THEORY OF THE LATTICE BOLTZMANN METHOD FOR MULTI-PHASE AND MULTICOMPONENT FLUIDS

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Qun Li

In Partial Fulfilment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Physics

December 2006

Fargo, North Dakota

ABSTRACT

Li, Qun, M.S., Physics, College of Science and Mathematics, North Dakota State University, December 2006. Theory of the Lattice Boltzmann Method for Multi-Phase and Multicomponent Fluids. Major Professor: Dr. Alexander Wagner.

Although the lattice Boltzmann method has been employed to simulate the dynamics of some multicomponent systems, a general lattice Boltzmann algorithm that simulates the dynamics of an arbitrary multicomponent system with explicit thermodynamic consistency is still needed.

In this thesis, I developed a lattice Boltzmann algorithm from a free energy approach that simulated the dynamics of a system with an arbitrary number of components. The thermodynamic properties, such as the chemical potential of each component and the pressure of the overall system, were incorporated in the model. I derived a symmetrical convection diffusion equation which was of the same form for each component. The Navier Stokes equation and continuity equation for the overall system as well as a convection diffusion equation for each component were recovered. The algorithm was verified through simulations of binary and ternary systems in one-dimension. The equilibrium concentrations of components of binary and ternary systems simulated with my algorithm agreed well with the concentrations obtained by minimizing the free energy.

I also studied Galilean invariance violations in lattice Boltzmann methods. The sources of the Galilean invariance violation were analyzed from a unified perspective for both pressure and forcing methods. In this thesis, a measure of Galilean invariance and corrections reducing Galilean invariance violations are presented. I validated my analysis and compared the results of different corrections with simulations in one- and two-phase systems.

ACKNOWLEDGEMENTS

I would like to acknowledge my adviser, Dr. Alexander Wagner, for his valuable advice and guidance throughout this thesis. Sincere thanks are due as well to Dr. Alan R. Denton, Dr. Thomas Ihle, and Dr. Iskander Akhatov, members of my graduate faculty committee, for their thoughtful reading of this thesis. I would express my appreciation to Dr. Richard T. Hammod, who first supported me to study in the Physics Department at the North Dakota State University, to Dr. Daniel Kroll, Dr. Charles Sawicki, Dr. Craig Rottman, and Dr. Terry Pilling for their help. Thanks also to Ben Lu, Eric Foard, Goetz Kaehler, Stephan Loew, the students sharing the same office with me, for all the help in day to day life.

TABLE OF CONTENTS

ABSTRACT . . . iii

ACKNOWLEDGMENTS . . . v

LIST OF TABLES . . . ix

LIST OF FIGURES . . . xi

CHAPTER 1. INTRODUCTION . . . 1

CHAPTER 2. FUNDAMENTAL HYDRODYNAMIC EQUATIONS AND THERMODYNAMICS OF PHASE SEPARATION . . . 6

2.1. Hydrodynamic Equations . . . 6

2.2. Diffusion Processes . . . 7

2.3. Basic Phase Separation Theory . . . 10

2.4. Phase Separation of a System Described by a Landau Free Energy . . . 15

2.5. Phase Separation of a System Described by the Flory-Huggins Free Energy 20

2.5.1. Phase separation of a binary system . . . 21

2.5.2. Phase separation of a ternary system . . . 23

CHAPTER 3. LATTICE BOLTZMANN METHOD . . . 27

3.1. Single Relaxation Time Lattice Boltzmann . . . 27

3.1.1. One-dimensional three-velocity lattice Boltzmann model . . . 36

3.1.2. Lattice Boltzmann for a liquid-gas system . . . 38

3.2. Lattice Boltzmann for Multicomponent Systems . . . 42

3.3. Simulations of Multicomponent Systems 52

3.3.1. Simulations of binary systems 52

3.3.2. Simulations of ternary systems 56

CHAPTER 4. GALILEAN INVARIANCE OF THE LATTICE BOLTZMANN METHOD 60

4.1. Lattice Boltzmann Galilean Invariance Violations and Corrections . . . 60

4.2. Galilean Invariance Corrections for Sound Waves 62

4.3. Test of Correction in Liquid-Gas Phase Separation 66

REFERENCES 70

APPENDIX A. FLORY-HUGGINS MODEL OF POLYMER SOLUTION 75

APPENDIX B. GALILEAN INVARIANCE IN CONTINUUM 80

APPENDIX C. ANALYTICAL SOLUTION OF ONE-DIMENSIONAL SOUND WAVE 83

C.1. Sound Wave Equation Without Viscosity 83

C.2. Sound Wave Equation with Viscosity 85

APPENDIX D. THE DERIVATION OF PRESSURE TENSOR 89

APPENDIX E. GALILEAN INVARIANCE EXAMINATIONS 91

APPENDIX F. CODE FOR THEORETICAL SOLUTION OF A BINARY SYSTEM 95

APPENDIX G. CODE FOR THEORETICAL SOLUTION OF A TERNARY SYSTEM 98

APPENDIX H. CODE FOR LIQUID-GAS SYSTEM (HOLDRYCHQ) 105

APPENDIX I. CODE FOR LIQUID-GAS SYSTEM (FORCINGQ) 109

APPENDIX J. CODE FOR BINARY SYSTEM SIMULATION 113

APPENDIX K. CODE FOR TERNARY SYSTEM SIMULATION 118

LIST OF TABLES

Table | Page

3.1 The algorithms for the LB simulation of a non-ideal fluid 35

E.1 Velocity product summation(D2Q7) 91

E.2 Velocity product summation(D2Q9) 92

LIST OF FIGURES

Figure | Page

2.1 The phase separation of a system and the shape of its bulk free energy density function . . . 12

2.2 The local stability of a system and the shape of the free energy density function . . . 15

2.3 The binodal and spinodal points and the shape of the free energy density function of a system . . . 16

2.4 The domains of phase separation with the spinodal and binodal lines . . 17

2.5 The theoretical binodal and spinodal lines for a monomer binary system 23

2.6 Binodal lines of a polymer system represented in a Cartesian and a triangular coordinate . . . 26

3.1 Comparison of binodal points obtained by Holdych and ForcingQ to the theoretical values . . . 40

3.2 The comparison of thermodynamics obtained by HoldychQ to the theoretical values in a liquid-gas system . . . 41

3.3 The comparison of thermodynamics obtained by ForcingQ to the theoretical values in a liquid-gas system . . . 42

3.4 The comparison of the binodal points obtained by simulation to the theoretical solutions in a binary system . . . 54

3.5 The comparison of thermodynamics obtained by simulation to the theoretical values in a binary system . . . 55

3.6 The comparison of binodal points obtained by simulation to the theoretical values in a ternary system . . . 58

3.7 The comparison of thermodynamics obtained by simulation to the theoretical values in a ternary system . . . 59

4.1 The comparison of the Q correction to the theoretical value of a sound wave . . . 64

4.2 The comparison of error of relative error of a sound wave simulation with and without Q correction . . . 65

4.3 Comparison of the relative errors of a sound wave obtained by simulation with and without Q correction vs. u_0 66

4.4 Galilean invariance errors $E(u_0)$ of a liquid-gas system 67

CHAPTER 1

INTRODUCTION

The lattice Boltzmann (LB) method is a mesoscopic lattice simulation method which has been applied fruitfully to many research areas such as turbulence [15, 63], mesoscale blood flow [16], interfacial waves [8], magneto-hydrodynamics [12], and multicomponent systems [51, 65].

Historically, LB was derived from the lattice-gas automata (LGA) [17, 18, 60] in the 1990s , although the two methods are independent [25, 26, 41, 42]. The LGA traces particle movements on a lattice, and can recover the Navier-Stokes equations [18], thus simulating hydrodynamics. LGA is unconditionally stable and are very good to simulate micro-flow with large intrinsic fluctuations. However, LGA exhibits strong Galilean invariance (GI) violations, and they are limited to small Reynolds numbers [35]. To overcome these deficiencies, LB was developed [13, 43].

Instead of tracing the movement of particles, LB traces the evolution of a density distribution function, which depends on position and velocity. The velocity is discretized such that, in one time step, the densities move to the neighboring lattice sites to which their associated velocities point. This movement is called streaming. Between streaming steps, collisions occur at lattice sites and change the density distribution function. Qian et al. introduced Bhatnagar, Gross and Krook (BGK)'s [5] single relaxation time approximation to simplify the description of the collision [41]. The system evolves by means of one streaming and one collision per time step. The macroscopic physical quantities mass and momentum are given by the velocity moments of the density distribution function. Because the standard BGK model describes the collision of ideal gases, the standard LB algorithm can only simulate ideal gas dynamics.

To simulate non-ideal fluids, the attractive or repulsive interaction among molecules, which is referred to as the non-ideal interaction, should be included in the LB model. There are two approaches to incorporate non-ideal interactions. One is to mimic microscopic interaction forces directly, which is often referred to as Shan-Chen's approach [20, 44, 45, 46]; the other is to derive the non-ideal interaction from the total free energy of the system and then incorporate it a posteriori [2, 38, 48, 51], which is often referred to as the free energy approach. By using the free energy approach, the chemical potential of each component is utilized explicitly in the simulation, but in Shan-Chen's approach, it is not. Therefore, a free energy approach for multicomponent simulations is better suited to study the thermodynamics of a multicomponent system. In this thesis, I study the free energy approach only.

Free energy approaches introduce the non-ideal interaction into LB either through a pressure term [38, 48, 49] or a forcing term [44, 53, 55, 57]. Both approaches are equivalent as far as the recovery of the hydrodynamic equations is concerned. They give very similar simulation results but show a slight difference in the stability of the algorithms. The forcing approach, however, leads to non-negligible higher order terms for systems with large density gradients [55]. The analysis of the effect of these higher order terms is beyond the scope of this thesis.

The LB algorithm has been extended to simulate the thermodynamics and hydrodynamics of a multicomponent system. The challenge for the LB simulation of a multicomponent system lies in the fact that momentum conservation is only valid for the overall system but not for each component separately, and therefore diffusion occurs between the components. In addition, for a valid simulation scheme, any representation of the three components should give the same simulation results. That is, the scheme should be symmetric. For a binary system of components A and B with densities ρ^A and ρ^B, the simulation usually traces the evolution of the total density $\rho^A+\rho^B$ and the density difference $\rho^A - \rho^B$ [51]. Although this scheme is successful in the simulation of a binary system [38, 56], its generalization for the LB simulations of systems with an arbitrary number of components is asymmetric. For instance, to simulate a ternary system of components A, B, and C with densities ρ^A, ρ^B and ρ^C, the total density of the system, $\rho^A + \rho^B + \rho^C$, should be traced, and the other two densities to be traced may be chosen as, e.g., ρ^B and $\rho^A - \rho^C$ [29]. This approach is likely to be asymmetric because the three components are treated differently as is the case of Lamura's model [29]. If an LB method is not symmetric, it will lose generality although it may still be adequate for certain applications. In this thesis, I established a manifestly symmetric scheme which is generally valid for a simulation of a system of an arbitrary number of components.

Several papers considering LB simulations of multicomponent systems exist in the literature. Hudong Chen et al. successfully simulated the amphiphilic fluid with the Shan-Chen's approach, by treating the surfactant molecules as dipoles [11]. X. Shan et al. [45] established a multicomponent LB model that gave the convection diffusion equation for each component. While Shan-Chen's approach is manifestly symmetric, it has the disadvantage that thermodynamic properties of the system are not easily accessible. Lamura et al. [29] successfully simulated the amphiphilic ternary system with a free energy approach, but they chose the density parameters in an asymmetric way. Some researchers have discussed LB simulations of the diffusion process, which is an intrinsic character of the dynamics of the multicomponent system, but they did not give any ready recipe for the LB simulation of an arbitrary multicomponent system. For example, B.Deng et al. [14] presented an LB method to simulate the convection diffusion equation with a source term for a one-component system, but they have not discussed the simulation of the convection diffusion process for a multicomponent system. A. Akthakul et al. [1] developed a multicomponent free energy LB model by using Enskog Chapman expansion and applied it to the study of the convection diffusion process of a polymer ternary system. The convection diffusion equation they derived cannot guarantee that the chemical potential of each component would be strictly constant at equilibrium.

My LB model to simulate the dynamics of an arbitrary multicomponent system is a variant of the free energy approach, and the chemical potential of each component is incorporated explicitly. Therefore, the thermodynamic quantities in equilibrium (that is, the overall pressure of the system and the chemical potential of each component) should be constant macroscopically. In this thesis, a symmetric form of the convection and diffusion equation for each component has been derived which guarantees that the pressure and chemical potential will be constant in equilibrium. To test the validity of the model, I simulated phase separation in binary and ternary systems and checked that the systems reached a constant overall pressure and a constant chemical potential for each component in equilibrium. The theoretical value of the volume fraction of each component in equilibrium can be obtained by numerically determining the lowest free energy of the system (Appendix G). The simulation results showed that the volume fraction of each component in equilibrium in LB agreed well with the value predicted by minimizing the free energy. Therefore, my model has been validated, at least in equilibrium. However, in my LB model each component has the same viscosity and mobility. Therefore, an extension of the model to concentration dependent viscosity and mobility is still needed.

I also studied the Galilean invariance (GI) of the LB algorithm. Although every valid physical model should be Galilean invariant, the LB models, either for ideal gases or for non-ideal fluids, are not perfectly Galilean invariant. In some applications, the GI violation errors of LB are negligible, yet in others they cause problems. The source of GI violation in LB for ideal gas models has been analyzed by Y. Qian et al. [40]. The sources of GI violation in LB for free energy based non-ideal fluid LB models have been analyzed, and corrections have been suggested by several groups [21, 22, 31, 57]. In this thesis, the sources of GI violation for an ideal gas and a non-ideal fluid are analyzed from a unified perspective. A formula was presented that can check whether or not an LB model is Galilean invariant and can identify, to the lowest order, the terms that cause the GI violation. LB simulations of sound waves and phase separation are performed in one dimension to validate my analysis and to compare different correction methods. The content of the second part of this thesis has been published [57].

CHAPTER 2

FUNDAMENTAL HYDRODYNAMIC EQUATIONS AND THERMODYNAMICS OF PHASE SEPARATION

2.1. Hydrodynamic Equations

Three fundamental equations govern the hydrodynamic processes in a fluid [30, 28]. The continuity equation describes the conservation of mass:

$$\partial_t \rho + \nabla \cdot \mathbf{J} = 0, \tag{2.1}$$

where t is time, $\mathbf{J}$ is the mass flux which is defined as $\mathbf{J} \equiv \rho\,\mathbf{u}$, ρ is the mass density of the fluid, and $\mathbf{u}$ is the macroscopic velocity of the fluid. The Navier-Stokes equation describes the conservation of momentum:

$$\partial_t(\rho u_\alpha) + \partial_\beta(\rho u_\alpha u_\beta) = -\partial_\beta P_{\alpha\beta} + \partial_\beta \sigma_{\alpha\beta} + \rho F_\alpha, \tag{2.2}$$

where $\sigma_{\alpha\beta}$ is the stress tensor, F_α is the component α of an external force on a unit mass in a unit volume, and the Einstein summation convention is used. For Newtonian fluids, the stress tensor is given by [39]

$$\sigma_{\alpha\beta} = \eta \left(\partial_\beta u_\alpha + \partial_\alpha u_\beta - \frac{d}{2}\delta_{\alpha\beta} \nabla \cdot \mathbf{u} \right) + \mu_B \delta_{\alpha\beta} \nabla \cdot \mathbf{u}, \tag{2.3}$$

where η is the shear viscosity, and μ_B is bulk viscosity; $P_{\alpha\beta}$ is the pressure tensor; and d is the spacial dimension of the system. The energy equation (commonly called the heat equation) describes energy conservation [28]:

$$\rho \partial_t e + \rho u_\alpha \partial_\alpha e = -\partial_\alpha (k \partial_\alpha T) - P_{\alpha\beta}(\partial_\beta u_\alpha) + \sigma_{\alpha\beta}(\partial_\beta u_\alpha), \tag{2.4}$$

where e is the internal energy per unit mass of the fluid and k is the thermal conductivity. For a perfect gas $e = C_V T$, where C_V is the specific heat at constant volume and T is the temperature. However, in this thesis, I only studied isothermal processes during which the system can freely exchange heat energy with an external environment. The heat transfer process was replaced with a constant temperature condition in the simulations. Therefore, the simulations satisfy Eq. (2.1) and Eq. (2.2) but do not satisfy Eq. (2.4).

For a multicomponent system, besides the overall macroscopic fluid flow, diffusion also contributes to the mass transport process. The next section is devoted to the discussion of the diffusion process in a multicomponent system.

2.2. Diffusion Processes

In multicomponent systems, there are two mechanisms for mass transport: convection and diffusion. Convection is the flow of the overall fluid, while diffusion occurs where the average velocities of components are different. The velocity of the overall fluid is a macroscopic quantity because it is conserved, but the average velocities of the components are not. The macroscopic velocity of the fluid $\mathbf{u}$ can be expressed in terms of the density ρ^σ and velocity u^σ of each component in the form of

$$\mathbf{u} \equiv \frac{\sum_\sigma \rho^\sigma \mathbf{u}^\sigma}{\sum_\sigma \rho^\sigma}. \tag{2.5}$$

With the notation

$$\Delta \mathbf{u}^\sigma \equiv \mathbf{u}^\sigma - \mathbf{u}, \tag{2.6}$$

the flux of each component can be divided into a convection part and a diffusion part.

$$\mathbf{J}^\sigma \equiv \rho^\sigma \mathbf{u}^\sigma = \rho^\sigma(\mathbf{u} + \Delta \mathbf{u}^\sigma) = \mathbf{J}^{\sigma c} + \mathbf{J}^{\sigma d}, \tag{2.7}$$

where $\mathbf{J}^{\sigma c}$ is the convection part and $\mathbf{J}^{\sigma d}$ is the diffusion part. Because mass conservation still holds for each component, the continuity equation for each component is valid:

$$\partial_t \rho^\sigma + \nabla \cdot \mathbf{J}^\sigma = 0. \tag{2.8}$$

Substituting Eq. (2.7) into Eq. (2.8), the convection diffusion equation for a component can be obtained.

$$\partial_t \rho^\sigma + \nabla \cdot \mathbf{J}^{\sigma c} = -\nabla \cdot \mathbf{J}^{\sigma d}. \tag{2.9}$$

From Eqs. (2.5) and (2.6), I see that

$$\sum_\sigma \mathbf{J}^{\sigma d} = 0, \tag{2.10}$$

which is a constraint for all diffusion fluxes of the components. I will elaborate further on the diffusion flux of each component. Eq. (2.10) indicates that diffusion is a mixing process that has no contribution to the convection of the fluid, so when two components are mixing with each other, the particles of the two kinds exchange their positions. Therefore, the diffusion process between two components is related to the difference of the chemical potential of the two components, which is also called the exchange chemical potential [23]. Recognizing that the gradient of the exchange chemical potential determines the diffusion processes, I obtain a first order approximation for the diffusion flux of one component into all other components as

$$\mathbf{J}^{\sigma d} = -\sum_{\sigma'} M^{\sigma\sigma'} \nabla(\mu^{\sigma} - \mu^{\sigma'}), \tag{2.11}$$

where σ and σ' enumerate the components; μ^{σ} and $\mu^{\sigma'}$ are the chemical potentials of components σ and σ'; and $M^{\sigma\sigma'}$ is a symmetric positive definite mobility tensor.

A simple model for the diffusion process assumes that a diffusion flux between two components is proportional to the overall density and the concentration of each component. Then $M^{\sigma\sigma'}$ can be expressed as

$$M^{\sigma\sigma'} = k^{\sigma\sigma'} \rho \frac{\rho^{\sigma}}{\rho} \frac{\rho^{\sigma'}}{\rho} = k^{\sigma\sigma'} \frac{\rho^{\sigma} \rho^{\sigma'}}{\rho}, \tag{2.12}$$

where $k^{\sigma\sigma'}$ is the constant diffusion coefficient between components σ and σ'. It depends on components but is independent of the total densities and concentration of each component. Substituting Eq. (2.12) into Eq. (2.11),

$$\mathbf{J}^{\sigma d} = -\sum_{\sigma'} k^{\sigma\sigma'} \frac{\rho^{\sigma}\rho^{\sigma'}}{\rho} \nabla(\mu^{\sigma} - \mu^{\sigma'}). \tag{2.13}$$

Substituting Eq. (2.13) into Eq. (2.9), the general form of a convection diffusion equation is obtained as

$$\partial_t \rho^{\sigma} + \nabla \cdot (\rho^{\sigma}\mathbf{u}) = \nabla \left(\sum_{\sigma'} k^{\sigma\sigma'} \frac{\rho^{\sigma}\rho^{\sigma'}}{\rho} \nabla(\mu^{\sigma} - \mu^{\sigma'}) \right). \tag{2.14}$$

2.3. Basic Phase Separation Theory

Free energy, chemical potential, and pressure are key thermodynamic concepts to understand the phase behavior of a system. The total free energy of a multicomponent system is composed of the bulk part and the interfacial part. In its simplest form it is known as a Landau free energy and can be expressed as

$$\mathcal{F} = \int d\mathbf{r} \left(\psi(n^1, n^2, \cdots, n^s) + \sum_{\sigma=1}^{s} \sum_{\sigma'=\sigma}^{s} \kappa^{\sigma\sigma'} \nabla n^{\sigma} \cdot \nabla n^{\sigma'} \right), \tag{2.15}$$

where ψ is the bulk free energy density, σ and σ' are the superscripts for components; and n^1, n^2, $\cdots$, n^s are the number densities of s different components; $\kappa^{\sigma\sigma'}$ is an interfacial free energy parameter which is responsible for the introduction of the surface tension. In this expression, the contribution from the higher order terms of the density gradients is neglected [32].

The chemical potential of each component can be obtained by a functional derivative as

$$\mu^\sigma = \frac{\delta \mathcal{F}}{\delta n^\sigma}, \tag{2.16}$$

where μ^σ is the chemical potential of component σ; n^σ is the number density of component σ; and $\mathcal{F}$ is the total free energy of the system.

The pressure in a bulk phase in equilibrium is given by

$$p = \sum_\sigma n^\sigma \mu^\sigma - \psi. \tag{2.17}$$

The pressure tensor is determined by two constraints: $P_{\alpha\beta} = p\delta_{\alpha\beta}$ in the bulk and $\Delta P_{\alpha\beta} = \sum_\sigma n^\sigma \nabla \mu^\sigma$ everywhere. A more detailed analysis is shown in Appendix D.

The total free energy of a system is at its minimum in equilibrium. This requires that the chemical potential for each component is constant, and the pressure tensor is divergence free. A system may separate into two phases to minimize the total free energy of the system.

Figure 2.1 illustrates the phase separation mechanism of a one component system. The total free energy of a one-component system, from Eq. (2.15), is given by

$$\mathcal{F} = \int d\mathbf{r} \left(\psi(n, \theta) + \frac{\kappa}{2} (\nabla n)^2 \right), \tag{2.18}$$

where κ is a coefficient related to the surface tension of the interface of the two phases after phase phase separation. The chemical potential of a one-component system, using Eq. (2.16), is given by

$$\mu = \frac{\partial \psi}{\partial n} - \kappa \nabla^2 n. \tag{2.19}$$

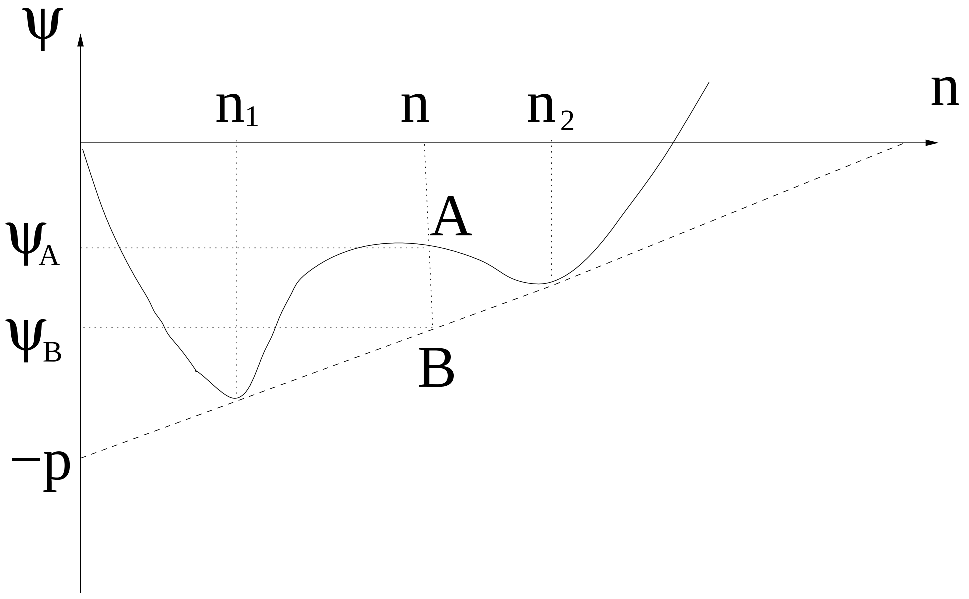


Figure 2.1. The phase separation of a system is determined by the shape of its bulk free energy density function. An initial homogeneous system of density n and bulk free energy density ψ_A separates into two phases of density n^A and n^B with the average bulk free energy density ψ_B to minimize the total free energy of the system. The pressure of the system in equilibrium is p.

The bulk pressure tensor of a one-component system in equilibrium, using Eq. (2.17), is given by

$$p = n\mu - \psi. \tag{2.20}$$

Eq. (2.19) indicates that in equilibrium, for a bulk phase, the chemical potential is the slope of the tangent line of the free energy density curve. Eq. (2.20) indicates that in the bulk phase, the negative of the pressure is the interception on the free energy density axis by the tangent line of the free energy density curve as shown in Figure 2.1. Slight deviations occurs where curved interfaces are present [59].

If the free energy density function of a system takes the shape shown in Figure 2.1, a system of average density n will separate into two phases of density n_1 and n_2 in equilibrium. Assume that the system has a volume V and the number density n in a homogeneous state. Then it separates into two states, one of volume V_1 with number density n_1 and the other of volume V_2 with density n_2. Because the system has a constant volume and constant total number of particles, I have the constraints:

$$\begin{aligned} V_1 + V_2 &= V, & (2.21) \\ n_1 V_1 + n_2 V_2 &= nV. & (2.22) \end{aligned}$$

From Eq. (2.21) and Eq. (2.22) the volume can be obtained as

$$\begin{aligned} V_2 &= \frac{n - n_1}{n_2 - n_1} V, & (2.23) \\ V_1 &= \frac{n_2 - n}{n_2 - n_1} V. & (2.24) \end{aligned}$$

The total free energy of the system in a homogeneous state is given by

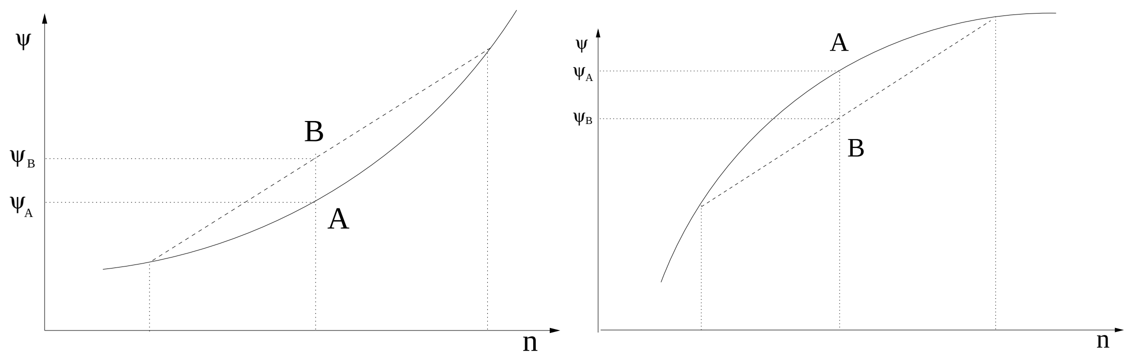
$$F_{\text{homogeneous}} \equiv \psi(n^A)\, V, \tag{2.25}$$

where ψ_A is the ψ value at point A in Figure 2.1. Using Eqs. (2.23) and (2.24), the total free energy of the system in a two-phase state can be obtained as

$$\begin{aligned} F_{\text{separated}} &= \psi_1 V_1 + \psi_2 V_2 \\ &= \psi_1 \frac{n_2 - n}{n_2 - n_1} V + \psi_2 \frac{n - n_1}{n_2 - n_1} V \\ &= \frac{\psi_2 (n - n_1) + \psi_1 (n_2 - n)}{n_2 - n_1} V \\ &\equiv \psi_B V, \end{aligned} \tag{2.26}$$

where ψ_B is the ψ value at point B in Figure 2.1. Since ψ_B is on the tangent line, it represents the minimum average bulk free energy density of a system of an average number density of n. The linear stability of a homogeneous phase is determined by the shape of the bulk free energy density function. Figure 2.2 (a) shows that a locally convex shape of the free energy curve implies stability with respect to small perturbations because the total free energy of the system would increase if the system phase separated. A phase at point A in Figure 2.2 (b) is unstable to infinitesimal perturbation because the total free energy of the system will decrease in the case of phase separation.

