An improved ghost-cell sharp interface immersed boundary method with direct forcing for particle laden flows

Maitri, R.V.; Das, S.; Kuipers, J.A.M.; Padding, Johan; Peters, E.A.J.F.

DOI
10.1016/j.compfluid.2018.08.018

Publication date
2018

Document Version
Accepted author manuscript

Published in
Computers & Fluids

An improved ghost-cell sharp interface immersed boundary method with direct forcing for particle laden flows

R. V. Maitri[a], S. Das[a], J. A. M. Kuipers[a], J. T. Padding[a,b], E. A. J. F. Peters[a,*]

[a] *Multiphase Reactors Group, Department of Chemical Engineering and Chemistry, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*

[b] *Intensified Reaction and Separation Systems, Department of Process and Energy, Delft University of Technology, Leeghwaterstraat 39, 2628 CB Delft, The Netherlands*

Abstract

In this paper, an accurate and stable sharp interface immersed boundary method(IBM) is presented for the direct numerical simulation of particle laden flows. The current IBM method is based on the direct-forcing method by incorporating the ghost-cell approach implicitly. An important feature of this IBM is the sharp representation of the solid surface, contrary to other variants of IBM for freely moving particles in which the solid surface is diffuse. Moreover, a correction of the diameter is not necessary for obtaining accurate results. The current ghost-cell IBM is stable because spurious oscillations incurred due to discontinuity in the pressure and velocity field in moving particle simulations is avoided. An algorithm for accurate torque computation is developed. The proposed algorithm is verified by comparison to an analytical expression and is shown to give a substantial improvement

*Correspoding author
E-mail address: E.A.J.F.Peters@tue.nl(E.A.J.F. Peters)

Preprint submitted to Computers and Fluids *July 6, 2018*

over the existing method. Finally, the present IBM is validated for various test cases of single and multi-particle systems and is shown to be accurate and robust for a wide range of flow conditions.

Keywords: Immersed boundary method, Ghost cell approach, Particle laden flow, Projected area, Torque computation, Spurious oscillations

1. Introduction

The numerical simulation of fluid-particle flows is important for many engineering applications as well as for the fundamental understanding of particle-laden flows. Applications include fluidized beds in the chemical process industry, oil and gas extraction processes like hydraulic fracturing, biological and environmental flows, and many more. The simulation of such systems requires a robust, efficient and accurate numerical model.

Traditionally, numerical methods with a body-fitted mesh are used to obtain a higher accuracy. However, this approach is computationally expensive in case a large number of moving particles is present since it requires generation of the grid after every few time steps. An efficient alternative to simulate systems involving a large number of moving particles is to use a fixed Cartesian grid embedding the I(mmersed) B(oundary) M(ethod). IBM was initially proposed by Peskin [1] to simulate blood flow in heart valves. Since then it has been adopted for a wider range of applications by improving the model. The advancements in the immersed boundary approaches are summarized in several review papers [2, 3, 4, 5].

The essence of the IBM is that the momentum equation for the fluid phase is modified to impose the no-slip boundary condition on the surface of the solid particle. This requires a forcing term to be added to the momentum equation for the fluid phase. Originally, Peskin [1] proposed a feedback forcing for this purpose. This approach was later used for non-deformable solid objects by Goldstein et al. [6] and Saiki and Biringen [7]. The major drawback of this method is that it contains free parameters which must be tuned according to the flow under consideration. Moreover, this approach poses a severe restriction on the time step size to avoid spurious oscillations in unsteady flows.

To work around this problem, Mohd-Yusof [8] proposed a direct forcing approach which does not require free constants. Mohd-Yusof [8] used this approach in a spectral code by applying a forcing on the solid boundary or inside the solid. The method is classified as direct forcing because the magnitude of the forcing term is based on the difference between the interpolated fluid velocity at the particle's surface and the desired velocity to be imposed at the same location. Fadlun et al. [9] implemented the direct forcing approach in a fully three-dimensional finite-difference framework and demonstrated that the direct forcing is more efficient than the feedback forcing. Fadlun et al. [9] used a forcing on the first fluid node next to the solid surface. The velocity at the immediate fluid node is obtained by linear interpolation of the velocity at the solid surface and the second fluid node from the solid surface. Since the velocity at the second fluid node is unknown, it is obtained by directly solving

the Navier-Stokes equation. The direct forcing approach of Mohd-Yusof [8] was later used by Kim et al. [10] in a finite-volume framework by improving mass conservation on the solid boundaries. An alternative sub-method of direct forcing, named ghost-cell immersed boundary method, is also used by some researchers [11, 12] for complex geometries. In this method, the ghost cell (solid node which has at least one neighbouring fluid node) is given a velocity such that the interpolated velocity at the solid surface is equal to the desired velocity to be imposed. This approach has been used to model solid objects in multiphase flows [13, 14, 15] as well as for moving boundary problems [16, 17, 18].

All the methods listed above are for static objects or moving objects whose motion is predefined and does not evolve based on the fluid forces acting on them. The seminal paper by Uhlmann [19] proposed a new IBM for moving particles which combines the direct forcing approach with the interpolation of forcing term from the Lagrangian marker points on a solid surface to Eulerian fluid nodes using a regularized delta function. This approach replaces the sharp interface of a particle by a porous shell of a width equivalent to the implemented delta function. As the Lagrangian marker points are located on the particle surface, half the width of the delta function lies outside the particle surface, thereby increasing the effective diameter of the particle and leading to an increase in the drag experienced by the particle. However, it was shown by Uhlmann [19] that the increased width of the delta function reduces oscillations and increases the stability. Another drawback of this method is its instability for applications where the parti-

cle density is close to the fluid density. The method of Uhlmann [19] has been improved by Kempe and Fröhlich [20] by avoiding a problematic factor $1/(\rho_p - \rho_f)$ through numerical integration of a volume integral instead of a rigid body assumption in solving the particle's equation of motion. This approach allows to simulate systems with density ratio $\rho_p/\rho_f \geq 0.3$, which removes the instability associated with neutrally buoyant particles. Later, Schwarz et al. [21] and Tschisgale et al. [22] further improved the immersed boundary method to simulated even lower density ratios. Breugem [23] extended the IBM of Uhlmann [19] by implementing the approach of Yu and Shao [24] for retracting the location of the Lagrangian marker points on the particle surface to increase the accuracy of the method. A generalized approach, which uses a calibration of the retraction distance as a function of local Reynolds number and solids volume fraction, was developed by Tang et al. [25] and accurate results were obtained for the flow in monodisperse arrays of spheres. Breugem [23] also implemented the multidirect forcing scheme of Luo et al. [26] and Kriebitzsch et al. [27] to improve enforcement of the no-slip boundary condition on the particle surface.

