

A high-order approach to solving nonlinear differential equations applied to direct numerical simulation of two-phase unsteady flow

J. Tveit

University of Bergen, BKK Production, Norway

Abstract

A method for solving nonlinear differential equations, which facilitates the computation of solutions of a high polynomial degree on a grid, is tested for use in direct numerical simulation (DNS) of two-phase unsteady flow.

The method uses a grid discretization to approximate continuously distributed variables, represented by functions of time and space, in a given set of differential equations. The grid contains information about both the values and the values of the derivatives of the unknown functions at the grid points in the computational domain. With this method the derivatives are thus explicitly defined at each grid point rather than, as in conventional numerical schemes, implicitly given by the function values at the surrounding grid points. Using piecewise polynomial interpolation functions can be represented with an arbitrary order of continuity over the entire computational domain.

The high polynomial order used in this method allows for simulation of flow features smaller than the interval separating each grid point. This reduces the required number of grid points and the need to adapt the grid to complex boundary geometry or to the interphase between different fluid phases. This simplifies grid generation and reduces the computational cost.

Keywords: discretization, high order, direct numerical simulation, two-phase unsteady flow.



1 Introduction

The mathematical framework and algorithms employed are described in detail in ref. [1], together with computed results for the lid-driven cavity test case. This method has been developed for a finite element, residual minimizing type of approach.

In the current work we apply the method to three dimensional unsteady two phase flow. Simulations of a bubble in a cubical domain are carried out as a proof of concept.

The current results are obtained after some improvements have been made. We will therefore make a short review of these, as well as the changes that have been made in order to perform two-phase flow simulations.

2 Adaption to two-phase unsteady flow

2.1 Basis functions and conditioning

As shown in [1] the choice of interpolating basis functions is important with respect to the numerical conditioning of the resulting system of equations. Bernstein polynomials were found to have acceptable properties. However, in the current work we use a different set of polynomials (Table 1). The polynomials given in Table 1 are chosen especially such that they produce a well conditioned system. These polynomials are constructed such that at the end points (where the interpolating variable, x , is either zero or one) they satisfy the conditions given in Equations (1a)–(1b). Note that, for each order of continuity, there are an even number of basis functions. Of each set, the lower half ($\lambda \in \{0 \dots \Lambda/2 - 1\}$) corresponds to the point at $x = 0$ while the rest ($\lambda \in \{\Lambda/2 \dots \Lambda - 1\}$) corresponds to the point at $x = 1$.

$$\frac{\partial^k}{\partial x^k} b_\lambda^\Lambda(x)|_{x=0} = a_k \delta_{k\lambda} \quad (1a)$$

$$\frac{\partial^k}{\partial x^k} b_\lambda^\Lambda(x)|_{x=1} = a_k \delta_{(k+\Lambda/2)\lambda} \quad (1b)$$

$$\left| \int_{x=0}^1 b_\lambda^\Lambda(x) dx \right| = 1 \quad (1c)$$

$$k \in \{0 \dots \Lambda/2 - 1\} \quad (1d)$$

Here a_k is a positive normalization constant chosen such that sub-equation (1c) is satisfied. Thus, a polynomial approximation of a function $f(x)$, based on the function values at the two end points (i.e. grid points) is given directly by the



Table 1: Polynomial basis functions, b_γ^Γ , for different orders of continuity.

C^1	$b_0^4(x) = 2x^3 - 3x^2 + 1$ $b_1^4(x) = 6x^3 - 12x^2 + 6x$ $b_2^4(x) = -2x^3 + 3x^2$ $b_3^4(x) = -6x^3 + 6x^2$
C^2	$b_0^6(x) = -6x^5 + 15x^4 - 10x^3 + 1$ $b_1^6(x) = -15x^5 + 40x^4 - 30x^3 + 5x$ $b_2^6(x) = -30x^5 + 90x^4 - 90x^3 + 30x^2$ $b_3^6(x) = 6x^5 - 15x^4 + 10x^3$ $b_4^6(x) = -15x^5 + 35x^4 - 20x^3$ $b_5^6(x) = 30x^5 - 60x^4 + 30x^3$
C^3	$b_0^8(x) = 20x^7 - 70x^6 + 84x^5 - 35x^4 + 1$ $b_1^8(x) = (140x^7)/3 - 168x^6 + 210x^5 - (280x^4)/3 + (14x)/3$ $b_2^8(x) = 84x^7 - 315x^6 + 420x^5 - 210x^4 + 21x^2$ $b_3^8(x) = 140x^7 - 560x^6 + 840x^5 - 560x^4 + 140x^3$ $b_4^8(x) = -20x^7 + 70x^6 - 84x^5 + 35x^4$ $b_5^8(x) = (140x^7)/3 - (476x^6)/3 + 182x^5 - 70x^4$ $b_6^8(x) = -84x^7 + 273x^6 - 294x^5 + 105x^4$ $b_7^8(x) = 140x^7 - 420x^6 + 420x^5 - 140x^4$

