Computers & Fluids 93 (2014) 1–17

Contents lists available at ScienceDirect

Computers & Fluids

journal homepage: www.elsevier.com/locate/compfluid

Efficient GPGPU implementation of a lattice Boltzmann model for multiphase flows with high density ratios

Amir Banari [a], Christian Janßen [a,b], Stephan T. Grilli [a,*], Manfred Krafczyk [c]

[a] Department of Ocean Engineering, University of Rhode Island, USA
[b] Inst. M-8, Fluid Dynamics and Ship Theory, Hamburg University of Technology, Germany
[c] Institute for Computational Modeling in Civil Engineering, Technische Universitaet Braunschweig, Germany

ARTICLE INFO

Article history:
Received 19 May 2013
Received in revised form 2 January 2014
Accepted 8 January 2014
Available online 18 January 2014

Keywords:
Lattice Boltzmann Method
Multiphase flows
High density ratio
GPGPU parallel implementation

ABSTRACT

We present the development of a Lattice Boltzmann Method (LBM) for the numerical simulation of multiphase flows with high density ratios, such as found in ocean surface wave and air–sea interaction problems, and its efficient implementation on a massively parallel General Purpose Graphical Processing Unit (GPGPU). The LBM extends Inamuro's et al.'s (2004) multiphase method by solving the Cahn–Hilliard equation on the basis of a rigorously derived diffusive interface model. Similar to Inamuro et al., instabilities resulting from high density ratios are eliminated by solving an additional Poisson equation for the fluid pressure. We first show that LBM results obtained on a GPGPU agree well with standard analytic benchmark problems for: (i) a two-fluid laminar Poiseuille flow between infinite plates, where numerical errors exhibit the expected convergence as a function of the spatial discretization; and (ii) a stationary droplet case, which validates the accuracy of the surface tension force treatment as well as its convergence with increasing grid resolution. Then, simulations of a rising bubble simultaneously validate the modeling of viscosity (including drag forces) and surface tension effects at the fluid interface, for an unsteady flow case. Finally, the numerical validation of more complex flows, such as Rayleigh–Taylor instability and wave breaking, is investigated. In all cases, numerical results agree well with reference data, indicating that the newly developed model can be used as an accurate tool for investigating the complex physics of multiphase flows with high density ratios. Importantly, the GPGPU implementation proves highly efficient for this type of models, yielding large speed-ups of computational time. Although only two-dimensional cases are presented here, for which computational effort is low, the LBM model can (and will) be implemented in three-dimensions in future work, which makes it very important using an efficient solution.

1. Introduction

The numerical simulation of multiphase and multi-component fluid flows is a challenging problem in Computational Fluid Dynamics (CFD), both for conventional macroscopic and mesoscopic methods, such as the Lattice-Boltzmann Method (LBM). In classical CFD methods, multiphase flows are simulated by coupling a Navier–Stokes (NS) equation solver to an advection or advection–diffusion equation scheme, for the updating of interfaces between fluids [1]. In earlier work, advection equations have been used in combination with either sharp or diffuse interface models (although this may seem less adequate in the latter case), whereas advection–diffusion equations have mostly been used with diffusive interface models. The interface itself is typically represented by a capturing method such as the widely used Volume Of Fluid (VOF) method [2], or an interface tracking method. Most of the interface tracking methods assume a sharp interface, i.e., they consider the phase transition to be clearly defined and thus the interface between two fluids to be infinitely thin. By contrast, the interface capturing methods allow for both sharp or diffusive interface representations, depending on the equation solved. An additional challenge when using a sharp interface method is the accurate computation of the interface curvature and related surface tension forces. This has encouraged many researchers to use diffusive interface methods, in which surface tension forces at interfaces are modeled as a continuum, by distributing them over thin but numerically resolved layers [3]. Such models have recently attracted much interest, owing to their computational advantages [4,5]. Because of these various options, when developing and implementing a free surface or multiphase CFD model, one has to make a priori decisions regarding using: (i) a sharp or diffusive interface method; (ii) an advection or advection–diffusion equation for free surface updating; and (iii) a tracking or a capturing

* Corresponding author.
E-mail address: grilli@oce.uri.edu (S.T. Grilli).

0045-7930/$ - see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compfluid.2014.01.004

method for the interface representation. In our proposed two-phase model, detailed below, interface motion is modeled by a Cahn–Hilliard's (CH) interface capturing, advection–diffusion equation [6], using a scalar order parameter ϕ_α ($\alpha = 1, 2$) to identify each phase. The interface between the two phases is then defined as a smooth transition from ϕ_1 to ϕ_2 and vice versa.

Recently, the LBM has matured into a powerful alternative to classical NS solvers, both for simulating single phase, and multiphase and multi-component flows [7–10]. The LBM discretizes the Boltzmann equation, which governs the dynamics of molecular probability distribution functions from a microscopic scale point of view, based on a discrete velocity set. This yields a numerical method for computing macroscopic distribution functions on a Cartesian grid (the lattice). Macroscopic hydrodynamic quantities, such as pressure and velocity, are obtained as low-order moments of these distribution functions. The resulting formulation can be shown to converge towards the solution of classical macroscopic equations such as NS, to the second-order in space and-first order in time [11]. The LBM has several solver-specific features, which allow taking full advantage of recent advances in massively parallel General Purpose Graphical Processing Units (GPGPU) [12], such as a an operator locality and fairly regular algorithms, which yield significantly more efficient parallel codes than those of more traditional CFD solvers.

While there have been numerous applications of classical CFD solvers to multiphase flows, whose exhaustive review is beyond the scope of this paper, over the past two decades, several noteworthy methods have been developed for simulating multiphase flows in the context of the LBM. These are: Rothman and Keller's color method [13], the Shan and Chen model (SC) [14], Swift's free energy method [7], and the method of He et al. [8]. In the SC method, separate probability distribution functions are introduced for each phase, and these are modified by a forcing term that models "molecular" interactions with neighboring lattice nodes in the other phase. Swift et al. used a free energy concept, in which the stress tensor is modified by adding the effects of surface tension forces. In their method, two sets of particle distribution functions are required, one for solving NS equations and one for solving the approximate CH interface capturing equation. He et al. transformed the classical discrete Boltzmann equation for a single phase, from a mass and momentum to a pressure and momentum formulation. This transformation helps reducing potential instabilities due to high gradients in fluid density near the interface. Similar to Swift et al., a second set of particle distribution functions is used to track the interface.

While all of the above methods successfully solved multiphase flows, the maximum fluid density ratio achievable in computations was limited by the occurrence of instabilities for high density ratios (typically larger than 10–20). Developing methods to overcome this limitation is challenging and represents an active research area in LBM. In this work, we aimed at developing an accurate and efficient LBM method for investigating the complex physics of ocean wave and air–sea interaction processes. Hence, our method must deal with large density ratios of about 1000. Our proposed approach builds and improves on some recent progress achieved in the LBM modeling of multiphase flows. In particular:

- Lee and Lin [9] used an approach similar to that of He et al. [8] to solve discrete Boltzmann equations for the pressure and momentum in multiphase flows. In those, they split up intermolecular forces for non-ideal gas into hydrodynamic pressure, thermodynamic pressure, and surface tension force contributions. They reported that "parasitic currents" affected numerical results at interfaces, due to the imbalance between thermodynamic pressure and surface tension forces resulting from truncation errors, particularly, in relation to curvature computations. They nearly eliminated this problem by using a thermodynamic identity to recast the intermolecular forcing term from a stress to a potential form. Furthermore, they used different discretization patterns (i.e., centered, staggered, and mixed differences) at different steps of the simulation, to make their numerical scheme stable for large density ratios. With this discretization scheme, they were able to simulate two phase flows with density ratio of up to 1000. However, their method was only valid for low Reynolds and Mach numbers. Additionally, numerical efficiency seemed to be quite low, due to the need for calculating various forms of first- and second-order derivatives of the macroscopic variables.
- Inamuro et al.'s [10] LBM method overcame numerical instabilities resulting from high density ratios by removing density from the advection part of the equilibrium distribution functions, resulting in the absence of a pressure gradient in the momentum equation (referred to in the following as "pressureless" NS equations). They then corrected the velocity field by solving a Poisson equation for the pressure. Unlike in classical LBMs, in their model, the fluid viscosity is not related to the relaxation time, because of the absence of pressure and density in the equilibrium distribution functions. Therefore, viscous effects are modeled by adding: (i) an extra term to the equilibrium distribution functions, which removes the dependency of relaxation time on viscosity; and (ii) the viscous stress tensor as a body force to the collision operator. However, specifying viscous effects this way in the model yields additional nonphysical terms in the corresponding momentum equation, which decrease the model accuracy.

In this work, we developed and implemented a LBM based in part on Inamuro et al.'s [10,15] approach of removing the pressure gradient from the momentum equation. However, in our model we use a modified primary set of equilibrium functions for the "pressureless" NS equations, in which viscosity is still present and related to the relaxation time as in a classical LBM. Thus in our method, the corresponding NS equations do not have the undesired terms that appear in Inamuro et al.'s model. Additionally, Inamuro et al. used a convection–diffusion equation in their interface capturing method, whose theoretical derivation did not seem to be fully rigorous. By contrast, to this effect, we use the standard Cahn–Hilliard (CH) equation, in which surface tension and equivalent body forces are rigorously derived, and we solve it using an LBM scheme, by way of a second set of equilibrium functions; this yields a more accurate and efficient solution than in earlier implementations, particularly on a GPGPU. Finally, we similarly correct the "pressureless" velocity field by solving a Poisson equation for the pressure, but here this is done by way of a third set of LBM equilibrium functions, again providing an efficient scheme when implemented on a GPGPU.

More specifically, it has been demonstrated in various publications [16,12,17,2,18] that LBM methods can be made very efficient when implemented on massively parallel GPGPUs (single and multiple units). Hence, as indicated above, our proposed model's algorithm, which uses 3 separate sets of LB equilibrium functions and related collision-propagation operators is optimally formulated for such an implementation. Accordingly, we developed our LBM code in the nVIDIA CUDA framework, which made it possible efficiently implementing and validating it on the latest generation GPGPUs (e.g., nVIDIA Tesla C2070, which provide up to 448 cores, 6 GB of main memory, and a double precision computing capability). For all applications presented herein, this GPGPU implementation led to computational speedups of about two orders of magnitude, as compared to a single-core CPU implementation of the same model. However, because the Poisson equation must be (iteratively) solved over the entire computational domain for each

time step of the solution, these are still very demanding computations and only two-dimensional (2D) problems have been solved so far, on a single GPGPU, although the method could be quite easily extended to three dimensions and to multiple GPGPUs.

The paper is organized as follows. Section 2 provides an introduction to the free energy method, applied to diffusive interfaces modeled using the CH equation. Our proposed LBM for multiphase flows with high density ratio is detailed in Section 3, where we separately describe the LB solution of the momentum, CH advection–diffusion, and pressure Poisson equations. The GPGPU implementation is briefly described in Section 4. In Section 5, the method is validated by comparing 2D numerical results to reference solutions for two-component Poiseuille flows, stationary droplets of one fluid embedded in another, a bubble of a lighter fluid rising in another fluid, the Rayleigh Taylor instability, and breaking ocean surface waves. Finally, Section 6 offers conclusions and perspectives for future work.

2. Diffusive interface models

As mentioned in the introduction, numerical schemes based on a sharp interface representation, while usually more accurate, may require addressing additional numerical problems in their implementation, as compared to diffusive interface models. In particular, although not strictly necessary, sharp interface models may use a moving numerical grid, whereas diffusive interface models naturally accommodate fixed grids (such as used in the LBM). Sharp interface models also face difficulties for accurately computing the interface curvature and the related surface tension forces. This often leads to the appearance of "parasitic currents" in the numerical solution along the interface. These problems disappear when using a diffusive-interface representation based on the continuous variation of an order parameter (such as density or a function of density), in a way that is physically consistent with microscopic theories of interfacial processes. Three main types of diffusive-interface models have been proposed in the literature: (i) tracking force models [4]; (ii) continuum surface force models [5]; and (iii) phase-field models [3].

In the current work, we use the latter approach, in which the total free energy $\mathcal{F}$ of a two-fluid system is specified to be minimum for the equilibrium interface profile $\phi(\zeta)$, where ϕ denotes a continuously varying order parameter (with values ϕ_1 and ϕ_2 referring to fluid 1 and 2 on either side of the interface, respectively; and $\phi_1 > \phi_2, \phi \in [\phi_1, \phi_2]$), and ζ is a coordinate normal to the interface (positive when pointing from fluid 1 to 2). More specifically, Cahn and Hilliard [6] expressed the free energy density of an isothermal two-phase/fluid system as,

$$f = \frac{k}{2}|\nabla\phi|^2 + \beta\Psi(\phi). \tag{2.1}$$

The first term in this equation is related to the energy gradient and the second one to the bulk free-energy density $\Psi(\phi)$. In the following, we will express the two parameters β and k in Eq. (2.1) as a function of the standard surface tension coefficient σ_{12} of the two fluid system and an assumed interface thickness W.

The existence of two phases is possible if Ψ has two minima, such as when posing, $\Psi(\phi) = (\phi-\phi_2)^2(\phi-\phi_1)^2$. Based on Eq. (2.1), the total free-energy of the two-phase system in domain Ω reads,

$$\mathcal{F} = \int_\Omega f(\phi, \nabla\phi)\, d\Omega = \int_\Omega \left\{\frac{k}{2}|\nabla\phi|^2 + \beta\Psi(\phi)\right\} d\Omega. \tag{2.2}$$

The chemical potential μ_ϕ is then defined as the functional derivative of the free energy with respect to ϕ, which can be easily derived from the Euler–Lagrange equation as,

$$\mu_\phi = \frac{\delta\mathcal{F}}{\delta\phi} = \frac{\partial f}{\partial\phi} - \nabla\left(\frac{\partial f}{\partial(\nabla\phi)}\right) = \beta\Psi' - k\nabla^2\phi, \tag{2.3}$$

As indicated, the equilibrium state of the interface is defined such that the variation of ϕ across the interface minimizes $\mathcal{F}$, hence, this corresponds to $\mu_\phi = 0$ or,

$$k\frac{d^2\phi}{d^2\zeta} = \beta\frac{d\Psi}{d\phi}, \tag{2.4}$$

since ϕ is only a function of ζ. Multiplying both sides of this equation by $d\phi/d\zeta$ and integrating with respect to ζ yields,

$$k\left(\frac{d\phi}{d\zeta}\right)^2 = 2\beta\Psi \tag{2.5}$$

which, combined with the above definition of $\Psi(\phi)$, can be solved for ϕ as,

$$\phi(\zeta) = \frac{\phi_1+\phi_2}{2} + \frac{\phi_1-\phi_2}{2}\tanh\left(\frac{2\zeta}{W}\right), \tag{2.6}$$

where the equivalent interface thickness has been defined as,

$$W = \frac{4}{\phi_1-\phi_2}\sqrt{\frac{k}{2\beta}}. \tag{2.7}$$

This equation predicts that, at a distance $W/2$ on either side of the interface, the order parameter reaches 76% of its value in each fluid, ϕ_1 or ϕ_2, respectively.

