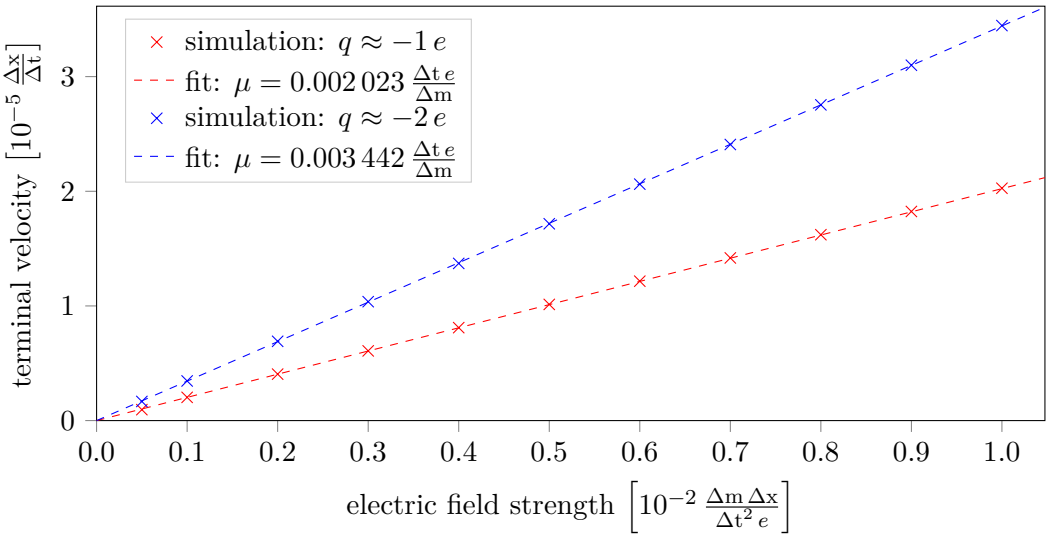


Figure 4.26.: Measured terminal velocity of the droplet above the applied electric field strength for the simulation. The mobility is the proportionality constant measured with a least-squares fit.

Table 4.9.: Simulation parameters used in the electrophoretic mobility testcase.

quantity	symbol	value	unit
domain size		(128, 128)	Δx
diffusion coefficient	D	0.0070	$\frac{\Delta x^2}{\Delta t}$
energy unit	$k_B T$	1	ΔE
valency	z	1	
vacuum permittivity	ε_0	0.0014	$\frac{e^2\,\Delta t^2}{\Delta m\,\Delta x^3}$
relative permittivity	ε	80	
ion density	$n_{a/c}$	0.0006	$\frac{1}{\Delta x^3}$
LB fluid density	$\varrho_{A/B}$	20	$\frac{\Delta m}{\Delta x^3}$
LB relaxation time	$\tau_{A/B}$	9.8619	Δt
Shan-Chen interaction	G_{AB}	0.3	$\frac{\Delta x^3}{\Delta m}$
Gibbs transfer energy	$\Delta\mu_{a/c}$	$\mp 3.7, \mp 5.5$	ΔE

5. Application: Salt-dependent Contact Angle

In Section 2.1 the altering of the wettability of oil droplets in the vicinity of brines of different salinity was introduced which is experimentally observed in [2]. The implementation of the different models was done with the goal of observing the wettability changes in the simulation. To achieve this, an oil droplet is simulated as two non-compressible immiscible Lattice Boltzmann fluids using the Shan-Chen model with one component initially placed as a half sphere at the wall and the other component filling the remaining outer space. The different wetting states are enforced with the pseudopotential interaction of the fluid with the wall as introduced in Section 2.3.3 which was shown to achieve different contact angles in the testcase in Section 4.3 by imposing a non-physical Neumann boundary condition for the density at the wall in the force-calculation. To include the ions in the simulation the lattice electrokinetic model is used. It is initially set-up with two separate ion species for monovalent anions and cations which are initially configured uniformly in the simulation domain. For the boundary conditions the wall is placed at the bottom of the domain using a No-Slip boundary condition for the Lattice Boltzmann fluid populations, the aforementioned Neumann boundary condition for the fluid densities, the Shan-Chen boundary force for the wettability and the No-Flux boundary condition for each of the ion species. For the fluids the same boundary conditions are used on the opposite side of the domain and a constant density boundary condition is imposed for the ions to provide a specific ion density in the outer bulk phase of the ions. In the remaining direction a periodic boundary condition is used.

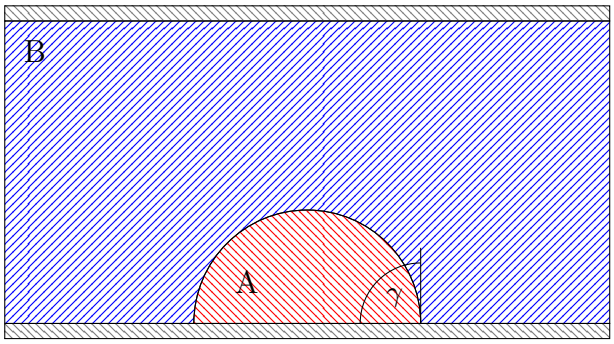


Figure 5.1.: Sketch of a slice through the center of the droplet of the initial simulation set-up. The ions are not shown but are distributed uniformly in the simulation domain.

For these systems the well known *Young equation* [35] describes the relation between the contact angle and the surface tensions in the system. It reads

$$\cos(\gamma) = \frac{\sigma_{\mathrm{WB}} - \sigma_{\mathrm{WA}}}{\sigma_{\mathrm{AB}}}, \tag{5.1}$$

where σ_{WB} is the surface tension between the surrounding fluid component and the wall, σ_{WA} the surface tension between the droplet and the wall and σ_{AB} the surface tension between the two fluid components. From this equation it can be seen that the sign of the difference between the surface tensions with the wall are the factor determining if the droplet equilibrates towards a wetting or a non-wetting state.