values of $f(x)$ and its derivatives at the end points by

$$f(x) = \sum_{\lambda=0}^{\Lambda/2-1} b_\lambda^\Lambda(x) a_\lambda \frac{\partial^\lambda f(x)}{\partial x^\lambda} \Big|_{x=0} + \sum_{\lambda=0}^{\Lambda/2-1} b_{\lambda+\Lambda/2}^\Lambda(x) a_\lambda \frac{\partial^\lambda f(x)}{\partial x^\lambda} \Big|_{x=1} + \mathcal{O}(x^\Lambda)$$

At each grid point then, the values $\{a_0 f, a_1 f', a_2 f'', \dots\} \equiv \{\hat{f}, \hat{f}', \hat{f}'', \dots\}$ up to a desired order of continuity are stored (here, prime denotes derivative and $\hat{}$ indicates a normalized quantity). A matrix inversion is no longer needed to produce the piecewise polynomial approximation for each cell. As a consequence the floating point accuracy is no longer a limiting factor (see [1] section 2). Since the higher derivatives tend to take on values of greatly varying magnitude even with

small variations of the flow configuration, computing the scaled values directly rather than derivatives, improves the conditioning of the resulting equation system.

2.2 Basis functions in three dimensions

The basis functions are generalized to higher dimensions by taking the product, $B_{k,l,m}^\Lambda(x,y,z) = b_k^\Lambda(x)b_l^\Lambda(y)b_m^\Lambda(z)$. In the current work we use this discretization with the same order of continuity in the three spatial dimensions, and implicit marching in the temporal direction (it is also possible to employ this discretization in the temporal dimension, and to use different order of continuity in different directions).

2.3 Unsteady flow

The continuity and momentum equations depend on the fluid phase in a way which is not easily linearized. As a consequence we do not linearize all the governing equations into a single system. Instead the velocity, pressure and phase are mapped into separate linearized global equation systems, where the time derivatives of the next time-step are the unknowns (including time derivatives of the spatial derivatives, up to the given order of continuity). This is an implicit time–marching scheme (see Table 2) where the solution for each time step is found by repeatedly solving for velocity, pressure and phase, taking the previous solution as constant in each step (the nonlinear optimization used in [1] was not implemented, as current procedure alone produced acceptable convergence rates).

Table 2: Butcher tableau for the implicit marching scheme.

0	0	0
1	0	1
<hr/>		
	0	1

2.4 Interphase tracking

Numerical methods solving two-phase unsteady flow typically rely on either adjusting the discretization geometry of the computational domain to fit the interphase between the different fluid phases, or by using particles moving with the flow, like the Particle in Cell (PIC) method [2, 3] and its successor, the Smoothed Particle Hydrodynamics (SPH) [4] method.

The current method uses a constant grid combined with a sub-grid integration scheme to achieve sub-grid accuracy. An iso-surface of a scalar function, f , is

used to track the interphase between different fluid phases. This approach is also employed by for example the Volume of Fluid (VOF) method [5].

The scalar function, f , is discretized in the same manner as the velocity and density. Its time evolution is determined by convection along with the fluid flow (see appendix for details). At each sample point, the distance, r , from the interphase is approximated by $r \approx f/\sqrt{\nabla f \cdot \nabla f}$. A smoothing function, $s(r)$, which is nonzero for small values of r determines the surface effects (see appendix, Equation (7)). The smoothing function $s(r)$ is a polynomial with continuous first and second derivatives. The interphase is thus approximated by a layer near $f = 0$ of finite thickness. The necessary thickness depends on the density of the sample points. In the current work the interphase thickness was approximately 7.79×10^{-3} (relative to the size of the computational domain).