Generally, a system will undergo phase separation if the shape of its bulk free energy density function contains a concave section as shown in Figure 2.3. The system will phase-separate into phases C and D, and CD is a straight line tangent to the free energy line $\psi(x)$ at points C and D. Points C and D are called binodal points, at which the system has minimum total free energy. C' and D' are points that separate locally unstable regions and locally stable regions. Between C' and D', the system is unstable and will separate into two phases with only infinitesimal perturbation. Between C and C' and between D and D', the system is metastable and will undergo



(a) Stable (b) Unstable

Figure 2.2. The local stability of a system is determined by the concave or convex of the free energy density function. The free energy density of a homogeneous system is ψ_A. The average free energy density of the system with two separated phase is ψ_B. (a) A system is stable to a small perturbation when its free energy density function is convex locally. (b) A system is unstable to a small perturbation when its free energy density function is concave locally.

phase separation only with finite perturbations. Points C' and D' are called spinodal points. Mathematically, they are determined by the condition that $\psi''(x_{C'})$ and $\psi''(x_{D'})$ are zero.

A typical phase diagram is shown in Figure 2.4. In domain I, the system is unconditionally unstable and separates into two phases by spinodal decomposition. In domain II, the system is metastable and phase separates by nucleation. In domain III, the system is stable, and the homogeneous state has the lowest free energy.

2.4. Phase Separation of a System Described by a Landau Free Energy

The Landau free energy for a fluid with a liquid-gas transition has a form given by Eq. (2.18). To study the phase behavior near the critical point, it is convenient to

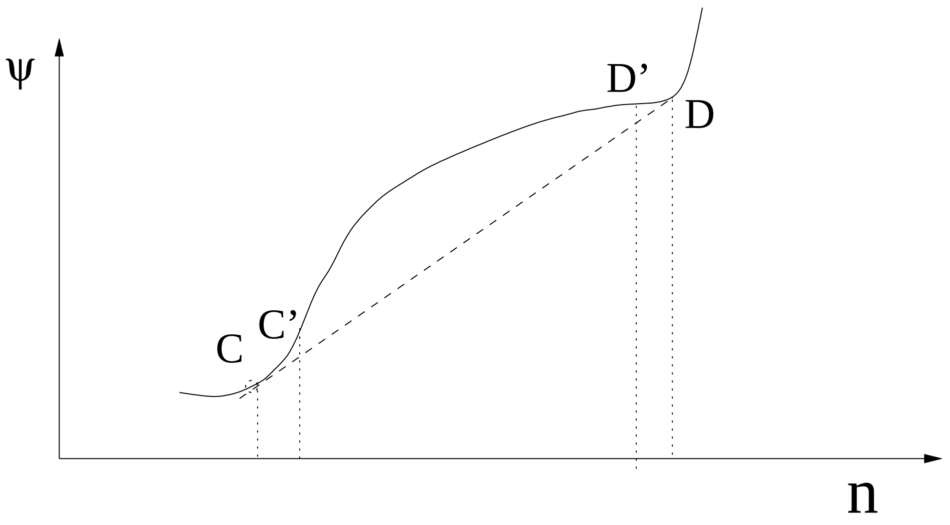


Figure 2.3. The binodal and spinodal points are determined by the shape of the free energy density function of a system. The binodal points C and D correspond to two equilibrium phases after phase separation. The spinodal points C' and D' separate the stable and unstable region.

express the free energy density as [7]

$$\psi(T,n) = W(n,T) + n\mu_b - p_b, \tag{2.27}$$

where $W(n,T)$ is the excess free energy density; $\mu_b \equiv \partial_n \psi|_{n=n_b}$ is the bulk chemical potential; and p_b is the bulk pressure. I chose the simplest excess free energy function

$$W(\nu,\tau) = p_c(\nu^2 - \beta\tau)^2. \tag{2.28}$$

Here $\nu = (n-n_c)/n_c$ is the reduced density; $\tau \equiv (T_c-T)/T_c$ is the reduced temperature; and T_c, p_c ,n_c are the critical temperature , critical pressure, and critical density,

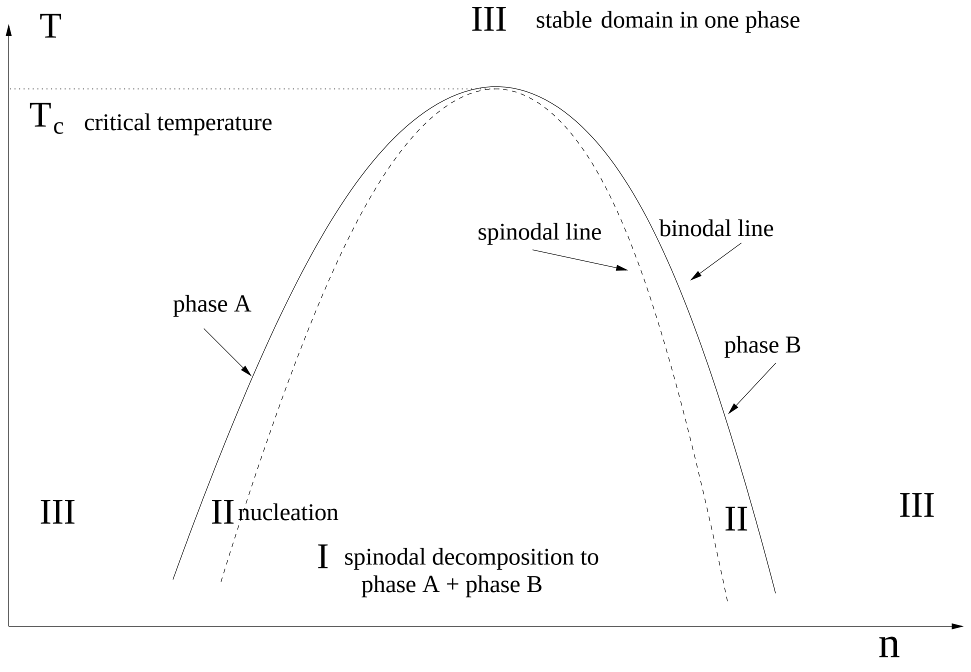


Figure 2.4. Inside the binodal lines the system tends to phase separate into two phases corresponding to the densities given by the two branches of the binodal line. Inside the spinodal lines the system is unconditionally unstable and a homogeneous system will phase separate through spinodal decomposition. In the region between the binodal and spinodal lines a homogeneous system will phase separate through nucleation.

respectively. The bulk chemical potential is given by

$$\mu_b = \frac{4p_c}{n_c}(1 - \beta\tau), \tag{2.29}$$

where β is a constant. The bulk pressure is

$$p_b = p_c(1 - \beta\tau)^2. \tag{2.30}$$

Therefore,

$$\psi = p_c(\nu + 1)^2(\nu^2 - 2\nu + 3 - 2\beta\tau). \tag{2.31}$$

When two phases coexist in equilibrium, the chemical potential without the surface tension terms is given by

$$\frac{\partial \psi}{\partial n} = \frac{4p_c}{n_c}(\nu + 1)(\nu^2 - \nu + 1 - \beta\tau). \tag{2.32}$$

The equilibrium gas and liquid densities are given by $\nu = \pm\sqrt{\beta\tau}$. Introducing $\theta \equiv -\beta\tau$, the density of liquid and gas are

$$n_l = n_c(1 + \sqrt{-\theta}), \qquad n_g = n_c(1 - \sqrt{-\theta}). \tag{2.33}$$

To solve the density profile n(x) of the interface of the system at the equilibrium state, I use two constraints. One is that the free energy of the system must be minimal. The other is the mass conservation of the system. Using the variation method with subsidiary constraints[19], I get

$$F_n - \frac{d}{dx}F_{n_x} + \lambda(n_n - \frac{d}{dx}n_{n_x}) = 0, \tag{2.34}$$

where $F = \psi(t, \tau) + \frac{\kappa}{2}(\partial_\alpha n)^2$; λ is a Lagrange multiplier; and the subscripts signify derivatives. For example, F_{n_x} signifies the derivative of function F with respect to n_x, and n_x signifies the derivative of function n with respect to x. Consequently,

$$\frac{4p_c}{n_c}(\nu+1)(\nu^2-\nu+1-\beta\tau)-\kappa\frac{d^2n}{dx^2}+\lambda=0, \tag{2.35}$$

where the Lagrange multiplier λ is $4p_c(1-\beta\tau)/n_c$. The interfacial profile can be obtained by solving the differential equation (2.35). The physically correct solution for a single interface is

$$n(x)=n_c[1+\sqrt{\beta\tau}\tanh(\frac{x}{\sqrt{2}\xi})], \tag{2.36}$$

where the interface width is

$$\xi=\sqrt{\frac{\kappa n_c^2}{4\beta\tau p_c}}. \tag{2.37}$$

The surface tension [7] is

$$\sigma=\frac{4}{3}\sqrt{2\kappa p_c}(\beta\tau)^{3/2}\,n_c. \tag{2.38}$$

The pressure tensor is (Appendix D),

$$P_{\alpha\beta}=[p_o-\kappa n\bigtriangledown^2 n-\frac{\kappa}{2}(\partial_\gamma n)^2]\,\delta_{\alpha\beta}+\kappa(\partial_\alpha n)(\partial_\beta n), \tag{2.39}$$

where $p_o=n\partial_n\psi-\psi$. For my one dimension simulations, the pressure tensor reduces to

$$p(x)=p_c(\nu+1)^2(3\nu^2-2\nu+1-2\beta\tau)-\kappa n\partial_x^2 n+\frac{\kappa}{2}(\partial_x n)^2. \tag{2.40}$$

The chemical potential with the surface tension term is

$$\mu = \frac{\partial \psi}{\partial n} = \frac{4p_c}{n_c}(\nu + 1)(\nu^2 - \nu + 1 - \beta\tau) - \kappa\nabla^2 n. \tag{2.41}$$

2.5. Phase Separation of a System Described by the Flory-Huggins Free Energy

The Flory-Huggins model is often employed to calculate the free energy of a polymer solution. It assumes that each segment of a polymer, or mer, occupies one lattice point. As shown in Appendix A, the Flory-Huggins free energy is given by

$$\begin{aligned} \mathcal{F} &= \int \Bigg[-\sum_{\sigma=1}^{s} \theta n^\sigma m^\sigma + \sum_{\sigma=1}^{s} (n^\sigma \theta \ln \phi^\sigma) + \sum_{\sigma=1}^{s} \sum_{\sigma'=\sigma+1}^{s} (\chi^{\sigma\sigma'} \theta m^\sigma n^\sigma \phi^{\sigma'}) \\ &\quad + \sum_{\sigma=1}^{s} \sum_{\sigma'=1}^{s} (\kappa^{\sigma\sigma'} \nabla n^\sigma \cdot \nabla n^{\sigma'}) \Bigg] \, dV, \end{aligned} \tag{2.42}$$

where m^σ is the polymerization of the component σ, which is the average number of mers per polymer; n^σ is the number density of component σ; and ϕ^σ is the volume fraction of component σ. It is defined as

$$\phi^\sigma = \frac{m^\sigma n^\sigma}{\sum_\sigma (m^\sigma n^\sigma)} = \frac{\rho^\sigma}{\rho}, \tag{2.43}$$

where ρ^σ is the mer density of component σ and ρ is the mer density of the system, which is a constant in the Flory-Huggins model.

2.5.1. Phase separation of a binary system

The equilibrium state of a binary system can be obtained by minimizing the free energy of the system. Derived from Eq. (2.42), the Flory-Huggins free energy density of a binary polymer system without the interfacial free energy contribution is given by

$$\psi = \theta \left(-\rho^A - \rho^B + \frac{\rho^A}{m^A} \ln \phi^A + \frac{\rho^B}{m^B} \ln \phi^B + \chi \rho^A \phi^B \right). \tag{2.44}$$

In many applications, a solution contains only one polymer component. So it is reasonable to choose $m^A = m$, where m is the polymerization of component A and $m^B = 1$. In the Flory-Huggins model, the lattice density is constant; therefore $\rho \equiv \rho^A + \rho^B$ is a constant.[1] For the calculation of binodal points, I can assume $\rho\theta = 1$ and neglect an arbitrary constant without affecting the final results. With the constraint $\phi^A + \phi^B = 1$, Eq. (2.44) has only one independent variable $\phi \equiv \phi^A$. Then Eq. (2.44) can be simplified to

$$\psi = \frac{1}{m} \phi \ln \phi + (1 - \phi) \ln(1 - \phi) + \chi \phi (1 - \phi). \tag{2.45}$$

From Eq. (2.45), the whole phase diagram of a binary system can be established. First, let us determine the critical point. Because the critical point is on the spinodal line, it satisfies the spinodal line equation $d^2F/d\phi^2 = 0$. Because the two spinodal lines meet

[1] This is not an explicit constraint in a LB simulation. However, in a LB simulation, when an equilibrium state has been reached, the total density is very close to a constant.

at the critical point, $d^3F/d\phi^3 = 0$.

$$\begin{aligned} F'' &= \frac{1}{m}\frac{1}{\phi} + \frac{1}{1-\phi} - 2\chi = 0, && (2.46) \\ F''' &= -\frac{1}{m}\frac{1}{\phi^2} + \frac{1}{(1-\phi)^2} = 0. && (2.47) \end{aligned}$$

Solving Eq. (2.46) and Eq. (2.47), the coordinates of the critical point is obtained as

$$\begin{aligned} \phi &= \frac{1}{\sqrt{m}+1}, \\ \chi &= \frac{m + 2\sqrt{m} + 1}{2m}. && (2.48) \end{aligned}$$

Eq. (2.46) can be rearranged as

$$2m\chi\phi^2 + (1 - m - 2m\chi)\phi + m = 0. \quad (2.49)$$

Eq. (2.49) yields the spinodal lines with the equation

$$\phi_{1,2} = \frac{-(1 - m - 2m\chi) \pm \sqrt{\Delta_{sp}}}{4m\chi}, \quad (2.50)$$

where $\Delta_{sp} = (1 - m - 2m\chi)^2 - 8m^2\chi$.

The theoretical binodal lines, which correspond to the two equilibrium states of the system, can be obtained numerically by minimizing the total free energy of the system, and the numerical method is presented in Appendix F. The spinodal and binodal lines are shown in Figure 2.5 for different polymerizations.

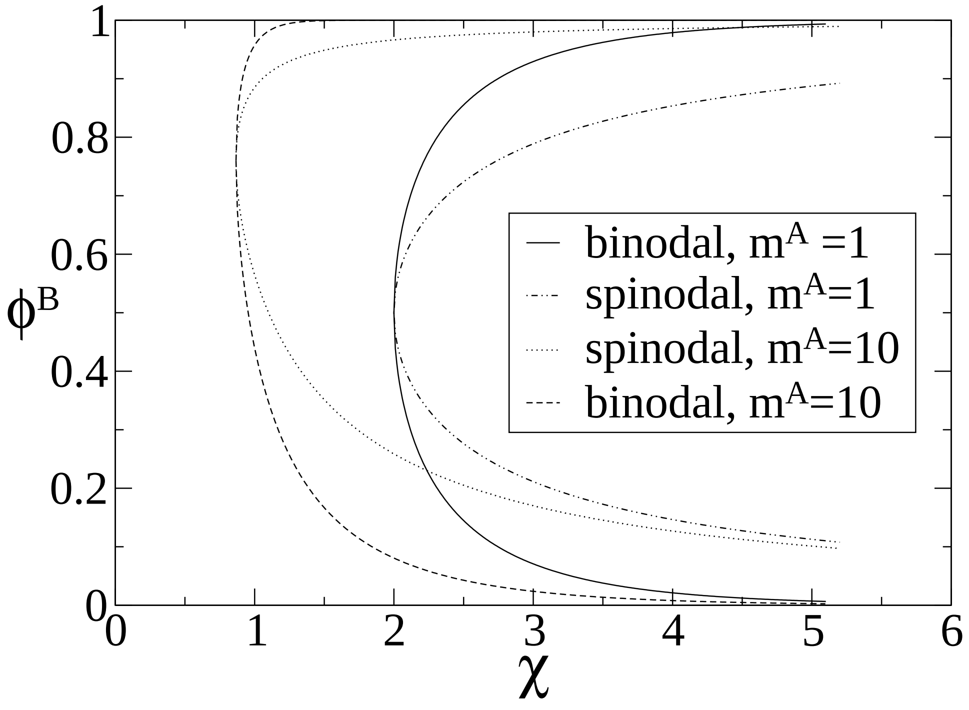


Figure 2.5. The theoretical binodal and spinodal lines for a monomer binary system ($m^A = 1, m^B = 1$) and a polymer system ($m^A = 10$, $m^B = 1$) are presented.

2.5.2. Phase separation of a ternary system

The total free energy of a ternary system can be obtained from Eq. (2.42) as

$$\begin{aligned}
\mathcal{F} \;\; = \;\; & \int [\theta(-m^A n^A - m^B n^B - m^C n^C \\
& + n^A \ln \phi^A + n^B \ln \phi^B + n^C \ln \phi^C \\
& + \chi^{AB} n^A m^A \phi^B + \chi^{AC} n^A m^A \phi^C + \chi^{BC} n^B m^B \phi^C) \\
& + \kappa^{AA} (\nabla n^A)^2 + \kappa^{BB} (\nabla n^B)^2 + \kappa^{CC} (\nabla n^C)^2 \\
& + \kappa^{AB} \nabla n^A \nabla n^B + \kappa^{AC} \nabla n^A \nabla n^C + \kappa^{BC} \nabla n^B \nabla n^C] dV, \qquad (2.51)
\end{aligned}$$

where m^σ is the polymerization of component σ. Of course, m^σ is one for a monomer.

A ternary system has three components, but a Flory-Huggins model of a ternary system has only two independent variables because the density factors out of the free energy. Therefore, ϕ_A and ϕ_B can be chosen as two independent variables, and ϕ_C can be obtained by $\phi_C = 1 - \phi_A - \phi_B$.

Another constraint for a phase separation in a ternary system is that the initial homogeneous phase and the two separated phases are always in a straight line in a ϕ^A-ϕ^B diagram. The simple analysis below will make this clear.

Suppose the two volume fractions are chosen as phase parameters. The initial phase is (ϕ^A, ϕ^B), and the total volume of the system is V. The system evolves into two phases: (ϕ_1^A, ϕ_1^B) in volume V_1, and (ϕ_2^A, ϕ_2^B) in volume V_2. From the constraint that the total mass of each component is conserved, so I get

$$V\phi^A = V_1\phi_1^A + V_2\phi_2^A, \tag{2.52}$$

$$V\phi^B = V_1\phi_1^B + V_2\phi_2^B. \tag{2.53}$$

From the constraint that the volume of the system is constant $V = V_1 + V_2$, it follows that

$$\frac{\phi^B - \phi_1^B}{\phi^A - \phi_1^A} = \frac{\phi_2^B - \phi_1^B}{\phi_2^A - \phi_1^A}. \tag{2.54}$$

Eq. (2.54) shows that the three points are in a straight line in a ϕ^A-ϕ^B graph. Therefore, in a ternary system, a phase separates into two phases along a straight line, which is called the tie line.[2]

[2]In a general case there is the possibility of a three phase region which is ignored here.

The theoretical thermodynamics of an equilibrium ternary system were obtained by minimizing the total free energy of the system numerically as discussed. In this thesis, the binodal lines of a ternary system is obtained by varying the initial volume fraction of the components while holding the χ parameters constant. The simulation of phase separation of a ternary system with constant χ parameters sheds light to the study of an immersion precipitation process [27] in which the χ parameters are constant. In contrast, the binodal lines of a binary system are obtained by varying χ while keeping the initial volume fraction of the components constant.

Starting from the critical point, the volume fractions of two components of the initial state (ϕ^A,ϕ^B) was increased linearly towards the final point ($\phi^A = 0.5, \phi^B = 0.5$). For each initial state, two corresponding binodal points are obtained along a straight line. The tie lines of different initial states usually are not parallel to each other. Therefore, to calculate the binodal lines of a ternary system, I first calculate the two phases that minimize the total free energy of the system along a straight line; then, I rotate the straight line back and forth. Iterating this procedure allowed me to find the exact direction of the tie line along which the total free energy of the system is minimal. I have checked that the chemical potentials of the two phases of each component are equal to a precision of at least of 10^{-5}. That means that the binodal points obtained with this approach represent equilibrium states. The collection of all the binodal points generated the binodal line. The implementation of this algorithm is shown Appendix G.

The phase diagram of a ternary system can be presented in Cartesian coordinates, but a triangular representation is preferred in the literature [4, 6, 9]. Actually, the

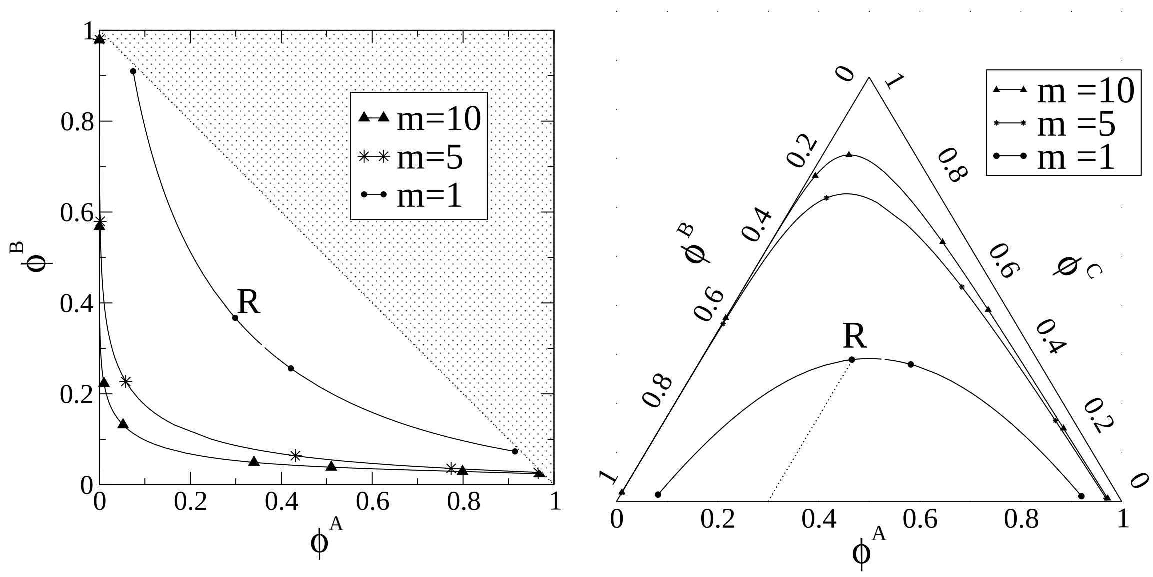


Figure 2.6. The binodal lines for polymerization m = 1, 5 and 10 of a ternary system are illustrated in a Cartesian and a triangular coordinate. The shaded area is inaccessible because of the constraint $\phi^A + \phi^B \leq 1$. The system has $\chi^{AB} = 3$, $\chi^{AC} = 0.5$, $\chi^{BC} = 0.2$. Point R corresponds to (0.30, 0.37, 0.33). The dotted line from point R shows the approach to determine ϕ^A of the point.

representations are equivalent, and a linear transformation exists between the two systems. The phase diagram of a ternary system is represented in both coordinates in Figure 2.6. The next chapter introduces the lattice Boltzmann method.

CHAPTER 3

LATTICE BOLTZMANN METHOD

3.1. Single Relaxation Time Lattice Boltzmann

The Lattice Boltzmann Equation (LBE) can be understood as a discrete Boltzmann Equation [36]. The LBE with a single relaxation time from the BGK model can be expressed as [41, 57]

$$f_i(\mathbf{r} + \mathbf{v_i}\Delta t, t + \Delta t) - f_i(\mathbf{r}, t) = \Delta t \left(\frac{1}{\tau}(f_i^e(\mathbf{r}, t) - f_i(\mathbf{r}, t) + G_i) + F_i \right), \tag{3.1}$$

where $\mathbf{r}$ is the lattice position vector; $\mathbf{v_i}$ is particle velocity; t is time; τ is the single relaxation time parameter, but the inverse relation time $\Omega \equiv 1/\tau$ is often used instead of τ as simulation parameter; G_i introduces non ideal effects into the hydrodynamic pressure tensor [38]; F_i is a forcing term that can be used to introduce non-ideal effects into the bulk force [20, 34, 44, 64]; $f_i(\mathbf{r}, t)$ denotes the particle distribution associated with the discrete velocity $\mathbf{v}_i$; and f_i^e indicates the local equilibrium distribution corresponding to an ideal gas. The discrete velocity $\mathbf{v}_i$ is chosen such that the $\mathbf{v}_i\Delta t$ is a lattice vector.

In this thesis, I limit my study to isothermal systems in which mass and momentum are conserved in the collision, and energy conservation is abandoned in favor of a constant temperature requirement. I define the density ρ as $\rho(x) \equiv \sum_i f_i(x)$ and the velocity $\mathbf{u}$ as $\rho(x)\mathbf{u}(x) \equiv \sum_i f_i(x)v_i$. From mass and momentum conservation the first two moments of the local equilibrium distribution function of an ideal gas are

$$\sum_i (f_i^e - f_i) = 0 \;\Rightarrow\; \sum_i f_i^e = \sum_i f_i \equiv \rho, \tag{3.2}$$

$$\sum_i (f_i^e - f_i)\mathbf{v}_i = 0 \;\Rightarrow\; \sum_i f_i^e \mathbf{v}_i = \sum_i f_i \mathbf{v}_i \equiv \rho\mathbf{u}. \tag{3.3}$$

Analogous to the continuous case, the other two moments of the equilibrium distribution function of an ideal gas can be defined as [41]

$$\sum_i f_i^e v_{i\alpha} v_{i\beta} = \frac{1}{3}\rho\delta_{\alpha\beta} + \rho u_\alpha u_\beta, \tag{3.4}$$

$$\sum_i f_i^e v_{i\alpha} v_{i\beta} v_{i\gamma} = \frac{1}{3}\rho(u_\alpha\delta_{\beta\gamma} + u_\beta\delta_{\alpha\gamma} + u_\gamma\delta_{\alpha\beta}) + \rho u_\alpha u_\beta u_\gamma + Q_{\alpha\beta\gamma}, \tag{3.5}$$

where $Q_{\alpha\beta\gamma}$ is a tensor term that is zero in the equivalent continuous integral. Because $v_{ix} = v_{ix}^3$, I have

$$\sum_i f_i^e v_{ix}^3 = \sum_i f_i^e v_{ix} = \rho\mathbf{u}. \tag{3.6}$$

Therefore, $Q_{\alpha\beta\gamma}$ is usually chosen to be $-\rho u_\alpha u_\beta u_\gamma$ to make Eq. (3.5) satisfy Eq. (3.6). But this choice of $Q_{\alpha\beta\gamma}$ causes a small Galilean Invariance violation for moderate $|u|$.

This will be discussed in Chapter 4. The moments of G_i are given by

$$\begin{aligned}
\sum_i G_i &= 0, && (3.7)\\
\sum_i G_i v_{i\alpha} &= 0, && (3.8)\\
\sum_i G_i v_{i\alpha} v_{i\beta} &= A_{\alpha\beta}, && (3.9)\\
\sum_i G_i v_{i\alpha} v_{i\beta} v_{i\gamma} &= 0, && (3.10)
\end{aligned}$$

where $A_{\alpha\beta}$ is a tensor describing the non-ideal part of the pressure tensor of the liquid. The velocity moments of the F_i are given by

$$\begin{aligned}
\sum_i F_i &= 0, && (3.11)\\
\sum_i F_i v_{i\alpha} &= \rho a_\alpha, && (3.12)\\
\sum_i F_i v_{i\alpha} v_{i\beta} &= \rho(a_\alpha u_\beta + a_\beta u_\alpha), && (3.13)\\
\sum_i F_i v_{i\alpha} v_{i\beta} v_{i\gamma} &= \frac{1}{3}\rho(a_\alpha \delta_{\beta\gamma} + a_\beta \delta_{\alpha\beta} + a_\gamma \delta_{\alpha\beta}), && (3.14)
\end{aligned}$$

where a_α is the acceleration.

Non-ideal fluid flow can be simulated by using a free energy from which the chemical potential can be derived. There are two ways to include the interaction into the LB evolution equation: the pressure approach ($F_i = 0$) and the forcing approach ($G_i = 0$). The two approaches are subtly different because of higher order effects. This is beyond the scope of this thesis. Interested readers can refer to a current paper by A.J. Wagner [55].

To recover the hydrodynamic equations, I use a moment method. First I perform a Taylor expansion to the left side of Eq. (3.1),

$$\sum_{k=1}^{\infty} \frac{(\Delta t)^k}{k!} (\partial_t + v_{i\alpha}\partial_\alpha)^k f_i = \Delta t \left(\frac{1}{\tau}(f_i^e - f_i + G_i) + F_i \right). \tag{3.15}$$

In the long wavelength and small frequency limit, ∂_α and ∂_t result in small quantities of the order of ϵ which indicates the order of smallness. Neglecting all the terms of $O(\epsilon^2)$, I get

$$\Delta t(\partial_t + v_{i\alpha}\partial_\alpha) f_i + O(\epsilon^2) = \Delta t \left(\frac{1}{\tau}(f_i^e - f_i + G_i) + F_i \right). \tag{3.16}$$

With Eq. (3.16), f_i can be expressed in terms of the f_i^e, F_i and G_i, for which all moments are known. Neglecting all the terms of $O(\epsilon^3)$ in Eq. (3.15),

$$\Delta t(\partial_t + v_{i\alpha}\partial_\alpha) f_i + \frac{(\Delta t)^2}{2}(\partial_t + v_{i\alpha}\partial_\alpha)^2 f_i + O(\epsilon^3) = \Delta t \left(\frac{1}{\tau}(f_i^e - f_i + G_i) + F_i \right). \tag{3.17}$$

Higher order terms are not considered here because the hydrodynamic equations only contain second order derivatives.

