# Performance portability of lattice Boltzmann methods for two-phase flows with phase change

Werner Verdier, Pierre Kestener, Alain Cartalade

▶ **To cite this version:**

Werner Verdier, Pierre Kestener, Alain Cartalade. Performance portability of lattice Boltzmann methods for two-phase flows with phase change. Computer Methods in Applied Mechanics and Engineering, 2020, 370, pp.113266. 10.1016/j.cma.2020.113266 . cea-02895761

**HAL Id: cea-02895761**

**https://cea.hal.science/cea-02895761**

Submitted on 18 Jul 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

Version of Record: https://www.sciencedirect.com/science/article/pii/S0045782520304515
Manuscript_b8f889a407f4cd5bb3d4e16b869c10bb

# Performance portability of lattice Boltzmann methods for two-phase flows with phase change

WERNER VERDIER[a], PIERRE KESTENER[b], ALAIN CARTALADE[a,*]

[a]*DES, ISAS, DM2S, STMF, LMSF, CEA, Université de Paris-Saclay, F-91191, Gif-sur-Yvette, France.*
[b]*DRF – Maison de la Simulation, CEA, Université de Paris-Saclay, F-91191, Gif-sur-Yvette, France.*

**Abstract**

Numerical codes using the lattice Boltzmann methods (LBM) for simulating one- or two-phase flows are widely compiled and run on graphical process units. However, those computational units necessitate to re-write the program by using a low-level language which is suited to those architectures (e.g. `CUDA` for GPU NVIDIA® or `OpenCL`). In this paper we focus our effort on the performance portability of LBM i.e. the possibility of writing LB algorithms with a high-level of abstraction while remaining efficient on a wide range of architectures such as multicores x86, GPU NVIDIA®, ARM, and so on. For such a purpose, implementation of LBM is carried out by developing a unique code, `LBM_saclay` written in the C++ language, coupled with the `Kokkos` library for performance portability in the context of High Performance Computing. In this paper, the LBM is used to simulate a phase-field model for two-phase flow problems with phase change. The mathematical model is composed of the incompressible Navier-Stokes equations coupled with the conservative Allen-Cahn model. Initially developed in the literature for immiscible binary fluids, the model is extended here to simulate phase change occurring at the interface between liquid and gas. For that purpose, a heat equation is added with a source term involving the time derivative of the phase field. In the phase-field equation a source term is added to approximate the mass production rate at the interface. Several validations are carried out to check step-by-step the implementation of the full model. Finally, computational times are compared on CPU and GPU platforms for the physical problem of film boiling.

*Keywords:*
Lattice Boltzmann method, phase-field model, two-phase flows with phase change, performance portability, `Kokkos` library, `LBM_saclay`, conservative Allen-Cahn model.

## 1. Introduction

The Lattice Boltzmann Method (LBM) [1, 2] is a very attractive method to simulate problems involving fluid flows. Since more than ten years, numerical codes using that method are widely compiled and run on Graphical Process Units (GPU) [3–6]. The GPUs allow for a very high calculation throughput and they are particularly efficient for repetitive workloads with simple memory access patterns. These units were initially designed for image processing or graphics rendering, but LBM simulations can also benefit from their use, because the stages of streaming and collide are two simple (stencil-like) computational operations. Numerous works have demonstrated the efficiency of LBM on single GPU (e.g. [5]) and later on clusters of GPUs (e.g. [7, 8]). However, those computational units necessitate to re-write the code by using a low-level language which is suited to their specific architectures (e.g. `CUDA` for GPU NVIDIA® or `OpenCL`). In this paper we focus our effort on the performance portability of LBM i.e. the possibility of writing LBM algorithms with a high-level of abstraction, but by remaining efficient on a wide range of architectures such as multicores x86, GPU NVIDIA®, ARM, and so on.

The issue of performance portability has already been studied and implementation of numerical algorithms running on various architectures (GPU and so on) can be done by directive approaches (mostly `OpenMP` or `OpenACC`). Directive-based parallel programming solutions consist in decorating source code with comments that are interpreted by the compiler to derive the actual parallel code. They are useful when porting a legacy simulation code with a large number of lines, for which it is not reasonable to rewrite it from scratch. However, those programming models deal with computational patterns (`for` loops, reduction loops, ...) and do not provide tools for data or memory containers. Here we present an application of a more promising approach that uses a library-based solution which offers high-level abstract programming concepts and hardware agnostic solution for a better integration into C++ codes. Among libraries sharing the same goal of performance portability (like `RAJA` or `SYCL`), the `Kokkos` library [9] is used for simulating two-phase flows with LBM. `Kokkos` implements a programming model in C++ for writing performance portable applications targeting all major High Performance Computing (HPC) platforms. Programming tools provide abstractions for both parallel execution of code and

*Corresponding author. Tel.:+33 (0)1 69 08 40 67
*Email addresses:* `werner.verdier@cea.fr` (WERNER VERDIER), `pierre.kestener@cea.fr` (PIERRE KESTENER), `alain.cartalade@cea.fr` (ALAIN CARTALADE)



© 2020 published by Elsevier. This manuscript is made available under the CC BY NC user license
https://creativecommons.org/licenses/by-nc/4.0/

data management, i.e. they provide memory containers (multidimensional arrays) where the actual memory layout will be chosen by the library during compilation. Directive-based solution does not provide such advanced features regarding memory. The `Kokkos` library can currently use `OpenMP`, `Pthreads` and `CUDA` as backend programming models. The library has already been applied to accelerate high-order mesh optimization in [10].

Because of its explicit scheme and local interactions, the LBM ideally exploits the massively parallel supercomputers based on either CPUs or GPUs or heterogeneous architectures. In this paper, we take advantage of those benefits to study two-phase flows. Several topical reviews exist in the literature for modeling two-phase flows in LBM framework [11, 12]. The main families of methods are the color-gradient method [13], the pseudo-potential method [14, 15], the free-energy method [16], and the phase-field method [17]. Most of approaches consider the interface as a diffuse zone (characterized by a thickness and a surface tension) which can be seen as a small region of transition between bulk phases. In pseudo-potential methods [14, 15] an additional force term is added in the Navier-Stokes equations to take into account an equation of state which is not the classical law of perfect gases [18]. In that case, the density plays the role of a phase index varying smoothly between densities of gas and liquid. Several recent applications use that method for simulating liquid-gas phase change [19, 20]. Another class of diffuse interface methods is the color-gradient model [13] for which two distribution functions are introduced for computation of each phase (red and blue). In those approaches, surface tension is derived from a recoloring step involving both distribution functions [21, 22]. The final approach that is commonly applied in the LBM literature is to capture the interfacial behavior through a phase-field equation. In this paper, we follow this latter method: the phase-field theory for two-phase flows [23]. The phase-field method is quite similar to the free-energy lattice Boltzmann method [16] in the sense that both models are thermodynamically consistent and can be derived from a free-energy functional. However, in the free-energy LB approach, the density gradient appears explicitly in the free-energy functional and the phase separation is described by a non-ideal equation of state. For that purpose, the equilibrium distribution function is modified to include a non-ideal thermodynamic pressure tensor. In this paper, both fluids are considered as quasi-incompressible, i.e. we assume that the incompressibility condition holds in the bulk phases except in the interfacial zone where the mass production rate $\dot{m}'''$ acts. That mass production rate comes from the phase change between the gas and the liquid. A new function $\phi$ is introduced to track the interfacial zone where the density varies.

Two main phase-field models for interface tracking between two immiscible fluids exist in literature: the first one is the Cahn-Hilliard (CH) model [17, 24, 25] which was extensively applied in LBM literature for simulating spinodal decomposition [26], buoyancy of bubbles [27], drop impact [28], Rayleigh-Taylor instability [29] and so on. The second one is a more recent model, called the conservative Allen-Cahn (CAC) model, which was first developed in [30] and derived in conservative form in [31]. The model became popular in the LB community [32–34] and several papers compare the Cahn-Hilliard and conservative Allen-Cahn models, e.g. [35] without LBM and [36] with LBM. In this work the CAC model is chosen for interface tracking in order to eliminate the curvature-driven interface motion which is implicitly contained in the CH equation (see Section 2). Moreover, the CAC model involves only a second-order derivative and does not require to compute the fourth-order derivative (Laplacian of chemical potential) which appears in the CH equation.

In this paper, we take advantage of the simplicity of LBM to develop a new portable code for simulating two phase flows with the coupled Navier-Stokes/conservative Allen-Cahn (NS/CAC) model. The new code, called `LBM_saclay`, targets all major HPC platforms such as multi-GPUs and multi-CPUs. In this paper, we also check the capability of the NS/CAC model to simulate phase change problems in the vicinity of the critical temperature. Near the critical temperature, properties of each phase vary smoothly and the range of variation of those parameters remains small. Several fluid flow models of phase change have already been proposed in the literature with the Cahn-Hilliard equation [37, 38]. Following those references, the NS/CAC model is extended here by adding a source term in both the mass balance and the CAC equations. The source term involves the mass production rate $\dot{m}'''$ occurring at the interface. In references [37, 38], the liquid is often considered at saturation temperature and its thermal conductivity is neglected. Under those assumptions, $\dot{m}'''$ is calculated by a gradient operator (Fourier's law) involving only the thermal conductivity of gas. Moreover, in order to avoid computing the temperature equation in liquid phase (because the thermal conductivity is neglected), a cut-off value of the phase-field is introduced beyond which the temperature equation will not be computed [38]. Here we propose an alternative way to calculate $\dot{m}'''$ that avoids computing this gradient and avoids introducing this cut-off value. For that purpose, $\dot{m}'''$ will be related to the normal interface velocity and expressed as a source term close to what is done in solidification models (section 2.3). Implementation of lattice Boltzmann methods will be checked step-by-step by considering separately solutions of the phase-field equation, the phase-field coupled successively with a fluid flow, and the phase-field coupled only with temperature for which the ratio of physical properties remain low. Finally, the aspects of two-phase flow, phase change and heat transfer are coupled to simulate the phenomena of film boiling [39].

This paper is organized as follows. Section 2 presents the continuous mathematical model based on the conservative Allen-Cahn equation which is extended to handle phase change. The model derivation will be reminded, as well as definition of the chemical potential and interpolation methods for kinematics viscosities and densities. Section 3 presents the Lattice Boltzmann schemes based on the Bhatnagar-Gross-Krook (BGK) collision operator for each equation. That collision operator is chosen because of its simplicity of implementation. Several improvements exist such as the two-relaxation-times (TRT) and the multiple-relaxation-times (MRT). Their benefits will be quickly discussed in that section. Computation of gradient and Laplacian operators that are involved in equations of phase-field and fluid flow will also be specified. Details on numerical implementation with the `Kokkos` library and various optimizations of LBM kernel will be discussed in Section

3.5. In Section 4, several basic code verifications are presented to check the implementation of each equation step-by-step. In Section 5, two purely qualitative simulations will be presented on the two-dimensional test case of film boiling. The first one will illustrate the capability of the model to simulate the detachment of bubbles on nodes and antinodes. The second one will illustrate the influence of the Jacob number on their detachment and shape. Here, we give a comparison of the code performance running on two architectures (CPU Intel and GPU NVIDIA®). Finally, Section 6 and three appendices will conclude this paper.

## 2. Two-phase flow with mass transfer

### 2.1. Phase change model

A single component fluid is considered, which can be either in a liquid ($l$) or gas ($g$) phase. The system is then composed with two incompressible fluids with constant densities $\rho_l$ and $\rho_g$. A phase index $\phi \equiv \phi(\mathbf{x},t)$ is introduced which can vary between 0 and 1 with $\phi = 0$ (respectively $\phi = 1$) corresponding to fluid $l$ (resp. $g$) which is characterized by its density $\rho_l$ (resp. $\rho_g$) and its kinematic viscosity $\nu_l$ (resp. $\nu_g$). All other values of $\phi$ represent the interfacial zone or a mixture of both fluids $l$ and $g$. When $0 < \phi < 1$, the densities $\rho(\phi)$ and the kinematic viscosities $\nu(\phi)$ are respectively interpolated by

$$\rho(\phi) = \phi(\mathbf{x},t)\rho_g + (1-\phi(\mathbf{x},t))\rho_l, \tag{1a}$$

$$\nu(\phi) = \frac{\nu_l\nu_g}{\phi(\mathbf{x},t)\nu_l + (1-\phi(\mathbf{x},t))\nu_g}. \tag{1b}$$

Local densities depending on position and time are noted $\tilde{\rho}_\chi$ (for $\chi = g, l$) and write $\tilde{\rho}_g(\mathbf{x},t) = \rho_g\phi(\mathbf{x},t)$ and $\tilde{\rho}_l(\mathbf{x},t) = (1-\phi(\mathbf{x},t))\rho_l$. The total density writes $\rho(\mathbf{x},t) = \rho_g\phi(\mathbf{x},t) + (1-\phi(\mathbf{x},t))\rho_l$. The method of harmonic mean is used in this work to interpolate the viscosity (Eq. (1b)) for simulating flows with viscosity contrast ([29, Eq. (29c)]). A comparison of both interpolation methods (linear and harmonic mean) is presented on the double-Poiseuille flow in Section 4.1. The local velocity $\mathbf{u}_\chi$ of each component $\chi$ is related to the volume averaged velocity $\mathbf{u}$, the constant bulk density value $\rho_\chi$, and the volume diffusive flow rate $\mathbf{j}_\chi$ by [28] $\rho_\chi\mathbf{j}_\chi = \tilde{\rho}_\chi(\mathbf{u}_\chi - \mathbf{u})$ i.e. $\tilde{\rho}_\chi\mathbf{u}_\chi = \tilde{\rho}_\chi\mathbf{u} + \rho_\chi\mathbf{j}_\chi$. The mass balance equations for each phase $g$ and $l$ writes

$$\frac{\partial\tilde{\rho}_g}{\partial t} + \boldsymbol{\nabla}\cdot(\tilde{\rho}_g\mathbf{u} + \rho_g\mathbf{j}_g) = +\dot{m}''', \tag{2a}$$

$$\frac{\partial\tilde{\rho}_l}{\partial t} + \boldsymbol{\nabla}\cdot(\tilde{\rho}_l\mathbf{u} + \rho_l\mathbf{j}_l) = -\dot{m}''', \tag{2b}$$

where $\dot{m}'''$ is the volumic production term ($+$) or sink term ($-$) due to phase change. Its physical dimension is $\mathrm{M.L^{-3}.T^{-1}}$ and its computation will be discussed in Section 2.3. In Eqs. (2a) and (2b), signs are chosen such as the phase change produces gas phase $g$ to the detriment of liquid phase $l$. The mass flux relative to advection in each phase is $\tilde{\rho}_\chi\mathbf{u}$. In interfacial region, the mass flux $\rho_\chi\mathbf{j}_\chi$ has a diffusive origin and results of a regular transition of composition between two phases. By expressing Eqs. (2a) and (2b) with respect to $\phi(\mathbf{x},t)$ and assuming that the fluxes $\mathbf{j}_g$ and $\mathbf{j}_l$ are identical and opposite, $\mathbf{j} = \mathbf{j}_g = -\mathbf{j}_l$, the following equations are obtained:

$$\frac{\partial\phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi + \mathbf{j}) = +\frac{\dot{m}'''}{\rho_g}, \tag{3a}$$

$$\frac{\partial(1-\phi)}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}(1-\phi) - \mathbf{j}) = -\frac{\dot{m}'''}{\rho_l}, \tag{3b}$$

which after summing yield

$$\boldsymbol{\nabla}\cdot\mathbf{u} = \dot{m}'''\left(\frac{1}{\rho_g} - \frac{1}{\rho_l}\right). \tag{4}$$

To derive the interface tracking equation, in references [28, 38] the flux $\mathbf{j}$ is assumed to be given by the Cahn-Hilliard flux defined by $\mathbf{j} = -M_\phi\boldsymbol{\nabla}\mu_\phi$ where $\mu_\phi$ is the chemical potential. In that case Eq. (3a) becomes the CH equation with a source term of production in the second member. The Navier-Stokes/Cahn-Hilliard (NS/CH) model is very popular for simulations of two-phase flow since more than twenty years (e.g. without LBM [17, 25] and [26–29] with LBM). However the chemical potential can be interpreted as the product of surface tension $\sigma$ and curvature $\kappa$ (see details in Section 2.2), and the CH equation imposes in its formulation a motion due to $\sigma$ and $\kappa$ even without coupling with a fluid flow. Here, in order to eliminate the curvature-driven interface motion inside the phase-field equation, we assume that the flux is defined by [30, 31] $\mathbf{j} = -M_\phi(\boldsymbol{\nabla}\phi - 4\phi(1-\phi)\mathbf{n}/W)$ and Eq. (3a) becomes the conservative Allen-Cahn (CAC) model with a source term:

$$\frac{\partial \phi}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{u}\phi) = \boldsymbol{\nabla} \cdot \left[ M_\phi \left( \boldsymbol{\nabla}\phi - \frac{4}{W}\phi(1-\phi)\mathbf{n} \right) \right] + \frac{\dot{m}'''}{\rho_g}. \tag{5}$$

In Eq. (5), $M_\phi$ is the interface mobility, $W$ is the diffuse interface width and

$$\mathbf{n} = \frac{\boldsymbol{\nabla}\phi}{|\boldsymbol{\nabla}\phi|} \tag{6}$$

is the unit normal vector at the interface directed from liquid toward gas. Eq. (5) is the Conservative version of Allen-Cahn (CAC) equation with a source term for modeling interface tracking with phase change. The accuracy of the phase-field simulations depends on two parameters: the interface thickness $W$ and the mobility $M_\phi$. In reference [17, Sec. 5], a discussion is given regarding the numerical convergence of the phase-field method and the choice of those parameters in relation to the discretization step $\delta x$. For the Cahn-Hilliard equation, the mobility affects the thickness and perturbation magnitude of the chemical potential boundary layers. Here, for simulations of film boiling, preliminary sensitivity tests are performed on $M_\phi$ and some details of its effects will be given in Section 5. The choice of $\dot{m}'''$ will be discussed in Section 2.3. In the original paper [30], this equation is derived by assuming that the total advection velocity is composed of two terms: the external advective velocity $\mathbf{u}$, plus the normal velocity to the interface $u_n\mathbf{n}$. That velocity $u_n$ is also defined as the sum of one term depending on the curvature $\kappa$, plus one independent of $\kappa$: $u_n\mathbf{n} = (\tilde{v} - M_\phi\kappa)\mathbf{n}$. In the right-hand side of Eq. (5), the first term $\boldsymbol{\nabla}\cdot\mathbf{j}$ is an equivalent expression to the curvature term that is corrected with a "counter term" $-M_\phi\kappa|\boldsymbol{\nabla}\phi|$ [40], in order to cancel the curvature-driven interface motion. The derivation is reminded in AppendixA by using the usual definition of curvature $\kappa = \boldsymbol{\nabla}\cdot\mathbf{n}$ with $\mathbf{n}$ defined by Eq. (6), and introducing the kernel function

$$\phi = \frac{1}{2}\left[1 + \tanh\left(\frac{2\zeta}{W}\right)\right] \tag{7}$$

in order to give an expression of $|\boldsymbol{\nabla}\phi|$ (see Eq. (A.7) in AppendixA):

$$|\boldsymbol{\nabla}\phi| = \frac{4}{W}\phi(1-\phi). \tag{8}$$

That choice of kernel function imposes bulk phases for $\phi = 0$ and $\phi = 1$. Similar reasoning that cancels the curvature term can be found in [41] in order to eliminate effects of surface tension (inherent in phase-field models) for membranes embedded in a Newtonian fluid. Let us notice that in this work the standard convention $0 \leq \phi \leq 1$ is used. Other conventions are possible, particularly when studying two-phase flow with high density ratio e.g. $-\phi^\star \leq \phi \leq \phi^\star$ where $\phi^\star$ is defined by $\rho_g$ and $\rho_l$ (e.g. [27, Eq. (31)]). More generally, the inequality $\phi_l \leq \phi \leq \phi_g$ can be chosen. In that case the kernel function (Eq. (7)) and the expression of $|\boldsymbol{\nabla}\phi|$ must change. Moreover the source term in Eq. (5) must be modified by (see [12, Eq. (188)]): $\dot{m}'''(\phi_g/\rho_g - \phi_l/\rho_l)$. Here, that expression is simplified to $\dot{m}'''/\rho_g$ with the standard choice $\phi_g = 1$ and $\phi_l = 0$.

