Numerical Simulation of Air Bubbles in Super-Saturated Water, Lagrangian-Eulerian Approach

by

Gedi Zhou

An essay submitted to The Johns Hopkins University in conformity with the requirements for the degree of Master of Science.

Baltimore, Maryland

August, 2015

Abstract

Super-saturated water body can leads to the death of fish inhabiting it. A possible way to accelerate the reduction of the air concentration in water is to inject bubbles in order to increase the gas-liquid exchange surface. To investigate the effectiveness of this technique, two different methods, i.e., Lagrangian-Eulerian(L-E) and Eulerian-Eulerian(E-E), can be used to simulate the bubble-water mixture. Numerical simulations using these two different methods have been set up, and comparison between the results from these two methods have been conducted. A good match between the two methods when the fluid is quiescent is found, while there is a small difference when the liquid is allowed to flow by the drag of the buoyantly rising bubbles. Possible reasons for these differences will be described.

Primary Reader: Prof. Andrea Prosperetti

Secondary Reader: Prof. Rajat Mittal

Contents

CONTENTS

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Super-saturated water bodies are reported[1] to kill their inhabitants, fish. The process is similar to the one that leads to the so called decompression sickness experienced by deep sea divers, that is, dissolved nitrogen and oxygen form gas bubble inside fish's blood vessel because, in a lower-pressure environment, gas dissolved in the blood tends to come out of solution.

To solve this problem, it is necessary to reduce the amount of super-saturated air in water. However natural processes including diffusion and convection are very inefficient. Diffusion is inefficient because the mass diffusivity of dissolved air in water is very small; convection can bring the water to the surface, but the last few millimetres present a big obstacle because air has to diffuse the layer. In a previous

work, Prosperetti and Geng[2] proposed a way to accelerate the reduction of air by injecting bubbles into the super-saturated water body, and performed some laboratory experiments showing the conceptual feasibility of a method of this type. According to them, injecting bubbles can lead to the reduction of air by two processes:

1. Firstly, the bubbles greatly increase the mass exchange surface between air and water. Dissolved air can be absorbed by the air bubbles by the process of diffusion, and then carried to the atmosphere with the bubbles.

2. Secondly, the bubbles can bring up water from deep regions towards the surface to directly contact with atmosphere. In this process, mixing between low and high concentration regions will also be increased, which in turn leads to the reduction of concentration in more super-saturated region.

The aim of this essay is to perform a full and in-depth examination of this technique by numerical simulation. There are two phases in this system, which are coupled to each other by mass and momentum transfer, and both are subject to two-way forcing, that is, not only are the bubbles affected by the fluid field, but they also provide reaction effects. Currently, there are three possible models to simulate this bubble-water mixture, i.e., fully resolved bubble model, Lagrangian-Eulerian(L-E) model, and Eulerian-Eulerian(E-E) model. Fully resolved bubble model[3,4] treats the bubble surfaces as moving boundaries for the liquid, and applies certain boundary conditions for velocity, pressure and concentration on them. While it captures most

of the physics, it is so computationally expensive that the number of bubbles it is able to simulate is way insufficient with respect to that required by the problem described above. The latter two models are less accurate physically, but they make it possible to simulate realistic conditions with millions of bubbles.

The L-E approach treats the dispersed air bubble phase as point sources and tracks the motion of each bubble individually, while the liquid phase is described using the ordinary Eulerian representation. The E-E approach uses the Eulerian description for both dispersed and liquid phases. We have developed the computational code based on flow solver, the so called *Bluebottle* developed by Adam Sierakowski,[5] and the bubble model is modified from Shigan Chu's original work.[6] Powered by CUDA parallel computing, the code is able to efficiently simulate this multiphase flow with up to tens of millions of bubbles and yield intriguing and instructive results.

1.2 Structure of the essay

This essay will present in detail the works related with the L-E model, including its theoretical basis and numerical implementation. It will also give a brief summary of the governing equations used in E-E model and the difference between L-E and E-E models theoretically. Finally, numerical comparison between these two models will be shown.

The liquid we are concerning about is water under standard conditions. The

bubbles are constituted by air, which is assumed to be an ideal gas. The physical and computational values for all the parameters involved in this problem are summarized in Table 1.1

Table 1.1: Parameter values.

Parameters	Values
ρ_f	10^3 kg/m^3
ν	10^{-6}m^2/s
ρ_0	1.225 kg/m^3
D	2×10^{-9}m^2/s
H	10.3m
$c_{sat,0}$	2.27×10^{-2}kg/m^3
g	9.8kg$\times$ m/s^2

The simulation domain and boundary conditions are chosen carefully in such a way that an experiment water tank or a segment of river can be simulated. Detailed information about this can be found in Chapter 4. Air bubbles are injected at a constant rate through the bottom of the domain by a model of bubble generator, which will be described in section 3.3, and they will disappear immediately as they reach the top the domain as long as the boundary conditions on the top are not periodicity. Bubbles passing a periodic boundary will come back into the domain through the opposing boundary. This is the same as what will happen in reality. There are two computational results we are particularly interested in. The first one is the average dissolved air concentration in the domain, which directly shows the effectiveness and efficiency of the technique of injecting bubble on the reduction of dissolved air supersaturation. The second one is the total bubble number in the

domain, which is closely related to the rate of dissolved air concentration reduction. As mentioned above, bubbles are injected at a constant rate and will disappear after they reach the top of the domain. Given the bubble residence time being the same, the total bubble number should be only dependent on the bubble injection rate. However, the flow prompted by the injection of the bubbles will change the bubble residence time dramatically, thus in turn change the total bubble number. So the total bubble number gives a overall description of the flow field. The comparison between L-E and E-E models will focus on the above two results.

Chapter 2

Theoretical Model

The Lagrangian-Eulerian approach is widely used to calculate phenomena in multiphase flow systems, such as particle laden flow.[7] Its theoretical foundations have been lay down by previous researchers, and its mathematical formation has long been available.[8] There are many possible representations for the dispersed phase,[9] e.g.,point particles,[10] statistically averaged field and so on.[11] The method this essay uses for its greatest part is adopted from previous works by Prosperetti[12] for the specific problem addressed.

The merit of the L-E approach lies in the fact that it finds a good balance between computational complexity and simulation accuracy. It is not as accurate as the methods where air bubbles are fully resolved, but makes it possible to simulate millions of bubbles, as well as showing some detailed flow and bubble features with relatively inexpensive computational costs.

2.1 Momentum equation for water-bubble mixture

We start from the general form of the volume-averaged momentum equation for water-air mixture:

$$3(1-\alpha)\rho_f \frac{d\mathbf{u}}{dt} + \frac{1}{V}\sum_i \frac{d(m_i \boldsymbol{v}_i)}{dt} = \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} + (1-\alpha)\rho_f \mathbf{g} + \frac{1}{V}\sum_i m_i \mathbf{g}, \tag{2.1}$$

where

$$\alpha = \frac{1}{V}\sum_i v_i, \tag{2.2}$$

is the volume fraction of air; $\mathbf{u}$ is the liquid velocity field; $m_i, \boldsymbol{v}_i, v_i$ are the mass, velocity and volume for the i^{th} bubble; V is the volume of the domain. The summations are over all the bubbles. Rewrite the above equation as

$$\rho_f \frac{d\mathbf{u}}{dt} = \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} + \rho_f \mathbf{g} + \alpha\rho_f \left(\frac{d\mathbf{u}}{dt} - \mathbf{g}\right) + \frac{1}{V}\sum_i \left[m_i \mathbf{g} - \frac{d(m_i \mathbf{v}_i)}{dt}\right]. \tag{2.3}$$

On the right hand side, the last term is the reaction force from air bubbles to water, and the second term is due to the volume occupancy by the air bubbles. Furthermore, the equation can be put into the point-particle form, which is

$$\rho_f \frac{d\mathbf{u}}{dt} = \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} + \rho_f \mathbf{g} + \sum_{i=1}^{N_p}\left[m_{f,i}\frac{d\mathbf{u_i}}{dt} - \frac{d(m_i \mathbf{v}_i)}{dt} - (m_{f,i} - m_i)\mathbf{g}\right]\delta(\boldsymbol{x} - \boldsymbol{x}_i), \tag{2.4}$$

where $m_{f,i} = \rho_f v_i$ is the mass of water that would occupy the volume occupied by the air bubble with index i. The average stress in the mixture is approximated by the stress in water, which is treated as an incompressible Newtonian fluid:

$$\boldsymbol{\sigma} = -p + \mu(\boldsymbol{\nabla u} + \boldsymbol{u\nabla}), \tag{2.5}$$

and we use the modified pressure $\tilde{p} = -(p - \rho g z)$,which includes the hydrostatic pressure. Finally the averaged momentum equation is:

$$\rho_f \frac{d\mathbf{u}}{dt} = -\boldsymbol{\nabla}\tilde{p} + \mu\nabla^2\boldsymbol{u} + \mathbf{F}, \tag{2.6}$$

where $\mathbf{F}$ is the total force exerted on the fluid due to the replacement of air bubble for water, and is defined as follows:

$$\mathbf{F} = \sum_{i=1}^{N_p} \left[m_{f,i} \frac{d\mathbf{u_i}}{dt} - \frac{d(m_i \mathbf{v}_i)}{dt} - (m_{f,i} - m_i)\mathbf{g} \right] \delta(\mathbf{x} - \mathbf{x_i}). \tag{2.7}$$

The first term is a correction to the inertia of the liquid, the second term comes from the momentum change of the air bubbles, and the third term is the buoyancy force, which is the dominant term.