An alternative methods without the need of a calibration of a particle diameter have been proposed [18, 28]. It can be expected that a sharp representation of a particle alleviates the problem of a diffuse interface. However, the major challenge associated with sharp interface methods is the spurious oscillations near the solid surface, especially for moving boundaries. Different reasons are reported in literature for the cause of these oscillations. Lee et al. [29] reported two main reasons: One is associated with the spa-

tial discontinuity in the pressure field for freshly cleared cells (solid cells becoming fluid cells) whereas the second reason is related to the temporal discontinuity in the velocity field for the case of freshly occupied cells (fluid cells becoming solid cells). Moreover, Seo and Mittal [30] emphasized that the local mass conservation error near the immersed boundary is the main source of these oscillations. There have been efforts to mitigate these drawbacks by using a field extension method [31, 32] to obtain a physical value of flow variables at freshly cleared cells, applying local mass source and sink terms [33] or using a cut cell method [30] for better mass conservation. Although these methods have been successfully applied for complex moving boundaries, particulate flows with freely moving particles, where the particles' motion is determined by the flow field around them, is not widely simulated using sharp interface IBMs. Uhlmann [19] commented that sharp interface methods lead to force oscillations due to lack of smoothing and are undesirable for moving particle simulations. Kempe and Fröhlich [20] also support the claims of [19] and mention that there is a need of at least two or more orders of magnitude of grid cells across the particle diameter to obtain accurate results. However, Deen et al. [18] used a ghost cell method by implementing the sharp-interface approach and validated the model with the resolution of $D_p/h = 15$ for hydrodynamic test cases where the Reynolds number was below 40. Later, this model [18] was used to simulate a liquid fluidized bed. Ghost cell method of Deen et al. [18] is similar to the basic idea presented by Tseng and Ferziger [12] but [12] uses a boundary forcing term for ghost cells whereas Deen et al. [18] modifies the discretized fluid momentum equation and does not include any explicit forcing term. Mittal

et al. [17] also uses the approach of modifying discretized momentum equation based on the multidimensional interpolation by considering the image point in the normal direction of the surface. However, Deen et al. [18] uses one-directional extrapolation to maintain a compact structure of the stencil for a momentum equation. The issue of spurious oscillations is not investigated by Deen et al. [18]. In the current paper, the ghost cell method of Deen et al. [18] is modified to increase its accuracy and stability. Investigation of spurious oscillations is performed using modified method for moving particle case. Finally, this improved model is verified and validated for various test cases to prove its wide range of applicability without severe restrictions on time step or grid resolution.

2. Governing equations

In two phase particle-laden flows, the fluid phase is described by the Navier-Stokes equations along with the continuity equation, whereas the solid phase is described with the Newton-Euler equations.

2.1. Fluid phase:

The governing equations for incompressible Newtonian fluid flow are:

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

$$\rho_f \left(\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{uu} \right) = -\nabla p_a - \nabla p + \mu_f \nabla^2 \mathbf{u} \tag{2}$$

Here: $\mathbf{u} = (u_1, u_2, u_3)$ represents the velocity vector along the (x, y, z) coordinates, respectively. ∇p_a represents the constant pressure gradient applied to drive the flow; the dynamic pressure $p = (p_{\text{total}} - p_a - p_{\text{hydrostatic}})$. ρ_f and μ_f are the density and dynamic viscosity of the fluid, respectively.

2.2. Solid phase:

The surface velocity ($\mathbf{U_p}$) at a given location is obtained from particle's translational and rotational velocities:

$$\mathbf{U_p} = \mathbf{u_p} + \boldsymbol{\omega}_p \times \mathbf{r} \tag{3}$$

The position vector ($\mathbf{r}$) points from the particle centre-of-mass to the given location ($\mathbf{x}$) on the particle's surface:

$$\mathbf{r} = \mathbf{x} - \mathbf{x_p} \tag{4}$$

$\mathbf{x_p}$, $\boldsymbol{\omega}_p$ and $\mathbf{u_p}$ are centre-of-mass location, angular velocity and translational velocity of the particle, respectively. The time evolution of $\mathbf{u_p}$, $\mathbf{x_p}$ and $\omega_\mathbf{p}$ is given by the following equations:

$$\rho_p V_p \frac{d\mathbf{u_p}}{dt} = -\oint_{dS} \boldsymbol{\tau} \cdot \mathbf{n}\, dA - \oint_{dS} p\mathbf{n}\, dA - V_p \nabla p_a + (\rho_p - \rho_f) V_p\, \mathbf{g} + \mathbf{F}_{s\to p} \tag{5}$$

$$\frac{d\mathbf{x_p}}{dt} = \mathbf{u_p} \tag{6}$$

$$I_p \frac{d\omega_\mathbf{p}}{dt} = -\oint_{dS} \mathbf{r} \times (\boldsymbol{\tau} \cdot \mathbf{n})\, dA + \mathbf{T}_{s\to p} \tag{7}$$

In these equations, $\mathbf{F_{s\to p}}$ and $\mathbf{T_{s\to p}}$ are the force and torque exerted on a particle by another particle or the wall in the form of (sub-grid scale) lubrication or collision mechanisms. The immersed boundary method is used for fluid equations in the vicinity of the solid surface. The current IBM imposes the no slip boundary condition on the particle surface by modifying the discretized form of the momentum equation.

3. Numerical method

3.1. Fluid phase

In this section, the implemented immersed boundary method is explained in detail. The current IBM does not use a forcing term in the Navier-Stokes equation but the coefficient matrix of velocities that arises from discretizing the NS equation is modified for grid cells near solid surfaces. The given equations are discretized on a uniform, staggered Cartesian grid using the finite-volume method. The temporal discretization of the convection and diffusion terms is carried out by a second order accurate Adams-Bashforth and Crank-Nicolson method, respectively. The spatial derivatives are treated with the central-difference scheme.

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \frac{\Delta t}{\rho_f}\left[-\nabla p^{n+1} - \nabla p_a - \left(\frac{3}{2}\mathbf{C}^n - \frac{1}{2}\mathbf{C}^{n-1}\right) + \frac{1}{2}\left(\mathbf{D}^n + \mathbf{D}^{n+1}\right)\right] \tag{8}$$

where $\mathbf{C} = \rho_f \nabla \cdot \mathbf{uu}$ and $\mathbf{D} = \mu_f \nabla^2 \mathbf{u}$. The solution for the $\mathbf{u}^{n+1}$ is obtained in two steps. First, a predicted velocity $\mathbf{u}^*$ is obtained using the pressure field (p^n) at the old time step, as shown in Eq. 9, and then the final

velocity ($\mathbf{u}^{n+1}$) is computed by including the corrected pressure (Eq. 10) which is obtained by solving the Poisson equation (Eq. 11).

$$\mathbf{u}^* = \mathbf{u}^n + \frac{\Delta t}{\rho_f}\left[-\nabla p^n - \nabla p_a - \left(\frac{3}{2}\mathbf{C}^n - \frac{1}{2}\mathbf{C}^{n-1}\right) + \frac{1}{2}\left(\mathbf{D}^n + \mathbf{D}^*\right)\right] \tag{9}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{\Delta t}{\rho_f}\nabla\widetilde{p} \tag{10}$$

$$\nabla^2\widetilde{p} = \frac{\rho_f}{\Delta t}\nabla\cdot\mathbf{u}^* \tag{11}$$

$$p^{n+1} = p^n + \widetilde{p} \tag{12}$$

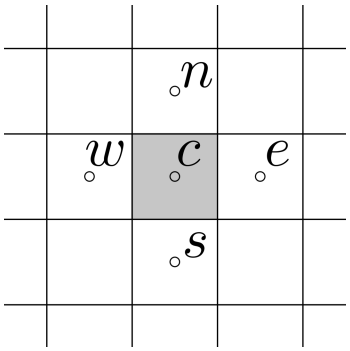


Figure 1: Representation of the stencil in 2D.