Jacqmin [3] further expressed surface tension forces as a function of the variation of the order parameter across the interface (such as in Eq. (2.6)), by assuming that the rate of change of $\mathcal{F}$ due to convection is equal and opposite to the rate of change of the kinetic energy due to surface tension forces $\boldsymbol{F}$, i.e.,

$$\int_\Omega \boldsymbol{F}\cdot\boldsymbol{u}\, d\Omega = \int_\Omega \mu_\phi\nabla\cdot(\phi\boldsymbol{u})\, d\Omega = -\int_\Omega(\phi\nabla\mu_\phi)\cdot\boldsymbol{u}\, d\Omega, \tag{2.8}$$

where $\boldsymbol{u}$ denotes the interface velocity. Using Eq. (2.3) and noting that surface tension forces only exist within the plane tangent to the interface, yields,

$$\boldsymbol{F} = -\phi\nabla\mu_\phi = k\phi\nabla(\nabla^2\phi). \tag{2.9}$$

Eq. (2.9) represents surface tension by an equivalent volume force, which could be directly inserted into the governing momentum (NS) equations (see, e.g., [3]).

Introducing a stress representation of surface tension forces into Eq. (2.9), Jacqmin showed that, for a plane interface, the equivalent surface tension coefficient for the two-fluid system can be calculated as,

$$\sigma_{12} = k\int_{-\infty}^{+\infty}\left(\frac{d\phi}{d\zeta}\right)^2 d\zeta. \tag{2.10}$$

Combining Eqs. (2.10) and (2.6), we find,

$$\sigma_{12} = \frac{(\phi_1-\phi_2)^3}{6}\sqrt{2k\beta}. \tag{2.11}$$

Hence, in a given application, once the interface thickness W and the surface tension coefficient σ_{12} are specified, the two governing parameters of the diffusive interface model, β and k, can be calculated with Eqs. (2.7) and (2.11).

Finally, the motion of the diffusive interface is modeled, following Jacqmin [3], as a function of the order parameter by extending the Cahn–Hilliard (CH) equation [6] to include convection, as,

$$\frac{\partial\phi}{\partial t} + \nabla\cdot(\phi\boldsymbol{u}) = M\nabla^2\mu_\phi, \tag{2.12}$$

where the right hand side represents the interface diffusive transport, expressed as a function of a mobility coefficient M and the chemical potential μ_ϕ, defined above.

3. Lattice Boltzmann model

In this work, two-dimensional (2D) multiphase flows are simulated by solving two sets of equations: (i) the NS equations, which provide the flow fields, based on the conservation of mass and momentum, with the addition of the volumetric surface tension force term of Eq. (2.9); (ii) the extended Cahn–Hilliard equation (2.12), which describes the interface motion. We solve these equations using a new Lattice Boltzmann Method (LBM), which is an extension of Inamuro et al.'s [10] method, developed to accurately simulate multiphase flows with large density ratios (such as air–water). As will be detailed in the following, our algorithms significantly differ from Inamuro et al.'s in several aspects.

To develop our LBM equations, we first introduce two sets of LB particle distribution functions, one for each equation (i) and (ii), and then find the corresponding mesoscopic equilibrium distribution functions, which reproduce the desired macroscopic equations. To discretize the 2D-LBM equations, we use the so-called D2Q9 set of particle velocities (Fig. 3.1), which is based on 9 discrete particle velocities in directions $\mathbf{e}_i$ defined as [19],

$$\begin{aligned}
&\boldsymbol{e}_0 = (0,0);\\
&\boldsymbol{e}_i = c(\cos((i-1)\pi/4), \sin((i-1)\pi/4)); \quad i = 1,3,5,7\\
&\boldsymbol{e}_i = \sqrt{2}c(\cos((i-1)\pi/4), \sin((i-1)\pi/4)); \quad i = 2,4,6,8
\end{aligned} \tag{3.1}$$

with Δx and Δt the lattice constant mesh and time step sizes, respectively, and $c = \Delta x/\Delta t$ defining the particle propagation speed on the lattice.

In the LBM, it is customary to use non-dimensional lattice variables (here denoted by a prime) scaled on the basis of a length scale λ, time scale τ and mass scale ϖ; thus, for the mesh parameters, $\Delta x' = \Delta x/\lambda, \Delta t' = \Delta t/\tau$ and $c' = c\tau/\lambda$. It is also customary to assume that $c' = 1$, which is akin to having the mesh Courant number be unity. If the length scale is further defined as $\lambda = \Delta x$, we then have $\Delta x' = 1$, which requires $\tau = \Delta t$ and $\Delta t' = 1$ as well. Hence, with these definitions, in lattice variables, we always have $c' = \Delta x' = \Delta t' = 1$ [19].

3.1. Lattice Bolzmann solution of momentum equation

3.1.1. Classical LBM solution of NS equations

The macroscopic continuity and momentum (i.e., NS) equations for compressible isothermal fluids read (using tensor notations),

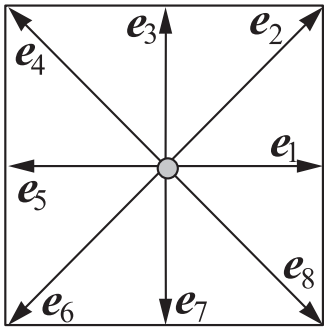


Fig. 3.1. D2Q9 lattice for definition of particle velocities.

$$\frac{\partial \rho}{\partial t} + \rho \frac{\partial u_\alpha}{\partial x_\alpha} = 0, \tag{3.2}$$

$$\rho\left\{\frac{\partial u_\alpha}{\partial t} + u_\beta \frac{\partial u_\alpha}{\partial x_\beta}\right\} = \frac{\partial \sigma_{\alpha\beta}}{\partial x_\beta} + B_\alpha, \tag{3.3}$$

where $\rho(\phi)$ is the local density of the two-fluid system, B_α is a body force (e.g., gravity: $B_\alpha = \rho g_\alpha$), and $\sigma_{\alpha\beta}$ denotes the stress tensor, which, for two-phase flow problems, can be decomposed into three parts [9],

$$\sigma_{\alpha\beta} = -p\delta_{\alpha\beta} + \sigma^{visc}_{\alpha\beta} + \sigma^{ST}_{\alpha\beta}, \tag{3.4}$$

where p is pressure, $\sigma^{visc}_{\alpha\beta}$ the viscous stress tensor, and $\sigma^{ST}_{\alpha\beta}$ a stress tensor representing the volumetric effects of surface tension forces at the two fluid interface. For Newtonian fluids, these tensors read,

$$\sigma^{visc}_{\alpha\beta} = \mu\left(\frac{\partial u_\alpha}{\partial x_\beta} + \frac{\partial u_\beta}{\partial x_\alpha}\right), \tag{3.5}$$

$$\sigma^{ST}_{\alpha\beta} = \left(\frac{k}{2}\frac{\partial \phi}{\partial x_\gamma}\frac{\partial \phi}{\partial x_\gamma} + k\phi\frac{\partial^2 \phi}{\partial x_\gamma \partial x_\gamma}\right)\delta_{\alpha\beta} - k\frac{\partial \phi}{\partial x_\alpha}\frac{\partial \phi}{\partial x_\beta}, \tag{3.6}$$

where $\mu(\phi)$ denotes the local dynamic viscosity, and the second equation directly follows from the definition of surface tension forces in Eq. (2.9) (i.e., $F_\alpha = \partial\sigma^{ST}_{\alpha\beta}/\partial x_\beta$).

Lee and Lin [9], however, reported that numerical schemes where the stress tensor is directly based on Eqs. (3.4), (3.5) and (3.6) are often unstable, as a result of an imbalance of the pressure and surface tension terms due to truncation errors, yielding parasitic currents near the phase interface. To alleviate this problem, they introduced a modified pressure p^m, which includes parts of the surface tension effects,

$$p^m = p + \frac{k}{2}\frac{\partial \phi}{\partial x_\gamma}\frac{\partial \phi}{\partial x_\gamma} - k\phi\frac{\partial^2 \phi}{\partial x_\gamma \partial x_\gamma}, \tag{3.7}$$

and a tensor $\sigma^{STm}_{\alpha\beta}$ also modified accordingly. They showed in applications that the variation of the modified pressure is now smooth across the phase interface, compared to that of pressure p, which greatly improves the stability of the numerical model in the case of large surface tension forces.

With these new definitions the total stress tensor is reformulated as,

$$\sigma_{\alpha\beta} = -p^m\delta_{\alpha\beta} + k\left(\frac{\partial \phi}{\partial x_\gamma}\frac{\partial \phi}{\partial x_\gamma}\delta_{\alpha\beta} - \frac{\partial \phi}{\partial x_\alpha}\frac{\partial \phi}{\partial x_\beta}\right) + \mu\left(\frac{\partial u_\alpha}{\partial x_\beta} + \frac{\partial u_\beta}{\partial x_\alpha}\right). \tag{3.8}$$

We now introduce a set of particle distribution functions $g_i(\boldsymbol{x}, t)$ to satisfy the equations of conservation of mass (3.2) and momentum (3.3), with the stress tensor defined by Eq. (3.8). In a standard LBM ansatz, the time evolution of these particle distribution functions is computed as (assuming a single relaxation time (SRT) formulation [20]) ($i = 0, \ldots, 8$),

$$g_i(\boldsymbol{x} + \boldsymbol{e}_i\Delta t, t + \Delta t) = g_i(\boldsymbol{x}, t) - \frac{\Delta t}{\tau_g}(g_i(\boldsymbol{x}, t) - g_i^{(eq)}(\boldsymbol{x}, t)) + \Delta t B_i, \tag{3.9}$$

where $g_i^{(eq)}$ denotes the equilibrium state, to which the particle distribution functions are locally driven, and B_i represents the effects of body forces B_α in Eqs. (3.3). Following Buick and Greated [21] the latter can be expressed as $B_i = w_i e_{i\alpha} B_\alpha / c_s^2$, where w_i is a weight factor (defined later). The relaxation time τ_g is related to the fluid viscosity and the assumed speed of sound in the medium, c_s (also detailed later).

With a proper definition of the equilibrium distribution functions, such LBM schemes converge to the solution of NS equations [11]. In standard LBMs used for two-phase flows with low density ratios, the macroscopic values of fluid density, momentum, and stresses are obtained from the moments of the particle distribution

functions [20]. Assuming no other body forces besides gravity, we have, respectively,

$$\sum_{i=0}^{b} g_i^{eq} = \rho, \tag{3.10}$$

$$\sum_{i=0}^{b} g_i^{eq} e_{i\alpha} = \rho u_\alpha, \tag{3.11}$$

$$\sum_{i=0}^{b} g_i^{eq} e_{i\alpha} e_{i\beta} = \left(\rho c_s^2 - k\frac{\partial\phi}{\partial x_\gamma}\frac{\partial\phi}{\partial x_\gamma}\right)\delta_{\alpha\beta} + k\frac{\partial\phi}{\partial x_\alpha}\frac{\partial\phi}{\partial x_\beta} + \rho u_\alpha u_\beta. \tag{3.12}$$

3.1.2. Modified LBM for two fluids with high density ratio

When using the classical LBM equations to simulate two fluid flows with high density ratio, the large density gradient near the interface between fluids will usually cause large truncation errors that could trigger numerical instabilities. To eliminate this problem, following Inamuro et al., we eliminate the bulk density from the previous equations, leading to modified NS equations, which no longer have a pressure gradient term (so-called pressureless NS equations). The velocity field $\boldsymbol{u}^*$ found as solution of these equations, however, will have to be corrected by solving an additional Poisson equation. This is detailed in the following.

First, we define a new equilibrium state where the moments of g_i's or $g_i^{(eq)}$'s are defined without the fluid density ρ as,

$$\sum_{i=0}^{b} g_i^{eq} = 0, \tag{3.13}$$

$$\sum_{i=0}^{b} g_i^{(eq)} e_{i\alpha} = u_\alpha^*, \tag{3.14}$$

$$\sum_{i=0}^{b} g_i^{(eq)} e_{i\alpha} e_{i\beta} = \left(c_s^2 - \frac{k}{\rho}\frac{\partial\phi}{\partial x_\gamma}\frac{\partial\phi}{\partial x_\gamma}\right)\delta_{\alpha\beta} + \frac{k}{\rho}\frac{\partial\phi}{\partial x_\alpha}\frac{\partial\phi}{\partial x_\beta} + u_\alpha^* u_\beta^*, \tag{3.15}$$

and the new equilibrium distribution functions for a D2Q9 model, which both modify and extend the classical formulation (e.g., [22]), read $(i = 0, \ldots, 8)$,

$$g_i^{(eq)} = w_i\left\{\frac{e_{i\alpha}u_\alpha^*}{c_s^2} + \frac{(e_{i\alpha}u_\alpha^*)^2}{2c_s^4} - \frac{|\boldsymbol{u}^*|^2}{2c_s^2}\right\} + w_i\frac{k}{\rho}G_{\alpha\beta}e_{i\alpha}e_{i\beta} - v_i\frac{k}{2\rho}|\nabla\phi|^2, \tag{3.16}$$

where the summation is performed over indices α and β (but not on i), c_s is the speed of sound defined, for a D2Q9 lattice as, $c_s = c/\sqrt{3}$ [19], and w_i and v_i are weighting functions defined as,

$$w_0 = \frac{4}{9}; \quad w_{1,3,5,7} = \frac{1}{9}; \quad w_{2,4,6,8} = \frac{1}{36}, \tag{3.17}$$

$$v_0 = -\frac{5}{3c^2}; \quad v_i = \frac{3}{c^2}w_i \quad (i = 1, 2, \ldots, 8), \tag{3.18}$$

and

$$G_{\alpha\beta}(\phi) = \frac{9}{2c^4}\frac{\partial\phi}{\partial x_\alpha}\frac{\partial\phi}{\partial x_\beta} - \frac{9}{4c^4}\frac{\partial\phi}{\partial x_\gamma}\frac{\partial\phi}{\partial x_\gamma}\delta_{\alpha\beta}. \tag{3.19}$$

[Note, these values of v_i and $G_{\alpha\beta}$ are similar to those of Inamuro et al.'s [10] and Swift et al.'s [23] model.]