The complete form of the mass conservation equation for the liquid phase is:

$$\nabla \cdot \mathbf{u} = \frac{\partial \alpha}{\partial t} + \alpha \nabla \cdot \mathbf{u} + \mathbf{u} \cdot \nabla \alpha. \tag{2.8}$$

The problem this essay deals with has a low volume fraction for the air bubbles part (within 5%). The bubble injection rate we will use also ensure that the rate of change of volume fraction $\partial\alpha/\partial t$ is small. Simple dimensional analysis will show that the third term on the right hand side of equation (2.8) is also small compared to the term on the left hand side. The length scale L for the variances of velocity and volume fraction are about to be the same, because both of them are caused by the injection of air bubbles. Suppose the velocity scale is U we have:

$$\nabla \cdot \mathbf{u} \sim \frac{U}{L}, \tag{2.9}$$

$$\mathbf{u} \cdot \nabla\alpha \sim \alpha\frac{U}{L}, \tag{2.10}$$

so the relative magnitude of these two terms is:

$$\frac{\mathbf{u} \cdot \nabla\alpha}{\nabla \cdot \mathbf{u}} \sim \alpha \ll 1. \tag{2.11}$$

From the above discussion, it can be seen that all the three term on the right hand side of equation 2.8 can be neglected to retain the continuity equation for incompressible flow:

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0. \tag{2.12}$$

2.2 Scalar diffusion equation

We start from the general conservation equation in a control volume:

$$\frac{d}{dt}\int_{V_f} c dV_f = \int_{V_f} Q dV_f - \oint_{S_f} c\boldsymbol{u}\cdot\boldsymbol{n} dS_f - \oint_{S_f} \boldsymbol{j}\cdot\boldsymbol{n} dS_f, \tag{2.13}$$

where c is the density of air dissolved in water, and it has a dimension of kg$\cdot$ m^{-3}. On the right hand side, the first term is the volume source,

$$Q = -\sum_i^{N_p} \frac{dm_i}{dt}\delta(\mathbf{x}-\mathbf{x}_i).$$

The third term is the diffusive transport of mass through the control surface, where $\boldsymbol{j}$ is the average diffusive mass flux, and according to Fick's law,

$$\boldsymbol{j} = -D\boldsymbol{\nabla} c,$$

where D is the mass diffusivity of air in water. So upon volume averaging and using the continuity equation 2.12, the general form of differential equation for concentration of air in water is:

$$(1-\alpha)\frac{dc}{dt} = -\sum_i^{N_p} \frac{dm_i}{dt} + \boldsymbol{\nabla}\cdot((1-\alpha)D\nabla c). \tag{2.14}$$

Again it can be put into the point-particle form, omitting the term $D\alpha$ because $\alpha \ll 1$, then we have:

$$\frac{dc}{dt} = D\nabla^2 c + \sum_i^{N_p} \left[\frac{d(m_{c,i})}{dt} - \frac{dm_i}{dt} \right] \delta(\boldsymbol{x} - \boldsymbol{x}_i), \tag{2.15}$$

where $m_{c,i} = c_i v_i$ is the mass of air that would be dissolved in the volume occupied by i^{th} bubble. Because $c_i << \rho_{air}$, $m_{c,i} << m_i$ and thus is negligible, finally the scalar diffusion equation is

$$\frac{dc}{dt} = D\nabla^2 c - \sum_i^{N_p} \frac{dm_i}{dt} \delta(\boldsymbol{x} - \boldsymbol{x}_i) \tag{2.16}$$

2.3 Air bubble model

The Lagrangian approach is applied here, so the air bubble position $\boldsymbol{x}_i$ can be found by integrating

$$\frac{d\boldsymbol{x}_i}{dt} = \boldsymbol{v}_i. \tag{2.17}$$

The equation of motion for each air bubble is

$$\begin{aligned} \frac{d(\rho_b v \boldsymbol{v})}{dt} = -3\pi C_d \rho_f d_b (\boldsymbol{v} - \boldsymbol{u}) + C_a \rho_f \left[v \left(\frac{d\boldsymbol{u}}{dt} - \frac{d\boldsymbol{v}}{dt} \right) + (\boldsymbol{u} - \boldsymbol{v}) \frac{dv}{dt} \right] \\ + \rho_f v \frac{d\boldsymbol{u}}{dt} + C_l \rho_f v (\boldsymbol{\nabla} \times \boldsymbol{u}) \times (\boldsymbol{v} - \boldsymbol{u}) + (\rho_b - \rho_f) v \boldsymbol{g}. \end{aligned}$$

On the right hand side, the first term is the drag force, the second one is added mass force, the third one is the virtual buoyancy, followed by lift force, gravity and buoyancy. Virtual buoyancy comes from the pressure distribution on the surface of

the bubble due to the acceleration of the fluid. Lift force is neglected, giving

$$\frac{d(m\boldsymbol{v})}{dt} = -3\pi f \mu d_b(\boldsymbol{v}-\boldsymbol{u}) + C_a\rho_f\left[v\left(\frac{d\boldsymbol{u}}{dt}-\frac{d\boldsymbol{v}}{dt}\right)+(\boldsymbol{u}-\boldsymbol{v})\frac{dv}{dt}\right]+(m-m_f)\boldsymbol{g}. \tag{2.18}$$

$C_a = \frac{1}{2}$, and f can be determined from the following experimental correlation:

$$f = 1 + 0.15Re_p^{0.687}, \tag{2.19}$$

where $Re_b = \rho d_b|\boldsymbol{u}-\boldsymbol{v}|/\mu$, is the Reynolds number defined on the basis of velocity difference between the bubble and local fluid. The change of bubble mass is modelled by

$$\frac{dm_i}{dt} = \pi d_b^2 h_i(c_i - c_{sat,i}), \tag{2.20}$$

where $c_{sat,i}$ is the saturation concentration at the surface of bubble with index i. h_i can be determined by the standard correlation

$$Sh = \frac{h_i d_b}{D} = 2 + 0.6Re_b^{1/2}S_c^{1/3}, \tag{2.21}$$

where Sh is the Sherwood number and $S_c = \nu/D$ is the Schmidt number. Therefore the final form of the mass transfer equation is:

$$\frac{dm_i}{dt} = \pi d_b ShD(c_i - c_{sat,i}). \tag{2.22}$$

Air is assumed to be an ideal gas, so ideal gas law can be applied to every bubble:

$$v_i = \frac{m_i R_s T}{p_i}, \tag{2.23}$$

where R_s is the specific gas constant. Volume change of the air bubble can then be described by the following equation:

$$\frac{dv_i}{dt} = \frac{d}{dt}\left(\frac{m_i R_s T}{p_i}\right) = \frac{R_s T}{p_i}\pi d_b Sh D(c_i - c_{sat,i}) - \frac{m_i R_s T}{p_i^2}\boldsymbol{\nabla} p_i \cdot \boldsymbol{v}_i. \tag{2.24}$$

Define $H = p_0/(\rho_f g) \simeq 10.3$ m. By further assuming that temperature is constant and pressure in water is approximately the hydrostatic pressure, $p = p_0 + \rho_f g h$,the above equation can be rewritten as:

$$\frac{dv_i}{dt} = \frac{H}{H+h}\pi d_b Sh D\frac{(c_i - c_{sat,i})}{\rho_0} + \frac{1}{H+h} v_i w_i, \tag{2.25}$$

where $w_i = -\frac{dh}{dt}$ is the vertical velocity of i_{th} bubble. It is preferable to write the above equation in the form of bubble diameter to have a nicer form:

$$\frac{dd_b}{dt} = \frac{2}{1+h/H}\frac{ShD}{d_b}\frac{(c_i - c_{sat,i})}{\rho_0} + \frac{1}{3}\frac{1}{H+h} d_b w_i \tag{2.26}$$

Finally, $c_{sat,i}$ can be computed from Henry's Law

$$p = k_H c, \tag{2.27}$$

where k_H is constant for dissolved air. We assume that the pressure disturbance caused by the flow prompted by the injected bubbles are negligible compared to the hydrostatic pressure. $c_{sat,0}$ is the saturation concentration of dissolved air in water under standard atmospheric pressure, then $c_{sat,i}$ can be represented as

$$c_{sat,i} = (1 + \frac{h}{H})c_{sat,0}. \tag{2.28}$$

2.4 Summary of equations in E-E model

The governing equations for the E-E model also comes from the previous works by Prosperetti,[13,14] where 5 extra field variables are introduced to describe the dispersed bubble phase:

1. n: number density field, which describes the distribution of bubble position;
2. v_b: bubble volume field, which describes the distribution of bubble volume; from v_b, the bubble mass field m_b and bubble diameter field d_b can be directly computed;
3. $\boldsymbol{w}$: bubble velocity filed, which describes the distribution of bubble velocity.

With the definitions given above, the governing equations for E-E models can be developed.

First, the mixture momentum equation is:

$$\rho_f \frac{d\mathbf{u}}{dt} = -\boldsymbol{\nabla}\tilde{p} + \mu\nabla^2\boldsymbol{u} + \mathbf{F}, \tag{2.29}$$

where

$$\mathbf{F} = nv_b\frac{\rho_b - \rho}{\rho}\boldsymbol{g}. \tag{2.30}$$

$\mathbf{F}$ is just the buoyancy force.

Then, the bubble motion equation is:

$$\frac{\partial n}{\partial t} + \nabla \cdot (\boldsymbol{w}n) = 0. \tag{2.31}$$

Bubble number density is allowed to be changed by convection, and the convective velocity is the bubble velocity field, which is explicitly computed from

$$\boldsymbol{w} = \boldsymbol{u} - w_T\frac{\boldsymbol{g}}{\|\boldsymbol{g}\|}, \tag{2.32}$$

where w_T is the terminal velocity field, and is computed by implicitly solving the equation balancing drag and buoyancy:

$$w_T = \frac{\rho_f - \rho_b}{\rho_f}\frac{d_b^2\|\boldsymbol{g}\|}{18\nu_f f}, \tag{2.33}$$

where f is available from equation (2.19).

Finally the mass exchange equation is :

$$\frac{dc}{dt} = D\nabla^2 c - \dot{M}, \tag{2.34}$$

$$\frac{\partial}{\partial t}(\rho v) + \boldsymbol{w} \cdot (\rho v) = \dot{M}, \tag{2.35}$$

$$\dot{M} = \pi d_b Sh D(c - c_{sat,i}). \tag{2.36}$$

The L-E model and E-E model resemble each other in many ways, e.g., the mixture momentum equation and the concentration equation have the same form. However the difference between the two models are noteworthy, and they are the following:

1. The fundamental difference between the two models lies in that the L-E model tracks each bubble, while the E-E model describes bubbles using a density;

2. The L-E model allows bubbles to accelerate and accounts the volume fraction, which adds two extra forcing terms into the mixture momentum equation;

3. The E-E model assumes all the bubbles to be equal in each cell.

2.5 Average model

For the purpose of numerical testing, a simplified average model is developed. The volume average of any quantity q in water phase is defined as,

$$\langle q \rangle(\mathbf{x}, t) = \frac{1}{V_L} \int_{V_L} q(\mathbf{x} + \boldsymbol{\xi}, t) d^3\xi. \tag{2.37}$$

All the quantities in angle bracket are the volume averages of that quantity, and can be computed from the above equation. Assuming all periodicity boundary conditions on the faces of the computational domain, then scalar diffusion equation (2.16) can be integrated over the entire computational domain,

$$\int_{V_L} \frac{\partial c}{\partial t} dV_L = -\sum_i^{N_p} \dot{m}_i \int_V \delta(\mathbf{x} - \mathbf{x}_i) dlV = -\frac{1}{V} \sum_i^{N_p} \dot{m}_i. \tag{2.38}$$

Note that the volume integral of the convection and diffusion terms are zero because of the periodicity boundary conditions and thus don't appear in equation 2.38. Further assume that the rate of change of α is negligible. The above equation can be simplified to:

$$(1-\alpha)\frac{d}{dt}\langle c \rangle = -\frac{1}{V} \sum_i^{N_p} \dot{m}_i. \tag{2.39}$$

