Journal of Computational Physics 228 (2009) 6938–6956

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Consistent projection methods for variable density incompressible Navier–Stokes equations with continuous surface forces on a rectangular collocated mesh

Ming-Jiu Ni *

College of Physical Sciences, Graduate University of Chinese Academy of Sciences, Beijing 100049, China

ARTICLE INFO

Article history:
Received 13 July 2008
Received in revised form 29 April 2009
Accepted 11 June 2009
Available online 17 June 2009

Keywords:
Consistent projection method
Divergence-free velocity
Variable density Navier–Stokes equations
Surface tension
Pressure gradient

ABSTRACT

Two consistent projection methods of second-order temporal and spatial accuracy have been developed on a rectangular collocated mesh for variable density Navier–Stokes equations with a continuous surface force. Instead of the original projection methods (denoted as algorithms I and II in this paper), in which the updated cell center velocity from the intermediate velocity and the pressure gradient is not guaranteed solenoidal, the consistent projection methods (denoted as algorithms III and IV) obtain the cell center velocity based on an interpolation from a conservative fluxes with velocity unit on surrounding cell faces. Dependent on treatment of the continuous surface force, the pressure gradient in algorithm III or the sum of the pressure gradient and the surface force in algorithm IV at a cell center is then conducted from the difference between the updated velocity and the intermediate velocity in a consistent projection method. A non-viscous 3D static drop with serials of density ratios is numerically simulated. Using the consistent projection methods, the spurious currents can be greatly reduced and the pressure jump across the interface can be accurately captured without oscillations. The developed consistent projection method are also applied for simulation of interface evolution of an initial ellipse driven by the surface tension and of an initial sphere bubble driven by the buoyancy with good accuracy and good resolution.

1. Introduction

Accurate modeling of multi-fluid flows with surface tension, which are frequently encountered in engineering application, is a challenging work because of the interface deformation, the discontinuity in material properties across the interface and the interfacial boundary conditions due to surface forces. Methods for interface advection in multi-fluid flows include but not limited to the volume of fluid (VOF) [9,21,41], level set [22,31], immersed boundary method [24], front tracking [36,37], CIP [33,40], phase field [10]. The interfacial methods using fixed Eulerian grids solve incompressible variable density Navier–Stokes equation with surface tension as source terms. Special treatments are needed to deal with the balance between the pressure gradient and the surface tension. Local imbalances between the pressure gradient and the surface tension leads to spurious (parasitic) currents [12], which do not disappear with mesh refinement. When the surface tension is dominant, these spurious currents may destroy the interface and cause disruptive instability. For simulations of multi-fluid flows using VOF method, this imbalance may be induced either by poor discretization of the pressure gradient and the surface

* Tel.: +86 010 8825 6474.
E-mail addresses: mjni@gucas.ac.cn, mjniatcas@gmail.com.
URL: http://www.gucas.ac.cn.

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.06.004

tension or by inaccurate calculation of the curvature rate [6]. An accurate calculation of curvature rate can be yielded using coupled level set and VOF method [32], least-square method for the reconstruction of surface tension [27], an estimator function tuned with a least-square-fit against the reference data [14], height function technique [5,13,28], convolution technique, which is not the research issue of this paper. The modeling of the pressure gradient and the surface tension will be discussed in this paper.

An accurate implementation of surface tension force is a key to reduce the spurious currents. Popinet and Zaleski [25] reduced the spurious currents considerably by improving the pressure gradient calculation using a pressure gradient correction procedure in their front-tracking method. In the PROST algorithm, Renardy and Renardy [27] noted that a balance between the two nonzero terms of the surface tension and the pressure gradient terms must be maintained numerically at equilibrium. The pressure is divided into two parts p_1 and p_2, in which p_1 and p_2 are chosen to ensure $\boldsymbol{S} - \nabla p_1$ ($\boldsymbol{S}$ denotes surface tension) and the discrete velocity divergence-free, respectively. This idea of the PROST was employed by Tong and Wang [34] for their simulation of capillarity-dominant free-surface flow with PBM (pressure boundary method). Shirani et al. [30] developed a cheap method for the calculation of the pressure on the interface location (PCIL). The method is directly derived by applying the momentum balance on each interface cell. Torres and Brackbill [35] employed a curl projection method for incompressible flows to reduce the spurious currents with an unconnected front-tracking method. Jamet et al. [11] used a second-gradient method to reduce the truncation error in the computation of the energy exchanges between the surface and the kinetic energies. By ensuring energy conservation, the parasitic currents were reduced drastically. Shin et al. [29] present a simple and effective improvement in suppressing spurious currents to a minimal level by a hybrid formulation for the calculation of the surface tension force in the context of a front-tracking interface method. To maintain the balance between the pressure gradient and the surface tension, a balanced-force projection method was designed by Francois et al. [6] on a collocated mesh, in which both the surface tension and the pressure gradient were directly estimated at cell faces and the values at cell centers are interpolated from the values on surrounding cell faces. A simple interpolation formula has been given in the paper. The balanced-force algorithm [6] calculates the pressure term and the surface tension together at every step of a projection method using the same technique to discretize the gradients of the pressure and the volume fraction. This algorithm has been proven very effective in the balance of the pressure gradient and the surface force on a collocated mesh.

For incompressible flows, $\nabla \cdot \boldsymbol{u} = 0$ is an important constraint condition to construct a conservative scheme. On a collocated mesh, the divergence of velocity at a cell center calculated from the fluxes on the surrounding cell faces are guaranteed divergence-free after a convergent solution of the pressure Poisson equation is used to project the intermediate velocity on a divergence-free vector space. However, the calculated cell center velocity based on the intermediate velocity and the pressure gradient is not guaranteed solenoidal. Motivated by the successful application of a consistent and conservative scheme for the simulation of liquid metal flows under a strong magnetic field [18,19], in which the conservative current fluxes on surrounding cell faces are interpolated to obtain the current density at a cell center, a consistent projection method on a collocated mesh is designed to enforce the solenoidal cell center velocity field. The velocity at a cell center will be yielded through interpolation of conservative fluxes on the surrounding cell faces. Depending on treatment of the surface tension, two consistent projection methods calculate the pressure gradient or the sum of the pressure gradient and the surface tension at a cell center from the difference between the solenoidal cell center velocity and the non-solenoidal intermediate velocity, respectively. In the consistent projection methods, the pressure are divided into two parts. The first part is chosen to obtain a second-order conservative velocity fluxes on cell faces and the second-part is chosen to yield second-order solenoidal discrete cell center velocities. To validate the consistent projection methods, a three-dimensional non-viscous static drop with surface tension is simulated. The interface is represented by the zero level set function. Detailed comparisons between the original projection methods and the corresponding consistent projection methods are conducted. The numerical results will show that the consistent projection methods predict the velocity field with dramatically reduced spurious currents and the accurate pressure jump without oscillations. The consistent projection methods are also successfully applied for simulation of interface evolutions of an initial ellipse driven by surface tension and of an initial sphere bubble driven by buoyancy.

2. Numerical methods for variable density incompressible Navier–Stokes equations with surface tension

2.1. Governing equations

In the level set method, a single set of mass and momentum conservation equations is solved on a fixed Eulerian grid and the level set function ϕ with $\phi = 0$ represented as interface is evolved with an advection equation. The flow is assumed to be incompressible. The governing equations are the momentum conservative equation:

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left(\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right) - \frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{1}{\rho}S_i \tag{1}$$

mass conservative equation:

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{2}$$

and the advection equation of the level set function:

$$\frac{\partial \phi}{\partial t} + u_j \frac{\partial \phi}{\partial x_j} = 0 \tag{3}$$

where u_i is the velocity at the x_i direction of a Cartesian coordinate, p the total pressure, S_i the body force at the x_i direction, which may include gravitational acceleration or surface tension. ρ and μ are the fluid density and viscosity, respectively, which are defined using the Heaviside function $h(\phi)$ as:

$$\rho = h(\phi)\rho_1 + (1 - h(\phi))\rho_2 \tag{4}$$

$$\mu = h(\phi)\mu_1 + (1 - h(\phi))\mu_2 \tag{5}$$

Here the subscripts 1 and 2 denote fluid 1 and fluid 2, respectively. Using the CSF (continuous surface force) model [2] based on the level set function, the surface tension at x_i direction can be expressed as:

$$S_i = \sigma\kappa(\phi)\delta(\phi)\frac{\partial \phi}{\partial x_i} \tag{6}$$

where σ is the surface tension coefficient, κ is the interfacial curvature, δ is the Dirac function.

2.2. Review of algorithms for incompressible Navier–Stokes equations

A key issue in the design of numerical methods for incompressible flows is the development of an appropriate discrete form of the incompressibility constraint of Eq. (2). Famous primitive variable numerical methods include the explicit MAC method [7], implicit and/or semi-implicit projection methods [4] and the SIMPLE method [23]. They all have been extensively used and have served well. Implicit or semi-implicit methods are attractive as a means of avoiding restrictions on the explicit time step. Both the projection methods and the SIMPLE type methods have been successfully applied to unsteady flows and steady flows [1,3,23,26,38,39]. The standard SIMPLE method has a second-order temporal accuracy for unsteady flows [16] and a bridge between the projection method and the SIMPLE method is built up [17], where general three-step and four-step projection methods of second-order temporal accuracy are developed for incompressible flows without body force included. The general second-order projection methods are extended to incompressible Navier–Stokes equation of MHD (magnetohydrodynamics) with the Lorentz force included as a body force [18,19], in which a consistent and conservative scheme is designed to obtain a solenoidal current density at a cell center from an interpolation of conservative current density fluxes on the surrounding cell faces. Based on a four-step projection method [3], which is one case of the general four-step projection method in [17] with $\theta = 1$, and on a momentum interpolation [42] to overcome the pressure checkboard phenomena on a collocated mesh, algorithms for variable density incompressible Navier–Stokes equations with the surface tension included can be reviewed here.

A simple nomenclature is necessary for detailed discussion of the algorithms. As illustrated in Fig. 1, the subscripts c and f denote the cell center and the cell faces of the control volume c, respectively. f_i denotes cell faces at x_i direction of the control volume c with f_i^+ the cell face at positive x_i direction and f_i^- the cell face at negative x_i direction. The center of the neighbor control volumes of f_i is denoted as c_i with c_i^+ the neighbor of f_i^+ and c_i^- the neighbor of f_i^-. The coordinates of the cell center and its neighbor centers are $(x_j)_c, (x_j)_{c_i^+}$ and $(x_j)_{c_i^-}$, respectively. The coordinates of the cell faces are $(x_j)_{f_i^+}, (x_j)_{f_i^-}$, respectively. The superscript n denotes the time level. The subscripts i, j represent 1, 2, 3, which denote the three Cartesian coordinate directions, respectively. In this paper, the subscripts f_i and c_i do not do Einstein sum with the variables in the equations.

Depending on treatments of the source term of the surface tension in the Navier–Stokes equations, two algorithms on a collocated mesh are listed here from time level n to time level $n+1$ with titles of algorithms I and II, respectively. In the

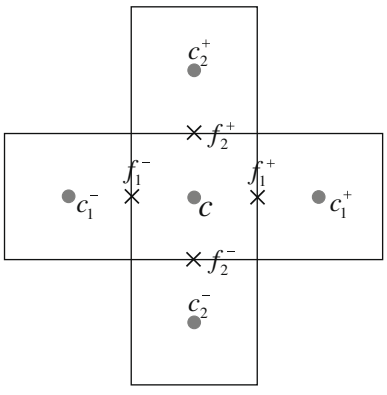


Fig. 1. Illustration of computation cell.

algorithm I, the source will be calculated only in the first predictor step. In the algorithm II, the source will be calculated together with the pressure gradient in every step of a projection method.