It should be pointed out that, since pressure is solved for separately using a Poisson equation, the zeroth-order moments of equilibrium functions in the above Eq. (3.13) does not need to be equal to density (or pressure). This is unlike classical LB models where the zeroth-order moments of equilibrium functions needs to be equal to density to satisfy mass conservation. Here, the latter is enforced by the Poisson equation, which determines the pressure gradient in the resulting NS equations.

The convergence of the solution of these modified LBM equations to that of NS equations (without a pressure gradient term) is verified by applying the Chapman–Enskog expansion [11] to Eq. (3.9), with the equilibrium distribution functions of Eqs. (3.16)–(3.18) and (3.19). This leads to:

$$\frac{\partial u_\alpha^*}{\partial t} + u_\beta^*\frac{\partial u_\alpha^*}{\partial x_\beta} = \frac{k}{\rho}\frac{\partial}{\partial x_\beta}\left\{\frac{\partial\phi}{\partial x_\gamma}\frac{\partial\phi}{\partial x_\gamma}\delta_{\alpha\beta} - \frac{\partial\phi}{\partial x_\alpha}\frac{\partial\phi}{\partial x_\beta}\right\} + \frac{\partial}{\partial x_\beta}\left\{c_s^2\left(\tau_g - \frac{1}{2}\Delta t\right)\left(\frac{\partial u_\alpha^*}{\partial x_\beta} + \frac{\partial u_\beta^*}{\partial x_\alpha}\right)\right\} + \frac{B_\alpha}{\rho}, \tag{3.20}$$

whose right-hand-side should be the gradient of the stress tensor defined in Eq. (3.8) without a pressure gradient term. As there is no density in the second term in the right-hand-side, however, the relaxation time τ_g cannot immediately be related to the fluid kinematic viscosity ν, by contrast with a classical LBM [20]. To do so, we first have to rewrite the governing pressureless NS Eqs. (3.3) and (3.8) as,

$$\frac{\partial u_\alpha^*}{\partial t} + u_\beta^*\frac{\partial u_\alpha^*}{\partial x_\beta} = \frac{k}{\rho}\frac{\partial}{\partial x_\beta}\left\{\frac{\partial\phi}{\partial x_\gamma}\frac{\partial\phi}{\partial x_\gamma}\delta_{\alpha\beta} - \frac{\partial\phi}{\partial x_\alpha}\frac{\partial\phi}{\partial x_\beta}\right\} + \frac{\partial}{\partial x_\beta}\left\{\frac{\mu}{\rho}\left(\frac{\partial u_\alpha^*}{\partial x_\beta} + \frac{\partial u_\beta^*}{\partial x_\alpha}\right)\right\} - \mu\left(\frac{\partial u_\alpha^*}{\partial x_\beta} + \frac{\partial u_\beta^*}{\partial x_\alpha}\right)\frac{\partial}{\partial x_\beta}\left(\frac{1}{\rho}\right) + \frac{B_\alpha}{\rho}. \tag{3.21}$$

Now, Eq. (3.21) can be made identical to Eq. (3.20) if one defines $\mu/\rho = \nu = c_s^2(\tau_g - \frac{1}{2}\Delta t)$, which is the standard relationship in classical LBMs (e.g., [20]), and adds the next to last term of its right-hand-side to the LBM evolution Eq. (3.9) as an equivalent body force. According to Buick and Greated's formulation, this reads,

$$g_i(\boldsymbol{x} + \boldsymbol{e}_i\Delta t, t + \Delta t) = g_i(\boldsymbol{x}, t) - \frac{\Delta t}{\tau_g}\left(g_i(\boldsymbol{x}, t) - g_i^{(eq)}(\boldsymbol{x}, t)\right) + \frac{3}{c^2}w_i e_{i\alpha}\Delta t\left\{\frac{B_\alpha}{\rho} - \sigma_{\alpha\beta}^{visc,*}\frac{\partial}{\partial x_\beta}\left(\frac{1}{\rho}\right)\right\}, \tag{3.22}$$

where $\sigma_{\alpha\beta}^{visc,*}$ is given by Eq. (3.5), when using the pressureless velocity $\boldsymbol{u}^*$. It is noted that the last term in Eq. (3.22) is Buick and Greated's classical body force divided by density.

Based on the above Chapman–Enskog expansion, the relaxation time is thus expressed as,

$$\tau_g = \frac{\nu}{c_s^2} + \frac{1}{2}\Delta t, \tag{3.23}$$

Note, if one introduces the non-dimensional LBM kinematic viscosity $\nu' = \nu\tau/\lambda^2$ and speed of sound $c_s' = c'/\sqrt{3} = 1/\sqrt{3}$, Eq. (3.23) yields the standard LBM relaxation time, $\tau_g' = \nu'/c_s'^2 + \Delta t'/2 = 3\nu' + 1/2$.

3.1.3. Correction of velocity field based on a Poisson equation

Due to the absence of a pressure gradient term in the modified NS Eq. (3.21), the velocity field, $\boldsymbol{u}^*$, which is calculated at every time step with Eq. (3.14) based on the modified distribution functions, computed as a function of time with Eq. (3.22) in the LBM, is only an approximation of the actual velocity field $\boldsymbol{u}$. One additional step is thus required to both compute the pressure field and a corresponding correction $\Delta\boldsymbol{u}$ of the velocity, in order to satisfy the full NS Eqs. (3.2) and (3.3). Following Inamuro et al. [10] we define,

$$\boldsymbol{u} = \boldsymbol{u}^* + \Delta\boldsymbol{u} \quad \text{with} \quad \Delta\boldsymbol{u} \simeq -\Delta t\frac{\nabla p}{\rho}. \tag{3.24}$$

Thus, for the actual velocity field to satisfy continuity equation (i.e., $\nabla.\boldsymbol{u}=0$, assuming an incompressible fluid), $\boldsymbol{u}^*$ must satisfy the following Poisson equation,

$$\nabla\cdot\left(\frac{\Delta t\nabla p}{\rho}\right)=\nabla\cdot\boldsymbol{u}^*. \tag{3.25}$$

This Poisson equation (3.25) could be discretized and solved by various methods. Here, we iteratively solve it in the LBM framework, using the following evolution equation and a second set of particle distribution functions h_i $(i=0,\ldots,8)$,

$$h_i^n(\boldsymbol{x}+\boldsymbol{e}_i\Delta t,t+\Delta t)=h_i^n(\boldsymbol{x},t)-\frac{\Delta t}{\tau_h}\left(h_i^n-h_i^{(eq,n)}\right)-\Delta t w_i(\nabla\cdot\boldsymbol{u}^*(t)), \tag{3.26}$$

where n denotes the nth iteration in the solution. The equilibrium distribution functions are simply defined as,

$$h_i^{(eq,n)}=\frac{w_i p^n(\boldsymbol{x},t)}{\rho_0 c^2}, \tag{3.27}$$

where ρ_0 is a reference density. The relaxation time τ_h is related to the density by,

$$\tau_h=\Delta t\left\{\frac{1}{2}+\frac{\rho_0 c^2}{\rho c_s^2}\right\}, \tag{3.28}$$

or in LBM variables, $\tau_h'=1/2+\rho_0'/\rho'$, with $\rho'=\rho\lambda^3/\varpi$ and $\rho_0'=\rho_0\lambda^3/\varpi$. In this LBM scheme, pressure is simply calculated as the zero-th order moment of the particle distribution functions as,

$$p^{n+1}=\rho_0 c^2\sum_{i=0}^{b}h_i^n. \tag{3.29}$$

This scheme is iteratively run at a given time t, until the pressure field converges to a stable solution. Once this is achieved, the correction to the velocity field is calculated using Eq. (3.24). In our new method, the two previously derived LBM schemes thus solve, at time t: (i) the pressureless NS equations for high density ratios, with surface tension forces partly included in the formulation of the equilibrium distribution functions and in the body forces, which yields $\boldsymbol{u}^*$; and (ii) a Poisson equation using the approximate velocity field as an equivalent "volume force" to account for pressure gradients, which yields $\boldsymbol{u}$ and p. By contrast with sharp interface methods, the calculation of the interface curvature is not necessary in this method, but only the gradients of the phase field function ϕ are needed.

3.2. LBM for solving Cahn–Hilliard equation

The diffusive interface motion is modeled by the Cahn–Hilliard equation (2.12), where the left hand side describes the interface advection, and the right hand side the diffusive transport; M denotes the mobility and the chemical potential μ_ϕ is defined by Eq. (2.3), as a function of the bulk free-energy density Ψ and the phase field parameter ϕ. To solve this equation, we also use an LBM and introduce a third set of probability distribution functions, $f_i(\boldsymbol{x},t)$, whose evolution is again governed by a standard LBM scheme,

$$f_i(\boldsymbol{x}+\boldsymbol{e}_i\Delta t,t+\Delta t)=f_i(\boldsymbol{x},t)-\frac{\Delta t}{\tau_f}\left(f_i(\boldsymbol{x},t)-f_i^{(eq)}(\boldsymbol{x},t)\right). \tag{3.30}$$

This formulation also uses the SRT collision operator where, by analogy with Swift et al. [23], we define the moments of f_i to be the phase field parameter, its flux, and a higher-order moment, respectively, as,

$$\sum_{i=0}^{b}f_i=\phi, \tag{3.31}$$

$$\sum_{i=0}^{b}f_i e_{i\alpha}=\phi u_\alpha, \tag{3.32}$$

$$\sum_{i=0}^{b}f_i e_{i\alpha}e_{i\beta}=\frac{M\mu_\phi\delta_{\alpha\beta}}{\tau_f-\frac{1}{2}\Delta t}+\phi u_\alpha u_\beta. \tag{3.33}$$

The equilibrium distribution functions for $f_i(\boldsymbol{x},t)$ are further defined as,

$$f_i^{(eq)}=H_i\phi+v_i\frac{M}{\tau_f-\frac{1}{2}\Delta t}\mu_\phi+\phi w_i\left\{\frac{e_{i\alpha}u_\alpha}{c_s^2}+\frac{(e_{i\alpha}u_\alpha)^2}{2c_s^4}-\frac{|\boldsymbol{u}|^2}{2c_s^2}\right\}, \tag{3.34}$$

where w_i and v_i are the weighing functions defined in Eqs. (3.17) and (3.18), and

$$H_0=1,\quad H_i=0;\quad (i=1,2,\ldots,8). \tag{3.35}$$

It can be shown by Chapman–Enskog expansion that with these definitions, the LBM scheme solves the CH convection–diffusion equation.

Density ρ is assumed to vary smoothly across the two-fluid interface and is calculated throughout the LBM domain as a function of the order parameter ϕ, as,

$$\rho(\phi)=\begin{cases}\rho_2 & \phi\leqslant\phi_2\\ \frac{\phi-\phi_2}{\phi_1-\phi_2}(\rho_1-\rho_2)+\rho_2 & \phi_2<\phi<\phi_1\\ \rho_1 & \phi\geqslant\phi_1,\end{cases} \tag{3.36}$$

Similarly, both kinematic and dynamic viscosities are calculated as a function of density as,

$$\nu(\rho)=\frac{\rho-\rho_2}{\rho_1-\rho_2}(\nu_1-\nu_2)+\nu_2, \tag{3.37}$$

$$\mu(\rho)=\frac{\rho-\rho_2}{\rho_1-\rho_2}(\mu_1-\mu_2)+\mu_2, \tag{3.38}$$

respectively.

3.3. Boundary conditions

Wall boundary conditions are introduced here, for the three LBM distribution functions f_i, g_i, and h_i.

For f_i and g_i, which are used for the fluid momentum and interface tracking equations, Eqs. (3.11) and (3.32) indicate that the first-order moment is related to the macroscopic velocity, as in standard LBM approaches. Thus, for no-slip boundary conditions along solid walls, velocities are zero and hence the unknown particle distribution functions can be obtained from standard LBM bounce back schemes. In those, particles are specified to reflect back off the wall, into the fluid domain, resulting in a zero fluid velocity at the wall surface [20].

For the boundaries with periodic conditions, the unknown particle distribution functions on one boundary are set equal to the particle distribution functions on the other boundary, where the periodicity condition has been implemented [20].

At a stationary wall, the boundary condition of distribution functions h_i, used in the LBM solution of the pressure Poisson equation, follows from the NS momentum equation. For instance, assuming that the wall is planar and perpendicular to gravity $\boldsymbol{g}=-g\boldsymbol{j}$, Eqs. (3.2), (3.3) and (3.4) yield,

$$\frac{\partial p}{\partial y}=-\rho g+\mu\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right). \tag{3.39}$$

An approximation of the pressure at the wall ($x = x_w$) as a function of its values at neighboring lattice points can then be obtained from a Taylor series expansion in the direction perpendicular to the wall as,

$$p(x_w, y_w, t) = p(x_w, y_w - \Delta x, t) - \frac{\partial p}{\partial y}(x_w, y_w, t)\Delta x + \mathcal{O}(\Delta x^2), \quad (3.40)$$

where we can substitute the pressure gradient from its value in Eq. (3.39). Based on this equation, boundary values of the unknown particle distribution functions h_i are finally specified at the wall by assuming these are equal to the equilibrium distribution functions [24], $h_i(x_w, t) = h_i^{(eq)}(p(x_w, t))$, calculated with Eq. (3.27). For more complex geometries, same procedure can be done by finding the pressure value on the neighbor lattice node to the wall. compute the pressure on the wall bye a Taylor series and then use Eq. (3.27) to find unknown distribution functions.