The temperature equation is derived from the conservation law of total enthalpy $\rho\mathcal{H}$ where $\mathcal{H}$ is the enthalpy (physical dimension $\mathrm{E.M^{-1}}$ where E is used for Energy) as carried out in crystal growth simulations [42]:

$$\frac{\partial(\rho\mathcal{H})}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\rho\mathcal{H}) = \boldsymbol{\nabla}\cdot(\mathcal{K}\boldsymbol{\nabla}T) \tag{9}$$

where the diffusive flux is given by the Fourier's law $\mathbf{j}_T = -\mathcal{K}\boldsymbol{\nabla}T$ with $T$ being the temperature and $\mathcal{K}$ the thermal conductivity (physical dimension $\mathrm{E.T^{-1}.L^{-1}.\Theta^{-1}}$). The enthalpy is defined by $\mathcal{H} = \mathcal{C}_p T + \phi\mathcal{L}$ where $\mathcal{C}_p$ is the specific heat ($\mathrm{E.M^{-1}.\Theta^{-1}}$) and $\mathcal{L}$ is the latent heat of phase change ($\mathrm{E.M^{-1}}$). With this relation, enthalpies of liquid and gas are respectively equal to $\mathcal{H}_l = \mathcal{C}_p T$ for $\phi = 0$ and $\mathcal{H}_g = \mathcal{C}_p T + \mathcal{L}$ for $\phi = 1$. With those notations and definitions the heat equation for temperature writes

$$\frac{\partial T}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}T) = \alpha\boldsymbol{\nabla}^2 T - \frac{\mathcal{L}}{\mathcal{C}_p}\left[\frac{\partial\phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi)\right], \tag{10}$$

where $\alpha = \mathcal{K}/(\rho\mathcal{C}_p)$ is the thermal diffusivity, the second term in the right-hand side of Eq. (10) is interpreted as the release (or production) of latent heat during the displacement of the interface. When $\mathbf{u} = \mathbf{0}$ the movement of the interface is only due to phase change between liquid and gas. Solving only Eq. (5) and (10) must be equivalent to solve the Stefan problem of phase change (see validation of Section 4).

Finally, the complete model of two-phase flows with phase change writes:

$$\boldsymbol{\nabla}\cdot\mathbf{u} = \dot{m}'''\left(\frac{1}{\rho_g}-\frac{1}{\rho_l}\right), \tag{11a}$$

$$\left[\frac{\partial(\rho\mathbf{u})}{\partial t}+\boldsymbol{\nabla}\cdot(\rho\mathbf{u}\mathbf{u})\right] = -\boldsymbol{\nabla}p+\boldsymbol{\nabla}\cdot\left[\eta\left(\boldsymbol{\nabla}\mathbf{u}+\boldsymbol{\nabla}\mathbf{u}^T\right)\right]+\mathbf{F}_{tot}, \tag{11b}$$

$$\frac{\partial\phi}{\partial t}+\boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = \boldsymbol{\nabla}\cdot\left[M_\phi\left(\boldsymbol{\nabla}\phi-\frac{4}{W}\phi(1-\phi)\mathbf{n}\right)\right]+\frac{\dot{m}'''}{\rho_g}, \tag{11c}$$

$$\frac{\partial T}{\partial t}+\boldsymbol{\nabla}\cdot(\mathbf{u}T) = \alpha\boldsymbol{\nabla}^2T-\frac{\mathcal{L}}{C_p}\left[\frac{\partial\phi}{\partial t}+\boldsymbol{\nabla}\cdot(\mathbf{u}\phi)\right]. \tag{11d}$$

Eqs. (11a) and (11b) are the Navier-Stokes equations for modeling two Newtonian and incompressible fluids. In those equations $p$ is the pressure, $\rho(\phi)$ is the density depending on the phase-field $\phi$ and $\eta(\phi)$ is the dynamic viscosity. $\mathbf{F}_{tot}$ is the total force term defined as:

$$\mathbf{F}_{tot} = \mathbf{F}_s+\mathbf{F}_v \tag{12}$$

where $\mathbf{F}_s$ is the surface tension force that is defined in the next subsection. The volumic force $\mathbf{F}_v$ is the buoyancy force. Among different formulations of that force [43, Sec. 3.7], in this work the buoyancy is defined such as $\mathbf{F}_v = (\rho_l-\rho(\phi))\mathbf{g}$ where $\mathbf{g}$ is the constant acceleration due to the gravity. With that formulation, the gravity acts only on the gas phase for simulations of film boiling in Section 5.

### 2.2. *Chemical potential and Cahn-Hilliard equation*

The surface tension force $\mathbf{F}_s$ is expressed here in its potential form [17]:

$$\mathbf{F}_s = \mu_\phi\boldsymbol{\nabla}\phi \tag{13}$$

where $\mu_\phi$ is the chemical potential which is defined as the change of free energy for a small variation of local composition of mixture: $\mu_\phi = \delta\mathscr{F}/\delta\phi$. When the free energy is defined such as $\mathscr{F}(\phi) = \int_v[\mathscr{V}(\phi)+K|\boldsymbol{\nabla}\phi|^2/2]dv$ with $\mathscr{V}(\phi) = H\phi^2(1-\phi)^2$, the chemical potential writes

$$\mu_\phi = 4H\phi(\phi-1)\left(\phi-\frac{1}{2}\right)-K\boldsymbol{\nabla}^2\phi. \tag{14}$$

The first term of the right-hand side of Eq. (14) is the derivative of $\mathscr{V}(\phi)$ with respect to $\phi$ and the second term comes from the gradient energy term. The double-well ensures minima at $\phi=0$ and $\phi=1$. Coefficient $H$ is the height of double-well and $K$ is the gradient energy coefficient. It is well-known that the one-dimensional solution at equilibrium (i.e. $\mu_\phi=0$) of Eq. (14) is the hyperbolic tangent function defined by Eq. (7). A dimensional analysis of $\mathscr{F}(\phi)$ indicates that $H$ has the dimension of energy per volume unit, whereas $K$ has the dimension of energy per length unit. In this formalism, the surface tension $\sigma$ and the diffuse interface width $W$ are proportional to the product and the ratio of both coefficients:

$$\sigma = \frac{1}{6}\sqrt{2KH} \quad \text{and} \quad W = \sqrt{\frac{8K}{H}} \tag{15a}$$

We also note that $\sqrt{KH}$ is homogeneous to an energy per surface unit which corresponds to the physical dimension of surface tension. The term $\sqrt{K/H}$ is homogeneous to a length as expected for the interface thickness. For the simulations of section 4, values of $\sigma$ and $W$ will be set and $K$ and $H$ will be derived by inverting those two relationships:

$$K = \frac{3}{2}W\sigma \quad \text{and} \quad H = 12\frac{\sigma}{W}. \tag{15b}$$

Let us notice that, if we use Eqs. (14) and (15b), the surface tension force $\mathbf{F}_s = \mu_\phi\boldsymbol{\nabla}\phi$ can be written as $\mu_\phi\boldsymbol{\nabla}\phi = -(3/2)W\sigma\left[\boldsymbol{\nabla}^2\phi-16\phi(1-\phi)(1-2\phi)/W^2\right]\boldsymbol{\nabla}\phi$. The term inside the brackets is the curvature term $\kappa|\boldsymbol{\nabla}\phi|$ provided that the kernel function Eq. (7) is used for the second term (see Eq. (A.9) in AppendixA). In that case, the surface tension $\sigma$ and the curvature $\kappa$ appear explicitly in the definition of the chemical potential $\mu_\phi$ and the surface tension force is $\mathbf{F}_s = \mu_\phi\boldsymbol{\nabla}\phi = -(3/2)W\sigma\kappa|\boldsymbol{\nabla}\phi|\boldsymbol{\nabla}\phi$. Besides, if we set $K=\varepsilon^2$ and $H=1/4$ in Eq. (15a), then we find $(3/2)W = 6\sqrt{2}\varepsilon$. The surface tension force is $\mathbf{F}_s = -\sigma(6\sqrt{2}\varepsilon)(\boldsymbol{\nabla}\cdot\mathbf{n})|\boldsymbol{\nabla}\phi|\boldsymbol{\nabla}\phi$ which is the same relation in [44, Eq. (13)] provided that the kernel function Eq. (7) is applied for $\kappa$. As mentioned earlier, when the diffusive flux is proportional to the gradient of the chemical potential, then the evolution of $\phi$ follows the Cahn-Hilliard equation:

$$\frac{\partial\phi}{\partial t}+\boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = \boldsymbol{\nabla}\cdot(M_\phi\boldsymbol{\nabla}\mu_\phi), \tag{16}$$

with $\mu_\phi$ defined by Eq. (14). Compared to the standard CH equation, the main advantage of the conservative Allen-Cahn model lies in the computation of the right-hand side term. Indeed, the CH equation involves a fourth-order derivative because the flux is assumed to be proportional to gradient of chemical potential. A first Laplacian appears in Eq. (14) and a second one appears in the conservative equation Eq. (16). In the conservative Allen-Cahn equation (Eq. (11c)), only the second-order derivative is involved in its definition.

### 2.3. Production rate $\dot{m}'''$

#### 2.3.1. Interface velocity of phase change

In sharp interface methods, the surface production rate $\dot{m}''$ (physical dimension $\mathrm{M.L^{-2}.T^{-1}}$) occurs on the separation area between liquid and gas. It is usually defined by [45, 46] $\dot{m}'' = \rho_g(\mathbf{u}_g - \mathbf{V}_I)\cdot\mathbf{n} = \rho_l(\mathbf{u}_l - \mathbf{V}_I)\cdot\mathbf{n}$ where $\mathbf{V}_I$ is the velocity of the interface, and $\mathbf{u}_l$ and $\mathbf{u}_g$ are respectively the velocities on liquid and gas sides. This relation is derived by integrating the mass conservation across the interface. Integration of the energy conservation yields an additional relation on $\dot{m}''$ which can be calculated in its simplest form by the difference of heat fluxes, $\dot{m}'' = (\mathcal{K}\boldsymbol{\nabla}T|_l - \mathcal{K}\boldsymbol{\nabla}T|_g)\cdot\mathbf{n}/\mathcal{L}$. The driving force of evaporation is the heat quantity which is transferred at the interface. In [38], the liquid is assumed to be at saturation temperature $T_{sat}$ and in that case, only the heat quantity of the gas is considered and the temperature equation is solved only in the gas phase. Because of the diffuse interface, the rate $\dot{m}''$ is transformed to a volumic quantity $\dot{m}'''$ by $\dot{m}''' = \dot{m}''|\boldsymbol{\nabla}\phi| = \mathcal{K}\boldsymbol{\nabla}T\cdot\boldsymbol{\nabla}\phi/\mathcal{L}$ where $\phi$ follows the Cahn-Hilliard equation. The model was extended in [47] to include the gradient of the vapor concentration at the liquid-vapor interface as the driving force for vaporization. The model [38] was also applied in [48] to simulate nucleate pool boiling, including the bubble growth on and periodic departure from a superheated wall. Several other popular mass transfer models are reviewed in [39, Section 4.2] for phase change simulations.

Here, we notice that the source term $\dot{m}'''/\rho_g$ in Eq. (11c) can be identified as the normal velocity of the interface $-\tilde{v}|\boldsymbol{\nabla}\phi|$ (see Eq. (A.8) in AppendixA) i.e. $\dot{m}'''/\rho_g = -\tilde{v}$ (because $\dot{m}''' = \dot{m}''|\boldsymbol{\nabla}\phi|$). In Eq. (11c), the total velocity is the sum of an external velocity $\mathbf{u}$ plus the interface normal velocity. The latter has also been separated into one velocity depending on the curvature $-M_\phi\kappa$ (which has been canceled) plus one velocity $\tilde{v}$ independent of the curvature. That velocity is responsible for the displacement of the interface because of the phase change. Its expression can be approximated by [30, Eq. (A.5)]:

$$\tilde{v} = \frac{\alpha}{\mathscr{A}}\frac{\theta_I - \theta}{W}, \tag{17}$$

where $\theta$ is the dimensionless temperature defined as $\theta = (\mathcal{C}_p/\mathcal{L})(T - T_{sat})$, $\theta_I$ is the dimensionless interface temperature and $\mathscr{A}$ is a constant of proportionality that will be specified in section 2.3.2. Finally, if the kernel function $|\boldsymbol{\nabla}\phi| = (4/W)\phi(1-\phi)$ is used (see Eq. (8)), the source term $\dot{m}'''/\rho_g$ in Eq. (11c) takes the form

$$\frac{\dot{m}'''}{\rho_g} = -\tilde{v}|\boldsymbol{\nabla}\phi| = -\frac{4\alpha}{\mathscr{A}W^2}(\theta_I - \theta)\phi(1-\phi). \tag{18}$$

#### 2.3.2. Value of coefficient $\mathscr{A}$

In order to derive the value of $\mathscr{A}$ in Eq. (18), we proceed by analogy with the model of phase change for solidification and crystallization [49]. First, Eq. (11c) with Eq. (18) are re-written in order to make appear the derivatives of the double-well potential $f(\phi)$ and the interpolation function $p(\phi)$. Those functions are used in the solidification models derived from variational formulation based on the minimization of free energy [49]. The interface is tracked by Eq. (11c) by assuming that the movement due to curvature is cancelled. That equation can be re-written (see AppendixA):

$$\frac{\partial\phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = M_\phi\left[\boldsymbol{\nabla}^2\phi - \frac{\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}|\boldsymbol{\nabla}\phi|}{|\boldsymbol{\nabla}\phi|}\right] - M_\phi\kappa|\boldsymbol{\nabla}\phi| - \frac{4\alpha}{\mathscr{A}W^2}(\theta_I - \theta)\phi(1-\phi). \tag{19}$$

If the interface temperature is considered at saturation (i.e. $\theta_I = 0$), the source term is simplified to $(4\alpha/\mathscr{A}W^2)\theta\phi(1-\phi)$. With the kernel function Eq. (7), the second term in the brackets writes (see Eq. (A.9)) $\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}|\boldsymbol{\nabla}\phi|/|\boldsymbol{\nabla}\phi| = (16/W^2)\phi(1-\phi)(1-2\phi)$. That term is proportional to the derivative (with respect to $\phi$) of a double-well potential defined by $f(\phi) = H\phi^2(1-\phi)^2$ with $H = 1$, hence $\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}|\boldsymbol{\nabla}\phi|/|\boldsymbol{\nabla}\phi| = (8/W^2)\partial f/\partial\phi$. Besides if we set $K \equiv \varepsilon^2$, then the two relationships Eqs. (15b) with $H = 1$ yields $\varepsilon^2 = W^2/8$. We also set $M_\phi = \varepsilon^2/\mathcal{T}$ where $\mathcal{T}$ is the kinetic time, then Eq. (19) becomes

$$\mathcal{T}\left[\frac{\partial\phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi)\right] = \varepsilon^2\boldsymbol{\nabla}^2\phi - \frac{\partial f}{\partial\phi} - \varepsilon^2\kappa|\boldsymbol{\nabla}\phi| - \frac{4\mathcal{T}\alpha}{\mathscr{A}W^2}(\theta_I - \theta)\frac{\partial p}{\partial\phi}. \tag{20}$$

In the right-hand side of Eq. (20), the second term is the derivative of the double-well and the third term is the counter term. The last term is the coupling with temperature which involves the derivative (with respect to $\phi$) of an interpolation function defined as $p(\phi) = \phi^2/2 - \phi^3/3$. The factor 4 comes from the choice $a = 1/2$ in the kernel function (Eq. (A.6)) and we set $W_0 = W/2$. If we compare the coupling term of reference [49] with the last term of Eq. (20), we can identify

$$\lambda^\star = \frac{\mathcal{T}\alpha}{\mathscr{A}W_0^2}, \tag{21}$$

where $W_0^2 = W^2/4$ and $\lambda$ is the coupling coefficient in solidification/crystallization phase-field models. The star of $\lambda^\star$ means it is the particular value of $\lambda$ that cancels the kinetic coefficient in the Gibbs-Thomson condition recovered by the matched asymptotic analysis of the phase-field model. Hence, that coupling term (Eq. (21)) means this is the particular model of phase change which cancels the kinetic coefficient in the Gibbs-Thomson equation. Besides, the curvature term is also removed by the counter term $-\varepsilon^2\kappa|\boldsymbol{\nabla}\phi|$. Finally, the coefficient $\mathscr{A}$ is identified to the coefficient $a_2$ in reference [49]. Its value is $a_2 = 0.6267$ when the phase-field varies between $-1 \leq \phi \leq +1$ and when the derivative of the interpolating function of temperature is $p_\phi(\phi) = 1-\phi^2$ (the index $\phi$ indicates the derivative with respect to $\phi$). In the present paper, the phase-field $\phi$ varies between 0 and 1 and the derivative of the polynomial function is $p_\phi = \phi(1-\phi)$. Because of those differences, the value of $\mathscr{A}$ must be computed from integrals obtained from the matched asymptotic expansion of the phase-field model. In AppendixB, details are given to obtain $\mathscr{A} = 10/48 \approx 0.21$, value that will be used for all simulations of this paper.