Define particle average $\overline{p}$ as

$$\overline{p} = \frac{1}{N_p} \sum_i p_i. \tag{2.40}$$

The above equation then becomes

$$\frac{d}{dt}\langle c \rangle = -\frac{\langle n \rangle}{1-\alpha} \overline{\dot{m}_i}, \tag{2.41}$$

where $\langle n \rangle = N_p/V$ is the average number density in the entire computational domain. Upon putting the expression for $\dot{m}_i$ into the right hand side, we find:

$$\frac{d}{dt}\langle c \rangle = -\frac{\langle n \rangle}{1-\alpha}\overline{\pi d_b ShD(c_i - c_{sat,i})}. \tag{2.42}$$

Define $c_{over,i} = c_i - c_{sat,i}$, which is the concentration over saturation. $c_{sat,i}$ is considered to be constant over the entire computation domain, so we have

$$\langle c_{over} \rangle = \langle c \rangle - c_{sat,i}, \qquad \frac{d}{dt}\langle c_{over} \rangle = \frac{d}{dt}\langle c \rangle, \qquad \overline{c}_{over,i} = \overline{c}_i - c_{sat,i}. \tag{2.43}$$

Equation (35) can then be rewritten in $c_{over,i}$; for convenience, the subscript *over* is omitted from here on,

$$\frac{d}{dt}\langle c \rangle = -\frac{\langle n \rangle}{1-\alpha}\overline{\pi d_b ShDc_i}. \tag{2.44}$$

Let

$$\overline{\pi d_b ShDc_i} = \pi ShD\overline{d}_b\overline{c}_i + [\pi ShDd_b c_i]'. \tag{2.45}$$

If the bubble number is large and they are evenly distributed in the domain, $\langle c \rangle$ and $\overline{c}_i$ are approximately the same; upon neglecting the second term on the right hand side in the above equation, we find:

$$\overline{\pi d_b ShDc_i} = \pi ShD\overline{d}_b\langle c \rangle. \tag{2.46}$$

Finally equation (2.44) becomes

$$\frac{d}{dt}\langle c\rangle = -\frac{\langle n\rangle}{1-\alpha}\pi ShD\bar{d}_b\langle c\rangle. \tag{2.47}$$

Apply particle average on the diameter changing equation (2.26)

$$\frac{d\bar{d}_b}{dt} = \overline{\frac{2}{1+h/H}\frac{ShD}{d_b}\frac{c_i}{\rho_0}} + \frac{1}{3}\overline{\frac{1}{H+h}d_b w_i}. \tag{2.48}$$

The second term in the right hand side accounts for diameter change due to pressure change, which is much smaller the the first term. Moreover, it is exactly zero for a static bubble, so we have

$$\frac{d\bar{d}_b}{dt} = \overline{\frac{2}{1+h/H}\frac{ShD}{d_b}\frac{c_i}{\rho_0}} \tag{2.49}$$

Still we can rewrite it into the combination of two terms:

$$\overline{\frac{2}{1+h/H}\frac{ShD}{d_b}\frac{c_i}{\rho_0}} = \frac{2}{1+\overline{h}/H}\frac{ShD}{\rho_0}\frac{\overline{c}_i}{\overline{d}_b} + \left[\frac{2}{1+h/H}\frac{ShD}{d_b}\frac{c_i}{\rho_0}\right]', \tag{2.50}$$

where the second term is neglected. The final form becomes

$$\frac{d\bar{d}_b}{dt} = \frac{2}{1+\overline{h}/H}\frac{ShD}{\rho_0}\frac{\overline{c}_i}{\overline{d}_b} \simeq \frac{2}{1+\overline{h}/H}\frac{ShD}{\rho_0}\frac{\langle c\rangle}{\overline{d}_b} \tag{2.51}$$

Upon dividing equation (2.47) by equation (2.51), we find:

$$\frac{\langle \dot{c} \rangle}{\overline{\dot{d}_b}} = -\frac{\pi(1+\overline{h}/H)\langle n \rangle \rho_0 \overline{d_b}^2}{2(1-\alpha)} \overline{d_b}^2 = -3\kappa \langle n \rangle \rho_0 \overline{d_b}^2, \tag{2.52}$$

where κ is a positive dimensionless number defined by

$$\kappa = \frac{\pi(1+\overline{h}/H)}{6(1-\alpha)}. \tag{2.53}$$

So

$$\frac{d}{dt}(\langle c \rangle + \kappa \langle n \rangle \rho_0 \overline{d_b}^3) = 0. \tag{2.54}$$

Suppose the initial condition is $\langle c \rangle|_{t=0} = c_0, \overline{d_b}|_{t=0} = d_0$, from the above equation, we have the equality

$$\langle c \rangle + \kappa \langle n \rangle \rho_0 \overline{d_b}^3 = c_0 + \kappa \langle n \rangle \rho_0 d_0^3 = m_0. \tag{2.55}$$

If the pressure variation in the computational domain can be neglected, i.e.,$\overline{h} \ll H, \rho_{bubble} = \rho_0$,

$$\kappa = \frac{\pi}{6(1-\alpha)} = \frac{\pi}{6}\frac{V}{V_f}. \tag{2.56}$$

Then equation (2.55) is equivalent to

$$\langle c \rangle V_f + N_p \frac{\pi}{6} \rho_0 \overline{d_b}^3 = c_0 V_f + N_p \frac{\pi}{6} \rho_0 d_0^3, \tag{2.57}$$

which is just the mass conservation of air in the computational domain: total mass of dissolved air in water and air in bubbles doesn't change. Eliminate $\langle c \rangle$ in equation (2.51) using equation (2.55),

$$\frac{\overline{d_b^*} d\overline{d_b^*}}{1 - \overline{d_b^*}^3} = \frac{\pi Sh}{1-\alpha} D\langle n \rangle d_1 dt, \tag{2.58}$$

where $d_1^3 = d_0^3 + c_0/\kappa\langle n \rangle \rho_0$ is the biggest average diameter the bubbles can ever reach; $\overline{d_b^*} = \frac{\overline{d_b}}{d_1}$. By integrating the above equation and using initial conditions, we find:

$$\begin{aligned} &\frac{1}{3}\ln\left(\frac{d_1 - d_0}{d_1 - \overline{d_b}}\right) + \frac{1}{6}\ln\left(\frac{\overline{d_b}^2 + \overline{d_b} d_1 + d_1^2}{d_0^2 + d_0 d_1 + d_1^2}\right) \\ &- 2\arctan\left(\frac{12 d_1(\overline{d_b} - d_0)}{13 d_1^2 + 16 d_b d_0 + 8 d_1(d_b + d_0)}\right) = \frac{\pi Sh}{1-\alpha} D d_1 \langle n \rangle t. \end{aligned} \tag{2.59}$$

By using equation (2.55), similar result can be found for $\langle c \rangle$. The evolution of $\langle c \rangle$ and $\overline{d_b}$ can be easily found by solving equation 2.59 and the corresponding equation for $\langle c \rangle$, which can be used to approximately test the code with the full model.

Chapter 3

Numerical Implementation

The system of equations is discretized with finite difference methods, and a code was developed accordingly to integrate them to find the numerical solutions. The code was written in C with the parallel computing platform CUDA[15] created by NVIDA. CUDA makes use of the powerful modern GPUs, which increase the computing performance dramatically .

3.1 Flow solver with scalar convection-diffusion equation

The grid arrangement and discretization for the momentum equation follows the convention adopted from the flow solver. A regular Cartesian grid in a staggered-grid arrangement is used, with scalars located at cell centers and each velocity and body

force component at their respective face centers. A second order projection method[16] is used to discretize the incompressible Navier-Stokes equations:

$$\mathbf{u}^* = u^n + \Delta t \left[-(\mathbf{u} \cdot \nabla \mathbf{u})^{n+1/2} + \nu (\nabla^2 \mathbf{u})^{n+1/2} \right], \tag{3.1}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{\Delta t}{\rho_f} \nabla p^{n+1/2}. \tag{3.2}$$

The terms with superscript $n + \frac{1}{2}$ are intermediate terms, which are advanced using the Adams-Bashforth method. To enforce the divergence-free condition in the N-S equations, pressure Poisson equation is obtained by taking the divergence of equation (3.2)

$$\nabla^2 p^{n+1/2} = \rho_f \frac{\nabla \cdot \mathbf{u}^*}{\Delta t}. \tag{3.3}$$

This equation is solved with zero normal gradient boundary conditions on the 6 surfaces of the domain. Finally, the intermediate velocity $\mathbf{u}^*$ is projected onto $\mathbf{u}^{n+1}$ via (3.2).

The convective terms are first written in conservative forms, then discretized using

a second order central difference scheme. The formula for the x-component is :

$$\begin{aligned}(\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{e}_x\Big|_{i,j,k} &= \frac{(u_{i+1,j,k}-u_{i,j,k})^2-(u_{i+1,j,k}+u_{i-1,j,k})^2}{4\Delta x}\\ &+\frac{(u_{i,j+1,k}+u_{i,j,k})(v_{i,j+1,k}+v_{i-1,j+1,k})-(u_{i,j,k}+u_{i,j-1,k})(v_{i,j,k}+v_{i-1,j,k})}{4\Delta x}\\ &+\frac{(u_{i,j,k+1}+u_{i,j,k})(w_{i,j,k+1}+v_{i-1,j,k+1})-(u_{i,j,k}+u_{i,jj,k-1})(2_{i,j,k}+w_{i-1,j,k})}{4\Delta x}.\end{aligned} \tag{3.4}$$

Similarly, the discretized formula for the x-component of the the diffusive terms is:

$$\begin{aligned}(\nabla^2\mathbf{u})\cdot\mathbf{e}_x\Big| &= \frac{u_{i+1,j,k}-2u_{i,j,k}+u_{i-1,j,k}}{\Delta x^2}+\frac{u_{i,j+1,k}-2u_{i,j,k}+u_{i,j-1,k}}{\Delta y^2}\\ &+\frac{u_{i,j,k+1}-2u_{i,j,k}+u_{i,j,k-1}}{\Delta z^2}.\end{aligned} \tag{3.5}$$

The time step size of forward in time central in space(FTCS) scheme for a typical convection-diffusion equation should satisfy the stability constraint,[17] that is:

$$\Delta t \le \min\left\{\frac{\Delta x}{|u|},\frac{\Delta x^2}{2\nu},\frac{2\nu}{u^2},\frac{\Delta y}{|v|},\frac{\Delta y^2}{2\nu},\frac{2\nu}{v^2},\frac{\Delta z}{|w|},\frac{\Delta z^2}{2\nu},\frac{2\nu}{w^2}\right\}. \tag{3.6}$$