Eqs. (3.2)-(3.5) establish the relations between the macroscopic variables, ρ and $\mathbf{u}$, and the moments of the equilibrium distributions. In order to use those relations, f_i is expanded in terms of f_i^e . Using Eq. (3.16) recursively,

$$f_i = f_i^e + G_i + \tau F_i - \tau(\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i) + O(\epsilon^2). \tag{3.18}$$

Substituting Eq. (3.18) into the left side of Eq. (3.17),

$$\begin{aligned} & (\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i) - w(\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^e + G_i + \tau F_i) + O(\epsilon^3) \\ = \; & \frac{1}{\tau}(f_i^e + G_i + \tau F_i - f_i), \end{aligned} \tag{3.19}$$

where $w \equiv \tau - \Delta t/2$.

To obtain the mass conservation equation, I sum Eq. (3.19) over i and obtain,

$$\partial_t \rho + \partial_\alpha(\rho u_\alpha) + \tau\partial_\alpha(\rho a_\alpha) - w\sum_i(\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^e + G_i + \tau F_i) + O(\epsilon^3) = 0. \tag{3.20}$$

Eq. (3.20) shows that $\partial_t \rho + \partial_\alpha(\rho u_\alpha) + \tau\partial_\alpha(\rho a_\alpha)$ is of the order of $O(\epsilon^2)$, so Eq. (3.20) can be simplified to

$$\begin{aligned} & \partial_t \rho + \partial_\alpha(\rho u_\alpha) + \tau\partial_\alpha(\rho a_\alpha) - w\partial_\beta \Bigg[\partial_t(\rho u_\beta) + \tau\partial_t(\rho a_\beta) + \partial_\alpha \sum_i f_i^o v_{i\alpha} v_{i\beta} \\ & + \partial_\alpha A_{\alpha\beta} + \partial_\alpha \sum_i F_i v_{i\alpha} v_{i\beta}\Bigg] + O(\epsilon^3) = 0. \end{aligned} \tag{3.21}$$

Multiplying Eq. (3.19) with $v_{i\beta}$ and summing over i, I get

$$\sum_i v_{i\beta}(\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i) - \sum_i v_{i\beta} w(\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^e + G_i + \tau F_i) = \rho a_\beta. \tag{3.22}$$

Eq. (3.22) shows that

$$\sum_i v_{i\beta}(\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i) = \rho a_\beta + O(\epsilon^2). \tag{3.23}$$

Substituting Eq. (3.23) into Eq. (3.21),

$$\partial_t \rho + \partial_\alpha \left[\rho(u_\alpha + \frac{\Delta t}{2} a_\alpha) \right] + O(\epsilon^3) = 0. \tag{3.24}$$

Therefore, by denoting the macroscopic velocity $U_\alpha \equiv u_\alpha + a_\alpha \Delta t/2$, the continuity equation (2.1) can be recovered to the second order:

$$\partial_t \rho + \partial_\alpha (\rho U_\alpha) = 0 + O(\epsilon^3). \tag{3.25}$$

To obtain the momentum conservation equation, I substitute Eq. (3.23) into Eq. (3.22).

$$\begin{aligned} &\sum_i v_{i\beta}(\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i) - \sum_i w v_{i\beta} v_{i\gamma} \partial_\gamma (\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i). \\ &-w\partial_t \rho a_\beta + O(\epsilon^3) = \rho a_\beta. \end{aligned} \tag{3.26}$$

Eq. (3.26) shows that **a** is of the oder $O(\epsilon)$. Eq. (3.12) shows that F_i is of the same order as **a**, i.e. $O(\epsilon)$. The second term in Eq. (3.26) is simplified, step by step, as follows:

$$\begin{aligned} & \sum_i w v_{i\beta} v_{i\gamma} \partial_\gamma (\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i + \tau F_i) \\ = & \sum_i w v_{i\beta} v_{i\gamma} \partial_\gamma (\partial_t + v_{i\alpha}\partial_\alpha)(f_i^e + G_i) + O(\epsilon^3) \\ = & w\partial_\gamma \left(\partial_t \sum_i f_i^e v_{i\beta} v_{i\gamma} + \partial_t A_{\beta\gamma} + \partial_\alpha \sum_i f_i^e v_{i\alpha} v_{i\beta} v_{i\gamma} \right) + O(\epsilon^3). \end{aligned} \tag{3.27}$$

Using Eq. (3.23),

$$\partial_t(\rho u_\beta) = -\partial_\alpha \left(\frac{\rho}{3}\delta_{\alpha\beta} + A_{\alpha\beta} + \rho u_\alpha u_\beta\right) + \rho a_\beta + O(\epsilon^2). \tag{3.28}$$

Substituting Eq. (3.25) into Eq. (3.28),

$$\rho\partial_t u_\beta = -\rho u_\alpha \partial_\alpha u_\beta - \frac{\partial_\beta \rho}{3} - \partial_\alpha A_{\alpha\beta} + \rho a_\beta + O(\epsilon^2). \tag{3.29}$$

Using Eqs. (3.25), (3.28) and (3.29),

$$\begin{aligned}
& \partial_t \sum_i f_i^e v_{i\beta} v_{i\gamma} + \partial_\alpha \sum_i f_i^e v_{i\alpha} v_{i\beta} v_{i\gamma} \\
= & \ \partial_t \left(\frac{\rho}{3}\delta_{\beta\gamma} + \rho u_\beta u_\gamma\right) \\
& +\partial_\alpha \left[\frac{\rho}{3}(u_\alpha \delta_{\beta\gamma} + u_\beta \delta_{\alpha\gamma} + u_\gamma \delta_{\alpha\beta} + \rho u_\alpha u_\beta u_\gamma + Q_{\alpha\beta\gamma} + O(\epsilon)\right] \\
= & \ \frac{1}{3}\delta_{\beta\gamma}(-\partial_\alpha(\rho U_\alpha)) + u_\gamma \partial_t(\rho u_\beta) + \rho u_\beta \partial_t u_\gamma \\
& +\frac{1}{3}\delta_{\beta\gamma}\partial_\alpha(\rho u_\alpha) + \frac{1}{3}\partial_\gamma(\rho u_\beta) + \frac{1}{3}\partial_\beta(\rho u_\gamma) + \partial_\alpha(\rho u_\alpha u_\beta u_\gamma) + \partial_\alpha Q_{\alpha\beta\gamma} + O(\epsilon^2) \\
= & \ u_\gamma \left(-\frac{1}{3}\partial_\beta \rho - \partial_\alpha A_{\alpha\beta} - \partial_\alpha(\rho u_\alpha u_\beta) + \rho a_\beta\right) + u_\beta \left(-u_\alpha u_\alpha u_\gamma - \partial_\alpha A_{\alpha\beta} - \frac{1}{3}\partial_\gamma \rho + \rho a_\gamma\right) \\
& +\frac{1}{3}\partial_\gamma(\rho u_\beta) + \frac{1}{3}\partial_\beta(\rho u_\gamma) + \partial_\alpha(\rho u_\alpha u_\beta u_\gamma) + \partial_\alpha Q_{\alpha\beta\gamma} + O(\epsilon^2) \\
= & \ \frac{1}{3}\rho\partial_\gamma u_\beta + \frac{1}{3}\rho\partial_\beta u_\gamma + \rho u_\gamma a_\beta + \rho u_\beta a_\gamma \\
& -u_\gamma \partial_\alpha A_{\alpha\beta} - u_\beta \partial_\alpha A_{\alpha\gamma} + \partial_\alpha Q_{\alpha\beta\gamma} + O(\epsilon^2).
\end{aligned} \tag{3.30}$$

From Eqs. (3.25), (3.26), and (3.30), I obtained with some algebra,

$$\begin{aligned}\partial_t(\rho U_\alpha) &+ \partial_\beta(\rho U_\alpha U_\beta) = -\partial_\beta\left(\frac{1}{3}\rho\delta_{\alpha\beta} + A_{\alpha\beta}\right) + \partial_\beta\left[\frac{w}{3}\rho(\partial_\beta U_\alpha + \partial_\alpha U_\beta)\right] + \rho a_\alpha \\ &+ w\partial_\gamma(-u_\gamma\partial_\beta A_{\alpha\beta} - u_\alpha\partial_\beta A_{\beta\gamma} + \partial_t A_{\alpha\gamma} + \partial_\beta Q_{\alpha\beta\gamma}) + O(\epsilon^3). \qquad (3.31)\end{aligned}$$

Eq. (3.31) is a Navier Stokes equation except the terms of the last line that cause Galilean invariance violations (Section 4.1.).

In other literature, a second approach exists to derive the hydrodynamic equations of the lattice Boltzmann method (LBM). It is a multi-scale expansion, referred to as the "Chapman Enskog" approach. The two approaches give the same results as far as the recovery of the hydrodynamic equations are concerned [41, 62].

Identical forms of Eq. (3.31) can be implemented with different choices of A or $\mathbf{a}$. If both A and $\mathbf{a}$ are zero, Eq. (3.31) becomes the Navier Stokes equation for an ideal gas. But even in the ideal gas LB model, the $w\partial_\gamma\partial_\beta Q_{\alpha\beta\gamma}$ exists and causes the GI violation. The non-ideal model for LB simulation can be obtained by choosing either A or a to be zero. The choice of $\mathbf{a} = 0$ and $A \neq 0$ is referred to as the pressure approach; the choice of $\mathbf{a} \neq 0$ and $A = 0$ is referred to as the force approach. Swift et al. [48] invented a model to simulate a non-ideal fluid in pressure approach by choosing $A_{\alpha\beta} = P_{\alpha\beta} - \frac{1}{3}\rho$. This model is deficient and cause a severe Galilean invariance violation. The errors come from the terms in the second line in Eq. (3.31). This problem was addressed by Holdych et al. [21], Inamuro et al. [22], and Kalarakis et al. [24] independently. They solved the problem by redefining

$$A_{\alpha\beta} = P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta} - \nu(\partial_\alpha \rho u_\beta + \partial_\beta \rho u_\alpha + u_\gamma \partial_\gamma \rho \delta_{\alpha\beta}). \tag{3.32}$$

Eq. (3.32) leaves the density gradient terms in the second line of Eq. (3.31) to the second order, which are assumed to be very small because the derivatives of the pressure tensor are much smaller than the derivatives of density at the interface near equilibrium[24]. Therefore, Holdych's model gives a more accurate result and diminished the GI violation significantly. To simulate a one component phase separating system with a forcing approach, $\rho a_\alpha = \partial_\beta(P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta})$ can be chosen. To eliminate the Q term error in ideal gas model or in non-ideal models of pressure or forcing approach, a forcing term is introduced as $\rho a_\alpha = w\partial_\beta\partial_\gamma Q_{\alpha\beta\gamma}$. A Holdych model with a Q correction is referred to as HoldychQ, similarly is the ForcingQ. Table 3.1. summarizes the simulation approaches explored in this thesis.

Table 3.1. The algorithms for the LB simulation of a non-ideal fluid

Approach	$\mathbf{A}_{\alpha\beta}$	$\boldsymbol{\rho}\mathbf{a}_{\alpha}$
Pressure	$P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta}$	0
Holdych	$P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta}$ $-\nu(\partial_\alpha \rho u_\beta + \partial_\beta \rho u_\alpha + u_\gamma \partial_\gamma \rho \delta_{\alpha\beta})$	0
HoldychQ	$P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta}$ $-\nu(\partial_\alpha \rho u_\beta + \partial_\beta \rho u_\alpha + u_\gamma \partial_\gamma \rho \delta_{\alpha\beta})$	$w\partial_\beta\partial_\gamma Q_{\alpha\beta\gamma}$
Forcing	0	$\partial_\beta(P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta})$
ForcingQ	0	$\partial_\beta(P_{\alpha\beta} - \frac{1}{3}\rho\delta_{\alpha\beta}) + w\partial_\beta\partial_\gamma Q_{\alpha\beta\gamma}$

3.1.1. One-dimensional three-velocity lattice Boltzmann model

Many lattice Boltzmann models are available such as the one-dimensional three-velocity model (D1Q3), the two-dimensional seven-velocity model (D2Q7), the two dimensional nine velocity model (D2Q9), the three dimensional and fifteen velocity model (D3Q15), and the three dimensional and twenty seven velocity model (D3Q27). Although the D1Q3 model is the simplest, it is, in fact, a projection of all the other models mentioned above. Therefore any a problem in the D1Q3 model exists also in the higher dimensional models. In this thesis, D1Q3 models were used for all simulations and are introduced next.

The D1Q3 model has three velocities: $v_0 = 0$, $v_1 = 1$, $v_2 = -1$. In one dimension, Eqs. (3.2) and (3.3) become

$$\rho = f_0 + f_1 + f_2, \tag{3.33}$$

$$\rho u = f_1 - f_2, \tag{3.34}$$

and the local equilibrium distribution satisfies

$$\rho = f_0^e + f_1^e + f_2^e, \tag{3.35}$$

$$\rho u = f_1^e - f_2^e, \tag{3.36}$$

$$\rho u^2 + \frac{\rho}{3} = f_1^e + f_2^e. \tag{3.37}$$

I first derive the equilibrium distribution for the ideal gas. With Eqs. (3.35), (3.36), and (3.37),

$$f_0^e = \frac{2}{3}\rho - \rho u^2, \tag{3.38}$$

$$f_1^e = \frac{1}{2}(\rho/3 + \rho u + \rho u^2), \tag{3.39}$$

$$f_2^e = \frac{1}{2}(\rho/3 - \rho u + \rho u^2) \tag{3.40}$$

For a non-ideal gas with a pressure approach, Eqs. (3.7), (3.8), and (3.9) give

$$G_0 + G_1 + G_2 = 0, \tag{3.41}$$

$$G_1 - G_2 = 0, \tag{3.42}$$

$$G_1 + G_2 = A, \tag{3.43}$$

which yields

$$G_0 = 0, \tag{3.44}$$

$$G_1 = A/2, \tag{3.45}$$

$$G_2 = A/2. \tag{3.46}$$

For a non-ideal gas with a forcing approach, Eqs. (3.11), (3.12), and (3.13) give

$$F_0 + F_1 + F_2 = 0, \tag{3.47}$$

$$F_1 - F_2 = \rho a, \tag{3.48}$$

$$F_1 + F_2 = 2\rho a u. \tag{3.49}$$

This can be solved for the F_i to give

$$\begin{aligned}
F_0 &= -2\rho a u, &(3.50)\\
F_1 &= (u+0.5)\rho a, &(3.51)\\
F_2 &= (u-0.5)\rho a, &(3.52)
\end{aligned}$$

where ρa is the interaction force derived from the non-ideal gas interaction. The evolution equations for a D1Q3 model is then:

$$\begin{aligned}
f_0(i,t+1) &= f_0(i,t) + \frac{1}{\tau}(f_0^e(i,t) - f_0(i,t) + G_0(i,t)) + F_0(i,t), &(3.53)\\
f_1(i+1,t+1) &= f_1(i,t) + \frac{1}{\tau}(f_1^e(i,t) - f_1(i,t) + G_1(i,t)) + F_1(i,t), &(3.54)\\
f_2(i-1,t+1) &= f_2(i,t) + \frac{1}{\tau}(f_2^e(i,t) - f_2(i,t) + G_2(i,t)) + F_2(i,t), &(3.55)
\end{aligned}$$

where i is the lattice coordinate, and t is the time step.

3.1.2. Lattice Boltzmann for a liquid-gas system

To validate my theoretical analysis of LB simulations of non-ideal fluids, I performed LB simulations of a simple one component non-ideal system described by the Landau free energy model of Eq. (2.27) with Holdych's approach and a forcing approach. The equilibrium thermodynamic properties (density, pressure, and chemical potential) of the system obtained by the LB simulation through the two approaches are compared to the theoretical results derived in Section 2.4. The theoretical density profile is given by Eq. (2.36). The pressure is given by Eq. (2.40), and the chemical potential is given by Eq. (2.41).

My LB simulations were performed with the D1Q3 models of Table 3.1. Each simulation was performed on a lattice with 100 lattice points, unless noted otherwise. Initially the system had a uniform density of 1, plus a sinusoidal perturbation to trigger phase separation. The amplitude of the perturbation was 0.1 and its wavelength was the lattice size.

To obtain the binodal lines of a liquid-gas system, the LB simulations were performed starting with these initial conditions with increasing value of θ starting from the critical point. A pair of binodal points were obtained by the LB simulation from each initial condition. I continued to increase θ for each simulation until I observed numerical instabilities. The binodal points were than given by the maximum and minimum densities in equilibrium. The system evolved into a stable state after about 1000 time steps. The measurements were taken after 10000 time steps to be sure that the system was in equilibrium.

Figure 3.1 shows the comparison of the binodal points obtained by the "Holdych" and "ForcingQ" approaches to the theoretical binodal lines. The simulation results approach the theoretical result, but the deviations increases as θ increases. The Holdych approach is more accurate, but the ForcingQ approach has a larger stability range. However, a stability analysis is outside the scope of this thesis; for more details on the stability of the LB methods, an interested reader may refer to [2, 3, 37, 50, 52, 47, 54, 61].

Figure 3.2 shows the comparison of the total density, pressure, and and chemical potentials obtained by HoldychQ approach to the theoretical values. The equilibrium density profile almost overlapped with the theoretical profile. The equilibrium pressure

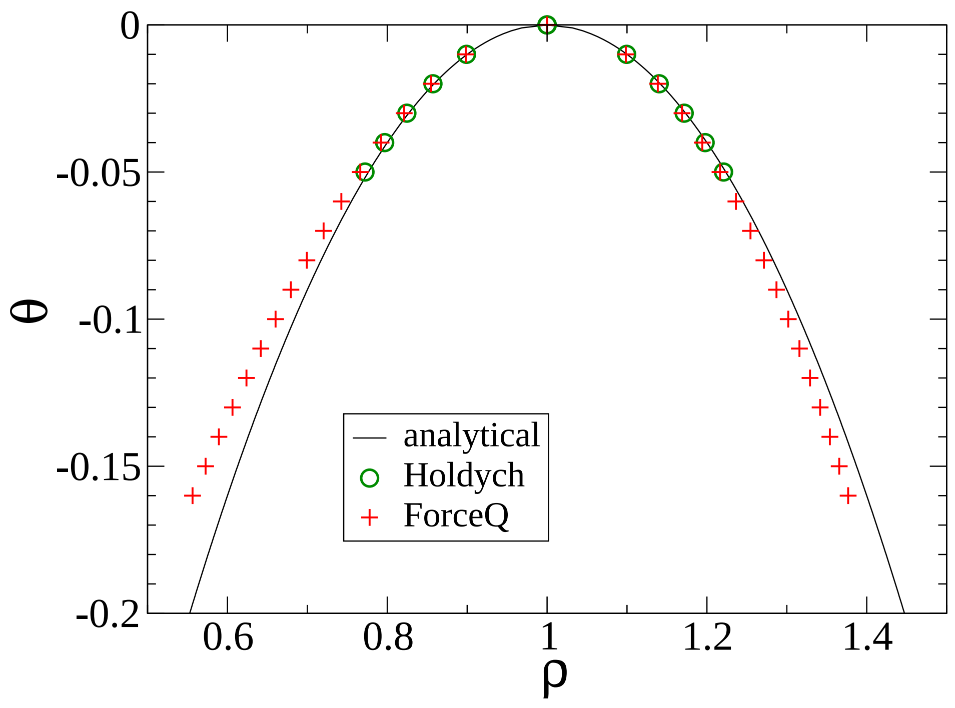


Figure 3.1. The binodal points of a liquid-gas system obtained by the LB simulation with the "ForcingQ" and "Holdych" approaches are compared to the analytical solution. The system has $\kappa = 0.1$, $n_c = 1$, $p_c = 0.42$, and $\nu = 1/6$. For the LB simulation, $\omega = 1.0$.

obtained by the simulation was slightly lower than the theoretical values, but the difference was less than 0.1% when compared to the theoretical value. The equilibrium chemical potential obtained by the LB simulation in the two bulk phases differ only within 0.01%. There are two spikes on the chemical potential at the interface of the two phases, but the spikes are less than 0.1% compared to the bulk chemical potentials. The chemical potential obtained by the LBM are slightly lower than the theoretical value, but the difference was less than 0.1% compared to the theoretical value.

Figure (3.3) shows the comparison of the total density, pressure, and and chemical potentials obtained by ForcingQ approach to the theoretical values. The density profile

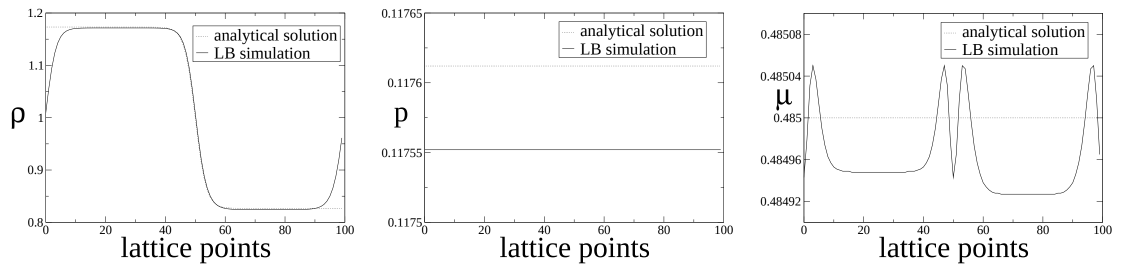


Figure 3.2. The density, pressure, and the chemical potential of an equilibrium liquid-gas system obtained by the HoldychQ approach LB simulation are compared to the analytical solutions. The system has $\kappa = 0.1$, $n_c = 1$, $p_c = 0.125$, $\beta\tau = 0.03$. For the LB simulation, $\omega = 1.0$.

in equilibrium by the ForcingQ approach agrees well with the theoretical value, but the errors are slight larger than that by the HoldychQ approach. The pressures of the two bulk phases by the LB were almost equal, but spikes occurred at the interface. The amplitude of the spikes was less than 0.5% of the average bulk pressure. The difference of average pressure by ForcingQ and the analytical solution was less than 0.1%. The analysis and corrections of the errors were given by Wagner [55]. The chemical potentials obtained with the ForcingQ approach were almost equal in the bulk phases, but small spikes occurred at the interfaces. The amplitude of the chemical potential spikes was less than 0.3% of the average bulk chemical potential. The difference of the average chemical potential by the ForcingQ and the theoretical value is less than 0.1%.

In summary, the phase separation in a liquid-gas system were successfully simulated with Holdych and Forcing approaches. In equilibrium, the density profile, the pressure, and the chemical potential were very close to the theoretical values. The interface density profile obtained by the Holdych approach, however, is closer to the theoretical

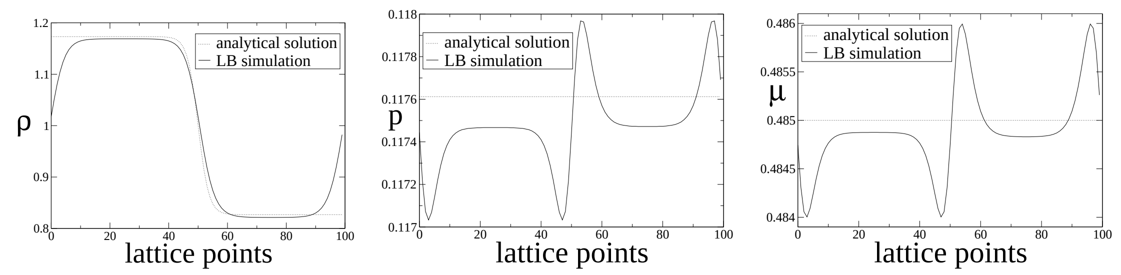


Figure 3.3. The density, pressure, and the chemical potential of an equilibrium liquid-gas system obtained by the ForcingQ approach LB simulation are compared to the analytical solutions. The system has $\kappa = 0.1$, $n_c = 1$, $p_c = 0.125$, $\beta\tau = 0.03$. For the LB simulation, $\omega = 1.0$.

potential than that obtained by the Forcing approach. In the next section, I extend the LBM to multicomponent systems.

3.2. Lattice Boltzmann for Multicomponent Systems

A multicomponent system may contain arbitrary number of components. In a multicomponent system, the continuity equation and the Navier Stokes equation are valid for the overall fluid. For each component, the continuity equation is still valid because mass conservation is still valid for each component. However, there is only one Navier Stokes equation for the total momentum because the momentum of each component is not conserved. Expressing the continuity equation in terms of the mean velocity I obtain a convection diffusion equation for each component as shown in Section 2.2.

With reference to my LB theory for a one component system, I established the a symmetric LB theory of a multicomponent system that can recover the continuity equation and the NSE for the overall fluid and can recover the convection diffusion equation for each component. I started my derivation of the hydrodynamic equations for a multicomponent system from the LB equation of one component. In analogy to Eq. (3.1), the LBE for a component σ of a multicomponent system is given by

$$f_i^\sigma(\mathbf{r}+\mathbf{v_i}\Delta t, t+\Delta t) - f_i^\sigma(\mathbf{r},t) = \Delta t \left(\frac{1}{\tau}(f_i^{\sigma e}(\mathbf{r},t) - f_i^\sigma(\mathbf{r},t) + G_i^\sigma) + F_i^\sigma \right), \tag{3.56}$$

where $f_i^\sigma(\mathbf{r},t)$ is the particle distribution function with velocity $\mathbf{v}_i$ for component σ; $f^{\sigma e}(\mathbf{r},t)$ is its equilibrium distribution; G_i^σ reflects the non-ideal effect due to pressure tensor on component σ; and F_i^σ is the forcing term of component σ due to the mean potential field generated by the interaction of the component σ with the other components.

The density of each component and the total density are given by

$$\rho^\sigma = \sum_i f_i^\sigma, \tag{3.57}$$

$$\rho = \sum_\sigma \rho^\sigma. \tag{3.58}$$

The average velocity of one component σ and the overall fluid can be defined as

$$\rho^\sigma u_\alpha^\sigma \equiv \sum_i f^\sigma v_{i\alpha}, \tag{3.59}$$

$$\rho u_\alpha \equiv \sum_\sigma \rho^\sigma u_\alpha^\sigma, \tag{3.60}$$

where u_α^σ is the average velocity of the component σ, and u_α is the average velocity of the overall fluid.

The moments of equilibrium distributions for one component are

$$\begin{aligned}
\sum_i f_i^{\sigma e} &= \rho^\sigma, \\
\sum_i f_i^{\sigma e} v_{i\alpha} &= \rho^\sigma u_\alpha, \\
\sum_i f^{\sigma e} v_{i\alpha} v_{i\beta} &= \frac{\rho^\sigma}{3}\delta_{\alpha\beta} + \rho^\sigma u_\alpha u_\beta, \\
\sum_i f^{\sigma e} v_{i\alpha} v_{i\beta} v_{i\gamma} &= \frac{\rho^\sigma}{3}(u_\alpha \delta_{\alpha\beta} + u_\beta \delta_{\alpha\gamma} + u_\gamma \delta_{\alpha\beta}) + \rho^\sigma u_\alpha u_\beta u_\gamma + Q^\sigma_{\alpha\beta\gamma}, \qquad (3.61)
\end{aligned}$$

where $Q^\sigma_{\alpha\beta\gamma}$ can be chosen to be $-\rho^\sigma u_\alpha u_\beta u_\gamma$ for the same reason as in the one-component system as shown in Section 3.1.