2.2.1. Algorithms I – with source term only needed in the predictor step

In this algorithm, the surface tension is only calculated in the first predictor step. With the level set function ϕ_c^n known, the density ρ_c^n and viscosity μ_c^n at cell centers can be yielded from Eqs. (4) and (5). The pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ and the surface tension $\left(\frac{1}{\rho}S_i\right)_c^n$ at time level it n have been calculated based on known p_c^n and ϕ_c^n. The detailed computational procedures at the Cartesian coordinate can be given here as:

(1) By solving the interface evolving Eq. (3), the level set function ϕ_c^{n+1} can be yielded based on ϕ_c^n and $(u_i)_c^n$. And the density ρ_c^{n+1} and viscosity μ_c^{n+1} at the time level $n+1$ can be obtained from Eqs. (4) and (5).

(2) The first predictor velocity $(u_i)_c^*$ at cell center can be obtained based on the velocity $(u_i)_c^n$, the pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ and the surface tension $(\frac{1}{\rho}S_i)_c^n$ at time level n from:

$$\frac{(u_i)_c^* - (u_i)_c^n}{\Delta t} = -\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n + \left(\frac{1}{\rho}S_i\right)_c^n - \left(u_j\frac{\partial u_i}{\partial x_j}\right)_c^{n+\frac{1}{2}} + \left(\frac{1}{\rho}\frac{\partial}{\partial x_j}\left(\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\right)\right)_c^{n+\frac{1}{2}} \tag{7}$$

To ensure second-order temporal accuracy, the diffusion term can be updated using the Crank–Nicholson scheme for the sake of stability:

$$\left(\frac{1}{\rho}\frac{\partial \tau_{ij}}{\partial x_j}\right)_c^{n+\frac{1}{2}} = \frac{1}{2}\left(\left(\frac{1}{\rho}\frac{\partial \tau_{ij}}{\partial x_j}\right)_c^n + \left(\frac{1}{\rho}\frac{\partial \tau_{ij}}{\partial x_j}\right)_c^*\right) \tag{8}$$

with $\tau_{ij} = \frac{\partial}{\partial x_j}\left(\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\right)$, and the convective term can be updated using the explicit Runge–Kutta technique or the following Adams–Bashforth technique for the sake of simplicity:

$$\left(u_j\frac{\partial u_i}{\partial x_j}\right)_c^{n+\frac{1}{2}} = \frac{3}{2}\left(u_j\frac{\partial u_i}{\partial x_j}\right)_c^n - \frac{1}{2}\left(u_j\frac{\partial u_i}{\partial x_j}\right)_c^{n-1} \tag{9}$$

(3) The second predictor velocity $(u_i)_c^{**}$ at cell center can be obtained based on the first predictor velocity $(u_i)_c^*$ and the pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ from:

$$(u_i)_c^{**} = (u_i)_c^* + \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n \tag{10}$$

The second predictor velocity will be interpolated to obtain the intermediate velocity flux $(u_i)_{f_i}^{**}$ on the corresponding cell face f_i, and an interpolation formula will be given later.

(4) The pressure p_c^{n+1} at time level $n+1$ can be acquired by solving the pressure Poisson equation based on the second predictor velocity at cell faces and material properties at time level $n+1$:

$$\left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i}^{n+1}\right)_c = \frac{1}{\Delta t}\left(\frac{\partial (u_i)_{f_i}^{**}}{\partial x_i}\right)_c \tag{11}$$

The half-discretization formula of the Poisson equation can be given as:

$$\sum_{i=1}^{3}\frac{\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^+}^{n+1} - \left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^-}^{n+1}}{(x_i)_{f_i^+} - (x_i)_{f_i^-}} = \frac{1}{\Delta t}\sum_{i=1}^{3}\frac{(u_i)_{f_i^+}^{**} - (u_i)_{f_i^-}^{**}}{(x_i)_{f_i^+} - (x_i)_{f_i^-}} \tag{12}$$

The pressure Poisson equation is solved using multigrid technique and ADI technique with underrelaxation method to improve the numerical stability and convergence. A stability analysis of discretized convection–diffusion equation has been conducted in [20].

(5) The normal velocity at the cell face f_i can be updated to time level $n+1$ as:

$$(u_i)_{f_i}^{n+1} = (u_i)_{f_i}^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i}^{n+1} \tag{13}$$

(6) The velocity vector at a cell center can be updated by:

$$(u_i)_c^{n+1} = (u_i)_c^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^{n+1} \tag{14}$$

This is the computational procedure from time level n to time level $n+1$ for the algorithm I. Set $n+1$ to n, and go back to the step (2) for the next time level.

As it has been explained, the subscript f_i denote cell faces at x_i direction of a control volume with f_i^+ and f_i^- the cell faces at positive and negative x_i directions respectively. The Eq. (13) can be written as:

$$(u_i)_{f_i^\pm}^{n+1} = \frac{1}{2}\left((u_i)_{c_i^\pm}^{**} + (u_i)_c^{**}\right) - \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^\pm}^{n+1} \tag{15}$$

where the following interpolation technique is employed to get the intermediate velocity fluxes on the corresponding cell faces:

$$(u_i)_{f_i^\pm}^{**} = \frac{1}{2}\left((u_i)_{c_i^\pm}^{**} + (u_i)_c^{**}\right) \tag{16}$$

2.2.2. Algorithm II – with source term needed in every step

In this algorithm, the surface tension is calculated in every step together with the pressure gradient. With the density ρ_c^n and viscosity μ_c^n known at cell center, the pressure gradient and surface tension at the time level n calculated at cell center, the detailed computational procedures at the Cartesian coordinate can be given here as:

(1) The density ρ_c^{n+1} and viscosity μ_c^{n+1} at cell center can be obtained using Eqs. (4) and (5) from ϕ_c^{n+1}, which is acquired by solving the advection Eq. (3) of the level set function based on ϕ_c^n and $(u_i)_c^n$. At the new time level, the surface tension $\left(\frac{1}{\rho}S_i\right)_c^{n+1}$ at a cell center and $(\frac{1}{\rho}S_i)_{f_i}^{n+1}$ on a cell face can be calculated based on ϕ_c^{n+1} and ρ_c^{n+1}, respectively, using techniques listed in Section 2.3.

(2) The first predictor velocity $(u_i)_c^*$ at cell center can be obtained based on the velocity $(u_i)_c^n$, the pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ and the surface tension $\left(\frac{1}{\rho}S_i\right)_c^n$ from

$$\frac{(u_i)_c^* - (u_i)_c^n}{\Delta t} = -\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n + \left(\frac{1}{\rho}S_i\right)_c^n - \left(u_j\frac{\partial u_i}{\partial x_j}\right)_c^{n+\frac{1}{2}} + \left(\frac{1}{\rho}\frac{\partial}{\partial x_j}\left(\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right)\right)_c^{n+\frac{1}{2}} \tag{17}$$

Again, the convective term can be updated using a second-order explicit technique for simplicity and the diffusion term can be updated using the semi-implicit Crank–Nicholson technique for stability.

(3) The second predictor velocity at cell center can be obtained based on the first predictor velocity $(u_i)_c^*$ and the pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ and the surface tension $\left(\frac{1}{\rho}S_i\right)_c^n$ from

$$(u_i)_c^{**} = (u_i)_c^* + \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n - \Delta t\left(\frac{1}{\rho}S_i\right)_c^n \tag{18}$$

The second predictor velocity will be interpolated to obtain the intermediate velocity $(u_i)_{f_i}^{**}$ on the corresponding cell faces f_i by using the interpolation of Eq. (16).

(4) The pressure p_c^{n+1} at time level $n+1$ can be acquired by solving the pressure Poisson equation based on the second predictor velocity and the calculated $n+1$ time level's surface tension on cell faces and material properties at time level $n+1$:

$$\left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i}^{n+1}\right)_c = \frac{1}{\Delta t}\left(\frac{\partial (u_i)_{f_i}^{**}}{\partial x_i}\right)_c + \left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}S_i\right)_{f_i}^{n+1}\right)_c \tag{19}$$

(5) The normal velocity at the cell face f_i can then be updated to time level $n+1$ from the calculated p_c^{n+1} as:

$$(u_i)_{f_i}^{n+1} = (u_i)_{f_i}^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i}^{n+1} + \Delta t\left(\frac{1}{\rho}S_i\right)_{f_i}^{n+1} \tag{20}$$

(6) The velocity vector at a cell center can be updated to time level $n+1$ as:

$$(u_i)_c^{n+1} = (u_i)_c^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^{n+1} + \Delta t\left(\frac{1}{\rho}S_i\right)_c^{n+1} \tag{21}$$

Algorithm I has been broadly applied for the simulation of incompressible Navier–Stokes equations with a source term included, such as for MHD in [18,19] and for multi-fludi flows in [12,26,31]. For multi-fluid flows, algorithm II is firstly developed by Francois et al. [6] to design a balanced-force method with the surface tension and the pressure gradient calculated together at the same cell position using the same technique for discretization of the pressure gradient and the gradient of volume fraction, which has been extended to an unstructured mesh [8]. In most of literatures, a three-step projection method is used, such as the algorithm II developed in [6]. Here, a four-step projection method is used to illustrate the two algorithms for variable density Navier–Stokes equations with a source term included. Ni and Abdou [17] has proven the equivalence of the three-step and the four-step projection methods for incompressible flows.

2.3. Discretization of pressure gradient and surface tension

2.3.1. Pressure gradient and surface tension at cell face

On a collocated mesh, the pressure gradient $\frac{\partial p}{\partial x_i}$ at the corresponding cell face $f_i^{\pm}$ can be discretized to overcome the pressure checkboard phenomena by:

$$\left(\frac{\partial p}{\partial x_i}\right)_{f_i^{\pm}} = \frac{p_{c_i^{\pm}} - p_c}{(x_i)_{c_i^{\pm}} - (x_i)_c} \tag{22}$$

And the pressure gradient with density included of $\frac{1}{\rho}\frac{\partial p}{\partial x_i}$ at cell faces $f_i^{\pm}$ can be calculated by:

$$\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^{\pm}} = \left(\frac{1}{\rho}\right)_{f_i^{\pm}}\left(\frac{\partial p}{\partial x_i}\right)_{f_i^{\pm}} = \left(\frac{1}{\rho}\right)_{f_i^{\pm}}\frac{p_{c_i^{\pm}} - p_c}{(x_i)_{c_i^{\pm}} - (x_i)_c} \tag{23}$$

This is the only technique used in this paper to calculate the normal pressure gradient at a cell face.

Also the surface tension S_i at the corresponding cell face f_i is calculated using:

$$(S_i)_{f_i^{\pm}} = \sigma\kappa_{f_i^{\pm}}\delta(\phi_{f_i^{\pm}})\left(\frac{\partial \phi}{\partial x_i}\right)_{f_i^{\pm}} \tag{24}$$

where the normal gradient of the level set function is conducted using:

$$\left(\frac{\partial \phi}{\partial x_i}\right)_{f_i^{\pm}} = \frac{\phi_{c_i^{\pm}} - \phi_c}{(x_i)_{c_i^{\pm}} - (x_i)_c} \tag{25}$$

Again, the normal surface tension with density included can be discretized by:

$$\left(\frac{1}{\rho}S_i\right)_{f_i^{\pm}} = \sigma\kappa_{f_i^{\pm}}\delta(\phi_{f_i^{\pm}})\left(\frac{1}{\rho}\right)_{f_i^{\pm}}\left(\frac{\partial \phi}{\partial x_i}\right)_{f_i^{\pm}} \tag{26}$$

This is the only technique used in this paper to estimate the surface tension at a cell face.