In the following simulations the Gibbs transfer energy for the ions is always chosen equal for both species to avoid the droplet and prefer the surrounding fluid component. Because all parameters for the ions are equal, the behavior is also equal which means that the system is electroneutral for every grid cell. Therefore electrostatics does not contribute to the forces in this system. At first a reference simulation is performed without ions for different contact angles. Afterwards the system is simulated for different ion concentrations and different Gibbs transfer energies. In each simulation run, the contact angle is measured using the Marching Squares algorithm to find the fluid-fluid interface as described in Section 4.3.

These simulations are specifically set-up in such a way, that the ions avoid the droplet but are uniformly distributed in the outer fluid component. Assuming that the ions do not influence the surface-tensions of the fluids with the wall σ_{WA} and σ_{WB}, the only influence of the ions is a density gradient at the fluid-fluid interface. This gradient results in an additional normal force only on the fluid-fluid interface which is equivalent to an increase in the fluid-fluid surface tension

$$\sigma_{\mathrm{AB}}^{\mathrm{new}} = \sigma_{\mathrm{AB}} + \sigma^*, \tag{5.2}$$

where $\sigma_{\mathrm{AB}}^{\mathrm{new}}$ is the updated surface tension, σ_{AB} the surface tension for the system without ions and σ^* is the influence of the ion interaction. The contact angle for the new surface tension $\sigma_{\mathrm{AB}}^{\mathrm{new}}$ is given by the Young-equation (5.1) and reads

$$\cos(\gamma_0 + \Delta\gamma) = \frac{\sigma_{\mathrm{WB}} - \sigma_{\mathrm{WA}}}{\sigma_{\mathrm{AB}}^{\mathrm{new}}} = \frac{\sigma_{\mathrm{WB}} - \sigma_{\mathrm{WA}}}{\sigma_{\mathrm{AB}} + \sigma^*}, \tag{5.3}$$

where γ_0 is the reference contact angle without the ions in the system and $\Delta\gamma$ is the contact angle change. Rearranging Equation (5.1) to eliminate the fluid surface tensions with the wall $\sigma_{\mathrm{WA}} - \sigma_{\mathrm{WB}}$ and substituting it in Equation (5.3) allows for the separation of the contact angle difference $\Delta\gamma$

$$\Delta\gamma(\gamma_0) = \arccos\left(\cos(\gamma_0)\frac{\sigma_{\mathrm{AB}}}{\sigma_{\mathrm{AB}} + \sigma^*}\right) - \gamma_0. \qquad \gamma_0 \in [0°, 180°] \tag{5.4}$$

To show that this function is odd with respect to the reference contact angle $\gamma_0 = 90°$ the substitution $\gamma_0 = 90° + \gamma'$ can be made. Using the relations $\cos(90° + x) = -\sin(x)$ and $\arccos(-x) = 90° + \arcsin(x)$ the equation can be rewritten

$$\Delta\gamma(\gamma') = \arcsin\left(\sin(\gamma')\frac{\sigma_{\mathrm{AB}}}{\sigma_{\mathrm{AB}}+\sigma^*}\right) - \gamma' \qquad \gamma' \in [-90°, 90°]\,. \tag{5.5}$$

The oddness of the contact angle difference can easily be seen by plugging $-\gamma'$ in the equation and using $\sin(-x) = -\sin(x)$ and $\arcsin(-x) = -\arcsin(x)$. As expected the result is $\Delta\gamma(-\gamma') = -\Delta\gamma(\gamma')$. Qualitatively this means that with the introduction of the ions the contact angle of the droplet is shifted towards 90° in the wetting as well as in the non-wetting case. Furthermore, one expects the additional surface tension contribution σ^* to increase with the ion density as well as with the Gibbs-transfer energy which also results in an increase in the contact angle change $\Delta\gamma$.

The contact angle measured in the simulations for different ion densities and Gibbs-transfer energies are shown in Figure 5.2a. The simulation parameters used are listed in Table 5.1. Because the differences in the measured contact angles are rather small, a different visualization is added that corresponds to the analytic expression in Equation (5.4) for the difference in the contact angles. For the simulation data this reads

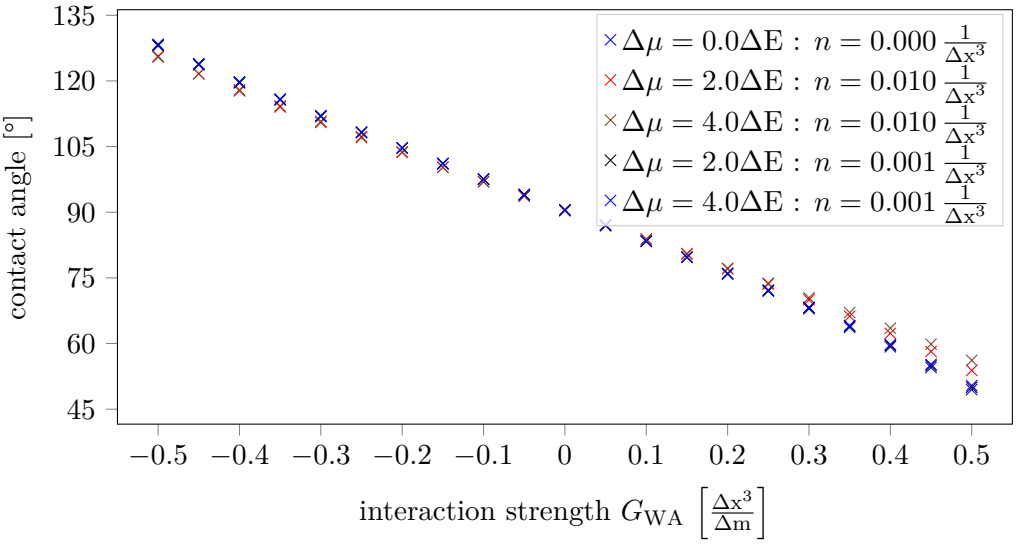
$$\Delta\gamma = \gamma - \gamma_0. \tag{5.6}$$