The algebraic representation of Eq. 9 is:

$$a_c u_{i,c} + \sum_{nb} a_{nb} u_{i,nb} = b_c + b_{diff} \tag{13}$$

a_c represents the coefficient of the velocity (u_i) at the central node ('c') for which the algebraic equation is formed. $i = (1, 2, 3)$ represents the direction of velocity along the (x, y, z) direction, respectively. a_{nb} ($nb = w, e, s, n$

in 2D) are coefficients for the velocities at neighbouring nodes of the central node. The schematic is shown in Fig. 1 for a 2D case. The right hand side of Eq. 13 is the summation of all explicit terms shown in Eq. 9. The separate representation of the contribution of the diffusion terms and the remaining terms is for the implementation of the Crank-Nicolson scheme in the current immersed boundary method (hence the usage of D^* in Eq. 9). The essence of the current IBM is based on the modification of Eq. 13 considering the presence of a nearby solid particle or not. In the whole flow domain, there will be three different cases, as discussed below.

Case 1 - Fluid node not in the vicinity of a solid particle:

Since there is no solid node near these nodes, we have a standard equation as in the single phase flow solver. The coefficients and the explicit terms are given by the following equations:

$$a_{nb} = -\frac{\Delta t}{2\Delta x_j^2}\mu_f \qquad a_c = \rho_f - \sum_{nb} a_{nb} \tag{14}$$

where Δx_j depends on the direction in which the given neighbouring node resides.

$$b_c = \rho_f {u_i}^n - \Delta t \left[\frac{\partial p^n}{\partial x_i} + \frac{\partial p_a^n}{\partial x_i} + \left(\frac{3}{2}{C_i}^n - \frac{1}{2}{C_i}^{n-1}\right)\right] \tag{15}$$

$$b_{diff} = \frac{\Delta t}{2}\mu_f \nabla^2 {u_i}^n \tag{16}$$

Case 2: Fluid node in the vicinity of a solid particle

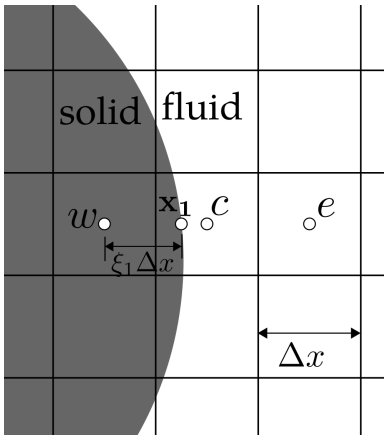


Figure 2: Representation of the fluid node neighbouring a single solid node.

In this case, the central node is still in the fluid but there exists at least one neighbouring node residing in the solid particle, as shown in Fig. 2. The basic idea of the current method is to represent the velocity at the solid node, i.e. $(u_{i,w})$ by second order extrapolation based on the velocity at the particle's surface $(U_p(\mathbf{x_1}))$, central node velocity $(u_{i,c})$ and the velocity of the following fluid node in the same direction $(u_{i,e})$. However, the velocity at 'c' and 'e' still needs to be solved. Therefore, the extrapolated value of $u_{i,w}$ as represented in Eq. 18 is put in Eq. 13. Now, the $u_{i,w}$ is eliminated from Eq. 13 and its contribution is reflected by modification of coefficients for $u_{i,c}$, $u_{i,e}$ as well as the explicit terms. Eq. 17 shows the modified equation and the underlying mathematics for the modified terms is given in Eqs. 19 - 21. It should be noted that the modification of the coefficients changes the contribution of the diffusion terms which has to be modified also for the explicit part of the diffusion to maintain the accuracy of the Crank-Nicolson scheme (Eq 21).

$$a_c^* u_{i,c} + \sum_{nb} a_{nb}^* u_{i,nb} = b_c^* + b_{diff}^* \tag{17}$$

case A: Only one neighbour in the solid particle

$$u_{i,w} = -\frac{2\xi_1}{1-\xi_1} u_{i,c} + \frac{\xi_1}{2-\xi_1} u_{i,e} + \frac{2}{(1-\xi_1)(2-\xi_1)} U_{p,i}(\mathbf{x_1}) \tag{18}$$

$$a_c^* = a_c - a_w \frac{2\xi_1}{1-\xi_1} \;\; ; \qquad a_e^* = a_e + a_w \frac{\xi_1}{2-\xi_1}; \qquad a_w^* = 0 \tag{19}$$

$$b_c^* = b_c - a_w \frac{2}{(1-\xi_1)(2-\xi_1)} U_{p,i}(\mathbf{x_1}) \tag{20}$$

$$b_{diff}^* = -\left(\sum_{j \neq w} a_j^* u_{i,j}^{old} - \rho_f u_{i,c}^{old} \right) - a_w \frac{2}{(1-\xi_1)(2-\xi_1)} U_{p,i}(\mathbf{x_1}) \tag{21}$$

case B: Two neighbours in the solid particle

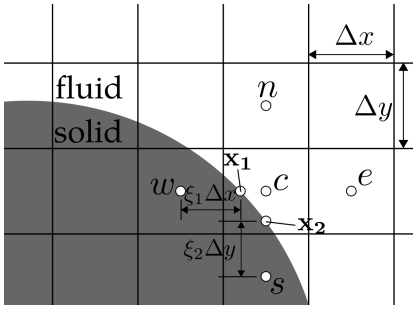
In this case, there are two neighbouring solid nodes - one in each direction, as shown in Fig. 3a. The extrapolation used in the current IBM is unidirectional, hence it allows us to modify the coefficient separately for each direction for this specific case. The coefficient and explicit terms are modified as shown in the Eqs. 22 - 25. The expressions for $u_{i,w}$, a_e^* and a_w^* are exactly same as for case A.

$$u_{i,s} = -\frac{2\xi_2}{1-\xi_2} u_{i,c} + \frac{\xi_2}{2-\xi_2} u_{i,n} + \frac{2}{(1-\xi_2)(2-\xi_2)} U_{p,i}(\mathbf{x_2}) \tag{22}$$

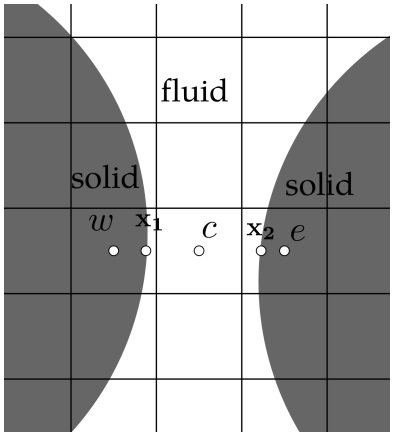
$$a_c^* = a_c - a_w \frac{2\xi_1}{1-\xi_1} - a_s \frac{2\xi_2}{1-\xi_2} \;\; ; \quad a_n^* = a_n + a_s \frac{\xi_2}{2-\xi_2} \;\; ; \quad a_s^* = 0 \tag{23}$$

$$b_c^* = b_c - a_w \frac{2}{(1-\xi_1)(2-\xi_2)} U_{p,i}(\mathbf{x_1}) - a_s \frac{2}{(1-\xi_2)(2-\xi_2)} U_{p,i}(\mathbf{x_2}) \tag{24}$$

$$b_{diff}^* = -\left(\sum_{j \neq w,s} a_j^* u_{i,j}^{old} - \rho_f u_{i,c}^{old} \right) - a_w \frac{2}{(1-\xi)(2-\xi)} U_{p,i}(\mathbf{x_1}) - a_s \frac{2}{(1-\xi)(2-\xi)} U_{p,i}(\mathbf{x_2}) \tag{25}$$



(a) One in each coordinate direction



(b) Both in same coordinate direction

Figure 3: Representation of the fluid node neighbouring two solid nodes.