The moments for the forcing terms of one component are

$$\begin{aligned}
\sum_i F_i^\sigma &= 0 \qquad (3.62) \\
\sum_i F_i^\sigma v_{i\alpha} &= \rho^\sigma a^\sigma_\alpha, \qquad (3.63) \\
\sum_i F_i^\sigma v_{i\alpha} v_{i\beta} &= \rho^\sigma (a^\sigma_\alpha u^\sigma_\beta + a^\sigma_\beta u^\sigma_\alpha), \qquad (3.64) \\
\sum_i F_i^\sigma v_{i\alpha} v_{i\beta} v_{i\gamma} &= \frac{1}{3}\rho^\sigma (a^\sigma_\alpha \delta_{\beta\gamma} + a^\sigma_\beta \delta_{\alpha\beta} + a^\sigma_\gamma \delta_{\alpha\beta}). \qquad (3.65)
\end{aligned}$$

The moments for the pressure tensor terms of one-component are

$$\begin{aligned}
\sum_i G_i^\sigma &= 0, \\
\sum_i G_i^\sigma v_{i\alpha} &= 0, \\
\sum_i G_i^\sigma v_{i\alpha} v_{i\beta} &= A^\sigma_{\alpha\beta}, \\
\sum_i G_i^\sigma v_{i\alpha} v_{i\beta} v_{i\gamma} &= 0. \qquad (3.66)
\end{aligned}$$

To utilize my analysis of the one component system I can establish a LB equation for the total density by defining

$$\begin{aligned}\sum_\sigma f_i^\sigma &= f_i,\\ \sum_\sigma F_i^\sigma &= F_i,\\ \sum_\sigma G_i^\sigma &= G_i,\\ \sum_\sigma A_{\alpha\beta}^\sigma &= A_{\alpha\beta},\\ \sum_\sigma \rho^\sigma a_\alpha^\sigma &= \rho a_\alpha.\end{aligned} \tag{3.67}$$

Similar to the counterparts of the one-component system, the moments for the overall equilibrium distribution function are given by

$$\begin{aligned}\sum_i f_i^e &= \rho,\\ \sum_i f_i^e v_{i\alpha} &= \rho u_\alpha,\\ \sum_i f_i^e v_{i\alpha} v_{i\beta} &= \frac{1}{3}\rho\delta_{\alpha\beta} + \rho u_\alpha u_\beta,\\ \sum_i f_i^e v_{i\alpha} v_{i\beta} v_{i\gamma} &= \frac{1}{3}\rho(u_\alpha\delta_{\beta\gamma} + u_\beta\delta_{\alpha\gamma} + u_\gamma\delta_{\alpha\beta}) + \rho u_\alpha u_\beta u_\gamma + Q_{\alpha\beta\gamma}.\end{aligned} \tag{3.68}$$

This is identical to the one-component cases of Eqs. (3.2)-(3.5). The moments for the overall force terms are then given by

$$\begin{aligned}\sum_i F_i &= 0,\\ \sum_i F_i v_{i\alpha} &= \rho a_\alpha,\end{aligned}$$

Using Eq. (3.64), I obtain

$$\begin{aligned}\sum_i F_i v_{i\alpha} v_{i\beta} &= \sum_\sigma (a^\sigma_\alpha u^\sigma_\beta + a_\beta u^\sigma_\alpha)\\ &= \rho^\sigma (a^\sigma_\alpha (u_\beta + \Delta u^\sigma_\beta + a_\beta (u_\alpha + \Delta u^\sigma_\alpha))\\ &= \rho(a_\alpha u_\beta + a_\beta u_\alpha) + \sum_\sigma (a^\sigma_\alpha \Delta u^\sigma_\beta + a^\sigma_\beta \Delta u^\sigma_\alpha), \qquad (3.69)\end{aligned}$$

where the second term of Eq. (3.69) is of a higher order smallness than the first terms, and therefore does not enter the hydrodynamic equations to second order.

$$\sum_i F_i v_{i\alpha} v_{i\beta} v_{i\gamma} = \frac{1}{3}\rho(a_\alpha \delta_{\beta\gamma} + a_\beta \delta_{\alpha\beta} + a_\gamma \delta_{\alpha\beta}). \qquad (3.70)$$

The moments for the overall pressure terms are then given by

$$\begin{aligned}\sum_i G_i &= 0,\\ \sum_i G_i v_{i\alpha} &= 0,\\ \sum_i G_i v_{i\alpha} v_{i\beta} &= A_{\alpha\beta},\\ \sum_i G_i v_{i\alpha} v_{i\beta} v_{i\gamma} &= 0. \qquad (3.71)\end{aligned}$$

By summing Eq. (3.56) over σ, an effective LB equation for the total density is

$$f_i(\mathbf{r} + \mathbf{v_i}\Delta t, t + \Delta t) - f_i(\mathbf{r}, t) = \Delta t \left(\frac{1}{\tau}(f_i^e(\mathbf{r}, t) - f_i(\mathbf{r}, t) + G_i) + F_i \right), \tag{3.72}$$

which is identical to Eq. (3.1). Therefore, the continuity equation and the Navier Stokes equation of the overall fluid of a multicomponent system are identical to those of a one-component system. They are

$$\partial_t \rho + \partial_\alpha(\rho U_\alpha) = 0 + O(\epsilon^3), \tag{3.73}$$

where $U_\alpha \equiv u_\alpha + a_\alpha \Delta t/2$ is the macroscopic velocity of the fluid. The Navier Stokes equation for the overall fluid is:

$$\begin{aligned} \partial_t(\rho U_\beta) + \partial_\alpha(\rho U_\alpha U_\beta) &= -\partial_\alpha \left(\frac{1}{3}\rho\delta_{\alpha\beta} + A_{\alpha\beta} \right) + \partial_\alpha(\frac{w}{3}\rho(\partial_\alpha U_\beta + \partial_\beta U_\alpha)) + \rho a_\beta \\ &+ w\partial_\gamma(-u_\gamma \partial_\alpha A_{\alpha\beta} - u_\beta \partial_\alpha A_{\alpha\gamma} + \partial_t A_{\beta\gamma} + \partial_\alpha Q_{\alpha\beta\gamma}). \end{aligned} \tag{3.74}$$

s To recover the convection diffusion equation of each component, I did a Taylor expansion to the left of Eq. (3.56), and referring to Eq. (3.15),

$$\sum_{k=1}^{\infty} \frac{(\Delta t)^k}{k!} (\partial_t + v_{i\alpha}\partial_\alpha)^k f_i^\sigma = \Delta t \left(\frac{1}{\tau}(f_i^{\sigma e} - f_i^\sigma + G_i^\sigma) + F_i^\sigma \right). \tag{3.75}$$

To $O(\epsilon)$, referring to Eq. (3.16), it followed that

$$\Delta t(\partial_t + v_{i\alpha}\partial_\alpha) f_i^\sigma + O(\epsilon^2) = \Delta t \left(\frac{1}{\tau}(f_i^{\sigma e} - f_i^\sigma + G_i^\sigma) + F_i^\sigma \right). \tag{3.76}$$

To the order of $O(\epsilon^2)$, referring to Eq. (3.17),

$$\begin{aligned} & \Delta t(\partial_t + v_{i\alpha}\partial_\alpha) f_i^\sigma + \frac{(\Delta t)^2}{2}(\partial_t + v_{i\alpha}\partial_\alpha)^2 f_i^\sigma + O(\epsilon^3) \\ = \quad & \Delta t \left(\frac{1}{\tau}(f_i^{\sigma e} - f_i^\sigma + G_i^\sigma) + F_i^\sigma \right). \end{aligned} \tag{3.77}$$

Because of the recursive nature of Eq. (3.76), f_i^σ can be expressed by $f_i^{\sigma e}$ and derivatives of $f_i^{\sigma e}$; referring to Eq. (3.18),

$$f_i^\sigma = f_i^{\sigma e} + G_i^\sigma + \tau F_i^\sigma - \tau(\partial_t + v_{i\alpha}\partial_\alpha)(f_i^{\sigma e} + G_i^\sigma + \tau F_i^\sigma) + O(\epsilon^2). \tag{3.78}$$

Substituting Eq. (3.78) into the left side of Eq. (3.77). I obtained, as in Eq. (3.19),

$$\begin{aligned} & (\partial_t + v_{i\alpha}\partial_\alpha)(f_i^{\sigma e} + G_i^\sigma + \tau F_i^\sigma) - w(\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^{\sigma e} + G_{\sigma i} + \tau F_i^\sigma) + O(\epsilon^3). \\ = \quad & \frac{1}{\tau}(f_i^{\sigma e} + G_i^\sigma + \tau F_i^\sigma - f_i^\sigma). \end{aligned} \tag{3.79}$$

By summing over i, Eq. (3.79) gave,

$$\partial_t \rho^\sigma + \partial_\alpha(\rho^\sigma u_\alpha) + \tau\partial_\alpha(\rho^\sigma a_\alpha^\sigma) - w\sum_i (\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^{\sigma e} + G_i^\sigma + \tau F_i^\sigma) + O(\epsilon^3) = 0, \tag{3.80}$$

which is equivalent to Eq. (3.20). However the moments of f_i^e and $f_i^{\sigma e}$ are not identical here, so the continuity equation cannot be obtained. Eq. (3.80) shows that $\partial_t\rho^\sigma + \partial_\alpha(\rho^\sigma u_\alpha) + \tau\partial_\alpha(\rho^\sigma a_\alpha^\sigma)$ is of order $O(\epsilon^2)$, and F_i^σ is of order $O(\epsilon)$ as discussed in Section

3.1. Therefore $\partial_t\rho^\sigma + \partial_\alpha(\rho^\sigma u_\alpha)$ is of order $O(\epsilon^2)$, and the result is

$$\begin{aligned}
& w\sum_i(\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^{\sigma e} + G_i^\sigma + \tau F_i^\sigma) \\
= \ & w\sum_i(\partial_t + v_{i\alpha}\partial_\alpha)^2(f_i^{\sigma e} + G_i^\sigma) \\
= \ & w\sum_i(\partial_t + v_{i\alpha}\partial_\alpha)(\partial_t f_i^{\sigma e} + v_{i\alpha} f_i^{\sigma e} + \partial_\alpha G_i^\alpha v_{i\alpha}) \\
= \ & w\partial_t(\partial_t\sum_i f_i^{\sigma e} + \partial_\alpha\sum_i f_i^{\sigma e}v_{i\alpha} + \partial_\alpha\sum_i G_i^\sigma v_{i\alpha}) \\
& + w\partial_\beta(\partial_t\sum_i f_i^{\sigma e}v_{i\beta} + \partial_\alpha\sum_i f_i^{\sigma e}v_{i\alpha}v_{i\beta} + \partial_\alpha\sum_i G_i^\sigma v_{i\alpha}v_{i\beta}) \\
= \ & w\partial_\beta(\partial_t(\rho^\sigma u_\beta) + \partial_\alpha(\frac{\rho^\sigma}{3} + \rho^\sigma u_\alpha u_\beta) + \partial_\alpha A_{\alpha\beta}) + O(\epsilon^3).
\end{aligned}$$

So Eq. (3.80) can be simplified to

$$\begin{aligned}
& \partial_t\rho^\sigma + \partial_\alpha(\rho^\sigma u_\alpha) + \tau\partial_\alpha\rho^\sigma a_\alpha^\sigma \\
& -w\partial_\beta[\partial_t(\rho^\sigma u_\beta) + \partial_\alpha(\frac{\rho^\sigma}{3} + \rho^\sigma u_\alpha u_\beta) + \partial_\alpha A_{\alpha\beta}] + O(\epsilon^3) = 0.
\end{aligned} \tag{3.81}$$

Using Eqs. (3.74) and (3.73) I obtained with some algebra,

$$\partial_t U_\beta = -U_\alpha\partial_\alpha U_\beta - \frac{1}{\rho}\partial_\alpha\left(\frac{\rho}{3}\delta_{\alpha\beta} + A_{\alpha\beta}\right) + a_\beta + O(\epsilon^2). \tag{3.82}$$

From Eq. (3.80) if follows that

$$\partial_t\rho^\sigma = -\partial_\alpha(\rho^\sigma U_\alpha) + O(\epsilon^2). \tag{3.83}$$

With Eq. (3.82) and (3.83) it can be derived that

$$\partial_t(\rho^\sigma u_\beta) = -\partial_\alpha(\rho^\sigma U_\alpha U_\beta) - \frac{\rho^\sigma}{\rho}\partial_\alpha\left(\frac{\rho}{3}\delta_{\alpha\beta} + A_{\alpha\beta}\right) + \rho^\sigma a_\beta + O(\epsilon^2). \tag{3.84}$$

Substituting Eq. (3.84) into Eq. (3.81) results in,

$$\begin{aligned} & \partial_t\rho^\sigma + \partial_\alpha(\rho^\sigma U_\alpha) \\ = \; & w\partial_\beta\left[-\frac{\rho^\sigma}{\rho}\partial_\alpha\left(\frac{\rho}{3}\delta_{\alpha\beta} + A_{\alpha\beta}\right) + \partial_\alpha\left(A^\sigma_{\alpha\beta} + \frac{\rho^\sigma}{3}\delta_{\alpha\beta}\right)\right] \\ & + \tau\partial_\alpha\left(\frac{\rho^\sigma}{\rho}\rho a_\alpha - \rho^\sigma a^\sigma_\alpha\right). \end{aligned} \tag{3.85}$$

I used the force approach for my simulation because the force $\rho^\sigma\nabla\mu^\sigma$ is easily obtained. For a pressure approach we would need to identify a non-ideal pressure P^σ of component σ. However in general there is no pressure term with $\nabla P^\sigma = \rho^\sigma\nabla\mu^\sigma$. From the free energy density I derived the chemical potential with Eq. (A.13). Then I needed to obtain the non-ideal interaction, from which the forcing terms in the evolution equation can be derived with Eqs. (3.50), (3.51), and (3.52). In the force approach approach, I set $A_{\alpha\beta} = 0$, and Eq. (3.85) becomes

$$\partial_t\rho^\sigma + \partial_\alpha(\rho^\sigma U_\alpha) = \partial_\alpha\left[-\tau\rho^\sigma a^\sigma_\alpha + \tau\rho^\sigma a_\alpha - w\frac{\rho^\sigma}{\rho}\partial_\alpha\left(\frac{\rho}{3}\right) + w\partial_\alpha\left(\frac{\rho^\sigma}{3}\right)\right], \tag{3.86}$$

where $\rho^\sigma a^\sigma_\alpha$ is the non-ideal aspect of the force on component σ. In the LB model, $\rho^\sigma a^\sigma_\alpha$

can be defined as

$$\rho^\sigma a^\sigma_\alpha \equiv -\frac{w}{\tau}\rho^\sigma \partial_\alpha(\mu^\sigma - \frac{1}{3}\ln\rho^\sigma) = -\frac{w}{\tau}\left[\rho^\sigma\partial_\alpha\mu^\sigma - \frac{1}{3}\partial_\alpha\rho^\sigma\right], \tag{3.87}$$

where the coefficient is $\frac{w}{\tau} = 1 - \frac{\Delta t}{2\tau}$ due to the discretization of the LB model. This coefficient approaches 1 as Δt approaches 0, as one would expect from the continuum limit. Eq. (3.87) can be employed in my simulation to obtain the non-ideal interaction.

Plugging Eq. (3.87) into Eq. (3.86), with some algebra,

$$\partial_t\rho^\sigma + \partial_\alpha(\rho^\sigma U_\alpha) = \partial_\alpha\left[w\frac{\rho^\sigma}{\rho}\sum_{\sigma'}\rho^{\sigma'}\partial_\alpha(\mu^\sigma - \mu^{\sigma'})\right], \tag{3.88}$$

where σ' and σ refer to components.

Referring to Eq. (2.13) and Eq. (2.14), I recognized that Eq. (3.88) is the convection diffusion equation, and the diffusion flux of component σ is

$$J^{\sigma d} = -w\frac{\rho^\sigma}{\rho}\sum_{\sigma'}\rho^{\sigma'}\partial_\alpha(\mu^\sigma - \mu^{\sigma'}). \tag{3.89}$$

By comparing Eq. (3.89) to Eq. (2.13), it is clear that the w in Eq. (3.89) is equivalent to $k^{\sigma\sigma'}$ in Eq. (2.13). Furthermore, in my current approach, w is the same for all $k^{\sigma\sigma'}$. Eq. (3.89) is thermodynamically consistent in that the diffusion flux of each component is 0 in equilibrium when the chemical potential of each component is constant. Clearly the extension of the model to allow different values for w^σ for each component deserves future research.

3.3. Simulations of Multicomponent Systems

To validate my LB algorithm, I simulated phase separation in binary and ternary systems. The binodal lines obtained by my algorithm was compared with the theoretical ones obtained by minimizing the free energy. I used the interfacial tension parameter $\kappa = 0$ in all my LB simulations of binary and ternary systems, because there is an intrinsic surface tension in the LB simulation due to higher order terms [55], which did not appear explicitly in the second order Taylor expansion presented in this thesis. This choice can both simplify my simulation and avoid the unnecessary simulation instability caused by a possible improper choice of κ.

3.3.1. Simulations of binary systems

To validate my algorithm for binary systems, I performed LB simulations of binary systems. I then compared the equilibrium properties of those simulations to my theoretical predictions. The theoretical binodal lines of a binary system are obtained in Section 2.5.1. The free energy density is given by Eq. (2.45). The critical point of a binary system is given by Eqs. (2.48) and (2.48). Starting from the critical point, moving the initial state by increasing χ and keeping the volume fraction constant, I obtained a pair of binodal points for each initial condition. The collection of all the binodal points formed the binodal line. The theoretical equilibrium chemical potentials of the two components of a binary system were calculated by using Eqs. (A.9) and (A.10). Each component had two equal chemical potentials at the two phases to the precision of at least 10^{-6}. This indicated that my algorithm had determined the equilibrium densities. The pressures of the systems in equilibrium were calculated by substituting the chemical potentials into Eq. (2.17). However, the pressure of the overall fluid obtained by Eq. (2.17) is zero, which indicated that the Flory-Huggins free energy model may not be a good choice for the hydrodynamic simulation of a binary system. Since I was only concerned with one dimensional systems, hydrodynamic effects did not enter significantly into my simulations.

I performed LB simulations with two binary systems: a monomer system with $m^A = 1$ and $m^B = 1$, and a polymer system with $m^A = 10$ and $m^B = 1$. For both systems, the total density was $\rho = 100$. The critical volume fractions for the monomer system are $\phi^A = 0.5$ and $\phi^B = 0.5$ and for the polymer systems are $\phi^A = 0.24$ and $\phi^B = 0.76$. To induce phase separation a small perturbation is to be introduced in the initial conditions. The amplitude of the perturbation is 0.1 and its wavelength is the lattice size. The initial volume fraction of component A is given by $\phi^A(x) = \phi^{A0} + \Delta\phi(x)$. The initial volume fraction of component B is given by $\phi^B(x) = \phi^{B0} - \Delta\phi(x)$. The initial volume fraction of component C is obtained naturally from $\phi^C(x) = 1 - \phi^A(x) - \phi^B(x)$, so initially $\phi^C(x) = 1 - \phi^{A0}(x) - \phi^{B0}(x)$ (Appendix K).

The monomer system was simulated with different reverse relaxation times $\Omega = 0.7, 0.8,$ and 0.9 to verify that my algorithm converges for different relaxation times. The polymer system was simulated with only one inverse relaxation time of $\Omega = 0.9$. Starting from the critical point and increasing the χ value for each initial condition until the simulations were numerically unstable, I obtained a pair of binodal points for each initial condition. The system reached a stable state after about 5000 time steps. My measurement were taken after 50000 time steps to be sure that an equilibrium state

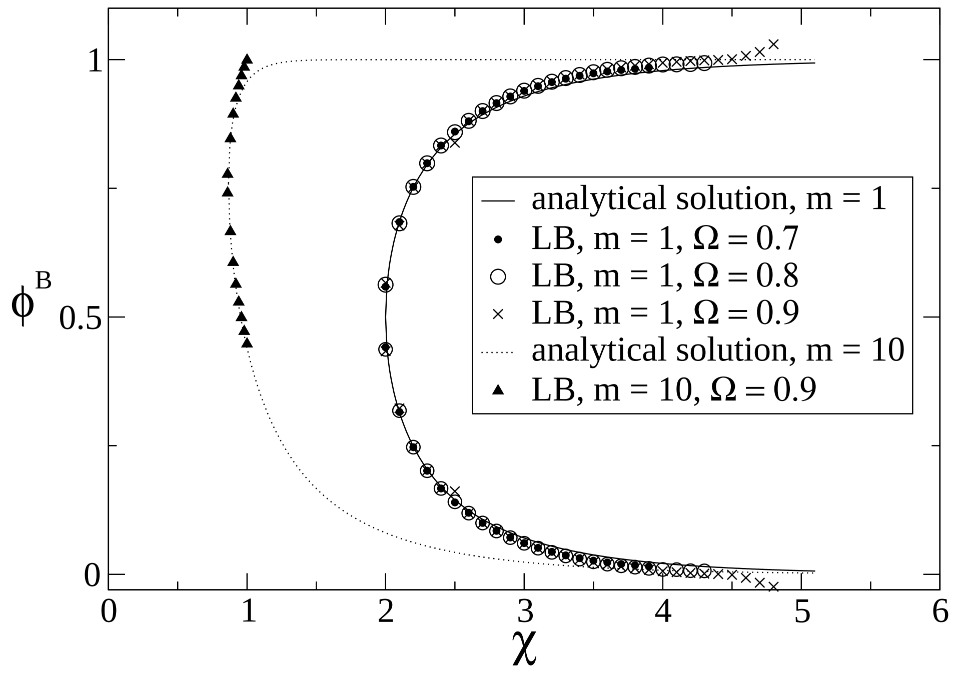


Figure 3.4. The binodal points obtained by the LB simulation are compared to the theoretical solutions. For $m^A = 1, m^B = 1$, the binodal points by LB with different ω converge at the theoretical values. For $m^A = 10, m^B = 1$, the binodal points by LB with $\Omega = 0.9$ match the binodal points by minimizing free energy.

had been reached.

Figure 3.4 shows that the binodal lines of the monomer system at different Ω almost overlap with the theoretical values. The only notable difference is that the stability ranges vary slightly with Ω. The simulations tended to be unstable when one component was nearly depleted ($\phi^\sigma \to 0$).

For the polymer system, Figure 3.5 shows the comparison of the total density, pressure, and the volume fractions and chemical potentials of each component to the corresponding theoretical values. The total density of a system in equilibrium by LB is nearly constant, but a slight density difference exists for two phases and spikes occur

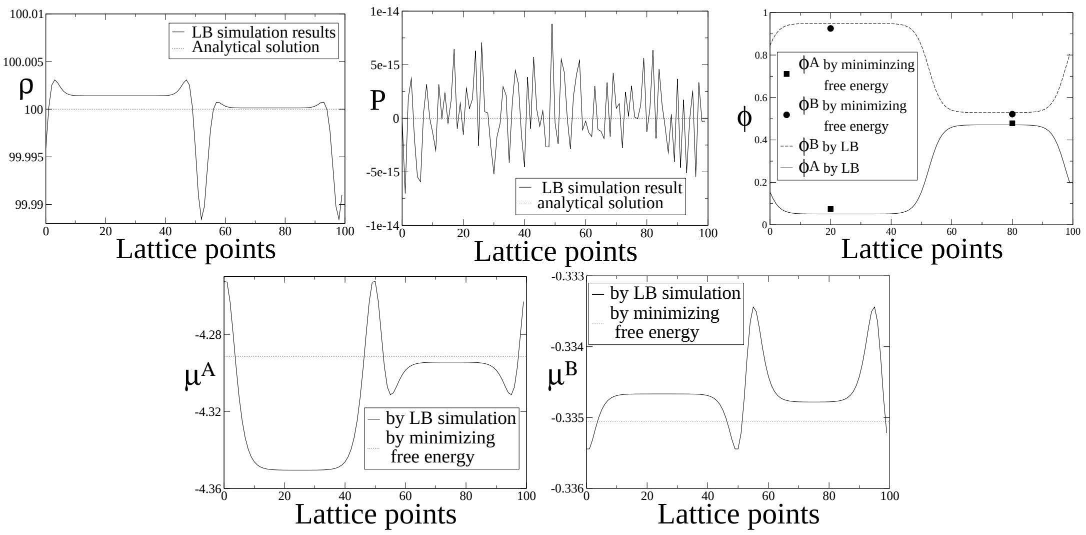


Figure 3.5. The total density, pressure, volume fraction and the chemical potentials of the two components of a binary system by LB simulation are compared to the theoretical values. The system has $m^A = 10$, $m^B = 1$, $\chi = 0.94$, $\kappa = 0$. For LB simulation, $\omega = 0.9$.

at the interface between the two phases. The Flory-Huggins model assumes constant total density but the total density in LB simulation is only nearly constant. This may account for the discrepancy of the simulation results and the theoretical values. The volume fractions of each component by LB simulation agrees well with the theoretical values. The pressure of the system by the LB simulation is 0 to the machine precision in agreement with the theoretical value.

The chemical potential of each component by the LB simulation was very close to the theoretical value. Chemical potential μ_A of the two bulk phases differed slightly while μ_B of the two bulk phases are nearly the same. Spikes existed at the interface of the two bulk phases, the amplitude of which were less than 3% of the theoretical value.

3.3.2. Simulations of ternary systems

To validate my algorithm for ternary systems, I perform LB simulations and compared the equilibrium results to the theoretical predictions. The theoretical binodal lines are obtained in Section 2.5.2. The theoretical equilibrium chemical potentials of the three components of a ternary system were calculated by using Eqs. (A.11), (A.12), and (A.13). The pressure of the systems in equilibrium was calculated by substituting the chemical potentials into Eq. (2.17). However, the pressure of the overall fluid obtained by Eq. (2.17) are again zero to numerical accuracy.

I performed LB simulations with two ternary systems: a monomer system with $m^A = 1$, $m^B = 1$, and $m^C = 1$ and a polymer system with $m^A = 10$, $m^B = 1$, and $m^C = 1$. The χ parameters for both systems were $\chi^{AB} = 3$, $\chi^{AC} = 0.5$, and $\chi^{BC} = 0.2$. The inverse relaxation time constant for both simulations was $\Omega = 0.9$. The critical point for the monomer system was $\phi^A = 0.32$, $\phi^B = 0.32$, and $\phi^C = 0.36$. The critical point for the polymer system was $\phi^A = 0.14$, $\phi^B = 0.11$, and $\phi^C = 0.75$. The initial state of each simulation were set from the critical points towards the end point ($\phi^A = 0.5$, $\phi^B = 0.5$, $\phi^C = 0$). Initially a small sinusoidal wave perturbation of an amplitude of 0.1 and wavelength of the lattice size was superimposed on the initial volume fractions of the components to induce phase separation. I performed a LB simulation for each set of initial volume fractions and obtained the volume fractions of the two phases in the equilibrium state, resulting in two binodal points. The simulation reached a stable state after about 20,000 time steps. The measurements were taken after 200,000 time steps to make sure the equilibrium state was reached.

Figure (3.6) shows the comparison of the binodal points by LB simulation to the theoretical binodal lines of both systems. The binodal points obtained by the LB simulation agree reasonably well with the theoretical binodal lines for the monomer and polymer systems. The simulation becomes unstable when ϕ^A is close to zero, i.e. when one component is nearly depleted. It is also noticeable that the binodal points obtained by the LB simulation deviate from the theoretical binodal lines when one component is nearly depleted. Immediately near the critical point, the evolution of the system becomes extremely slow so the slight deviation between the binodal points obtained through the LB simulation and the theoretical ones probably indicates that the LB simulation was not yet fully equilibrated.

Figure 3.7 shows the comparison of the total density, pressure, and the volume fractions and chemical potentials of each component. The total density of the system obtained by LB simulation is very close to the theoretical value except the spikes at the interface region. The amplitudes of these spikes are less than 1% of the theoretical density. The Flory-Huggins model assumes constant total density but the total density in LB simulations is only nearly constant. This may account for some of the discrepancy between the simulation results and the theoretical predictions. The volume fractions of each phase by LB simulation were very close to their theoretical values. The pressure of the system by the LB simulation is again 0 to the machine precision. The chemical potential of component A was slightly different in two phases, while the chemical potentials of components B and C were much closer in the two phases, although there were small spikes at the interface. The amplitude of the spikes are less than 2% of the theoretical value. The chemical potential by LB agrees with the theoretical value.

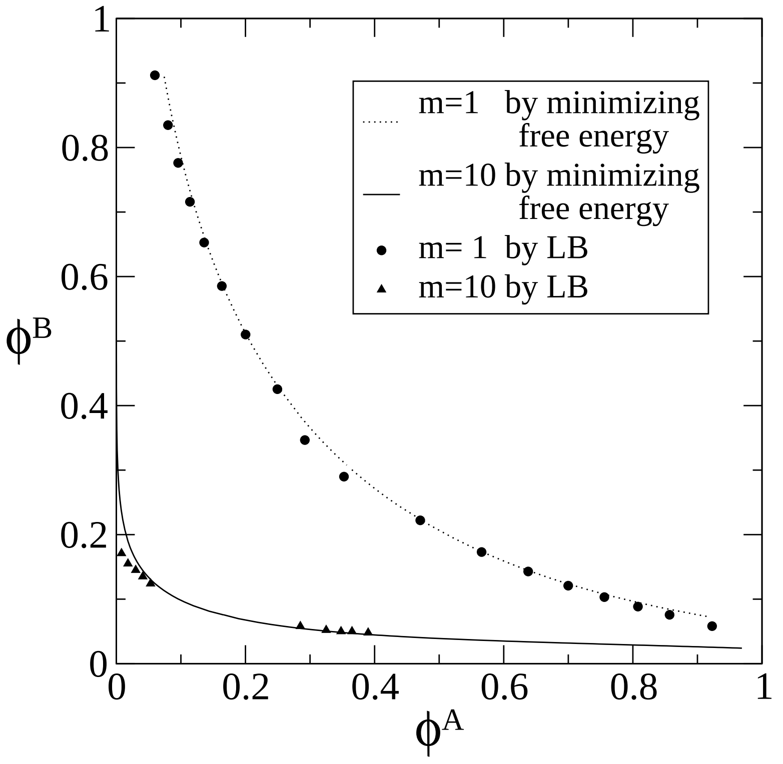


Figure 3.6. The binodal points of a ternary system obtained by LB simulation are compared to the theoretical values. One ternary system has $m^A = 1$, $m^B = 1$, and $m^C = 1$; the other has $m^A = 10$, $m^B = 1$, and $m^C = 1$. The parameters for both systems are χ^{AB} =3, $\chi^{AC} = 0.5$, $\chi^{BC} = 0.2$. $\Omega = 0.9$.

To sum up, the equilibrium properties of the simulations in binary and ternary systems validate my multicomponent LB algorithm for at least three components. The binodal points obtained with my algorithm of different relaxation times converged to the theoretical values. The pressure of the overall fluid was constant, and the chemical potential for each component was also constant in equilibrium. Therefore, my model was consistent with the expectations of equilibrium thermodynamics.