This quantity is shown above the reference contact angle in Figure 5.2b. From this figure it can be seen that results are qualitatively matching to the expected analytic result, the expected zero crossing at the 90°-simulations are recovered, the small mismatches for some of the simulations are artifacts of the lattice resolution. Because the change in the surface tension σ^* only depends on ion parameters and not on the reference contact angle the shown point symmetry of the contact angle difference in Equation (5.5) is also expected in the curves for a fixed Gibbs-transfer energy and a fixed ion density. This symmetry observation cannot be seen in the curves, the difference in the contact angle is significantly larger in the non-wetting case compared to the wetting case. The reason for this behavior is not clear. This means that either the assumption that the ions only alter the fluid-fluid surface tension σ_{AB} in this set-up is wrong, or that there are simulation artifacts which have to be further investigated to be able to apply this model for more complex systems. This problem is also present for smaller and larger droplets as well as for a smaller surface tension σ_{AB} simulated with $G_{\mathrm{AB}} = 5.5\,\frac{\Delta \mathrm{x}^3}{\Delta \mathrm{m}}$ as it can be seen in Figure 5.3. The change in the contact angle difference between the simulations of different surface tension σ_{AB} shows that even without the correct symmetry the contact angle difference is significantly increased in the case of a lower surface tension. This behavior is expected because the change of the surface tension σ^* is the same for both simulations and therefore has a larger influence if the fluid-fluid surface tension σ_{AB} in the system is smaller. From the curves it can also be seen that the expected influence of the Gibbs-transfer energy and the ion density on the contact angle difference increases for both of the parameters.

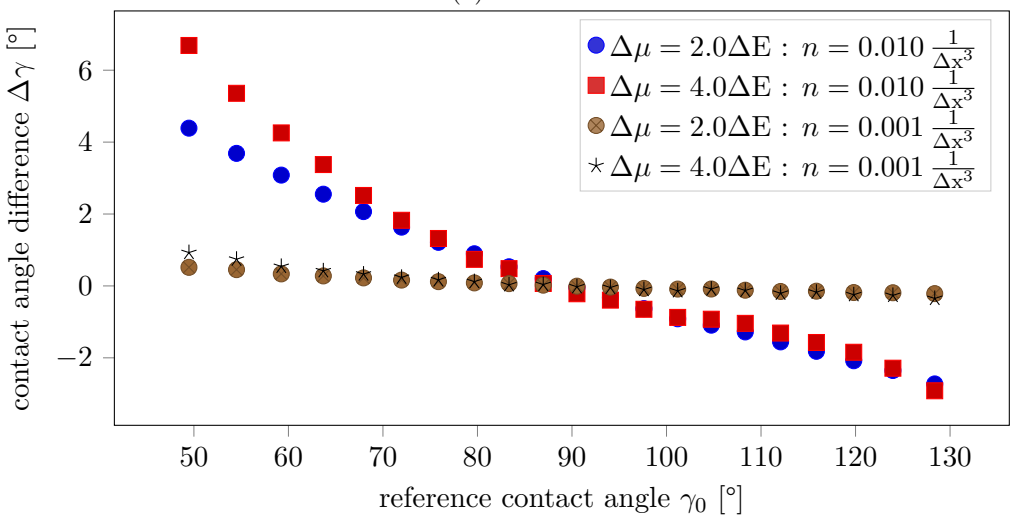
To analytically predict the change for the contact angle the only missing quantity for Equation (5.4) is the change in the surface tension σ^* because the remaining quantities can be extracted from previous simulations (Section 4.1.2 and 4.3). To measure the altered surface tension $\sigma_{\mathrm{AB}}^{\mathrm{new}}$ the same approach as in Section 4.1.2 could be taken where the Young-Laplace equation is used to calculate the surface tension by measuring the radius and the pressure difference for a fluid droplet. The problem with this approach is that equation of state for this system is not known. The reason is that the ions alter the equation of state for the system similar to the changes in Equation (2.3.5) with the introduction of the Shan-Chen forces. The original approach for the derivation of the equation of state in [15] can not be taken due to the difference in the mathematical form of the forces which not only act on the fluid but also on the ions. At the point of writing it is not clear how the derivation of the equation can be done but it is necessary for the comparison because the computed pressure in the simulations is influenced by the presence of the ions. This could then also be used as a reference point for the simulation results to help figure out the reason for the observed mismatch of the expected symmetry.

Table 5.1.: Simulation parameters used in the contact angle simulations.

quantity	symbol	value	unit
domain size		(300, 140)	$\Delta\mathrm{x}$
diffusion coefficient	D	0.0700	$\frac{\Delta\mathrm{x}^2}{\Delta\mathrm{t}}$
energy unit	$k_{\mathrm{B}}T$	1	$\Delta\mathrm{E}$
valency	z	1	
LB fluid density	ϱ	1	$\frac{\Delta\mathrm{m}}{\Delta\mathrm{x}^3}$
LB relaxation time	τ	1	$\Delta\mathrm{t}$
Shan-Chen interaction	G_{AB}	6	$\frac{\Delta\mathrm{x}^3}{\Delta\mathrm{m}}$

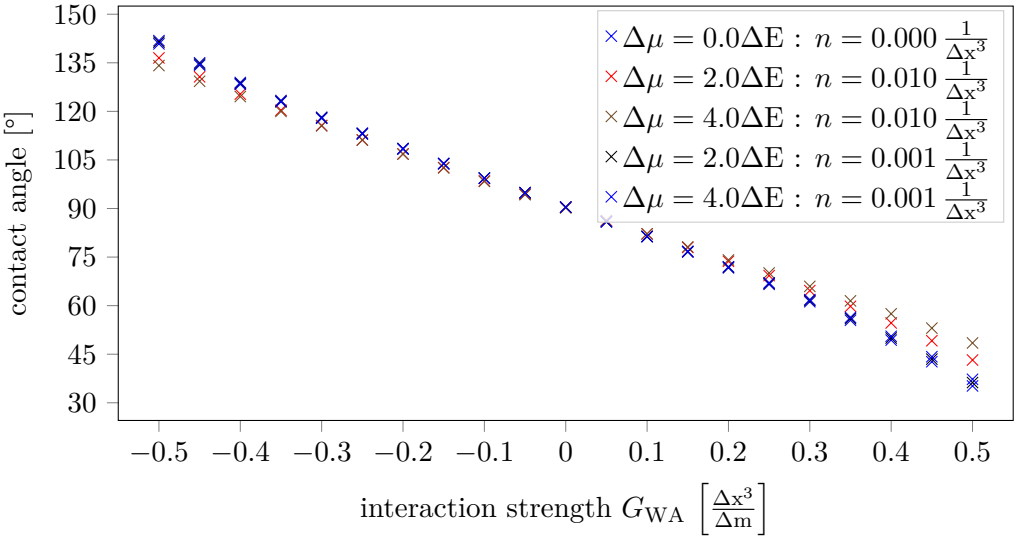


(a)