2.5 Preconditioning

Solving the global equation system for the velocity is a potential bottleneck as the resolution increases (the cost of the direct solution grows as $N^3(k+1)^3$, with N and C^k being the number of grid points and order of continuity). However, by using the Cholesky factorization of the initial equation system as a preconditioner, the system may be solved very efficiently in the subsequent iterations using the conjugate gradient (CG) iteration (typically around five CG iterations).

3 Governing equations

The differential form of the Navier–Stokes equations (dimensionless, scaled with appropriate physical quantities) are solved. Table 3 shows the numerical values of the different parameters determining the fluid properties. The grid dependent parameters are $\tau = \frac{T-1}{L-1}$ and $\eta = L - 1$, where L and T are the spatial and temporal grid resolutions, respectively. The reader may examine the appendix for a detailed formulation of the governing equations.

4 Simulation

The simulation is of a fictitious fluid with high viscosity. The aim is to demonstrate the method's applicability to two-phase unsteady flow together with boundary details on a sub-grid scale. The reader may refer to [1] for a verification and comparison of the results of this method with conventional methods. Figure 1 shows the set up. The grid used in this case uses C^2 continuity and thus a spatial (polynomial) order of five ($\mathcal{O}(x^6)$ terms are discarded). With seven-cubed grid points we have $L = 7 \Rightarrow \eta = L - 1 = 6$. Further we let one time unit correspond to sixty steps, thus $\tau = (T - 1)/(L - 1) = 10$.



Table 3: The numerical values of the parameters of the governing equations. The two fluids have equal properties except for the density, which is lower (by a factor 100) for the bubble (phase II), resulting in a higher value of β .

	phase I	phase II
α	10^{11}	10^{11}
β	1/10	10
Re	100	100
\vec{g}	(0, 0, -1)	(0, 0, -1)
σ	10^{-3}	10^{-3}

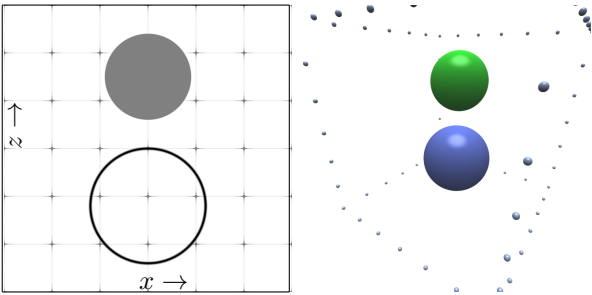


Figure 1: This figure shows the initial conditions of the system. The computational domain is a cube. A solid, spherical object with radius 0.15 is fixed at the center of the x - y plane at a (center) height 0.75. The lightest phase is initially a sphere with radius 0.2 positioned at the center of the x - y plane at a height 0.25 (scales relative to the size of the computational domain). A cross section of the computational domain of the grid is shown on the left hand side. On the right hand side we have a perspective rendering showing the solid and the interphase at its initial position. The initial flow velocity is zero and the initial pressure is constant. No-slip Dirichlet boundary conditions are enforced throughout the simulation. The grid resolution is indicated by dots (in this case $7^3 = 343$ grid points).

4.1 System configuration

Figure 1 describes the computational domain and the initial conditions. The initial distribution of the light fluid phase is axially symmetric and the computational domain is cubical.

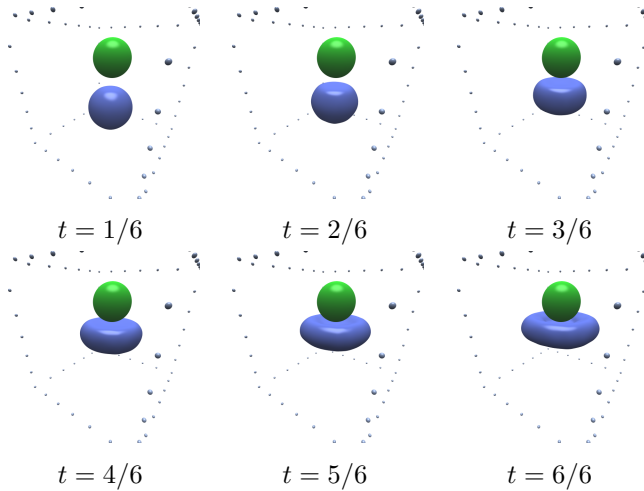


Figure 2: This figure shows a perspective rendering (obtained with ray casting) of the bubble interphase at different times. The surrounding dots are grid-points at the edge of the computational domain.