The density at a cell face is needed in the discretization of the pressure Poisson equation and in the calculation of the pressure gradient and the surface tension with density included, which can be interpolated from the densities at the two neighbor cell centers of the cell face. A simple linear average and a reciprocal average are employed and evaluated in the paper, respectively, as:

$$(\rho)_{f_i^{\pm}} = \frac{1}{2}(\rho_c + \rho_{c_i^{\pm}}) \tag{27}$$

$$\left(\frac{1}{\rho}\right)_{f_i^{\pm}} = \frac{1}{2}\left(\frac{1}{\rho_c} + \frac{1}{\rho_{c_i^{\pm}}}\right) \tag{28}$$

2.3.2. Pressure gradient and surface tension at cell center

The pressure gradient and the surface force at cell center are required in the predictor and updated steps. The following two techniques will be employed for the evaluation of the pressure gradient and the surface tension at cell centers.

(1) Technique IThe pressure gradient at cell center is directly discretized using a second-order center-difference scheme as:

$$\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c = \left(\frac{1}{\rho_c}\right)\frac{p_{c_i^+} - p_{c_i^-}}{(x_i)_{c_i^+} - (x_i)_{c_i^-}} \tag{29}$$

and the surface tension is calculated using the following equation:

$$\left(\frac{1}{\rho}S_i\right)_c = \sigma\kappa_c\delta(\phi_c)\frac{1}{\rho_c}\left(\frac{\partial \phi}{\partial x_i}\right)_c = \sigma\kappa_c\delta(\phi_c)\frac{1}{\rho_c}\frac{\phi_{c_i^+} - \phi_{c_i^-}}{(x_i)_{c_i^+} - (x_i)_{c_i^-}} \tag{30}$$

(2) Technique IIWith the pressure gradient at cell face calculated using (23) and the surface tension at cell face calculated using Eq. (26), the pressure gradient and the surface tension at a cell center can be interpolated as:

$$\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c = \frac{1}{2}\left(\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^+} + \left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^-}\right) \tag{31}$$

$$\left(\frac{1}{\rho}S_i\right)_c = \frac{1}{2}\left(\left(\frac{1}{\rho}S_i\right)_{f_i^+} + \left(\frac{1}{\rho}S_i\right)_{f_i^-}\right) \tag{32}$$

2.4. Development of consistent projection methods

For the algorithms I and II, the updated velocity at a cell center based on the non-solenoidal intermediate velocity and the pressure gradient (sum of the pressure gradient and the surface tension for algorithm II) is not divergence-free. From our numerical practice, which will be illustrated in Section 3, these two algorithms will produce big spurious currents and pressure oscillations at a high density ratio. Motivated by our successful application of a consistent and conservative scheme for calculation of current density at a cell center from a conservative interpolation of current fluxes on the surrounding cell faces, two consistent projection methods of second-order temporal and spatial accuracy will be developed here. In the consistent projection methods, the velocity at a cell center is not directly updated from the intermediate velocity and the pressure gradient at a cell center, but interpolated from conservative fluxes on the surrounding cell faces.

2.4.1. Algorithm III – a consistent projection methods based on Algorithm I

Without loss of generality, the algorithm I is taken as an example to develop a consistent projection method. In the algorithm I, the divergence of velocity at a cell center based on the fluxes updated by Eq. (13) is free. However, the velocity vector at a cell center updated by Eq. (14) is not guaranteed solenoidal. The vector at a cell center can be obtained by a simple central interpolation from its corresponding fluxes on the surrounding cell faces of the control volume, which conserves the charge in the simulation of MHD [18]. Instead of Eq. (14), a simple central interpolation can be employed here to obtain the velocity vector at a cell center as:

$$(u_i)_c^{n+1} = \frac{1}{2}\left((u_i)_{f_i^+}^{n+1} + (u_i)_{f_i^-}^{n+1}\right) \tag{33}$$

Comparing with the update of velocity vector using Eq. (14) in algorithm I, this interpolation can guarantee a solenoidal cell center velocity vector. However, this interpolation introduce a second-order dissipation. With $(u_i)_{f_i^+}^{n+1}$ and $(u_i)_{f_i^-}^{n+1}$ updated by Eqs. (13) and $(u_i)_{f_i}^{**}$ interpolated by Eq. (16) and (33) can be reformulated as:

$$(u_i)_c^{n+1} = \frac{(u_i)_{c_i^+}^{**} + 2(u_i)_c^{**} + (u_i)_{c_i^-}^{**}}{4} - \frac{\Delta t}{2}\left(\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^+}^{n+1} + \left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_{f_i^-}^{n+1}\right) \tag{34}$$

Suppose the pressure gradient at a cell center is acquired using the technique II in Section 2.3, the following formula can be deduced:

$$(u_i)_c^{n+1} = (u_i)_c^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^{n+1} + \frac{1}{4}\left((u_i)_{c_i^+}^{**} - 2(u_i)_c^{**} + (u_i)_{c_i^-}^{**}\right) = ((u_i)_c^{n+1})^{AI} + \frac{1}{4}\left(\frac{\partial^2 u_i^{**}}{\partial x^2}\right)_c (\triangle x)^2 \tag{35}$$

where $((u_i)_c^{n+1})^{AI}$ represents the velocity vector at a cell center obtained by Eq. (14) from algorithm I. The velocity vector from a simple interpolation of Eq. (33) is a second-order spatially accurate approximation of $((u_i)_c^{n+1})^{AI}$, but it is equivalent to an second-order dissipation term added in the Navier–Stokes equation. In algorithm III, to eliminate this effect, a new flux is constructed based on the second predictor velocity as:

$$(g_i)_{f_i^+}^{**} = -\frac{1}{8}\left(\left(\frac{\partial^2 u_i^{**}}{\partial x^2}\right)_c + \left(\frac{\partial^2 u_i^{**}}{\partial x^2}\right)_{c_i^+}\right)(\triangle x)^2 \tag{36}$$

A divergence at a cell center based on this new flux is apparently not free. A gradient of a scalar p_2 is introduced in the algorithm III to ensure the updated velocity fluxes conservative. Algorithm III is a consistent projection method based on the algorithm I, in which the surface tension is only calculated in the first predictor step. With the level set function ϕ_c^n known, the density ρ_c^n and viscosity μ_c^n at cell centers can be yielded from Eqs. (4) and (5). The pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c$ and the surface tension $\left(\frac{1}{\rho}S_i\right)_c^n$ at time level n have been calculated. The detailed computational procedures of the algorithm III can be given here as:

(1) By solving the interface evolving Eq. (3), the level set function ϕ_c^{n+1} can be yielded based on ϕ_c^n and $(u_i)_c^n$. And the density ρ_c^{n+1} and viscosity μ_c^{n+1} at the time level $n+1$ can be obtained from Eqs. (4) and (5).
(2) The first predictor velocity $(u_i)_c^*$ at cell center can be obtained based on the velocity $(u_i)_c^n$, the pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ and the surface tension $\left(\frac{1}{\rho}S_i\right)_c^n$ at time level n from Eq. (7).
(3) The second predictor velocity at cell center can be obtained based on the first predictor velocity $(u_i)_c^*$ and the pressure gradient $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n$ from Eq. (10). The intermediate velocity flux $(u_i)_{f_i}^{**}$ on the cell face f_i is interpolated from its corresponding velocity at neighbor centers by Eq. (16).

(4) The pressure $(p_1)_c^{n+1}$ at time level $n+1$ can be acquired by solving the following Poisson equation:

$$\left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}\frac{\partial p_1}{\partial x_i}\right)^{n+1}_{f_i}\right)_c = \frac{1}{\Delta t}\left(\frac{\partial (u_i)^{**}_{f_i}}{\partial x_i}\right)_c \tag{37}$$

(5) The normal velocity at cell face f_i can be updated to time level $n+1$ as:

$$(u_i)^{n+1}_{f_i} = (u_i)^{**}_{f_i} - \Delta t\left(\frac{1}{\rho}\frac{\partial p_1}{\partial x_i}\right)^{n+1}_{f_i} \tag{38}$$

(6) A new flux of $(g_i)^{**}_{f_i}$ is calculated from Eq. (36) to remove the numerical dissipation induced by a simple central interpolation from the velocity fluxes. A scalar p_2 is introduced to ensure the divergence from updated flux of $(g_i)^{n+1}_{f_i}$ free. The scalar p_2 is obtained by solving the following Poisson equation:

$$\left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}\frac{\partial p_2}{\partial x_i}\right)^{n+1}_{f_i}\right)_c = \frac{1}{\Delta t}\left(\frac{\partial (g_i)^{**}_{f_i}}{\partial x_i}\right)_c \tag{39}$$

(7) The gradient of scalar p_2 is used to update the flux of $(g_i)^{**}_{f_i}$, we then have:

$$(g_i)^{n+1}_{f_i^+} = (g_i)^{**}_{f_i^+} - \Delta t\left(\frac{1}{\rho}\frac{\partial p_2}{\partial x_i}\right)^{n+1}_{f_i^+} \tag{40}$$

A divergence at a cell center based on the updated fluxes of $(g_i)^{n+1}_{f_i^+}$ on the surrounding faces is free.

(8) In this algorithm, the velocity vector at a cell center is not directly updated from Eq. (14). The velocity flux $(u_i)^{n+1}_{f_i^\pm}$ and the updated new flux $(g_i)^{n+1}_{f_i^\pm}$ are then used to obtain the velocity vector at a cell center by the following interpolation:

$$(u_i)^{n+1}_c = \frac{1}{2}\left((u_i)^{n+1}_{f_i^+} + (u_i)^{n+1}_{f_i^-}\right) + \frac{1}{2}\left((g_i)^{n+1}_{f_i^+} + (g_i)^{n+1}_{f_i^-}\right) \tag{41}$$

This interpolation reduces the numerical dissipation due to a simple interpolation of Eq. (33) from the velocity fluxes on cell faces. This interpolation can guarantee the velocity vector solenoidal comparing with the algorithm I.

(9) Instead of the discretization of the pressure gradient using the technique I and/or the technique II in Section 2.3, the pressure gradient at a cell center can be calculated through the following equation:

$$\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)^{n+1}_c = \frac{1}{\Delta t}\left((u_i)^{**}_c - (u_i)^{n+1}_c\right) \tag{42}$$

which is called as the technique III of the algorithm III for the pressure gradient in this paper. In algorithm III, the surface tension is calculated from the technique II in Section 2.3.

This is the computational procedure from time level n to time level $n+1$ for the algorithm III of a consistent projection method. Set $n+1$ to n, and go back to the step (1) for the next time level.

Algorithm III divides the pressure into two parts p_1 and p_2. The gradient of p_1 will ensure the divergence at a cell center based on the velocity fluxes $(u_i)^{n+1}_{f_i}$ on the surrounding cell faces free. The gradient of p_2 will ensure divergence from the new fluxes of $(g_i)^{n+1}_{f_i}$ free, which is used to remove the numerical dissipation introduce by a simple central interpolation from the velocity fluxes. The velocity vector at a cell center is then interpolated from the sum of fluxes of $(u_i)^{n+1}_{f_i} + (g_i)^{n+1}_{f_i}$, which is solenoidal comparing with the original projection method of algorithm I. The pressure gradient at a cell center is calculated from the difference between the updated velocity vector and the second predictor velocity vector, which is called technique III of the algorithm III in this paper.