To satisfy this condition, the flow solver determines the time step by the following formula:

$$\Delta t = \frac{CFL}{\frac{|u|}{\Delta x}+\frac{2\nu}{\Delta x^2}+\frac{u^2}{2\nu}+\frac{|v|}{\Delta y}+\frac{2\nu}{\Delta y^2}+\frac{v^2}{2\nu}+\frac{|w|}{\Delta z}+\frac{2\nu}{\Delta z^2}+\frac{w^2}{2\nu}}, \tag{3.7}$$

where CFL is a constant between zero and one. In our simulation, we set CFL to be

equal to 1. An extra scalar convection-diffusion equation needs to be solved. Values of the concentration field are located at each cell center. As long as the flow field at a certain time step is known, the scalar equation can be integrated independently. The convective term is firstly written in conservative form:

$$\mathbf{u} \cdot \boldsymbol{\nabla} c = \boldsymbol{\nabla} \cdot (\mathbf{u} c) = \frac{\partial (uc)}{\partial x} + \frac{\partial (vc)}{\partial y} + \frac{\partial (wc)}{\partial z}. \tag{3.8}$$

Since velocity values are located at face centers, while concentration values are located at cell centers, all three velocity components are interpolated to cell center using simple linear interpolation, then a compact first order upwind scheme is used. Suppose the convective term is discretized around a cell center, indexed with i, j, k. Define:

$$u_c = \frac{u_{i-1/2,j,k} + u_{i+1/2,j,k}}{2}, \tag{3.9}$$

$$c_w = \frac{c_{i-1,j,k} + c_{i,j,k}}{2}, \tag{3.10}$$

$$c_e = \frac{c_{i+1,j,k} + c_{i,j,k}}{2}. \tag{3.11}$$

Index $1/2$ indicates that u is face centred, so u_c is just the interpolated x-velocity on cell center, and c_e and c_w are concentrations at two neighbouring face centers in x-direction, where the x-position of c_e is to the left of that of c_w. Then the term

$\partial(uc)/\partial x$ can be discretized as:

$$\text{If } u_c > 0: \quad \left.\frac{\partial(uc)}{\partial x}\right|_{i,j,k} = \frac{c_{i,j,k}u_c - c_w u_{i-1/2,j,k}}{\frac{1}{2}\Delta x},$$
$$\text{If } u_c < 0: \quad \left.\frac{\partial(uc)}{\partial x}\right|_{i,j,k} = \frac{c_e u_{i+1/2,j,k} - c_{i,j,k}u_c}{\frac{1}{2}\Delta x}. \tag{3.12}$$

The other two terms are computed similarly.

The diffusive term is discretized with second order central difference scheme:

$$\left.(\nabla^2 c)\right|_{i,j,k} = \frac{c_{i+1,j,k} - 2c_{i,j,k} + c_{i-1,j,k}}{\Delta x^2} + \frac{c_{i,j+1,k} - 2c_{i,j,k} + c_{i,j-1,k}}{\Delta y^2} + \frac{c_{i,j,k+1} - 2c_{i,j,k} + c_{i,j,k-1}}{\Delta z^2}. \tag{3.13}$$

A second-order variable time step Adams-Bashforth method is also used to integrate the equation over time. Similar stability constraints also apply to the scalar convection-diffusion equation, which is:

$$\Delta t \leq \min\left\{\frac{\Delta x}{|u|}, \frac{\Delta x^2}{2D}, \frac{2D}{u^2}, \frac{\Delta y}{|v|}, \frac{\Delta y^2}{2D}, \frac{2D}{v^2}, \frac{\Delta z}{|w|}, \frac{\Delta z^2}{2D}, \frac{2D}{w^2}\right\}. \tag{3.14}$$

In the simulations, the dominant part of (3.7) and (3.14) is mostly $w^2/2D$ and other similar terms. And since D is much smaller than ν, the time step size determined from (3.14) is mostly much smaller than that from (3.7). There are two ways to handle this. The first way is to use the smaller time step as a sub time step, that

is integrate the N-S equations using the longer time step, then integrate the scalar equation as well as the bubble equations using the smaller time step until it reaches the span of the longer time step. This is more computationally efficient, however, it compromises accuracy especially since there is a two-way coupling between the flow and the bubbles. So the second way is preferred, i.e., using the smaller time step for all the equations.

In the mixture momentum equation, scalar diffusion equation and the bubble momentum equation, there are terms with subscript i, like $\mathbf{u}_i, d\mathbf{u}_i/dt, c_i$, which are fluid properties at the bubble position. All these values are interpolated with a second order Lagrangian interpolation using values from eight neighboring cells. In one dimension, the weight functions are

$$l_x[1] = \frac{x - x_1}{x_2 - x_1}, l_x[2] = \frac{x - x_2}{x_1 - x_2}, l_y[1] = \frac{y - y_1}{y_2 - y_1},$$
$$l_y[2] = \frac{y - y_2}{y_1 - y_2}, l_z[1] = \frac{z - z_1}{z_2 - z_1}, l_z[2] = \frac{z - z_2}{z_1 - z_2},$$

where $x_1, x_2, y_1, y_2, z_1, z_3$ can be the position of the cell center or face center depending on the value type needs to be interpolated. Finally the 3-D weight is

$$w[i][j][k] = \frac{l_x[i]l_y[j]l_z[k]}{\sum_{m,n,p=1}^{2} l_x[m]l_y[n]l_z[p]} \qquad i, j, k = 1, 2. \tag{3.15}$$

3.2 Delta function and source implementation

The source terms have the form $f_b\delta(\mathbf{x}-\mathbf{x}_b)$. To numerically approximate them, value of f_b needs to be available at all the cells in the domain. The conservation of mass or momentum in equation 2.6 and 2.16 requires

$$\int_V f_b\delta(\mathbf{x}-\mathbf{x}_b)dV = \sum_{i,j,k} v_c f_{i,j,k}, \tag{3.16}$$

where V is the volume of the entire domain, v_c is the volume of a cell, and $f_{i,j,k}$ is the value of source terms at cell with index i,j,k. Therefore, we have

$$\frac{f_b}{v_c} = \sum_{i,j,k} f_{i,j,k}. \tag{3.17}$$

Suppose that $f_{i,j,k}$ can be represented by f_b/v_c using some weight function $w_{i,j,k}$, so that $f_{i,j,k} = w_{i,j,k}f_b/v_c$. From the above conservation law, one can conclude that

$$\sum_{i,j,k} w_{i,j,k} = 1. \tag{3.18}$$

The summation is over all the cells in the domain. In the essay two kinds of weight functions have been used. The first one uses the simplest delta function weight, by setting $w_{i,j,k} = 1$ in the cell where the bubble center is and $w_{i,j,k} = 0$ in all the other

cells. The second one uses a Gaussian function

$$g(|\mathbf{x}-\mathbf{x}_i|) = \frac{1}{(2\pi\sigma^2)^{3/2}} e^{-\frac{|\mathbf{x}-\mathbf{x}_i|^2}{2\sigma^2}}, \tag{3.19}$$

where σ is the width of the Gaussian kernel and is taken to be the smallest length of the three sides of each cell. In our simulation, we cut off the Gaussian weight to a cube of 27 cells and rescale it to make sure the conservation law is satisfied. Suppose the index for the cell where the bubble center is is i_0, j_0, k_0, define $\Delta i = i - i_0, \Delta j = j - j_0, \Delta k = k - k_0$, $l^2_{i,j,k} = (\Delta i \Delta x)^2 + (\Delta j \Delta y)^2 + (\Delta k \Delta z)^2$, where $\Delta x, \Delta y, \Delta z$ are the length of the three sides of a cell. If $-1 \leq \Delta i, \Delta j, \Delta k \leq 1$, formula for the Gaussian weight we use is as follow:

$$I = \sum_{-1\leq\Delta i,\Delta j,\Delta k\leq 1} \frac{1}{(2\pi\sigma^2)^{3/2}} e^{-\frac{l^2_{i,j,k}}{2\sigma^2}}, \tag{3.20}$$

$$w_{i,j,k} = \frac{1}{I(2\pi\sigma^2)^{3/2}} e^{-\frac{l^2_{i,j,k}}{2\sigma^2}}; \tag{3.21}$$

otherwise, $w_{i,j,k}$ is zero. By either using the delta or Gaussian weights, the source terms can be mollified to be a field in the domain so that they can be integrated directly in the mixture momentum equation and the scalar diffusion equation. In practice, the Gaussian weight is much smoother than the delta weight, as expected, yet both of them will make the scalar field oscillatory to some extent. Because both of them use a certain range of cells, and there will be a sudden jump when a bubble

moves across the boundary of a cell in this range.

At the early stage of the project, we found another problem which arose when integrating the mass transfer equation (2.22). In a natural diffusion process, if the ambient water body is super-saturated, an air bubble will grow monotonically with time and the concentration in water body will not go below saturation. This is not the case in numerical integration. The mass transfer rate of a bubble will increase with the growth of its diameter. At that time we were using a rather small cell size, so the bubble was able to grow as large as several tens of cell size. Then the mass transfer rate will be so high that the concentration in all the cells where sources are present went below saturation or even below zero in a single time step. This in turn led to the loss of mass of the air bubbles, which made the concentration in those cells to be very high, and the concentration even lower in the next time step. This process repeated, worsened, and come to an end when the mass of the air bubble becomes negative. The reason why this happened is that either weight function has a fixed width, and the bubble radius may go very far beyond this width. Although this rarely happens in the simulations we are running right now, for we are using larger cell size, but it motivated us to use a way to implement the sources which is robust for any cell size.

The method we used was developed by Capecelatro and Desjardins.[18] Besides mollification, they introduced a second step to spread source implementation accord-

ing to a diffusion equation

$$\frac{\partial c}{\partial \tau} = D\nabla^2 c.$$

The purpose of this step is to diffuse the source field from mollification to a length scale comparable to the bubble radius. This is equivalent to having a weight kernel that will change width with the growth of bubble radius. According to their work, the diffusion time τ_f is determined from

$$D\tau_f = \frac{\max(\delta_f^2 - \Delta x^2, 0)}{16 \ln 2}. \tag{3.22}$$

For a mono-disperse system, $\delta_f = 3d_b$, and for a poly-disperse systems, $\delta_f = \max(d_b)$. Simulations with this method have been set up, and the scalar field and mass transfer rate become smoother than those without the diffusion step.

3.3 Bubble motion and bubble generator implementation

Bubble positions are directly integrated from the equation (2.17) using the Euler method. A neat exponential integration technique[19] is used to integrate the bubble motion equation (2.18) to get the bubble velocities. The method is the following.