## 3. Lattice Boltzmann schemes

In this Section, we detail the lattice Boltzmann methods that are used to simulate the phase change model of Section 2 composed of Eqs (11a)–(11d) with Eq (13) for surface tension force and Eq. (18) for mass production rate. Simulations are performed by using three distribution functions $\vartheta_i(\mathbf{x},t) \equiv \vartheta_i$ for $\vartheta = f, h, s$ where $i = 0, ..., N_{pop}$ and $N_{pop}$ is the total number of moving directions $\mathbf{e}_i$ on a lattice (defined below). The first distribution function $f_i$ is used to recover the Navier-Stokes model (subsection 3.1); the second one $g_i$ is used for the phase-field equation (subsection 3.2) and the last one $s_i$ is used for the temperature equation (subsection 3.3). Each distribution function follows its own discrete lattice Boltzmann equation in which the collision term is considered with the Bhatnagar-Gross-Krook (BGK) approximation. That collision operator uses a unique relaxation parameter that is related to the diffusive parameter of the PDE (kinematic viscosity, mobility or diffusion coefficient). Several improvements exist such as the TRT [50] or MRT [51, 52] collision operators. They both use additional relaxation parameters (only one for TRT). With MRT, some of them can be related to physical parameters (e.g. anisotropic diffusion coefficient for transport equation) and the other ones control the stability of the algorithm when increasing the Reynolds number or Péclet number. Hence a wider range of parameters can be reached when simulations are performed with TRT and MRT. Let us mention that other alternatives exist in the literature (entropic, central moments, cumulants, ...) but an in-depth discussion of their benefits and drawbacks is out of the scope of this work. In Eq. (25), each discrete Boltzmann equation is expressed in terms of new variables $\overline{f}_i$, $\overline{g}_i$ and $\overline{s}_i$, each one of them being defined by an appropriate variable change [53] (see details in AppendixC):

$$\overline{\vartheta}_i = \vartheta_i + \frac{\delta t}{2\tau_\vartheta}\left(\vartheta_i - \vartheta_i^{eq}\right) - \frac{\delta t}{2}\mathcal{S}_i^\vartheta \quad \text{for } \vartheta = f, h, s, \tag{22}$$

where $\tau_\vartheta$ and $\mathcal{S}_i^\vartheta$ are respectively the collision time and the source term relative to the distribution function $\vartheta$; $\delta t$ is the time step and $\vartheta_i^{eq}$ is the equilibrium distribution function. Two other notations are introduced: $\overline{\tau}_\vartheta$ and $\vartheta_i^\star$. The first one is the dimensionless collision rate that is defined by $\overline{\tau}_\vartheta = \tau_\vartheta/\delta t$ for each $\vartheta$. The second one is the distribution function that is obtained after the stages of collision and streaming: $\vartheta_i^\star \equiv \vartheta_i(\mathbf{x}+\mathbf{c}_i\delta t, t+\delta t)$. The use of this variable change (Eq. (22)) modifies the calculation of the zeroth-order moment $\mathcal{M}_0^\vartheta$ of the distribution function $\overline{\vartheta}_i$ by (see AppendixC)

$$\mathcal{M}_0^\vartheta = \sum_i \overline{\vartheta}_i + \frac{\delta t}{2}\mathcal{S}_i^\vartheta \quad \text{for } \vartheta = f, h, s. \tag{23}$$

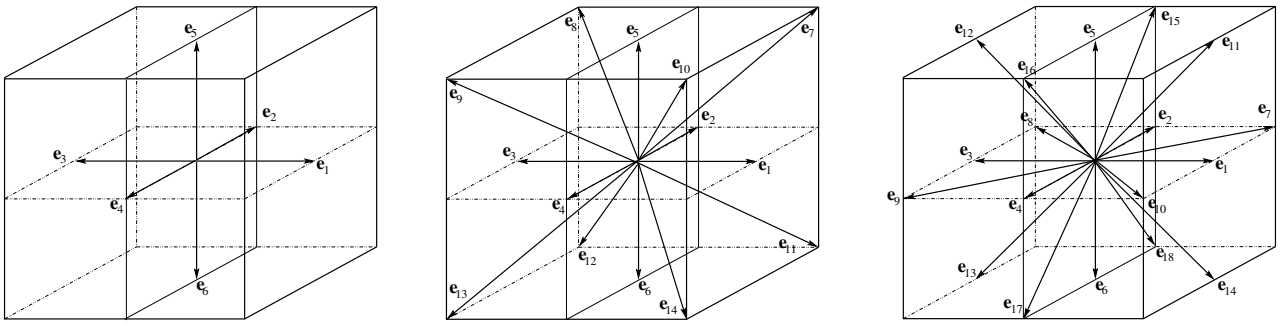
It is also useful to introduce the variable change for the equilibrium function (see AppendixC.1)

$$\overline{\vartheta}_i^{eq} = \vartheta_i^{eq} - \frac{\delta t}{2}\mathcal{S}_i^\vartheta \quad \text{for } \vartheta = f, h, s, \tag{24}$$

so that, with all those notations, the lattice Boltzmann equation writes

$$\overline{\vartheta}_i^\star = \overline{\vartheta}_i - \frac{1}{\overline{\tau}_\vartheta + 1/2}\left[\overline{\vartheta}_i - \overline{\vartheta}_i^{eq}\right] + \mathcal{S}_i^\vartheta \delta t \tag{25}$$

for each distribution function $\vartheta = f, h, s$. Before defining the equilibrium distribution functions and source terms, several lattices are introduced. In this work, the D2Q9 lattice and three 3D lattices are used: D3Q7, D3Q15 and D3Q19 (Fig. 1). For D2Q9 the moving vectors are defined by $\mathbf{e}_0 = (0,0)$, $\mathbf{e}_{1,3} = (\pm 1, 0)$, $\mathbf{e}_{2,4} = (0, \pm 1)$, $\mathbf{e}_{5,6} = (\pm 1, 1)$ and $\mathbf{e}_{7,8} = (\mp 1, -1)$. for 3D lattices, the moving vectors $\mathbf{e}_i$ are defined such as $\mathbf{e}_1 = (1,0,0)^T$, $\mathbf{e}_2 = (0,1,0)^T$, ..., $\mathbf{e}_6 = (0,0,-1)^T$ for D3Q7 (Fig. 1a). For D3Q15, additional diagonal vectors are defined such as (see Fig. 1b) $\mathbf{e}_7 = (1,1,1)^T$, $\mathbf{e}_8 = (-1,1,1)^T$, ..., $\mathbf{e}_{14} = (1,-1,-1)^T$. Finally for D3Q19 (Fig. 1c): $\mathbf{e}_{7,8} = (\pm 1, 1, 0)^T$, $\mathbf{e}_{9,10} = (\pm 1, -1, -0)^T$, $\mathbf{e}_{11,12} = (\pm 1, 0, 1)$, $\mathbf{e}_{13,14} = (\pm 1, 0, -1)^T$, $\mathbf{e}_{15,16} = (0, \pm 1, 1)^T$, $\mathbf{e}_{17,18} = (0, \pm 1, -1)^T$. For D3Q7 $N_{pop} = 6$, $e^2 = 1/4$, $w_0 = 1/4$ and $w_{1,...,6} = 1/8$. For D3Q15 $N_{pop} = 14$, $e^2 = 1/3$, $w_0 = 2/9$, $w_{1,...,6} = 1/9$ and $w_{7,...,14} = 1/72$. For D3Q19 $N_{pop} = 18$, $e^2 = 1/3$, $w_0 = 1/3$, $w_{1,...,6} = 1/18$ and $w_{7,...,18} = 1/36$. The standard notations will be used: $\mathbf{c}_i = \mathbf{e}_i c$ with $c = \delta x/\delta t$ where $\delta x$ and $\delta t$ are the space- and time-steps respectively and $c_s^2 = c^2/3$.




(a) D3Q7 (b) D3Q15 (c) D3Q19

Figure 1: 3D lattices of LB scheme.

### *3.1. Incompressible Navier-Stokes*

Several lattice Boltzmann schemes exist for incompressible version of Navier-Stokes equations. The fully incompressible condition has already been proposed in literature but necessitates to solve an additional Poisson equation [54] or an additional predictor-corrector step [29]. Here we prefer to apply the artificial compressibility method [55] for which the solenoidal condition $\boldsymbol{\nabla}\cdot\mathbf{u}=0$ is approximated by $(1/\beta)\partial p/\partial t+\boldsymbol{\nabla}\cdot\mathbf{u}=0$ where $\beta$ is the artificial compressibility coefficient. In LB framework, the method was derived in [56] with $\beta=\rho_0 c_s^2$ where $\rho_0$ is the constant density of bulk phase. The LB scheme writes

$$\overline{f}_i^{\star}=\overline{f}_i-\frac{1}{\overline{\tau}_f+1/2}\left[\overline{f}_i-\overline{f}_i^{eq}\right]+\mathcal{S}_i^f\delta t, \tag{26a}$$

$$f_i^{eq}=w_i\left[p+\rho(\phi)c_s^2\left(\frac{\mathbf{c}_i\cdot\mathbf{u}}{c_s^2}+\frac{(\mathbf{c}_i\cdot\mathbf{u})^2}{2c_s^4}-\frac{\mathbf{u}\cdot\mathbf{u}}{2c_s^2}\right)\right], \tag{26b}$$

with $\overline{f}_i^{eq}=f_i^{eq}-\mathcal{S}_i^f\delta t/2$ and $\rho(\phi)$ is given by Eq. (1a). In Eq. (26a) $\overline{\tau}_f$ is the collision rate which is related to the kinematic viscosity by $\nu=\overline{\tau}_f c_s^2\delta t$. Hence, the collision rate is obtained by $\overline{\tau}_f(\phi)=3\nu(\phi)(\delta t/\delta x^2)$ with the kinematic viscosity $\nu(\phi)$ interpolated by Eq. (1b). In Eq. (26a), the source term $\mathcal{S}_i^f$ contains contributions of external forces (involving $\mathbf{F}_{tot}$) plus the production term in mass conservation (involving $\dot{m}'''$):

$$\mathcal{S}_i^f=\mathcal{F}_i^f+\mathcal{P}_i^f \tag{27a}$$

with [57]

$$\mathcal{F}_i^f=(\mathbf{c}_i-\mathbf{u})\cdot\left[(\Gamma_i-w_i)\boldsymbol{\nabla}\rho(\phi)c_s^2+\Gamma_i\mathbf{F}_{tot}\right], \tag{27b}$$

$$\mathcal{P}_i^f=w_i\rho c_s^2\dot{m}'''\left(\frac{1}{\rho_g}-\frac{1}{\rho_l}\right). \tag{27c}$$

In Eq. (27b), $\mathbf{F}_{tot}$ is the external force defined by Eq. (12) and the function $\Gamma_i\equiv\Gamma_i(\mathbf{u})$ is defined by:

$$\Gamma_i=w_i\left[1+\frac{\mathbf{c}_i\cdot\mathbf{u}}{c_s^2}+\frac{(\mathbf{c}_i\cdot\mathbf{u})^2}{2c_s^4}-\frac{\mathbf{u}\cdot\mathbf{u}}{2c_s^2}\right]. \tag{27d}$$

After the stages of collision and streaming, the first-order moment (momentum) and the zeroth-order moment (pressure) are updated by [57]

$$\rho\mathbf{u}=\frac{1}{c_s^2}\sum_i\overline{f}_i\mathbf{c}_i+\frac{\delta t}{2}\mathbf{F}_{tot}, \tag{28a}$$

$$p=\sum_i\overline{f}_i+\frac{\delta t}{2}\left\{\mathbf{u}\cdot\boldsymbol{\nabla}\rho c_s^2+\rho c_s^2\dot{m}'''\left(\frac{1}{\rho_g}-\frac{1}{\rho_l}\right)\right\}. \tag{28b}$$

### 3.2. Conservative Allen-Cahn model

The lattice Boltzmann equation for the conservative Allen-Cahn model acts on the distribution function $\overline{g}_i$. The evolution equation is

$$\overline{g}_i^\star = \overline{g}_i - \frac{1}{\overline{\tau}_g + 1/2}\left[\overline{g}_i - \overline{g}_i^{eq}\right] + \mathcal{S}_i^g \delta t, \tag{29a}$$

$$g^{eq} = \phi \Gamma_i, \tag{29b}$$

with the variable change $\overline{g}_i^{eq} = g^{eq} - \delta t \mathcal{S}_i^g/2$. The mobility coefficient is related to the collision rate by $M_\phi = \overline{\tau}_g c_s^2 \delta t$. The source term $\mathcal{S}_i^g$ contains two contributions:

$$\mathcal{S}_i^g = \mathcal{F}_i^g + \mathcal{P}_i^g, \tag{30a}$$

where the first one $\mathcal{F}_i^g$ involves the counter term with the normal vector $\mathbf{n}$ [34], and the second one $\mathcal{P}_i^g$ involves the mass production term $\dot{m}'''$:

$$\mathcal{F}_i^g = \frac{4}{W}\phi(1-\phi) w_i \mathbf{c}_i \cdot \mathbf{n} \quad \text{and} \quad \mathcal{P}_i^g = w_i \frac{\dot{m}'''}{\rho_g}. \tag{30b}$$

Let us notice that the scheme is equivalent (see AppendixC.2) to the lattice Boltzmann equation

$$\overline{g}_i^\star = \overline{g}_i - \frac{1}{\overline{\tau}_g + 1/2}\left[\overline{g}_i - \overline{g}_i^{eq,CAC}\right] + \mathcal{P}_i^g \delta t \tag{31a}$$

where only the source term $\mathcal{P}_i^g$ appears in the source term and the equilibrium distribution function is redefined as [33]

$$g_i^{eq,CAC} = \phi \Gamma_i + M_\phi \frac{4}{W}\phi(1-\phi) w_i \frac{\mathbf{c}_i \cdot \mathbf{n}}{c_s^2} \tag{31b}$$

with $\overline{g}_i^{eq,CAC} = g_i^{eq,CAC} - \delta t \mathcal{P}_i^g/2$.

After the stages of collision and streaming, the new phase-field is obtained by the zeroth-order moment of $\overline{g}_i$ which must be corrected with the production term:

$$\phi(\mathbf{x}, t) = \sum_i \overline{g}_i + \frac{\delta t}{2}\sum_i \mathcal{P}_i^g. \tag{32}$$

This relation holds for both formulations that use $\overline{g}_i^{eq}$ and $\overline{g}_i^{eq,CAC}$ because $\sum_i \mathcal{F}_i^g \delta t/2 = 0$.

### 3.3. Temperature equation

The lattice Boltzmann scheme for temperature equation writes:

$$\overline{s}_i^\star = \overline{s}_i - \frac{1}{\overline{\tau}_s + 1/2}\left[\overline{s}_i - \overline{s}_i^{eq}\right] + \mathcal{S}_i^s \delta t \tag{33a}$$

$$s_i^{eq} = T\Gamma_i \tag{33b}$$

where the thermal diffusivity $\alpha$ is related to the collision rate by $\alpha = \overline{\tau}_s c_s^2 \delta t$. The source term $\mathcal{S}_i^s$ is defined such as:

$$\mathcal{S}_i^s = \mathcal{F}_i^s + \mathcal{P}_i^s \tag{33c}$$

where

$$\mathcal{F}_i^s = w_i \frac{\mathcal{L}}{\mathcal{C}_p}\boldsymbol{\nabla}\cdot(\mathbf{u}\phi) \quad \text{and} \quad \mathcal{P}_i^s = w_i \frac{\mathcal{L}}{\mathcal{C}_p}\frac{\partial \phi}{\partial t} \tag{33d}$$

Finally, the new temperature is computed by

$$T = \sum_i \overline{s}_i - \frac{\delta t}{2}\frac{\mathcal{L}}{\mathcal{C}_p}\left[\frac{\partial \phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi)\right]. \tag{34}$$

In Sections 4 and 5, simulations will be carried out with Dirichlet boundary conditions applied on temperature $T$ and phase-field $\phi$. In order to impose such a condition, for example on temperature $T_w$ on left boundary of a D2Q9 lattice, the unknown distribution functions $\overline{s}_i|_{unknown}$ are updated with the anti bounce-back method [58]: $\overline{s}_i|_{unknown} = -\overline{s}_{i'} + 2w_i T_w$ where $i'$ is the opposite direction of $i$.

| Intel CPUs | NVidia GPUs | ARM | IBM | AMD |
|---|---|---|---|---|
| Sandy/Ivy Bridge | Kepler | ThunderX | Blue gene Q | AMD CPUs |
| Haswell | Maxwell | ARMv8.0 | Power7 | |
| Skylake | Pascal | ARMv8.1 | Power8 | |
| Westmere CPUs | Volta | | Power9 | |
| Knights Landing/Corner Xeon Phi | Turing | | | |
| Broadwell Xeon E-class | | | | |

Table 1: List of architectures that are currently compatible with the `Kokkos` library.

### *3.4. Computations of gradients and Laplacian*

The unit normal vector $\mathbf{n}$ and force term $\mathbf{F}_s$ require computation of gradients. Moreover the chemical potential $\mu_\phi$ necessitates to calculate the Laplacian of $\phi$. Gradients and Laplacian that are involved in definitions of $\mathbf{n}$ (Eq. (6)) and $\mu_\phi$ (Eq. (14)) are discretized by using the directional derivatives methods. The method has already demonstrated its performance for hydrodynamics problem in order to reduce parasitic currents for two-phase flow problem [28, 59, 60]. The directional derivative is the derivative along each moving direction on the lattice. Taylor's expansion at second-order of a differentiable scalar function $\phi(\mathbf{x})$ at $\mathbf{x}+\mathbf{e}_i\delta x$ and $\mathbf{x}-\mathbf{e}_i\delta x$ yields the following approximation of directional derivatives:

$$\mathbf{e}_i\cdot\boldsymbol{\nabla}\phi\big|_{\mathbf{x}} = \frac{1}{2\delta x}\left[\phi(\mathbf{x}+\mathbf{e}_i\delta x)-\phi(\mathbf{x}-\mathbf{e}_i\delta x)\right] \tag{35a}$$

The number of directional derivatives is equal to the number of moving direction $\mathbf{e}_i$ on the lattice i.e. $N_{pop}$. The gradient is obtained by

$$\boldsymbol{\nabla}\phi\big|_{\mathbf{x}} = 3\sum_{i=1}^{N_{pop}} w_i\mathbf{e}_i\left(\mathbf{e}_i\cdot\boldsymbol{\nabla}\phi\big|_{\mathbf{x}}\right). \tag{35b}$$

The three components of the gradient $\partial_x\phi$, $\partial_y\phi$ and $\partial_z\phi$ are obtained by calculating each directional derivative $\mathbf{e}_i\cdot\boldsymbol{\nabla}\phi\big|_{\mathbf{x}}$ and next, by calculating the moment of first-order $\boldsymbol{\nabla}\phi\big|_{\mathbf{x}}$. For the calculation of $\boldsymbol{\nabla}^2\phi$, all directions of propagation are taken into account by

$$(\mathbf{e}_i\cdot\boldsymbol{\nabla})^2\phi\big|_{\mathbf{x}} = \frac{1}{\delta x^2}\left[\phi(\mathbf{x}+\mathbf{e}_i\delta x)-2\phi(\mathbf{x})+\phi(\mathbf{x}-\mathbf{e}_i\delta x)\right]. \tag{36a}$$

The Laplacian is obtained by summing and weighting each term with

$$\boldsymbol{\nabla}^2\phi\big|_{\mathbf{x}} = 3\sum_{i=1}^{N_{pop}} w_i(\mathbf{e}_i\cdot\boldsymbol{\nabla})^2\phi\big|_{\mathbf{x}}. \tag{36b}$$

Other approximations exist [28, 61] such as the first-order and second-order upwind schemes (or biased differences) respectively defined by $\mathbf{e}_i\cdot\boldsymbol{\nabla}^{up_1}\phi\big|_{\mathbf{x}} = [\phi(\mathbf{x}+\mathbf{e}_i\delta x)-\phi(\mathbf{x})]/\delta x$ and $\mathbf{e}_i\cdot\boldsymbol{\nabla}^{up_2}\phi\big|_{\mathbf{x}} = [-\phi(\mathbf{x}+2\mathbf{e}_i\delta x)+4\phi(\mathbf{x}+\mathbf{e}_i\delta x)-3\phi(\mathbf{x})]/(2\delta x)$. Here, by simplicity, the central difference approximation is applied for all simulations even though that approximation fails to capture the velocity profiles in low density regions [62] and biased directional derivatives can fix that issue [2]. Those biased differences could be tested in future works with `LBM_saclay`.