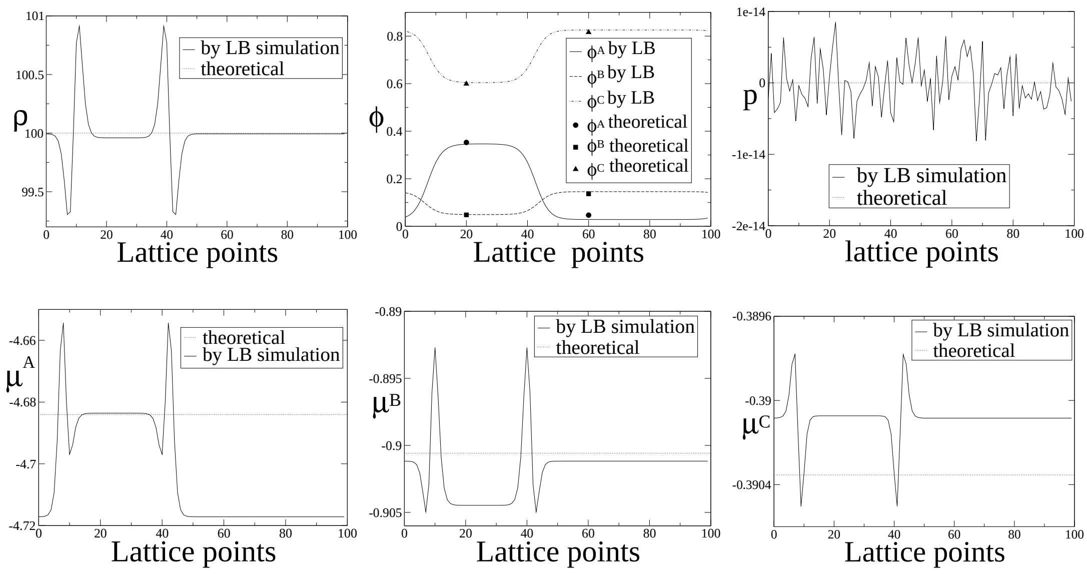


Figure 3.7. The total density, pressure, and volume fractions and the chemical potentials of each components of a ternary system by LB simulation are compared to the theoretical results obtained by minimizing the free energy.

CHAPTER 4

GALILEAN INVARIANCE OF THE LATTICE BOLTZMANN METHOD

4.1. Lattice Boltzmann Galilean Invariance Violations and Corrections

The Galilean Invariance Principle states that the fundamental laws of physics are the same in all inertial (uniform-velocity) frames of reference in the non-relativistic domain. Although the continuum Navier Stokes equation is Galilean invariant as shown in Appendix B, LB simulations suffer from Galilean invariance violations because the lattice frame is fixed and even the recovery of NSE in LB is not complete. Eq. (3.31) shows that the error derived from the incomplete recovery of NSE increases as the fluid velocity increases. Therefore, it is reasonable to deduce that the error of GI violation also increase with the relative speed between the stationary frame and the moving frame. Both the ideal gas LB model and the non-ideal gas LB model suffer from GI violations, but the error sources for the two models are different. As shown in Eq. (3.31) and Eq. (3.74), the ideal gas LB model suffers from GI violations because of the $w\partial_\gamma\partial_\beta Q_{\alpha\beta\gamma}$ term, which was obvious when both $\mathbf{a}$ and A were set to zero. For the non-ideal LB simulation, some extra terms were introduced through the A term. This was especially the case when a non-ideal LB model was employed to simulate phase separation. Because near equilibrium at the interface region, the density gradient is much larger than the pressure gradient thus contributes more to the Galilean invariance violation through the density gradient in the A term.

The GI violation of the LBM is not always serious and may be acceptable in some applications. But in other applications, it can cause problems. Recently, research has investigated Lees-Edwards-like boundary conditions in lattice Boltzmann simulations of sheared systems [58]. It turns out that these boundary conditions were problematic [10], and I presume that Galilean invariance violations may be to blame. This motivated me to investigate the GI of LB in detail.

To begin my analysis of the GI violation of LB, I introduced a criterion which can be used to judge if an LB model is Galilean invariant and, if not, which terms cause the errors. For any LB model that is Galilean invariant, its equilibrium density distribution function must satisfy the following equation [8]:

$$M_j(\mathbf{u}) = M_j(\mathbf{0}), \tag{4.1}$$

where the j order tensor $M_j(\mathbf{u})$ is defined as $M_j(\mathbf{u}) \equiv \sum_{i=1}^{b}(\mathbf{v}_i - \mathbf{u})^j f^{eq}(\mathbf{u})$; b is the number of discrete velocities; $\mathbf{v}$ is the velocity of the distribution; and $\mathbf{u}$ is the average velocity. For the purpose of GI judgement, terms of $j \leq 3$ are of most importance because we required those terms in Section 3.1. to derive the hydrodynamic equations. Although higher order terms do have some effect on the simulations, their effects are minor compared with those of the lower order terms. Buick used Eqs. (4.1) when he derived coefficients of an LB model [8], but it can be used to test the GI of an LB model. The following moments are easily obtained by substituting Eqs. (3.38) , (3.39), and (3.40) into Eq. (4.1),

$$M_0(\mathbf{0}) = \rho, \qquad M_0(\mathbf{u}) = \rho; \tag{4.2}$$

$$M_1(\mathbf{0})_\gamma = 0, \qquad M_1(\mathbf{u})_\gamma = 0; \tag{4.3}$$

$$M_2(\mathbf{0})_{\gamma\delta} = \rho/3, \qquad M_2(\mathbf{u})_{\gamma\delta} = \rho/3; \tag{4.4}$$

$$M_3(\mathbf{0})_{\gamma\delta\theta} = 0, \qquad M_3(\mathbf{u})_{\gamma\delta\theta} = -\rho u^3. \tag{4.5}$$

The result clearly shows the GI violation of the model because of the ρu^3 term in M_3.

I applied Eq. (4.1) to the D1Q3 model here because I use this model exclusively to verify my theoretical analysis. Actually, Eq. (4.1) is applicable to all LB models. The results of the application of Eq. (4.1) to additional ideal and non-ideal models are described in Appendix E.

Simulations are performed to test various approaches to GI violation corrections in Table 3.1. A sound wave propagation simulation was performed to check the effects of corrections in a one-phase scenario. A one-component phase separation simulation was performed to check the effects of the corrections in a two-phase scenario.

4.2. Galilean Invariance Corrections for Sound Waves

In this section, I analyze the simulation of one-dimensional isothermal sound wave propagation with LB ideal gas models. I simulated the sound wave propagation with an isothermal LB simulation because those are the models we are interested in, even though an adiabatic model would be more physical. The results of the simulations with and without GI corrections were compared to the analytical solution derived in Appendix C,

$$\rho = \rho_0 + \delta\rho_0 \exp(-c_s^2 k^2 (\tau - \Delta t/2) t) \sin(k(x - c_s \sqrt{1 - c_s^2 k^2 (\tau - \Delta t/2)^2}\ t\ )), \qquad (4.6)$$

where ρ is the density, ρ_0 is the constant density of medium in the absence of a sound wave, $\delta\rho_0$ is the initial amplitude of the density variation constituting the sound wave, c_s is the sound speed, k is the wave number, τ is the relaxation time of LB, and Δt is the time step which is 1 in LB simulation. The density of the ideal gas is $\rho_0 = 100$, and 3 wavelengths are in one frame. The initial condition for the simulation can be obtained by using continuity equation Eq.(2.1). The initial density of the system is $\rho[i] = 100 + \delta\rho_0 \sin(6\pi i/L)$, where i is the lattice site and L is the number of the total lattice sites. The initial velocity is $u[i] = (-c_s^2 k(\tau - \Delta t/2)\cos(kx) + c_s \delta\rho_0 \sin(6\pi i/L))/\rho[i] + u_0$, where u_0 is the velocity of the frame of the medium with respect to the fixed lattice. In my simulation $\delta\rho_0 = 1$. See Appendix C for the derivation.

To measure the effect of my Q correction on the GI violation, I introduced some measurement definitions. Because a sound wave causes a density variation on a medium, I defined the norm of the sound wave as $N \equiv \langle(\rho(i) - \rho_0(i))^2\rangle$, where i is the lattice site, $\rho(i)$ is the density of the medium with a sound wave, $\rho_0(i)$ is the density of medium in the absence of a sound wave. The absolute error is defined as $E \equiv \langle(\rho_{u_0}(i) - \rho(i))^2\rangle$, where ρ_{u_0} is the density when the frame on medium has a velocity u_0 with respect to the static frame on the lattice, $\rho(i)$ is density measured when u_0 is 0. When measured,

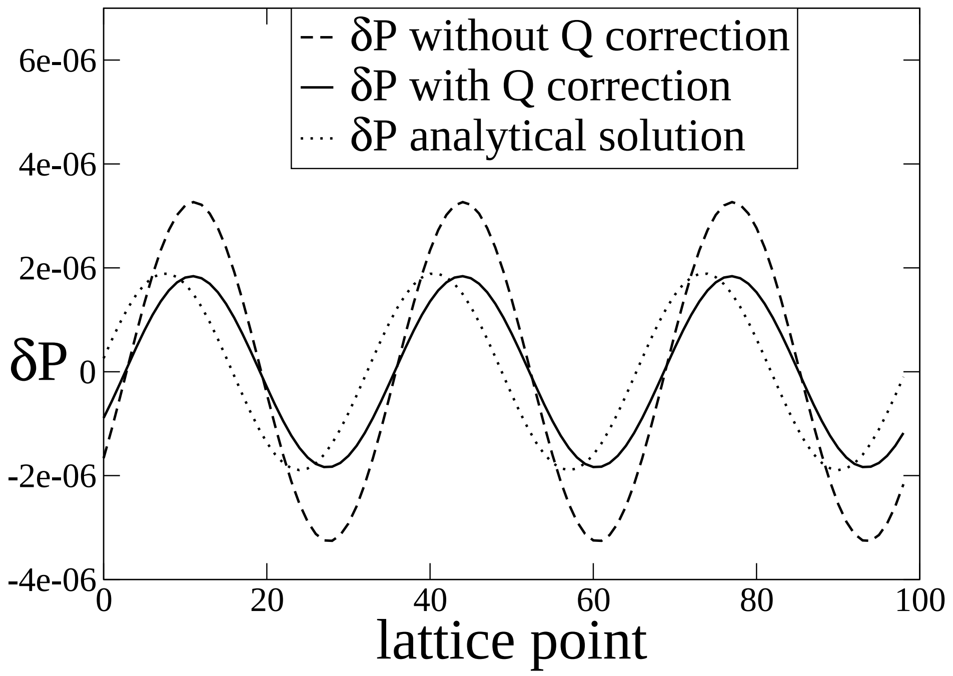


Figure 4.1. The waveforms of sound waves in an ideal gas obtained by the LB simulation with and without Q correction were compared to the theoretical values. The medium moved with a velocity of 0.1 with respect to the static frame.

$\rho_{u_0}(i)$ and $\rho(i)$ should be in the same phase with respect to i. The relative error is defined as $E_r \equiv \sqrt{E/N}$.

The attenuation of the sound waves in an ideal gas with and without Q correction are compared in Figure 4.1. The measurement was taken after 2000 iterations. The figure shows that the algorithm with correction gives a better result, namely it is closer to the analytical solution. Figure 4.1 also shows a small phase shift between the theoretical solution and the LB simulation. The source of which merits further investigation.

The errors and relative errors of the sound waves versus iterations with and without Q correction are compared in Figure 4.2. We see that both the error and relative error of the sound wave with correction are smaller than that without correction.

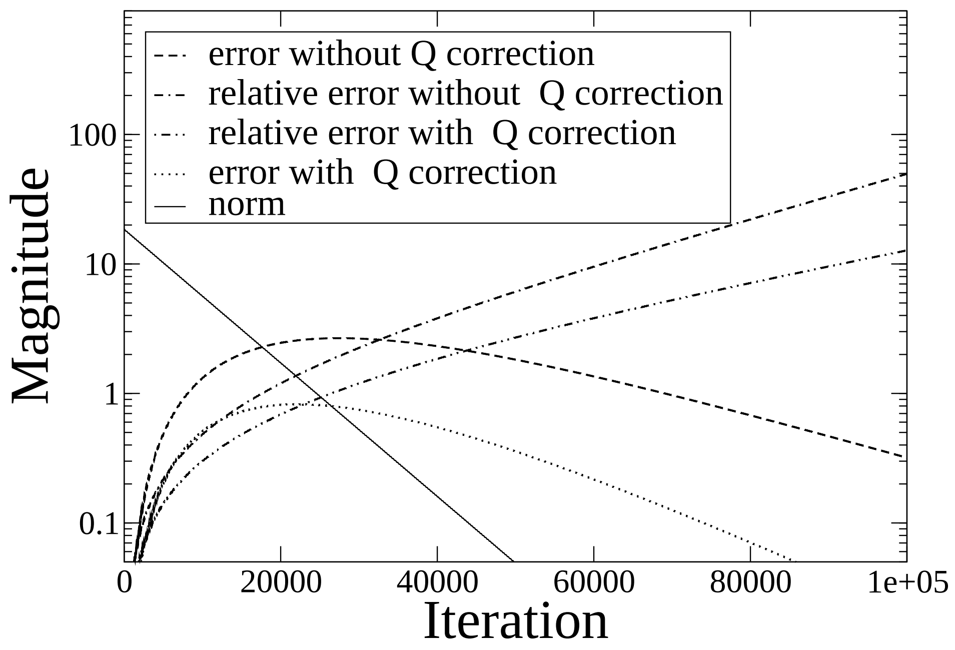


Figure 4.2. The error and relative error of a sound wave in an ideal gas obtained by the LB simulation with Q correction are compared to that without Q correction for $u_0 = 0.1$.

The simulation parameters of this simulation are the same as those in the attenuation simulation, except $u_0 = 0.1$.

The relative error of the sound waves with and without correction are also compared as a function of u_0 in Figure 4.3. The parameters for this simulation were the same as those in the attenuation simulation except u_0 changes from 0.01 to 4.29, and the iteration for each u_0 was 3000. Figure 4.3 shows that the relative error with correction is smaller than that without correction. For both algorithms, the relative errors increase with the velocity of the whole system.

All the simulation results show that the LB algorithm with correction performed better than the algorithm without correction. Next, the effect of the GI correction by the LB simulations in a system of two phases is checked.

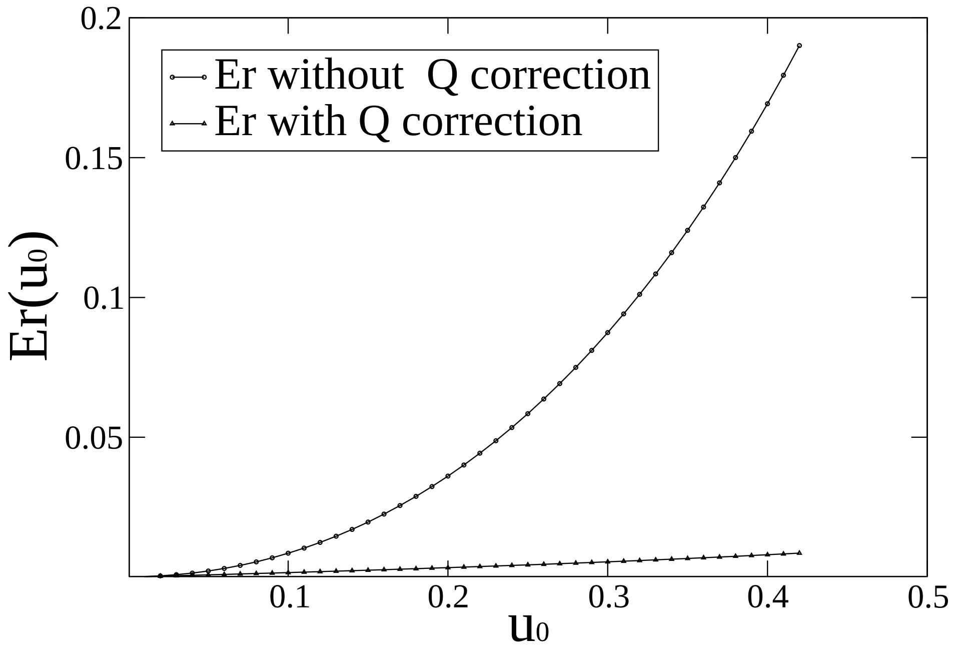


Figure 4.3. The relative errors of a sound wave in an ideal gas obtained by LB simulation with and without Q correction are compared for different u_0 after 3000 iterations.

4.3. Test of Correction in Liquid-Gas Phase Separation

To validate our GI violation correction approach, I simulated liquid-gas phase separation described by a Landau free energy model (Eq. 2.31). The thermodynamics of such a system has been discussed in Section 2.4.. The LB simulation results with and without Q term correction were compared with the analytical solutions.

In order to measure the effects of the GI violation in the LB simulation, I used two frames of reference: one was fixed on the lattice, and the other moved along with the fluid at a velocity u_0. For a stationary system, the density profile is given by a profile $\rho_0(x)$ and the velocity is $u(x) = 0$. When the same system is moving with velocity u_0, I expect a profile $\rho(x) = \rho_0(x - u_0 t)$ and a velocity $u(x) = u_0$. Any deviation of Galilean

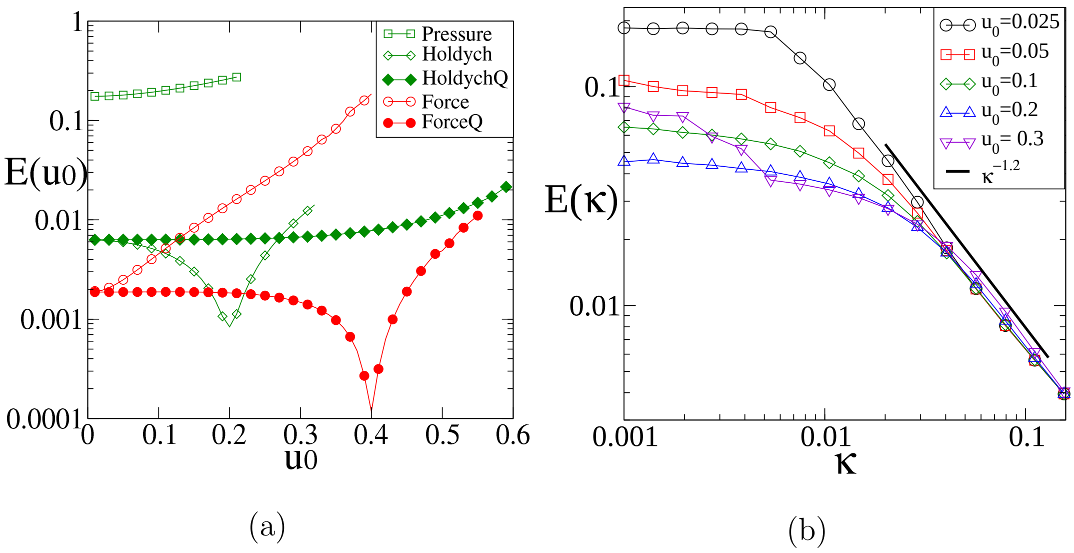


Figure 4.4. Galilean invariance errors $E(u_0)$ of a liquid-gas system with different LB approaches are compared through u_0 in (a) and κ in (b). The simulation parameters are $n_c = 1$, $p_c = 0.42$, $\theta = -0.03$. (a) $E(u_0)$ at $\kappa = 0.1$. (b) $E(\kappa)$ of the HoldychQ method for $u_0 = 0.025, 0.05, 0.1, 0.2, 0.3$.

invariant behavior shows up in deviations of the profiles of $\rho(x)$ and $u(x)$. Because the analytical solution for the velocity profile was translation invariant, it was easy to devise a measure of the Galilean invariance violation based on the velocity profile. I defined the dimensionless error function as

$$E(u_0) = \sqrt{\sum_i (u(i)/u_0 - 1)^2/L}\,. \tag{4.7}$$

The simulation showed that it typically takes less than 1000 iterations for the density profile of the system to reach a stable state. The measurement was taken after 10^5 iterations to be sure that the system was in an equilibrium state. The measurements of different correction approaches are shown in Figure 4.4.

Figure 4.4 (a) shows the Galilean invariance error $E(u_0)$ for the different methods of Table 3.1. The errors of pressure approach were the largest because the density profile in the LB simulation of this approach remained static at all values of u_0. The correction terms in the Holdych approach led to a significant improvement. The additional Q correction term in the HoldychQ method did not lead to a noticeable improvement, except for $u_0 \geq 0.3$, for which this method becomes more accurate and more stable. The Forcing approach led to a good performance at a small u_0, but the error increased exponentially with u_0, which can be expressed approximately as $E(u_0) = 10^{-3} \exp(11.65\, u_0)$. The Q correction term in the ForcingQ method yielded a significant improvement. For a small u_0, the Forcing and Holdych methods gave the same results as their Q corrected versions because the correction term was cubic in u_0 and, therefore, gave a negligible contribution to the forcing term.

When interpreting these results, it is important to note that the parameter space for the Galilean invariance problem includes not only u_0 but also the parameters determining the equilibrium density profile κ, θ, and p_c as well as the relaxation time τ. This parameter space is so large that it is not feasible to examine exhaustively here. Since the Galilean invariance violations are related to gradient terms, it is reasonable to assume that a wider interface leads to a smaller error. Because of the intrinsic surface tension in the Forcing Q approach, the thickness of the interface cannot be adjusted by changing the value of κ [55]. Therefore, only the HoldychQ method was employed for my simulation. I performed simulations for different interface widths. The interface width ξ is related to κ through $\xi \propto \kappa^{1/2}$ [7]. Figure 4.4 (b) shows that $\kappa^{-1.2} \propto \xi^{-2.4}$ for large κ. So increasing the interface width is another means by which Galilean invariance violations can be reduced. For thin interfaces the Galilean invariance violation was worse at smaller velocities. This effect is attributed to the pinning of the interface to the lattice since the interface can be more easily dislodged by a larger advective velocity.

To sum up, I have identified the Galilean invariance violation terms for both of the ideal gas LB models and the non-ideal fluid LB model in the Navier Stokes equation level. The corrections for both cases were presented. The simulations in one-phase and two-phase systems validated the corrected algorithms.

REFERENCES

[1] A. Akthakul, C.E. Scott, A.M. Mayes, and A.J. Wagner. Lattice Boltzmann simulation of asymmetric membrane formation by immersion precipitation. *J. Membrane Sci.*, 249:213–226, 2005.

[2] S. Ansumali and I. V. Karlin. Stabilization of the lattice Boltzmann method by the H theorem: A numerical test. *Phys. Rev. E*, 62:7999–8003, 2000.

[3] M.K. Banda, W.A. Yong, and A. Klar. A stability notation for lattice Boltzmann equations. *SIAM J. Sci. Comput.*, 27:2098–2111, 2006.

[4] B.F. Barton and A.J. Mchuch. Kinetics of thermally induced phase separation in ternary polymer solutions. I. modeling of phase separation dynamics. *J. of Polymer Sci.: Part B: Polymer Phys.*, 37:1449–1460, 1999.

[5] P.L. Bhatnagar, E.P. Gross, and M. Krook. A model for collision processes in gases. I. small amplitude processes in charged and neutral one-component system. *Phys. Rev.*, 94:511–525, 1954.

[6] R.M. Boom, T. Boomgaard, and C.A. Smolders. Mass transfer and thermodynamics during immersion precipitation for a two-polymer system: Evaluation with the system pes-pvp-nmp-water. *J. of Membrane Sci.*, 90:231–249, 1994.

[7] A.J. Briant, A.J. Wagner, and J.M. Yeomans. Lattice Boltzmann simulations of contact line motion. I. liquid-gas systems. *Phys. Rev. E*, 69:031602.1–031602.4, 2004.

[8] J.M. Buick. *Lattice Boltzmann methods in interfacial wave modeling.* Ph.D. dissertation, The University of Edinburgh, U.K., 1997.

[9] A.M.W. Bulte, E.M. Naafs, F. Mulder, and C.A. Smolders. Equilibrium thermodynamics of the ternary membrane-forming system nylon, formic acid and water. *Polyer*, 37:1647–1655, 1996.

[10] M.E. Cates, K. Stratford, R. Adhikari, P. Stansell, J-C.Desplat, I. Pagonabarraga, and A.J. Wagner. Simulating colloid hydrodynamics with lattice Boltzmann methods. *J. Phys.:Cond. Mat.*, 16:S3903–S3915, 2004.

[11] H. Chen, B.M. Boghosian, P.V. Coveney, and M. Nekovee. A ternary lattice Boltzmann model for amphiphilic fluids. *Proc. Roy. Soc. A*, 456:2043–2057, 2000.

[12] S. Chen, H. Chen, D. Martnez, and W. Matthaeus. Lattice Boltzmann model for simulation of magnetohydrodynamics. *Phys. Rev. Lett.*, 67:3776–3779, 1991.

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

[14] B. Deng, B. Shi, and G. Wang. A new lattice Bhatnagar-Gross-Krook model for the convection-diffusion equation with a source term. *Chin. Phys. Lett.*, 22:267–270, 2005.

[15] L. Djenidi. Lattice-Boltzmann simulation of grid-generated turbulence. *J. Fluid Mech.*, 552:13–35, 2006.

[16] M.M. Dupin, I. Halliday, and C.M. Care. Multi-component lattice Boltzmann equation for mesoscale blood flow. *J. Phys. A: Math. Gen.*, 36:8517–8534, 2003.

[17] U. Frisch, D. D'Humieres, B. Hasslacher, P. Lallemand, Y. Pomeau, and J.P. Rivert. Lattice gas hydrodynamics in two and three dimensions. *Complex Systems*, 1:649–707, 1987.

[18] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the Navier-Stokes equation. *Phys. Rev. Lett.*, 56:1505–1508, 1986.

[19] I.M. Gelfand and S.V. Fomin. *Calculus of Variations.* Prentice Hall, 1963.

[20] X. He, X. Shan, and G.D. Doolen. Discrete Boltzmann equation model for nonideal gases. *Phys. Rev. E*, 57:57.R13–57.R16, 1998.

[21] D.J. Holdych, D. Rovas, J.G. Georgiadis, and R.O. Buchius. An improved hydrodynamics formulation for multiphase flow lattice-Boltzmann models. *Int. J. Mod. Phys. C*, 9:1393–1404, 1998.

[22] T. Inamuro, N. Konishi, and F. Ogino. A Galilean invariant model of the lattice Boltzmann method for multiphase fluid flows using free-energy approach. *Comp. Phys. Comm.*, 129:32–45, 2000.

[23] R.A.L. Jones. *Soft Condensed Matter.* Oxford University Press, 2002.

[24] A.N. Kalarakis, V.N. Burganos, and A.C. Payatakes. Galilean-invariant lattice-Boltzmann simulation of liquid-vapor interface dynamics. *Phys.Rev. E*, 65:056702.1–056702.13, 2002.

[25] I.V. Karlin, A. Ferrante, and H.C. Ottinger. Perfect entropy functions of the lattice Boltzmann method. *Europhys. Lett.*, 47:182–188, 1999.

[26] I.V. Karlin, A.N. Gorban, S. Succi, and V. Boffi. Maximum entropy principle for lattice kinetic equations. *Phys. Rev. Lett.*, 81(1):6–9, 1998.

[27] W.F.C. Kools. *Membrane formation by phase inversion in multicomponent polymer systems.* Ph.D. dissertation, University of Twente, Netherland, Feb 1998.

[28] P.K. Kundu. *Fluid Mechanics.* Academic Press, 1990.

[29] A. Lamura, G. Gonnella, and J.M. Yeomans. A lattice Boltzmann model of ternary fluid mixtures. *Europhys. Lett.*, 45:314–320, 1999.

[30] L.D. Landau and E.M. Lifshitz. *Fluid Mechanics.* Pergamon Press, London, 1959.

[31] P. Landmand and L.-S. Luo. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance and stability. *Phys. Rev. E*, 61:6546–6562, 2000.

[32] K. Langaas. *Modeling of immiscible two-phase flow in porous dedia with the binary fluid lattice Boltzmann method.* Ph.D. dissertation, University of Bergen, 1999.

[33] L. Loo. Polymer solution thermodynamics. lecture note at www.che.utexas.edu/lloo/che355f05.htm, 2003.

[34] L.-S. Luo. Unified theory of lattice Boltzmann methods for nonideal gases. *Phys. Rev. E*, 62:1618–1621, 1998.

[35] L.-S. Luo. The lattice-gas and lattice Boltzmann methods: past, present, and future. http://research.nianet.org/ luo/Reprints-luo/2000/ACFD4-2000.pdf, 2000.

[36] L.-S. Luo. Theory of the lattice Boltzmann method: Lattice Boltzmann models for non-ideal gases. *Phys. Rev. E*, 62:4982–4996, 2000.

[37] X.D. Niu, C. Shu, Y.T. Chew, and T.G. Wang. Investigation of stability and hydrodynamics of different lattice Boltzmann models. *J. Stat. Phys.*, 117:665–680, 2004.

[38] W.R. Osborn, E. Orlandini, M. R. Swift, J.M. Yeomans, and J. R. Banavar. Lattice Boltzmann study of hydrodynamic spinodal decomposition. *Phys. Rev. Lett.*, 75:4031–4034, 1995.