In case of a multi-particle system, there will be cases where both solid nodes are in the same coordinate direction (i.e. $w \,\&\, e$ or $n \,\&\, s$), one in each particle as shown in Fig. 3b. In these cases, we do not have two fluid nodes for the second order extrapolation. Hence, linear extrapolation is used based

on the $u_{i,c}$ and $U_{p,i}(\mathbf{x_1})$ for $u_{i,w}$, and $u_{i,c}$ and $U_{p,i}(\mathbf{x_2})$ for $u_{i,e}$.

It should also be noted that the range of non-dimensional intersection distance (i.e. ξ) is {0,1}. Hence, terms like $(1-\xi)$ in the denominator will give large numbers as $\xi \to 1$ and make the simulations unstable causing spurious oscillations. To make the method more stable, the second order extrapolation is changed to first order whenever $\xi > \xi_{cutoff}$. The choice of ξ_{cutoff} depends on the trade-off between stability and accuracy. However, it was found that the stability of the method is maintained even when $\xi_{cutoff} = 0.999$.

Case 3: Solid node:

Having dealt with all the fluid nodes, it leaves us with nodes residing in the solid particle. In a moving particle system, a given node can be interchanged between a solid and a fluid for every time step according to the particles' location. It is important for these nodes to have a continuous history, without sudden jumps in the velocity and pressure. For this purpose, a continuous grid is used without any gaps in the solid region. The velocity at the nodes residing in the solid phase, including ghost nodes, is imposed with the velocity of the solid particle at that location ($u_{i,c} = U_{p,i}(\mathbf{x_c})$). The configuration is shown in Fig. 4. The following equations show the modified coefficients and explicit terms:

$$a_c^* = 1 \ , \ a_{nb}^* = 0 \ , \ b_c^* = U_{p,i}(\mathbf{x_c}) \ , \ b_{diff}^* = 0 \tag{26}$$

where $\mathbf{U_p}(\mathbf{x_c}) = \mathbf{u_p} + \boldsymbol{\omega} \times (\mathbf{x_c} - \mathbf{x_p})$.

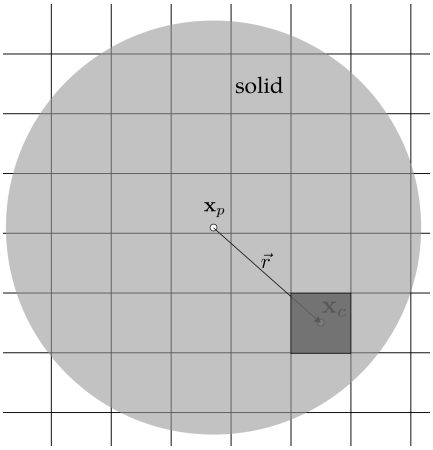


Figure 4: Representation of the solid node.

The described IBM algorithm is programmed in C with parallelization using the message-passing interface (MPI) for distributed computing on multiple processors. For the parallelization of the fluid part, a standard domain decomposition approach is used. The system of linear equations for velocity and pressure is solved using the Bi-CGSTAB solver for the asymmetric matrices with PFMG as a pre-conditioner using the hypre library [34].

3.2. Solid phase:

After solving the fluid flow from time step n_f to $n_f + 1$, the motion of solid particles is obtained by solving the Newton-Euler equations. The fluid forces (viscous and pressure) on a particle are computed at the start of the time step ($F^{n_f}_{f\to s}$) and after solving the flow field ($F^{n_f+1}_{f\to s}$).

$$\mathbf{F}_{f\to p} = \left[\frac{\mathbf{F}^{n_f}_{vis} + \mathbf{F}^{n_f+1}_{vis}}{2} + \frac{\mathbf{F}^{n_f}_{pres} + \mathbf{F}^{n_f+1}_{pres}}{2} \right] \tag{27}$$

$$\mathbf{T}_{f\to p} = \left[\frac{\mathbf{T}_{vis}^{n_f} + \mathbf{T}_{vis}^{n_f+1}}{2}\right] \tag{28}$$

where the subscripts vis and pres refer to viscous and pressure forces, respectively. These will be dealt with in more detail in the next subsection. For the cases where sub-grid scale lubrication and collision forces are acting, a finer time step (Δt_{dem}) is used compared to the fluid time step ($\Delta t_f = n\Delta t_{dem}$) for updating the particle location as well as linear and angular velocities. For all the DEM time steps within a single fluid time step, the average of the pressure and viscous forces (Eq. 27) and the torque (Eq. 28) is kept constant. The integration is carried out based on the velocity Verlet method as follows:

```
for i = 1:n
```

$$\mathbf{u}_{p,t_{i+\frac{1}{2}}} = \mathbf{u}_{p,t_i} + \frac{1}{m_p}\left[\mathbf{F}_{f\to p} + \mathbf{F}_{lub}^{i} + \mathbf{F}_{col}^{i} + \mathbf{F}_{body}\right]\frac{\Delta t_{dem}}{2} \tag{29}$$

$$\boldsymbol{\omega}_{p,t_{i+\frac{1}{2}}} = \boldsymbol{\omega}_{p,t_i} + \frac{1}{I_p}\left[\mathbf{T}_{f\to p} + \mathbf{T}_{col}^{i}\right]\frac{\Delta t_{dem}}{2} \tag{30}$$

$$\mathbf{x}_{p,t_{i+1}} = \mathbf{x}_{p,t_i} + \Delta t_{dem}\ \mathbf{u}_{p,t_{i+\frac{1}{2}}} \tag{31}$$

$$\boldsymbol{\omega}_{p,t_{i+1}} = \boldsymbol{\omega}_{p,t_{i+\frac{1}{2}}} + \frac{1}{I_p}\left[\mathbf{T}_{f\to p} + \mathbf{T}_{col}^{i+1}\right]\frac{\Delta t_{dem}}{2} \tag{32}$$

$$\mathbf{u}_{p,t_{i+1}} = \mathbf{u}_{p,t_{i+\frac{1}{2}}} + \frac{1}{m_p}\left[\mathbf{F}_{f\to p} + \mathbf{F}_{lub}^{i+1} + \mathbf{F}_{col}^{i+1} + \mathbf{F}_{body}\right]\frac{\Delta t_{dem}}{2} \tag{33}$$

```
end
```

For any variable changing with the DEM step,

$$\left(\phi^{n+1}\right)_{n_f} = \left(\phi^{1}\right)_{n_f+1} \tag{34}$$

3.2.1. Computation of the drag force on a particle

The drag force and torque on a solid element dS of a sphere with local (outward pointing) normal $\mathbf{n}$ is:

$$d\mathbf{F} = \underbrace{-\boldsymbol{\tau} \cdot \mathbf{n}\, dS}_{\mathbf{dF}_{vis}} - \underbrace{p\mathbf{n}dS}_{\mathbf{dF}_{pres}} \tag{35}$$

$$d\mathbf{T} = \mathbf{r} \times d\mathbf{F} = -\mathbf{r} \times (\boldsymbol{\tau} \cdot \mathbf{n})\, dS - \cancelto{0}{(\mathbf{r} \times \mathbf{n})\, p\, dS} \tag{36}$$

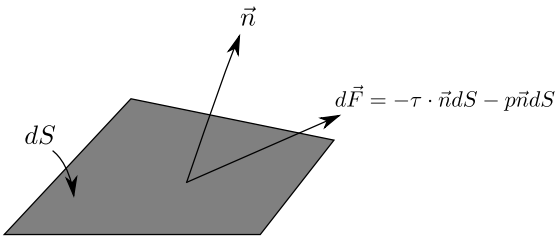


Figure 5: Calculation of a fluid force on a solid surface.