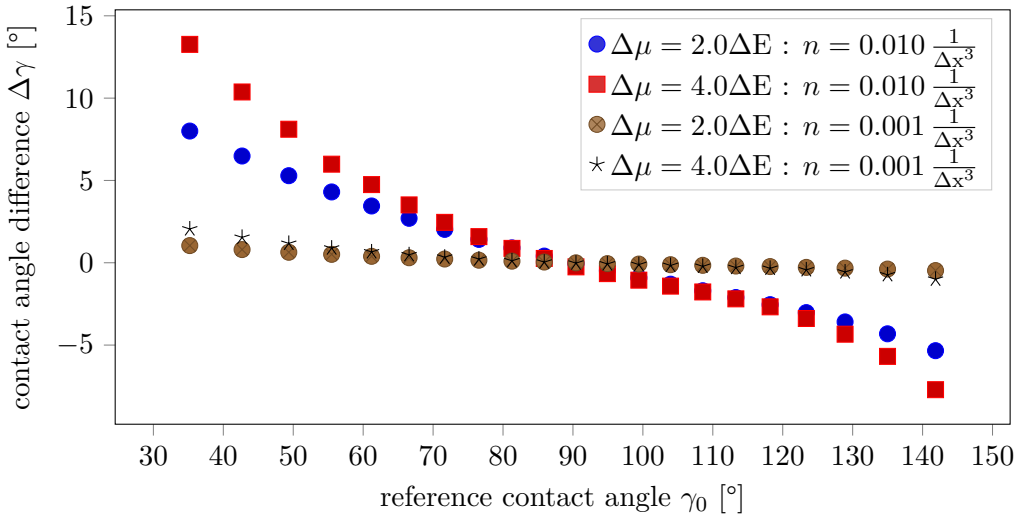


(b)

Figure 5.2.: Measured contact angle (a) for a fluid droplet at different initial contact angles without ions and with ions with different Gibbs-transfer energies and ion densities. The difference of the contact angle to the reference system without ions is shown in (b) where Equation (5.6) was used for the calculation. The simulation parameters are listed in Table 5.1.



(a)



(b)

Figure 5.3.: Measured contact angle (a) for a fluid droplet at different initial contact angles without ions and with ions with different Gibbs-transfer energies and ion densities in the same configuration as in Figure 5.2 with the difference that a lower fluid-fluid surface tension σ_{AB} with $G_{\mathrm{AB}} = 5.5 \frac{\Delta x^3}{\Delta m}$ is used. The difference of the contact angle to the reference system without ions is shown in (b).

6. Summary and Outlook

In this work the Shan-Chen multicomponent extension for the Lattice Boltzmann Method was successfully implemented using the *pystencils* and *lbmpy* code-generation framework. Moreover, we developed several testcases. For this model we use a virtual fluid density inside walls to enforce different wetting states of fluid droplets. This procedure has been shown to enforce different contact angles of a fluid droplet at a wall. In these simulations it was demonstrated that the regular stencil force-calculation for the Shan-Chen interaction at the wall could not be used without modification and the non-physical fluid density boundary condition has to be adjusted. The contact line was estimated using a marching squares/cubes algorithm and the contact angle was measured geometrically by fitting a circle to the contact line.

A lattice electrokinetics solver was also successfully implemented in the *pystencils* framework including a solver for the electrostatic Poisson equation. This implementation was also tested against problems with analytical solutions. The coupling between the two models using the Gibbs-transfer energy has been adapted from the literature to depend on the phase field for multicomponent simulations which removes a non-physical input parameter to the simulation that has to be extracted from separate simulations. These couplings were also tested against an analytical solution and a linear response scaling behavior.

The implemented models were used to investigate the influence of surface effects to the contact angle in oil-water wetting phenomena. A reference simulation was set-up to verify the correct behavior of the simulation. The system was set-up in such a way that the ions impose an additional surface force on the fluid-fluid contact line of the fluid droplet which results in an increase in the surface tension that can be compared to an analytical linear response approximation. The resulting measurements do not feature the analytic predicted symmetry and the reason for this behavior is not presently understood. A different approach to predict the contact angle change which could help to resolve this open issue was discussed, however could not be applied due to the unknown equation of state of the system under investigation.

Outlook

The investigation of the influence of surface effects on the contact angle could not be completed in this work because the reference simulation showed an unexpected and probably unphysical behavior. After resolving this problem, more complex surface effects can be investigated e.g. surface charge densities, spatially varying dielectric permittivity or the modeling of ion-wall interactions that depend on the fluid component present which could model changes in the chemical nature of the surface.

In systems with spatially varying dielectric permittivity, the permittivity is calculated from the phase field which itself is a function of the fluid densities. Therefore, the permittivity is formally a function of the fluid density which gives rise to an additional electrostatic force contribution known as the polarization force. A derivation for this additional term is outlined in Appendix A.2.

A further contribution that could be interesting is the inclusion of chemical reactions which can be done with an additional source term in the lattice electrokinetics equations. This would allow the system to be simulated at different pH values, corresponding to the simulation of H_3O^+ and OH^- ions. This reaction equation can also be spatially dependent therefore allowing for the modeling of catalytic surfaces.

In dynamic systems a spatially varying diffusion coefficient can have an influence on the ion distribution. The systems used in this thesis all assume a uniform diffusion coefficient in the domain that can be different between species but not vary spatially. To include such a spatially varying diffusion coefficient formally an additional source term in the differential equation should appear. The implementation discussed in Section 3 already includes this feature but was not used in this work.

For dynamic systems it is possible that thermal fluctuations can have a significant influence on the fluid flow or the density distribution in the system. Such fluctuations can be added to the models. For the Lattice Boltzmann Method the thermal fluctuations are introduced in the mode-space on all non-physical modes and is already included in the *lbmpy* framework. For the lattice electrokinetics the thermal fluctuations are performed on the fluxes, which is described in [36]. This can give rise to a fully thermalized two-component electrokinetic model that should be useful when considering nano-sized domains.