### *3.5. Numerical implementation and kernel optimization*

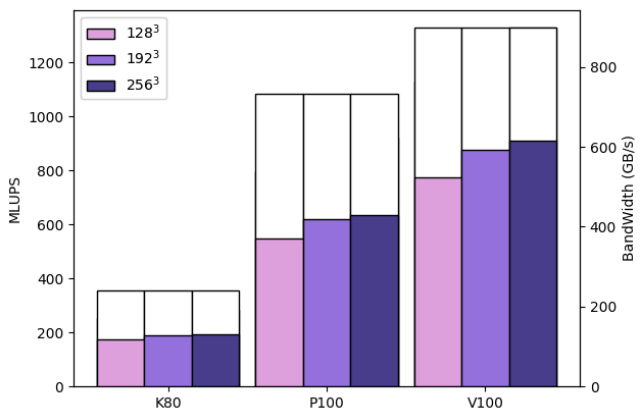
All LBM schemes of this Section were implemented in a new code called `LBM_saclay` written in C++. The main advantage of this new code is its portability targeting all major HPC platforms and especially those based on GPU- and CPU-architectures. Actually, `LBM_saclay` can run without modification on any architecture that `Kokkos` supports. The current compatibilities are indicated in [63] and summarized in Tab. 1. For more information, the reader can refer to the `Kokkos` documentation. Let us mention that the current support for AMD GPU is experimental through the C++ library `HIP` (Heterogeneous-Compute Interface for Portability) and it is planned to be supported at the end of 2020.

Two levels of parallelism are implemented in the code. The first one is the intra-node parallelism (shared memory) with the `Kokkos` library, an opensource C++ library with parallel algorithmic patterns and data containers. Specific commands of the `Kokkos` library optimize loops with `OpenMP`, `Pthreads` or `CUDA` during compilation. An example of using `Kokkos`' functionalities is presented on Fig. 2 to compute at each time-step the zeroth-order moment of a distribution function. The second level of parallelism is a standard domain decomposition performed with `MPI`: the full computational domain is cut into several sub-domains associated with each computational node (distributed memory).

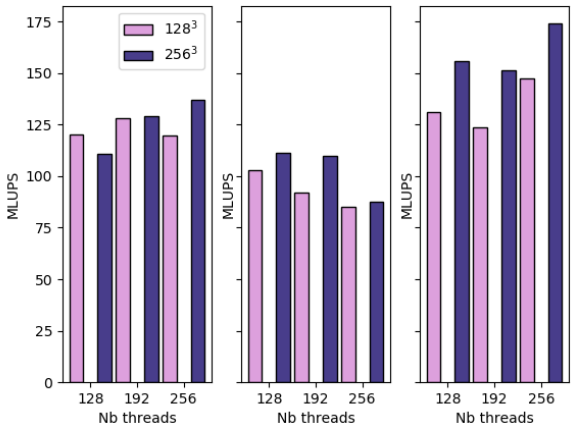
When developing the code, several optimizations were implemented and compared in particular to enhance its performance on each architecture. The first way to consider the stages of collision and streaming is to "fuse" those two steps inside a single kernel, i.e. both stages are simply done in one single "`for`-loop" performed on the lattice nodes. The "fused" version does not require an intermediate memory load contrary to standard implementation for which both stages are well separated.

```
class CalcMoment0_Functor {
public:
        CalcMoment0_Functor(/* ... */) { /* ... */ };
        KOKKOS_INLINE_FUNCTION
        void operator()(const int& index) const {
                const int isize = /* Lattice's maximum x-axis index. */;
                const int jsize = /* Lattice's maximum y-axis index. */;
                const int npop  = /* Lattice's direction count.      */;
                int i, j;
                /* Maps the 1D dispatch index to the 2D node index. */
                index2coord(index, i, j, isize, jsize);
                if (j < jsize && i < isize ) {
                        double mom = 0;
                        for (int k = 0; k < npop; ++k) {
                                mom = mom + f(i,j,k);
                        }
                        density(i, j) = mom;
                }
        }

        // Multi-dimensional containers allocated on the CPU or GPU.
        Kokkos::View<double**, Device>  density; /* 2 space dimensions.    */
        Kokkos::View<double***, Device> f;       /* 2 space dimensions,    */
                                                 /* + 1 velocity dimension */
};

// Elsewhere in the code, the CalcMoment0_Functor's operator() can be
// dispatched by Kokkos.
CalcMoment0_Functor functor(/* ... */);
const int iter_count = isize*jsize;
Kokkos::parallel_for(iter_count, functor);
```

Figure 2: Example of using the `Kokkos` library to compute the zeroth-order moment of distribution function.




(a) Comparison of computational times for three NVIDIA® graphical cards: K80 (oldest), P100 and V100 (newest).




(b) Comparisons of computational times for three optimizations of LBM kernel for Intel® KNL: fused (left), CSoA (middle) and CSoA2 (right).

Figure 3: Computational times (in Million Lattice Updates Per Second – MLUPS) for a diffusive problem with a D3Q19 lattice. (a) GPU for three mesh sizes and (b) on CPU (Intel® KNL) for two mesh sizes.

However, the fused kernel contains more floating point operations per iteration of the "`for`-loop". This is a drawback if the number of floating point operations becomes large enough to exhaust the amount of register memory available on the architecture (this number is significantly lower on GPU than on CPU). So, if the register memory is full, additional variables will be allocated in the external DRAM memory, generating additional traffic on the memory bus and degrading performance. For GPUs NVIDIA® and CPUs Intel® Skylake, best performance is obtained with the fused version.

Alternatively, two optimizations were tested which are well suited for Intel® KNL (KNights Landing) processors [64]: the first one is the "CSoA" optimization (Cluster of Structure of Array) i.e. for each line of the lattice, LBM nodes are stored in memory modulo $M$ where typically $M = 8$ and each line is padded to be a multiple of $M$. The access of data container is done with `data(iMem,j,k,ipop)` where `iMem` is computed from the physical node location `i`. The CSoA optimization improves vectorization and memory alignment for streaming stage but performance decreases for large domain on D2Q9 lattice. The second optimization for KNL is "CSoA2", i.e. the population index `ipop` of `data(i,j,k,ipop)` is interverted to `data(i,ipop,j,k)`, where `i,j,k` are indices of position. With this permutation, the memory locality is restored for the collision stage.

Comparisons were performed on a simplified diffusive problem. The CSoA2 optimization enhances performance on KNL processors, but on Fig. 3, we can see that it remains far below to that obtained on GPUs, even older generation GPUs (K80). Computational times are expressed in Million Lattice Updates per Second (MLUPS) as an effective metric measuring the number of millions of node lattice update per seconds. That performance metric is used by node so that it is independent of the type of lattice (e.g. D3Q7, D3Q15, D3Q19). With that metric, a larger lattice will give a smaller MLUPS. In the rest of this paper, most of validations and simulations of Sections 4 and 5 are carried out on GPUs. In Section 5.3, comparisons of computational times on GPU and CPU will be presented on the test case of film boiling for two mesh sizes.

Finally, let us mention that all kernels (Navier-Stokes, phase-field and temperature equations) have been developed in 2D and in 3D. They all run in 3D separately. However, all coupling terms, i.e. the surface tension force (Eq. (13)), the chemical potential (Eq. (14)) and the advective term in Eq. (33d)) were developed and checked only in 2D. Hence, verification of couplings in Section 4 and film boiling simulations of Section 5 will be presented only in 2D. The three-dimensional extension of coupling terms is planned for future works.

## 4. Code verifications

In this section, the numerical implementation of the LBM schemes of Section 3 is checked by comparison with well-known solutions. Validations are gathered into two parts in order to check implementations step-by-step. In subsection 4.1, verifications are done without phase change, i.e. by neglecting the temperature equation and by assuming that the mass transfer is zero ($\dot{m}''' = 0$ in Eq. (11a) and (11c)). The conservative Allen-Cahn model, and the coupling with fluid flow are verified successively. In subsection 4.2, the phase change model is checked by considering the phase-field equation coupled with temperature. The LBM code is compared with an analytical solution of Stefan's problem with two different diffusivities.

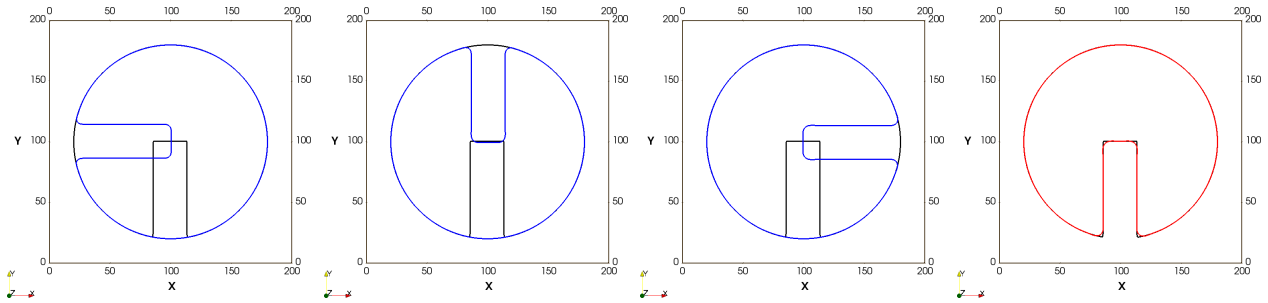
### *4.1. Verifications without phase change*

We first compare implementation of the conservative Allen-Cahn model on two test cases: Zalesak's slotted disk and interface deformation inside a vortex. Next the coupling with Navier-Stokes model will be considered with the layered Poiseuille flow and the Laplace law.

#### *4.1.1. Verifications of the phase-field model*

Two verifications of phase-field implementation are presented. In the first one, we check that the contour of a slotted disk is well conserved inside a rotating fluid [65]. In the second one, we check that the simulation retrieves a circle when an initial disk is deformed inside a vortex that changes its direction of rotation over time. For both simulations, the mesh is composed of $201 \times 201 \times 3$ nodes with periodic boundary conditions applied on all faces, the time-step is $\delta t = 10^{-4}$ and the space-step $\delta x = 5 \times 10^{-3}$.

*Zalesak's slotted disk.* Inside a domain of lengths $L_x = L_y = 1$, and $L_z = 0.01$, a disk is initialized at the center of the domain $\mathbf{x}_c = (100, 100, 1)^T$ by $\phi(\mathbf{x}, 0) = \left[1 + \tanh\left((R - d_c)/\sqrt{2}W_0\right)\right]/2$ with $d_c = \sqrt{(x-x_c)^2 + (y-y_c)^2 + (z-z_c)^2}$, $W_0 = 2$ and $R = 80$ l.u. (lattice units). The diffuse disk is slotted by imposing $\phi(\mathbf{x}, 0) = 0$ if $x_c - R/6 \leq x \leq x_c + R/6$ and $y_c - 1.1R \leq y \leq y_c$. Components of velocity are imposed by $u_x(\mathbf{x}) = u_0(2y - 1)$, $u_y(\mathbf{x}) = u_0(1 - 2x)$ and $u_z(\mathbf{x}) = 0$. The value of $u_0$ is chosen such that the slotted disk performs one complete rotation at $T_f = 4$, i.e. $u_0 = 0.7853975$ and both parameters of CAC model are set as $M_\phi = 5 \times 10^{-4}$ and $W = 6\delta x$. The rotation of the slotted disk is presented on Fig. 4 where the interface position $\phi = 1/2$ is superimposed to the initial condition at four times. At the final time of simulation $t = T_f$ (Fig. 4d), the contour $\phi = 0.5$ (red) is superimposed to the initial one (black) although the slot corners are slightly rounded.



(a) $t = T_f/4$ (b) $t = T_f/2$ (c) $t = 3T_f/4$ (d) $t = T_f$

Figure 4: Zalesak's slotted disk with the conservative Allen-Cahn model.

*Vortex.* We study the deformation of an initial disk standing inside a 2D vortex. The three components of velocity are defined by $u_x(\mathbf{x}) = -u_0 \cos[\pi(x-0.5)] \sin[\pi(y-0.5)]$, $u_y(\mathbf{x}) = u_0 \sin[\pi(x-0.5)] \cos[\pi(y-0.5)]$ and $u_z(\mathbf{x}) = 0$. LB simulations are performed on a D3Q19 lattice for a 3D domain with a very small thickness in $z$-direction. The initial condition $\phi(\mathbf{x}, 0)$ is defined by a full disk centered at $\mathbf{x}_c = (100, 60, 1)^T$, with $W = 2$ and $R = 40$ l.u. The initial condition ($\phi = 0.5$) and streamlines for $u_0 = 0.7853975$ are presented on Fig. 5a-(i). The rotation is directed counterclockwise. Parameters are $T_f = 4$, $W = 6\delta x$ and $M_\phi = 5 \times 10^{-4}$. For $t = T_f/2$ (Fig. 5a-(ii)) and $t = T_f$ (Fig. 5a-(iii)) black contours $\phi = 0.5$ are comparable to those presented in reference [32, Fig. 4]. Next, the velocity is changed during the simulation by multiplying $\mathbf{u}(\mathbf{x})$ with a factor depending on time: $\mathbf{u}'(\mathbf{x}, t) = \mathbf{u}(\mathbf{x}) \times \cos(\pi t/2T_f)$. With the cosine function, the velocity $\mathbf{u}'(\mathbf{x}, t)$ presents three stages during the simulation: when $t < T_f$, the direction of rotation is counterclockwise (Fig. 5b-(i)); when $t = T_f$ the cosine function cancels the velocity $\mathbf{u}'$ (Fig. 5b-(ii)); and when $t > T_f$, the sign changes and the direction of rotation becomes clockwise (Fig. 5b-(iii)). At the end of simulation $t = 2T_f$, we expect to find the shape of initial disk. That is what we observe on Fig. 5b-(iv) which confirms that the interface position $\phi = 0.5$ is similar to the initial condition one (Fig. 5a-(i)).

### *4.1.2. Verifications of phase-field with fluid flow model*

Two classical test cases are presented to check the coupling of phase-field equation and fluid flow model: the layered Poiseuille flow and the Laplace law.

*Layered Poiseuille flow.* The Navier-Stokes implementation is checked with the analytical solution of a layered Poiseuille flow [29] for two fluids named $A$ and $B$:

$$u_x(y) = \begin{cases} \frac{Gh^2}{2\eta_A}\left[-\left(\frac{y}{h}\right)^2 - \frac{y}{h}\left(\frac{\eta_A-\eta_B}{\eta_A+\eta_B}\right) + \frac{2\eta_A}{\eta_A+\eta_B}\right] & (-h \le y \le 0) \\ \frac{Gh^2}{2\eta_B}\left[-\left(\frac{y}{h}\right)^2 - \frac{y}{h}\left(\frac{\eta_A-\eta_B}{\eta_A+\eta_B}\right) + \frac{2\eta_B}{\eta_A+\eta_B}\right] & (0 \le y \le h) \end{cases} \tag{37}$$

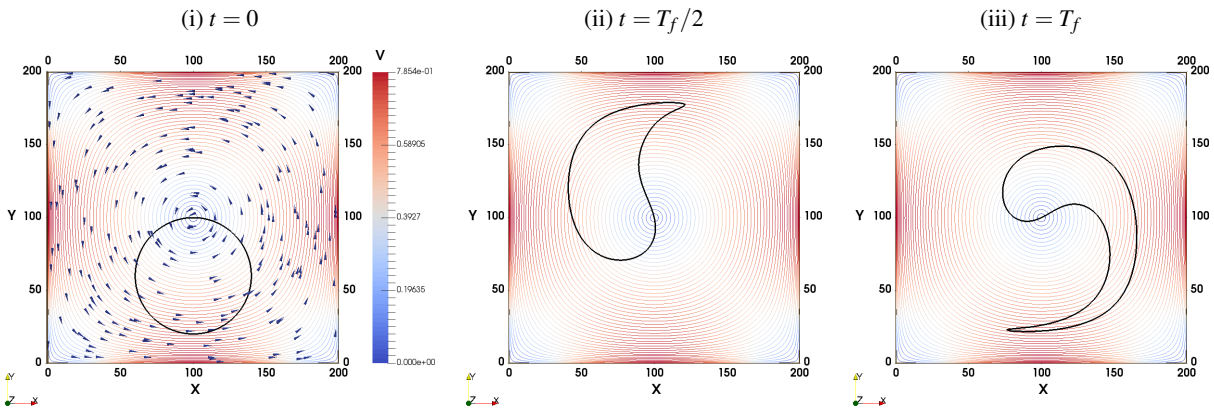
where $\eta_A$ and $\eta_B$ are the dynamic viscosities and $2h$ is the channel width. The pressure gradient is defined by $G = u_c(\eta_A + \eta_B)/h^2$ with $u_c = 5 \times 10^{-5}$. For the LB simulation, the mesh is composed of $101 \times 101 \times 3$ nodes and the pressure gradient is replaced by a force term defined by $\mathbf{F} = (G, 0, 0)^T$. Periodic boundary conditions are set for all limits except for planes of normal vector directed in $y$-direction where no-slip conditions are imposed with the half bounce-back method. Two layers of different viscosity are defined as initial condition for $\phi$: $\phi(\mathbf{x}, 0) = 0.5\{1 + \tanh[2(y - y_0)/W]\}$ where $W = 6\delta x$ controls the slope of the hyperbolic tangent function and $y_0 = (y_{max} + y_{min})/2$. The mobility coefficient is $M_\phi = 0.1$. Comparisons between the LBM code and the analytical solution are presented for two cases. In the first one, the density is identical for both fluids ($\rho_A = \rho_B = 1$) and three viscosity ratios are checked on Fig. 6a: $\eta_B/\eta_A = 1/3, 1/5, 1/10$. For the first ratio $\nu_A = 0.1$ and $\nu_B = 0.3$; for the second one $\nu_A = 0.07$ and $\nu_B = 0.35$ and for the third one $\nu_A = 0.01$ and $\nu_B = 0.1$. For the second test case, the viscosity of each phase is set equal to $\nu_A = \nu_B = 0.07$ and three density ratios are checked on Fig. 6b: $\rho_A/\rho_B = 1/1.658, 1/2, 1/3$. The ratio $1/1.658$ is used in the simulations of film boiling as well as the viscosity ratio $\nu_A/\nu_B = 1/6$. In Fig. 6c this viscosity ratio is checked for two cases. In the first simulation (red curve), the density ratio is equal to one and the viscosity is interpolated by two methods: the linear (black squares) and the harmonic mean (red circles) defined by

$$\nu(\phi) = [1 - \phi(\mathbf{x}, t)]\nu_A + \phi(\mathbf{x}, t)\nu_B, \tag{38}$$

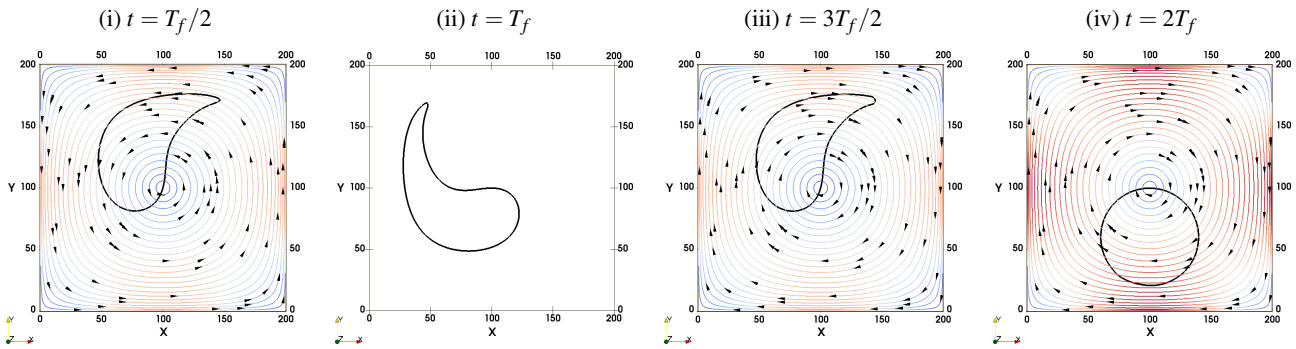
and Eq. (1b) respectively. The differences observed with the former method justify the choice of using the latter in the second simulation (blue curve) which combines both ratios of viscosity and density.