For a Newtonian fluid:

$$\boldsymbol{\tau} = -\mu_f(\nabla\mathbf{u} + \nabla\mathbf{u}^T) \tag{37}$$

Integrating the force equation for viscous drag on a closed solid surface:

$$\mathbf{F}_{vis} = -\oint_S \boldsymbol{\tau} \cdot \mathbf{n}\, dS = \mu_f \oint_S (\nabla \mathbf{u} + \nabla \mathbf{u}^{\mathbf{T}}) \cdot \mathbf{n}\, dS \tag{38}$$

By applying Gauss's theorem, the transpose term in Eq. 38 cancels due to the continuity equation.

$$\mathbf{F}_{vis} = \mu_f \oint_S (\nabla \mathbf{u} + \cancelto{0}{\nabla \mathbf{u}^{\mathbf{T}}}) \cdot \mathbf{n}\, dS = \mu_f \oint_S \mathbf{n} \cdot \nabla \mathbf{u}\, dS \tag{39}$$

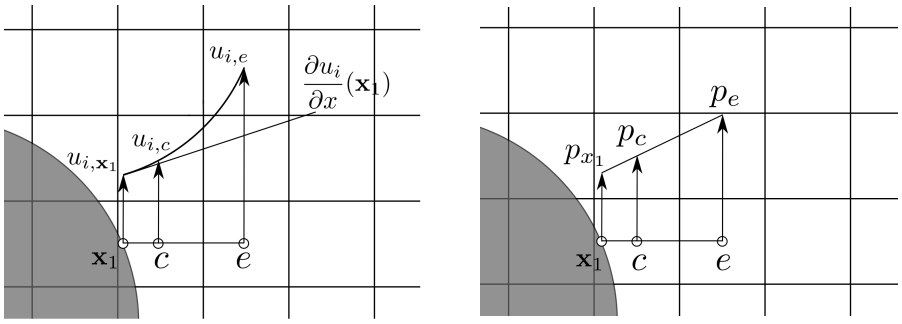
The final expression for the force in the i-direction is:

$$F_i = \oint_S dF_i = \oint_S \left[\mu_f \left(\frac{\partial u_i}{\partial x_1} n_1 + \frac{\partial u_i}{\partial x_2} n_2 + \frac{\partial u_i}{\partial x_3} n_3 \right) - p n_i \right] dS \tag{40}$$

Original approach:

In the original method of Deen et al. [18], numerical integration of each term of Eq. 40 is carried out with the following steps:

1. All the fluid nodes are identified which have a neighbouring solid node in the i-direction.
2. The shear stress at the intersection is computed based on a second order fit for velocity and extrapolation for pressure in the i^{th} direction, as shown in Fig. 6. If the distance between x_1 and c (i.e. $(1-\xi)\Delta x_i$) is close to 0, then the second order fit can give a singularity ($\xi = 1$). Since, the particle movement is based on the drag force, an abrupt increase in the force can cause instability in the code. Therefore, the shear rate at the particle surface is calculated with a linear fit for $\xi > \xi_{cutoff}$ (Eq. 41).



(a) Shear rate at the surface (b) Pressure at the surface

Figure 6: Interpolation/extrapolation to obtain pressure and shear rate at the solid surface.

$$\frac{\partial u_i}{\partial x}(x_1) = \begin{cases} \frac{1}{\Delta x}\left(\frac{2-\xi}{1-\xi}u_{i,c} - \frac{1-\xi}{2-\xi}u_{i,e} - \frac{3-2\xi}{(1-\xi)(2-\xi)}U_{p,i}(x_1)\right) & \xi < \xi_{cutoff} \\ \frac{1}{\Delta x}\left(u_{i,e} - u_{i,c}\right) & \xi > \xi_{cutoff} \end{cases} \tag{41}$$

$$p_{x_1} = (2-\xi)\,p_c - (1-\xi)\,p_e \tag{42}$$

3. The term '$n_i\,dS$' is the projected area in the i-direction and it is taken as $\Delta V_{cell}/\Delta x_i$.

Improved method :

In the original method, the obtained force values at each fluid cell neighbouring a solid surface were also used for the torque computation. However, it should be noted that the cancelled term in Eq. 39 will have a contribution to the torque computed in each cell neighbouring a solid. This term will not cancel when integrated over a whole surface. Moreover, the contribution of the fluid force on a solid particle in the highlighted cells as shown in Fig. 10

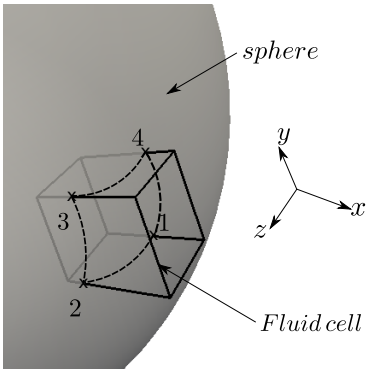


Figure 7: Intersection of the Eulerian cell with a spherical particle.

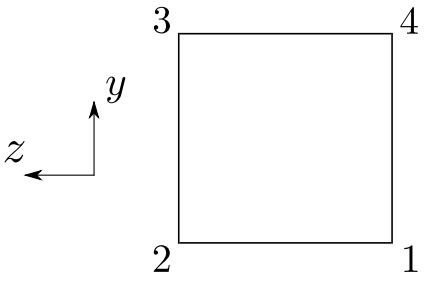
(referred to as missed cells hereafter) is not considered in the original scheme. First, the inclusion of the transpose term $\nabla \mathbf{u}^T$ in force and torque computation is explained, followed by the description of a method for considering the contribution of missed cells.