Equation (2.18) can be written in the following form

$$M\frac{d\boldsymbol{v}}{dt} = -a\boldsymbol{v} + \boldsymbol{F}, \tag{3.23}$$

where

$$M = m + C_a\rho_f v, \tag{3.24}$$

is the "actual" mass of bubbles in the liquid,

$$a = 3\pi C_d\rho_f d_b + \frac{dm}{dt} + C_a\frac{dv}{dt}, \tag{3.25}$$

is the coefficient for all the forces that are proportional to $\boldsymbol{v}$, and

$$\boldsymbol{F} = 3\pi C_d\rho_f d_b\boldsymbol{u} + C_a\rho_f v\frac{d\boldsymbol{u}}{dt} + C_a\rho_f\boldsymbol{u}\frac{dv}{dt} + (m - m_f)\boldsymbol{g}. \tag{3.26}$$

The dominant part in F is $(m - m_f)\boldsymbol{g}$, so F is approximately constant. An equation of the above form can be integrated as follows,

$$\mathbf{v}(t + \delta t) = \exp(-\delta t\frac{a}{M})[\mathbf{v}(t) - \frac{\mathbf{F}}{a}] + \frac{\mathbf{F}}{a}, \tag{3.27}$$

where δt is the integration time step. The term $\mathbf{F}/a$ has the dimension of velocity, which actually is what would be the terminal velocity if the bubble was only subjected to buoyancy. With this integration technique, bubble velocities are guaranteed to

converge to their terminal values when gravity dominates and larger time steps can be used. For accuracy, it is preferable to use a time step that satisfies $\delta t < \frac{M}{a}$. This is a rather strong constraint on the time step, and since all the equations, including the mixture momentum equation, use the same time step, imposing this constraint will greatly increase the computational cost. The inaccuracy cost by using a large time step is most significant at the time of injecting bubbles, where all the bubbles are accelerated from zero to terminal velocity. As a consequence, instead of requiring $\Delta t < \frac{M}{a}$, we inject the bubbles with terminal velocity. Then the difference between bubble velocities and their terminal values are always small, so that using a bigger time step is legitimate.

The bubble generator is a model that injects air bubbles into the domain at a constant rate from the bottom of the domain. It can also determine whether bubbles have moved out of the domain or not and kill those that have moved out. The numerical implementation is basically a data structure and memory management problem.

An array of integers $a[i]$ is used to track whether each bubble is in the domain or not. The corresponding element will be set to 0 if the answer is yes, and 1 otherwise. After each time step, this array will have a value of 0 or 1 for each of its element. Then a second array of integers $b[i]$ is created. The value of k^{th} element of the second array is

$$b[k] = \sum_{0}^{k} a[i],$$

which is just the summation of all the elements in the first array before and including index k. So the value of $b[k]$ is the total number of bubbles that are out of the domain among all the bubbles with index from 0 to k. The last component of the array b is the total number of bubbles that should be killed. The data for all the bubbles are stored in an array of C structure. After obtaining the value of array b, the data of all the bubbles inside the domain can be copied to a new array. In this process values of array a are used to determine whether a certain element of the old bubble structure should be copied or not, and values of array b are used to determine the index of a bubble in the new array,

$$i_{new} = i_{old} - b[i_{old}]. \tag{3.28}$$

Finally the memory used to store the old array is freed.

The tricky part is to find a parallel algorithm to compute b based on a, which is a so called inclusive scan. In serial code, it's trivial. A single loop can be used to sum up a to get b. A parallel algorithm developed by Hillis and Steele (1986)[20] is the current solution for the problem. The following figure shows how it works:

Suppose we need to perform an inclusive scan on a array x with n elements. At the first stage of the algorithm, the operation is:

$$x_1[i] = x[i] + x[i-1] \quad \text{if} \quad i-1 \geq 0,$$

$$x_1[i] = x[i] \qquad\qquad\qquad \text{if} \quad i-1 < 0,$$

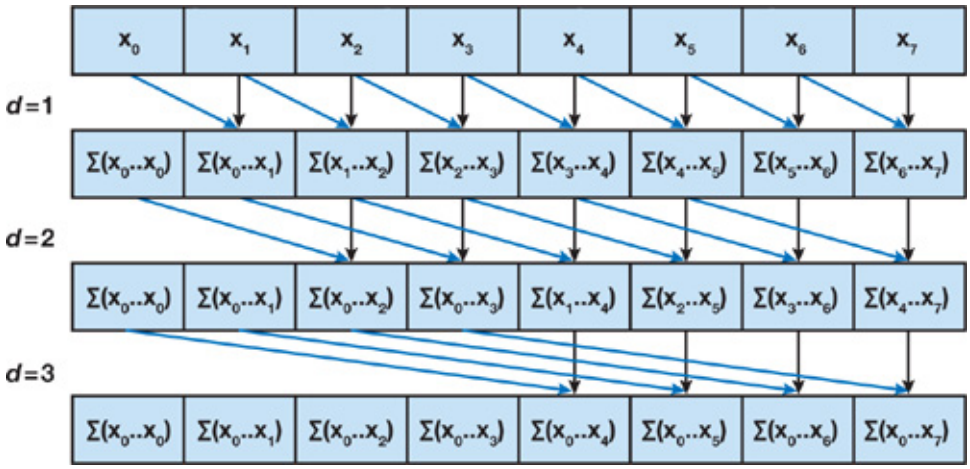

Figure 3.1: Computing a scan of an array of 8 elements using Hillis and Steele scan algorithm

where x_1 is the array after the first stage of operation. At the second stage, the operation becomes:

$$x_2[i] = x_1[i] + x_1[i-2] \quad \text{if} \quad i-2 \geq 0,$$

$$x_2[i] = x_1[i] \qquad\qquad\qquad \text{if} \quad i-2 < 0.$$

At the k^{th} stage, we have:

$$x_k[i] = x_{k-1}[i] + k_{k-1}[i-2] \quad \text{if} \quad i-2^{k-1} \geq 0,$$

$$x_k[i] = x_{k-1}[i] \qquad\qquad\qquad \text{if} \quad i-2^{k-1} < 0.$$

After $\log_2 n$ stages of operation, the array we get is the inclusive scan of the initial array. At every stage, number n addition needs to be done, so the computation complexity of this algorithm is $\mathrm{O}(n \log_2 n)$, which is a great improvement over the serial

algorithm with computational complexity $O(n^2)$. With this method, the memory used to store bubbles data can be allocated and freed dynamically.

The last thing to describe is the bubble injection area. A typical injection area is a square area located at the center of the bottom of the domain. The liquid in the cells where the bubbles are injected will be subjected to forces from the bubbles, and those without will not be. A problem may arise near the edge of bubble injection area, where a large difference in forcing within neighboring cells may occur. In practice this has been found to lead to a divergence of pressure-Poisson equation. To resolve this problem, bubbles are injected in such a way that their positions satisfy the following probability density function:

$$P(x,y,z) = \frac{1}{I}\exp\left(\frac{(z-z_0)^2}{2\sigma_z^2}\right)\left(1+\tanh\frac{x-L_{x1}}{\epsilon_{x1}}\right)\left(1+\tanh\frac{L_{x2}-x}{\epsilon_{x2}}\right) \left(1+\tanh\frac{y-L_{y1}}{\epsilon_{y1}}\right)\left(1+\tanh\frac{L_{y2}-y}{\epsilon_{y2}}\right), \quad (3.29)$$

where I is a normalization factor, which can be find by requiring the integration of P over the entire domain to be 1. $z_0, L_{x1}, L_{x2}, L_{y1}, L_{y2}$ determine the bubble injection area. The bubble injection rate then is given by

$$\dot{N}(x,y,z) = \dot{N}_t P(x,y,z), \quad (3.30)$$

where $\dot{N}(x,y,z)$ is the bubble injection rate at position (x,y,z), and $\dot{N}_t$ is the total bubble injection rate.

Because of the property of the hyperbolic tangent function, the value of this distribution within the region $L_{x1} < x < L_{x2}$ and $L_{y1} < y < L_{y2}$ is almost uniform. Its value outside this region is almost 0. In the area near $x = L_{x1}, x = L_{x2}, y = L_{y1}, y = L_{y2}$, its value decreases, although not sharply, but continuously from $1/I$ to 0, which resolves the problem caused by the discontinuity of the bubble injection boundary. The smoothness of the distribution at boundary is controlled by $\epsilon_{x1}, \epsilon_{x2}, \epsilon_{y1}, \epsilon_{y2}$. By numerical testing, the value we use is $\epsilon_i = L_i/5, i = x1, x2, y1, y2$, so that majority of bubbles will still be injected in the square area, and the edges are smooth enough to prevent the divergence of pressure-Poisson equation.

Numerical realization of this method is by firstly generating certain number of uniformly distributed bubble positions using a pseudo random number generator. Then a filter is used to transform the uniform distributed numbers to the desired distribution. The filter can be found by using the inverse transform method.[21] Suppose X is a random variable that satisfies uniform distribution. One wants to generate a random variable Y with the cumulative distribution function $F(Y)$ by transformation $Y = G(X)$. Then $G = F^{-1}$. The probability distribution function desired is given by equation (3.31), so its cumulative distribution function F can be found by numerically integrating P. We only generate values for F at all the cell centers. F^{-1} is readily given at all the cell centers, and it is a monotonically increasing function. Given a (x_p, y_p, z_p) that is not the position of a cell center, a simple binary search algorithm is used to find the neighboring two cell center positions. The value of F^{-1}

at (x_p, y_p, z_p) can be interpolated from the values of F^{-1} at the two neighboring cell center positions.

3.4 Computational Parcels

In L-E simulations, one possible trick to save computational resources is using computational 'parcels'.[22,23] N_p injected air bubbles can be represented by N_c computational parcels. The motion of every computational parcel is just like that of one single actual bubble, while the total mass transfer and momentum transfer between parcels and liquid is exactly the same as actual air bubbles. N_c isn't necessarily equal to N_p. In fact, it can be a multiple of N_p so as to save computational cost as well as making simulation with more bubbles possible. Validity of this method needs careful examination and rigorous analysis.

Intuitively, the method will make the flow field and concentration field less accurate. However, as long as this inaccuracy is not too large to change the bubble residence time significantly, the process of bubbles absorbing mass out of the liquid should also not be affected too much. The ratio of N_c to N_p is an important factor, and the volume fraction of air bubble should be the other. Numerical results showing the effect of N_c/N_p will be shown later.

Chapter 4

Numerical Results

In this chapter, numerical results from the code built with the numerical methods discussed in the previous chapter will be shown. Firstly, results from two simple verification cases will be shown, followed by results showing the effect of different weight function in the source implementation and the size of computational parcel, and finally an emphasis will be put on the discussion about the comparison between results from the L-E model and the E-E model. The results of the E-E model comes from the works by my colleague Yuhang Zhang.[24,25]

The domain is always a cuboid, so there are 6 boundary surfaces. For the velocity field, no-slip non-penetration boundary condition are used for the bottom surface to simulate the solid bottom of river or tank. No-penetration free-slip boundary condition are used for the top surface to simulate the free surface of water-atmosphere interface, and periodicity boundary conditions are used for the four lateral surfaces.

For the pressure field, zero-gradient boundary conditions are used on both top and bottom domain, and periodicity boundary conditions are used for the four lateral surfaces. For the concentration field, zero gradient boundary conditions are used on both top and bottom surfaces to ensure that no dissolved air can leave the domain via diffusion. Recall that in the velocity boundary condition, the normal velocity at top and bottom boundaries is zero, so that dissolved air can't leave the domain via convection. Thus the only way for the total mass of dissolved air in the domain to decrease is by transferring into the air bubbles and leaving the domain with the bubbles. In all of the following discussions, the above boundary conditions are used unless otherwise specified.