3.4. Computation of spatial derivatives

The first and second spatial derivatives in the various LBM equations defined above (e.g., Eqs. (2.3), (3.16), (3.19), (3.22) and (3.24)) are computed using the following centered finite difference schemes, that are typical of standard LBM implementations [9], i.e., for the D2Q9 scheme and $\alpha = 1, 2$,

$$\frac{\partial \Phi}{\partial x_\alpha}(\boldsymbol{x}) = \sum_{i=1}^{9} w_i e_{i\alpha} \frac{\Phi(\boldsymbol{x} + \boldsymbol{e}_i\Delta t) - \Phi(\boldsymbol{x} - \boldsymbol{e}_i\Delta t)}{2c_s^2\Delta t} + \mathcal{O}(\Delta x^2), \quad (3.41)$$

and,

$$\frac{\partial^2 \Phi}{\partial x_\alpha^2}(\boldsymbol{x}) = \sum_{i=1}^{9} w_i \frac{\Phi(\boldsymbol{x} + \boldsymbol{e}_i\Delta t) - 2\Phi(\boldsymbol{x}) + \Phi(\boldsymbol{x} - \boldsymbol{e}_i\Delta t)}{c_s^2\Delta t^2} + \mathcal{O}(\Delta x^2), \quad (3.42)$$

for an LBM cell of coordinate $\boldsymbol{x}$, with Φ denoting any of relevant flow parameter and w_i the weight factors defined in Eq. (3.17). Along the boundary, except when a periodicity condition is specified, we use first-order de-centered finite difference schemes.

Algorithm 1. Algorithm for LBM computation of flow fields and phase interface updating

for $t < t_{end}$ **do**
  Compute $f_i(\boldsymbol{x}, t + \Delta t)$ using Eq. (3.30)
  Compute $g_i(\boldsymbol{x}, t + \Delta t)$ using Eq. (3.22),
  Compute $\phi(\boldsymbol{x}, t + \Delta t)$ and $\boldsymbol{u}^*(\boldsymbol{x}, t + \Delta t)$ with Eqs. (3.31) and (3.14);
  $\rho(\boldsymbol{x}, t + \Delta t)$ and $\nu(\boldsymbol{x}, t + \Delta t)$ are calculated using Eqs. (3.36) and (3.37)
  while $\left|\frac{p^{n+1} - p^n}{p^n}\right| > \varepsilon$ **do**
    Compute $p^{n+1}(\boldsymbol{x}, t + \Delta t)$ using Eqs. (3.26)–(3.28) and (3.29).
  end while
  Compute $\boldsymbol{u}(\boldsymbol{x}, t + \Delta t)$ using Eq. (3.24).
end for

3.5. Summary of LBM algorithm

The resulting LBM algorithm for the calculation of 2D flows of two fluids having a high density ratio, and the updating of the phase interface is summarized in Algorithm 1.

As indicated in the introduction, although we built our work in part based on Inamuro et al.'s [10] results, there are significant differences between our approach and theirs. These are summarized in the following. First of all, in their work, they solve the following equation for the interface tracking,

$$\frac{\partial \phi}{\partial t} + \frac{\partial(\phi u_\alpha)}{\partial x_\alpha} = \Theta \frac{\partial^2 P_{\alpha\beta}}{\partial x_\alpha x_\beta}, \quad (3.43)$$

where Θ is a diffusion coefficient and $P_{\alpha\beta}$ is defined as,

$$P_{\alpha\beta} = \left[\phi \frac{\partial \Psi}{\partial \phi} - \Psi - k_f \phi \frac{\partial^2 \phi}{\partial x_\gamma \partial x_\gamma} - \frac{k_f}{2}\frac{\partial \phi}{\partial x_\gamma}\frac{\partial \phi}{\partial x_\gamma}\right]\delta_{\alpha\beta}. \quad (3.44)$$

By comparing this equation with the Cahn–Hilliard equation (2.12) used in our model, we see that the right hand sides of each equation are different and, unlike in our case, Inamuro et al. did not provide a clear physical interpretation for their equation.

Second, Inamuro et al. defined three separate variables k_f, k_g and T for calculating surface tension and interface thickness. As showed before, in our method, only two different coefficients k and β are used, which can be expressed as a function of interface thickness W and the surface tension coefficient σ_{12}, using Eqs. (2.7) and (2.11).

Third, in Inamuro et al.'s scheme, the particle distribution functions g_i's used for calculating the hydrodynamic fields are just solving the advection part of the (NS) momentum equation. The effect of the viscous stress tensor is implemented by adding an extra term to the collision part of the g_i equation (corresponding to our Eq. (3.9)), and the viscosity effects resulting from this extra term have no clear physical interpretation. By contrast, in our model, dynamic viscosity is rigorously related to relaxation time in a way that is consistent with classical LBM schemes.

Finally, and importantly, our LBM scheme is fully optimized and implemented as a highly efficient parallel code on a GPGPU hardware, as summarized in Section 4.

4. GPGPU implementation of the LBM code

GPGPUs are computing hardware with a large number of cores (448 on the nVIDIA Tesla C2070), and a shared memory (6 MB for the nVIDIA Tesla C2070), that execute a number of computing threads in parallel. To manage these threads, the CUDA programming environment offers two levels of parallelism. First, all the threads are grouped in one *thread block* where extremely fast memory is shared between the threads, which can be synchronized. Each thread is identified by its three-dimensional thread index, which gives the position in the thread block within the hardware. To efficiently use the hardware, the total number of threads per block should be in the range of 64–512. This number can be adjusted up or down depending on the size of local and shared memory available on each particular GPGPU. Threads are executed in warps containing 32 threads each, on one of the GPGPU multiprocessors. Second, the thread blocks are bundled into the *grid*. Unlike threads located within the same thread block, threads in different blocks can only communicate via the GPGPU shared memory and a synchronization is not possible. Blocks are identified by their two-dimensional block index, namely their position within the grid. Further details on the thread processing, grouping in warps, and distribution among the GPGPU multiprocessors can be found in [25,26].

4.1. Topology and grid mapping

The main design element in the GPGPU implementation of a numerical method is the mapping of the numerical grid onto the computational hardware, i.e., in our case the mapping of LBM nodes onto the GPGPU processors, blocks, and threads. In earlier GPGPU hardwares, several restrictions existed on memory access

patterns that needed to be taken into account in order to achieve maximum performance (see, e.g., [12]). However, recent GPGPUs dedicated to numerical computations offer higher flexibility, so that in this model we decided not to use the earlier shared memory particle propagation pattern but instead to access the GPGPU main memory directly in the propagation step. Hence, by contrast with earlier implementations, the thread blocks can be designed almost arbitrarily.

Specifically, in our grid mapping, we assign one single lattice node to one CUDA thread. The memory is allocated as a one-dimensional array and, as proposed by [16], the memory index is calculated as `k = nx * y + x`, for a node at position (x, y) and a total of $n_x \times n_y$ nodes. The dimension of the grid and the number of threads are specified in the CUDA kernel call and the coordinates of an LBM node can then be determined via, `x = threadId.x`, and `y = blockIdx.x`. After the kernel launch, CUDA manages the exact distribution of tasks among the multiprocessors and cores on the GPGPU.

4.2. Implementation details

The implementation of most of the LBM kernels is straightforward, as demonstrated in [16]. The CUDA interface supports C-style programming, so that standard C codes written for single processors can be easily transferred. Note that all the computations in this work require double precision variables to ensure accuracy and convergence. On the latest nVIDIA boards, double precision computations are only a factor of 2 slower than single precision ones. Due to the doubled memory requirement of double precision, memory transfers are also a factor of 2 slower, so that for our LBM algorithm, we can approximately estimate that the performance in double precision is half that of single precision.

In general, the performance of our proposed LBM multiphase model highly depends on the number of Poisson iterations performed at each time step, which depends on the problem physics. Hence no generally valid performance value of the multiphase scheme can be given and performance must instead be assessed on a case-by-case basis. Performance details are given below for the applications presented in the validation section.

4.2.1. Memory allocation

In GPGPU implementations, data transfer between the host (i.e., the CPU computer controlling the GPGPU hardware) and GPGPU memory, usually significantly penalizes performance and hence must be minimized. To do so, in this LBM, unlike in previous implementations, we do not allocate host memory for the full 3 sets of particle distribution functions (PDFs), but instead these are only allocated on the GPGPU. Hence, in the post-processing step, which involves data transfer from the GPGPU to the host, only the macroscopic values, such as pressure, velocity and phase field parameter (i.e., 3–5 double precision variables) are copied to the host memory, instead of copying the full sets of PDFs.

Additionally, in GPGPUs, the memory is accessed as one single vector (with the limitation in CUDA C codes that the function parameter space of the kernel calls be less than 256 bytes). In a D2Q9 double precision model, $2 \times 9 \times 8 = 144$ bytes are needed for pointers to the GPGPU memory, to refer to the kernels where data is located. To reduce the number of pointers, the memory for all PDFs is allocated at once, leading to a single linear memory segment. The individual memory locations are then computed within each thread. Hence, as the data layout structure is clearly defined, it is sufficient to only send the start address of the PDF arrays to the kernels. This saves function parameter space, which can thus be used for storing pointers to other variables, such as the macroscopic variables, derivatives, and so on.

4.2.2. Boundary conditions

Boundary conditions (BCs) disturb the flow of the LBM algorithm, as they require additional operations on a specific subset of nodes; hence this affects model performance. In general, a single LB kernel for all lattice nodes is preferable on a data- and thread-parallel system, for optimal load balancing. However, this is not possible for all the boundary conditions used in this work, which require additional kernel launches to process the particle distribution functions. To optimize parallel LBM computations, in our model, ghost layers of lattice nodes surround the whole computational domain, so that all particle distribution functions can be advected to neighboring nodes (propagation step), even at the domain boundary; this way, no logical test is required inside the LBM kernels regarding boundaries. Then, after the standard kernels for collision and propagation have been run, BCs are applied. No-slip BCs for instance are simply specified by bouncing the PDFs, that have been advected into the ghost layer, back into the domain (bounce-back scheme). In extrapolation (or open) boundary conditions, values from the next-to-last fluid node are copied to the last one. Here, the problem of thread synchronization leads to a second justification for using separate BC kernels and launching them after the calculation of the flow field is complete. Such BCs indeed rely on consistent and valid particle distribution functions at the neighboring lattice nodes, so that these nodes need to have terminated their flow field updating before the BCs can be applied. In general, we found that the additional computational overhead related to the BC kernel launches is more than compensated by a higher flexibility in setting up BCs in the model (i.e., allowing to easily switch boundary conditions), and an easier model upgrade (i.e., implementation of further boundary condition types without having to modify the basic LB kernels).

4.2.3. Convergence check for Poisson iteration

A loop over all lattice nodes is needed to evaluate the maximum error during iterations in the solution of the pressure Poisson equation. To improve performance, this error is only computed on the GPGPU without copying all node results back to the host memory. However, such operations in thread-parallel systems require a careful treatment to avoid "race conditions" among threads, which would lead to inaccuracies. Thus, a first kernel computes the maximum error in one thread block, where local errors are calculated for each computational node and the maximum error in the block is then calculated in a "divide and conquer" strategy, with the help of the thread-global shared memory and thread block synchronization points. Maximum errors of each thread block are stored in the GPGPU memory (as a vector of n_y values), and the same steps are applied to calculate the global maximum error among blocks, which is finally copied to the memory of the CPU host. The latter is responsible for kernel flow control and stopping the Poisson iterations, once a certain relative error threshold ε is reached. [Although, in our algorithm, convergence of the Poisson equation solution is checked after each iteration, this is not necessary and a minimum number of iterations could be set beyond which convergence would be checked, leading to a slightly more efficient solution.] The relative error is checked every 20 iteration to the performance of the computations.

4.2.4. Computational algorithm

The GPGPU computational algorithm is summarized in Algorithm 2, including the previously mentioned approach for boundary conditions and the Poisson iteration convergence check. Moreover, additional kernels are introduced for the calculation of derivatives, such as the divergence of the predicted (pressureless) flow field $\nabla \cdot \boldsymbol{u}^*$. Before the time loop starts, several iterations of the phase field kernel (with a zero velocity field) are run, to let the interface between the two fluids converge to an initial equilib-

rium diffusive interface shape. This is of great importance when no analytical solution for the initial interface shape exists.

Algorithm 2. GPGPU implementation of the LBM multiphase algorithm

Allocate memory on host (CPU) and device (GPGPU)
Initialize phase field ϕ, velocity $\boldsymbol{u}$, and pressure p on the host
Copy phase field ϕ, velocity $\boldsymbol{u}$, and pressure p from the host to the GPGPU memory
Initialize the particle distribution functions on the GPGPU, consistent with the initial conditions

for $t < t_{init}$ **do**
  Update chemical potential μ_ϕ by Eq. (4)
  LB kernel for the phase field ϕ, based on Eq. (3.30)
  BC for the phase field ϕ (periodic or bounce-back)
  Update ϕ, ρ, ν, and μ according to Eqs. (3.31), (3.36), (3.37) and (3.38)
end for

for $t < t_{end}$ **do**
  Update chemical potential μ_ϕ
  LB kernel for the phase field ϕ, based on Eq. (3.30)
  BC for the phase field ϕ (periodic or bounce-back BCs)
  LB kernel for the flow field $\boldsymbol{u}^*$, based on Eq. (3.9),
  BC for the flow field (periodic or bounce-back)
  Update ϕ, ρ and ν, according to Eqs. (3.31), (3.36) and (3.37)
  Calculate predicted flow field $\boldsymbol{u}^*$ as given by Eq. (3.14)
  Calculate the divergence of the predicted flow field, $\nabla \cdot \boldsymbol{u}^*$ (check every 20 iteration)
  if *PmaxError* $> \varepsilon$ **then**
    LB kernel for the Poisson equation (Eqs. (3.26)–(3.28) and (3.29))
    BC for the Poisson equation
    calculate maximum error *PmaxError* on Poisson solution
  end if
  Correct the predicted flow field $\boldsymbol{u}^*$ with Eq. (3.25)
end for

4.3. Performance

As indicated before, the performance of our LBM multiphase model highly depends on the number of Poisson iterations performed at each time step, which depends on the problem physics. Hence performance is assessed on a case-by-case basis for each case presented in the application Section 5. Thus, Table (4.1) lists the average value of "Million Node Updates Per Second" (MNUPS) achieved for the different test cases. We see that the best performance (52) is achieved in the first case, which does not require solving the pressure Poisson equation, while the other cases, which require such a solution have their performance reduced by a factor of 5–6. For comparison, a FORTRAN single-processor (CPU) implementation done earlier for a similar code achieved MNUPS values about 40 times smaller than on the GPGPU.