*Laplace law.* The two-dimensional Laplace law is checked by initializing a drop at the center of a square domain of length $L_x = L_y = 2.56$ discretized with $256 \times 256$ nodes. By varying the radius $R$, the difference between pressure inside the drop ($p_{in}$) minus the pressure outside ($p_{out}$) must vary proportionally with the surface tension $\sigma$:

$$p_{in} - p_{out} = \frac{\sigma}{R}. \tag{39}$$




(a) Without change of rotation during simulation. Streamlines of $\mathbf{u}$ (colored lines) and interface $\phi = 0.5$ (black line) at three times.




(b) With a change of rotation direction during simulation. Streamlines of $\mathbf{u}'$ (colored lines) and contours $\phi = 0.5$ (black lines) for four times.

Figure 5: Deformation of an initial disk standing inside a vortex. (a) Without change of rotation during simulation. (b) With change of rotation.

In order to check that relationship, an initial drop of radius $R$ and surface tension $\sigma$ is initialized at the center of the domain ($x_c = y_c = 1.28$). The density ratio $\rho_g/\rho_l$ is set equal to two ($\rho_g = 2$, $\rho_l = 1$) and the viscosities are identical for each phase: $\nu_l = \nu_g = 0.04$. The interface parameters are $M_\phi = 0.04$ and $W = 0.05 = 5\delta x$. The LBM code is run with a time-step equal to $\delta t = 10^{-4}$ until the stationary solution is obtained. At the end of simulation, the difference between numerical pressures $\Delta p = p_{in} - p_{out}$ is plotted for three values of surface tension $\sigma = 0.04, 0.08, 0.15$. For each value of surface tension, six LBM simulations are run for six values of radius corresponding to each dot on Fig. 6d. On that plot, the slopes of LBM vary linearly and fit quite well to the Laplace law.

### *4.2. Verifications with phase change: one-dimensional Stefan problem*

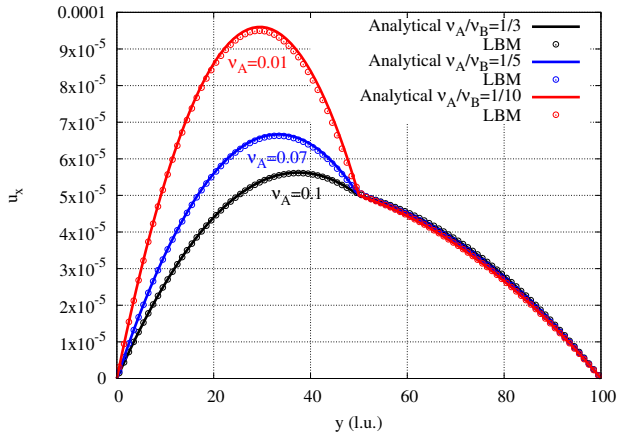
In this section, we consider the problem of phase change without flow ($\mathbf{u} = \mathbf{0}$). The objective is to validate the coupling between equations of phase-field and temperature. More precisely, we check the new approximation (Eq. (18)) of mass production rate $\dot{m}'''$ in the phase-field equation (Eq. (11c)) and the latent heat release in the temperature equation (Eq. (11d)), i.e. the source term $-\partial\phi/\partial t$. Validation is carried out with the Stefan problem for which several analytical solutions exist [66, Chapter 12]. Here we consider one of the most general one-dimensional problem where the three unknowns are the interface position varying with time $x_I(t)$, the liquid temperature $T_l(x,t)$ and the gas temperature $T_g(x,t)$. Besides, the thermal diffusivities of each phase $\alpha_l$ and $\alpha_g$ can be different. The one-dimensional domain $]0,\infty[$, is initially filled with gas with constant temperature $T_g(x,t)\big|_{x>0,t=0} = T_\infty$ that is greater than the saturation temperature $T_{sat}$. The left wall $x = 0$ is maintained at $T_w$ for $t \geq 0$. As a result, condensation starts at the boundary $x = 0$ and the liquid-gas interface propagates in the positive direction. At $x \to \infty$, the temperature is kept at $T_\infty$.

*Analytical solutions.* The mathematical formulation of this problem writes [66, Section 12-3]

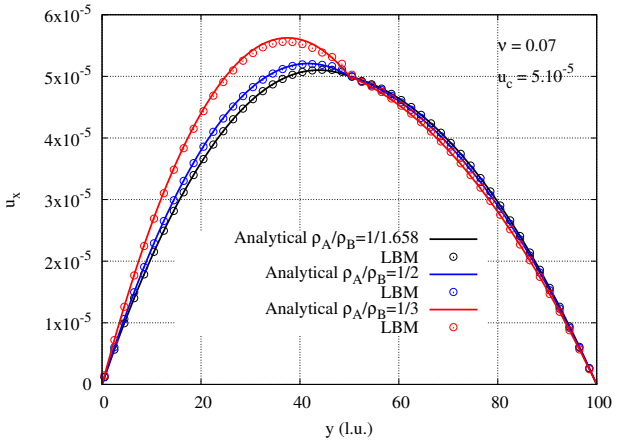
$$\frac{\partial T_l}{\partial t} = \alpha_l \frac{\partial^2 T_l}{\partial x^2} \tag{40a}$$

for $0 < x < x_I(t)$, with the left boundary condition imposed at $T_l(x,t)|_{x=0} = T_w$. The evolution of the gas phase is formulated as

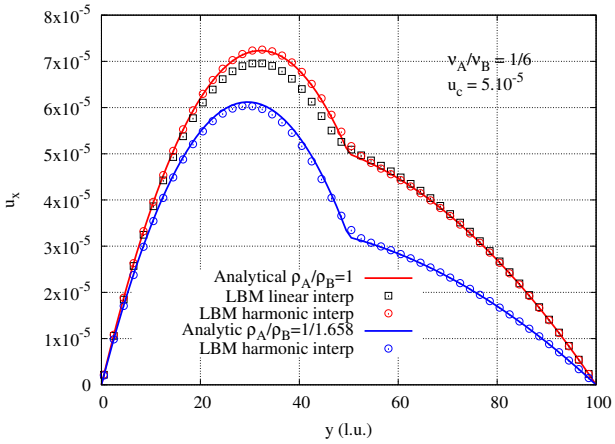
$$\frac{\partial T_g}{\partial t} = \alpha_g \frac{\partial^2 T_g}{\partial x^2} \tag{40b}$$



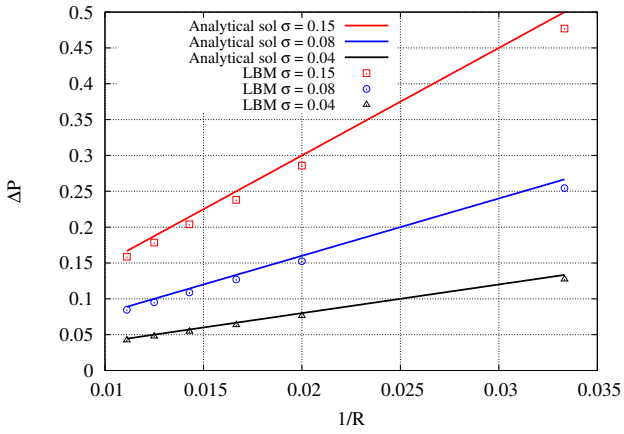

(a) Comparisons between LBM and the double-Poiseuille analytical solution for three viscosity ratios and $\rho_A/\rho_B = 1$.




(b) Comparisons between LBM and the double-Poiseuille analytical solution for three density ratios and $\nu_A/\nu_B = 1$.




(c) Red line – comparisons between two interpolation methods of viscosity: linear (black squares) and harmonic mean (red circles) defined by Eqs. (38) and (1b) respectively. Blue line – verification for $\rho_A/\rho_B = 1/1.658$.




(d) Laplace's law verification for three values of surface tension.

Figure 6: Verification of coupling between the phase-field equation and fluid flow model without phase change. (a) Double-Poiseuille flow with three viscosity ratios. (b) Double-Poiseuille for three density ratios. (c) Effect of linear interpolation and harmonic mean of viscosity. (d) Laplace law.

for $x_I(t) < x < \infty$ with $T_g(x \to \infty, t) = T_\infty$, with the initial condition $T_g(x, t = 0) = T_\infty$ and boundary condition $T_g(x \to \infty, t) = T_\infty$. Interfacial conditions are specified by

$$T_l(x,t)\big|_{x=x_I(t)} = T_g(x,t)\big|_{x=x_I(t)} = T_I, \tag{40c}$$

$$\mathcal{K}_l \left.\frac{\partial T_l}{\partial x}\right|_{x=x_I(t)} - \mathcal{K}_g \left.\frac{\partial T_g}{\partial x}\right|_{x=x_I(t)} = \rho \mathcal{L} \frac{dx_I(t)}{dt}. \tag{40d}$$

In Eq. (40d), $\mathcal{K}_l$ and $\mathcal{K}_g$ are the thermal conductivities of each phase. We consider identical specific heat $\mathcal{C}_p^l = \mathcal{C}_p^g = \mathcal{C}_p$ and we set $\mathcal{C}_p = 1$, $\mathcal{L} = 1$ and $\rho = 1$. Solutions of interface position and temperature profiles [66, p. 469] are

$$x_I(t) = 2\xi\sqrt{\alpha_l t}, \tag{41a}$$

$$\theta_l(x,t) = \theta_w + (\theta_I - \theta_w)\frac{\mathrm{erf}(x/2\sqrt{\alpha_l t})}{\mathrm{erf}(\xi)}, \tag{41b}$$

$$\theta_g(x,t) = \theta_\infty + (\theta_I - \theta_\infty)\frac{\mathrm{erfc}(x/2\sqrt{\alpha_g t})}{\mathrm{erfc}(\xi\sqrt{\alpha_l/\alpha_g})}, \tag{41c}$$

where the temperatures are re-written in dimensionless form with $\theta = \mathcal{C}_p(T - T_{sat})/\mathcal{L}$. When $\theta = 0$ the temperature of system is at saturation temperature $T_{sat}$ and when $\theta > 0$ (resp. $\theta < 0$), the system is superheated (resp. undercooled). In Eqs. (41a)–(40d), $\xi$ is solution of the transcendental equation

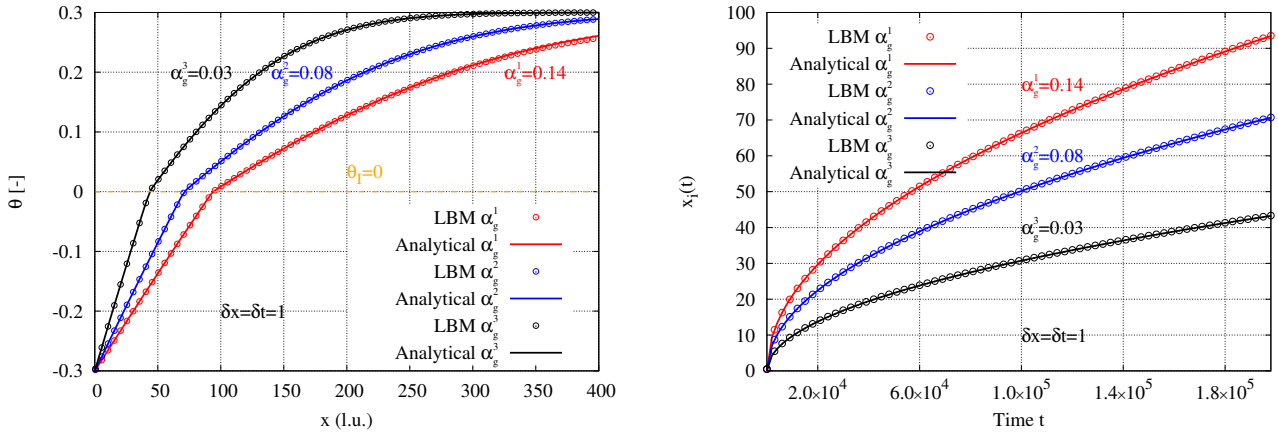
$$\frac{e^{-\xi^2}}{\mathrm{erf}(\xi)} + \left(\frac{\alpha_g}{\alpha_l}\right)^{1/2} \frac{\theta_I - \theta_\infty}{\theta_I - \theta_w} \frac{e^{-\xi^2(\alpha_l/\alpha_g)}}{\mathrm{erfc}(\xi\sqrt{\alpha_l/\alpha_g})} = -\frac{\xi\sqrt{\pi}}{\theta_w} \tag{41d}$$

where $\theta_w$ in the right-hand side is the Stefan number defined by $St = \mathcal{C}_p(T_w - T_{sat})/\mathcal{L}$. Those solutions are compared with `LBM_saclay`, first with identical thermal diffusivities $\alpha_l = \alpha_g$ and an interface temperature $\theta_I$ equals to zero. The second validation considers three ratios of diffusivity $\alpha_l^j/\alpha_g^j$ (for $j = 1, 2, 3$) with an interface temperature which is different of the saturation one ($\theta_I \neq 0$).

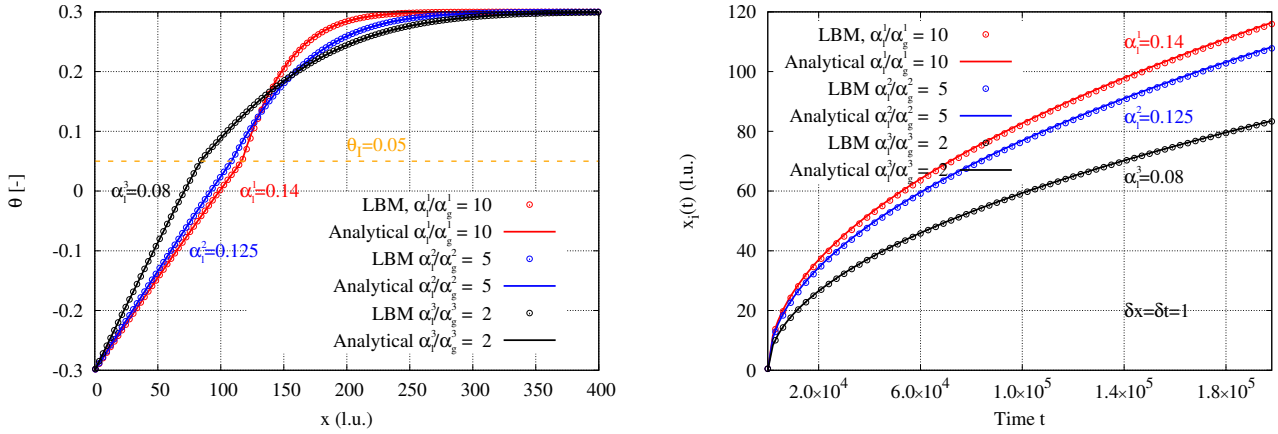
*Data entry of LBM simulations.* For LBM simulations, the two-dimensional D2Q9 lattice is used for the temperature and phase-field equations. The LBM computational domain is $[\ell_x, L_x] \times [\ell_y, L_y] = [0, 512] \times [0, 32]$ which is discretized by $N_x \times N_y = 512 \times 32$ nodes i.e. $\delta x = 1$. The time-step is also set to $\delta t = 1$. Boundary conditions are periodic for $s_i$ and $g_i$ at $\ell_y$ and $L_y$ (bottom and top walls respectively) and Dirichlet boundary conditions are applied on left ($x = \ell_x$) and right ($x = L_x$) walls by anti-bounceback method on $g_i$ and $s_i$. For phase-field, the Dirichlet boundary conditions are $\phi(x,t)\big|_{x=\ell_x} = 0$ and $\phi(x,t)\big|_{x=L_x} = 1$. For the temperature equation, they are $\theta(x,t)\big|_{x=\ell_x} = \theta_w$ and $\theta(x,t)\big|_{x=L_x} = \theta_\infty$. The temperature is initialized with $\theta(x,0) = \theta_\infty$ for $0 < x \leq L_x$ and the phase-field with $\phi(x,0) = 0.5\,[1 + \tanh(2x/W)]$. The mobility parameter is $M_\phi = 0.08$, the interface thickness is $W = 3\delta x$.

*Validations for $\alpha_l/\alpha_g = 1$ and $\theta_I = 0$.* Before considering the more general case $\alpha_l/\alpha_g \neq 1$ and $\theta_I \neq 0$, we assume that thermal diffusivities are the same in liquid and gas ($\alpha_l = \alpha_g = \alpha$) and the interface temperature is at saturation ($\theta_I = 0$). In that case, whatever the diffusivity value $\alpha$, the solution of the transcendental equation (Eq. (41d)) depends only on $\theta_w$ and $\theta_\infty$. With $\theta_w = -0.3$ and $\theta_\infty = 0.3$, its solution is $\xi = 0.280680$. Comparisons between analytical solutions and LBM simulations are presented on Fig. 7a for three values of thermal diffusivity $\alpha_g^j = 0.14, 0.08, 0.03$ with $j = 1, 2, 3$. LBM temperature profiles are superimposed with the analytical solution (Eqs. (41b) and (41c)) at the final time of simulation $t_f = 2 \times 10^5$ (Fig. 7a, left). Successive positions of vapor/liquid interface also fit with the analytical solution (Fig. 7a, right) for three values of thermal diffusivity.

*Validations for $\alpha_l/\alpha_g \neq 1$ and $\theta_I \neq 0$.* Now we consider a more general case for which the diffusivities of liquid and gas can be different. Three ratios are simulated $\alpha_l^j/\alpha_g^j = 10, 5, 2$ for $j = 1, 2, 3$ with $\alpha_l^1 = 0.14$, $\alpha_l^2 = 0.125$ and $\alpha_l^3 = 0.08$. Same values of $\theta_w = -0.3$ and $\theta_\infty = 0.3$ are kept, and the interface temperature is now equal to $\theta_I = 0.05$. For those values, the corresponding solutions of the transcendental equation are $\xi^1 = 0.349635$, $\xi^2 = 0.343882$ and $\xi^3 = 0.331864$. For LBM simulations, all numerical values are identical except for interface temperature and diffusivities of each phase. As confirmed by temperature profiles (Fig. 7b, left) and the evolution of interface position (Fig. 7b, right), the model of phase change is well adapted to simulate the phase change problem with different diffusivities in each phase and an interface temperature not equal to zero. Finally this test case validates the approximation of the mass production rate $\dot{m}'''$ defined by Eq. (18) and implementation of LBM for the phase-field and temperature equations.