2.4.2. Algorithm IV – A consistent projection methods based on Algorithm II

Based on algorithm II, a simple central interpolation of Eq. (33) can be employed to obtain the velocity vector at a cell center, which will introduce a numerical dissipation as:

$$(u_i)^{n+1}_c = ((u_i)^{n+1}_c)^{\mathrm{AII}} + \frac{1}{4}\left(\frac{\partial^2 u^{**}}{\partial x^2}\right)_c (\triangle x)^2 \tag{43}$$

where $((u_i)^{n+1}_c)^{\mathrm{AII}}$ represents the cell center velocity vector calculated from the algorithm II. To remove the effect of the numerical dissipation, a consistent projection method of algorithm IV can be developed by introducing a new flux based on the algorithm II. In this algorithm, the source term is calculated in every step together with the pressure gradient. With the density ρ^n_c and viscosity μ^n_c at cell center known, the sum of the pressure gradient and the surface tension at cell center of $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)^n_c - \left(\frac{1}{\rho}S_i\right)^n_c$ calculated at the time level n, the detailed computational procedures at the Cartesian coordinate can be given here as:

(1) The density ρ_c^{n+1} and viscosity μ_c^{n+1} at cell center can be obtained using Eqs. (4) and (5) from ϕ_c^{n+1}, which is acquired by solving the advection Eq. (3) of the level set function based on ϕ_c^n and $(u_i)_c^n$. At the new time level, the surface tension $\left(\frac{1}{\rho}S_i\right)_{f_i}^{n+1}$ on a cell face can be calculated using techniques listed in Section 2.3. The surface tension $\left(\frac{1}{\rho}S_i\right)_c^{n+1}$ at a cell center is not calculated separately.

(2) The first predictor velocity $(u_i)_c^*$ at cell center can be obtained based on the velocity $(u_i)_c^n$, the sum of the pressure gradient and the surface tension $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n - \left(\frac{1}{\rho}S_i\right)_c^n$ from Eq. (17).

(3) The second predictor velocity at cell center can be obtained based on the first predictor velocity $(u_i)_c^*$ and the sum of the pressure gradient and the surface tension $\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^n - \left(\frac{1}{\rho}S_i\right)_c^n$ from Eq. (18). And the second predictor velocity will be interpolated to obtain the intermediate normal velocity $(u_i)_{f_i}^{**}$ on the cell faces f_i by using the interpolation of Eq. (16).

(4) The pressure $(p_1)_c^{n+1}$ at time level $n+1$ can be acquired by solving the pressure Poisson equation based on the second predictor velocity and the calculated surface tension on cell faces and material properties at time level $n+1$:

$$\left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}\frac{\partial p_1}{\partial x_i}\right)_{f_i}^{n+1}\right)_c = \frac{1}{\Delta t}\left(\frac{\partial (u_i)_{f_i}^{**}}{\partial x_i}\right)_c + \left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}S_i\right)_{f_i}^{n+1}\right)_c \tag{44}$$

(5) The normal velocity at the cell face f_i can then be updated to time level $n+1$ from the calculated $(p_1)_c^{n+1}$ as:

$$(u_i)_{f_i}^{n+1} = (u_i)_{f_i}^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p_1}{\partial x_i}\right)_{f_i}^{n+1} + \Delta t\left(\frac{1}{\rho}S_i\right)_{f_i}^{n+1} \tag{45}$$

(6) A new flux of $(g_i)_{f_i}^{**}$ is calculated from Eq. (36) to remove the numerical dissipation induced by a simple central interpolation from the velocity fluxes. A scalar of p_2 is introduced to ensure the divergence from updated flux of $(g_i)_{f_i}^{n+1}$ free. The scalar p_2 is obtained by solving the following Poisson equation:

$$\left(\frac{\partial}{\partial x_i}\left(\frac{1}{\rho}\frac{\partial p_2}{\partial x_i}\right)_{f_i}^{n+1}\right)_c = \frac{1}{\Delta t}\left(\frac{\partial (g_i)_{f_i}^{**}}{\partial x_i}\right)_c \tag{46}$$

(7) The gradient of scalar p_2 is used to update the flux of $(g_i)_{f_i}^{**}$, we then have:

$$(g_i)_{f_i^+}^{n+1} = (g_i)_{f_i^+}^{**} - \Delta t\left(\frac{1}{\rho}\frac{\partial p_2}{\partial x_i}\right)_{f_i^+}^{n+1} \tag{47}$$

A divergence at a cell center based on the updated fluxes of $(g_i)_{f_i^+}^{n+1}$ on the surrounding faces is free.

(8) In this algorithm, the velocity vector at a cell center is not directly updated from Eq. (21). The velocity flux $(u_i)_{f_i^\pm}^{n+1}$ and the updated new flux $(g_i)_{f_i^\pm}^{n+1}$ are then used to obtain the velocity vector at a cell center by the following interpolation:

$$(u_i)_c^{n+1} = \frac{1}{2}\left((u_i)_{f_i^+}^{n+1} + (u_i)_{f_i^-}^{n+1}\right) + \frac{1}{2}\left((g_i)_{f_i^+}^{n+1} + (g_i)_{f_i^-}^{n+1}\right) \tag{48}$$

This interpolation removes the numerical dissipation in Eq. (43) due to a simple interpolation of Eq. (33) from the velocity fluxes on cell faces. This interpolation can guarantee the velocity vector solenoidal comparing with the algorithm I.

(9) Instead of the discretization of the pressure gradient and the surface tension using the technique I and/or the technique II in Section 2.3, the sum of the pressure gradient and the surface tension at a cell center can be calculated through the following equation:

$$\left(\frac{1}{\rho}\frac{\partial p}{\partial x_i}\right)_c^{n+1} - \left(\frac{1}{\rho}S_i\right)_c^{n+1} = \frac{1}{\Delta t}\left((u_i)_c^{**} - (u_i)_c^{n+1}\right) \tag{49}$$

which is called as the technique III of the algorithm IV for the calculation of the sum of the pressure gradient and the surface tension.

This is the computational procedure from time level n to time level $n+1$ for the consistent projection method of algorithm IV. Set $n+1$ to n, and go back to the step (1) for the next time level. Algorithm IV divides the pressure into two parts p_1 and p_2. The gradient of p_1 will ensure the divergence at a cell center based on the velocity fluxes $(u_i)_{f_i}^{n+1}$ on the surrounding cell faces free. The gradient of p_2 will ensure divergence from the new fluxes of $(g_i)_{f_i}^{n+1}$ free, which is used to remove the numerical dissipation introduce by a simple central interpolation from the velocity fluxes. The cell center velocity vector is then calculated through an interpolation of the sum of fluxes $(u_i)_{f_i}^{n+1} + (g_i)_{f_i}^{n+1}$. The sum of the pressure gradient and the surface tension at a cell center is calculated from the difference between the updated velocity vector and the second predictor velocity vector.

To remove the dissipation introduced by a simple interpolation of Eq. (33), a new flux of Eq. (36) is constructed in algorithms III and IV. Here, we introduce another flux to remove the numerical dissipation as:

$$(g_i)^{**}_{f_i^+} = -\frac{1}{8}\frac{1}{\rho_{f_i^+}}\left(\rho_c\left(\frac{\partial^2 u_i^{**}}{\partial x^2}\right)_c + \rho_{c_i^+}\left(\frac{\partial^2 u_i^{**}}{\partial x^2}\right)_{c_i^+}\right)(\triangle x)^2 \tag{50}$$

which is called as reconstruction of the new flux based on density interpolation in comparison with the reconstruction based on a simple interpolation of Eq. (36).

3. Validation of the consistent projection method

3.1. Spherical droplet without gravity

The algorithms have been tested on a temporal evolution of a 3D droplet. A static spherical droplet with a radius $R = 0.5$ in the absence of gravity is positioned at the center of a domain of $4R \times 4R \times 4R$ cube. The center of the sphere is coordinated as $(0,0,0)$. Theoretically, the velocity field should remain zero throughout the simulation. Numerically, the spurious currents were found in many interfacial methods. In the present simulation, the surface tension coefficient σ is taken to be 0.1, the background fluid density density $\rho_1 = 1$, and density inside the drop ρ_2 is varied from 0.001 to 1000. The exact jump in pressure across the droplet is $(\Delta p)_{exact} = \sigma\kappa$ and the exact curvature is given by $\kappa_{exact} = 2/R$ for a 3D case. The exact pressure

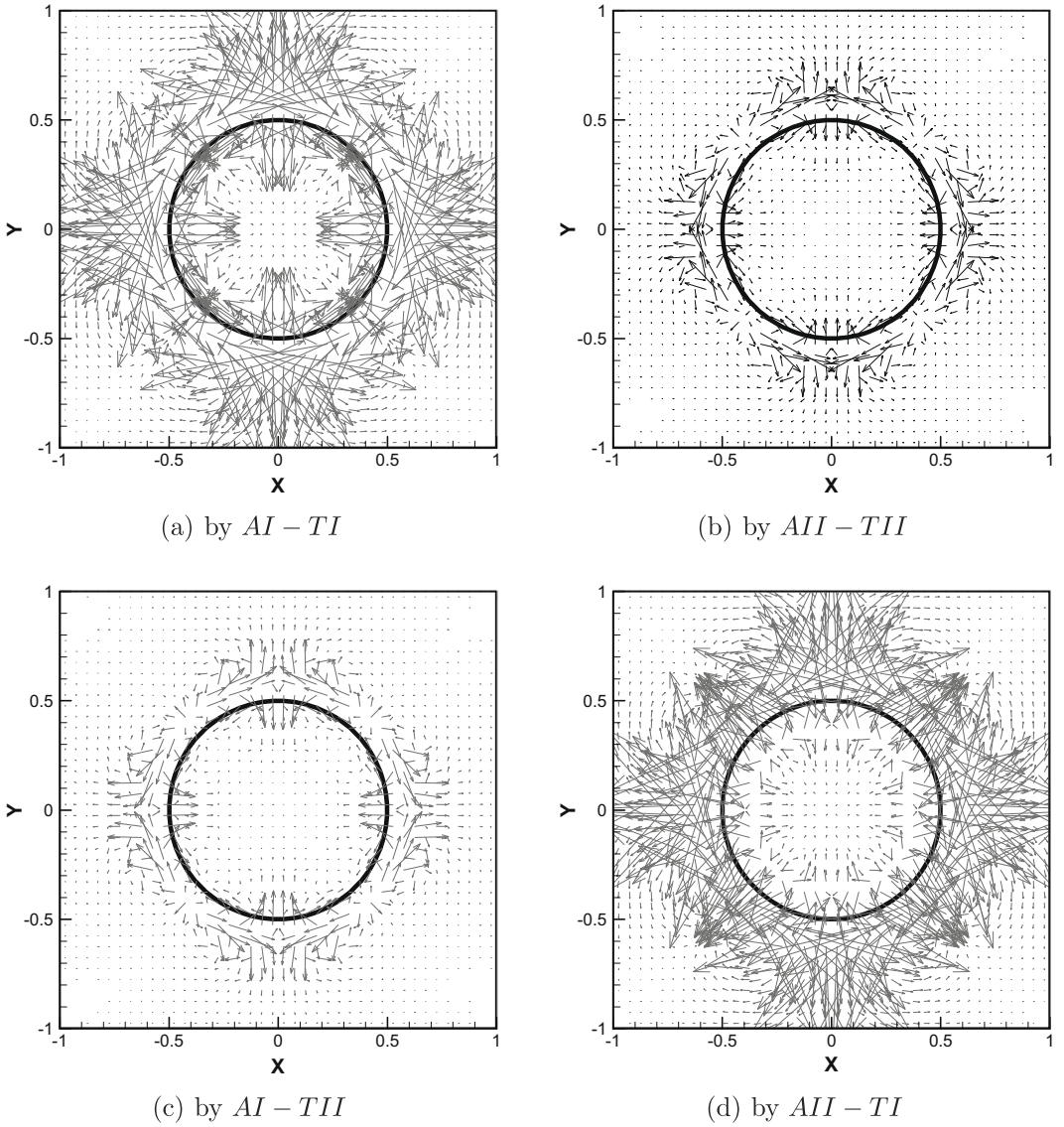


Fig. 2. Spurious current at time 0.1 by four models.

difference Δp is 0.4 in three dimensions. The velocity and pressure are initially set as zero. Velocity boundary conditions are non-slip wall condition. The computational grid is fixed, rectangular and uniform. The grid size and time step are held fixed.