5. Applications

Here, we present applications that validate our newly developed LBM for multiphase flows, by comparing numerical results to the solution of analytical and experimental benchmark problems. First, the LBM scheme is applied to simulating a two-fluid laminar Poiseuille flow between infinite plates, for which there is an analytical solution; this assesses the method's accuracy in the absence of surface tension effects (since the fluid interface curvature is zero) in case of large density gradients. Second, we solve the case of a stationary bubble of a lighter fluid in a non-moving heavier fluid, in the absence of gravity; this validates the computation of surface tension effects, by comparison with Laplace's law. Then, we model a lighter fluid bubble rising in a heavier stationary fluid, and compare LBM results to other numerical results, which provide an independent reference simulation. Finally, we apply the LBM model to more complex two-fluid flows with a large density ratio: (i) a Rayleigh–Taylor instability and (ii) ocean wave breaking.

Table 4.1
Performance for different test cases presented in the application section.

Case	MNDUPS
Two-fluid Poiseuille flow	52
Stationary bubble	12
Rising bubble (case (a))	9.6
Rising bubble (case (b))	8.4
Rising bubble (case (c))	8.2
Rayleigh–Taylor instability	11
Breaking wave	8.1

5.1. Definitions

In all the following simulations, values are provided for non-dimensional lattice variables (denoted by a prime), which are the parameters actually used in LBM computations. Besides having $\Delta x' = \Delta t' = c' = 1$, as stated before, in all cases the non-dimensional relaxation time for solving the momentum equations is kept within the limits, $0.5 < \tau'_g \leqslant 1$, in order to ensure stability of the LBM solution [20]. With Eq. (3.23), this requires $0 < \nu' \leqslant 1/6$, and hence the time step is defined as, $\Delta t = (\Delta x)^2 \nu'/\nu \leqslant (\Delta x)^2/(6\nu)$. The non-dimensional surface tension coefficient is further defined as,

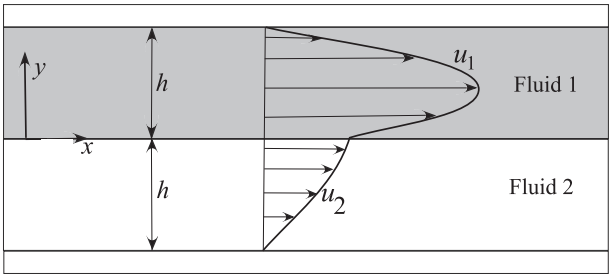


Fig. 5.1. Definition sketch of a two-fluid Poiseuille flow between infinite plates, with typical analytical solution for the horizontal velocity profile $u(y)$.

Table 5.1
L_2-norm error (between analytical and LB results) and body force magnitude g', as a function of LBM discretization size h', for the two-fluid Poiseuille flow test case of Fig. 5.2 (see, Fig. 5.3 for error plot).

Case	h'	g'	L_2-norm (%)
(a)	50	2.871×10^{-08}	6.2
(b)	60	2.390×10^{-08}	5.4
(c)	70	2.051×10^{-08}	4.4
(d)	100	1.436×10^{-08}	3.2
(e)	125	1.149×10^{-08}	2.7
(f)	150	9.572×10^{-09}	1.9

$$\sigma'_{12} = \frac{\sigma_{12}\tau^2}{\rho_0 \lambda^3} = \frac{\sigma_{12}}{\rho_0 c^2 \Delta x}. \tag{5.1}$$

The non-dimensional relaxation time for solving the Cahn–Hilliard equation (3.30) is set to $\tau'_f = \tau_f/\Delta t = 1$. The non-dimensional mobility $M' = M\varpi/(\tau\lambda^3)$ is set to $M' = 0.02/\beta'$ to get the most stable results [27]; additionally, we specify $\phi_1 = 0.4$ and $\phi_2 = 0.1$ in all cases, and the interface thickness is assumed to be of 4 lattice meshes: $W = 4\Delta x$ or $W' = 4$, which by combining Eqs. (2.7) and (2.11) yields the interface lattice parameters,

$$\beta' = \frac{3\sigma'_{12}}{(\phi_1 - \phi_2)^4} \quad \text{and} \quad k' = \frac{6\sigma'_{12}}{(\phi_1 - \phi_2)^2}, \tag{5.2}$$

where $\beta' = \beta/(\rho_0 c^2)$ and $k' = k/(\rho_0 c^2 \lambda^2)$. Although other values of W (both smaller and larger) were tested in applications, the selected interface thickness was found to yield the most accurate results in applications, as compared to reference results, by ensuring a sufficient sharpness of the interface gradients while not causing numerical instabilities with unnecessary large gradients.

The non-dimensional relaxation time used for solving the pressure Poisson equation (3.26) is similarly kept within the range $0.5 < \tau'_h \leqslant 1$ [24]; with the latter constraint, Eq. (3.28) yields that for either fluid the non-dimensional density $(\rho'_1, \rho'_2) \geqslant 6$. Finally, the non-dimensional reference density is set to $\rho'_0 = 1$, which implies that, $\varpi = \rho_0(\Delta x)^3$ (usually, one will also assume for simplicity, $\rho_0 = 1\ \text{kg/m}^3$) and $\rho'_i = \rho_i/\rho_0$, for fluid $i = 1, 2$.

Fig. 5.3. L_2-norm error (between analytical and LB results) as a function of LBM discretization size h', for the two-fluid Poiseuille flow test case (Fig. 5.1): (•) data from Table 5.1; (——) power curve fit $L_2 \propto (h')^{-1.008}$ ($R^2 = 0.99$).

5.2. Two-fluid Poiseuille flow

The two-fluid Poiseuille flow, between two infinite plates, inclined at an angle α with respect to the horizontal, is a good

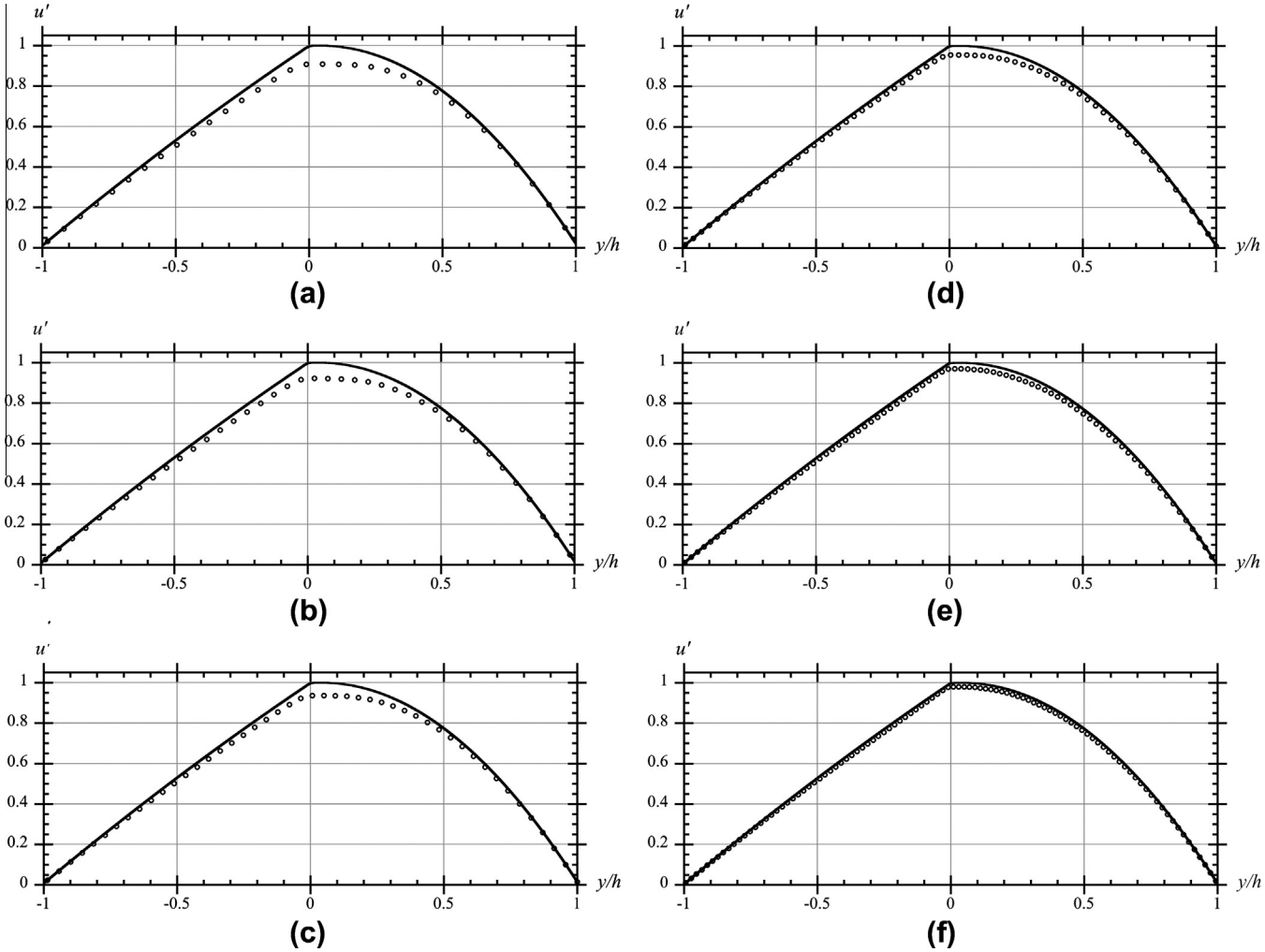


Fig. 5.2. Non-dimensional velocity profiles $u' = u/u_1^{max}$ in two-fluid Poiseuille flow (Fig. 5.1), for $\text{Re} = 100, \text{Ma} = 0.01, \rho_1/\rho_2 = 100$, and $\nu_1/\nu_2 = 0.1$: (——) analytical; (○) LB results (only 33% of nodes are plotted for clarity), for different grid resolutions h' (see, Table 5.1).

analytical test case to validate the method for both high fluid dynamic viscosity and density ratios in the absence of surface tension effects. Two immiscible fluids are accelerated in between the plates by a body force (i.e., projected gravity, $\rho g \sin\alpha$) and slowed by the viscous shear along the plate surfaces (Fig. 5.1). At the planar two-fluid interface, the continuity of fluid velocity and stresses has to be satisfied and, in such a case, geometry also implies that surface tensions forces are zero. An analytical solution of the steady state NS equations can be derived for this case, to which the fully developed velocity field computed by the LBM can be compared.

Specifically, the analytical reference solution is derived by solving the following simplified, but exact, one-dimensional momentum equation for the horizontal velocity component u_i in each fluid $(i = 1,2)$ (with $y = x_2$),

$$-\rho g \sin\alpha = \mu \frac{d^2 u_i}{dy^2}, \tag{5.3}$$

which yields [28],

$$u_1 = \frac{g\sin\alpha}{2}\left\{h^2\frac{(\rho_1+\rho_2)}{(\mu_1+\mu_2)} - yh\frac{(\mu_1\rho_2-\mu_2\rho_1)}{\mu_1(\mu_1+\mu_2)} - y^2\frac{\rho_1}{\mu_1}\right\}, \tag{5.4}$$

$$u_2 = \frac{g\sin\alpha}{2}\left\{h^2\frac{(\rho_1+\rho_2)}{(\mu_1+\mu_2)} - yh\frac{(\mu_1\rho_2-\mu_2\rho_1)}{\mu_2(\mu_1+\mu_2)} - y^2\frac{\rho_2}{\mu_2}\right\}. \tag{5.5}$$

with fluid densities ρ_1 and ρ_2, and dynamic viscosities μ_1 and μ_2, respectively; g denotes the gravitational acceleration and $2h$ is the distance between the 2 plates. This analytical solution yields velocity profiles that are parabolic in each fluid, with a discontinuity of the vertical velocity gradient at the interface, owing to the identical horizontal stress on either side of the interface, for different viscosity values (Fig. 5.1).

LBM simulations are started from a state of rest in a 2D channel similar to that sketched in Fig. 5.1. A periodic boundary condition is specified for lateral upstream and downstream boundaries, in the flow direction $x = x_1$, and no-slip boundary conditions are specified on the plate surfaces ($y = \pm h$). The interface parameters of the system are set to $k' = 0.01$ and $\beta' = 0.05$. Thanks to the periodicity in flow direction and the laminar nature of the flow, the grid resolution in the x direction can be low, and only 4 grid points were used. The number of grid points in the vertical y direction is $N = 2h' = 100$–300, with $\Delta x = 2h/N$ (Table 5.1).

A convergence study of LB results accuracy as a function of mesh size N is performed for a series of flows defined by a constant Reynolds number, $\mathrm{Re} = u_1^{max} 2h/\nu_1 = 100$ (based on maximum velocity in the channel, assuming this occurs in fluid 1, corresponding kinematic viscosity, and channel width); the LBM Mach number in these computations is also fixed at, $\mathrm{Ma} = u_1^{max}/c_s = 0.01$, which provides sound speed c_s to use for a given flow. For each flow parameters and LBM discretization N or h', the non-dimensional body force magnitude, $g' = g\sin\alpha(\tau^2/\lambda) = g\sin\alpha\Delta t/c^2$ (with $c^2 = 3c_s^2$) is adjusted by varying the channel angle α, in order for the maximum flow velocity u_1^{max} (obtained from the analytical solution) to satisfy the desired Reynolds number. In practice, given fluid and geometry parameters $\rho_1/\rho_2, \nu_1/\nu_2, h$ and g and combining the expression for u_1^{max} with the definitions of Re and Ma, we find g' as a function of h' (Table 5.1).