[39] C.M. Pooley. *Mesoscopic modeling techniques for complex fluids.* Ph.D. dissertation, University of Oxford, Michaelmas Term, 2003.

[40] Y. Qian and Y. Zhou. Complete Galilean-invariant lattice BGK models for the Navier-Stokes equation. Technical Report 208702, ICASE Report No. 98-38, 1998.

[41] Y.H. Qian, D. D'Humieres, and P. Lallemand. Lattice BGK models for Navier-Stokes equation. *Europhys. Lett.*, 17:479–484, 1992.

[42] Y.H. Qian and S.A. Orszag. Lattice BGK models for the navier-stokes equation: Nonlinear deviation in compressible regimes. *Europhys. Lett.*, 21:255–259, 1993.

[43] D. Raabe. Overview of the lattice Boltzmann method for nano- and micro-scale fluid dynamics in materials science and engineering. *Modeling Simul. Mater. Sci. Eng.*, 12:R13–R46, 2004.

[44] X. Shan and H. Chen. Lattice Boltzmann model for simulating flows with multiple phases and components. *Phys. Rev. E*, 47:1815–1819, 1993.

[45] X. Shan and G. Doolen. Diffusion in a multicomponent lattice Boltzmann equation model. *Phys. Rev. E*, 54:3614–3620, 1996.

[46] X.W. Shan and H. Chen. Simulation of nonideal gases and liquid-gas phase transitions by the lattice Boltzmann equation. *Phys. Rev. E*, 62:2941–2948, 1994.

[47] J.D. Sterling and S. Chen. Stability analysis of lattice Boltzmann methods. *J. Comp. Phys.*, 123:196–206, 1996.

[48] M. Swift and and J. M. Yeomans W. R. Osborn. Lattice Boltzmann simulation of non-ideal fluids. *Phys. Rev. Lett.*, 75:830–833, 1995.

[49] M. R. Swift, E. Orlandini, W.R. Osorn, and J.M. Yeomans. Lattice Boltzmann simulations of liquid-gas and binary fluid systems. *Phys. Rev. E*, 54:5041–5052, 1996.

[50] R. Verberg and A.J.C. Ladd. Accuracy and stability of a lattice-Boltzmann model with subgrid scale boundary conditions. *Phys. Rev. E*, 65:016701–016707, 2001.

[51] A.J. Wagner. *Theory and applications of the lattice Boltzmann method.* D.Phil. dissertation, University of Oxford, Michaelmas Term, 1997.

[52] A.J. Wagner. An H-theorem for the lattice Boltzmann approach to hydrodynamics. *Europhys. Lett.*, 44:144–149, 1998.

[53] A.J. Wagner. The origin of spurious velocities in lattice Boltzmann. *J. Mod. Phys. B*, 17:193–196, 2003.

[54] A.J. Wagner. Lattice Boltzmann simulations of strongly phase-separated liquid-gas systems. unpublished, 2006.

[55] A.J. Wagner. Thermodynamic consistency of liquid-gas lattice Boltzmann simulations. *Phys. Rev. E*, 74:056703.1–056703.12, 2006.

[56] A.J. Wagner and M.E. Cates. Phase ordering of two-dimensional symmetric binary fluids: A droplet scaling state. *Europhys. Lett.*, 56:556–562, 2001.

[57] A.J. Wagner and Q. Li. Investigation of Galilean invariance of multi-phase lattice Boltzmann methods. *Physica A*, 326:105–110, 2006.

[58] A.J. Wagner and I. Pagonabarraga. Lees-Edwards boundary conditions for lattice Boltzmann. *J. Stat. Phys.*, 107:521–537, 2002.

[59] A.J. Wagner and J.M. Yeomans. Effect of shear on droplets in a binary mixture. *Int. J. Mod. Phys. C*, 8:773–782, 1997.

[60] S. Wolfram. Cellular automaton fluids 1: Basic theory. *J. Stat. Phys.*, 45:471–526, 1986.

[61] R. A. Worthing, J. Mozer, and G. Seeley. Stability of lattice Boltzmann methods in hydrodynamic regimes. *Phys. Rev. E*, 56:2243–2253, 1997.

[62] D. Yu. *Navier-Stokes flow computations with the lattice-Boltzmann model.* Ph.D. dissertation, University of Florida, 2002.

[63] H. Yu. *Lattice Boltzmann equation simulations of turbulence, mixing, and combustion.* Ph.D. dissertation, University of Bergen, Texas A&M University, December 2004.

[64] R. Zhang and H. Chen. Lattice Boltzmann method for simulations of liquid-vapor thermal flows. *Phys. Rev. E*, 67:066711.1–066711.6, 2003.

[65] B. Zhou and A.C. Powell. Phase field simulations of early stage structure formation during immersion precipitation of polymeric membranes in 2d and 3d. *J. Membrane Sci.*, 268:150–164, 2006.

APPENDIX A

FLORY-HUGGINS MODEL OF POLYMER SOLUTION

The Flory Huggins free energy model is very popular in studying phase behavior of polymer solutions. One motivation to develop the multicomponent LB method is to apply it to simulate the hydrodynamics and phase-separation of multicomponent polymer solutions. Therefore, I derive the free energy of the system from the model [33]. I will start from a binary polymer solution then extend our conclusions to a multicomponent solution.

For an isothermal system, the free energy change of the system can be calculated by the entropy change and the internal energy change. For a binary system of components A and B, A is monomer and B is polymer with polymerization m. The system can be modeled as two kinds of balls in a three-dimensional lattice. Every lattice site can only be filled with one ball. The entropy of a system can be calculated using Boltzmann's equation: $S = k \ln \Omega$, where Ω is the total number of distinguishable arrangements having equal energy. For a binary system, the entropy change is

$$\Delta S_{mix} = k[\ln \Omega_{AB} - (\ln \Omega_A + \ln \Omega_B)], \tag{A.1}$$

where Ω_{AB} represents the arrangement in the mixture while Ω_A and Ω_B represent the arrangement in the pure components. For our binary system, the total number of sites is $N_A + mN_B$. Consider the arrangement for the first polymer chain first. The number of arrangements of the first mer is $N_A + mN_B$. The number of arrangements of the second mer is z, which is the lattice coordination number. The number of arrangements of the third mer is $z - 1$ (one less because one coordination is occupied by the first mer). The number of arrangements of the fourth mer is $z - 1$. The number of arrangements of the fifth mer is $(z - 1)\varepsilon$, where $\varepsilon < 1$ accounts for the fact that some of the $z - 1$ neighboring sites not occupied by the third mer could be occupied by the first or second. The "mean field approximation" is used because rigorous treatment is difficult. Because ε can be approximated as the overall fraction of unfilled sites in the lattice, $\varepsilon_5 = (N_A + mN_B - 4)/(N_A + mN_2)$ for the fifth mer, and ε_j is assumed to be the same in one chain. The arrangements of the first polymer chain ν_1 is $\nu_1 = (N_A + N_B)z(z-1)^{m-2}\varepsilon_j^{m-4}$. Similarly, the arrangements of the second polymer chain can be obtained in the form $\nu_2 = [N_A + m(N_B - 1)]z(z - 1)^{x-2}\varepsilon_j^{m-1}$ while $N_A + m(N_B - 1)$ for the first mer, $z\varepsilon_j$ for the second mer, and $(z - 1)\varepsilon_j$ for the third mer and so on. In general, $\nu_{i+1} = [N_A + m(N_B - 1)]z(z - 1)^{x-2}\varepsilon_j^{m-1}$. Since N_1 and $N - 2$ are large numbers, ε_j can be computed approximately for each chain rather than each mer: $\varepsilon_j = (N_A + m(N_B - j))/(N_A + mN_B)$. The approximation of $z \approx z - 1$ yields

$$\begin{aligned} \nu_{i+1} &\approx [N_A + m(N_B - i)]\left[(z-1)\frac{N_A + m(N_B - i)}{N_A - mN_B}\right]^{m-1} \\ &= [N_A + m(N_B - i)]^m \left[\frac{z-1}{N_A - mN_B}\right]^{m-1} \end{aligned} \tag{A.2}$$

$$\begin{aligned}
\Omega_{AB} &= \frac{1}{N_B!}\Pi_0^{N_B-1}\nu_{i+1} \\
&= \frac{1}{N_B!}\left[\frac{z-1}{N_A+mN_B}\right]^{N_B(m-1)}\Pi_0^{N_B-1}[N_A+m(N_B-i)]^m \\
&= \frac{(N_A+mN_B)!}{N_A!N_B!}\left[\frac{z-1}{N_A+mN_B}\right]^{N_B(m-1)}.
\end{aligned} \tag{A.3}$$

The arrangement of a pure polymer Ω_B can be found by setting $N_A = 0$:

$$\Omega_B = \frac{(mN_B)!}{N_B!}\left[\frac{z-1}{mN_B}\right]^{N_B(m-1)}. \tag{A.4}$$

Using $\Omega_A = 1$ for a pure solvent and Sterling's approximation, I obtain

$$\begin{aligned}
&\Delta S_{mix} \\
&= k[\ln\Omega_{AB} - (\ln\Omega_a + \ln\Omega_B)] \\
&= k\left[\ln N_A + mN_B)! - \ln N_A! - \ln(mN_B)! + N_B(m-1)\ln\left(\frac{mN_B}{N_A+mN_B}\right)\right] \\
&= -k(N_A\ln\phi_A + N_B\ln\phi_B),
\end{aligned} \tag{A.5}$$

where $\phi_A \equiv \frac{N_A}{N_A+mN_B}$ and $\phi_B \equiv \frac{mN_B}{N_A+mN_B}$ are the volume fractions. The enthalpy change of mixing can be calculated with a simple model. Let e_{AA} represent the contact energy between two A mers and e_{BB} represent the contact energy between two B mers. The enthalpy change for mixing to form one pair of A B contacts from pure A and pure B contacts is $\Delta e = e_{AB} - \frac{1}{2}(e_{AA} + e_{BB})$, as illustrated by ($\bullet\bullet + \circ\circ \longrightarrow 2\bullet\circ$), where $\bullet$ and $\circ$ represent an A mer and a B mer. For a polymer solution, the enthalpy of mixing is $\Delta H_{mix} = p_{AB}\Delta e$, where p_{AB} is the total number of AB contacts in the solution. Lattice sites adjacent to the end mer are $z-1$, and lattice sites adjacent to each mer that is not at the end is $z-2$. So the total lattice sites for a single chain with m degree of polymerization are $m(z-1)+2 \approx m(z-2)$. And the lattice sites for all polymer chains are $N_B m(z-2)$. With mean field approximation, ϕ_A of the total sites are filled by solvent, $p_{AB} = N_B(z-2)m\phi_A = (z-2)N_A\phi_B$. Therefore, the enthalpy change of mixing is $\Delta H_{mix} = (z-2)N_A\phi_B\Delta e = kTN_A\phi_B\chi$, where $\chi \equiv (z-2)\Delta e/(kT)$ is the interaction parameter. The free energy change of the mixing is

$$\Delta G_{mix} = \Delta H_{mix} - T\Delta S_{mix} = kT(N_A \ln\phi_A + N_B \ln\phi_B + N_A\phi_B\chi), \tag{A.6}$$

where $\theta = kT$.

Generalizing Eq. (3.1) to an arbitrary multicomponent system, considering the contribution from gradient terms, I get the total Helmholtz free energy of the mixing of a multicomponent system in Eq. (2.42).

The chemical potential can be calculated by substituting Eq. (2.42) into Eq. (2.16). Considering the surface tension contribution, the free energy density of a binary system can be obtained with Eq. (2.15) as

$$\phi = \psi(n^A, n^B) + \frac{1}{2}\kappa^{AA}\nabla n^A\nabla n^A + \frac{1}{2}\kappa^{AB}\nabla n^A\nabla n^B + \frac{1}{2}\kappa^{BB}\nabla n^B\nabla n^B, \tag{A.7}$$

where $\psi(n^A, n^B)$ is bulk free energy density, κ is the surface tension parameter.

$$
\begin{aligned}
\mu^A &= \frac{\partial \phi}{\partial n^A} + \nabla \frac{\partial \phi}{\partial \nabla n^A} && \text{(A.8)} \\
&= \theta[-m_A + \ln \phi^A + \phi^B(1 - m_A/m_B) + m^A \chi_{AB} \phi_B^2)] \\
&\quad -2\kappa^A \nabla^2 n^A - \kappa^{AB} \nabla^2 n^B, && \text{(A.9)} \\
\mu^B &= \theta[-m_B + \ln \phi^B + \phi^A(1 - m_B/m_A) + m^B \chi_{AB} \phi_A^2)] \\
&\quad -2\kappa^B \nabla^2 n^B - \kappa^{AB} \nabla^2 n^A. && \text{(A.10)}
\end{aligned}
$$

Similarly, the chemical potential for ternary systems are

$$
\begin{aligned}
\mu^A &= \theta[-m_A + \ln \phi^A + (1 - \phi^A) - \phi^B \frac{m^A}{m^B} - \phi^C \frac{m^A}{m^C} \\
&\quad + m^A(1 - \phi_A)(\phi^B \chi^{AB} + \phi^C \chi^{AC}) - m^A \chi^{BC} \phi^B \phi^C] \\
&\quad -2\kappa^A \nabla^2 n^A - \kappa^{AB} \nabla^2 n^B - \kappa^{AC} \nabla^2 n^C, && \text{(A.11)} \\
\mu^B &= \theta[-m_B + \ln \phi^B + (1 - \phi^B) - \phi^C \frac{m^B}{m^C} - \phi^A \frac{m^B}{m^A} \\
&\quad + m^B(1 - \phi_B)(\phi^C \chi^{BC} + \phi^A \chi^{AB}) - m^B \chi^{AC} \phi^A \phi^C] \\
&\quad -2\kappa^B \nabla^2 n^B - \kappa^{BC} \nabla^2 n^C - \kappa^{AB} \nabla^2 n^A, && \text{(A.12)} \\
\mu^C &= \theta[-m_C + \ln \phi^C + (1 - \phi^C) - \phi^A \frac{m^C}{m^A} - \phi^B \frac{m^C}{m^B} \\
&\quad + m^C(1 - \phi_C)(\phi^A \chi^{AC} + \phi^B \chi^{BC}) - m^C \chi^{AB} \phi^A \phi^B] \\
&\quad -2\kappa^C \nabla^2 n^C - \kappa^{AC} \nabla^2 n^A - \kappa^{BC} \nabla^2 n^B. && \text{(A.13)}
\end{aligned}
$$

APPENDIX B

GALILEAN INVARIANCE IN CONTINUUM

To discuss the Galilean Invariance problem of LB, it will be beneficial to understand that NSE itself obeys Galilean invariance. For two systems A' and A, if A is moving with a constant velocity U with reference to A', the Galilean transformation is as follows:

$$\begin{cases} \mathbf{r}' = \mathbf{r} + \mathbf{U}t \\ t' = t \\ \mathbf{u}' = \mathbf{u} + \mathbf{U} \end{cases}$$

and also

$$\begin{cases} \mathbf{r} = \mathbf{r}' - \mathbf{U}t' \\ t = t' \\ \mathbf{u} = \mathbf{u}' - \mathbf{U}. \end{cases}$$

Suppose $F(\mathbf{r}, t)$ is an arbitrary function in the A system and $F'(\mathbf{r}', t')$ is the corresponding function in A' system. The two functions describe the same physical field. We have

$$F(\mathbf{r}, t) = F'(\mathbf{r} + \mathbf{U}t, t), \tag{B.1}$$

$$F'(\mathbf{r}', t') = F(\mathbf{r}' - \mathbf{U}t', t'). \tag{B.2}$$

Because $\partial_{\mathbf{r}}\mathbf{r}' = \partial_{\mathbf{r}}(\mathbf{r}+\mathbf{U}t) = \partial_{\mathbf{r}}\mathbf{r}$, we have

$$\partial_{\mathbf{r}}F = \partial_{\mathbf{r}'}F' \tag{B.3}$$

$$\partial_t F(\mathbf{r},t) = \partial_t F'(\mathbf{r}+\mathbf{U}t,t) = \partial_{t'}F' + \mathbf{u}\cdot\partial_{\mathbf{r}'}F'. \tag{B.4}$$

Similarly, we can get

$$\partial_{t'}F' = \partial_t F - \mathbf{U}\cdot\partial_{\mathbf{r}}F. \tag{B.5}$$

The continuity equation in system A is

$$\partial_t\rho + \partial_{\mathbf{r}}(\rho\mathbf{u}) = 0. \tag{B.6}$$

By using Eq. (B.3) and Eq. (B.5), the continuity equation in system A' is

$$\begin{aligned} & \partial_{t'}\rho' + \partial_{\mathbf{r}'}(\rho' u') \\ = \; & \partial_t\rho - \mathbf{U}\cdot\partial_{\mathbf{r}}\rho + \partial_{\mathbf{r}}(\rho\mathbf{u}) + \partial_{\mathbf{r}}(\rho\mathbf{U}) \\ = \; & \partial_t\rho + \partial_{\mathbf{r}}\rho\mathbf{u} \\ = \; & 0. \end{aligned}$$

The right side of the NSE equation contains only spatial derivatives that are equivalent in the two systems. The left side in the A' system is

$$\begin{aligned}
& \rho'(\partial_{t'} u'_i + u'_k \partial_k u_i) \\
= \;& \rho'(\partial_{t'} u'_i + u'_k \partial_k u'_i) \\
= \;& \rho[(\partial_t - U_k \partial_k)(u_i + U_i) + (u_k + U_k)\partial_k (u_i + U_i)] \\
= \;& \rho(\partial_t u_i + u_k \partial_k u_i).
\end{aligned}$$

Therefore, the NSEs are of the same form in both the A and the A' system.

APPENDIX C

ANALYTICAL SOLUTION OF ONE-DIMENSIONAL SOUND WAVE

The theoretical equation describing the sound wave propagation in an isothermal ideal gas in one dimension is needed as a reference to validate the LB simulation algorithm.

C.1. Sound Wave Equation Without Viscosity

The sound wave equation in isothermal ideal gases in one dimension can be derived from the continuity equation Eq. (2.1) and the Navier Stokes equation Eq. (2.2). Since the sound wave equation is expected to be a second order differential equation, terms of higher order smallness can be neglected in the derivation. The velocity of the fluid **u** caused by a sound wave can be treated as a small quantity of the first order of smallness when the amplitude of the sound wave is small. Suppose the viscosity is negligible, Eq. (2.2) yields

$$\partial_t(\rho\mathbf{u}) + \nabla P = 0. \tag{C.1}$$

Doing time derivative of Eq. (2.1) gives

$$\partial_t^2\rho + \partial_t\nabla\cdot(\rho\mathbf{u}) = 0. \tag{C.2}$$

Substituting Eq. (C.2) into Eq. (C.1) leads to

$$\partial_t^2 \rho - \nabla^2 P = 0. \tag{C.3}$$

In ideal gas, the $P = kT\rho$ is a constant, so in the isothermal case $\partial_\rho P = kT$.

$$\nabla^2 P = kT\nabla^2 \rho. \tag{C.4}$$

The wave equation is given by

$$\partial_t^2 \rho = kT\nabla^2 \rho \tag{C.5}$$

In one dimension, Eq. (C.5) reduces to

$$\partial_t^2 \rho = kT\partial_x^2 \rho \tag{C.6}$$

The sound velocity obviously is given by

$$c_s = \sqrt{kT}. \tag{C.7}$$

In the LB lattice model, the sound velocity $c_s = 1/\sqrt{3}$.

C.2. Sound Wave Equation with Viscosity

To derive the sound wave equation with viscosity, I express the stress tenor of Eq. (2.3) in an alternative form

$$\sigma_{ij} = \eta(\partial_j u_i + \partial_i u_j) + \lambda \delta_{ij} \nabla \cdot \mathbf{u}, \tag{C.8}$$

where η is the shear viscosity and λ is the second viscosity.

The NSE with this viscosity definition is given by

$$\rho \partial_t \mathbf{u} + \rho \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla P + (2\eta + \lambda) \nabla (\nabla \cdot \mathbf{u}). \tag{C.9}$$

Eq. (C.9) in one dimension becomes

$$\rho u_t + \rho u u_x = (2\eta + \lambda) u_{xx} - p_x. \tag{C.10}$$

In order to find an analytical solution of a sound wave, the sound wave amplitude should be considered.

$$\rho = \delta\rho + \rho_0, \quad p = \delta p + p_0, \tag{C.11}$$

where $\delta\rho$ and δp are the perturbations of density and pressure caused by sound waves, and ρ_0 and p_0 are the density and pressure of the ideal gas without a sound wave. Note that $\delta\rho$, δp , and u are of the first order smallness, which is denoted as $O(\epsilon)$. In sound wave analysis, the long wave length assumption is not applicable, and the time and space derivative of a variable would not increase its order of smallness. Plugging Eq. (C.11) into Eq. (C.10) and retaining only the first order terms give

$$\rho_0 u_t + \delta p_x = (2\eta + \lambda) u_{xx} + O(\epsilon^2). \tag{C.12}$$

Eq. (C.7) implies in isothermal case,

$$\delta p = c_s^2 \delta\rho \tag{C.13}$$

Substituting Eq. (C.11) into Eq. (2.1) yields

$$\delta\rho_t + \rho_0 u_x = O(\epsilon^2). \tag{C.14}$$

Denoting $s \equiv 2\eta + \lambda$ and doing t derivative to Eq. (C.12) lead to

$$\rho_0 u_{tx} + \delta p_{xx} = s u_{xxx} + O(\epsilon^3). \tag{C.15}$$

Doing x derivative to Eq. (C.13) gives

$$\delta p_x = c_s^2 \delta\rho_x. \tag{C.16}$$

Doing t derivative to Eq. (C.14) gives

$$\delta\rho_{tt} + \rho_0 u_{xt} = O(\epsilon^3). \tag{C.17}$$

Doing x derivative to Eq. (C.14) gives

$$\delta\rho_{tx} + \rho_0 u_{xx} = O(\epsilon^3). \tag{C.18}$$

Substituting Eqs. (C.16), (C.17), and (C.18) into Eq. (C.15) and keeping the terms only to the order of $O(\epsilon^2)$, the wave equation is obtained in the form of

$$-\delta\rho_{tt} + c_s^2 \delta p_{xx} + \frac{s}{\rho_0}\delta\rho_{txx} = 0. \tag{C.19}$$

We assume the analytical solution is of the form $\delta\rho = \delta\rho_0 \exp(i(kx - \omega t))$ where $\omega = \alpha - i\beta$ and β is the attenuation coefficient. So I have

$$\begin{aligned} \delta\rho_{tt} &= -\omega^2\delta\rho \qquad (C.20)\\ \delta\rho_{xx} &= -k^2\delta\rho \qquad (C.21)\\ \delta\rho_{xxt} &= ik^2\omega\delta\rho \qquad (C.22) \end{aligned}$$

Substituting Eqs. (C.20), (C.21), and (C.22) into Eq. (C.19) gives

$$\omega^2 - c_s^2 k^2 + i\frac{s\omega k^2}{\rho_0} = 0 \tag{C.23}$$

In the one dimensional LB model, by noting $c_s = 1/\sqrt{3}$, $s = 2\nu + 0 = 2\rho(\tau - \delta t/2)/3$, and with the substitution of complex expression of ω into equation C.23, an algebra equation can be obtained as

$$\alpha^2 - i2\alpha\beta - \beta^2 - c_s^2k^2 + i\alpha c_s^2k^2(2\tau - \Delta t) + \beta c_s^2k^2(2\tau - \Delta t) = 0. \tag{C.24}$$

The real and imaginary parts of the equation C.24 should both equal to zero. The solution of the simultaneous equations obtained is

$$\beta = c_s^2k^2(\tau - \Delta t/2) \tag{C.25}$$

$$\alpha = \sqrt{c_s^2k^2 - c_s^4k^4(\tau - \Delta t/2)^2} \tag{C.26}$$

A general solution is then of the form

$$\rho = \rho_0 + \delta\rho_0 \exp(-c_s^2k^2(\tau - \Delta t/2)t)\sin(k(x - c_s\sqrt{1 - c_s^2k^2(\tau - \Delta t/2)^2}\ t\ )). \tag{C.27}$$

To find the initial condition for my LB simulation, I substitute the Eq. (C.27) into the continuity equation Eq. (2.1) and obtain

$$-\delta\rho_0\beta\exp(-\beta t)\sin(kx - \alpha t) - \delta\rho_0\alpha\exp(-\beta t)\cos(kx - \alpha t) + \frac{\partial}{\partial x}(\rho u) = 0 \tag{C.28}$$

Integrating Eq. (C.28) with respect to x, applying $t = 0$ and Eqs. (C.25) and (C.26), I get the initial condition for $u(x)$ in a static medium frame as

$$u(x) = \delta\rho_0\left(-c_s^2k(\tau - \Delta t/2)\cos(kx) + c_s\sqrt{1 - c_s^2k^2(\tau - \Delta t/2)^2}\sin(kx)\right)/\rho(x) \tag{C.29}$$

APPENDIX D

THE DERIVATION OF PRESSURE TENSOR

The pressure tensor can be derived from the total free energy expression of a system, which is given by Eq. (2.18). The force on density can be obtained from the derivative of the pressure tensor or from the product of the density and the the derivative of the chemical potential, which is given by Eq. (2.16). Therefore,

$$\begin{aligned}\partial_\beta P_{\alpha\beta} &= n\partial_\alpha \left(\frac{\delta\mathcal{F}}{\delta n}\right) \\ &= n\partial_\alpha \left(\frac{d\psi}{dn} - \kappa\partial_\beta\partial_\beta n\right). \end{aligned} \tag{D.1}$$

The first term of the equation(D.1) can be written

$$\partial_\alpha \left(n\frac{d\psi}{dn}\right) - \frac{d\psi}{dn}\partial_\alpha n = \partial_\alpha \left(n\frac{d\psi}{dn} - \psi\right). \tag{D.2}$$

And the second term becomes

$$\begin{aligned} &\kappa[(\partial_\alpha n)(\partial_\beta\partial_\beta n) - \partial_\alpha(n\partial_\beta\partial_\beta n)] \\ =\ &\kappa[\partial_\beta\{(\partial_\alpha)(\partial_\beta n)\} - (\partial_\beta n)\partial_\beta\partial_\alpha n - \partial_\alpha(n\nabla^2 n)] \\ =\ &\kappa\partial_\beta\left[(\partial_\alpha n)(\partial_\beta n) - \delta_{\alpha\beta}\left(n(\nabla^2 n) + \frac{1}{2}|\nabla n|^2\right)\right]. \end{aligned} \tag{D.3}$$

By Eqs (D.2) and (D.3) we have

$$\partial_\beta P_{\alpha\beta} = \partial_\beta \left[\delta_{\alpha\beta}\left(p_0 - \kappa n \nabla^2 n - \frac{\kappa}{2}|\nabla n|^2\right) + \kappa(\partial_\alpha n)(\partial_\beta n)\right], \tag{D.4}$$

where $p_0 = n\frac{d\psi}{dn} - \psi$. Therefore up to an arbitrary constant

$$P_{\alpha\beta} = \delta_{\alpha\beta}\left(p_0 - \kappa n \nabla^2 n - \frac{\kappa}{2}|\nabla n|^2\right) + \kappa(\partial_\alpha n)(\partial_\beta n). \tag{D.5}$$

Following the same process, the pressure tensor of a binary system can be obtained as [51]:

$$P_{\alpha\beta} = p\delta_{\alpha\beta} + \kappa_\rho(\partial_\alpha\rho)(\partial_\beta\rho) + \kappa_\varphi(\partial_\varphi\rho)(\partial_\beta\varphi), \tag{D.6}$$

where $p = p_0 - k_\rho\rho\nabla^2\rho - \frac{\kappa_\rho}{2}|\nabla\rho|^2 - k_\varphi\varphi\nabla^2\varphi - \frac{\kappa_\varphi}{2}|\nabla\varphi|^2$ and $p_0 = \rho\partial_\rho\psi + \varphi\partial_\varphi\psi - \psi$. Here $\rho = n_A + n_B$ and $\varphi = n_A - n_B$ are the sum and difference of the two components.