(a) Comparisons for three values of thermal diffusivity: $\alpha_g^1 = 0.14$ (red), $\alpha_g^2 = 0.125$ (blue) and $\alpha_g^3 = 0.08$ (black). Left: $x$-profiles of temperature $\theta$ at the end of simulation $t_f = 2 \times 10^5$. Right: evolution of interface position $x_I(t)$ tracked by $\phi = 1/2$. The temperature interface is $\theta_I = 0$.




(b) Comparisons for three ratios $\alpha_l^j/\alpha_g^j = 10, 5, 2$ for $j = 1, 2, 3$ with $\alpha_l^1 = 0.14$ (red), $\alpha_l^2 = 0.125$ (blue) and $\alpha_l^3 = 0.08$ (black). Left: $x$-profiles of temperature $\theta$ at the end of simulation $t_f = 2 \times 10^5$. Right: evolution of interface position $x_I(t)$ tracked by $\phi = 1/2$. The temperature interface is $\theta_I = 0.05$.

Figure 7: Comparisons between LBM (dots) and analytical solution of Stefan problem (solid lines). (a) With $\alpha_l/\alpha_g = 1$ and $\theta_I = 0$. (b) With $\alpha_l/\alpha_g \neq 1$ and $\theta_I \neq 0$.

## 5. Simulations of film boiling

Film boiling is a classical problem of two-phase flows with phase change. It has already been simulated with a lot of different numerical techniques (see [39] for a recent review) for studying the effect of geometries such as an horizontal cylinder [67] or for studying the effect of an electric field [68]. With the lattice Boltzmann method, several simulations use the Cahn-Hilliard model or the pseudo-potential method (respectively in [69, 70, and references therein]). Here we present the capability of the conservative Allen-Cahn equation with a production rate defined by Eq. (18) to simulate that problem. In section 5.1, the physical configuration is reminded; in section 5.2 one simulation of bubbles detachment on nodes and anti-nodes is detailed; in section 5.3, indications will be given on computational times for two mesh sizes: $1024^2$ for GPU and CPU and $4096 \times 3072$ for multi-GPUs.

### *5.1. Physical configuration*

Inside a two-dimensional domain $\Omega = \Pi_{\upsilon=x,y}[\ell_\upsilon, L_\upsilon]$, a thin film of gas of height $y_0$ is initialized near the bottom wall $y = \ell_y$ which is heated by applying a constant temperature $\theta|_{y=\ell_y} = \theta_w$. The liquid is above the thin film and the gravity acts downward $\mathbf{g} = (0, -g_y)^T$. On the top wall $y = L_y$, the temperature is imposed at saturation and the phase-field is equal to $\phi = +1$ (i.e. gas phase). The left and right walls are periodic. If the interface is destabilized by an initial condition defined by

$$y = y_0 + y_1 \sin\left(\frac{2\pi x}{\lambda}\right), \tag{42}$$

where $y_1$ and $\lambda$ are respectively the amplitude and the wavelength of the perturbation, then we can observe bubbles of gas that grow, detach and rise in the domain, provided that the wavelength of perturbation $\lambda$ is greater than a critical value $\lambda_c$ defined by

| **Liquid and gas properties** | | | **Interface properties** | | **Other parameters** | |
|---|---|---|---|---|---|---|
| | Liquid | Gas | Parameter | Value | Parameter | Value |
| Density | $\rho_l = 1.658$ | $\rho_g = 1$ | Surface tension | $\sigma = 5 \times 10^{-3}$ | Gravity | $g_y = 4$ |
| Kinematic viscosity | $\nu_l = 3 \times 10^{-3}$ | $\nu_g = 5 \times 10^{-4}$ | Interface temp. | $\theta_I = 0$ | Bottom temp. | $\theta_{y=\ell_y} = 0.025$ |
| Thermal diffusivity | $\alpha_l = 2.5 \times 10^{-4}$ | $\alpha_g = 2.5 \times 10^{-3}$ | Mobility | $M_\phi = 1.7 \times 10^{-3}$ | Top temp. | $\theta_{y=L_y} = 0$ |
| | | | Interface width | $W = 5 \times 10^{-3}$ | Latent/specific heat | $\mathcal{L}/\mathcal{C}_p = 1$ |

Table 2: Parameters for film boiling simulations.

$$\lambda_s = \sqrt{\frac{\sigma}{(\rho_l - \rho_g) g_y}}, \qquad \lambda_c = 2\pi\lambda_s. \tag{43}$$

The thermal-hydrodynamics of this problem is controlled by several dimensionless numbers: the Grashof number $Gr = \rho_g g_y (\rho_l - \rho_g)\lambda_s^3/\rho_g^2\nu_g^2$, the Prandtl number $Pr = \nu_g/\alpha_g$ and the Jacob number $Ja = \mathcal{C}_p(T_w - T_{sat})/\mathcal{L}$. Moreover the solution is sensitive to parameters that are involved in Eq. (42). Several sensitivity simulations on parameters of the initial condition can be found in [71].

Simulations of film boiling with `LBM_saclay` are first carried out inside a two-dimensional domain $\Omega = [0, 1.28]^2$ which is discretized with $N_x \times N_y = 1024 \times 1024$ nodes. The space- and time-steps are respectively equal to $\delta x = 1.25 \times 10^{-3}$ and $\delta t = 7.5 \times 10^{-5}$. The D2Q9 lattice is used for all distribution functions $f_i$, $g_i$ and $s_i$. For parameters of Table 2, the value of critical wavelength is $\lambda_c = 2\pi\lambda_s = 0.2738$, with $\lambda_s = 4.358 \times 10^{-2}$. The Jacob number is $Ja = 0.025$, the Prandtl $Pr = 0.2$ and the Grashof number is $Gr = 871.38$.

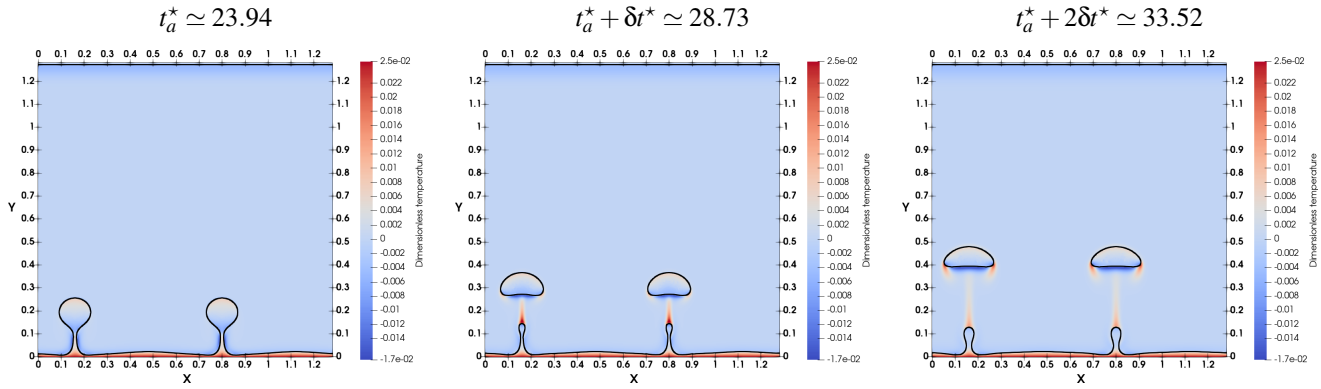
### 5.2. *Simulation of bubble detachment on nodes and antinodes*

We present one simulation for which the interface is initialized by Eq. (42) with $y_0 = 0.03$, $y_1 = 0.015$ and $\lambda = 0.64$. The choice $\lambda = 0.64$ was done after one first preliminary simulation which was performed with $\lambda = 0.32$ ($> \lambda_c = 0.2738$) to check detachment of bubbles. For $\lambda = 0.64$, the maximum value of $y$ is $y_{max} = 0.045$ for two positions $x^{(1)}_{ymax} = 0.16$ and $x^{(2)}_{ymax} = 0.8$. Its minimum value is $y_{min} = 0.015$ for two positions $x^{(1)}_{ymin} = 0.48$ and $x^{(2)}_{ymin} = 1.12$. Positions $x^{(1),(2)}_{ymax}$ are called "nodes" and $x^{(1),(2)}_{ymin}$ are called "anti-nodes". Here, we present one simulation to observe detachment of bubbles alternatively on nodes and anti-nodes. Actually, it is what we observe on Figs. 8a–8c which present the temperature fields and the iso-values $\phi = 1/2$ (black line) at several dimensionless times. The dimensionless time is defined by $t^\star = t/t_s$ where $t_s = \sqrt{\lambda_s/g_y} = 0.1044$. At the early stage of simulation (Fig. 8a), we can observe that the detachment of bubbles occurs on nodes. Later during the simulation (Fig. 8b), the bubbles that are emitted on nodes coalesce on the top on the domain, while two other bubbles grow and are detached from anti-nodes. Finally (Fig. 8c), the cycle is repeated periodically: bubbles emitted at anti-nodes coalesce and new bubbles on nodes detach and rise. Streamlines and velocity magnitude corresponding to the last time $t^\star \simeq 158.03$ are presented on Fig. 8d. In Table 2, the mobility was set to $M_\phi = 1.7 \times 10^{-3}$ after a sensitivity analysis. If $M_\phi$ is too low, the authors have observed the appearance of parasitic bubbles in the liquid phase. The mobility coefficient is directly related to the relaxation time $\tau_g$ and the algorithm can be unstable if its value is too low. It is expected that a wider range of parameter $M_\phi$ could be reached with the TRT or MRT collisions operators.

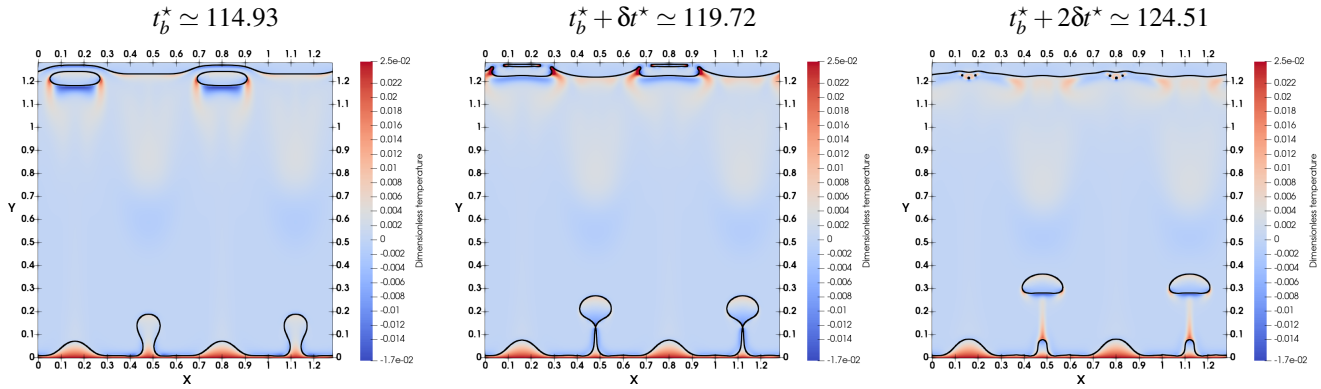
### 5.3. *Computational times*

A first comparison of computational times between GPUs and CPUs has been indicated in Section 3.5, but only for a diffusive problem. For a single- and double-Poiseuille flow of Section 4.1, the computational times on a $100 \times 100$ lattice are respectively 56 MLUPS and 38 MLUPS. Those computations have been performed on a computer equipped of one AMD CPU processor (Ryzen 5 2600, 3.4GHz with 12 threads). The MLUPS are higher for the single-phase because the algorithm requires much less floating points computations. There is neither Allen-Cahn equation nor intermediate gradient to update for a single-phase flow. However, let us note that the MLUPS for two distribution functions are higher than half of the value obtained with only one (i.e. 28 MLUPS), which indicates a good code optimization by resolving the phase-field equation.

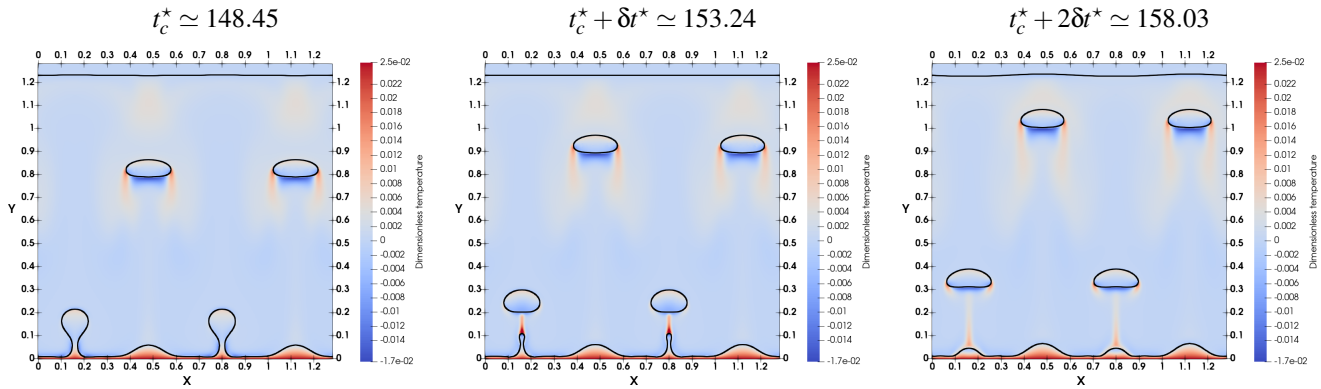
Simulation of diffusion or single-phase flow requires only one distribution and the double-Poiseuille flow requires two distribution functions. The film boiling simulation requires three Lattice Boltzmann equations with three distribution functions and the computation of additional gradients. In that case, to complete $5.33 \times 10^5$ time iterations on a computational domain of $1024^2$ nodes, the simulation took 1h56m (80.96 MLUPS) on a single GPU NVIDIA® K80. The same simulation took 12h57m (11.97 MLUPS) on 16-cores Intel® Xeon® CPU E5-2630 v3 2.40GHz. The computation on GPU is quicker than on CPU as expected after the preliminary diffusion simulation of section 3.5. The ratio is 6.7 times in favor of GPU compared to CPU. Next, the full grid ($1024^2$ nodes) is decomposed in four sub-domains composed of $256 \times 1024$ nodes, each one of them being taken in charge by one GPU. The simulation took 38 minutes (249.99 MLUPS) to perform the same number of time iterations on four parallel GPUs. The computational time is divided by a factor three compared to a single GPU. Finally, the computational domain is increased to $\Omega = [0, 5.12] \times [0, 3.84]$ and discretized by $N_x \times N_y = 4096 \times 3072$ nodes, i.e. the mesh size is twelve times bigger than the previous one. The initial condition is slightly modified to



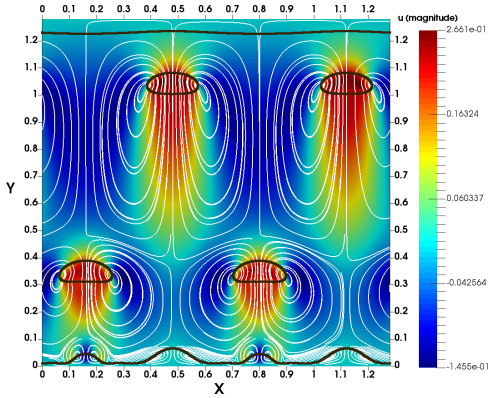

(a) Detachment of bubbles occurs on nodes at the early stage of the simulation.




(b) Coalescence is observed at the top of the domain for bubbles detached from nodes. It is also observed a detachment of bubbles at anti-nodes.

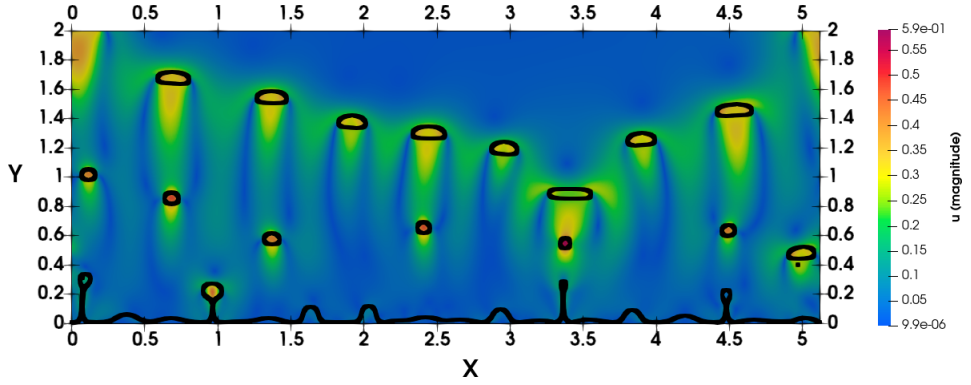



(c) Later during the simulation, bubbles are detached on nodes, the cycle is pursued periodically.

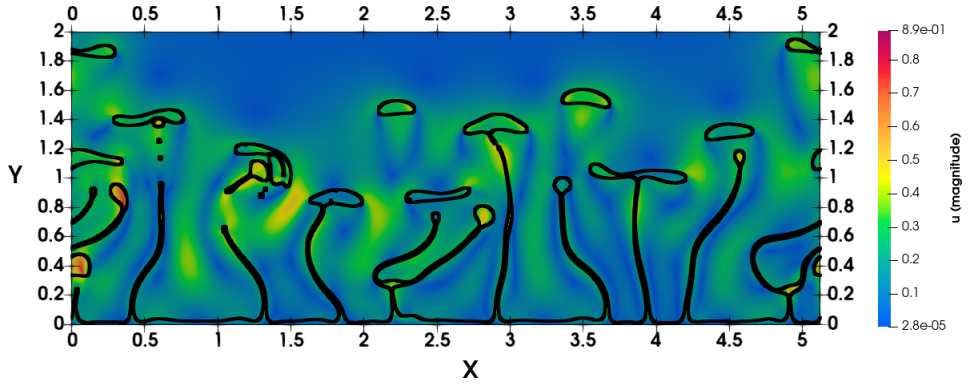



(d) Streamlines (white lines) and interface $\phi = 1/2$ (black lines) superimposed on the velocity magnitude (colored field) at $t^\star \simeq 158.03$.

Figure 8: Simulation of film boiling for $Ja = 0.025$. Interface position $\phi = 1/2$ superimposed on temperature field and for several dimensionless times of simulation. Three successive times from (a) $t_a^\star \simeq 23.94$, (b) $t_b^\star \simeq 114.93$ and (c) $t_c^\star \simeq 148.45$ with $\delta t^\star = 4.79$.




(a) $Ja = 0.025$.




(b) $Ja = 0.1$.

Figure 9: Velocity magnitude (colored field) and interface position $\phi = 1/2$ (black lines) at $t^\star = 95.78$ for (a) $Ja = 0.025$ and (b) $Ja = 0.1$.

$$y = y_0 + y_1 \sum_{i=1}^{16} \sin\left(\frac{2\pi x}{\lambda_i}\right) \tag{44}$$

where the interface position $y$ is perturbed with several modes $\lambda_i$ which are randomly picked, uniformly distributed between $0.5\lambda_c \leq \lambda_i \leq 1.5\sqrt{3}\lambda_c$. We simulate two values of wall temperature $\theta_w = 0.025$ and $\theta_w = 0.1$ corresponding to Jacob numbers respectively equal to $Ja = 0.025$ and $Ja = 0.1$. All other values of physical parameters remain identical (Table 2). A comparison on shapes of bubbles is given at $t^\star = 95.78$ on Fig. 9. When the Jacob number has the value of Section 5.2, discrete bubbles are released periodically from the initial condition (Fig. 9a). When the Jacob number is increased to 0.1, long vapor jets are observed below bubbles (Fig. 9b). That observation is consistent with those simulated with other techniques and even observed on experiments cited in [39, Sec 5.1.2 and Fig. 9]. The simulation took 80 minutes (713 MLUPS) on 8 parallel GPUs to complete $5.33 \times 10^5$ time iterations.