Let us first assume, the density of fluid phase 1 is $\rho_1' = 600$ and the density ratio is $\rho_1'/\rho_2' = 100$ (thus $\rho_2' = 6$); the kinematic viscosity ratio is $\nu_1/\nu_2 = 0.1$, which yields $\mu_1/\mu_2 = 10$. To check the accuracy of numerical results, we calculate the L_2-norm relative error (i.e., a RMS error between numerical and analytical velocities u, scaled by the RMS of the analytical velocity). LBM simulations were stopped when the difference between the L_2-norm of two consecutive time steps became less than 10^{-6}. Fig. 5.2 shows the analytical and numerical (steady-state) velocity profiles for different grid sizes h', and the convergence to the analytical reference solution is seen to be quite good. In previous studies of two-phase Poiseuille flows with LBM multiphase models [29,30], numerical oscillations of the fluid velocity were observed near the phase interface, even for low density ratios. Our LBM multiphase method does not trigger such oscillations and captures very well the slope discontinuity of the velocity profile at the phase interface. Table 5.1 summarizes the L_2-norm errors for different grid configurations. Convergence can clearly be observed. Fig. 5.3 further shows a first-order convergence of the relative error is achieved, as a function of the number of lattice nodes in the y direction.

Simulations are run next for different viscosity and density ratios, using a fixed LBM grid resolution with $N = 300$ in the y direction. Flow parameters are, $\mathrm{Re} = 1000$ and $\mathrm{Ma} = 0.005$.

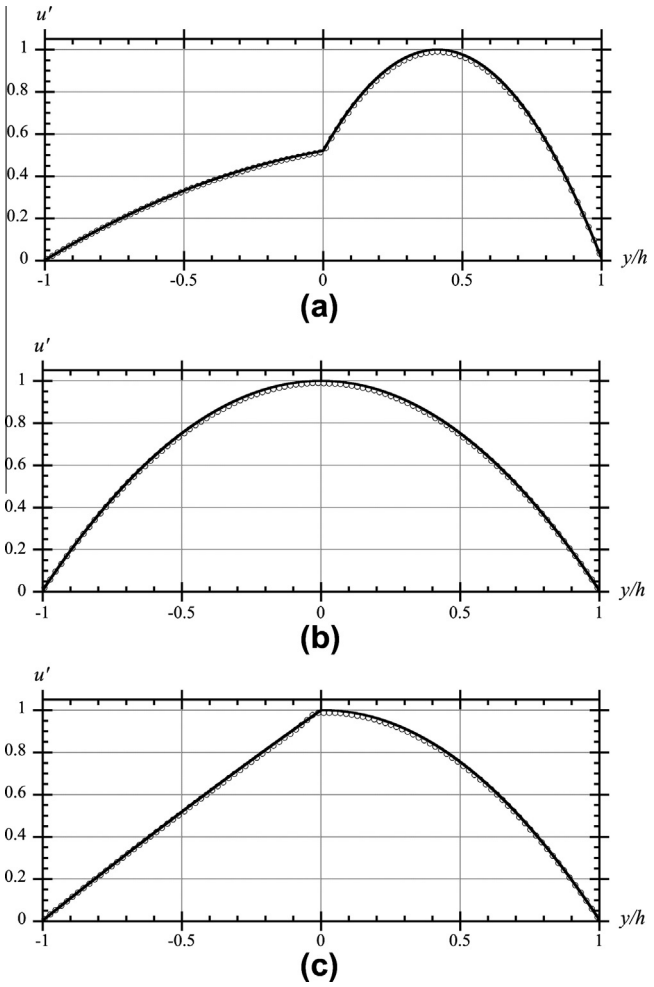


Fig. 5.4. Two-fluid Poiseuille flow (Fig. 5.1)). Non-dimensional velocity profiles $u' = u/u_1^{max}$ for $\mathrm{Re} = 1000$ and $\mathrm{Ma} = 0.005$: (––) analytical; (○) LB results for $N = 300$ LBM nodes (only 33% of nodes are plotted for clarity). (a) $\rho_1'/\rho_2' = 1$ and $\nu_1/\nu_2 = 0.1$, (b) $\rho_1'/\rho_2' = 100$ and $\nu_1/\nu_2 = 1$, and (c) $\rho_1'/\rho_2' = 1000$ and $\nu_1/\nu_2 = 0.0667$.

Table 5.2
Stationary bubble case. Comparison of computed and analytical (0.4 N/m^2) pressure jumps Δp for stationary circular droplets, as a function of LBM discretization.

$N_x = N_y$	σ'_{12}	k'	β'	Δp (N/m^2)	L_2-norm (%)
64	4.34×10^{-3}	0.28	1.6	0.389	2.7
128	2.17×10^{-3}	0.14	0.8	0.395	1.23
256	1.09×10^{-3}	0.07	0.4	0.397	0.72

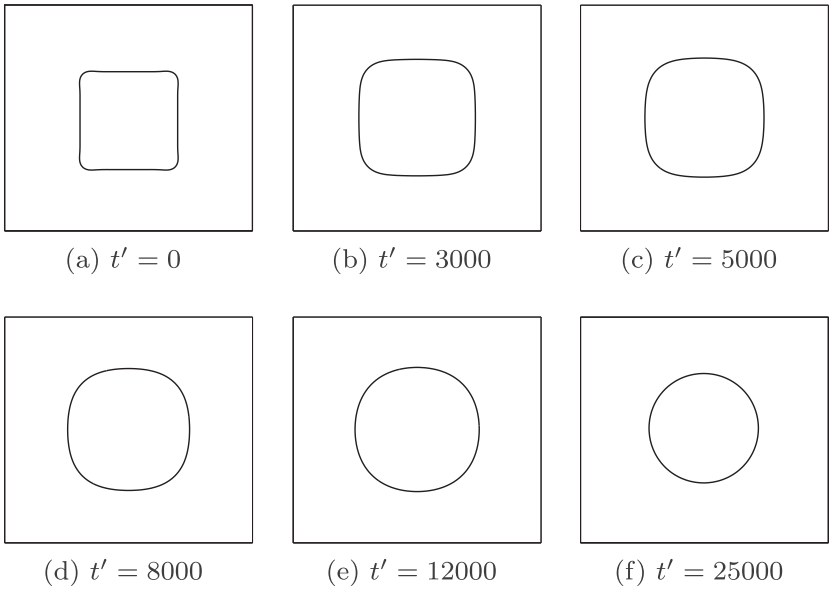


Fig. 5.5. Time evolution of the shape of an initially rectangular droplet towards a circle of radius $R' = 68$ $(\rho'_1 = 600, \rho_1/\rho_2 = 100)$, as a result of Laplace law; t' is non-dimensional time of computations (i.e., number of time steps).

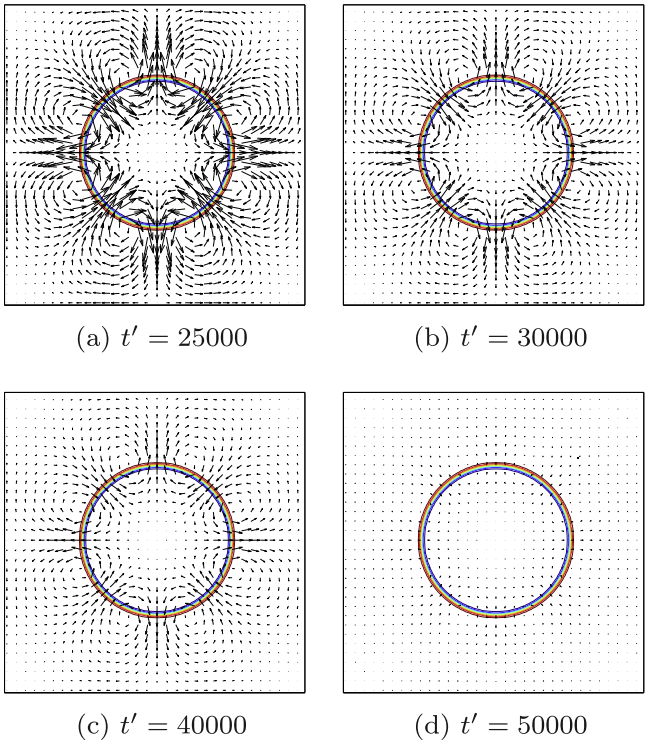


Fig. 5.6. Velocity vectors computed around the bubble interface at four different times equal or larger than the largest time in Fig. 5.5.

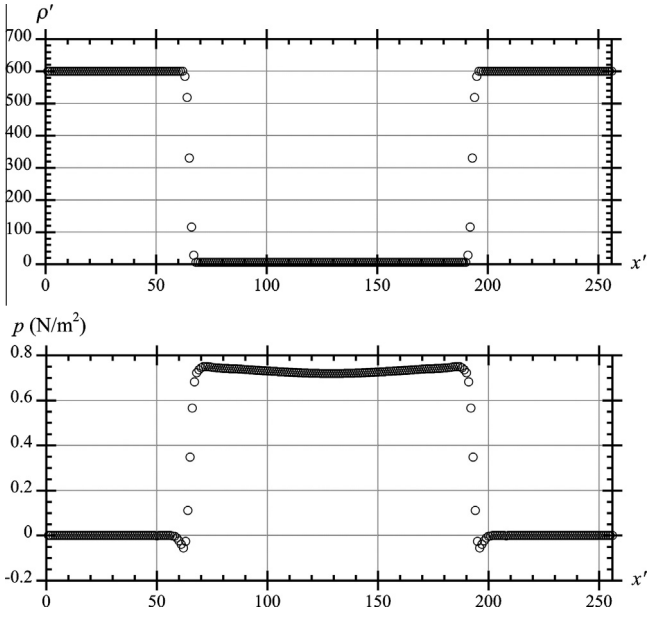
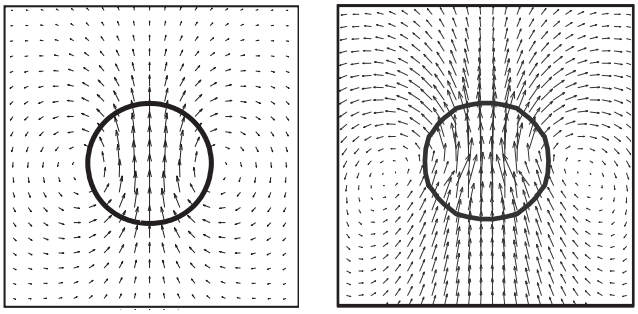


Fig. 5.7. Case of Fig. 5.5(f) (steady-state). (○) LBM simulation of non-dimensional density and pressure profiles across the domain mid-axis.

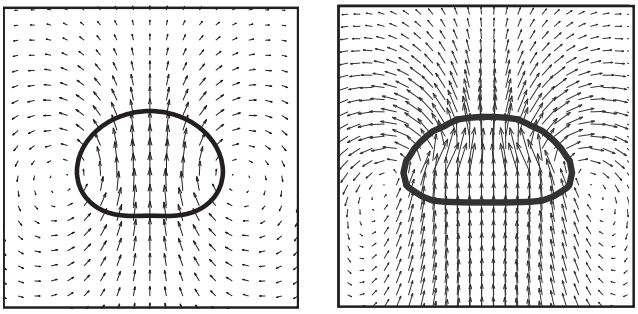
Table 5.3
LBM parameters for rising bubble computations.

Case	ν'_1	k'	β'	g'	Bo	Mo
(a)	2.0×10^{-3}	0.266	1.481	1.853×10^{-11}	0.1	0.001
(b)	2.0×10^{-3}	0.266	1.481	1.853×10^{-9}	10	0.1
(c)	1.0×10^{-2}	0.210	1.171	1.465×10^{-8}	100	1000

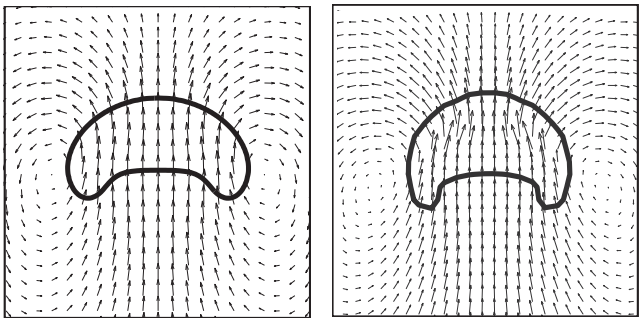
Fig. 5.4 shows steady-state velocity profiles for three different cases: (a) the fluid densities are identical $(\rho'_1/\rho'_2 = 1)$, and the kinematic viscosity ratio is $\nu_1/\nu_2 = 0.1$ (which yields different velocity gradients at the phase interface to satisfy the continuity of shear stress); (b) $\rho'_1/\rho'_2 = 100$ and $\nu_1/\nu_2 = 1$ (which yields continuous shear stress and a continuous velocity gradient at the phase interface: zero, due to the symmetry); and (c) $\rho'_1/\rho'_2 = 1000$ and $\nu_1/\nu_2 = 0.0667$, as for water and air, which are the fluids used in our final applications. The good agreement of numerical and analytical results, even in the latter case, confirms the accuracy of the LBM scheme and its applicability to practical problems such as at an air–sea interface.



(a) Bo=0.1, Mo=0.001



(b) Bo=10, Mo=0.1



(c) Bo=100, Mo=1000

Fig. 5.8. Terminal shape and flow velocity vectors for a bubble of density $\rho'_2 = 6$ rising under buoyancy in a fluid with density $\rho'_1 = 6000, \mu'_1/\mu'_2 = 1000$: (rightward panels) VOSET results of [33]; (leftward panels) present LB results (see Table 5.3).