The level set function is defined as $\phi = \sqrt{(x^2+y^2+z^2)} - 0.5$, and the interface curvature is calculated by $\kappa = \nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right)$. The interface in not evolved in the tested cases. In the comparison of the algorithms, the viscous term is ignored.

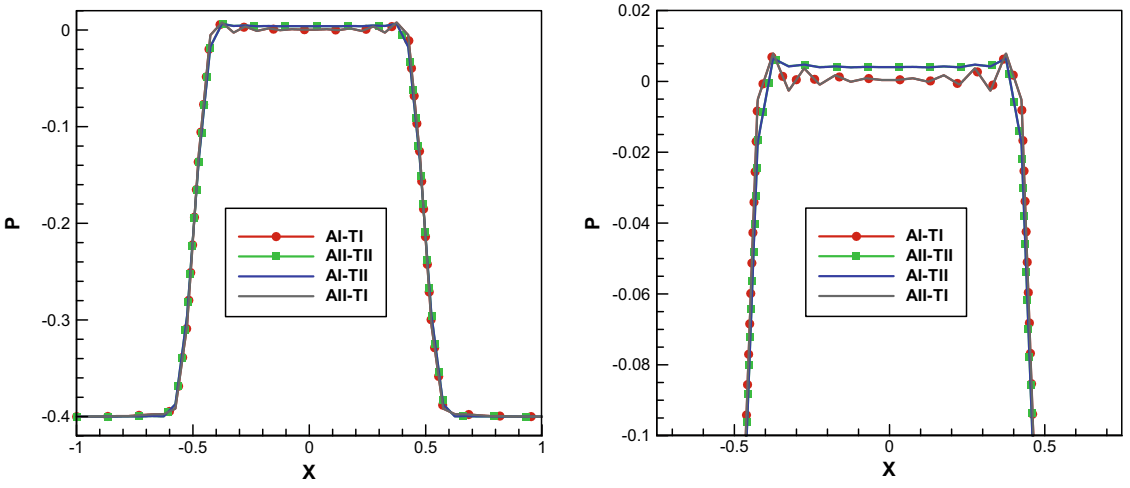
3.1.1. Comparison between Algorithms I and II

Using the algorithms I and II, the four models named AI–TI, AII–TII, AI–TII and AII–TI are firstly compared. AI and AII represent algorithms I and II for simulation of variable density incompressible Navier–Stokes equations, respectively. TI and TII represent technique I and technique II for calculation of the pressure gradient and surface tension at cell center, respectively. The detailed formulas have been given in Section 2.2 for the algorithms and in Section 2.3 for the techniques. AI–TI means the variable density Navier–Stokes equation is simulated using algorithm I, and the pressure gradient and the surface tension at a cell center are calculated using technique I, and so on. Most researchers employed the model AI–TI for the simulations of the multi-fluid flows. Francois et al. [6] developed the balanced-force model of AII–TII.

Using the reciprocal interpolation of Eq. (28) for the density at a cell face from its neighbor centers, the case of $\rho_2 = 10$ is simulated on a $40 \times 40 \times 40$ uniform collocated mesh with $\Delta t = 10^{-3}$. In this simulation, the errors in the pressure jump and in the velocity are investigated. The velocity vectors of time 0.1 at the plane $z = 0$ from models AI–TI, AII–TII, AI–TII and AII–TI are shown in Fig. 2(a)–(d), respectively. The velocities are all magnified 1000 times. Strong spurious currents can be seen in the vicinity of the free surface for model AI–TI, which is dramatically reduced by using the balanced-force model AII–TII, as it has been illustrated in [6]. The maximum velocity from model AI–TI is 7.189×10^{-3}, which is 5.282 times bigger than 1.361×10^{-3}, the maximum velocity from the model AII–TII.

More comparisons between algorithms I and II for incompressible Navier–Stokes equations with a source term are interesting. Ni et al. [18,19] conducted a comparison for incompressible flows with the Lorentz force. They prefer algorithm I for the MHD simulations. Francois et al. [6] developed the algorithm II and conducted a comparison for multi-fluid flows. They concluded algorithm II with the pressure gradient and the surface tension calculated together using the technique II is much better than algorithm I with the pressure gradient and the surface tension at a cell center using the technique I for incompressible flows with a continuous surface tension. The previous comparison between AI–TI and AII–TII conducted in this section also supports the conclusion. Here, one more comparison between the two algorithms is conducted for multi-fluid flows. For the algorithm I, when the technique II is employed for the discretization of the pressure gradient and the surface tension at a cell center, Fig. 2(c) clearly shows that the velocity by AI–TII is greatly reduced comparing with that from the model AI–TI. For the algorithm II, when the technique I is used for the discretization of the pressure and the surface tension at a cell center, Fig. 2(d) clearly shows the velocity from the model AII–TI is greatly increased comparing with that from the model AII–TII. This comparison illustrates the importance of the discretization of the pressure gradient and the surface tension at a cell center. More numerical results show the accuracy of algorithm I can be greatly improved if the technique II is employed for the discretization of the pressure gradient and the surface tension at a cell center.

With the pressure -0.4 fixed at $x = -0.95$, Fig. 3 illustrates the pressure distribution along the line of $z = 0, y = 0$ at time 0.1 for the case of drop density 10. Fig. 3(a) shows the pressure distribution along the whole line, where one can see that the predicted pressure jumps from the two models match well with the theoretical value of 0.4. Indeed, the relative pressure error is less than 0.1% from the model AII–TII. The pressure distribution inside the droplet is zoomed in Fig. 3(b). One can



(a) along the centerline (b) inside a droplet

Fig. 3. Computed pressure distribution at time 0.1 by four models.

clearly see the pressure oscillation near the free surface in the model AI–TI. This oscillation is greatly suppressed using the balanced-force algorithm AII–TII.

However, based on the reciprocal interpolation of the density at a cell face, when the density ratio is as big as 1000, there is a strong pressure oscillation at the vicinity of the bubble interface no matter which model is employed [15]. We therefore moved to the simple central interpolation for the density at a cell face by Eq. (27). Based on this simple interpolation, it is hard to get a convergent result for the models AI–TI and AII–TI at a large density ratio if the density at a cell center is directly applied for the discretization of the gradients using technique I. In the following calculations, only technique II is employed for the algorithms I and II. The comparison will be conducted among the models of AI–TII, AII–TII, AIII–TIII and AIV–TIII. Without further note and confusion, in the following description, this four models will be called as AI, AII, AIII and AIV, respectively.

3.1.2. Comparisons among four models

With the drop density 1000, the velocity vectors at time 0.1 from the models of AI, AII, AIII and AIV are illustrated in Fig. 4. The vectors are magnified 2000 times. Technique II is employed to calculate the pressure gradient and the surface tension in algorithms I and II, the velocity vector in Fig. 4(a) and (b) from the two models are almost same. Fig. 4(c) and (d) represent the results from the consistent projection methods of algorithms III and IV, respectively. The difference between the two figures is ignorable. However, one can clearly see that the spurious currents from the consistent projection methods are much

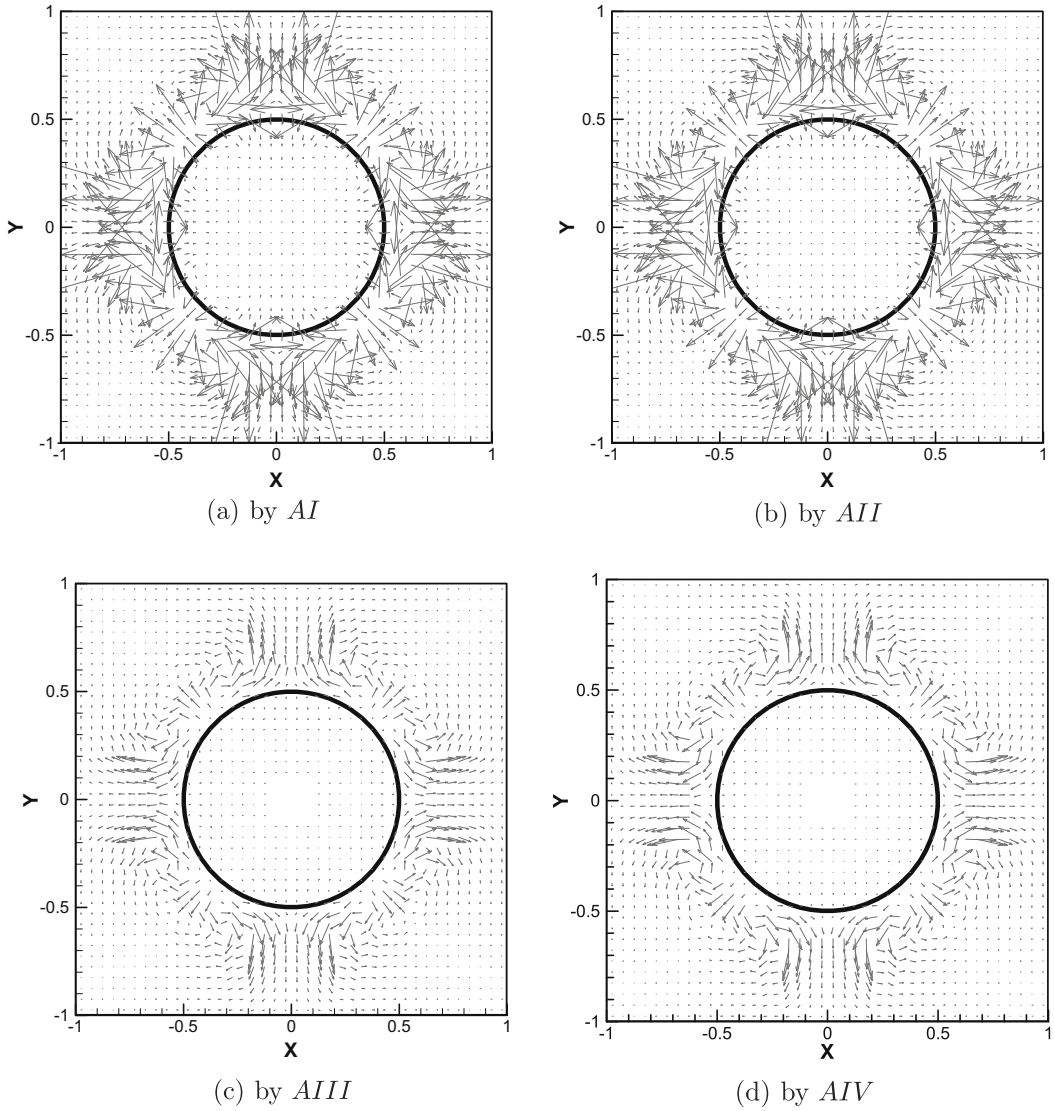


Fig. 4. Spurious current at time 0.1 by four models: 2000 times magnified.