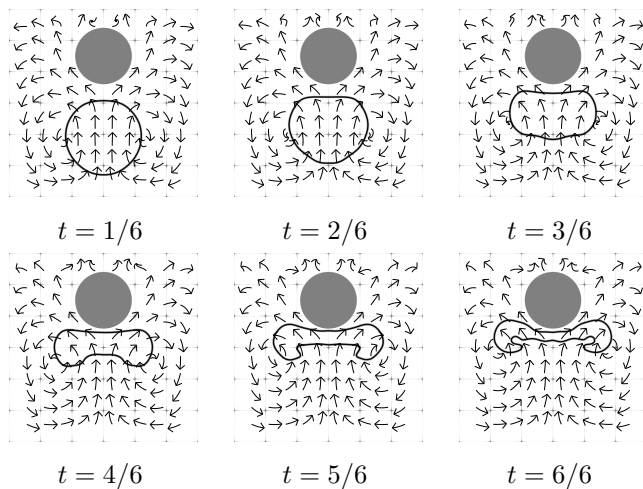


Figure 3: This figure shows the $x - z$ cross section (y centered) at different times. The arrows are of constant length in each figure and are drawn in a Lagrangian coordinate system, projected into the $x - z$ plane.

4.2 Time evolution

Figures 2 and 3 shows snapshots of the simulation at different times. Table 4 shows numerical values of theoretically verifiable quantities at different time-



Table 4: This figure shows computed numerical values of the physically constant quantities: total mass and total horizontal momentum in x - and y -directions (obtained with Monte Carlo integration). The exact value of the mass is $10 \left(1 - \frac{4}{3}\pi 0.2^3\right) + \frac{4}{3}\pi 0.2^3/10 \approx 9.668$ and the exact value of the horizontal momentum is zero.

step	mass	x -momentum	y -momentum
0	9.63863	0	0
5	9.63871	-1.69635×10^{-5}	8.60974×10^{-6}
10	9.63813	1.31676×10^{-5}	1.65046×10^{-5}
15	9.63808	9.96706×10^{-6}	5.95129×10^{-5}
20	9.63750	9.644×10^{-5}	9.89×10^{-5}
25	9.63620	0.000254756	0.000242913
30	9.63827	0.00035846	0.000307487
35	9.62337	0.000497964	0.000484558
40	9.63011	0.000585154	0.000473407
45	9.61777	0.000430384	0.000362024
50	9.61049	0.000440684	0.000301131
55	9.60459	0.000514161	0.000194891
60	9.60225	0.000688966	0.000309241

steps. As the bubble shape becomes stretched out and thinner compared to the grid resolution, an increased inaccuracy is observed.

5 Computational cost

The computational cost can be divided into two parts, (i) the numeric integration over all sample points which form the linearized system of equations, and (ii) the cost of solving these equations. In this simulation the numeric integration required most time (on average 262 seconds per iteration). Less than ten percent of the time was spent on solving the systems (on average 27 seconds per iteration) due to the rapid convergence of the CG iteration. It should be noted that the cost of the numeric integration grows linearly with the number of grid-points. It is also easily parallelizable.

5.1 Convergence

Figure 4 shows the convergence history for 12 different time-steps.



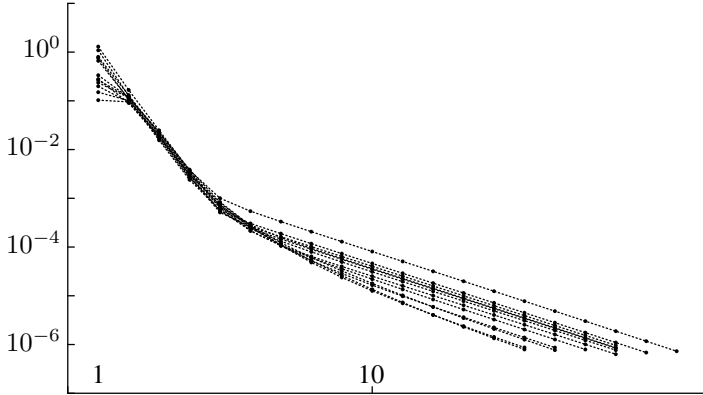


Figure 4: The convergence history of twelve different time-steps of the simulation is shown. The plotted value (dots) is the root of the sum of squares of the step length of all flow components in the grid for each iteration. Each iteration was terminated once this quantity dropped below 10^{-6} .