Bibliography

[1]J. Sheng, "Critical review of low-salinity waterflooding", Journal of Petroleum Science and Engineering **120**, 216–224 (2014).

[2]H. Mahani, S. Berg, D. Ilic, W.-B. Bartels, and V. Joekar-Niasar, "Kinetics of Low-Salinity-Flooding effect", SPE Journal **20**, 008–020 (2015).

[3]M. Bauer et al., "Code generation for massively parallel phase-field simulations", in Proceedings of the international conference for high performance computing, networking, storage and analysis (2019), pp. 1–32.

[4]M. Bauer, H. Köstler, and U. Rüde, "Lbmpy: automatic code generation for efficient parallel lattice boltzmann methods", Journal of Computational Science, 101269 (2020).

[5]A. Katende and F. Sagala, "A critical review of low salinity water flooding: mechanism, laboratory and field application", Journal of Molecular Liquids **278**, 627–649 (2019).

[6]T. Krüger et al., *The lattice Boltzmann method: principles and practice* (Springer, Cham, 2017).

[7]D. Frenkel and B. Smit, *Understanding molecular simulation*, From Algorithms to Applications, 2nd ed. (Academic Press, San Diego, 2002).

[8]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**, 511 (1954).

[9]S. Chapman and T. G. Cowling, *The mathematical theory of non-uniform gases: an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases*, 3. ed., transferred to digital printing, Cambridge Mathematical Library (Cambridge Univ. Press, 1998), 422 pp.

[10]Z. Guo, C. Zheng, and B. Shi, "Discrete lattice effects on the forcing term in the lattice Boltzmann method", Physical Review E **65**, 046308 (2002).

[11]U. D. Schiller, "Thermal fluctuations and boundary conditions in the lattice Boltzmann method", PhD thesis (Johannes Gutenberg-Universität Mainz, 2008).

[12]X. Shan and H. Chen, "Lattice Boltzmann model for simulating flows with multiple phases and components", Physical Review E **47**, 1815 (1993).

[13]X. Shan and H. Chen, "Simulation of nonideal gases and liquid-gas phase transitions by the lattice Boltzmann equation", Physical Review E **49**, 2941 (1994).

[14]L. Chen, Q. Kang, Y. Mu, Y.-L. He, and W.-Q. Tao, "A critical review of the pseudopotential multiphase Lattice Boltzmann model: methods and applications", International Journal of Heat and Mass Transfer **76**, 210–236 (2014).

[15]X. Shan and G. Doolen, "Multicomponent lattice-boltzmann model with interparticle interaction", Journal of Statistical Physics **81**, 379–393 (1995).

[16]P. Yuan and L. Schaefer, "Equations of state in a Lattice Boltzmann model", Physics of Fluids **18**, 042101 (2006).

[17]N. S. Martys and H. Chen, "Simulation of multicomponent fluids in complex three-dimensional geometries by the Lattice Boltzmann method", Physical Review E **53**, 743–750 (1996).

[18]X. Shan, "Analysis and reduction of the spurious current in a class of multiphase Lattice Boltzmann models", Physical Review E **73**, 047701 (2006).

[19]S. Wolfram, "Cellular automaton fluids 1: basic theory", Journal of statistical physics **45**, 471–526 (1986).

[20]A. Fick, "V. on liquid diffusion", The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **10**, 30–39 (1855).

[21]G. Rempfer, "A lattice based model for electrokinetics", Master's thesis (University of Stuttgart, 2013).

[22]G. Rempfer, G. B. Davies, C. Holm, and J. de Graaf, "Reducing spurious flow in simulations of electrokinetic phenomena", The Journal of Chemical Physics **145**, 044901 (2016).

[23]F. Capuani, I. Pagonabarraga, and D. Frenkel, "Discrete solution of the electrokinetic equations", The Journal of Chemical Physics **121**, 973–986 (2004).

[24]D. M. Young, "A historical overview of iterative methods", Computer Physics Communications **53**, 1–17 (1989).

[25]G. Fisicaro, L. Genovese, O. Andreussi, N. Marzari, and S. Goedecker, "A generalized Poisson and Poisson-Boltzmann solver for electrostatic environments", The Journal of Chemical Physics **144**, 014103 (2016).

[26]S. Tyagi et al., "An iterative, fast, linear-scaling method for computing induced charges on arbitrary dielectric boundaries", The Journal of Chemical Physics **132**, 154112 (2010).

[27]N. Rivas, S. Frijters, I. Pagonabarraga, and J. Harting, "Mesoscopic electrohydrodynamic simulations of binary colloidal suspensions", The Journal of Chemical Physics **148**, 144101 (2018).

[28]A. Meurer et al., "Sympy: symbolic computing in python", PeerJ Computer Science **3**, e103 (2017).

[29]M. Bauer et al., "Walberla: a block-structured high-performance framework for multiphysics simulations", Computers & Mathematics with Applications **81**, 478–501 (2021).

[30]C. R. Harris et al., "Array programming with NumPy", Nature **585**, 357–362 (2020).

[31]H. Huang, Z. Li, S. Liu, and X.-y. Lu, "Shan-and-chen-type multiphase lattice Boltzmann study of viscous coupling effects for two-phase flow in porous media", International Journal for Numerical Methods in Fluids **61**, 341–354 (2009).

[32]W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm", ACM SIGGRAPH Computer Graphics **21**, 163–169 (1987).

[33]G. Rempfer, "Lattice-boltzmann simulations in complex geometries", Bachelor thesis (University of Stuttgart, 2010).

[34]M. Bier, J. Zwanikken, and R. van Roij, "Liquid-liquid interfacial tension of electrolyte solutions", Physical Review Letters **101**, 046104 (2008).

[35]T. Young, "III. an essay on the cohesion of fluids", Philosophical Transactions of the Royal Society of London **95**, 65–87 (1805).

[36]R. Kaufmann, "Particle coupling in continuum elektrokinetics simulation", Master Thesis (2018).

[37]L. Landau and E. Lifshitz, *Electrodynamics of continuous media*, Vol. 8 (Pergamon Press, 1960).