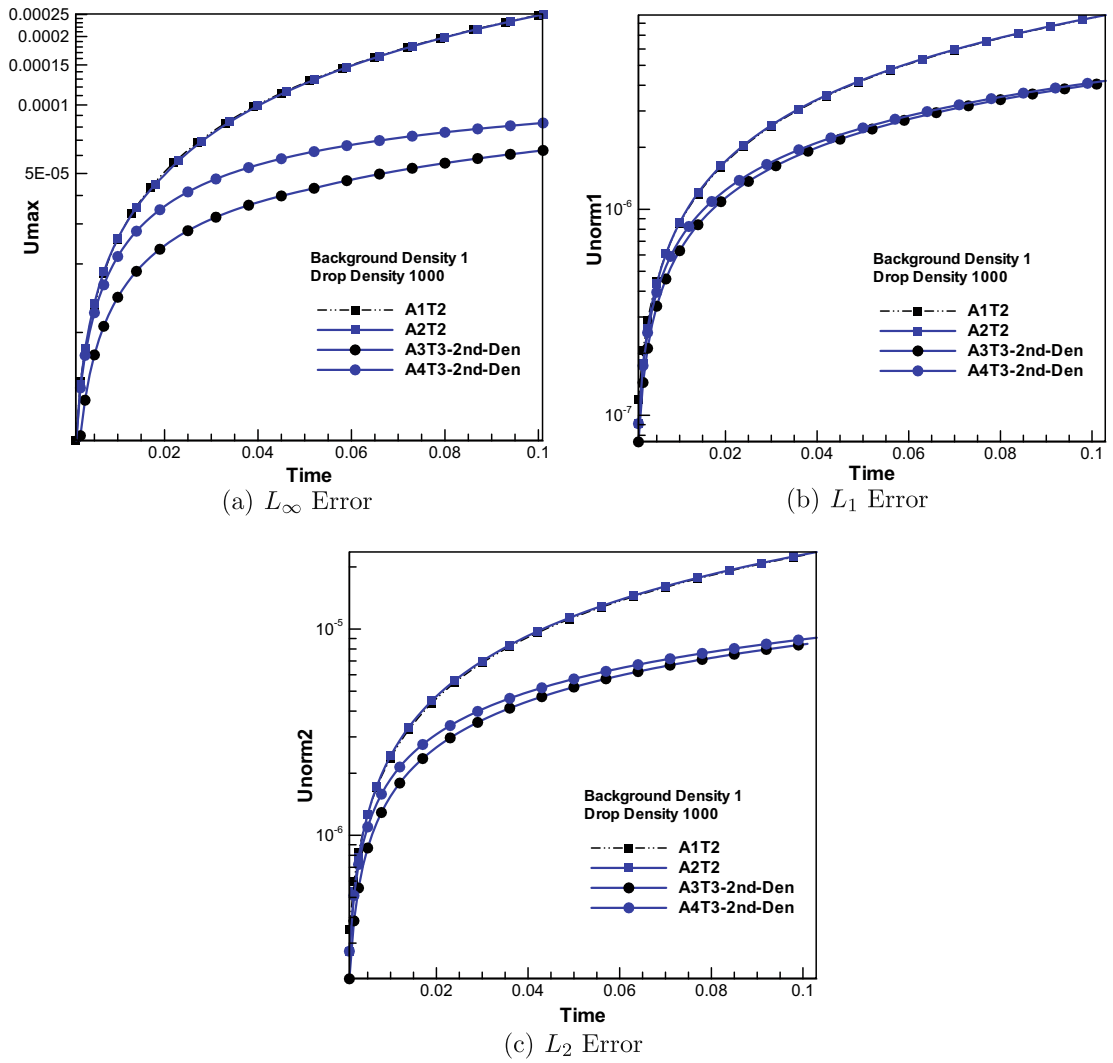


Fig. 5. Spurious velocity history using four models.

smaller than those from the original projection methods. And the velocity vectors from the consistent projection methods form closed loops, which show that the consistent projection method can guarantee the cell center velocity vector solenoidal comparing with the original projection methods. The L_∞, L_1 and L_2 errors of the spurious currents are shown in Fig. 5. From the three figures, one can see that the spurious currents from the algorithms I and II are much bigger than those from the algorithms III and IV. However, the difference between AI and AII, and the difference between AIII and AIV are very small and ignorable.

Fig. 6 illustrates the pressure distribution along the line of $z = 0,\ y = 0$, where the pressure at $x = -0.95$ is fixed as -0.4. In this figure, A1T2 and A2T2 represents the results from models AI and AII, respectively. A3T3 – 2nd and A4T3 – 2nd represents the models AIII and AIV, respectively, with the new flux constructed from Eq. (36). A3T3 – 2nd – Den and A4T3 – 2nd – Den represents the models AIII and AIV, respectively, with the new flux constructed from Eq. (47). A3T3 – 1st and A4T3 – 1st represents algorithms similar as the AIII and AIV, in which the new flux construction is not necessary and the velocity vectors are calculated through a simple center interpolation of velocity fluxes on surrounding cell faces from Eq. (33) with a second-order numerical dissipation as illustrated in Eqs. (35) and (40), respectively. Due to the numerical dissipation, A3T3 – 1st and A4T3 – 1st cannot accurately predict the pressure distribution. The pressures inside the bubble are far from the theoretical value of 0. All other models can reasonably predict the pressure distribution as shown in Fig. 6a. However, at the vicinity of the interface inside the bubble as shown in Fig. 6b, the pressures predicted by A1T2 and A2T2 oscillate. This oscillation is effectively restrained by the algorithms III and IV with new fluxes constructed. With the new flux constructed from Eq. (36), the resolution of A3T3 – 2nd and A4T3 – 2nd near the interface is not so good, the pressure is smeared a little too much at the vicinity of the interface with the distribution from A4T3 – 2nd a little better than from A3T3 – 2nd. With the new flux

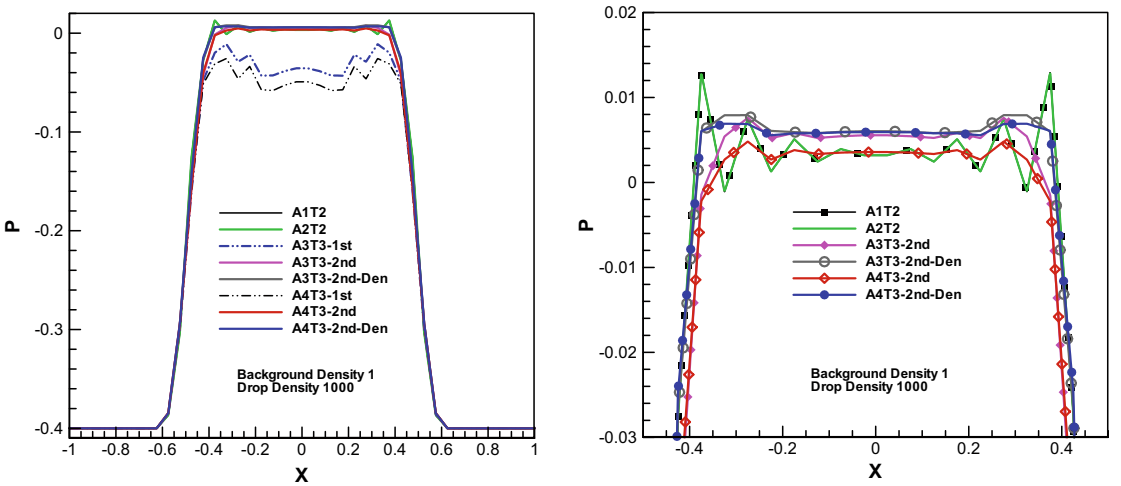


Fig. 6. Pressure distribution along a center line.

constructed from Eq. (40), the oscillations happened in AI and AII are almost completely restrained by the models A3T3 – 2nd – Den and A4T3 – 2nd – Den, although the value is a little bigger than the theoretical value.

With the drop density of 0.001, the pressure distributions are illustrated in Fig. 7 with pressure 0 fixed at the position of $x = 0$. A simple center interpolation of the velocity fluxes using Eq. (33) of A3T3 – 1st and A4T4 –1st cannot predict the pressure distribution accurately. All other models reasonably predict the pressure distribution. At the vicinity of the interface outside of the drop, construction of the new flux using Eq. (36) smears the pressure too much for the algorithm A3T3 – 2nd and A4T3 – 2nd, while the construction using Eq. (40) predicts the pressure with good resolution.

Using AI, AII, AIII and AIV, the histories of L_1 error of spurious currents for different density ratios are shown in Fig. 8. For large density ratio, there is no difference between AI and AII as illustrated in Fig. 8(a) for the drop density of 0.001 and in Fig. 5(b) for the drop density of 1000. However with small density ratio, the performance of AII is a little better than of AI as shown in Fig. 8(b) for the density of 10, in Fig. 8(c) for the drop density of 1 and in Fig. 8(d) for the density of 0.1. There is no big difference between AIII and AIV. The figure clearly illustrates that the performance of AIII and AIV is much better than the performance of AI and AII in reducing the spurious currents and in removing the pressure oscillation.

3.2. Three-dimensional droplet oscillation

Oscillating drops represents a standard and demanding test case for surface tension models. An initially non-spherical viscous drop with subsequent oscillating will decay to equilibrium static shape by the damping effect of viscosity. In the

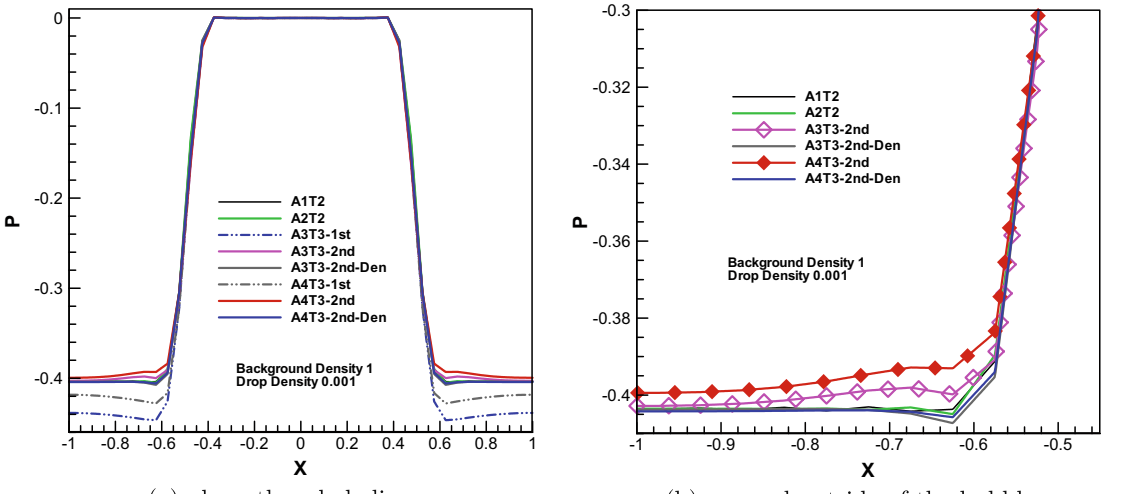
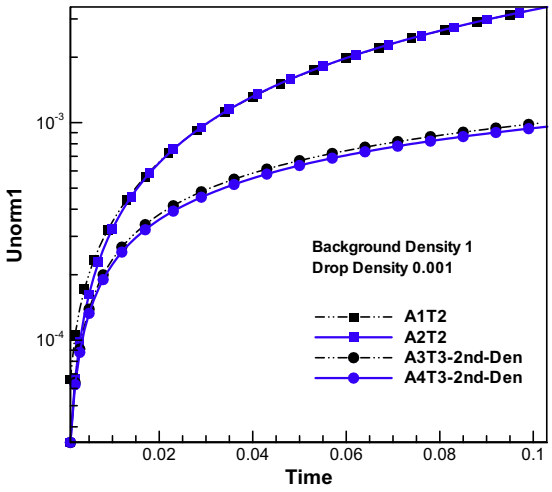
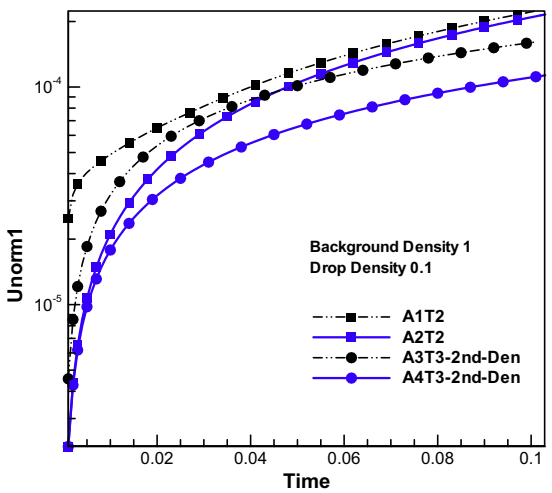


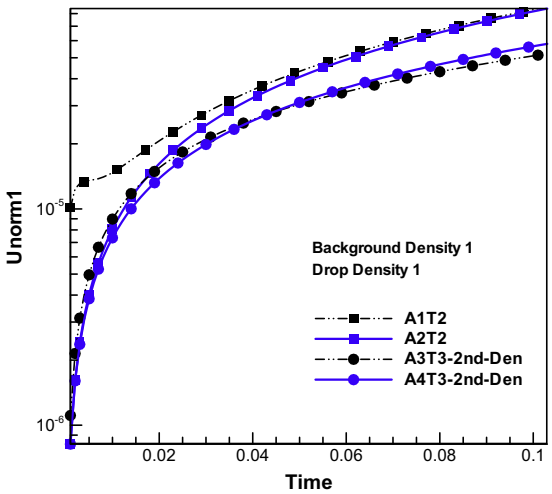
Fig. 7. Pressure distribution along a center line.