APPENDIX E

GALILEAN INVARIANCE EXAMINATIONS

Eq. (4.1) can be used to examine whether an LB model is strictly Galilean invariance and to indicate which terms break the GI. Four models are examined: the D2Q7 and D2Q9 models for ideal gases; Swift's and Inamuro's models for non-ideal gases. The moments in the D2Q7 ideal gas model are

Table E.1. Velocity product summation(D2Q7)

sum over i	**for i=** $2, \cdots, 7$
$\sum_{i=0}^{6} v_{i\alpha}$	0
$\sum_{i=0}^{6} v_{i\alpha}v_{i\beta}$	$3\delta_{\alpha\beta}$
$\sum_{i=0}^{6} v_{i\alpha}v_{i\beta}v_{i\gamma}$	0
$\sum_{i=0}^{6} v_{i\alpha}v_{i\beta}v_{i\gamma}v_{i\delta}$	$\frac{3}{4}(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma})$
$\sum_{i=0}^{6} v_{i\alpha}v_{i\beta}v_{i\gamma}v_{i\delta}v_{i\theta}$	0

$$
\begin{aligned}
&M_0(\mathbf{0}) = \rho, && M_1(\mathbf{u}) = \rho;\\
&M_1(\mathbf{0})_\gamma = 0, && M_1(\mathbf{u})_\gamma = 0;\\
&M_2(\mathbf{0})_{\gamma\delta} = \frac{\rho}{4}\delta_{\gamma\delta}, && M_2(\mathbf{u})_{\gamma\delta} = \frac{\rho}{4}\delta_{\gamma\delta};\\
&M_3(\mathbf{0})_{\gamma\delta\theta} = 0, && M_3(\mathbf{u})_{\gamma\delta\theta} = -\rho u_\gamma u_\delta u_\theta.
\end{aligned}
$$

The moments in the D2Q9 ideal gas model are

Table E.2. Velocity product summation(D2Q9)

sum over i	**for i=** $2, \cdots, 5$	**for i=** $6, \cdots, 9$
$\sum v_{i\alpha}$	0	0
$\sum v_{i\alpha}v_{i\beta}$	$2\delta_{\alpha\beta}$	$4\delta_{\alpha\beta}$
$\sum v_{i\alpha}v_{i\beta}v_{i\gamma}$	0	0
$\sum v_{i\alpha}v_{i\beta}v_{i\gamma}v_{i\sigma}$	$2\delta_{\alpha\beta}\delta_{\alpha\gamma}\delta_{\beta\sigma}$	$4(\delta_{\alpha\beta}\delta_{\gamma\sigma} + \delta_{\alpha\gamma}\delta_{\beta\sigma} + \delta_{\alpha\sigma}\delta_{\beta\gamma} - 2\delta_{\alpha\beta}\delta_{\alpha\gamma}\delta_{\beta\sigma})$
$\sum v_{i\alpha}v_{i\beta}c_{i\gamma}v_{i\sigma}v_{i\lambda}$	0	0

$$\begin{aligned}
M_0(\mathbf{0}) &= \rho, & M_0(\mathbf{u}) &= \rho;\\
M_1(\mathbf{0})_\gamma &= 0, & M_1(\mathbf{u}) &= 0;\\
M_2(\mathbf{0})_{\gamma\delta} &= \tfrac{\rho}{3}\delta_{\gamma\delta}, & M_2(\mathbf{u}) &= \tfrac{\rho}{3}\delta_{\gamma\delta};\\
M_3(\mathbf{0})_{\gamma\delta\theta} &= 0, & M_3(\mathbf{u}) &= -\rho u_\gamma u_\delta u_\theta\,.
\end{aligned}$$

Therefore, the above three ideal gas models suffer GI violation because of the $-\rho u_\gamma u_\delta u_\theta$ term, just as shown in Eq. (3.31) . Non-ideal fluid models also can be tested with the model. For the Swift model, the equilibrium distribution in D2Q7 model is of the form

$$f_i^{eq} = A + Bu_\alpha v_{i\alpha} + Cu^2 + Du_\alpha u_\beta v_{i\alpha}v_{i\beta} + G_{\alpha\beta}v_{i\alpha}v_{i\beta}, \tag{E.1}$$

for $i = 1, \cdots, 6$. For particles at rest,

$$f_0^{eq} = A_0 + C_0 u^2, \tag{E.2}$$

where the coefficients are

$$A_0 = n - 6A, A = (p_0 - \kappa - n\nabla^2 n)/3c^2, B = n/3c^2, C = -n/6c^2, C_0 = -n/c^2,$$
$$D = 2n/3c^4, G_{xx} = -G_{yy} = \frac{\kappa}{3c^4}\left\{(\partial_x n)^2 - (\partial_y n)^2\right\}, G_{xy} = \frac{2}{3c^4}\kappa\partial_x n\partial_y n.$$

Similarly, the moments can be obtained as

$$\begin{array}{ll} M_0(\mathbf{0}) = n, & M_0(\mathbf{u}) = n; \\ M_1(\mathbf{0})_\gamma = 0, & M_1(\mathbf{u})_\gamma = 0; \\ M_2(\mathbf{0})_{\gamma\delta} = 3A\delta_{\gamma\delta} + \frac{3}{2}G_{\gamma\delta}, & M_2(\mathbf{u})_{\gamma\delta} = 3A\delta_{\gamma\delta} + \frac{3}{2}G_{\gamma\delta}; \\ M_3(\mathbf{0})_{\gamma\delta\theta} = 0, & \begin{array}{l} M_3(\mathbf{u})_{\gamma\delta\theta} = -nu_\gamma u_\delta u_\theta + \frac{n}{4}(u_\theta\delta_{\gamma\delta} + u_\delta\delta_{\gamma\theta} + u_\gamma\delta_{\delta\theta}) \\ -3A(u_\theta\delta_{\gamma\delta} + u_\delta\delta_{\gamma\theta} + u_\gamma\delta_{\delta\theta}) - \frac{3}{2}(u_\theta G_{\gamma\delta} + u_\gamma G_{\theta\delta} + u_\delta G_{\theta\gamma}). \end{array} \end{array}$$

Inamuro suggested the following equilibrium distribution function in D2Q9 model. [22]:

$$f_i^{eq} = H_i\rho + F_i[(p_o - \kappa\rho\Delta^2\rho) + 2wu_\gamma\partial_\gamma\rho] \tag{E.3}$$
$$+E_i\rho[3u_\alpha v_{i\alpha} - \frac{3}{2}u^2 + \frac{9}{2}u_\alpha u_\beta v_{i\alpha}v_{i\beta}] + E_i G_{\alpha\beta}v_{i\alpha}v_{i\beta}, \tag{E.4}$$

where $i = 1, \cdots, 9$, and $\alpha, \beta, \gamma = 1, 2$ are the Cartesian coordinate with summation convention. $H_1 = 1$ and $H_i = 0$ for $i = 1; 2, \cdots, 9$; $E_1 = \frac{4}{9}$, $E_i = \frac{1}{9}$ for $i = 2, \cdots, 5$, and $E_i = \frac{1}{36}$ for $i = 6, \cdots, 9$; $F_1 = -\frac{5}{3}$, $F_i = \frac{1}{3}$ for $i = 2, \cdots, 5$, and $F_i = \frac{1}{12}$for $i = 6, \cdots, 9$.

$$G_{\alpha\beta} = \frac{9}{2}[\kappa\partial_\alpha\rho\partial_\beta\rho + w(u_\beta\partial_\alpha\rho + u_\alpha\rho_\beta)] - \frac{9}{4}[\kappa\partial_\gamma\rho\partial_\gamma\rho + 2wu_\gamma\partial_\gamma\rho]\delta_{\alpha\beta}. \tag{E.5}$$

where $\omega = \frac{1}{3}(\tau - \frac{1}{2})\Delta x$, and p_o is the pressure of a van der Waals fluid. Denote $p_1 \equiv p_0 - \kappa\rho\nabla^2\rho$.

$$\begin{aligned}
&M_0(\mathbf{0}) = \rho, && M_0(\mathbf{u}) = \rho;\\
&M_1(\mathbf{0})_\gamma = \mathbf{0}, && M_1(\mathbf{u}) = 0;\\
&M_2(\mathbf{0})_{\gamma\delta} = p_1\delta_{\gamma\delta} + \tfrac{2}{9}G_{\gamma\delta}, && M_2(\mathbf{u})_{\gamma\delta} = p_1\delta_{\gamma\delta} + \tfrac{2}{9}G_{\gamma\delta};\\
&M_3(\mathbf{0})_{\gamma\delta\theta} = 0, && M_3(\mathbf{u})_{\gamma\delta\theta} = -\rho u_\gamma u_\delta u_\theta - p_1(u_\gamma\delta_{\delta u_\theta} + u_\delta\delta_{\gamma\theta} + u_\theta\delta_{\gamma\delta})\\
& && \qquad -\tfrac{2}{9}(u_\gamma G_{\delta u_\theta} + u_\delta G_{\gamma\theta} + u_\theta G_{\gamma\delta}).
\end{aligned}$$

It is clear that the GI violation term $\rho u_\alpha u_\beta u_\gamma$ from the ideal gas model still remains uncorrected for all the non ideal fluid model corrections. Therefore a correction for $\rho u_\alpha u_\beta u_\gamma$ term has not been done in all the above models.

APPENDIX F
CODE FOR THEORETICAL SOLUTION OF A BINARY SYSTEM

```
/*This code is to calculate the binodal lines of a binary system.
As one phase separates into two phases, the free energy of the
system changes. We find the two binodal points by dislodging
the state variables of the two resulting phases until the free
energy of system reaches minimum.
   by Qun Li  on July, 2005 */
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>
#include<math.h>
#include<mygraph.h>
#define PRECISION 1e-10

/* calculate the the free energy of the system */
double FreeEnergy(double phi, double chi, double polymerization)
{double freeenergy;
 if(phi<=0 || phi>=1)
   {
     printf("The valid value of phi is in (0,1)" );
     return .0;
   }
 freeenergy = phi* log(phi) + (1.- phi)* log(1.- phi)
/polymerization + chi * phi * (1. - phi);
 return freeenergy;
}

/*calculate the free energy of the system when it phase separates
from a homogeneous state of phi0 into two phases, phi1 and phi2.*/
double Fmix(double phi0, double phi1, double phi2,
double polymerization, double chi)
{ double fmix;
 if(phi1< phi0 && phi0< phi2 )
 fmix= FreeEnergy(phi1,chi,polymerization)*(phi0-phi2)/(phi1-phi2)
+  FreeEnergy(phi2,chi,polymerization)*(phi1-phi0)/(phi1-phi2);
 else
 fmix=FreeEnergy(phi0,chi,polymerization);
 return fmix;
```

```
}

/* dislodge phi1 back and forth to find the minimum free energy
of the system*/
double FindPhi1min(double phi0, double phi1, double phi2,
double step,double polymerization, double chi,double boundary1)
{
  double phimin, fmin, f1, f2, f3, phi1l, phi1r;
  phi1l = phi1-step;
  phi1r = phi1+step;
  if(phi1l <=boundary1)
    phi1l = phi1;
  if(phi1+step > phi0)
    phi1r = phi0;
  f2=Fmix(phi0, phi1l, phi2, polymerization, chi);
  f1=Fmix(phi0, phi1, phi2, polymerization, chi);
  f3=Fmix(phi0, phi1r, phi2, polymerization, chi);
  fmin = f1;
  phimin=phi1;
  if(f2<fmin)
 {
   phimin=phi1l;
  fmin=f2;
 }
  if(f3<fmin)
    phimin=phi1r;
  return phimin;
}

/*dislodge phi2 back and forth to find the minimum free energy
 of the system */
double FindPhi2min(double phi0, double phi1, double phi2,
double step,double polymerization, double chi,double boundary2)
{
  double phimin, fmin, f1, f2, f3, phi2l, phi2r;
  phi2l = phi2-step;
  phi2r = phi2+step;
  if(phi2r >= boundary2)
    phi2r=phi2;
  if(phi0 + step > phi2)
    phi2l = phi0;
  f1=Fmix(phi0, phi1, phi2l, polymerization, chi);
  f2=Fmix(phi0, phi1, phi2, polymerization, chi);
  f3=Fmix(phi0, phi1, phi2r, polymerization, chi);
  fmin = f1;
  phimin=phi2l;
  if(f2<fmin)
 {
   phimin=phi2;
  fmin=f2;
 }
  if(f3<fmin)
    phimin=phi2r;
  return phimin;
}

main(){
FILE *pt;
int change1, change2;
double chi, step, phi0, phi1, phi2, phi1min, phi2min, polymerization,
epsilon, boundary1, boundary2, phi_critical, chi_critical;
pt=fopen("bim5.dat","w");
epsilon= 1e-10;
boundary1= epsilon;
boundary2=1.0-epsilon;
polymerization=5;
chi_critical =(polymerization + 1 + 2 * sqrt(polymerization))
                 /(2*polymerization);
phi_critical = 1/(sqrt(polymerization)+1); //volume fraction
for(chi = chi_critical; chi<5.1; chi=chi+0.01){
  step=0.01;
  phi0= 1- phi_critical; /* This is for solvent volume fraction */
  phi1 = phi0 - 2.0*step;
  phi2 = phi0 + 2.0*step;
  do{
       do{change1=0;  change2=0;
    phi1min = FindPhi1min(phi0, phi1, phi2, step,
                        polymerization, chi, boundary1);
  if(phi1min != phi1){change1=1; phi1 = phi1min;}
  phi2min =  FindPhi2min(phi0, phi1, phi2, step,
                polymerization, chi, boundary2);
  if(phi2min != phi2){change2 =1; phi2 = phi2min;}
          }while(change1==1 || change2==1);
        step = step/2. ;
     }while(step > epsilon);
  fprintf(pt, "%e  %e  %e \n", chi, phi1, phi2);}
fclose(pt);
return 0; }
```

APPENDIX G
CODE FOR THEORETICAL SOLUTION OF A TERNARY SYSTEM

```
/* to calculate the binodal points and chemical potentials
   with a  Flory-Huggins free energy model of  constant density */
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define SIZE 1000
#define PRECISION 1e-8
double mA=10.0,mB=1.0,mC=1.0,density =100,/*m=polymerization*/
can, dtheta_initial=0.001,chiAB=3,chiAC = 0.5, chiBC = 0.2;
/* chi is the interaction parameter between species */
 double  x[SIZE], y[SIZE], f[SIZE], ddf[SIZE],
/* ddf array is the second derivative of f */
   xx0, yy0, xx1, yy1, xx2, yy2, size=SIZE, d = 0.01,
/*d is the distant of two middle point */
  s= 1e-4,  /* s is step, xx1 is at left, xx2 is at right */
  tie_end_x1,tie_end_x2,tie_end_y1,tie_end_y2,sinitial=0.01,
  dinitial = 0.01, fminbi, theta, temperature = 0.3333333,
/*this temperature is theta in LB*/dtheta=0.0001,epsilon=1e-8;

double AlnA(double x){
  if(x < 1e-36) return 0;
  return  x*log(x);
}
/* the free energy per mer site */
double FreeEnergyDensity(double x, double y){
  double z, free;
  z=1-x-y;
  free = temperature*density*(1 + AlnA(x)/mA + AlnA(y)/mB
    + AlnA(z)/mC + chiAB*x*y + chiAC*x*z +chiBC*y*z);
  return free;
}
double chemA(double x, double y){
  double chem_po, z;
  z = 1-x-y;
  chem_po=temperature*(-mA +log(x)+(1- x)-y*mA/mB-z*mA/mC
          + mA*(1-x)*(y*chiAB + z*chiAC) - mA*chiBC*y*z);
  return chem_po;
}
```

```
double chemB(double x, double y){
  double chem_po, z;
  z = 1-x-y;
 chem_po = temperature*(-mB+log(y)+(1-y)-z*mB/mC-x*mB/mA
          + mB*(1-y)*(z*chiBC + x*chiAB)-mB*chiAC*x*z);
  return chem_po;
}
double chemC(double x, double y){
  double chem_po, z;
  z = 1- x-y;
  chem_po = temperature*(-mC+log(z)+(1- z)-x*mC/mA-y*mC/mB
          + mC*(1-z)*(x*chiAC + y*chiBC) - mC*chiAB*x*y);
  return chem_po;
}
/* Fmix is free energy of the system with (xx1, yy1),(xx2, yy2)
phases after phase separating from xx0, yy0 phase. The total
volume is assumed to be one and constant in phase separation.*/
double  Fmix(double xx0, double yy0, double xx1,
      double yy1, double xx2, double yy2) {
 if(xx1 == xx2) return FreeEnergyDensity(xx0, yy0);  return
 FreeEnergyDensity(xx1,yy1)*(xx0*yy2-xx2*yy0)/(xx1*yy2-xx2*yy1)
 +FreeEnergyDensity(xx2,yy2)*(xx0*yy1-xx1*yy0)/(xx2*yy1-xx1*yy2);
}
void SecondOrderDerivative( double a[], double dda[], int size){
  int i;
  for(i=1; i<size-1; ++i)
  dda[i] = (a[i+1] + a[i-1] - 2.0*a[i]);
  dda[0] = dda[1];
  dda[size-1]=dda[size-2];
}
void Tie(double theta, double xx0, double yy0){
 int i;
 tie_end_x1 = 0;
 tie_end_x2 = xx0 + yy0*tan(M_PI/2.0 - theta);
 tie_end_y1 = yy0 + xx0*tan(theta);
 tie_end_y2 = 0;
  for(i=0;i<size;i++)
    { x[i]=tie_end_x1+((double)i/(size-1.0))*(tie_end_x2-tie_end_x1);
      y[i]=tie_end_y1+((double)i/(size-1.0))*(tie_end_y2-tie_end_y1);
      f[i]=FreeEnergyDensity (x[i],y[i]);
      SecondOrderDerivative( f, ddf, size);
    }
}
void Pre_binodal(void){
 int i;
 for(i=0; i<size; i++)
   { if(ddf[i] >= 0 &&  ddf[i+1]<0)
     {xx1 = x[i+1];
     yy1  = y[i+1];
     }
   if(ddf[i]<0 && ddf[i+1]>=0)
     { xx2 = x[i];
     yy2 = y[i];
     break;
     }
   }
}
/* Findmin1 function is to determin the minimum total free energy
 of the two separated phases, given three points along a fixed
tie line. The middle points is the original one, and another two
points are at the outside and inside of the original point.
The range of the outside and inside points are limited */
void Findmin1_once(void){
  double fmin1, fout1, fin1, x1out, y1out, x1in, y1in, x1min, y1min;
  x1out=xx1-s*cos(theta);
  y1out=yy1+s*sin(theta);
  x1in = xx1 + s*cos(theta);
  y1in = yy1- s*sin(theta);
  if(x1out+y1out >1 || x1out<0){
      x1out = xx1;
      y1out =yy1;
    }
  if(x1in >xx0){
      x1in = xx0;
      y1in= yy0;
    }
  fmin1=Fmix(xx0,yy0,xx1,yy1,xx2,yy2);
  x1min=xx1;
  y1min=yy1;
  fout1 = Fmix(xx0,yy0,x1out,y1out,xx2,yy2);
  fin1 =  Fmix(xx0,yy0,x1in,y1in,xx2,yy2);
  if(fout1< fmin1){
    fmin1= fout1;
    x1min= x1out;
    y1min= y1out;
    }
  if(fin1 < fmin1){
    fmin1= fin1;
    x1min= x1in;
    y1min= y1in;
    }
  xx1=x1min;
  yy1=y1min;
}
/*The Findmin2 function behaves similar as Findmin1 */
void Findmin2_once(void){
 double fmin2,x2out,y2out,x2in,y2in,fout2,fin2,x2min,y2min;
  x2out = xx2+s*cos(theta);
  y2out = yy2-s*sin(theta);
  x2in = xx2-s*cos(theta);
  y2in = yy2+ s*sin(theta);
  if(x2out+y2out>1 || y2out <0){
      x2out = xx2;
      y2out = yy2;
    }
  if(x2in < xx0 ) {
      x2in = xx0;
      y2in= yy0;
    }
  fmin2=Fmix(xx0,yy0,xx1,yy1,xx2,yy2);
  x2min=xx2;
  y2min=yy2;
  fout2 = Fmix(xx0,yy0,xx1, yy1,x2out,y2out);
  fin2 =  Fmix(xx0, yy0, xx1, yy1, x2in, y2in);
  if(fmin2>fout2){
      fmin2 = fout2;
      x2min = x2out;
      y2min = y2out;
    }
  if(fmin2 > fin2){
     fmin2 = fin2;
     x2min = x2in;
     y2min = y2in;
    }
 xx2 = x2min;
 yy2 = y2min;
}
void Findmin1(){
  double oldxx1;
  do{  oldxx1= xx1;
       Findmin1_once();
  }while(oldxx1 != xx1);
}
void Findmin2(){
  double oldxx2;
  do{ oldxx2= xx2;
      Findmin2_once();
  }while(oldxx2 != xx2);
}
void Fix_binodal(){
  s = sinitial;
  do
    {
      Findmin1();
      Findmin2();
s = s/10.0 ;
    }
  while(s>PRECISION);
  fminbi= Fmix(xx0,yy0,xx1,yy1,xx2,yy2);
}
void Initial_condition(void)
{ s=sinitial;
 d=dinitial;
  xx0=0.4999;
  yy0=0.4999;
  theta=M_PI/4;
}
/*Find binodal points on line */
void Binodal_online(void){
Tie(theta, xx0, yy0);
  Pre_binodal();
  Fix_binodal();
}
/* the function is to find the initial binodal */
void Initial_binodal(void){
  Initial_condition();
  Binodal_online();
}
/*Find the new middle point from the old one by moving
 distance d from the old one along the direction perpendicular
 to the old tie line and towards the original point. */
void Next_center(void ){
  xx0=(xx1+xx2)/2.0;
  yy0=(yy1+yy2)/2.0;
  xx0=xx0 - d*sin(theta);
  yy0=yy0 - d*cos(theta);
}
/* the function is used after Tie() function to find the binodal
point with only one rotation */
void Rotation_once(void){
```

```
double the0, the1, the2, themin, fthemin,thex1,thex2,they1,they2;
  the1 = theta+dtheta;
  the2 = theta-dtheta;
  the0 = theta;
  theta = the1;
  Binodal_online();
  {
  fthemin = fminbi;
  themin = theta;
  thex1 = xx1;
  thex2 = xx2;
  they1 = yy1;
  they2 = yy2;
  }
  theta = the2;
  Binodal_online();
  if(fminbi<fthemin){
      thex1=xx1;
      thex2=xx2;
      they1=yy1;
      they2=yy2;
      themin = theta;
      fthemin = fminbi;
    }
  theta=the0;
    Binodal_online();
  if(fminbi<fthemin){
      thex1=xx1;
      thex2=xx2;
      they1=yy1;
      they2=yy2;
      themin=theta;
      fthemin = fminbi;
    }
  fminbi = fthemin;
    theta=themin;
    xx1=thex1;
    xx2=thex2;
    yy1=they1;
    yy2=they2;
}
Binodal_rotation(){
  double oldfminbi;
  do{
    oldfminbi = fminbi;
    Rotation_once();
  } while( oldfminbi != fminbi ) ;
}
/*To find the binodal point with the many rotations */
void Binodal(void){
 double thetaold;
 dtheta = dtheta_initial;
  do
    {
      thetaold=theta;
      Binodal_rotation();
dtheta = dtheta/10;
    }
  while(dtheta > PRECISION);
}
/* this function check if phase separation is possible */
int Can(double theta, double xx0, double yy0) {
  int i=0;
  Tie(theta, xx0, yy0);
  for(i=0; i<size; i++)
    if(ddf[i] < 0) return (1);
    return (0) ;
}
main(){
  FILE* pt;
 pt = fopen("ternchem1.dat", "w");
 Initial_condition();
 Initial_binodal();
 can =1;
 while(can ==1)
   {
      if( yy1 <= 0.4) d = 0.0001;
    Next_center();
    can = Can(theta, xx0, yy0);
    Binodal();
   fprintf(pt, "%f %f %f %f %f %f %f %f %f %f \n",xx1,yy1,xx2,yy2,
      chemA(xx1, yy1), chemA(xx2, yy2), chemB(xx1, yy1),
      chemB(xx2, yy2), chemC(xx1, yy1), chemC(xx2,yy2));
    }
 fclose(pt);
 return 0;
}
```

APPENDIX H
CODE FOR LIQUID-GAS SYSTEM (HOLDRYCHQ)

```
/*The code is to verify the Holdych correction to pressure tensor.
The presure tensor correction  = (\tau - 0.5)*u*\partial_x n,
 and therefore we have the equilibrium distribution function as:
 feq0=n-p-nu^2-c; feq1=(p+nu^2+nu+c)/2; feq2=(p+nu^2-nu+c)/2;*/
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<math.h>
#include<mygraph.h>
#define xdim 100
static int i, size = xdim, Repeat = 1, done= 0, sstep = 0,
Pause=1, densityreq = 0;
static double f0[xdim],f1[xdim],f2[xdim],omega = 1, n[xdim],
u[xdim],p[xdim],dn[xdim],mu[xdim], nu[xdim], ddn[xdim],
  kappa = 0.1, nc = 1,pc=0.125,betatau=0.03,u0=0,correction[xdim];
double chem_an, pressure_an, n_an[xdim];
Analytical(){
chem_an = 4*pc*(1-betatau)/nc;
   pressure_an = pc* (1-betatau)*(1-betatau);
   for(i=0; i<100; i++)
     n_an[i]=nc*(1+sqrt(betatau)*tanh((50-i)
    /sqrt(kappa*nc*nc/(2*betatau*pc))));
}
void init (void){
  int i;
  for(i=0; i < xdim; i++){
n[i] = 1 + 0.01*sin(2*M_PI*i/xdim);
     u[i] = u0;
     f0[i] =  (n[i]-n[i]*u[i]*u[i]-p[i]);
      f1[i] =  (n[i]*u[i]/2+n[i]*u[i]*u[i]/2+p[i]/2);
     f2[i] =  (-n[i]*u[i]/2+n[i]*u[i]*u[i]/2+p[i]/2);
   }
}
FirstOrderDerivative(double *a,double *da, int n){
  int i;
  da[0]=(a[1]-a[n-1])/2.0;
  da[n-1]=(a[0]-a[n-2])/2.0;
  for(i= 1; i<n-1; ++i)
    da[i]=(a[i+1]-a[i-1])/2.0;
}
void SecondOrderDerivative( double a[], double dda[], int n){
  int i;
  dda[0]= (a[1]+a[n-1]-2.0*a[0]);
  dda[n-1]=(a[0]+a[n-2]-2*a[n-1]);
  for(i=1; i<n-1; ++i)
    dda[i] = (a[i+1] + a[i-1] - 2.0*a[i]);
}
void  iterate(void){
  int i=0;
   double temp1, temp2 ;
  /*pc critical presure; beta a constant; nc critical density*/
  /*tau = (Tc-T)/Tc; */
  for (i=0;i<xdim;i++){
      n[i]=f0[i]+f1[i]+f2[i];
      nu[i] = (n[i] - nc)/nc;
      u[i]=(f1[i]-f2[i])/n[i];
      mu[i]=4*pc*(nu[i]+1)*(nu[i]*nu[i]-nu[i]+1.0-betatau)/nc
 -kappa*ddn[i];
    }
  FirstOrderDerivative(n,dn,size);
  SecondOrderDerivative(n,ddn,size);
  for (i=0;i<size;i++){
      p[i]=pc*(nu[i]+1)*(nu[i]+1)*(3*nu[i]*nu[i]-2*nu[i]+1-2*betatau)
-kappa*ddn[i] + kappa*dn[i]*dn[i]/2.0;
      correction[i]=(1.0/omega-0.5)*u[i]*dn[i];
    }
  for (i=0;i<xdim;i++){
      f0[i]+=omega*(n[i]-n[i]*u[i]*u[i]-p[i]-correction[i]-f0[i]);
      f1[i]+=omega*((n[i]*u[i]+n[i]*u[i]*u[i]+p[i]+correction[i])
/2.0-f1[i]);
      f2[i]+=omega*((-n[i]*u[i]+n[i]*u[i]*u[i]+p[i]+correction[i])
/2.0-f2[i]);
    }
  temp1 = f1[xdim-1];
  temp2=f2[0];
  for(i=1; i<xdim; i++){
      f1[xdim-i]=f1[xdim-i-1];
      f2[i-1]=f2[i];
    }
  f1[0]=temp1;
  f2[xdim-1]=temp2;
}
/*
void GUI()
```

```
{
  DefineGraphN_R("Density", n, &size, &densityreq);
  DefineGraphN_R("Chem. Pote.",mu, &size, &densityreq);
  DefineGraphN_R("U", u, &size, &densityreq);
  DefineGraphN_R("p",p,&size,&densityreq);
  StartMenu("GUI",1);
  DefineDouble("u0",&u0);
  DefineDouble("kappa",&kappa);
  DefineDouble("nc",&nc);
  DefineDouble("beta tau",&betatau);
  DefineDouble("Pc",&pc);
  DefineDouble("omega",&omega);
  StartMenu("Restart",0);
  DefineMod("xdim",&size, xdim+1);
  DefineFunction("Restart",&init);
  EndMenu();
  DefineGraph(curve2d_,"Density graph");
  DefineBool("pause",&Pause);
  DefineBool("Single step", &sstep);
  DefineInt("Repeat", &Repeat);
  DefineBool("Done", &done);
  EndMenu();
}

int main()
{
  int i;
  init();
  GUI();
  while (!done)
    {
      Events(1);
      DrawGraphs();
    if (!Pause||sstep)
      {
sstep=0;
for (i=0;i<Repeat;i++)iterate();
      }
    else
      {
sleep(1);
      }
    }
  return 0;
}
```

```
*/

int main(){
  FILE* pt;
  int i;
  pt = fopen("HoldychDensityPressure.dat","w");
  init();
  Analytical();
    for(i=0; i<100000; ++i)
      {
iterate();
      }
    for(i=0; i<100; i++){
      fprintf(pt,"%d  %f  %f  %f %f  %f %f \n", i, n_an[i], n[i],
 pressure_an, p[i], chem_an,  mu[i]);
      }
    fclose(pt);
    return 0;
}
```