4.1 Comparison with the simplest model

For verification purposes, a numerical simulation with the simplest model was set up. In this model, both the flow and the bubbles remain quiescent all the time, one and only one bubble sits in the center of each cell. With this simplification, the number of equations can be reduced to two, which are,

$$\frac{dc}{dt} = \frac{2\pi d_b D c}{V}, \tag{4.1}$$

$$\frac{dd_b}{dt} = \frac{4Dc}{d_b \rho_b}, \tag{4.2}$$

where V is the volume of one single cell. These two equation can be easily integrated by Matlab ODE solver, and the result can be used as verification.

In the simulation, the domain and bubble position arrangement can be visualized by figure 4.1. Both the flow and the bubble are set to be quiescent to match the condition of the simplest model. The domain size is $2\times2\times2$ cm^3 and the number of cells is $20\times$ $20\times$ 20.

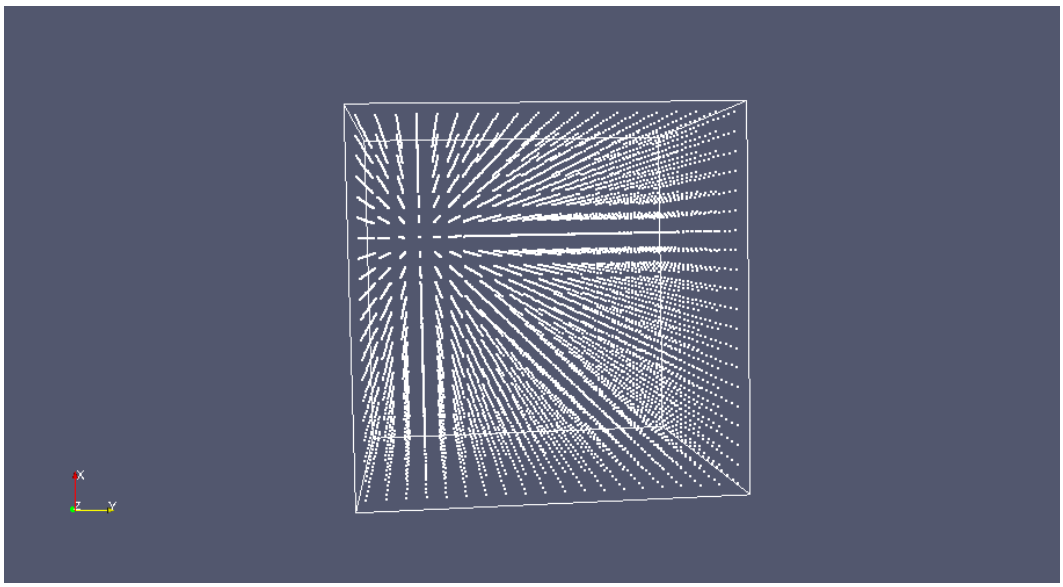

Figure 4.1: Bubble position arrangement

Figures 4.2 and 4.3 show the evolution of the radius of one single bubble and averaged concentration in the domain over time for the results from the simplest model and the L-E simulation, respectively. The concentration is normalized using the saturation concentration. It can be seen clearly from the two plots that the these two methods are indistinguishable. This verifies that the numerical implementation of concentration equation and bubble mass/radius equation are correct.

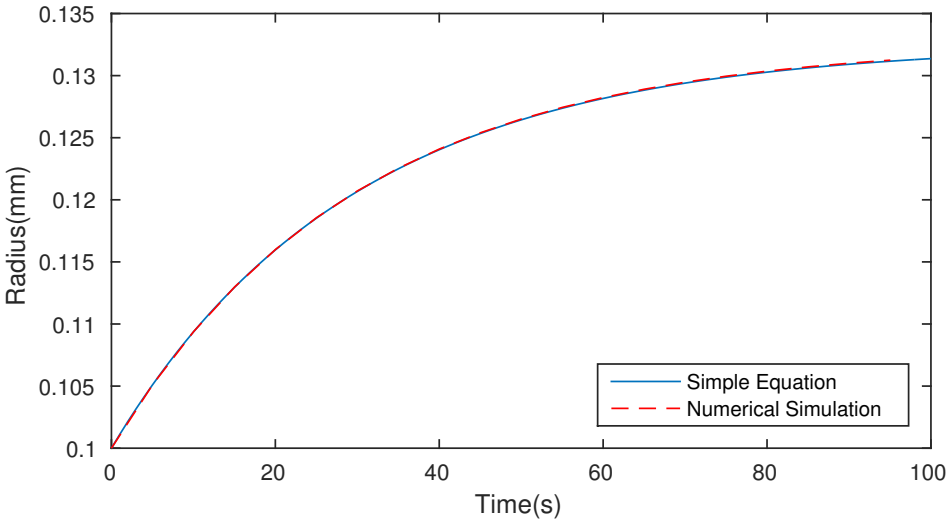

Figure 4.2: Evolution of radius for one bubble over time for result from simple equation integration and numerical simulation

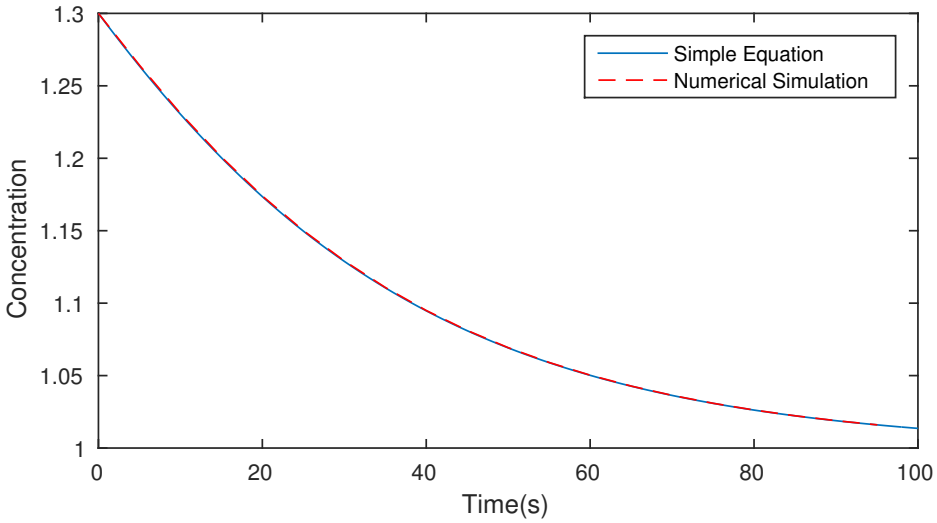

Figure 4.3: Evolution of average concentration over time for result from simple equation integration and numerical simulation

4.2 Comparison with the average model

In Chapter 2 section 2.5, the equations for the evolution of average concentration in the domain, equation 2.47, and average bubble diameter, equation 2.51, have been derived. These two equations can also be easily integrated, and the result used for comparison.

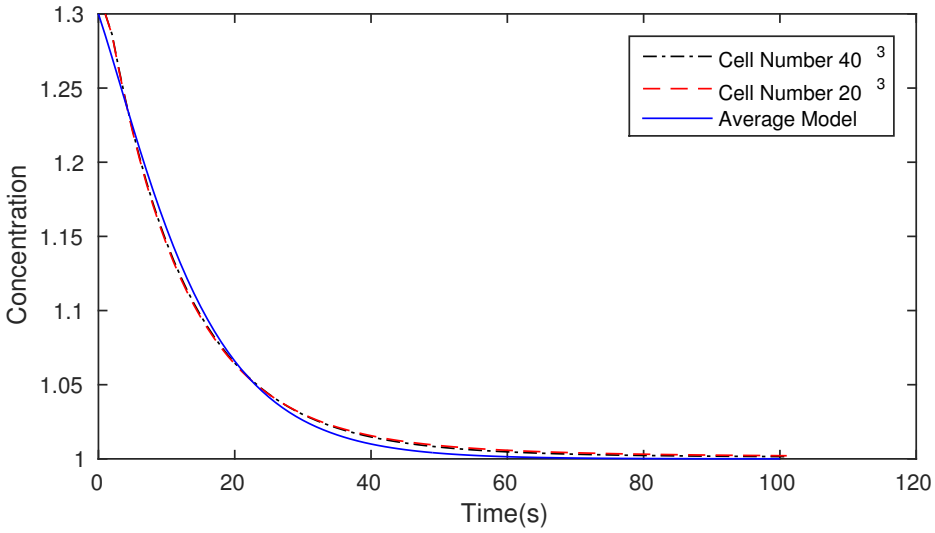

Figure 4.4: Evolution of average concentration over time for result from average equation integration and numerical simulation, bubble injection rate 50000/s

Since the average equation is derived assuming periodicity boundary conditions on all the boundaries, the simulation will also accommodate these boundary conditions. The flow is still set to be quiescent all the time while the bubbles are allowed to move. The domain size is still $2\times\ 2\times 2$ cm^3. Two different grid size with total cell numbers equal to 40^3 and 20^3 are used. In figure 4.4, the blue solid line shows the result from the average model by choosing the Sherwood number to be equal to 40.

This choice accounts for the change of mass transfer rate between a single bubble and the fluid due to the motion of the bubble. Numerical results with different grid size are indistinguishable, which shows grid convergence of the code. Good match between average model and numerical simulation is found.

4.3 Different weight function in source implementation

As mention in section 3.2, there are two weight functions that are implemented in the code, that is delta function and Gaussian function. Simulation conditions are summarized in table 4.1

Table 4.1: Simulation Configuration

Domain Size	4cm $\times$ 4cm $\times$ 10cm(x, y, z)
Resolution	$40 \times 40 \times 100(x, y, z)$
Initial Concentration	10 times saturation
Bubble Initial Radius	0.1 mm
Bubble Injection Rate	50000/s
Liquid allowed to flow	Yes

Note that a very small domain is used, while the initial concentration is very high, both of which are unrealistic. This is because, for numerical verification purposes, the point is not to simulate realistic conditions, but it is to push the numerics to see how robust the results are. The configuration for all the simulations discussed from here on are the same as shown in table 4.1 unless otherwise specified.