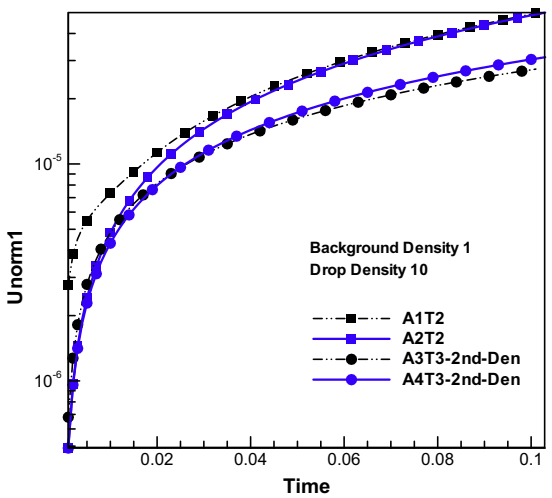
(a) Bubble Density 0.001



(b) Bubble Density 0.1



(c) Bubble Density 1



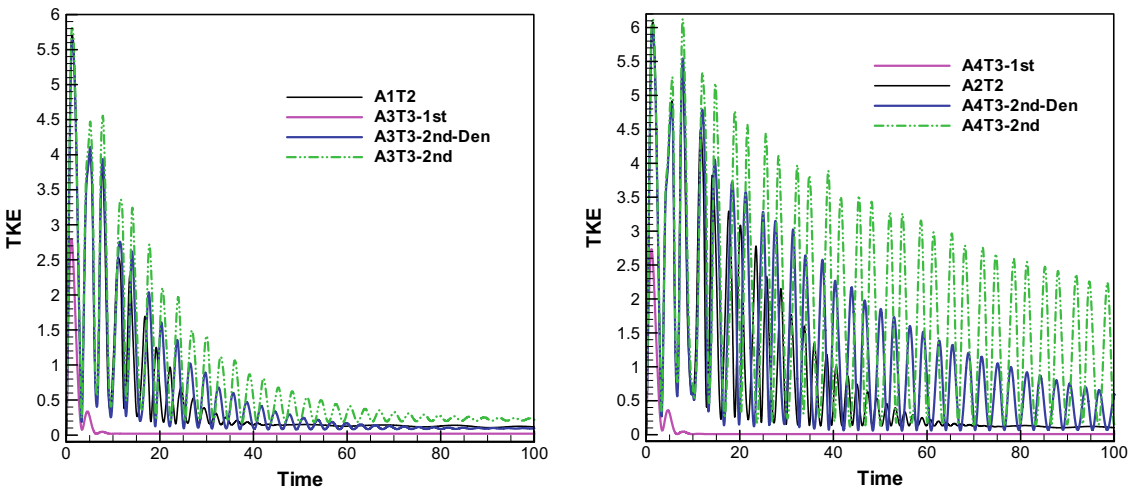
(d) Bubble Density 10

Fig. 8. L_1 error history of spurious velocity.

present study, an initial 3D ellipsoidal drop specified by equation $x^2/9 + y^2/4 + z^2 = 1$ is placed at the center of a $10 \times 10 \times 10$ computational domain in the absence of gravity. The densities (viscosities) inside and outside the elliptical sphere are 1.0 (0.01) and 0.01 (0.005), respectively. The surface tension coefficient $\sigma = 1$. A level set method is employed to capture the interface.

The kinetic energy, $\frac{1}{2}\int \rho \boldsymbol{u} \cdot \boldsymbol{u} dV$, versus time plots are shown in Fig. 9. Eight models are tested and compared in this test cases. In Fig. 9(a) and (b), all of the models can reach a static state. With a simple central interpolation of velocity fluxes using Eq. (33), the kinetic energy from A3T3 – 1st and A4T3 – 1st can reach a static state quickly. The models cannot capture the oscillation frequency. This is due to numerical dissipation terms added as shown in Eqs. (35) and (40), respectively. The kinetic energies from all other models gradually reach a static state. These models can reasonably predict the oscillation frequency with a little difference between A1T2 and A3T3, and between A2T2 and A4T3. The amplitudes from the consistent projection methods of A3T3 and A4T3 are bigger than the corresponding original projection methods of A1T2 and A2T2, respectively. The amplitudes from A2T2 and A4T3 are larger than those from A1T2 and A3T3, respectively. For consistent projection methods, such as A3T3 shown in Fig. 9(a) and A4T3 shown in Fig. 9(b), the amplitudes from A3T3 – 2nd – Den (A4T3 – 2nd – Den) with the new fluxes constructed from the density interpolation of Eq. (47) is less than the amplitudes from A3T3 – 2nd (A4T3 – 2nd) with the new fluxes constructed from the interpolation of Eq. (36). However, these models can predict the maximum amplitudes around the value of 6.

The spurious currents in static sphere drop predicted by the consistent projection methods of AIII and AIV are smaller than those from the original projection methods; while the kinetic energy in oscillating ellipsoidal drop predicted by the



(a) by algorithm I and algorithm III (b) by algorithm II and algorithm IV

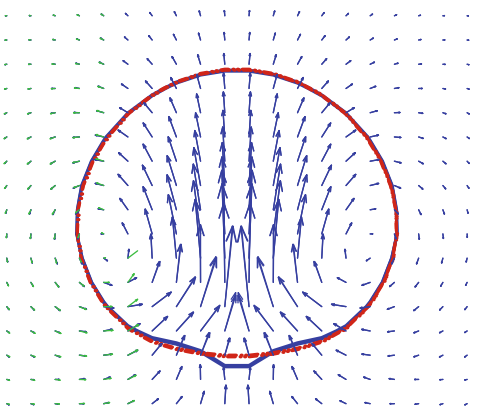
Fig. 9. Kinetic energy versus to time plot for a 3D oscillating drop.

consistent projection methods are larger than those from the original projection methods. We attribute it to the solenoidal cell center velocity predicted by the consistent projection methods, which play a great role in evolving the interface.

3.3. 3D bubble rise

The rise of an air bubble in water by buoyancy is simulated using the original algorithms of AI–TII and AII–TII and the consistent projection methods of AIII–TIII and AIV–TIII. The characteristic non-dimensional number for this situation is the Bond number: $Bo = \rho g R^2/\sigma$, where R is the bubble radius. The computational domain is chosen to be $[-2.5R, 2.5R] \times [0, 10R] \times [-2.5R, 2.5R]$ with $40 \times 80 \times 40$ mesh resolution. A bubble is initially placed at $(0, 1.5R, 0)$ with $R = 0.5$.

The densities and viscosities of air (water) are 1.226 (1000) and $1.137(1.78 \times 10^{-5})$, respectively. The downward-acting gravitational acceleration is 9.8. For the first case, the surface tension coefficient is 0.0728, which corresponds to a Bond number of $Bo = 41.3$. For the second case, the surface tension coefficient is increased to $\sigma = 72.8$, which corresponds to a smaller Bond numbers of $Bo = 0.0413$. Computed bubble shapes (measured by the $\phi = 0$ contour) are shown in Fig. 10 at



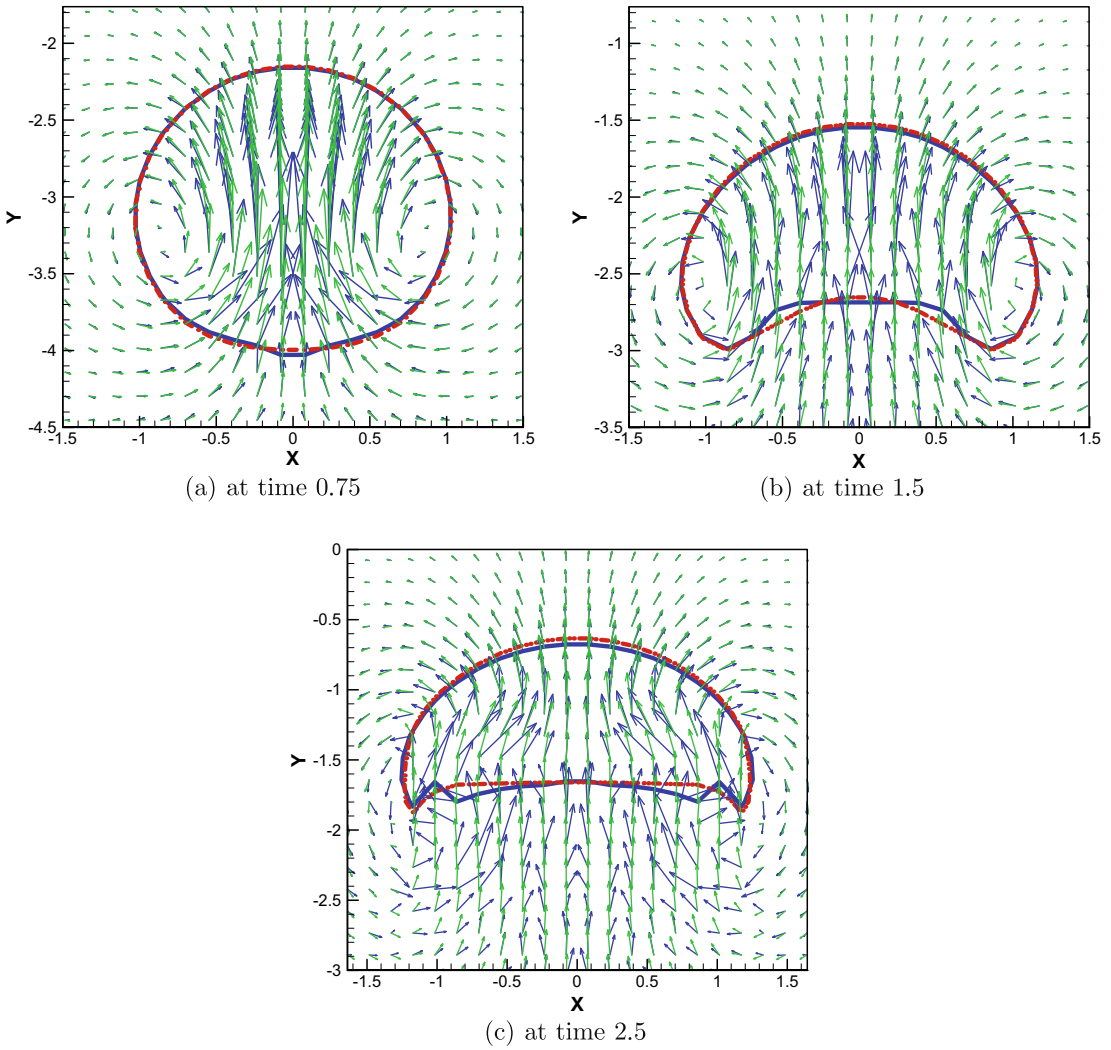


Fig. 11. Interface and velocity plot for 3D single bubble rise with Bond number 4.126×10^{-2}: blue dense lines and arrow lines denote interface and velocity vector from AII, red dashed dense line and green arrow lines denote interface and velocity vector from AIV. (For interpretation of the references in colour in this figure legend, the reader is referred to the web version of this article.)

non-dimensional times (characteristic length $L = R$ and characteristic velocity $U = \sqrt{gR}$) of $t = 0.75, 1.5$ for the case of $Bo = 41.3$ and Fig. 11 at times of $t = 0.75, 1.5, 2.5$ for the case of $Bo = 0.0413$, respectively.