The viscous force on a particle on a given surface area in the x-direction with the inclusion of transpose terms is:

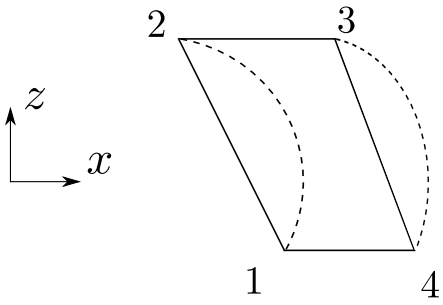
$$dF_{1,vis} = \mu_f \left[2\frac{\partial u_1}{\partial x_1} n_1 \, dS + \left(\frac{\partial u_1}{\partial x_2} + \underline{\frac{\partial u_2}{\partial x_1}} \right) n_2 \, dS + \left(\frac{\partial u_1}{\partial x_3} + \underline{\frac{\partial u_3}{\partial x_1}} \right) n_3 \, dS \right] \tag{43}$$

The integration of all terms except the underlined in Eq. 43 can be worked out using the original approach. However, the original approach uses the complete projected areas even in the case where all fluid edges don't intersect the spheres. Moreover, the computation of the underlined terms involves the computation of projected areas perpendicular to the direction of traversal, i.e. the x-direction for this example. The projected areas are

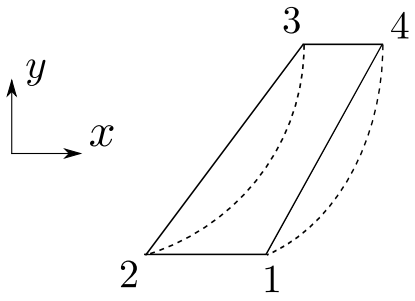
computed through the following steps:



(a) Projected area in y-z plane



(b) Projected area in x-z plane



(c) Projected area in x-y plane

Figure 8: Projected geometry on all the planes of a quadrilateral formed by the intersection points on a sphere.

1. The edges of the neighbouring fluid cells parallel to the x-axis are used to compute the points of intersection with the sphere (Fig. 7). If all the four edges intersect with the sphere, projected geometries of these intersection points in all the planes will be as shown in the Fig. 8
2. The projected area formed by the intersection points (if joined by straight lines) is denoted by $\mathbf{A_s}$ and is computed using Eq. 44. The

derivation of this equation is given in Appendix A.

$$\mathbf{A_s} = \frac{1}{2}\sum_{i=1}^{n}(\mathbf{r}_i \times \mathbf{r}_{i+1}) \tag{44}$$

Here, $\mathbf{r}_{n+1} = \mathbf{r}_1$. $\mathbf{A_s} = \{A_1, A_2, A_3\}$ where A_i is the projected area in the i-direction.

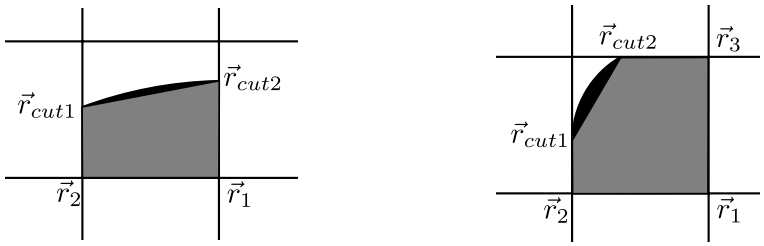
3. The projected areas can be computed even more accurately by correcting for the curvature and the expression for the modified projected area, $\mathbf{A_c}$, is given by the following equation as per the derivation in Appendix A.

$$\mathbf{A_c} = \mathbf{A_s} + \frac{1}{2}\sum_{i=1}^{n}(M_i - 1)\left[(\mathbf{r}_i - \mathbf{r_c}) \times (\mathbf{r}_{i+1} - \mathbf{r_c})\right] \tag{45}$$

$$M_i = \begin{cases} 1 & \text{straight edges} \\ \frac{\theta}{sin(\theta)} & \text{curved edges} \end{cases}$$

4. The cases where not all four edges intersect with the sphere are shown in Fig. 9. In these figures, the intersection positions r_{cut1} and r_{cut2} can be obtained analytically since these points lie on the circle formed by the intersection of a sphere with the plane passing through it's centre and perpendicular to the intersecting edges. It should be noted that in the original method of Deen et al. [18], the complete projected cell area (i.e.$\Delta y \Delta z$ for this case) is considered, which overestimates the real projected area.

5. In the previous steps, the drag force is calculated for pairs of fluid and neighbouring solid nodes which confirms the presence of the interface on the line connecting these nodes. Fig. 10 shows such combinations (•



(a) 2-edges intersecting

(b) 3-edges intersecting

Figure 9: Projected areas in the traversal direction for cases where all the 4 edges don't intersect with sphere.

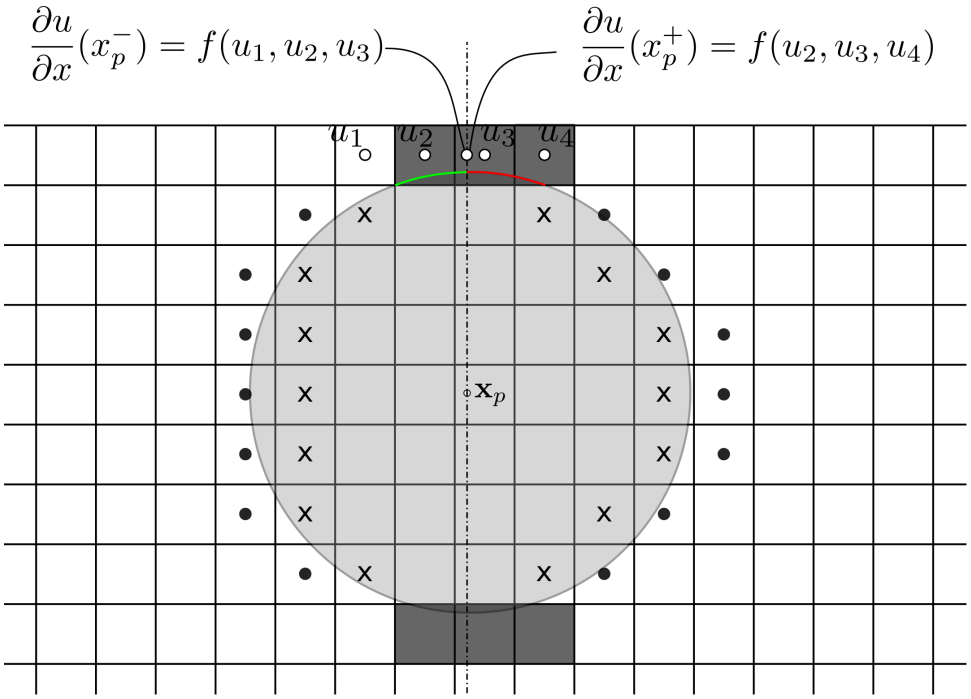


Figure 10: Representation of the the missed cells and the shear rate computation in those cells.

- fluid, × - solid). However, it should be noted that the highlighted cells (missed cells) are near a solid particle yet are not accounted for in the

previous studies. In this case, the projected areas are calculated cumulatively for the parts on each side of the centre plane passing through the sphere. The approximate representation of this case is shown in Fig. 11. Since we don't have the intersection points connecting the node centres for this case, the shear rate is calculated at the same x-location(x_p) as that of the sphere centre ($\mathbf{x}_p$) and the y and z-location same as that of the neighbouring nodes, as shown in Fig. 10. The shear rate is calculated based on a quadratic fit between the velocity values whose choice is based on the direction (Fig. 10). With the combination of the projected area and shear rate, force values can be obtained for all similar cells.

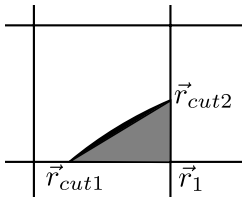


Figure 11: Projected area in the traversal direction for the missed cells.