## 6. Conclusion

In this paper, the LBM implementation of two-phase flows was revisited by improving two main points. The first one focuses on the model formulation of phase change and the second one focuses on the portability of the code on various platforms. The interface is tracked by the conservative Allen-Cahn model with a source term involving a mass production rate at the interface. In this work, that source term is simplified compared to approaches of literature, and the approximation avoids to calculate the gradients of temperature numerically. The model is able to simulate two phases of different thermal diffusivities with an interface temperature which is not necessarily at saturation. The phase-field model is coupled with the incompressible Navier-Stokes model where a source term was added in the mass balance equation. The source term is defined as the product of mass production rate times one term inversely proportional to densities. An additional equation on temperature completes the model. The time derivative of phase-field appears in the source term of that equation. It is interpreted as the release or absorption of latent heat at the interface.

The Lattice Boltzmann schemes for all equations are implemented in a new C++ code coupled with the `Kokkos` library for its performance portability. The new code, called `LBM_saclay`, can be run with good performance on several architectures such as Graphical Process Units (GPUs), Central Process Units (CPUs) and even multi-GPUs and multi-CPUs. Indeed, two levels of parallelism are developed inside the code. The first one uses `Kokkos` for intra-node parallelism, whereas `MPI` takes in charge the domain decomposition. Preliminary comparisons between GPUs and CPUs were carried out on a simple diffusive problem. As expected from literature, those tests show clearly that best performance is obtained with GPU compared to CPU

(Skylake or KNL) even for best optimization of LBM kernels (CSoA2) which has been developed for Intel Skylake. Here, comparisons were performed with the same C++ source code. No low-level language (`CUDA` or `OpenCL`) was used for GPUs.

Numerical implementation was checked with several test cases to validate step-by-step the full model of fluid flows with phase change. The conservative Allen-Cahn equation is validated with two test cases: (i) Zalesak's slotted disk and (ii) interface deformation inside a vortex. The coupling with Navier-Stokes equations is also checked with two test cases: the layered Poiseuille flow and Laplace law. Next, the coupling between equations of phase-field and temperature were compared to the most general one-dimensional analytical solution of the Stefan problem. Comparisons were done first by assuming identical thermal diffusivities, and next by using various ratios of diffusivities with an interface temperature that is different of the saturation one. The full model was simulated on the test case of film boiling on one GPU and one multicore CPU for two mesh sizes. Computational times are clearly in favor of GPUs. Finally, the film boiling problem is simulated with 8 parallel GPUs for mesh size that is twelve times bigger than the previous one.

In this paper, foundations have been laid for improving performance of lattice Boltzmann simulations in a context of quick evolution of HPC platforms. In the future, a three-dimensional extension of the coupling terms is planned. Next `LBM_saclay` could be enriched with other models requiring interface tracking such as crystal growth and demixing of ternary fluids. Besides, the range of physical parameters could be increased and the code stability could be enhanced by using alternative collision operators such as those based on the Two-Relaxation-Times and Multiple-Relaxation-Times.


### Acknowledgments

We would like to thank MATHIS PLAPP for the insightful discussions on theoretical aspects of phase-field models.


### AppendixA. Removal of the driven-curvature interface motion in Eq. (5)

In this Appendix, the derivation of first term in the right-hand side of Eq. (5) is reminded. The advection of phase index $\phi$ writes

$$\frac{\partial \phi}{\partial t} + \mathbf{V}\cdot\boldsymbol{\nabla}\phi = 0. \tag{A.1}$$

If the total velocity $\mathbf{V}$ is defined as the sum of an external advective velocity $\mathbf{u}$ of an incompressible fluid plus a normal velocity of the interface $v_n\mathbf{n}$, then $\mathbf{V}\cdot\boldsymbol{\nabla}\phi = \mathbf{u}\cdot\boldsymbol{\nabla}\phi + v_n\left|\boldsymbol{\nabla}\phi\right|$. For the second term, we have used the definition of normal vector $\mathbf{n} = \boldsymbol{\nabla}\phi/\left|\boldsymbol{\nabla}\phi\right|$. If the normal velocity $v_n$ is also assumed to be separated into one term, $-M_\phi\kappa$, depending on the curvature $\kappa$ and another one, $\tilde{v}$ independent on $\kappa$ then: $v_n\left|\boldsymbol{\nabla}\phi\right| = -M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right| + \tilde{v}\left|\boldsymbol{\nabla}\phi\right|$ and Eq. (A.1) writes:

$$\frac{\partial \phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right| - \tilde{v}\left|\boldsymbol{\nabla}\phi\right|. \tag{A.2}$$

For solidification problems, $\tilde{v}$ is the coupling with temperature equation and ensures that the Gibbs-Thomson condition is well recovered. A discussion on $\tilde{v}$ is presented at the end of this appendix. The next stage of the derivation is to cancel the driven-curvature interface motion $M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right|$, without setting $M_\phi = 0$, but by adding a supplementary counter term: $M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right| - M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right| = \mathcal{S}(\phi)$. The purpose is to transform an hyperbolic-type PDE into a parabolic-type PDE by expanding $\kappa$ in the first term with its definition $\kappa = \boldsymbol{\nabla}\cdot\mathbf{n} = \boldsymbol{\nabla}\cdot(\boldsymbol{\nabla}\phi/\left|\boldsymbol{\nabla}\phi\right|)$ in order to obtain an expression involving the laplacian of $\phi$:

$$\mathcal{S}(\phi) = M_\phi\left[\boldsymbol{\nabla}^2\phi - \frac{\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}\left|\boldsymbol{\nabla}\phi\right|}{\left|\boldsymbol{\nabla}\phi\right|}\right] - M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right|. \tag{A.3}$$

The main advantage of this formulation (Eq. (A.3)) is that, for a plane interface, i.e. $\kappa = 0$, the equilibrium solution of $\mathcal{S}(\phi) = 0$ is an hyperbolic tangent. By using the definition of $\mathbf{n}$, Eq. (A.3) becomes $\mathcal{S}(\phi) = M_\phi\left[\boldsymbol{\nabla}^2\phi - \mathbf{n}\cdot\boldsymbol{\nabla}\left|\boldsymbol{\nabla}\phi\right|\right] - M_\phi\left|\boldsymbol{\nabla}\phi\right|\boldsymbol{\nabla}\cdot\mathbf{n}$, i.e. Eq. (A.2) becomes

$$\frac{\partial \phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = M_\phi\left[\boldsymbol{\nabla}^2\phi - \mathbf{n}\cdot\boldsymbol{\nabla}\left|\boldsymbol{\nabla}\phi\right| - \left|\boldsymbol{\nabla}\phi\right|\boldsymbol{\nabla}\cdot\mathbf{n}\right] - \tilde{v}\left|\boldsymbol{\nabla}\phi\right| \tag{A.4}$$

which, after the straightforward manipulation $-\mathbf{n}\cdot\boldsymbol{\nabla}\left|\boldsymbol{\nabla}\phi\right| - \left|\boldsymbol{\nabla}\phi\right|\boldsymbol{\nabla}\cdot\mathbf{n} = -\boldsymbol{\nabla}\cdot(\left|\boldsymbol{\nabla}\phi\right|\mathbf{n})$ yields

$$\frac{\partial \phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = \boldsymbol{\nabla}\cdot\left[M_\phi(\boldsymbol{\nabla}\phi - \left|\boldsymbol{\nabla}\phi\right|\mathbf{n})\right] - \tilde{v}\left|\boldsymbol{\nabla}\phi\right|. \tag{A.5}$$

For calculating $\left|\boldsymbol{\nabla}\phi\right|$, the following kernel function is used

$$\phi = \frac{1}{2}\left[1 + \tanh\left(\frac{\zeta}{aW}\right)\right], \tag{A.6}$$

where $\zeta$ is the normal coordinate of the interface, $a$ controls the slope of the hyperbolic tangent and $W$ is the interface width. The above kernel function ensures an hyperbolic tangent profile at equilibrium. It is consistent with the profile obtained in

a thermodynamically derived phase-field model, such as the one used for computation of chemical potential (Eq. (14)) with bulk phases $\phi = 0$ and $\phi = 1$. The normal derivative of Eq. (A.6) leads to

$$\left|\boldsymbol{\nabla}\phi\right| = \frac{\partial\phi}{\partial\zeta} = \frac{2}{aW}\phi(1-\phi). \tag{A.7}$$

Finally by setting $a = 1/2$ the conservative Allen-Cahn equation with a source term is

$$\frac{\partial\phi}{\partial t} + \boldsymbol{\nabla}\cdot(\mathbf{u}\phi) = \boldsymbol{\nabla}\cdot\left[M_\phi\left(\boldsymbol{\nabla}\phi - \frac{4}{W}\phi(1-\phi)\mathbf{n}\right)\right] - \tilde{v}\frac{4}{W}\phi(1-\phi). \tag{A.8}$$

Eq. (A.8) is the Allen-Cahn equation for which the curvature-driven displacement of the interface has been canceled with a counter term. Let us notice that, if $\tilde{v}$ is chosen such as $\tilde{v} = \alpha(\theta_I - \theta)/(\mathscr{A}W)$ then $\tilde{v}\left|\boldsymbol{\nabla}\phi\right| \approx -(4\alpha/\mathscr{A}W^2)(\theta_I - \theta)\phi(1-\phi)$ can be used in Eq. (A.8) for the problem of phase change. The release or absorption of latent heat at the interface is taken into account in the temperature equation by the time derivative of $\phi$. If the physical problem necessitates a curvature-driven interface motion, the curvature term must be kept in the Allen-Cahn equation and then only the first term in the right-hand side of Eq. (A.3) appears in the derivation. With $a = 1/2$, the term $\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}\left|\boldsymbol{\nabla}\phi\right|/\left|\boldsymbol{\nabla}\phi\right|$ is equal to

$$\frac{\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}\left|\boldsymbol{\nabla}\phi\right|}{\left|\boldsymbol{\nabla}\phi\right|} = \frac{\partial^2\phi}{\partial\zeta^2} = \frac{16}{W^2}\phi(1-\phi)(1-2\phi) \tag{A.9}$$

The curvatuve-driven term writes

$$M_\phi\kappa\left|\boldsymbol{\nabla}\phi\right| = M_\phi\left[\boldsymbol{\nabla}^2\phi - \frac{16}{W^2}\phi(1-\phi)(1-2\phi)\right]. \tag{A.10}$$

## AppendixB. Numerical value of coefficient $\mathscr{A}$

When the matched asymptotic expansions are carried out on the one-dimensional phase-field model, the coefficient $\mathscr{A}$ is defined by four integrals $\mathscr{I}$, $\mathcal{J}$, $\mathcal{G}$ and $\mathcal{U}$ by (e.g. [49, Eq. (59)]):

$$\mathscr{A} = \frac{\mathcal{G} + \mathcal{J}\,\mathcal{U}}{2\mathscr{I}}, \tag{B.1}$$

with

$$\mathscr{I} = \int_{-\infty}^{\infty} d\zeta(\partial_\zeta\phi_0)^2, \quad \mathcal{J} = \int_{-\infty}^{\infty} d\zeta(\partial_\zeta\phi_0)p_\phi^0, \quad \mathcal{G} = \int_{-\infty}^{\infty} d\zeta(\partial_\zeta\phi_0)p_\phi^0\int_0^\zeta d\zeta' h^0, \quad \text{and} \quad \mathcal{U} = \int_{-\infty}^{0} d\zeta h^0. \tag{B.2}$$

In Eq. (B.2), the functions $\phi_0$, $p_\phi^0$ and $h^0$ of our model are defined such as

$$\phi_0 = \frac{1}{2}\left[1 + \tanh\left(\frac{2\zeta}{W}\right)\right], \qquad p_\phi^0 = \phi_0(1-\phi_0), \qquad \text{and} \qquad h^0 = \phi_0 \tag{B.3}$$

Those integrals can be computed analytically and yield a numerical value provided that the interface width $W$ is set. Here, to be consistent with the rescaling of space and the analysis performed in [49], it is enough to set $W = 2\sqrt{2}$, and the integrals are:

$$\mathscr{I} = \frac{1}{3\sqrt{2}}, \qquad \mathcal{J} = \frac{1}{6}, \qquad \mathcal{G} = -\frac{(12\ln 2 - 10)}{72\sqrt{2}}, \qquad \text{and} \qquad \mathcal{U} = \frac{\ln 2}{\sqrt{2}}. \tag{B.4}$$

Finally Eq. (B.1) yields

$$\mathscr{A} = \frac{10}{48} \cong 0.20833. \tag{B.5}$$

## AppendixC. Discrete lattice Boltzmann equations

In this Appendix, the variable change for the discrete lattice Boltzmann equation is reminded in AppendixC.1. In AppendixC.2, we will show that, for CAC model, the formulation with a source term is equivalent to the formulation with a modification of the equilibrium distribution function.

*AppendixC.1. Variable change for discrete lattice Boltzmann equation*

The discrete lattice Boltzmann equation with an external force or source term $\mathcal{S}_i^{\vartheta}$ can be written with the BGK collision term:

$$\frac{\partial \vartheta_i}{\partial t} + \mathbf{c}_i \cdot \boldsymbol{\nabla} \vartheta_i = -\frac{\vartheta_i - \vartheta_i^{eq}}{\tau_\vartheta} + \mathcal{S}_i^{\vartheta}. \tag{C.1}$$

In what follows, the calculations will be performed by setting $\vartheta \equiv f$, $\mathcal{S}_i^{\vartheta} = \mathcal{S}_i^{f} = \mathcal{S}_i$ and $\tau_\vartheta \equiv \tau$ but the variable change derivation holds also for $\vartheta \equiv h$ and $\vartheta \equiv s$. Terms that are evaluated at position $\mathbf{x}$ and time $t$ are noted $f_i \equiv f_i(\mathbf{x}, t)$, $f_i^{eq} \equiv f_i^{eq}(\mathbf{x}, t)$ and $\mathcal{S}_i \equiv \mathcal{S}_i(\mathbf{x}, t)$, whereas terms evaluated at position $\mathbf{x} + \mathbf{c}_i \delta t$ and time $t + \delta t$ are noted with a star: $f_i^\star \equiv f_i(\mathbf{x} + \mathbf{c}_i \delta t, t + \delta t)$, $f_i^{\star eq} \equiv f_i^{eq}(\mathbf{x} + \mathbf{c}_i \delta t, t + \delta t)$ and $\mathcal{S}_i^\star \equiv \mathcal{S}_i(\mathbf{x} + \mathbf{c}_i \delta t, t + \delta t)$. With those notations, integration of Eq. (C.1) over $t$ and $t + \delta t$ yields:

$$f_i^\star = f_i - \frac{\delta t}{2\tau}\left(f_i^\star - f_i^{\star eq}\right) - \frac{\delta t}{2\tau}\left(f_i - f_i^{eq}\right) + \frac{\delta t}{2}\mathcal{S}_i^\star + \frac{\delta t}{2}\mathcal{S}_i \tag{C.2}$$

where the trapezoidal rule was applied for the right-hand side of Eq. (C.1). In this expression, the natural variable change for implicit terms is

$$\overline{f}_i^\star = f_i^\star + \frac{\delta t}{2\tau}\left(f_i^\star - f_i^{\star eq}\right) - \frac{\delta t}{2}\mathcal{S}_i^\star. \tag{C.3}$$

The same variable change is used for $\overline{f}_i$:

$$\overline{f}_i = f_i + \frac{\delta t}{2\tau}\left(f_i - f_i^{eq}\right) - \frac{\delta t}{2}\mathcal{S}_i. \tag{C.4}$$

By inverting the latter relation in order to express $f_i$ with respect to $\overline{f}_i$ , we obtain:

$$f_i = \frac{2\tau}{2\tau + \delta t}\left(\overline{f}_i + \frac{\delta t}{2\tau} f_i^{eq} + \frac{\delta t}{2}\mathcal{S}_i\right). \tag{C.5}$$

With Eqs. (C.3) and (C.5), Eq. (C.2) becomes

$$\overline{f}_i^\star = \overline{f}_i - \frac{\delta t}{\tau + \delta t/2}\left(\overline{f}_i - f_i^{eq} + \frac{\delta t}{2}\mathcal{S}_i\right) + \delta t \mathcal{S}_i \tag{C.6}$$

At this stage, if we define a new variable change

$$\overline{f}_i^{eq} = f_i^{eq} - \frac{\delta t}{2}\mathcal{S}_i, \tag{C.7}$$

then Eq. (C.6) is equivalent to

$$\overline{f}_i^\star = \overline{f}_i - \frac{\delta t}{\tau + \delta t/2}\left(\overline{f}_i - \overline{f}_i^{eq}\right) + \delta t \mathcal{S}_i. \tag{C.8}$$

Without using the previous variable change for $f_i^{eq}$, Eq. (C.6) is equivalent to

$$\overline{f}_i^\star = \overline{f}_i - \frac{\delta t}{\tau + \delta t/2}\left(\overline{f}_i - f_i^{eq}\right) + \frac{\tau \delta t}{\tau + \delta t/2}\mathcal{S}_i, \tag{C.9}$$

where only the factor in front of the source term is modified.