A. Derivations

A.1. Derivation of the Two-Phase Poiseuille Flow

The derivation of the two-phase Poiseuille flow solutions in this work are solving the regular Stokes equation with the respective dynamic viscosity in the respective domain with an additional boundary condition for the fluid interfaces.

A.1.1. One Fluid Interface

The system set-up is sketched in Figure 4.8a and includes a single fluid-fluid interface and both fluid components have contact with one of the wall each. Assuming the interface at height h_0 the Stokes equations read

$$\begin{cases} \partial_y^2 v_x^{\mathrm{A}}(y) = -\frac{f}{\mu_{\mathrm{A}}} & \text{for } 0 \leq y \leq h_0 \\ \partial_y^2 v_x^{\mathrm{B}}(y) = -\frac{f}{\mu_{\mathrm{B}}} & \text{for } h_0 \leq y \leq h. \end{cases} \tag{A.1.1}$$

Compared to the one-phase Poiseuille flow the equations have doubled in the amount of unknowns which also forces additional formulations for the boundary conditions. The No-Slip boundary conditions in Equation (A.1.2a) are known from the regular Poisson flow and only have to be transferred to the corresponding velocity. The additional boundary conditions are formulated at the interface between the two fluid components. The first condition is that the fluid velocity has to be continuous at the fluid interface which is formulated in Equation (A.1.2b). The second additional condition is formulated for the shear stress at the fluid interface which is also forced to be continuous and is formulated in Equation (A.1.2c).

$$v_x^{\mathrm{A}}(0) = 0 = v_x^{\mathrm{B}}(h) \tag{A.1.2a}$$

$$v_x^{\mathrm{A}}(h_0) = v_x^{\mathrm{B}}(h_0) \tag{A.1.2b}$$

$$-\mu_{\mathrm{A}} \partial_y v_x^{\mathrm{A}}(y)\big|_{y=h_0} = -\mu_{\mathrm{B}} \partial_y v_x^{\mathrm{B}}(y)\big|_{y=h_0} \tag{A.1.2c}$$

With the additional boundary conditions the unknown velocity profile is fully determined. For solving the two partial differential equations (A.1.1) the ansatz

$$\begin{aligned} v_x^{\mathrm{A}}(y) &= a_{\mathrm{A}} y^2 + b_{\mathrm{A}} y + c_{\mathrm{A}} \\ v_x^{\mathrm{B}}(y) &= a_{\mathrm{B}} y^2 + b_{\mathrm{B}} y + c_{\mathrm{B}} \end{aligned} \tag{A.1.3}$$

is used which can also be found by integrating Equations (A.1.1) twice with respect to the positions y. This also eliminates the first two unknowns a_{A} and a_{B}.

$$a_{\mathrm{A}} = -\frac{f}{2\mu_{\mathrm{A}}} \qquad\qquad a_{\mathrm{B}} = -\frac{f}{2\mu_{\mathrm{B}}}. \tag{A.1.4}$$

With the two No-Slip boundary conditions in Equation (A.1.2a) one can eliminate two more unknowns from the equations

$$c_{\mathrm{A}} = 0 \qquad\qquad c_{\mathrm{B}} = -a_{\mathrm{B}}h^2 - b_{\mathrm{B}}h. \tag{A.1.5}$$

The stress boundary condition in Equation (A.1.2c) is used to fix the relation between the two unknowns b_{A} and b_{B} where the results in (A.1.4) have been used

$$\begin{aligned} \mu_{\mathrm{A}}\left(2a_{\mathrm{A}}h_0 + b_{\mathrm{A}}\right) &= \mu_{\mathrm{B}}\left(2a_{\mathrm{A}}h_0 + b_{\mathrm{B}}\right) \\ \mu_{\mathrm{A}}b_{\mathrm{A}} &= \mu_{\mathrm{B}}b_{\mathrm{B}}. \end{aligned} \tag{A.1.6}$$

The last remaining boundary condition (A.1.2b) therefore reads

$$a_{\mathrm{A}}h_0^2 + b_{\mathrm{A}}h_0 = a_{\mathrm{B}}\left(h_0^2 - h^2\right) + b_{\mathrm{B}}\left(h_0 - h\right). \tag{A.1.7}$$

Using the relation in Equation (A.1.6) one can separate the remaining unknown b_{B}

$$b_{\mathrm{B}} = \frac{a_{\mathrm{A}}h_0^2 - a_{\mathrm{B}}\left(h_0^2 - h^2\right)}{h_0\left(1 - \frac{\mu_{\mathrm{B}}}{\mu_{\mathrm{A}}}\right) - h}. \tag{A.1.8}$$

The complete solution therefore reads

$$\begin{aligned} v_x^{\mathrm{A}}(y) &= -\frac{f}{2\mu_{\mathrm{A}}}\left[y^2 - \frac{\mu_{\mathrm{A}}\left(h_0^2 - h^2\right) - \mu_{\mathrm{B}}h_0^2}{\mu_{\mathrm{A}}\left(h_0 - h\right) - \mu_{\mathrm{B}}h_0}y\right] \\ v_x^{\mathrm{B}}(y) &= -\frac{f}{2\mu_{\mathrm{B}}}\left[\left(y^2 - h^2\right) - \frac{\mu_{\mathrm{A}}\left(h_0^2 - h^2\right) - \mu_{\mathrm{B}}h_0^2}{\mu_{\mathrm{A}}\left(h_0 - h\right) - \mu_{\mathrm{B}}h_0}\left(y - h\right)\right]. \end{aligned} \tag{A.1.9}$$

A.1.2. Two Fluid Interfaces

The system set-up is sketched in Figure 4.8b and this time includes two fluid-fluid interfaces. This means that one fluid component is fully separating the other fluid component in two slices and does not have contact with the wall. Assuming the two fluid-fluid interfaces are at height h_1 and h_2 the equations read

$$\begin{cases} \partial_y^2 v_x^{\mathrm{A}}(y) = -\frac{f}{\mu_{\mathrm{A}}} & \text{for } x \in [0, h_1] \cup [h_2, h] \\ \partial_y^2 v_x^{\mathrm{B}}(y) = -\frac{f}{\mu_{\mathrm{B}}} & \text{for } x \in [h_1, h_2]\,. \end{cases} \tag{A.1.10}$$