6. There is another special case which occurs quite rarely but is still probable. Fig. 12 shows an accurate representation of such a case. Such type of cells are also ignored in the original method of Deen et al. [18]. The shear rate is calculated for such cells on the nearest intersection point. Again, the projected areas can be found with the algorithm outlined above.

7. This approach is repeated for velocity in each direction with the extension of traversal direction in y and z direction to compute the $d()/dy$

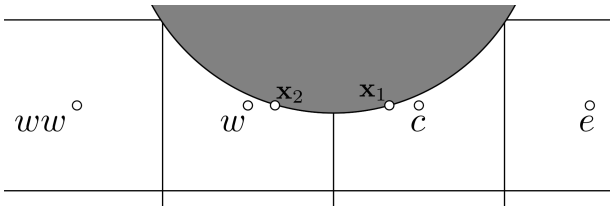


Figure 12: Representation of a case where the particle is present between two fluid nodes.

and $d()/dz$ terms, respectively. These shear rates are then multiplied with the respective projected areas to get the value of the force and torque.

The particle-particle and particle-wall collision force and torque values are computed using the a soft-sphere model of Cundall and Strack [35]. Unresolved (sub-grid scale) hydrodynamic lubrication forces are modeled by following an approach similar to Breugem [36]. In the current approach, the lubrication force is activated when a particle-wall gap width is closer than 2Δ and a particle-particle gap width is closer than Δ, whereas Breugem [36] used the activation distance based on the particle size instead of the grid size.

The original method of [18] with extrapolated pressure force computation in dense system of particles gave instability. Therefore volumetric pressure force approach was used which uses the pressure value at a neighbouring fluid cell instead of the pressure at particles' surface. In current method, contribution of a pressure force is ignored when an extrapolation involves pressure inside a solid object and the lubrication model takes care of the unresolved force. For remaining cases, linear extrapolation is used to get

the pressure value at the surface of a particle. This approach gives a better stability and accuracy for multiparticle simulations.

4. Results

In this section, the accuracy of the present IBM is demonstrated for fixed as well as *freely* moving particles. Since there is no forcing term associated with the particle grid in the current IBM, the force and torque have to be computed on the particle based on the available flow field. The accuracy of the present model is first verified for a torque computation where the flow field is prescribed. Later, different test cases are used for the validation of the complete IBM implementation by solving the flow field and equation of motion for the particles.

4.1. ***Verification***

In particle laden flows, the particles are always in translational as well as rotational motion. The force computation by the original IBM of Deen et al. [18] was validated for the translational motion indicating acceptable accuracy of the force computation. The original approach works quite well for the force computation since the contribution of the transpose terms of Eq. 39 is zero. Moreover, the neglected missed cells have a very small contribution due to their small projected area. However, the IBM method also has to be verified for the torque computation. The original IBM [18] is improved in this paper by including the transpose terms and performing accurate computation of the projection areas and consequently, the torque. The following

test case verifies the accuracy of the improved algorithm.

4.1.1. Torque on a rotating sphere

In this test case, the torque on a sphere which is slowly rotating in a quiescent fluid (in the Stokes regime) is computed. The analytical expression of the fluid velocity profile is given by Eq. 46 and the computed torque is compared with the analytical expression of the torque (Eq. 47) [37] (p. 96). Pressure does not contribute to the torque since pressure forces are always directed radially inwards. This test case has been simulated previously by researchers to test the accuracy of fixed as well as body fitted approaches for fluid-solid simulations [38, 39].

$$\mathbf{v} = (\boldsymbol{\omega} \times \mathbf{r}) \left(\frac{R_p}{r} \right)^3 \tag{46}$$

$$\mathbf{T}_p = -8\pi\mu R_p^3 \boldsymbol{\omega} \tag{47}$$

In this test case, we used: $\mu = 2$ kg/ms , $\boldsymbol{\omega} = (0,\, 0,\, 10^{-3})\text{s}^{-1}$, $R_p = 0.1$ m which gives rise to a torque of $50.265\,\mu Nm$. Table 1 summarises the torque values obtained using the original method [18] and the improved IBM. It can be clearly observed that the error in torque value is high for the original IBM, because the ignored transposed terms contribute 1/3 of the total analytical torque value. Baltussen [38] extended the original IBM [18] by including these transpose terms and using the approach that the projected areas scale proportionately to the magnitude of a normal vector in that specific direction. However, the contribution of missed cells was not considered in that

approach. The errors associated with the torque computation of [38] shows inconsistent convergence behaviour with the grid size. It is also reported that the error values are above 1% for a relatively fine resolution($R_p/h = 100$). The present algorithm is found to improve the results of torque computations significantly over both previous approaches. The projected areas calculated by using the curved edges further improves the results. At a relatively coarse grid resolution of $R_p/h = 5$, the error values are only around 3%. These results fall below 2% when the grid is refined by a factor of two. Since the present IBM is developed with the aim of simulation of a system with multiple particles, it is important to have a reasonable accuracy even on coarser grids. The presented results show that the current approach does not have a severe restriction on the grid size for obtaining accurate results for the case of rotating particles.

Table 1: Results for torque on a rotating sphere.

	Deen et al. [18]		Present (straight edges)		Present (curved edges)	
R_p/h	*Torque* (μNm)	*%Error*	*Torque* ((μNm)	*%Error*	(μNm)	*%Error*
5	34.0	**34.34** %	48.0	**4.54** %	48.7	**3.06** %
10	33.7	**33.04** %	49.4	**1.78** %	49.7	**1.21** %
20	33.5	**33.24** %	49.9	**0.68** %	50.0	**0.46** %
40	33.5	**33.32** %	50.1	**0.26** %	50.2	**0.17** %

4.2. ***Validation***

In this section, more test cases are presented to validate the present IBM for a wide range of flow regimes.

4.2.1. ***Flow through periodic array of spheres***

Flow through periodic array of spheres can be simulated using only one sphere in a cubic box with the periodic boundary condition in all directions. This assumption holds true as long as the flow is laminar. The simulation conditions used are similar to the one used by Breugem [23]. The diameter of a sphere is half the length of the cubical domain, corresponding to a solids volume fraction, $\epsilon_s = \frac{\pi}{6}\left(\frac{1}{2}\right)^3 \approx 0.065$. The pressure gradient, $dp/dx = -0.233\mu_f\nu_f/D_p^3$, is used to drive a flow. The Reynolds number based on the average bulk velocity(U_b) is less than 0.1. All the simulations in this paper, unless specified otherwise, are performed with $\xi_{cutoff} = 0.95$.