5.3. Stationary bubble in quiescent fluid

The exact solution (in the absence of gravity, i.e., volume forces) for a circular stationary bubble of a lighter fluid embedded within a heavier quiescent fluid is used as a benchmark to validate surface tension force computations in our LBM scheme, on the basis of the Cahn–Hilliard equation. Laplace law predicts that, if the two-fluid interface is curved, a pressure jump Δp occurs across the interface (with pressure being higher along the concave side); for a two-dimensional circular bubble of radius R, we have,

$$\Delta p = \frac{\sigma_{12}}{R} \tag{5.6}$$

A circular droplet of fluid of density $\rho_2 = 10\ \mathrm{kg/m^3}$ and radius $R = 0.005$ m is placed in the middle of a square LBM domain with side $d = 0.02$ m, filled with a fluid of density $\rho_1 = 600\ \mathrm{kg/m^3}$ (hence, the density ratio is $\rho'_1/\rho'_2 = 60$). The fluid kinematic viscosities are $\nu_1 = \nu_2 = 2 \times 10^{-3}\ \mathrm{m^2/s}$, and the surface tension coefficient $\sigma_{12} = 2 \times 10^{-3}$ N/m. Eq. (5.6) predicts that this situation corresponds to a pressure jump $\Delta p = 0.4\ \mathrm{N/m^2}$ across the interface.

In LBM computations, we use all the fixed parameter values specified in the definition section and select a lattice viscosity, $\nu'_1 = \nu'_2 = 1/6$, i.e., $\tau'_g = 1$. Three grid sizes: $N_x = N_y = 64$, 128 and 256 are successively used and periodic boundary conditions are specified on the 4 sides of the square domain. The mesh size is, $\Delta x = d/N_x$ and time step, $\Delta t = (\Delta x)^2/(6\nu_1)$. The dimensionless lattice surface tension coefficient σ'_{12} is then given by Eq. (5.1) and the interface lattice parameters β' and k' follow from Eq. (5.2). Table 5.2 lists values of the latter coefficients and gives results for the pressure jump, computed as the difference between the average pressure inside and outside of the droplet, as a function of the discretization size. We see that numerical results are in good agreement with the analytical results, with a 0.72% L_2-norm error in the finest discretization. The convergence of numerical errors also appears to be first-order in grid size, as in the previous application.

To further validate the surface tension force computation for a non-stationary case, a similar simulation is repeated for an initially square droplet of a lighter fluid, with side $128\Delta x$, embedded within the larger square domain of side $d = 256\Delta x = 0.02$ m used earlier ($\Delta x = 7.813 \times 10^{-5}$ m), filled with a heavier fluid. The LBM simulation parameters are, $\rho'_1 = 600, \rho_1/\rho_2 = 100, \nu_1 = \nu_2 = 2 \times 10^{-3}\ \mathrm{m^2/s}$, $\sigma_{12} = 4 \times 10^{-3}$ N/m and $\tau'_g = 1$. Hence, $\Delta t = (\Delta x)^2/(6\nu_1) = 5.086 \times 10^{-7}$ s, $c = 153.6$ m/s, which yields $\sigma'_{12} = 2.17 \times 10^{-3}, \beta' = 0.8$ and $k' = 0.14$. Fig. 5.5 shows the time-evolution of the droplet geometry, which as expected gradually relaxes to a circle to

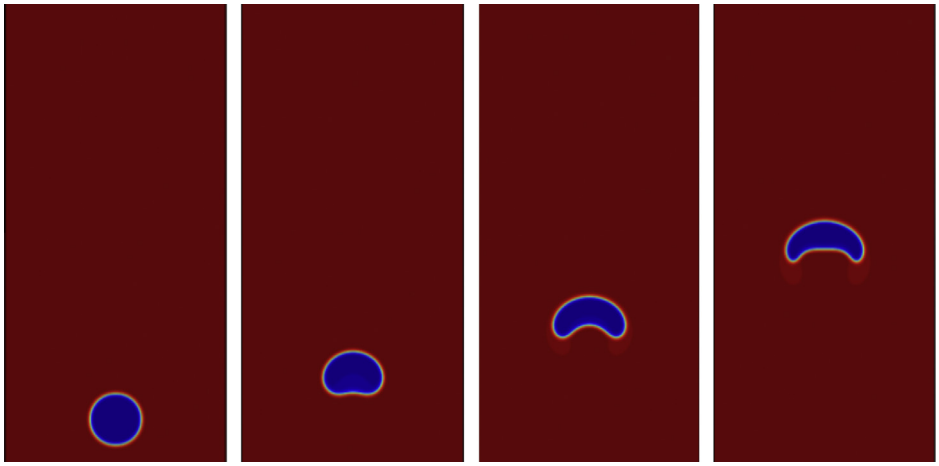


Fig. 5.9. Time evolution of a rising bubble shape for $\rho'_1/\rho'_2 = 1000, \mathrm{Bo} = 100, \mathrm{Mo} = 1000$ (case of Fig. 5.8(c)). From left to right, for $t' = 0, 10^3, 2 \times 10^3, 4 \times 10^3$. Note, only half the domain height is plotted.

both minimize and equalize the interface pressure jump, based on Laplace law. The relaxation time $\tau'_g = 1$ corresponds to a highly viscous fluid, so that the initial droplet relaxes to the equilibrium shape without significant oscillations.

In an accurate two-phase flow model, there should only be negligible flow velocities computed once the bubble reaches its equilibrium shape. In the LBM model, such spurious velocities are eliminated by the correction of the velocity field resulting from the solution of the Poisson equation for the modified pressure in Eq. (3.7). As a verification, Fig. 5.6 shows velocity vectors computed around the bubble interface as a function of time, once its shape has nearly reached equilibrium. We clearly see that spurious velocities gradually decay over time. When the steady state solution is reached, spurious currents have almost completely vanished.

Fig. 5.7 shows the pressure and density profiles computed across the interface, when reaching steady-state. As expected, the density variation is sharp over the interface and occurs over 4 grid cells. At this stage (Fig. 5.5(f)), the droplet is nearly circular, with a radius $R' = 68$ lattice nodes or $R = 5.3 \times 10^{-3}$ m, for which Laplace law predicts an expected pressure jump $\Delta p = 0.75\ \mathrm{N/m^2}$ ($\Delta p' = 3.179 \times 10^{-5}$), whereas on Fig. 5.7 we see a computed average pressure jump of about 0.745 N/m^2 (relative difference 0.67%).

5.4. Rising bubble in quiescent fluid

The dynamic behavior of a lighter fluid (ρ_2, ν_2) bubble rising in a heavier fluid (ρ_1, ν_1) under the buoyancy (gravitational) force is a standard test case extensively used for validating two-phase flow simulations. Although the LBM simulation setup in terms of grid initialization and boundary conditions is straightforward, the flow structure computed around the bubble is quite complex and governed by competing effects of viscosity, buoyancy, and surface tension forces. Several experimental studies have been conducted to measure the rise and deformation of single bubbles in a quiescent fluid [31,32], which indicate that the bubble shape greatly varies according to various flow regimes defined by values of non-dimensional parameters, such as the Bond number Bo (also known as Eotvos number, the ratio of gravity to surface tension forces), the Reynolds number Re, and the Morton number Mo, defined as,

$$\mathrm{Bo} = \frac{gD^2}{\sigma_{12}}(\rho_1 - \rho_2); \quad \mathrm{Mo} = \frac{g\mu_1^4}{\sigma_{12}^3 \rho_1}\left(1 - \frac{\rho_2}{\rho_1}\right); \quad \mathrm{Re} = \frac{UD}{\nu_1} \tag{5.7}$$

where U is the bubble terminal velocity and D its (equivalent) diameter, g is the gravitational acceleration, and ν_1 and μ_1 are the kinematic and dynamic viscosities of the heavier fluid, respectively. The terminal shapes of individual rising bubbles were experimentally observed for a range of Reynolds and Bond numbers [31], and can be generally regrouped into the following cap shape regimes: (a) spherical, (b) ellipsoidal, and (c) curved ellipsoidal. In the spherical regime, for small Bo, surface tension is dominant. The large surface tension force prevents the deformation of the bubble under inertia and viscous forces; consequently, the shape of the bubble remains (nearly) spherical during its rise. When increasing the Reynolds and Bond numbers, the contribution of surface tension gradually becomes less important as compared to inertia, and the terminal shape of the bubble becomes ellipsoidal for moderate Reynolds and Bond numbers ($10 < \mathrm{Re} < 500$ and $10 < \mathrm{Bo} < 100$), and spherical for high Reynolds and Bond numbers.

In the LBM simulations, a circular fluid bubble of density $\rho'_2 = 6$ and initial diameter $D'_0 = 60$ is located one bubble diameter above the bottom of a rectangular domain discretized with 256×1024 LBM cells, filled with a fluid of density $\rho'_1 = 6000$ (hence $\rho'_1/\rho'_2 = 1000$); the fluid viscosity ratio is $\mu'_1/\mu'_2 = 1000$. Both fluids are assumed to be stationary at initial time $t' = 0$ and we specify a periodic boundary condition on the lateral sides of the domain

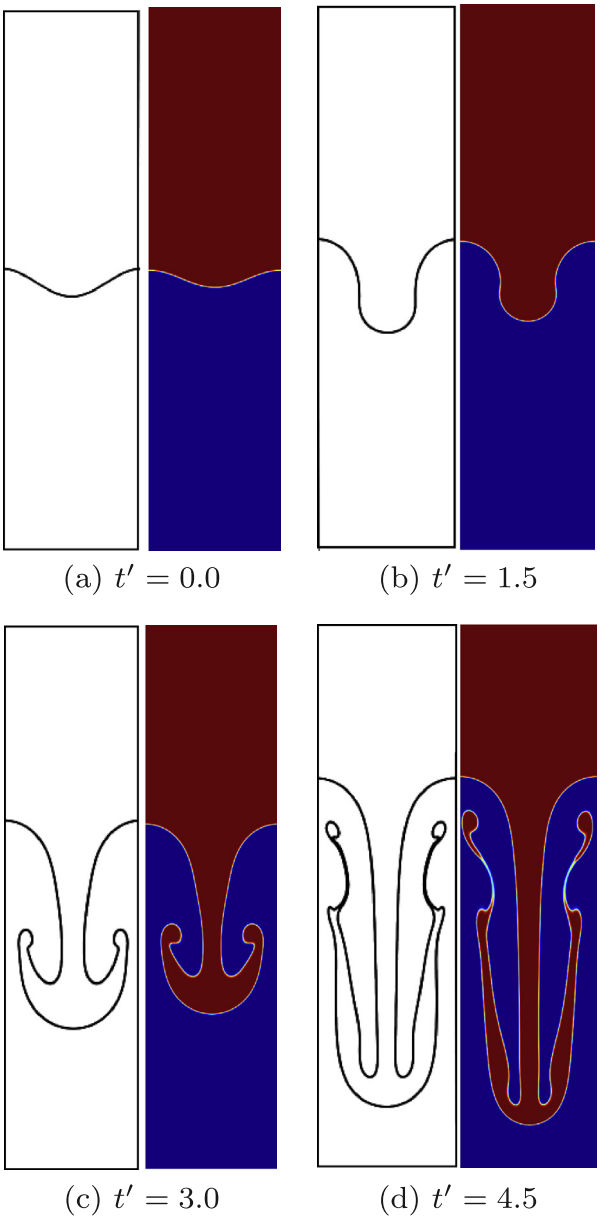


Fig. 5.10. Rayleigh–Taylor instability problem for $\rho_1/\rho_2 = 3, A = 0.5, \mathrm{Re} = 256$. Time evolution of the two-fluid interface for four dimensionless times $t' = t/\sqrt{L/g}$: (leftward panels) results of [8]; (rightward panels) present LB results.

and a bounce-back condition on the top and bottom boundaries. Simulations are run for 3 test cases (a–c) with different Mo and Bo values, given in Table 5.3 together with LBM and other flow parameters, corresponding to the three flow and bubble shape regimes discussed above.

Since we only solve for a two-dimensional (2D) flow, we cannot compare our LBM simulation results to experiments. However, we can validate results by comparing them to an independent 2D numerical solution, such as that of Sun and Tao [33], who used a hybrid volume-of-fluid and level set (VOSET) method to simulate incompressible two-phase flows. In Fig. 5.8, the terminal shapes of the bubbles and the velocity fields computed with the LBM for the 3 cases are compared to Sun and Tao's results. We see that both the predicted bubble shape and flow fields agree well with the reference solution.

Fig. 5.9 further shows the computed time evolution of the bubble shape during its rise, for the case of Fig. 5.8(c). During the early

stages of the simulation, buoyancy forces are dominant and the bubble starts rising; as it picks up speed, viscous drag gradually changes the bubble shape, bending it downstream. Eventually, the terminal shape of the bubble is formed when buoyancy, surface tension, and viscous forces are balanced.

5.5. Rayleigh–Taylor instability

The classical Rayleigh–Taylor instability at the interface between two immiscible fluids, with the heavier fluid being located above the lighter fluid, is used to demonstrate the accuracy of our LBM model to solve more complex two-phase flows, in which the interface becomes extremely deformed. Here, the heavier fluid will gradually sink into the lighter fluid, which is displaced upwards, under the influence of gravity. The dimensionless numbers that are important in this test case are the Atwood number and the Reynolds number, which are defined as,

$$A = \frac{\rho_1 - \rho_2}{\rho_1 + \rho_2} \quad \text{and} \quad \mathrm{Re} = \frac{\sqrt{LgL}}{\nu}, \tag{5.8}$$

where L is the width of the channel, and ρ_1 and ρ_2 are the densities of the heavy and light fluids, respectively. We set-up this simulation following [8] and specify no-slip boundary conditions along the top and bottom boundaries, and periodic boundary conditions on the lateral boundaries; gravity is chosen to satisfy $\sqrt{L'g'} = 0.04$. The kinematic viscosity is the same for both fluids $\nu = \nu_1 = \nu_2$, and the Atwood and Reynolds numbers are 0.5 and 256, respectively, with accordingly $\rho_1/\rho_2 = 3, \rho_2' = 6, k' = 0.07$ and $\beta' = 0.4$. LBM simulations are carried out in a grid with 256×1024 lattice nodes. Fig. 5.10 compares results of our method to the independent 2D numerical results of [8], for four selected time steps; the agreement between both methods is quite good.

5.6. Breaking wave

Previous work [34,35] shows that a periodic sinusoidal wave of large amplitude, with the initial velocities being calculated from linear wave theory, is not stable and rapidly breaks, since the initial velocity field is not in equilibrium with the initial wave profile, for the fully nonlinear flow equations. To limit computational time, as in this earlier work, the simulation is assumed to be 2D and periodic in the flow direction. This characteristic makes such periodic sinusoidal waves a convenient and efficient way of studying wave breaking [1].