Due to the initial set-up of the problem two fluid interfaces are present in the system. This allows the formulation of the boundary conditions on the interfaces by just using

the continuity of the velocity. The reason is that now two fluid interfaces are present and only one of them is enough to fully determine the free parameters from the Poiseuille flow as sees in the previous calculation. The boundary conditions read

$$v_x^{\mathrm{A}}(0) = 0 = v_x^{\mathrm{A}}(h) \tag{A.1.11a}$$
$$v_x^{\mathrm{A}}(h_1) = v_x^{\mathrm{B}}(h_1) \tag{A.1.11b}$$
$$v_x^{\mathrm{A}}(h_2) = v_x^{\mathrm{B}}(h_2). \tag{A.1.11c}$$

To solve the equations the same approach as in Equation (A.1.3) is taken which uses the ansatz

$$\begin{aligned} v_x^{\mathrm{A}}(y) &= a_{\mathrm{A}} y^2 + b_{\mathrm{A}} y + c_{\mathrm{A}} \\ v_x^{\mathrm{B}}(y) &= a_{\mathrm{B}} y^2 + b_{\mathrm{B}} y + c_{\mathrm{B}} \end{aligned}. \tag{A.1.12}$$

The quadratic factors can both be directly determined from the differential equations (A.1.10)

$$a_{\mathrm{A}} = -\frac{f}{2\mu_{\mathrm{A}}} \qquad\qquad a_{\mathrm{B}} = -\frac{f}{2\mu_{\mathrm{B}}}. \tag{A.1.13}$$

In this case the two No-Slip boundary conditions in Equation (A.1.11a) fully determine the solution for fluid component A

$$c_{\mathrm{A}} = 0 \qquad\qquad b_{\mathrm{A}} = -a_{\mathrm{A}} h \tag{A.1.14}$$

The two leftover unknown can both be calculated by solving the system of equations given from the two velocity boundary conditions (A.1.11b) and (A.1.11c). One way is to subtract one equation from the other to fix b_{B} and plug the resulting solution into one of the equations to calculate c_{B}. The resulting relations for the parameters read

$$b_{\mathrm{B}} = (a_{\mathrm{A}} - a_{\mathrm{B}})(h_1 + h_2) + b_{\mathrm{A}} \tag{A.1.15}$$
$$c_{\mathrm{B}} = (a_{\mathrm{A}} - a_{\mathrm{B}}) h_1^2 + (b_{\mathrm{A}} - b_{\mathrm{B}}) h_1. \tag{A.1.16}$$

The complete solution therefore reads

$$\begin{aligned} v_x^{\mathrm{A}}(y) &= -\frac{f}{2\mu_{\mathrm{A}}} y\,(y - h) \\ v_x^{\mathrm{B}}(y) &= -\frac{f}{2\mu_{\mathrm{B}}} \left[y^2 - y \left(h_1 + h_2 + \frac{\mu_{\mathrm{B}}}{\mu_{\mathrm{A}}} (h - h_1 - h_2) \right) + h_1 h_2 \left(1 - \frac{\mu_{\mathrm{B}}}{\mu_{\mathrm{A}}} \right) \right]. \end{aligned} \tag{A.1.17}$$

A.2. Derivation of the Polarization Force

The goal is to find an expression for the body force f_i of a dielectric fluid element which is exposed to a electric field E_i. The derivation is based on [37, pp. 64–69]. For this an expression for the stress tensor σ_{ij} has to be derived which can be transformed to the body force with

$$f_i = \frac{\partial \sigma_{ij}}{\partial x_i}. \tag{A.2.1}$$

Considering a volume element $\mathrm{d}V$ of a layer of dielectric fluid of thickness h which has without the loss of generality parallel side planes. Furthermore this volume element is supposed to be of uniform and isotropic temperature T and density ϱ. The electric field in this volume element can be thought of the one created by two conducing plates with appropriate surface charge.

Starting the calculation of the force on the volume element by displacing the upper plate by an infinitesimal virtual vector ξ_i. Since this change is infinitesimal small the potential at each point in this volume element does not change and the deformation is supposed to be homogeneous and isothermal. The resulting force on the surface of the volume element is given by $-\sigma_{ij} n_i$, the work done by this virtual displacement can be written as $-\sigma_{ij} n_i \xi_j$. For this displacement the variation of the density and the variation of the electric field are given by

$$\begin{aligned} \delta(\varrho) &= -\varrho \frac{\delta(h)}{h} = -\varrho \frac{n_i \xi_i}{h} \\ \delta(E_i) &= -E_i \frac{\delta(h)}{h} = -E_i \frac{n_j \xi_j}{h}, \end{aligned} \tag{A.2.2}$$

where both quantities are just stretched to the new volume and the variation of the height is given by $\delta(h) = n_i \xi_i$. The work done in this displacement is equivalent to the decrease in $\int \widetilde{F} \mathrm{d}V$ where $\widetilde{F}$ is the thermodynamic potential per unit volume which is defined for isotropic dielectric media as

$$\widetilde{F}(T, \varrho, E_i) = F_0(T, \varrho) - \frac{\varepsilon(\varrho) E^2}{8\pi}, \tag{A.2.3}$$

where $F_0(T, \varrho)$ is the *Helmholtz-free-energy* per unit volume without an electric field and $\varepsilon(\varrho)$ is the permittivity. This change in the thermodynamic potential can also be expressed as the change per unit surface $h\widetilde{F}$

$$\sigma_{ij} n_i \xi_j = \delta(h\widetilde{F}) = h\delta(\widetilde{F}) + \widetilde{F}\delta(h). \tag{A.2.4}$$