By introducing the dimensionless collision rate which is defined by $\overline{\tau} = \tau/\delta t$, Eq. (C.8) finally writes

$$\overline{f}_i^\star = \overline{f}_i - \frac{1}{\overline{\tau} + 1/2}\left(\overline{f}_i - \overline{f}_i^{eq}\right) + \delta t \mathcal{S}_i, \tag{C.10}$$

or alternatively,

$$\overline{f}_i^\star = \overline{f}_i - \frac{1}{\overline{\tau} + 1/2}\left(\overline{f}_i - f_i^{eq}\right) + \frac{\overline{\tau}\delta t}{\overline{\tau} + 1/2}\mathcal{S}_i. \tag{C.11}$$

In Section 3, Eq. (C.10) is the starting point for each lattice Boltzmann equation. The variable change Eq. (C.4) leads to the calculation of the zeroth-order moment:

$$\mathcal{M}_0 = \sum_i \overline{f}_i + \frac{\delta t}{2}\sum_i \mathcal{S}_i \tag{C.12}$$

*AppendixC.2. Equivalence of lattice Boltzmann formulations for the Allen-Cahn equation*

The purpose of this Appendix is to prove the equivalence between the source term and the modification of the equilibrium distribution function. The lattice Boltzmann scheme for the conservative Allen-Cahn equation is (Eq. (29a) with $\mathcal{S}_i^g$ defined by Eq. (30a)):

$$\overline{g}_i^\star = \overline{g}_i - \frac{1}{\overline{\tau}_g + 1/2}\left[\overline{g}_i - \overline{g}_i^{eq}\right] + \mathcal{P}_i^g \delta t + \mathcal{F}_i^g \delta t \tag{C.13}$$

with the mobility coefficient defined by $M_\phi = \overline{\tau}_g c_s^2 \delta t$. By using the definition of $g_i^{eq}$ for $\overline{g}_i^{eq} = \phi\Gamma_i - \delta t \mathcal{P}_i^g/2 - \mathcal{F}_i^g \delta t/2$ and gathering the term $\mathcal{F}_i^g \delta t$ inside the bracket, we obtain

$$\overline{g}_i^\star = \overline{g}_i - \frac{1}{\overline{\tau}_g + 1/2}\left[\overline{g}_i - \phi\Gamma_i - \mathcal{F}_i^g \overline{\tau}_g \delta t + \frac{\delta t}{2}\mathcal{P}_i^g\right] + \mathcal{P}_i^g \delta t \tag{C.14}$$

Next, the collision rate is replaced by its mobility $\overline{\tau}_g = M_\phi/(c_s^2 \delta t)$:

$$\overline{g}_i^\star = \overline{g}_i - \frac{1}{\overline{\tau}_g + 1/2}\left[\overline{g}_i - \phi\Gamma_i - \mathcal{F}_i^g \frac{M_\phi}{c_s^2} + \frac{\delta t}{2}\mathcal{P}_i^g\right] + \mathcal{P}_i^g \delta t \tag{C.15}$$

Finally, if we use the definition of $\mathcal{F}_i^g$ given by Eq. (30b), the Allen-Cahn equilibrium distribution function $g_i^{eq,CAC}$ can be defined by [33]

$$g_i^{eq,CAC} = \phi\Gamma_i + M_\phi \frac{4}{W}\phi(1-\phi) w_i \frac{\mathbf{c}_i \cdot \mathbf{n}}{c_s^2} \tag{C.16}$$

and the alternative lattice Boltzmann equation is

$$\overline{g}_i^\star = \overline{g}_i - \frac{1}{\overline{\tau}_g + 1/2}\left[\overline{g}_i - \overline{g}_i^{eq,CAC}\right] + \mathcal{P}_i^g \delta t \tag{C.17}$$

with $\overline{g}_i^{eq,CAC} = g_i^{eq,CAC} - \mathcal{P}_i^g \delta t/2$ with $\mathcal{P}_i^g$ defined by Eq. (30b).

## Bibliography


[1] T. Krueger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, E. Viggen, The Lattice Boltzmann Method: Principles and Practice, Graduate Texts in Physics, Springer, 2016.

[2] Z. Guo, C. Shu, Lattice Boltzmann Method and Its Applications in Engineering, WORLD SCIENTIFIC, 2013. doi:10.1142/8806.

[3] W. Li, X. Wei, A. Kaufman, Implementing lattice Boltzmann computation on graphics hardware, The Visual Computer 19 (7) (2003) 444–456. doi:10.1007/s00371-003-0210-6.

[4] J. Tölke, Implementation of a Lattice Boltzmann kernel using the Compute Unified Device Architecture developed by nVIDIA, Computing and Visualization in Science 13 (1) (2008) 29. doi:10.1007/s00791-008-0120-2.

[5] F. Kuznik, C. Obrecht, G. Rusaouen, J.-J. Roux, LBM based flow simulation using GPU computing processor, Computers & Mathematics with Applications 59 (7) (2010) 2380 – 2392. doi:10.1016/j.camwa.2009.08.052.

[6] H. Zhou, G. Mo, F. Wu, J. Zhao, M. Rui, K. Cen, GPU implementation of lattice Boltzmann method for flows with curved boundaries, Computer Methods in Applied Mechanics and Engineering 225-228 (2012) 65 – 73. doi:10.1016/j.cma.2012.03.011.

[7] C. Obrecht, F. Kuznik, B. Tourancheau, J.-J. Roux, Multi-GPU implementation of the lattice Boltzmann method, Computers & Mathematics with Applications 65 (2) (2013) 252 – 261, special Issue on Mesoscopic Methods in Engineering and Science (ICMMES-2010, Edmonton, Canada). doi:10.1016/j.camwa.2011.02.020.

[8] M. Januszewski, M. Kostur, Sailfish: A flexible multi-gpu implementation of the lattice boltzmann method, Computer Physics Communications 185 (9) (2014) 2350 – 2368. doi:10.1016/j.cpc.2014.04.018.

[9] H. C. Edwards, C. R. Trott, D. Sunderland, Kokkos: Enabling manycore performance portability through polymorphic memory access patterns, Journal of Parallel and Distributed Computing 74 (12) (2014) 3202 – 3216. doi:10.1016/j.jpdc.2014.07.003.

[10] J. Eichstädt, M. Green, M. Turner, J. Peiró, D. Moxey, Accelerating high-order mesh optimisation with an architecture-independent programming model, Computer Physics Communications 229 (2018) 36 – 53. doi:10.1016/j.cpc.2018.03.025.

[11] H. Huang, M. Sukop, X.-Y. Lu, Multiphase Lattice Boltzmann Methods. Theory and Application, Wiley & Sons, 2015.

[12] Q. Li, K. Luo, Q. Kang, Y. He, Q. Chen, Q. Liu, Lattice Boltzmann methods for multiphase flow and phase-change heat transfer, Progress in Energy and Combustion Science 52 (2016) 62 – 105. doi:10.1016/j.pecs.2015.10.001.

[13] A. K. Gunstensen, D. H. Rothman, S. Zaleski, G. Zanetti, Lattice Boltzmann model of immiscible fluids, Phys. Rev. A 43 (1991) 4320–4327. doi:10.1103/PhysRevA.43.4320.

[14] X. Shan, H. Chen, Lattice Boltzmann model for simulating flows with multiple phases and components, Phys. Rev. E 47 (1993) 1815–1819. doi:10.1103/PhysRevE.47.1815.

[15] X. Shan, H. Chen, Simulation of nonideal gases and liquid-gas phase transitions by the lattice Boltzmann equation, Phys. Rev. E 49 (1994) 2941–2948. doi:10.1103/PhysRevE.49.2941.

[16] M. R. Swift, E. Orlandini, W. R. Osborn, J. M. Yeomans, Lattice boltzmann simulations of liquid-gas and binary fluid systems, Phys. Rev. E 54 (1996) 5041–5052. doi:10.1103/PhysRevE.54.5041.

[17] D. Jacqmin, Calculation of Two-Phase Navier-Stokes Flows Using Phase-Field Modeling, Journal of Computational Physics 155 (1999) pp. 96–127. doi:10.1006/jcph.1999.6332.

[18] P. Yuan, L. Schaefer, Equations of state in a lattice Boltzmann model, Physics of Fluids 18 (2006) 042101. doi:10.1063/1.2187070.

[19] X. Li, P. Cheng, Lattice Boltzmann simulations for transition from dropwise to filmwise condensation on hydrophobic surfaces with hydrophilic spots, International Journal of Heat and Mass Transfer 110 (2017) 710 – 722. doi:10.1016/j.ijheatmasstransfer.2017.03.033.

[20] M. Li, C. Huber, Y. Mu, W. Tao, Lattice Boltzmann simulation of condensation in the presence of noncondensable gas, International Journal of Heat and Mass Transfer 109 (2017) 1004 – 1013. doi:10.1016/j.ijheatmasstransfer.2017.02.046.

[21] S. Leclaire, M. Reggio, J.-Y. Trépanier, Numerical evaluation of two recoloring operators for an immiscible two-phase flow lattice boltzmann model, Applied Mathematical Modelling 36 (5) (2012) 2237 – 2252. doi:10.1016/j.apm.2011.08.027.

[22] S. Leclaire, N. Pellerin, M. Reggio, J.-Y. Trépanier, Enhanced equilibrium distribution functions for simulating immiscible multiphase flows with variable density ratios in a class of lattice boltzmann models, International Journal of Multiphase Flow 57 (2013) 159 – 168. doi:10.1016/j.ijmultiphaseflow.2013.07.001.

[23] D. Anderson, G. McFadden, A. Wheeler, Diffuse-interface methods in fluid mechanics, Annual Reviews of Fluid Mechanics 30 (1998) pp. 139–165.

[24] J. Cahn, J. Hilliard, Free Energy of a Nonuniform System. I. Interfacial Free Energy, Journal of Chemical Physics 28 (2) (1958) pp. 258–267.

[25] D. Jasnow, J. Viñals, Coarse-grained description of thermo-capillary flow, Physics of Fluids 8 (3) (1996) pp. 660–669. doi:10.1063/1.868851.

[26] V. Kendon, M. Cates, I. Pagonabarraga, J.-C. Desplat, P. Bladon, Inertial effects in three-dimensional spinodal decomposition of a symmetric binary fluid mixture: a lattice Boltzmann study, Journal of Fluid Mechanics 440 (2001) pp. 147–203. doi:10.1017/S0022112001004682.

[27] H. Zheng, C. Shu, Y. Chew, A lattice boltzmann model for multiphase flows with large density ratio, Journal of Computational Physics 218 (2006) pp. 353–371. doi:10.1016/j.jcp.2006.02.015.

[28] T. Lee, L. Liu, Lattice Boltzmann simulations of micron-scale drop impact on dry surfaces, Journal of Computational Physics 229 (2010) 8045–8063. doi:10.1016/j.jcp.2010.07.007.

[29] Y. Q. Zu, S. He, Phase-field-based lattice Boltzmann model for incompressible binary fluid systems with density and viscosity contrasts, Phys. Rev. E 87 (2013) 043301. doi:10.1103/PhysRevE.87.043301.

[30] Y. Sun, C. Beckermann, Sharp interface tracking using the phase-field equation, Journal of Computational Physics 220 (2) (2007) 626 – 653. doi:10.1016/j.jcp.2006.05.025.

[31] P.-H. Chiu, Y.-T. Lin, A conservative phase field method for solving incompressible two-phase flows, Journal of Computational Physics 230 (1) (2011) 185 – 204. doi:10.1016/j.jcp.2010.09.021.

[32] M. Geier, A. Fakhari, T. Lee, Conservative phase-field lattice Boltzmann model for interface tracking equation, Phys. Rev. E 91 (2015) 063309. doi:10.1103/PhysRevE.91.063309.

[33] A. Fakhari, D. Bolster, L.-S. Luo, A weighted multiple-relaxation-time lattice Boltzmann method for multiphase flows and its application to partial coalescence cascades, Journal of Computational Physics 341 (2017) 22 – 43. doi:10.1016/j.jcp.2017.03.062.

[34] T. Mitchell, C. Leonardi, A. Fakhari, Development of a three-dimensional phase-field lattice Boltzmann method for the study of immiscible fluids at high density ratios, International Journal of Multiphase Flow 107 (2018) 1 – 15. doi:10.1016/j.ijmultiphaseflow.2018.05.004.

[35] D. Lee, J. Kim, Comparison study of the conservative Allen–Cahn and the Cahn–Hilliard equations, Mathematics and Computers in Simulation 119 (2016) 35 – 56. doi:10.1016/j.matcom.2015.08.018.

[36] H. L. Wang, Z. H. Chai, B. C. Shi, H. Liang, Comparative study of the lattice Boltzmann models for Allen-Cahn and Cahn-Hilliard equations, Phys. Rev. E 94 (2016) 033304. doi:10.1103/PhysRevE.94.033304.

[37] Z. Dong, W. Li, Y. Song, Lattice Boltzmann Simulation of Growth and Deformation for a Rising Vapor Bubble Through Superheated Liquid, Numerical Heat Transfer, Part A: Applications 55 (4) (2009) 381–400. doi:10.1080/10407780902720718.

[38] H. Safari, M. H. Rahimian, M. Krafczyk, Extended lattice Boltzmann method for numerical simulation of thermal phase change in two-phase fluid flow, Phys. Rev. E 88 (2013) 013304. doi:10.1103/PhysRevE.88.013304.

[39] C. R. Kharangate, I. Mudawar, Review of computational studies on boiling and condensation, International Journal of Heat and Mass Transfer 108 (2017) 1164 – 1196. doi:10.1016/j.ijheatmasstransfer.2016.12.065.

[40] R. Folch, J. Casademunt, A. Hernández-Machado, L. Ramírez-Piscina, Phase-field model for Hele-Shaw flows with arbitrary viscosity contrast. I. Theoretical approach, Phys. Rev. E 60 (1999) 1724–1733. doi:10.1103/PhysRevE.60.1724.

[41] D. Jamet, C. Misbah, Thermodynamically consistent picture of the phase-field model of vesicles: Elimination of the surface tension, Phys. Rev. E 78 (2008) 041903. doi:10.1103/PhysRevE.78.041903.

[42] R. Kobayashi, Modeling and numerical simulations of dendritic crystal growth, Physica D: Nonlinear Phenomena 63 (3) (1993) 410 – 423. doi:10.1016/0167-2789(93)90120-P.

[43] L. Amaya-Bower, T. Lee, Single bubble rising dynamics for moderate Reynolds number using Lattice Boltzmann Method, Computers & Fluids 39 (7) (2010) 1191 – 1207. doi:10.1016/j.compfluid.2010.03.003.

[44] J. Kim, A continuous surface tension force formulation for diffuse-interface models, Journal of Computational Physics 204 (2) (2005) 784 – 804. doi:10.1016/j.jcp.2004.10.032.

[45] J. Delhaye, Jump conditions and entropy sources in two-phase systems. Local instant formulation, International Journal of Multiphase Flow 1 (3) (1974) 395 – 409. doi:10.1016/0301-9322(74)90012-3.

[46] D. Juric, G. Tryggvason, Computations of boiling flows, International Journal of Multiphase Flow 24 (3) (1998) 387–410. doi:10.1016/S0301-9322(97)00050-5.

[47] H. Safari, M. H. Rahimian, M. Krafczyk, Consistent simulation of droplet evaporation based on the phase-field multiphase lattice boltzmann method, Phys. Rev. E 90 (2014) 033305. doi:10.1103/PhysRevE.90.033305.

[48] Amirhosein Begmohammadi, Mohsen Farhadzadeh, Mohammad Hassan Rahimian, Simulation of pool boiling and periodic bubble release at high density ratio using lattice Boltzmann method, International Communications in Heat and Mass Transfer 61 (2015) 78 – 87. doi:10.1016/j.icheatmasstransfer.2014.12.018.

[49] A. Karma, W.-J. Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, Physical Review E 57 (4) (1998) pp. 4323–4349. doi:10.1103/PhysRevE.57.4323.

[50] I. Ginzburg, Equilibrium-type and link-type lattice Boltzmann models for generic advection and anisotropic-dispersion equation, Advances in Water Resources 28 (11) (2005) 1171 – 1195. doi:10.1016/j.advwatres.2005.03.004.

[51] D. D'Humières, Generalized Lattice-Boltzmann Equations, 1992, pp. 450–458, prog. Astronaut. Aeronaut. 59. doi:10.2514/5.9781600866319.0450.0458.

[52] D. d'Humières, I. Ginzburg, M. Krafczyk, P. Lallemand, L.-S. Luo, Multiple-relaxation-time lattice Boltzmann models in three dimensions, Phil. Trans. R. Soc. Lond. A 360 (2002) pp. 437–451. doi:10.1098/rsta.2001.0955.

[53] X. He, S. Chen, G. D. Doolen, A Novel Thermal Model for the Lattice Boltzmann Method in Incompressible Limit, Journal of Computational Physics 146 (1) (1998) 282 – 300. doi:10.1006/jcph.1998.6057.

[54] T. Inamuro, T. Ogata, S. Tajima, N. Konishi, A lattice Boltzmann method for incompressible two-phase flows with large density differences, Journal of Computational Physics 198 (2) (2004) 628 – 644. doi:10.1016/j.jcp.2004.01.019.

[55] A. J. Chorin, A numerical method for solving incompressible viscous flow problems, Journal of Computational Physics 2 (1) (1967) 12 – 26. doi:10.1016/0021-9991(67)90037-X.

[56] X. He, L.-S. Luo, Lattice Boltzmann Model for the Incompressible Navier-Stokes Equation, Journal of Statistical Physics 88 (3/4) (1997) pp. 927–944.

[57] M. Mohammadi-Shad, T. Lee, Phase-field lattice Boltzmann modeling of boiling using a sharp-interface energy solver, Phys. Rev. E 96 (2017) 013306. doi:10.1103/PhysRevE.96.013306.

[58] I. Ginzburg, Generic boundary conditions for lattice Boltzmann models and their application to advection and anisotropic dispersion equations, Advances in Water Resources 28 (11) (2005) 1196 – 1216. doi:10.1016/j.advwatres.2005.03.009.

[59] T. Lee, P. Fischer, Eliminating parasitic currents in the lattice Boltzmann equation method for non ideal gases, Physical Review E 74 (2006) 046709. doi:10.1103/PhysRevE.74.046709.
[60] T. Lee, Effects of incompressibility on the elimination of parasitic currents in the lattice Boltzmann equation method for binary fluids, Computers and Mathematics with Applications 58 (2009) pp. 987–994. doi:10.1016/j.camwa.2009.02.017.
[61] T. Lee, C.-L. Lin, A stable discretization of the lattice boltzmann equation for simulation of incompressible two-phase flows at high density ratio, Journal of Computational Physics 206 (1) (2005) 16 – 47. doi:10.1016/j.jcp.2004.12.001.
[62] A. Fakhari, T. Mitchell, C. Leonardi, D. Bolster, Improved locality of the phase-field lattice-Boltzmann model for immiscible fluids at high density ratios, Phys. Rev. E 96 (2017) 053301. doi:10.1103/PhysRevE.96.053301.
[63] Compatibilities of Kokkos library. Web link accessible on 12 May 2020: https://github.com/kokkos/kokkos/wiki/Compiling.
[64] E. Calore, A. Gabbana, S. F. Schifano, R. Tripiccione, Early Experience on Using Knights Landing Processors for Lattice Boltzmann Applications, in: R. Wyrzykowski, J. Dongarra, E. Deelman, K. Karczewski (Eds.), Parallel Processing and Applied Mathematics, Springer International Publishing, Cham, 2018, pp. 519–530. doi:10.1007/978-3-319-78024-5_45.
[65] S. T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, Journal of Computational Physics 31 (3) (1979) 335 – 362. doi:doi.org/10.1016/0021-9991(79)90051-2.
[66] D. Hahn, M. Özisik, Heat Conduction. Third Edition, Wiley & Sons, 2012.
[67] A. Esmaeeli, G. Tryggvason, A front tracking method for computations of boiling in complex geometries, International Journal of Multiphase Flow 30 (7) (2004) 1037 – 1050, a Collection of Papers in Honor of Professor G. Yadigaroglu on the Occasion of his 65th Birthday. doi:10.1016/j.ijmultiphaseflow.2004.04.008.
[68] V. Pandey, G. Biswas, A. Dalal, Effect of superheat and electric field on saturated film boiling, Physics of Fluids 28 (5) (2016) 052102. doi:10.1063/1.4948545.
[69] A. Begmohammadi, M. Rahimian, M. Farhadzadeh, M. A. Hatani, Numerical simulation of single- and multi-mode film boiling using lattice Boltzmann method, Computers & Mathematics with Applications 71 (9) (2016) 1861 – 1874. doi:10.1016/j.camwa.2016.02.033.
[70] A. Hu, D. Liu, 2D Simulation of boiling heat transfer on the wall with an improved hybrid lattice Boltzmann model, Applied Thermal Engineering 159 (2019) 113788. doi:10.1016/j.applthermaleng.2019.113788.
[71] N. K. Singh, B. Premachandran, Numerical investigation of film boiling on a horizontal wavy wall, International Journal of Heat and Mass Transfer 150 (2020) 119371. doi:10.1016/j.ijheatmasstransfer.2020.119371.