6 Conclusion and outlook

Two main problems have been tested in these simulations. i) Two phase flow and ii) sub grid geometry. Both of these were studied simultaneously without fundamentally changing the method to fit either issue. Compared with the two dimensional computations presented in [1] we see that the main computational effort is spent on numeric integration, while solving the linear systems is comparatively cheap due to efficient use of preconditioners. Since the algorithms used for numeric integration are easily parallelizable and have a $\mathcal{O}(N)$ cost, the benefit of increasing the hardware capabilities should be high compared to other methods.

Appendix

The dimensionless Navier–Stokes equations for two phases

The characteristic length and velocity scales, which map to unity in the computational domain, are x_0 and v_0 ($t_0 = v_0/x_0$) and define the dimensionless (non-primed) quantities:

$$\vec{v} = v_0 \vec{v} \quad (2a)$$

$$\rho' = \rho_0 \rho \quad (2b)$$

$$p'_{amb} = \rho_0 v_0^2 p_{amb} \quad (2c)$$



$$p' = \rho_0 v_0^2 (p + p_{amb}) \quad (2d)$$

$$\vec{g}' = \frac{v_0^2}{x_0} \vec{g} \quad (2e)$$

$$\frac{\partial}{\partial t'} = \frac{v_0}{x_0} \frac{\partial}{\partial t} \quad (2f)$$

$$\nabla' = \frac{1}{x_0} \nabla \quad (2g)$$

$$\mathbb{T}' = \frac{\mu v_0}{x_0} \mathbb{T} = \frac{\mu v_0}{x_0} \left(\nabla \vec{v} + (\nabla \vec{v})^T + \frac{\lambda}{\mu} (\nabla \cdot \vec{v}) \mathbb{I} \right) \quad (2h)$$

$$\sigma' = \sigma_0 \sigma = \rho_0 x_0 v_0^2 \sigma \quad (2i)$$

$$f' = f \quad (2j)$$

Here, ρ is density, p is pressure, \vec{v} is velocity, \mathbb{T} is the viscous stress tensor. The sign of the scalar function, f , defines the fluid phase. The equation of state is approximated by a linear relation between density and pressure. The superscript T , is the transpose and \mathbb{I} is the identity tensor. The phase dependent properties are μ and λ (first and second viscosity coefficients), ρ_{amb} (ambient density) and $k = K_{amb}/\rho_{amb}$ where K_{amb} is the bulk modulus at ambient conditions.

The dimensionless phase dependent properties are determined by the dimensionless parameters Re (viscosity), α (compressibility) and β (density):

$$Re = \frac{x_0 v_0 \rho_{amb}}{\mu}, \quad \alpha = \frac{k \rho_{amb}}{v_0^2 \rho_0} = \frac{K_{amb}}{v_0^2 \rho_0}, \quad \beta = \frac{\rho_0}{\rho_{amb}} \quad (3)$$

We let $\lambda/\mu = -2/3$. The viscous stress term, written as an operator \mathbb{S} , acting on the velocity reads

$$\nabla \cdot \mathbb{T} = \mathbb{S} \cdot \vec{v} = \begin{bmatrix} \nabla^2 + \frac{1}{3} \frac{\partial^2}{\partial x^2} & \frac{2}{3} \frac{\partial^2}{\partial x \partial y} & \frac{2}{3} \frac{\partial^2}{\partial x \partial z} \\ \frac{2}{3} \frac{\partial^2}{\partial x \partial y} & \nabla^2 + \frac{1}{3} \frac{\partial^2}{\partial y^2} & \frac{2}{3} \frac{\partial^2}{\partial y \partial z} \\ \frac{2}{3} \frac{\partial^2}{\partial x \partial z} & \frac{2}{3} \frac{\partial^2}{\partial y \partial z} & \nabla^2 + \frac{1}{3} \frac{\partial^2}{\partial z^2} \end{bmatrix} \cdot \vec{v} \quad (4)$$