According to linear wave theory, in axes (x, z), the initial wave velocity and interface shape are given by,

$$\begin{aligned} \eta &= \frac{H}{2}\cos(kx) \\ u &= \frac{H}{2}\omega\frac{\cosh k(h+z)}{\sinh(kh)}\cos(kx) \\ v &= \frac{H}{2}\omega\frac{\sinh k(h+z)}{\sinh(kh)}\sin(kx) \end{aligned} \tag{5.9}$$

where ω is the wave angular frequency, k the wavenumber, and other wave parameters are defined in Fig. 5.11.

Fig. 5.12 shows the time evolution computed with the LBM, of a high amplitude sinusoidal wave with $H/L = 0.13$ in depth $h/L = 0.25$. The number of grid nodes used in this simulation is 512×256. We see that the wave is not stable and rapidly overturns and breaks, after traveling for about one wavelength from initialization. Results obtained for the plunging breaker shape are qualitatively similar to those of earlier numerical solutions [34,35].

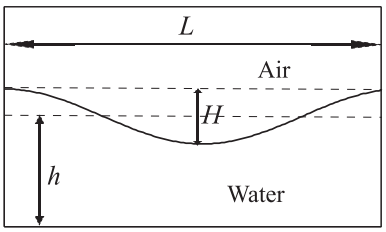


Fig. 5.11. Definition sketch for initial interface profile of a large amplitude sinusoidal wave.

6. Conclusions and outlook

In this paper, we reported on the development, efficient GPGPU implementation, and numerical validation, of an LBM model for the simulation of two-phase flows of fluids with large density ratios (up to at least $\rho_1/\rho_2 = 1000$), and possibly large viscosity ratios as well. In the model, we introduced three sets of LBM particle distribution functions to solve: (i) the "pressureless" Navier–Stokes equations in both fluids; (ii) the convection–diffusion Cahn–Hilliard equation for the two-fluid interface motion (including surface tension effects); and (iii) a (pressure) Poisson equation for correcting the pressureless velocity field. As a result of this homogeneous LBM formulation, this 2D scheme could be efficiently implemented in a GPGPU framework, owing to both the locality and simplicity of LB operators. This LBM scheme was applied to the simulation of various analytical or numerical benchmark problems, showing both good convergence towards reference results and efficiency in terms of computational time. This good agreement confirmed the prediction of the theoretical Chapman–Enskog expansions, of convergence of the scheme to the above-mentioned macroscopic equations.

While validating the LBM scheme for several benchmark problems, we found a linear convergence rate of the L_2-norm of numerical errors to the reference solution. This is consistent with the truncation errors of both the Chapman–Enskog expansions and time updating schemes corresponding to the various particle distribution functions. Additionally, some of the numerical errors result from the computation of the $-\mu(\partial_\alpha u_\beta + \partial_\beta u_\alpha)\partial_\beta(1/\rho)$ terms in Eq. (3.21), which are added to the standard LBM scheme (Eq. (3.9)) as an equivalent body force, to simulate the complete Navier–Stokes equations. Since these extra forcing terms are highly varying in both space and time near the two-fluid interface, unlike standard body forces such as gravity, additional terms appear in the Chapman–Enskog expansion for the momentum equation, that only linearly vanish with grid size. To obtain more accurate (and faster converging) numerical results, these terms should not be neglected, especially for high fluid density and viscosity ratios. Hence, higher-order LBM schemes for the body force terms should be used (e.g [36]). We are currently addressing this issue and developing such second-order schemes, which will be reported on in a future paper.

Note, a simple temporary solution to decrease this truncation error would have been to increase the interface thickness W, thus yielding smaller density gradients across the interface and hence a smaller total body force. While the nature of the real fluid interface is to be diffusive, however, this occurs over a very small length scale, that possibly is below LBM grid resolution, so that the interface thickness should be kept small. Additionally, the analytical reference solutions, such as for the two phase Poiseuille flow, are derived assuming a sharp phase interface. Hence, in order to compare our LBM results to theoretical solutions, we decided to limit the interface thickness to only 4 LBM grid cells.

A second significant model improvement that will be the object of future work, is the derivation of multiple relaxation time (MRT)

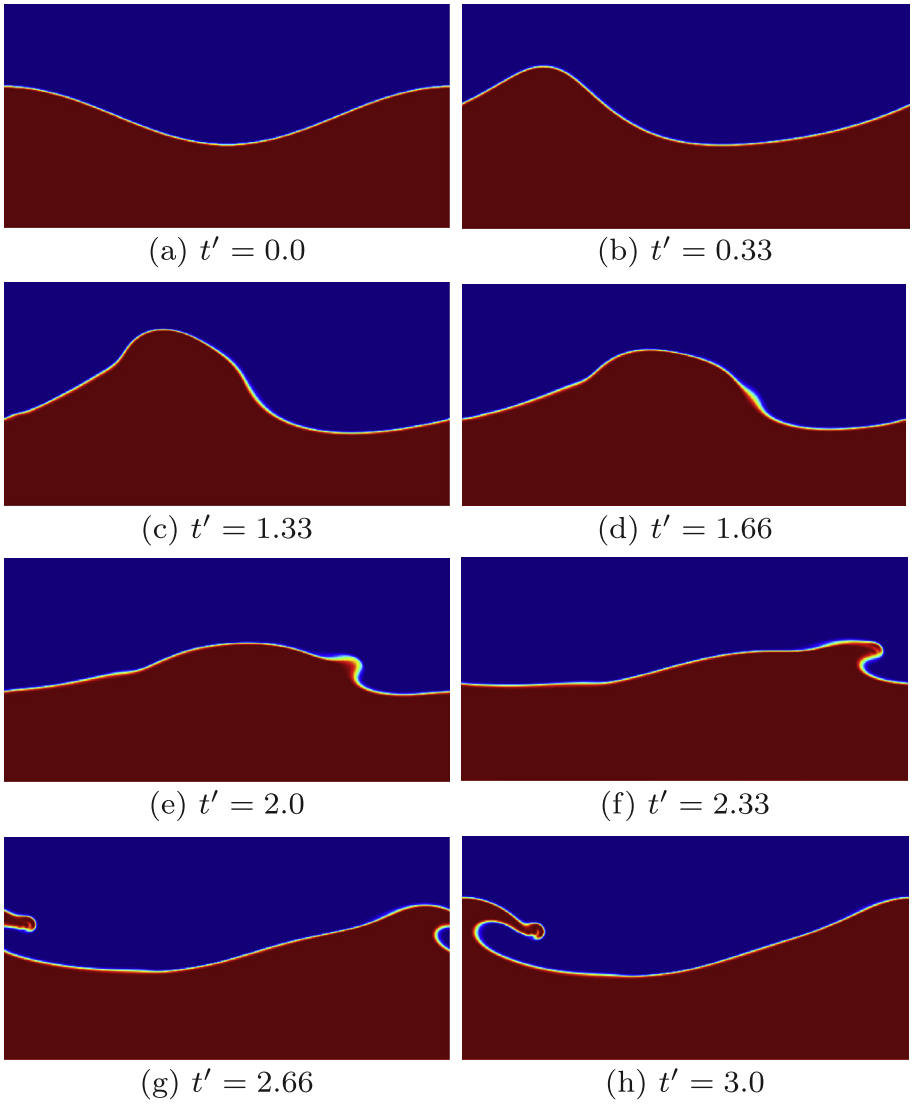


Fig. 5.12. Space-periodic wave breaking (Fig. 5.11). Time evolution of an overturning breaking wave with $H/L = 0.13$ and $h/L = 0.25$ (dimensionless time $t' = t/\sqrt{L/g}$).

collision operators. In these, the particle distribution functions are transformed to a well-defined moment space before doing the relaxation step. Several different relaxation rates can thus be used for the collision steps, unlike the single relaxation time used in the currently implemented SRT approach. This typically leads to a more efficient and stable numerical scheme, particularly for low fluid viscosities and high Reynolds numbers.

Finally, once the model has been further improved, such as by using the MRT, the GPGPU implementation will be extended to solving three-dimensional (3D) flows. As the iterations for solving the pressure Poisson equations may become much more computationally demanding in 3D, we are planning to explore using a multigrid algorithm to speed up this part of the simulations.

Acknowledgements

The authors gratefully acknowledge support for this research from the nVIDIA Academic Partnership Program (APP). In addition, MK acknowledges support from the Deutsche Forschungsgemeinschaft by funding RTG MUSIS.

The first three authors wish to acknowledge support from Grant OCE-09-27014 of the US National Sciences Foundation (NSF) Physical Oceanography Program.

References

[1] Lubin P, Vincent S, Abadie S, Caltagirone J-P. Three-dimensional large eddy simulation of air entrainment under plunging breaking waves. Coast Eng 2006;53:631–55.
[2] Janßen C, Grilli ST, Krafczyk M. On enhanced non-linear free surface flow simulations with a hybrid LBM–VOF approach. Comput Math Appl 2013;65(2):211–29.
[3] Jacqmin D. Calculation of two-phase Navier–Stokes flows using phase-field modeling. J Comput Phys 1999;155:96–127.
[4] Unverdi SO, Tryggvason G. A front-tracking method for viscous, incompressible, multi-fluid flows. J Comput Phys 1992;100:25–37.
[5] Brackbill J, Kothe DB, Zemach C. A continuum method for modeling surface tension. J Comput Phys 1992;100:335–54.
[6] Cahn J, Hilliard J. Free energy of a nonuniform system. J Chem Phys 1958;28:258–67.
[7] Swift M, Osborn W, Yeomans J. Lattice Boltzmann simulation of non-ideal fluids. Phys Rev Lett 1995;75:830–3.
[8] He X, Chen S, Zhang R. A lattice Boltzmann scheme for incompressible multiphase flow and its application in simulation of Rayleigh–Taylor instability. J Comput Phys 1999;152:642–63.
[9] Lee T, Lin C. A stable discretization of the lattice Boltzmann equation for simulation of incompressible two-phase flows at high density ratio. J Comput Phys 2005;206:16–47.
[10] Inamuro T, Ogata T, Tajima S, Konishi N. A lattice Boltzmann method for incompressible two-phase flows with large density differences. J Comput Phys 2004;198:628–44.
[11] Chapman S, Cowling T. The mathematical theory of nonuniform gases. Cambridge University Press; 1970.

[12] Tölke J, Krafczyk M. TeraFLOP computing on a desktop PC with GPUs for 3D CFD. Int J Comput Fluid Dyn 2008;22:443–56.
[13] Rothman D, Keller J. Immiscible cellular-automaton fluids. J Stat Phys 1988;52:1119–24.
[14] Shan X, Chen H. Lattice Boltzmann model for simulating flows with multiple phases and components. Phys Rev E 1993;47:1815–9.
[15] Inamuro T, Ogata T, Ogino F. Numerical simulation of bubble flows by the lattice Boltzmann method. Future Gener Comput Syst 2004;20:959–64.
[16] Tölke J, Krafczyk M. Implementation of a lattice Boltzmann kernel using the compute unified device architecture developed by nVIDIA. Comput Visual Sci 2008;1:29–39.
[17] Janßen C, Krafczyk M. Free surface flow simulations on GPGPUs using LBM. Comput Math Appl 2011;61(12):3549–63.
[18] Schonherr M, Kucher K, Geier M, Stiebler M, Freudiger S, Krafczyk M. Multi-thread implementations of the lattice Boltzmann method on non-uniform grids for CPUs and GPUs. Comput Math Appl 2011;61(12):3730–43.
[19] Qian Y, d'Humieres D, Lallemand P. Lattice BGK models for Navier Stokes equation. Europhys Lett 1992;17:479–84.
[20] Yu D, Mei R, Luo L, Shyy W. Viscous flow computations with the method of lattice Boltzmann equation. Progr Aerosp Sci 2003;39:329–67.
[21] Buick J, Greated C. Gravity in lattice Boltzmann models. Phys Rev E 2000;61:5307–20.
[22] He X, Luo LS. Theory of the lattice Boltzmann method: from the Boltzmann equation to the lattice Boltzmann equation. Phys Rev E 1997;56:6811–7.
[23] Swift M, Osborn W, Yeomans J. Lattice Boltzmann simulation of liquid–gas and binary fluid systems. Phys Rev E 1996;54:5041–52.
[24] Grunan D. Lattice methods for modeling hydrodynamics. PhD thesis, Colorado State University; 1993.
[25] nVIDIA. NVIDIA CUDA programming guide; 2010. <http://www.nvidia.com/object/cuda_develop.html>.
[26] nVIDIA. NVIDIA CUDA; 2011. <http://www.nvidia.com/cuda>.
[27] Lee T. Effects of incompressibility on the elimination of parasitic currents in lattice Boltzmann method for binary fluids. Comput Math 2009;58:987–94.
[28] South MJ, Hooper AP. Linear growth in two fluid plane Poiseuille flow. J Fluid Mech 1999;381:121–39.
[29] Gross M, Moradi N, Zikos G, Varnik F. Shear stress in nonideal fluid lattice Boltzmann simulations. Phys Rev E 2011;83:017701.
[30] Freudiger S. Entwicklung eines parallelen, adaptiven, komponentenbasierten Stroemungskerns fuer hierarchische Gitter auf Basis des Lattice-Boltzmann-Verfahrens. PhD thesis, Technischen Universitaet Carolo-Wilhelmina zu Braunschweig; 2009.
[31] Clift R, Grace J, Weber M. Bubbles drops and particles. Academic Press; 1978.
[32] Bhaga D, Weber M. Bubbles in viscous liquids: shapes, wakes and velocities. J Fluid Mech 1981;105:61–85.
[33] Sun D, Tao W. A coupled volume-of-fluid and level set (VOSET) method for computing incompressible two-phase flows. Int J Heat Mass Transf 2010;53:645–55.
[34] Longuet-Higgins MS, Cokelet ED. The deformation of steep surface waves on water. I. A numerical method of computation. Proc R Soc Lond A 1976;350:1–26.
[35] Grilli ST, Skourup J, Svendsen IA. An efficient boundary element method for nonlinear water waves. Eng Anal Bound Elem 1989;6:97–107.
[36] Guo Z, Zhen C, Shi B. Discrete lattice effects on the forcing term in the lattice Boltzmann method. Phys Rev A 2002;65:042102.