The isothermal ($\delta(T) = 0$) variation of the thermodynamic potential $\widetilde{F}$ is given by

$$\delta(\widetilde{F}) = \left(\frac{\partial \widetilde{F}}{\partial \varrho}\right)_{T,E} \delta(\varrho) + \left(\frac{\partial \widetilde{F}}{\partial E_i}\right)_{T,\varrho} \delta(E_i) = \left(\frac{\partial \widetilde{F}}{\partial \varrho}\right)_{T,E} \delta(\varrho) + \frac{\varepsilon(\varrho) E_i}{4\pi} \delta(E_i) \tag{A.2.5}$$

which can be substituted in Equation (A.2.4) together with the expressions for the variations in (A.2.2) results in

$$\sigma_{ij} n_i \xi_j = \left[\frac{E_i \varepsilon(\varrho) E_j}{4\pi} - \varrho \left(\frac{\partial \widetilde{F}}{\partial \varrho} \right)_{T,E} \delta_{ij} + \widetilde{F} \delta_{ij} \right] n_i \xi_j. \tag{A.2.6}$$

To further simplify this expression for the stress tensor one can make use of the thermodynamic relation for the pressure

$$\left(\frac{\partial F_0}{\partial V} \right)_{T,\varrho} = -p_0(T, \varrho) \tag{A.2.7}$$

which can be rewritten as

$$\left[\frac{\partial}{\partial \left(\frac{1}{\varrho} \right)} \left(\frac{F_0}{\varrho} \right) \right]_T = F_0 - \varrho \left(\frac{\partial F_0}{\partial \varrho} \right)_T = -p_0(T, \varrho). \tag{A.2.8}$$

Considering the central term in the expression for the stress tensor (A.2.6) and plugging (A.2.8) together with the definition for the thermodynamic potential (A.2.3) results in

$$\begin{aligned} -\varrho \left(\frac{\partial \widetilde{F}}{\partial \varrho} \right)_{T,E} &= -\varrho \left(\frac{\partial F_0}{\partial \varrho} \right)_T + \frac{E^2}{8\pi} \varrho \left(\frac{\partial \varepsilon(\varrho)}{\partial \varrho} \right)_{T,E} \\ &= -p_0(T, \varrho) - F_0 + \frac{E^2}{8\pi} \varrho \left(\frac{\partial \varepsilon(\varrho)}{\partial \varrho} \right)_{T,E} \\ &= -p_0(T, \varrho) - \widetilde{F} - \frac{\epsilon E^2}{8\pi} \left[\varepsilon(\varrho) - \varrho \left(\frac{\partial \varepsilon(\varrho)}{\partial \varrho} \right)_{T,E} \right]. \end{aligned} \tag{A.2.9}$$

This expression can be substituted in (A.2.6) results in the final expression for the stress tensor

$$\sigma_{ij} = -p_0(T, \varrho) \delta_{ij} - \frac{E^2}{8\pi} \left[\varepsilon(\varrho) - \varrho \left(\frac{\partial \varepsilon(\varrho)}{\partial \varrho} \right)_{T,E} \right] \delta_{ij} + \frac{\varepsilon(\varrho) E_i E_j}{4\pi}. \tag{A.2.10}$$

The body force f_i can now be calculated with Equation (A.2.1)

$$f_i = -\frac{\partial}{\partial x_i} p_0(T, \varrho) + \frac{1}{8\pi} \frac{\partial}{\partial x_i} \left[E^2 \varrho \left(\frac{\partial \varepsilon(\varrho)}{\partial \varrho} \right)_{T,E} \right] - \frac{E^2}{8\pi} \frac{\partial \varepsilon(\varrho)}{\partial x_i} + \frac{1}{4\pi} \left[\frac{\partial}{\partial x_j} \left(\varepsilon(\varrho) E_i E_j \right) - \frac{\epsilon}{2} \frac{\partial E^2}{\partial x_i} \right]. \tag{A.2.11}$$

The last expression of this equation can be shown to vanish because of $\vec{\nabla} \cdot \varepsilon(\varrho) \vec{E} = 0$ and $\vec{\nabla} \times \vec{E} = 0$

$$\frac{\partial}{\partial x_j}\left(\varepsilon(\varrho)E_iE_j\right) - \frac{\epsilon}{2}\frac{\partial E^2}{\partial x_i} = E_i \underbrace{\frac{\partial}{\partial x_j}\left(\varepsilon(\varrho)E_j\right)}_{\vec{\nabla}\cdot\epsilon\vec{E}=0} + \varepsilon(\varrho)E_j\frac{\partial E_i}{\partial x_j} - \varepsilon(\varrho)E_j\frac{\partial E_j}{\partial x_i}$$
$$= -\varepsilon(\varrho)E_j \underbrace{\left[\frac{\partial E_j}{\partial x_i} - \frac{\partial E_i}{\partial x_j}\right]}_{\vec{\nabla}\times\vec{E}=0} = 0 \tag{A.2.12}$$

which results in the final expression for the polarization force

$$\vec{f} = -\vec{\nabla}p_0(T,\varrho) + \frac{1}{8\pi}\vec{\nabla}\left[E^2\varrho\left(\frac{\partial\varepsilon(\varrho)}{\partial\varrho}\right)_{T,E}\right] - \frac{E^2}{8\pi}\vec{\nabla}\varepsilon(\varrho). \tag{A.2.13}$$

This expression for the force does not consider the presence of charges which can be simply added

$$\vec{f} = -\vec{\nabla}p_0(T,\varrho) + \frac{1}{8\pi}\vec{\nabla}\left[E^2\varrho\left(\frac{\partial\varepsilon(\varrho)}{\partial\varrho}\right)_{T,E}\right] - \frac{E^2}{8\pi}\vec{\nabla}\varepsilon(\varrho) + \varrho_{\text{charge}}\vec{E}. \tag{A.2.14}$$

In the context of two component simulations one can use a linear model for the dielectric permittivity

$$\varepsilon(\varrho) = a\varrho + b \tag{A.2.15}$$

where $a, b \in \mathbb{R}$ are constants. Evaluating the partial derivative of the permittivity in expression (A.2.14) results in

$$\varrho\left(\frac{\partial\varepsilon(\varrho)}{\partial\varrho}\right)_{T,E} = \varrho a = \varepsilon(\varrho) - b \tag{A.2.16}$$

which can be substituted and after applying the product rule to the expression including the substitution simplifies the expression for the body force to

$$\vec{f} = -\vec{\nabla}p_0(T,\varrho) + \frac{\varepsilon(\varrho)-b}{8\pi}\vec{\nabla}E^2 + \varrho_{\text{charge}}\vec{E}. \tag{A.2.17}$$