For the case of $Bo = 41.3$, the shapes predicted by AI and AII are almost same and the shapes predicted by AIII and AIV are also almost same. Observable differences in the shapes from AII and AIV are evident. This difference is attributed to the cell center velocity distribution. Fig. 10(a) and (b) shows the velocity vector distributions. The cell center velocity from AII is apparently not solenoidal, and the velocity vectors at both sides of the center line normal to the bottom wall move toward the center line. The predicted cell center velocities from the consistent projection method form closed loops and are solenoidal comparing with the velocity from AII.

For the second case of $Bo = 0.0413$, one can observe less deformations in the bubble shapes since surface tension is more dominant relative to the first case. However, one can still observe a little difference in the shapes at time of $t = 0.75$ from Fig. 11(a). The difference becomes bigger as time marching shown in Fig. 11(b) and (c). Smaller and smoother deformations are observed with the consistent projection methods. Again the velocity vectors are shown in the figures, which illustrate the solenoidal cell center velocity plays a great role in the evolving of interface.

4. Summary

Consistent projection methods AIII–TIII and AIV–TIII on a collocated mesh are developed for the variable density Navier–Stokes equations with the surface tension, in which the pressure is divided into two parts of p_1 and p_2. The gradient of p_1 will

ensure the velocity fluxes conservative, which means the divergence at a cell center based on the velocity fluxes $(u_i)_{f_i}^{n+1}$ on the surrounding cell faces is free. The gradient of p_2 will ensure divergence from the new fluxes of $(g_i)_{f_i}^{n+1}$ free, which is used to remove the numerical dissipation introduce by a simple central interpolation from the velocity fluxes. The velocity vector at a cell center is then interpolated from the sum of fluxes of $(u_i)_{f_i}^{n+1} + (g_i)_{f_i}^{n+1}$, which is solenoidal comparing with the original projection method of algorithms I and II. The pressure gradient at a cell center is calculated from the difference between the updated velocity vector and the second predictor velocity vector. The consistent projection methods are second-order temporally and spatially accurate.

A comparison study among the consistent projection methods of AIII and AIV and their corresponding algorithms of the original projection method (AI), the balanced-force projection method (AII) is conducted on a rectangular collocated mesh. Comparison is also conducted among the techniques for the calculation of the pressure gradient and the surface force at a cell center. The numerical simulation of a three-dimensional non-viscous static drop without gravity force is done with density ratios from 0.001 to 1000. The numerical results clearly show that the consistent projection methods perform better in reducing the spurious currents and in restraining the pressure oscillation at the vicinity of the interface.

The consistent projection methods also perform well in predicting the oscillation frequency of an ellipsoidal drop driven by surface tension without gravity. The oscillation amplitudes from the consistent projection methods are bigger than those from the corresponding original projection methods. However, the predicted frequencies and the maximum oscillation amplitudes are close.

Finally, the rise of an air bubble in water with two Bond numbers of $Bo = 41.3$ and $Bo = 0.0413$ are conducted on a $40 \times 80 \times 40$ mesh. The consistent projection methods can reasonably predict the shape deformation, which is smoother and smaller comparing with the original projection methods. And the cell center velocity vectors from the consistent projection methods form closed loops, while the original projection methods cannot accurately predict a solenoidal velocity near the center bottom of a deformed bubble.

Acknowledgments

The author acknowledges the anonymous reviewers' comments, which greatly improve the quality of this paper. The author acknowledges the financial support from NSFC under Grant # 10872212 and the support from the Chinese Academy of Science under the name of "*BaiRenJiHua*".

References

[1] J.B. Bell, P. Colella, H.M. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, Journal of Computational Physics 85 (1989) 257–283.
[2] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, Journal of Computational Physics 100 (1992) 335–354.
[3] H. Choi, P. Moin, Effects of the computational time step on numerical solutions of turbulent flow, Journal of Computational Physics 113 (1994) 1–4.
[4] A.J. Chorin, Numerical solution of Navier–Stokes equations, Mathematics of Computation 22 (1968) 745–762.
[5] S.J. Cummins, M.M. Francois, D.B. Kothe, Estimating curvature from volume fractions, Computers and Structure 83 (2005) 425–434.
[6] M.M. Francois, S.J. Cummins, E.D. Dendy, D.B. Kothe, J.M. Sicilian, M.W. Williams, A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework, Journal of Computational Physics 213 (2006) 141–173.
[7] F.H. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Physics of Fluids 8 (1965) 2182–2189.
[8] M. Herrmann, A balanced-force refined level set grid method for two-phase flows on unstructured flow solver grids, Center for Turbulence Research, Annual Research Briefs, 2006.
[9] C.W. Hirt, B.D. Nichols, Volume of fluid (VOF) method for dynamics of free boundaries, Journal of Computational Physics 39 (1981) 201–225.
[10] D. Jamet, O. Lebaigue, N. Coutris, J.M. Delhaye, The second gradient method for the direct numerical simulation of liquid–vapor flows with phase-change, Journal of Computational Physics 169 (2001) 624–651.
[11] D. Jamet, D. Torres, J.U. Brackbill, On the theory and computation of surface tension: the elimination of parasitic currents through energy conservation in the second-gradient method, Journal Computational Physics 182 (2002) 262–276.
[12] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, Journal of Computational Physics 113 (1994) 134–147.
[13] D. Lorstad, M.M. Francois, W. Shyy, L. Fuchs, Assessment of volume of fluid and immersed boundary methods for droplet computations, International Journal of Numerical Methods in Fluids 46 (2004) 109–125.
[14] M. Meimer, G. Yadigaroglu, B.L. Smith, A novel technique for including surface tension in PLIC-VOF methods, European Journal of Mechanics B – Fluids 21 (2002) 61–73.
[15] M.-J. Ni, Effects of density average at cell faces on computational accuracy of multi-fluid flowss driven by surface tension, Numerical Heat Transfer Part B, submitted for publication.
[16] M.-J. Ni, M.A. Abdou, Temporal second-order accuracy of SIMPLE-type methods for incompressible unsteady flows, Numerical Heat Transfer, Part B 46 (2005) 529–548.
[17] M.-J. Ni, M.A. Abdou, A bridge between projection methods and SIMPLE type methods for incompressible Navier–Stokes equations, International Journal for Numerical Methods in Engineering 72 (2007) 1490–1512.
[18] M.-J. Ni, R. Munipalli, N.B. Morley, P. Huang, M.A. Abdou, A current density conservative scheme for incompressible MHD flows at low magnetic reynolds number. Part I: On a rectangular mesh, Journal of Computational Physics 227 (2007) 174–204.
[19] M.-J. Ni, R. Munipalli, P. Huang, N.B. Morley, M.A. Abdou, A current density conservative scheme for incompressible MHD flows at low magnetic reynolds number. Part II: On an arbitrary collocated mesh, Journal of Computational Physics 227 (2007) 205–228.
[20] M.-J. Ni, W.-Q. Tao, S.-J. Wang, Stability analysis for discretized steady convective–diffusion equation, Numerical Heat Transfer, Part B 35 (1999) 369–388.
[21] B.D. Nichols, C.W. Hirt, Methods for calculating multi-dimensional, transient free surface flows past bodies, Technical Report LA-UR-75-1932, Los Alamos National Laboratory, 1975.
[22] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, Journal of Computational Physics 79 (1988) 12–49.

[23] S.V. Patankar, D.B. Spalding, A calculation procedure for heat mass and momentum transfer in three-dimensional parabolic flows, International Journal Heat and Mass Transfer 15 (1972) 1787–1802.
[24] C.S. Peskin, Numerical analysis of blood flow in the heart, Journal of Computational Physics 25 (1977) 220–252.
[25] S. Popinet, S. Zaleski, A front-tracking algorithm for the accurate representation of surface tension, International Journal of Numerical Methods in Fluids 30 (1999) 775–793.
[26] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, W.J. Rider, A second-order projection method for tracking fluid interfaces in variable density incompressible flows, Journal of Computational Physics 130 (1997) 269–282.
[27] Y. Renardy, M. Renardy, PROST: a parabolic reconstruction of surface tension for the volume-of-fluid method, Journal of Computational Physics 183 (2002) 400–421.
[28] M. Rudman, Volume tracking methods for interfacial flow calculations, International Journal for Numerical Methods in Fluids 24 (1998) 671–691.
[29] S. Shin, S.I. Abdel-Khalik, V. Daru, D. Juric, Accurate representation of surface tension using the level contour reconstruction method, Journal of Computational Physics 203 (2005) 493–516.
[30] E. Shirani, N. Ashgriz, J. Mostaghimi, Interface pressure calculation based on conservation of momentum for front capturing methods, Journal of Computational Physics 203 (2005) 154–175.
[31] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, Journal of Computational Physics 114 (1994) 146–159.
[32] M. Sussman, E.G. Puckett, A coupled level set and volume-of-fluid method for computing 3D and asymmetric incompressible two-phase flows, Journal of Computational Physics 162 (2000) 301–337.
[33] H. Takewaki, A. Nishiguchi, T. Yabe, Cubic interpolated pseudoparticle method (CIP) for solving hyperbolic type equations, Journal of Computational Physics 61 (1985) 261–268.
[34] A.Y. Tong, Z. Wang, A numerical method for capillarity-dominant free surface flows, Journal of Computational Physics 221 (2007) 506–523.
[35] D.J. Torres, J.U. Brackbill, The point-set method: front-tracking without connectivity, Journal of Computational Physics 165 (2000) 620–644.
[36] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for the computations of multiphase flow, Journal of Computational Physics 169 (2001) 708–759.
[37] S.O. Unverdi, G. Tryggvason, Computations of multi-fluid flows, Physica D 60 (1992) 70–83.
[38] J.P. Van Doormaal, G.D. Raithby, Enhancement of the SIMPLE method for predicting incompressible fluid flows, Numerical Heat Transfer 7 (1984) 147–163.
[39] T. Ye, R. Mittal, H.S. Udaykumar, W. Shyy, An accurate Cartesian Grid method for viscous incompressible flows with complex immersed boundaries, Journal of Computational Physics 156 (1999) 209–240.
[40] T. Yabe, F. Xiao, T. Utsumi, Constrained interpolation profile method for multiphase analysis, Journal of Computational Physics 169 (2001) 556–593.
[41] D.L. Youngs, Time-dependent multi-material flow with large fluid distoration, in: K.W. Morton, M.J. Baines (Eds.), Numerical Method for Fluid Dynamics, Academic Press, New York, 1982, pp. 273–281.
[42] Y. Zang, R.L. Street, J.R. Koseff, A non-staggered Grid, fractional step method for time-dependent incompressible Navier–Stokes equations in curvilinear coordinates, Journal of Computational Physics 114 (1994) 18–33.