APPENDIX I
CODE FOR LIQUID-GAS SYSTEM (FORCINGQ)

```
/*This code treats the pressure as two parts: pi, ideal gas pressure
and pni, non ideal gas pressure. The effect of  ideal gas pressure
is introduced into the LBE as presure equaling to  n/3. The effect
of the non ideal gas pressure is introduced into the LBE as  force
equaling to  na = divergence of the non ideal gas pressure.
Also the Q term correction is included.
na=(tau-1/2)\partial^2(nu^3).
F0= 2*tau*u*na; F1=-tau*(u+0.5)*na; F2=-tau*(u-0.5)*na;
also feq_i = f^o_i - F_i;  */
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>
#include<math.h>
#include<mygraph.h>
#define SIZE 100
static int i, size = SIZE, Repeat = 1, done= 0, sstep = 0,
Pause=1,nreq=0,iterations=0;
static double f0[SIZE],f1[SIZE],f2[SIZE], n[SIZE], dn[SIZE],
       ddn[SIZE],chem[SIZE], pni[SIZE],nu[SIZE],na[SIZE],
Q[SIZE],ddQ[SIZE],dpni[SIZE], u[SIZE], p[SIZE], kappa = 0.1,
 nc = 1.0, pc = 0.125, betatau=0.03, u0=0, omega =1.0;
double chem_an, pressure_an, n_an[SIZE];
Analytical(){
  chem_an = 4*pc*(1-betatau)/nc;
  pressure_an = pc* (1-betatau)*(1-betatau);
  for(i=0; i<100; i++)
    n_an[i]=nc*(1+sqrt(betatau)*tanh((50-i)
/sqrt(kappa*nc*nc/(2*betatau*pc))));
}
FirstOrderDerivative(double *a,double *da, int n){
  int i;
  da[0]=(a[1]-a[n-1])/2.0;
  da[n-1]=(a[0]-a[n-2])/2.0;
  for(i= 1; i<n-1; ++i)
    da[i]=(a[i+1]-a[i-1])/2.0;
}
void SecondOrderDerivative( double a[], double dda[], int n){
  int i;
  dda[0]= (a[1]+a[n-1]-2.0*a[0]);
  dda[n-1]=(a[0]+a[n-2]-2*a[n-1]);
  for(i=1; i<n-1; ++i)
  dda[i] = (a[i+1] + a[i-1] - 2.0*a[i]);
}
void init(void){
  int i;
  iterations=0;
  for(i=0; i < size; i++)
    {
    n[i] = 1 + 0.01*sin(2*M_PI*i/size);
    nu[i] = (n[i] - nc)/nc;
    u[i]=(f1[i]-f2[i])/n[i];
    Q[i]=n[i]*u[i]*u[i]*u[i];
    }
  FirstOrderDerivative(n,dn,size);
  SecondOrderDerivative(n,ddn,size);
  SecondOrderDerivative(Q,ddQ,size);
  for (i=0;i<size;i++){
    p[i]=pc*(nu[i]+1)*(nu[i]+1)*(3*nu[i]*nu[i]-2*nu[i]+1-2*betatau)
      -kappa*ddn[i] + kappa*dn[i]*dn[i]/2.0;
    chem[i]= 4*pc*(nu[i]+1)*(nu[i]*nu[i]-nu[i]+1-betatau)/nc
 - kappa*ddn[i];
    pni[i] = p[i] - n[i]/3;
    u[i] = u0;
    f0[i] = omega * (n[i]*2/3 - n[i]*u[i]*u[i])+2.0*u[i]*dpni[i]
-2.0*u[i]*na[i];
    f1[i] = omega * (n[i]/3.0+n[i]*u[i] + n[i]*u[i]*u[i])/2.0
-(u[i]+0.5)*dpni[i]+(u[i]+0.5)*na[i];
    f2[i] = omega * (n[i]/3.0-n[i]*u[i] + n[i]*u[i]*u[i])/2.0
-(u[i]-0.5)*dpni[i]+(u[i]-0.5)*na[i];
    }
}
void  iterate(void){
  int i=0;
  double temp1, temp2 ;
  for (i=0;i<size;i++){
    n[i]=f0[i]+f1[i]+f2[i];
    nu[i] = (n[i] - nc)/nc;
    u[i]=(f1[i]-f2[i])/n[i];
    Q[i]=n[i]*u[i]*u[i]*u[i];
  }
  FirstOrderDerivative(n,dn,size);
  SecondOrderDerivative(n,ddn,size);
  SecondOrderDerivative(Q,ddQ,size);
  for (i=0;i<size;i++){
    p[i]=pc*(nu[i]+1)*(nu[i]+1)*(3*nu[i]*nu[i]-2*nu[i]+1-2*betatau)
      -kappa*ddn[i] + kappa*dn[i]*dn[i]/2.0;
     chem[i]= 4*pc*(nu[i]+1)*(nu[i]*nu[i]-nu[i]+1-betatau)/nc;
    pni[i] = p[i] - n[i]/3;
    na[i]=(1.0/omega-0.5)*ddQ[i];
  }
  FirstOrderDerivative(pni,dpni,size);
  for (i=0;i<size;i++){
      f0[i]+=omega*(n[i]*2/3-n[i]*u[i]*u[i]-f0[i])+2.0*u[i]*dpni[i]
-2.0*u[i]*na[i];
      f1[i] += omega * (n[i]/6+n[i]*u[i]/2+n[i]*u[i]*u[i]/2 -f1[i])
-(u[i]+0.5)*dpni[i]+(u[i]+0.5)*na[i];
      f2[i] += omega * (n[i]/6-n[i]*u[i]/2+n[i]*u[i]*u[i]/2 -f2[i])
-(u[i]-0.5)*dpni[i]+(u[i]-0.5)*na[i];
    }
  temp1 = f1[size-1];
  temp2=f2[0];
  for(i=1; i<size; i++){
      f1[size-i]=f1[size-i-1];
      f2[i-1]=f2[i];
    }
  f1[0]=temp1;
  f2[size-1]=temp2;
  ++iterations;
}
/*  graphic interface by A. Wagner to dispaly evolution.
void GUI(){
  DefineGraphN_R("Density", n, &size, &nreq);
  DefineGraphN_R("U", u, &size, &nreq);
  DefineGraphN_R("pni",pni,&size,&nreq);
  DefineGraphN_R("dpni",dpni,&size,&nreq);
  DefineGraphN_R("p",p,&size,&nreq);
  StartMenu("GUI",1);
  DefineInt("Iterations",&iterations);
  DefineDouble("betatau",&betatau);
  DefineDouble("pc",&pc);
  DefineDouble("kappa",&kappa);
  DefineDouble("u0",&u0);
  DefineDouble("nc",&nc);
  DefineDouble("omega",&omega);
  StartMenu("Restart",0);
  DefineMod("size",&size, size+1);
  DefineFunction("Restart",&init);
  EndMenu();
  DefineGraph(curve2d_,"Density graph");
  DefineBool("pause",&Pause);
  DefineBool("Single step", &sstep);
  DefineInt("Repeat", &Repeat);
  DefineBool("Done", &done);
  EndMenu();
}
int main()
{
  int i;
  init();
  GUI();
  while (!done){
    Events(1);
    DrawGraphs();
    if (!Pause||sstep){
      sstep=0;
      for (i=0;i<Repeat;i++)iterate();
    } else {
      sleep(1);
    }
  }
  return 0;
}
*/
int main(){
  FILE * pt;
  int i;
  pt = fopen("FQ.dat", "w");
  init();
  for(i=0; i<10000; i++){
    iterate();
  }
   Analytical();
  for(i=0; i<100; i++){
    fprintf(pt, "%d  %f  %f  %f  %f  %f  %f\n", i, n_an[i], n[i],
pressure_an, p[i], chem_an, chem[i]);
  }
  fclose(pt);
  return 0;
}
```

APPENDIX J
CODE FOR BINARY SYSTEM SIMULATION

```
/* This code is to simulate a binary system with my LB approach.*/
 #include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>
#include<math.h>
#include<mygraph.h>
#define PRECISION 1e-10
#define xdim 100
static int  i, size = xdim, Repeat = 1, done=0, sstep=0,
Pause=1, nreq=0, iterations=0;
double  nA[xdim], nB[xdim], fA0[xdim], fA1[xdim], fA2[xdim],
M=1.0,A=1.0,F[xdim],P[xdim],chemA[xdim],chemB[xdim],
nAa[xdim], nBa[xdim], rhoA[xdim], rhoB[xdim], fB0[xdim],
fB1[xdim], fB2[xdim], u[xdim], n[xdim], FA0[xdim],
FA1[xdim], FA2[xdim],FB0[xdim],FB1[xdim], FB2[xdim],
dchemA[xdim], dchemB[xdim], phiA[xdim], phiB[xdim],
drhoA[xdim], drhoB[xdim];
double chi = 0.94, omega = 0.9,  kappa = 0.1,  h = 0.5,
mA=10,mB=1,density=100,rho[xdim],dphiA[xdim],dphiB[xdim],
   T=0.3333333333, Amplitude=0.1, theta=0.3333333333;
void FirstOrderDerivative(double *a, double *da, int n){
  int i;
  da[0]=(a[1]-a[n-1])/2.0;
  da[n-1]=(a[0]-a[n-2])/2.0;
  for(i= 1; i<n-1; ++i)
  da[i]=(a[i+1]-a[i-1])/2.0;
}
void SecondOrderDerivative( double a[], double dda[], int n){
  int i;
  dda[0]= (a[1]+a[n-1]-2.0*a[0]);
  dda[n-1]=(a[0]+a[n-2]-2*a[n-1]);
  for(i=1; i<n-1; ++i)
   dda[i] = (a[i+1] + a[i-1] - 2.0*a[i]);
}
void Initial(){
  Repeat=1; done=0; sstep=0; Pause=1; nreq=0; iterations=0;
  for(i=0; i < xdim; i++)
   {
     phiB[i] = 0.759 + Amplitude * sin(2*M_PI*i/xdim);
     phiA[i] = 1- phiB[i];
     rhoA[i] = density*phiA[i];
     rhoB[i]= phiB[i] * density;
     nA[i]=rhoA[i]/mA;
     nB[i] = rhoB[i]/mB;
     u[i] = 0;
     fA0[i] = rhoA[i]*2.0/3.0 - rhoA[i]*u[i]*u[i];
     fA1[i] = 0.5*(rhoA[i]/3.0+rhoA[i]*u[i]+rhoA[i]*u[i]*u[i]);
     fA2[i] = 0.5*(rhoA[i]/3.0-rhoA[i]*u[i]+rhoA[i]*u[i]*u[i]);
     fB0[i] = rhoB[i]*2.0/3.0 - rhoB[i]*u[i]*u[i];
     fB1[i] = 0.5*(rhoB[i]/3.0+rhoB[i]*u[i]+rhoB[i]*u[i]*u[i]);
     fB2[i] = 0.5*(rhoB[i]/3.0-rhoB[i]*u[i]+rhoB[i]*u[i]*u[i]);
     rho[i] = rhoA[i]+rhoB[i];
     chemA[i]= theta*(-mA + log(phiA[i]) + (1 - mA/mB)* phiB[i]
+ chi*mA*phiB[i]*phiB[i])/mA;
     chemB[i]= theta*(-mB + log(phiB[i]) + (1 - mB/mA)* phiA[i]
+ chi*mB*phiA[i]*phiA[i])/mB;
     F[i]= theta * (- rhoA[i]-rhoB[i] + rhoA[i]*log(phiA[i])/mA
+ rhoB[i]*log(phiB[i])/mB +chi*rhoA[i]*phiB[i]);
   }
}
void Iterate(void){
  int i=0;
  double temp1, temp2;
  iterations++;
  for(i=0; i<size; i++){
      rhoA[i] =  fA0[i] + fA1[i] + fA2[i];
      rhoB[i] =  fB0[i] + fB1[i] + fB2[i];
      nA[i]=rhoA[i]/mA;
      nB[i] = rhoB[i]/mB;
      rho[i]  =  rhoA[i] + rhoB[i];
      u[i]  =  (fA1[i] + fB1[i] - fA2[i] - fB2[i])/rho[i];
      phiA[i] = rhoA[i]/rho[i];
      phiB[i] = rhoB[i]/rho[i];
      F[i]= theta*(-rhoA[i]-rhoB[i]+rhoA[i]*log(phiA[i])/mA
+ rhoB[i]*log(phiB[i])/mB +chi*rhoA[i]*phiB[i]);
    }
  FirstOrderDerivative(rhoA,drhoA,size);
  FirstOrderDerivative(rhoB,drhoB,size);
  FirstOrderDerivative(phiA,dphiA,size);
  FirstOrderDerivative(phiB,dphiB,size);
   for(i=0; i<size; i++){
      chemA[i]=theta*(-mA+log(phiA[i])+(1 - mA/mB)* phiB[i]
+ chi*mA*phiB[i]*phiB[i])/mA;
```

```
      chemB[i]=theta*(-mB+log(phiB[i])+(1 - mB/mA)* phiA[i]
+ chi*mB*phiA[i]*phiA[i])/mB;
     }
 FirstOrderDerivative(chemA,dchemA,size);
 FirstOrderDerivative(chemB,dchemB,size);
   for(i=0; i<size; i++){
nAa[i]=-(1-0.5*omega)*(rhoA[i]*dchemA[i]-theta*drhoA[i]);
nBa[i]=-(1-0.5*omega )*(rhoB[i]*dchemB[i]-theta*drhoB[i]);
P[i] = rhoA[i]*chemA[i] + rhoB[i]*chemB[i] - F[i];
     }
   /* The forcing terms */
   for(i=0; i<size; i++){
       FA0[i]=-2*nAa[i]*u[i];
       FA1[i]= (u[i]+0.5)*nAa[i];
       FA2[i]= (u[i]-0.5)*nAa[i];
       FB0[i]= -2*nBa[i]*u[i];
       FB1[i]= (u[i]+0.5)*nBa[i];
       FB2[i]= (u[i]-0.5)*nBa[i];
     }
   /*The collision */
   for(i=0; i<size; i++){
       fA0[i] += omega*(rhoA[i]*2.0/3.0 - rhoA[i]*u[i]*u[i]
-fA0[i]) + FA0[i] ;
       fA1[i] += omega*(0.5*(rhoA[i]/3.0 + rhoA[i]*u[i]
+ rhoA[i]*u[i]*u[i])-fA1[i]) + FA1[i] ;
       fA2[i] += omega*(0.5*(rhoA[i]/3.0 - rhoA[i]*u[i]
+ rhoA[i]*u[i]*u[i])-fA2[i]) + FA2[i] ;
       fB0[i] +=omega*(rhoB[i]*2.0/3.0-rhoB[i]*u[i]*u[i]-fB0[i])
+ FB0[i];
       fB1[i] += omega*(0.5*(rhoB[i]/3.0 + rhoB[i]*u[i]
+ rhoB[i]*u[i]*u[i])-fB1[i]) + FB1[i] ;
       fB2[i] += omega*(0.5*(rhoB[i]/3.0 - rhoB[i]*u[i]
+ rhoB[i]*u[i]*u[i])-fB2[i]) + FB2[i] ;
     }
  /* the advection */
 temp1 = fA1[size-1];
  temp2=fA2[0];
  for(i=1; i<size; i++){
      fA1[size-i]=fA1[size-i-1];
      fA2[i-1]=fA2[i];
    }
  fA1[0] = temp1;
  fA2[size-1] = temp2;
  temp1 = fB1[size-1];
  temp2 = fB2[0];
  for(i=1; i<size; i++){
      fB1[size-i] = fB1[size-i-1];
      fB2[i-1] = fB2[i];
    }
  fB1[0] = temp1;
  fB2[size-1] = temp2;
}
/* graphic interface be A. Wagner to display evolution */
void GUI()
{
  DefineGraphN_R("phiA", phiA, &size, &nreq);
  DefineGraphN_R("u", u, &size, &nreq);
  DefineGraphN_R("phiB", phiB, &size, &nreq);
  DefineGraphN_R("rho", rho, &size, &nreq);
  DefineGraphN_R("chemA",chemA, &size, &nreq);
  DefineGraphN_R("chemB",chemB, &size, &nreq);
  DefineGraphN_R("P",P, &size, &nreq);
  StartMenu("GUI",1);
  DefineInt("iterations",&iterations);
  DefineDouble("mA", &mA);
  DefineDouble("mB", &mB);
  DefineDouble("A", &A);
  DefineDouble("kappa", &kappa);
  DefineDouble("chi",&chi);
  DefineDouble("Amplitude",&Amplitude);
  DefineDouble("omega",&omega);
  StartMenu("Restart",0);
  DefineMod("size",&size, size+1);
  DefineFunction("Restart", &Initial);
  EndMenu();
  DefineGraph(curve2d_,"Density graph");
  DefineBool("pause",&Pause);
  DefineBool("Single step", &sstep);
  DefineInt("Repeat", &Repeat);
  DefineBool("Done", &done);
  EndMenu();
}
int main(){
  int i;
  Initial();
  GUI();
  while (!done){
    Events(1);
    DrawGraphs();
    if (!Pause||sstep){
      sstep=0;
      for (i=0;i<Repeat;i++) Iterate();
    } else {
      sleep(1);
    }
  }
  return 0;
}
*/
int main(){
  int i;
  FILE* pt;
  pt = fopen("biLBm10Qun.dat", "w");
      Initial();
      for(i=0; i<500000; i++)
  Iterate();
      for(i=0; i<size; i++)
      fprintf(pt, "%d %e %e %e %e %e \n",i,phiA[i],
                phiB[i], chemA[i], chemB[i], P[i]);
  fclose(pt);
}
```

APPENDIX K
CODE FOR TERNARY SYSTEM SIMULATION

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>
#include<math.h>
#include<mygraph.h>
#define PRECISION 1e-10
#define xdim 100
static int i,size=xdim,Repeat=1,done=0,sstep=0,Pause=1,
  nreq=0, iterations=0;
double  fA0[xdim], fA1[xdim], fA2[xdim],fB0[xdim], fB1[xdim],
  fB2[xdim], fC0[xdim], fC1[xdim], fC2[xdim],
  M=1.0,A=1.0,/*A controls chemical potential in simulation.*/
  chemA[xdim], chemB[xdim], chemC[xdim],  nAa[xdim], nBa[xdim],
  nCa[xdim],rhoA[xdim],rhoB[xdim],rhoC[xdim], u[xdim], IA=0.14,
  IB=0.11, /* IA and IB are the initial phiA and phiB */
  FA0[xdim],FA1[xdim], FA2[xdim], FB0[xdim],FB1[xdim],
  FB2[xdim],FC0[xdim], FC1[xdim],FC2[xdim],dchemA[xdim],
  dchemB[xdim], dchemC[xdim],phiA[xdim], phiB[xdim],
  phiC[xdim],drhoA[xdim],drhoB[xdim],drhoC[xdim];
double  omega = 0.9,kappa=0.1,h=0.5,mA=10,mB=1,mC=1,chiAB=3,
  chiAC=0.5, chiBC=0.2, density=100, rho[xdim],dphiA[xdim],
  dphiB[xdim],dphiC[xdim],T=0.3333333333,
  Amplitude=0.1, theta=0.3333333333;
/*find the first order derivative*/
void FirstOrderDerivative(double *a, double *da, int n){
  int i;
  da[0]=(a[1]-a[n-1])/2.0;
  da[n-1]=(a[0]-a[n-2])/2.0;
  for(i= 1; i<n-1; ++i)
  da[i]=(a[i+1]-a[i-1])/2.0; }
/*find the second order derivative*/
void SecondOrderDerivative( double a[], double dda[], int n){
  int i;
  dda[0]= (a[1]+a[n-1]-2.0*a[0]);
  dda[n-1]=(a[0]+a[n-2]-2*a[n-1]);
  for(i=1; i<n-1; ++i)
   dda[i] = (a[i+1] + a[i-1] - 2.0*a[i]); }
/*set up intial conditions*/
```

```
void Initial(){
  Repeat=1; done=0; sstep=0; Pause=1; nreq=0; iterations=0;
  for(i=0; i < xdim; i++){
     phiA[i] = IA + Amplitude * sin(2*M_PI*i/xdim);
     phiB[i] = IB - Amplitude * sin(2*M_PI*i/xdim) ;
     phiC[i] = 1 - phiA[i] - phiB[i];
     rhoA[i] = density*phiA[i];
     rhoB[i] = density*phiB[i];
     rhoC[i] = density*phiC[i];
     u[i] = 0;
     fA0[i] = rhoA[i]*2.0/3.0  - rhoA[i]*u[i]*u[i];
     fA1[i]=0.5*(rhoA[i]/3.0+rhoA[i]*u[i]+rhoA[i]*u[i]*u[i]);
     fA2[i]=0.5*(rhoA[i]/3.0-rhoA[i]*u[i]+rhoA[i]*u[i]*u[i]);
     fB0[i]=rhoB[i]*2.0/3.0  - rhoB[i]*u[i]*u[i];
     fB1[i]=0.5*(rhoB[i]/3.0+rhoB[i]*u[i]+rhoB[i]*u[i]*u[i]);
     fB2[i]=0.5*(rhoB[i]/3.0-rhoB[i]*u[i]+rhoB[i]*u[i]*u[i]);
     fC0[i]=rhoC[i]*2.0/3.0-rhoC[i]*u[i]*u[i];
     fC1[i]=0.5*(rhoC[i]/3.0+rhoC[i]*u[i]+rhoC[i]*u[i]*u[i]);
     fC2[i]=0.5*(rhoC[i]/3.0-rhoC[i]*u[i]+rhoC[i]*u[i]*u[i]);
     rho[i] = rhoA[i]+rhoB[i]+rhoC[i];
    chemA[i]= A*theta*(log(phiA[i])-phiA[i] - phiB[i]* mA/mB
           -phiC[i]*mA/mC+mA*(phiB[i]+phiC[i])*(chiAB*phiB[i]
      + chiAC*phiC[i])- mA*chiBC*phiB[i]*phiC[i])/mA;
    chemB[i]= A*theta*(log(phiB[i]) - phiB[i] - phiC[i]* mB/mC
            -phiA[i]*mB/mA+mB*(phiC[i]+phiA[i])*(chiBC*phiC[i]
      + chiAB*phiA[i])-mB*chiAC*phiC[i]*phiA[i])/mB;
    chemC[i]= A*theta*(log(phiC[i])-phiC[i]-phiA[i]* mC/mA
            -phiB[i]*mC/mB+mC*(phiA[i]+phiB[i])*(chiAC*phiA[i]
               + chiBC*phiB[i] - mC*chiAB*phiA[i]*phiB[i])/mC;}
}
/*interates in LB algorithm*/
void Iterate(void){
  int i=0;
  double temp1, temp2;
  iterations++;
 for(i=0; i<size; i++){
      rhoA[i] =  fA0[i] + fA1[i] + fA2[i];
      rhoB[i] =  fB0[i] + fB1[i] + fB2[i];
      rhoC[i] =  fC0[i] + fC1[i] + fC2[i];
      rho[i]  =  rhoA[i] + rhoB[i] + rhoC[i];
      u[i]=(fA1[i]+fB1[i]+fC1[i]-fA2[i]-fB2[i]-fC2[i])/rho[i];
      phiA[i] = rhoA[i]/rho[i];
      phiB[i] = rhoB[i]/rho[i];
      phiC[i] = rhoC[i]/rho[i];
    }
  FirstOrderDerivative(rhoA,drhoA,size);
  FirstOrderDerivative(rhoB,drhoB,size);
  FirstOrderDerivative(rhoC,drhoC,size);
  FirstOrderDerivative(phiA,dphiA,size);
  FirstOrderDerivative(phiB,dphiB,size);
  FirstOrderDerivative(phiC,dphiC,size);
for(i=0; i<size; i++){
    chemA[i]= A*theta*(log(phiA[i]) - phiA[i] - phiB[i]* mA/mB
- phiC[i]*mA/mC + mA*(phiB[i]+phiC[i])*(chiAB*phiB[i]
+ chiAC*phiC[i]) - mA*chiBC*phiB[i]*phiC[i])/mA;
    chemB[i]= A*theta*(log(phiB[i])-phiB[i]-phiC[i]* mB/mC
-phiA[i]*mB/mA+mB*(phiC[i]+phiA[i])*(chiBC*phiC[i]
 + chiAB*phiA[i])-mB*chiAC*phiC[i]*phiA[i])/mB;
    chemC[i]= A*theta*(log(phiC[i])-phiC[i]-phiA[i]* mC/mA
- phiB[i]*mC/mB+mC*(phiA[i]+phiB[i])*(chiAC*phiA[i]
+ chiBC*phiB[i])- mC*chiAB*phiA[i]*phiB[i])/mC;}
 FirstOrderDerivative(chemA,dchemA,size);
 FirstOrderDerivative(chemB,dchemB,size);
 FirstOrderDerivative(chemC,dchemC,size);
   for(i=0; i<size; i++){
nAa[i]=-(1-0.5*omega)*(rhoA[i]*dchemA[i]-theta*drhoA[i]);
nBa[i]=-(1-0.5*omega)*(rhoB[i]*dchemB[i]-theta*drhoB[i]);
nCa[i]=-(1-0.5*omega)*(rhoC[i]*dchemC[i]-theta*drhoC[i]);
}
   /* The forcing terms */
   for(i=0; i<size; i++){
       FA0[i]=-2*nAa[i]*u[i];
       FA1[i]= (u[i]+0.5)*nAa[i];
       FA2[i]= (u[i]-0.5)*nAa[i];
       FB0[i]= -2*nBa[i]*u[i];
       FB1[i]= (u[i]+0.5)*nBa[i];
       FB2[i]= (u[i]-0.5)*nBa[i];
       FC0[i]=-2*nCa[i]*u[i];
       FC1[i]= (u[i]+0.5)*nCa[i];
       FC2[i]= (u[i]-0.5)*nCa[i];
     }
   /*The collision */  for(i=0; i<size; i++){
       fA0[i] += omega*(rhoA[i]*2.0/3.0 - rhoA[i]*u[i]*u[i]-fA0[i])
+ FA0[i] ;
       fA1[i] += omega*(0.5*(rhoA[i]/3.0 + rhoA[i]*u[i]
+ rhoA[i]*u[i]*u[i])-fA1[i]) + FA1[i] ;
       fA2[i] += omega*(0.5*(rhoA[i]/3.0 - rhoA[i]*u[i]
        + rhoA[i]*u[i]*u[i])-fA2[i]) + FA2[i] ;
       fB0[i] += omega*(rhoB[i]*2.0/3.0 - rhoB[i]*u[i]*u[i]-fB0[i])
 + FB0[i];
```

```
       fB1[i] += omega*(0.5*(rhoB[i]/3.0 + rhoB[i]*u[i]
         + rhoB[i]*u[i]*u[i])-fB1[i]) + FB1[i] ;
       fB2[i] += omega*(0.5*(rhoB[i]/3.0 - rhoB[i]*u[i]
         + rhoB[i]*u[i]*u[i])-fB2[i]) + FB2[i] ;
       fC0[i] += omega*(rhoC[i]*2.0/3.0 - rhoC[i]*u[i]*u[i]-fC0[i])
+ FC0[i] ;
       fC1[i] += omega*(0.5*(rhoC[i]/3.0 + rhoC[i]*u[i]
                + rhoC[i]*u[i]*u[i])-fC1[i]) + FC1[i] ;
       fC2[i] += omega*(0.5*(rhoC[i]/3.0 - rhoC[i]*u[i]
                + rhoC[i]*u[i]*u[i])-fC2[i]) + FC2[i] ;
     }
  /* the advection */
 temp1 = fA1[size-1];
  temp2=fA2[0];
  for(i=1; i<size; i++){
      fA1[size-i]=fA1[size-i-1];
      fA2[i-1]=fA2[i];
    }
  fA1[0] = temp1;
  fA2[size-1] = temp2;
  temp1 = fB1[size-1];
  temp2 = fB2[0];
  for(i=1; i<size; i++){
      fB1[size-i] = fB1[size-i-1];
      fB2[i-1] = fB2[i];
    }
  fB1[0] = temp1;
  fB2[size-1] = temp2;
 temp1 = fC1[size-1];
  temp2=fC2[0];
  for(i=1; i<size; i++){
      fC1[size-i]=fC1[size-i-1];
      fC2[i-1]=fC2[i];
    }
  fC1[0] = temp1;
  fC2[size-1] = temp2;
}
/*graphic interface by A.Wagner to display  evolution
void GUI()
{
  DefineGraphN_R("phiA", phiA, &size, &nreq);
  DefineGraphN_R("u", u, &size, &nreq);
  DefineGraphN_R("phiB", phiB, &size, &nreq);
  DefineGraphN_R("phiC", phiC, &size, &nreq);
  DefineGraphN_R("rho", rho, &size, &nreq);
  DefineGraphN_R("chemA",chemA, &size, &nreq);
  DefineGraphN_R("chemB",chemB, &size, &nreq);
  StartMenu("GUI",1);
  DefineInt("iterations",&iterations);
  DefineDouble("mA", &mA);
  DefineDouble("mB", &mB);
  DefineDouble("A", &A);
  DefineDouble("IA", &IA);
  DefineDouble("IB", &IB);
  DefineDouble("kappa", &kappa);
  DefineDouble("Amplitude",&Amplitude);
  DefineDouble("omega",&omega);
  StartMenu("Restart",0);
  DefineMod("size",&size, size+1);
  DefineFunction("Restart", &Initial);
  EndMenu();
  DefineGraph(curve2d_,"Density graph");
  DefineBool("pause",&Pause);
  DefineBool("Single step", &sstep);
  DefineInt("Repeat", &Repeat);
  DefineBool("Done", &done);
  EndMenu();
} */
int main(){
  int i;
  Initial();
  GUI();
  while (!done){
    Events(1);
    DrawGraphs();
    if (!Pause||sstep){
      sstep=0;
      for (i=0;i<Repeat;i++) Iterate();
    } else {
      sleep(1);
    }
  }
  return 0;
}
```