Figure 4.5 shows clearly that the difference due to the use of different weight function is negligible. Since the Gaussian weight function is more computationally

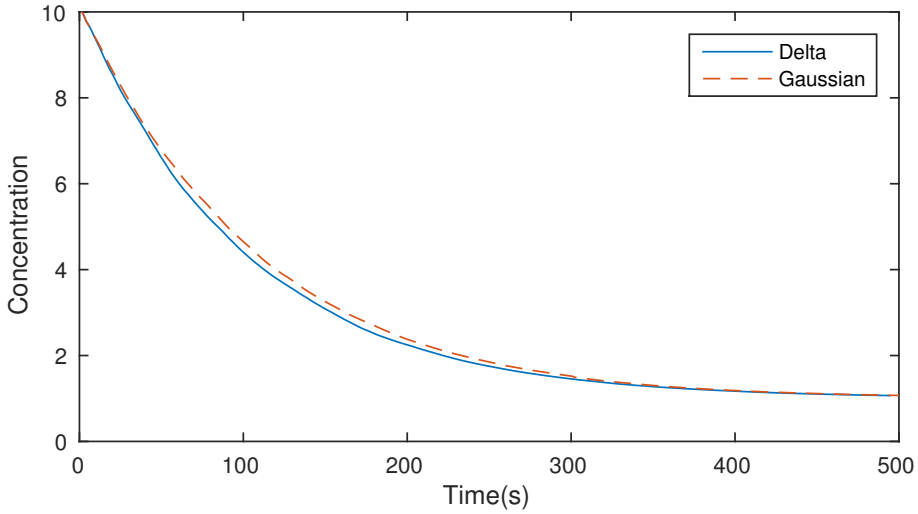

Figure 4.5: Evolution of average concentration with Delta and Gaussian weight function

expensive than the delta function, the delta function is preferable.

4.4 Effect of computation parcel size

Three simulation cases are set up with the same configuration except that different computational parcel sizes, $N_c/N_p = 0.1, 1, 10$ are used. The case with $N_c/N_p = 0.1$ is actually breaking a physical bubble into 10 computational parcels.

Figure 4.6 shows the evolution of the average concentration for the three cases respectively. It can be seen that the results from the three cases are pretty close. However, the average concentration tends to reduce faster with the decrease of computation parcel size. The reason is illustrated by figure 4.7, in which the evolution the total bubble number inside the computation domain is plotted. The bubble injection

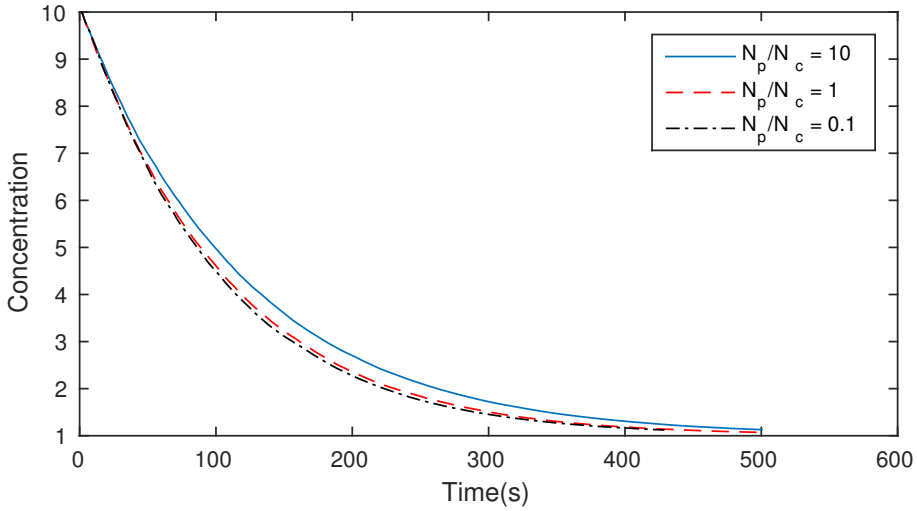

Figure 4.6: Evolution of average concentration number for $N_p/N_c = 0.1, 1, 10$

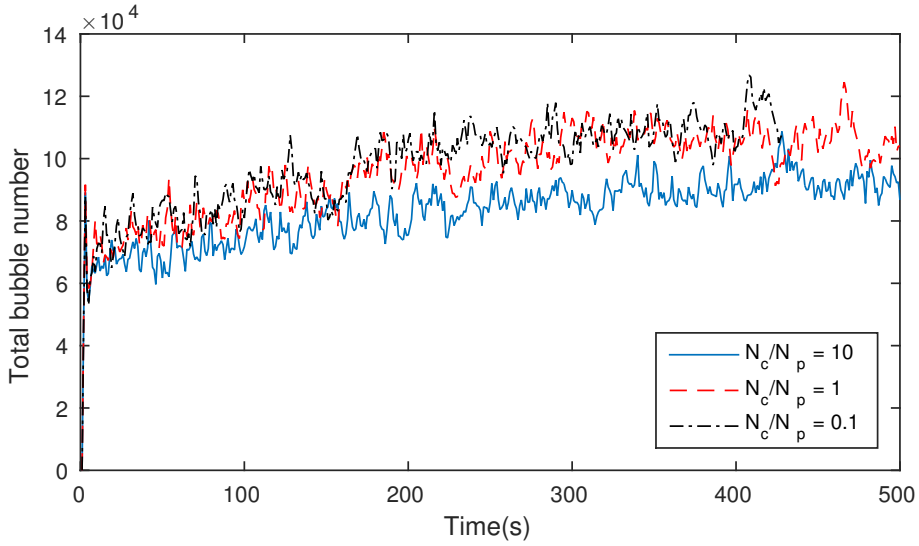

Figure 4.7: Evolution of total bubble number for $N_p/N_c = 0.1, 1, 10$

rate for the 3 cases are exactly the same, yet the induced flow by the injection of bubbles in the three cases are different, which in turn changes the bubble residence time. This effect accounts for the difference in the total bubble number. Figure 4.7 shows that the increase of the computation parcel size tends to reduce the bubble residence time. This is closely related to the induced flow pattern, which will be discussed in detail later.

4.5 Comparison between L-E and E-E models

Lagrangian-Eulerian and Eulerian-Eulerian models are different methods aiming to solve the same problem. The comparison between these two methods not only helps to justify the validity of each method, but it also helps find what impact the approximation each method makes will have.

4.5.1 Comparison without flow

The result of simulations without flow for both models are shown in figure 4.8.

Excellent match between the two models is find when the liquid remains quiescent all the time.

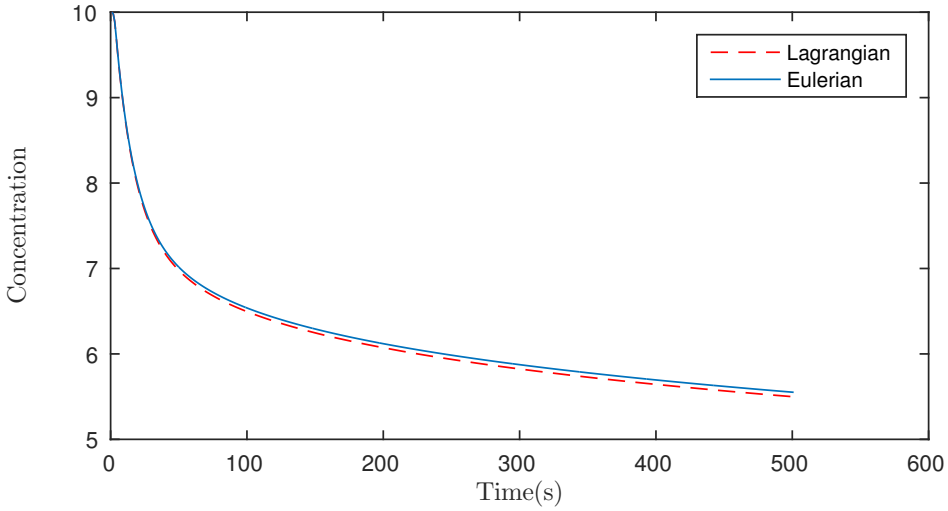

Figure 4.8: Evolution of average concentration for L-E and E-E model without flow

4.5.2 Comparison with flow

In the following simulations, the liquid is allowed to flow. Figure 4.9 shows that although the match between the two models is still pretty good, the difference is much bigger than the previous comparison, where the liquid remains quiescent. So the difference must come from the difference in the flow fields or more precisely, the difference in flow fields resulting from the difference between these two models in governing equations and numerical implementation.

Recall the difference between the two models discussed in section 2.4, one can conclude that the most obvious difference is that there are the two extra forcing terms in the L-E model mixture momentum equation. To see the effect of these two terms, we drop them in a comparison simulation. In figure 4.10, the additional black dash-dot line shows the result from modified L-E model. It can be seen that even

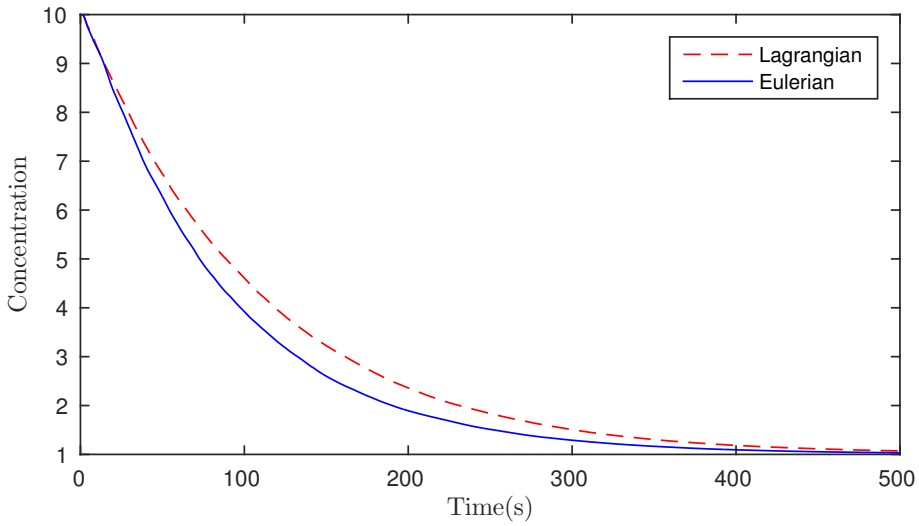

Figure 4.9: Evolution of average concentration for L-E and E-E models with flow

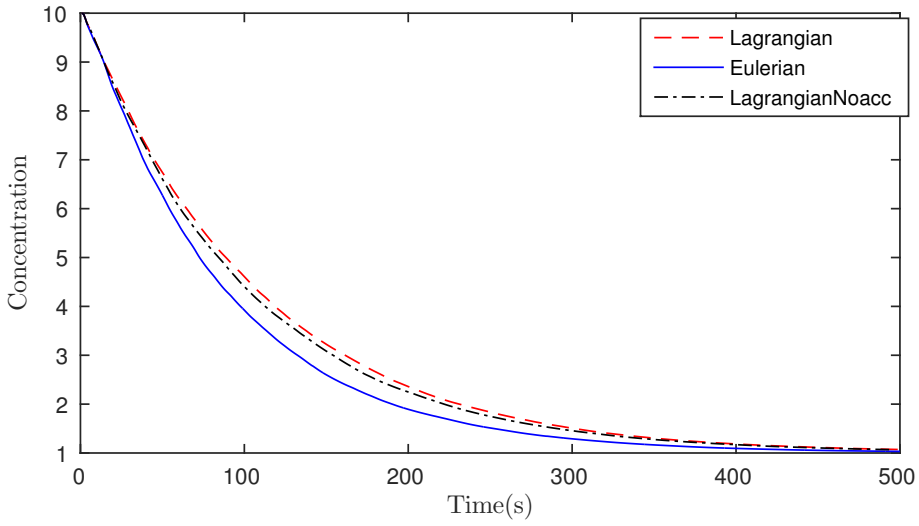

Figure 4.10: Evolution of average concentration for L-E, modified L-E, and E-E models with flow

though the result from the modified L-E model is closer to that of the E-E model, the improvement is very small, which indicates that the two extra forcing terms are unimportant. So it is legitimate to omit these two terms.