The dimensionless formulation is then:

$$0 = \frac{1}{\alpha} \left[\frac{\partial p}{\partial t} + p \nabla \cdot \vec{v} + \vec{v} \cdot \nabla p \right] + \nabla \cdot \vec{v} \quad (5a)$$

$$0 = \frac{1}{\alpha} \left[p \left(\frac{\partial \vec{v}}{\partial t} + v \cdot \nabla \vec{v} - \vec{g} \right) \right] + \left(\frac{\partial \vec{v}}{\partial t} + v \cdot \nabla \vec{v} - \vec{g} \right) + \beta \nabla p - \frac{1}{Re} \mathbb{S} \cdot \vec{v} \quad (5b)$$

$$0 = \frac{\partial f}{\partial t} + \vec{v} \cdot \nabla f \quad (5c)$$

Since the fluids are assumed weakly compressible, α is large and the bracketed terms make only a small contribution. If the flow is incompressible ($\alpha \rightarrow \infty \Rightarrow$

$\nabla \cdot \vec{v} = 0$) it can be shown that \mathbb{S} reduces to $\mathbb{I}\nabla^2$. In the single-phase case one would choose $\rho_0 = \rho_{amb}$ yielding $\beta = 1$. In the two-phase case ρ_0 may be set to ρ_{amb} of one of the fluids, or something in between.

Adapting spatial and temporal scales to grid dimensions

The spatial and temporal scales in Equation (5) are defined so that their size is equal to the interval $[0, 1]^4$ in the computational domain. If the grid is uniform, floating point round off errors might be reduced by defining the characteristic length scales so that they instead correspond to the interval $[0, L - 1]^3 \times [0, T - 1]$ in the computational domain. With L being the spatial grid resolution and T the temporal grid resolution (i.e. the grid has $L \times L \times L \times T$ grid-points) the spacing between grid points becomes equal to one. The corresponding set of equations are:

$$0 = \frac{1}{\alpha} \left[\tau \frac{\partial p}{\partial t} + p \nabla \cdot \vec{v} + \vec{v} \cdot \nabla p \right] + \nabla \cdot \vec{v} \quad (6a)$$

$$0 = \frac{1}{\alpha} \left[p \left(\tau \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} - \frac{\vec{g}}{\eta} \right) + \left(\tau \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} - \frac{\vec{g}}{\eta} \right) + \beta \nabla p - \frac{\eta}{\text{Re}} \mathbb{S} \cdot \vec{v} \right] \quad (6b)$$

$$0 = \tau \frac{\partial f}{\partial t} + \vec{v} \cdot \nabla f \quad (6c)$$

where $\tau = \frac{T-1}{L-1}$ and $\eta = L - 1$.

Surface tension

Surface tension gives rise to a pressure discontinuity in the equilibrium case. Since the discontinuity is difficult to express accurately with continuous basis functions we add it as an additional force (source term) in the momentum equation instead of incorporating it in the pressure directly. The interphase is approximated by a small interval around $f = 0$ with a smoothing function, s , depending on the distance, r , from the interphase. The momentum equation, with surface tension included reads

$$0 = \frac{1}{\alpha} \left[p \left(\tau \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} - \frac{\vec{g}}{\eta} \right) \right] + \left(\tau \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} - \frac{\vec{g}}{\eta} \right) + \beta \left(\nabla p - \eta \sigma \frac{\nabla f}{|\nabla f|} \frac{\partial s}{\partial r} \nabla \cdot \left(\frac{\nabla f}{|\nabla f|} \right) \right) - \frac{\eta}{\text{Re}} \mathbb{S} \cdot \vec{v} \quad (7)$$

References

- [1] Tveit, J., A numerical approach to solving nonlinear differential equations on a grid with potential applicability to computational fluid dynamics. *arXiv*, 2014.



- [2] Harlow, F.H. & Welch, J.E., Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, **8**, p. 2182, 1965.
- [3] Harlow, F.H. & Welch, J.E., Numerical study of large amplitude free-surface motions. *Physics of Fluids*, **9**, p. 842, 1966.
- [4] Hu, X.Y. & Adams, N.A., An incompressible multi-phase sph method. *Journal of Computational Physics*, **227**, pp. 264–278, 2007.
- [5] Hirt, C.W. & Nichols, B.D., Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of Computational Physics*, **39**, pp. 201–225, 1981.