Since the difference comes from the difference in flow field, a deep look into the flow field in the two calculations will be very helpful. Figure 4.11 shows the z-component of velocity field on YOZ plane at actual time 1 s, 2 s, 3 s, and 4 s for L-E and E-E model. It is clear that the flow field in L-E model is very unstable and becomes chaotic very quickly after the injection of air bubbles. On the other hand, the flow field in E-E model remains symmetric and stable for a much longer time. There are two instability mechanism in this system. Firstly, the bubbly liquid below is lighter than liquid without bubble above, which can lead to Rayleigh-Taylor instability. Secondly, a strong shear flow is induced, which has the potential for a Kelvin-Helmholtz instability. In the L-E model, the bubble initial positions are generated by filtering numbers generated by a pseudo random number generator. However good the statistical properties of the filter and random number generator are, there will be an non-uniformity in the initial bubble position, which provides a strong perturbation to the system and makes it to be unstable. Whereas in the E-E model, the initial bubble positions are described by a density field, the value of which can be made to be perfectly symmetric. Thus the initial perturbation is only of the order of the round-off error, which is much smaller than that of L-E model.

This also explains the results we found in section 4.4. Given the same number of

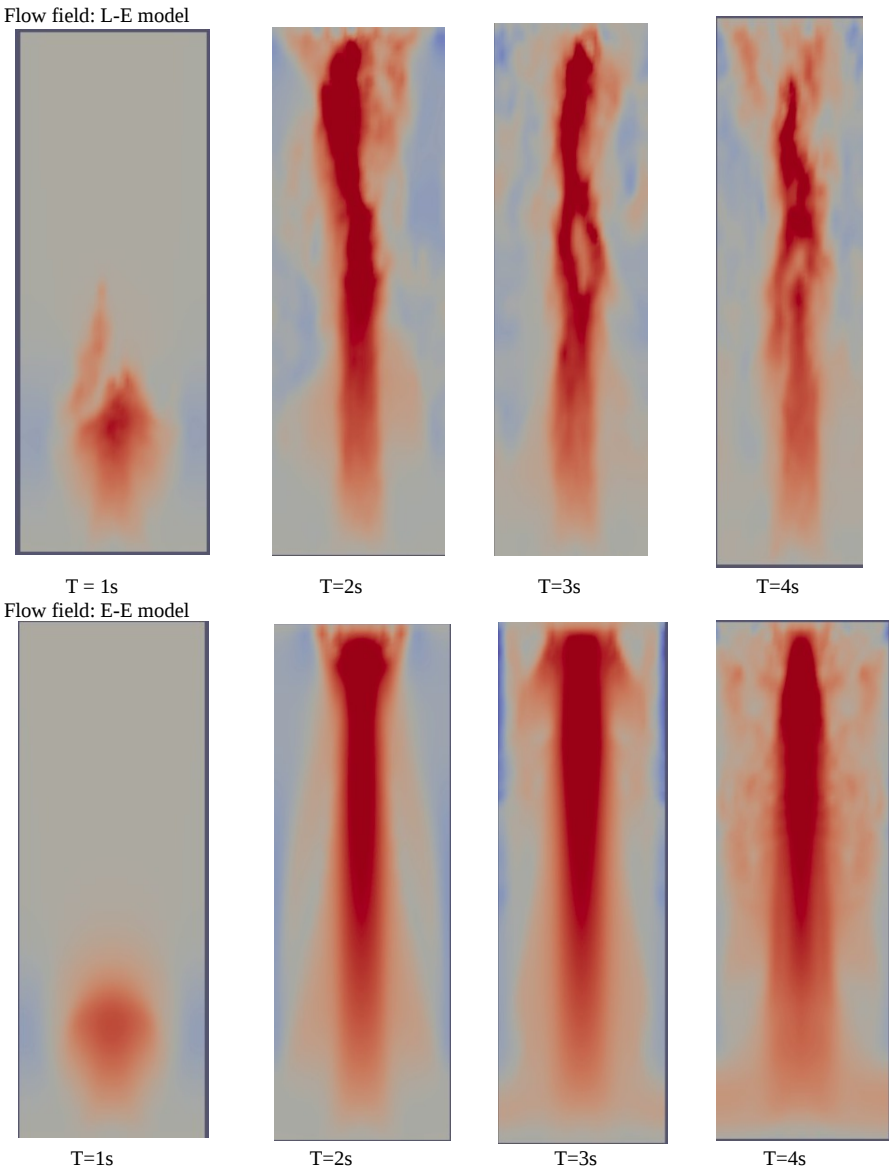

Figure 4.11: Flow field at time equals to 1 s, 2 s, 3 s, 4 s for L-E and E-E model. All the figures shows the value of vertical velocity at YOZ plane of the domain, which has a width of 4cm and a height of 10cm. The velocity range from -7cm/s to 15cm/s.

physical bubbles, the more computational parcels we have, the less the non-uniformity, and the longer the stability of the flow can be maintained.

References

[1] D. Weitkamp, "A review of dissolved gas supersaturation literature," *Transactions of the American Fisheries Society*, vol. 109, no. 6, pp. 659–702, November 1980.

[2] A. Prosperetti and X. Geng, "Study of the effectiveness of a laboratory scale system to reduce water super-saturation," Department of Mechnical Engineering, The Johns Hopkins University, Baltimore, MD, 21218, U.S.A, Tech. Rep., April 2006, to the U.S. Bureau of Reclamation, Water Treatment Engineering and Research.

[3] Z. Zhang and A. Prosperetti, "A second-order method for three-dimensional particle simulation," *Journal of Computational Physics*, vol. 210, no. 1, pp. 292–324, November 2005.

[4] A. Prosperetti and H. Ogũz, "Physalis: A new o(n) method for the numerical simulation of disperse systems: Potential flow of spheres," *Journal of Computational Physics*, vol. 167, no. 1, pp. 196–216, Feburary 2001.

REFERENCES

[5] A. Sierakowski. Bluebottle Wiki. [Online]. Available: http://lucan.me.jhu.edu/wiki/index.php/Bluebottle

[6] S. Chu. Original version of point particle implementation. [Online]. Available: https://github.com/ShiganChu/Bluebottle_pointParticle/tree/31e84dcb1fc6453f5acb095dce0bc1ccb9e73125

[7] S. Elghobashi and G. C. Truesdell, "Direct simulation of particle dispersion in a decaying isotropic turbulence," *Journal of Fluid Mechanics*, vol. 242, pp. 655–700, September 1992.

[8] J. K. Dukowicz, "A particle-fluid numerical model for liquid sprays," *Journal of Computational Physics*, vol. 35, no. 2, pp. 229–253, April 1980.

[9] S. Subramaniam, "Lagrangian-Eulerian methods for multiphase flows," *Progress in Energy and Combustion Science*, vol. 39, no. 2-3, pp. 215–245, April-June 2013.

[10] K. D. Squires and J. K. Eaton, "Measurements of particle dispersion obtained from direct numerical simulations of isotropic turbulence," *Journal of Fluid Mechanics*, vol. 226, pp. 1–35, May 1991.

[11] S. Subramaniam, "Statistical modeling of sprays using the droplet distribution function," *Physics of Fluids*, vol. 13, no. 3, pp. 624–642, March 2001.

[12] P. Oresta, F. Fornarelli, and A. Proeperetti, "Multiphase rayleigh-Bénard convection," *Mechanical Engineering Reviews*, vol. 1, no. 1, pp. 1–18, January 2014.

[13] D. Z. Zhang and A. Prosperetti, "Averaged equations for inviscid disperse two-phase flow," *Journal of Fluid Mechanics*, vol. 267, pp. 185–219, May 1994.

[14] ——, "Ensemble phase averaged equations for bubbly flows," *Physics of Fluids*, vol. 6, no. 9, pp. 2956–2970, September 1994.

[15] NVIDIA. CUDA zone. [Online]. Available: https://developer.nvidia.com/cuda-zone

[16] D. L. Brown, R. Cortez, and M. L. Minion, "Accurate projection methods for the incompressible navierstokes equations," *Journal of Computational Physics*, vol. 168, no. 2, pp. 464–499, April 2001.

[17] J. H. Ferziger and M. Peri, *Computational Methods for Fluid Dynamics*, 3rd ed. Berlin: Springer, 2002.

[18] J. Capecelatro and O. Desjardins, "An eulerlagrange strategy for simulating particle-laden flows," *Journal of Computational Physics*, vol. 238, no. 1, pp. 1–31, April 2013.

[19] D. A. Pope, "An exponential method of numerical integration of ordinary differential equations," *Communications of the ACM*, vol. 6, no. 8, pp. 491–493, August 1963.

[20] M. Harris, S. Sengupta, and J. D. Owens. Parallel prefix sum with cuda. [Online]. Available: http://http.developer.nvidia.com/GPUGems3/gpugems3_ch39.html

[21] L. Devroye, *NonUniform Random Variate Generation.* New York: Springer-Verlag, 1986.

[22] J. K. Dukowicz, "A particle-fluid numerical model for liquid sprays," *Journal of Computational Physics*, vol. 35, no. 2, pp. 229–253, April 1980.

[23] P. Pischke, D. Cordes, and R. Kneer, "The velocity decomposition method for second-order accuracy in stochastic parcel simulations," *International Journal of Multiphase Flow*, vol. 47, pp. 160–170, December 2012.

[24] Y. Zhang. Numerical code of the Eulerian-Eulerian model. [Online]. Available: https://github.com/yu-H-ang/bluebottle/tree/Eulerian/src

[25] ——, "Eulerian numerical simulation of bubble growth in super-saturated water," Master's thesis, Department of Mechanical Engineering, The Johns Hopkins University, Baltimore, MD, 21218, U.S.A, 2015.

Vita

Gedi Zhou received the B.S. degree in Mechanical Engineering from Peking University in 2013, and enrolled in the Mechanical Engineering M.S.E program at Johns Hopkins University in the same year. He won the Chalmers Design Award honoured by University of Toronto in 2013, and received a Research Assistantship from the department of Mechanical Engineering, Johns Hopkins University in 2015. Gedi began to work in the multiphase flow laboratory in Jan 2014, where he conducted cutting-edge researches concerning problems in multiphase flow systems.

Starting from October 2015, Gedi will work on the problem "Oscillating spherical particle in a shear flow bounded by wall", where he will help to get more insights about the motion of solid particles in liquid.