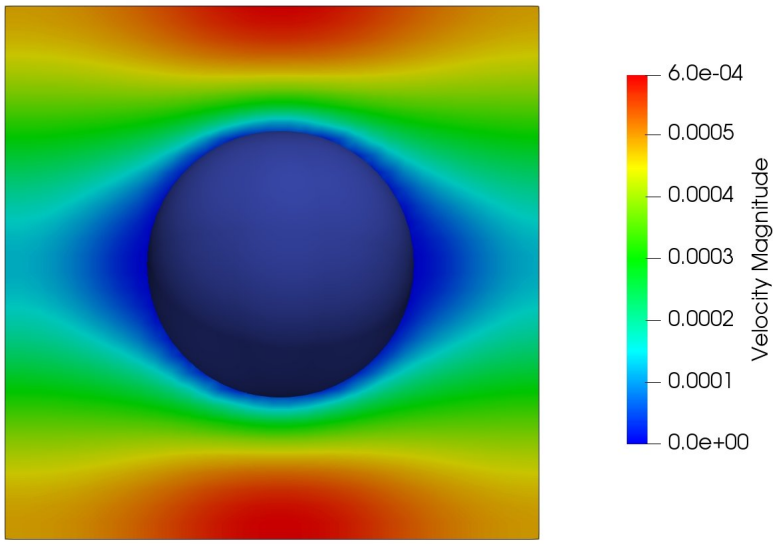


Figure 13: Cross-section of a laminar flow through a periodic array of spheres at a grid resolution of $D_p/h = 16$.

The simulations are performed for $D_p/h = 8$, 16, 32, 64. For the compar-

Table 2: Darcy number and non-dimensional drag force in the system of periodic array of spheres

D_p/h	Da	$\bar{F}$	$\Delta Da/Da_{64}$ (%)	$\Delta\bar{F}/\bar{F}_{64}$ (%)
8	0.29943	2.7141	0.2845	-0.537
16	0.29859	2.7303	0.0014	0.054
32	0.29856	2.7295	-0.0063	0.024
64	0.29858	2.7288	0.0	0.0

ison of results at different grid resolutions, the Darcy number, Da (Eq. 48) is used.

$$Da = \frac{\mu_f U_b}{(-dp/dx)D_p^2} \tag{48}$$

It can be observed from the expression for Darcy number that $\mu_f, dp/dx$ & D_p are constants for all the simulations. Hence, the variation of Darcy number eventually represents variation in the bulk velocity, U_b. The flow profile at a cross-section for $D_p/h = 16$ is shown in Fig. 13. The results for Darcy number, non-dimensional drag force and their respective errors are presented in the Table 2. The associated errors are computed in comparison with the values obtained at the finest grid ($D_p/h = 64$). Non-dimensional drag force is defined as:

$$\bar{F} = \frac{F_{f\to s}}{3\pi\mu D_p U_b} \tag{49}$$

Present IBM predicts the results within 1% accuracy even at a grid resolution of $D_p/h = 8$ compared to the grid converged results. Breugem [23], however, showed that the grid resolution of $D_p/h \sim 20$ is required to get the results below 1 % accuracy. This test case demonstrates an accuracy

of the present IBM with the sharp representation of a solid surface without the need of an effective diameter. The simulation of flow through periodic array of spheres is repeated for the grid resolution of $D_p/h = 5$ at solids volume fraction, $\epsilon_s = 0.065, 0.128, 0.303$ and the non-dimensional drag force is compared with the results of Zick and Homsy [40]. Obtained results show an error between 10 - 15 %. The deviation is in acceptable range and this resolution is later used for the simulation of a fluidized bed (section 4.2.9).

4.2.2. ***Sphere in a linear shear flow***

In this problem, the lift on a sphere which is kept at a fixed location is compared with the results of IBM simulations of Kempe and Fröhlich [20] and the pseudo-spectral code of Bagchi and Balachandar [41]. The simulations are performed for two cases: (i) non-rotating sphere (ii) freely-rotating sphere.

$$u_x = Sy \tag{50}$$

$$Re_p = \frac{\rho_f u_c D_p}{\mu_f} \tag{51}$$

$$G = \frac{S D_p}{u_c} \tag{52}$$

Here, S ($= 1\ \mathrm{s}^{-1}$) is the shear rate of the flow, G ($= 0.2$) is the shear parameter and Re_p is the Reynolds number based on the relative velocity u_c of the fluid at the particle centre. The spherical particle is resolved with $D_p/h = 20$ and h is 0.01 m. The computational domain is of size $25D_p \times 10D_p \times 10D_p$ with the spherical particle fixed at ($10D_p$, $5D_p$, $5D_p$). Free-slip boundary

conditions are applied on the side walls.

$$C_L = \frac{F_y}{\frac{1}{2}\rho_f u_c^2 A_c} \tag{53}$$

The obtained lift coefficients (Eq. 53) for the simulated cases are shown in Table 3 and compared with literature data. This study uses the lowest and highest Reynolds number considered by [41] for the comparison. Although results show small deviations, an acceptable agreement is found when compared with the reference data. Lift coefficient results found in literature using different numerical techniques generally have more scatter compared to drag coefficient data and it is important to note that our results fall within an acceptable range of values. For the freely rotating sphere case, the angular velocity of the sphere normalized with the rotation rate of the fluid (S/2) is also in an acceptable match with the literature data, as shown in Table 4. Moreover, the drag force on a particle is computed for the non-rotating case and the results match quite well with the predictions of [41] (not shown here). Bagchi and Balachandar [41] simulated another test case where the torque-free rotation rate obtained from the simulations is imposed on a particle in a uniform flow of inlet velocity u_c and the lift coefficient is computed. A similar simulation is repeated with the present IBM and the lift coefficient obtained for $Re_p = 20$ is *0.35*. This is in excellent agreement with the value of *0.36* predicted by [41]. These validations establish the accuracy of the present IBM for rotating as well as non-rotating particles.

Table 3: Lift coefficient on a sphere in a linear shear flow.

Re_p	$C_{L,non-rotating}$			$C_{L,rotating}$		
	Present	Bagchi [41]	Kempe [20]	Present	Bagchi [41]	Kempe [20]
20	0.010	0.011	0.012	0.042	0.047	0.053
200	-0.063	-0.058	-0.048	-0.054	-0.047	-0.039

Table 4: Torque-free rotation rate of sphere in a linear shear flow.

Re_p	$\omega_p/\ \omega_f$		
	Present	Bagchi [41]	Kempe [20]
20	0.674	0.695	0.694
200	0.154	0.170	0.161

4.2.3. ***Torque on a rotating sphere***

This test case is same as the rotating sphere case in section 4.1 except that the flow field is solved with the imposed sphere rotation rate(ω) of 10^{-3} s^{-1}. The simulations are carried out for three different grid resolutions of $D_p/h = 10, 20, 40$ with $D_p = 0.2$ m. The computational domain is of size $8D_p \times 8D_p \times 8D_p$ with the spherical particle fixed at the centre of the domain - $(4D_p, 4D_p, 4D_p)$. No-slip boundary condition is applied on all walls and ζ_{cutoff} is 0.99. For this test case,

$$Re_r = \frac{\omega R_p^2}{\nu_f} = 0.005 \tag{54}$$

The comparison of the obtained torque results with the analytical torque value in the case of infinite domain is presented in the Table 5. The results are in acceptable agreement for $D_p/h = 10$ and are in excellent agreement for

the finer grids. The error associated with $D_p/h = 20$ is smaller than the finer grid results of $D_p/h = 40$. This might be due to a fortunate cancellation of errors with the torque computation, similar to the observation of Vreman [39].

Table 5: Results for the torque computation for the rotating sphere.

D_p/h	10	20	40
$\frac{\lvert T_{sim} - T_{exact} \rvert}{\lvert T_{exact} \rvert} \times 100$	5.16 %	0.52 %	1.31 %

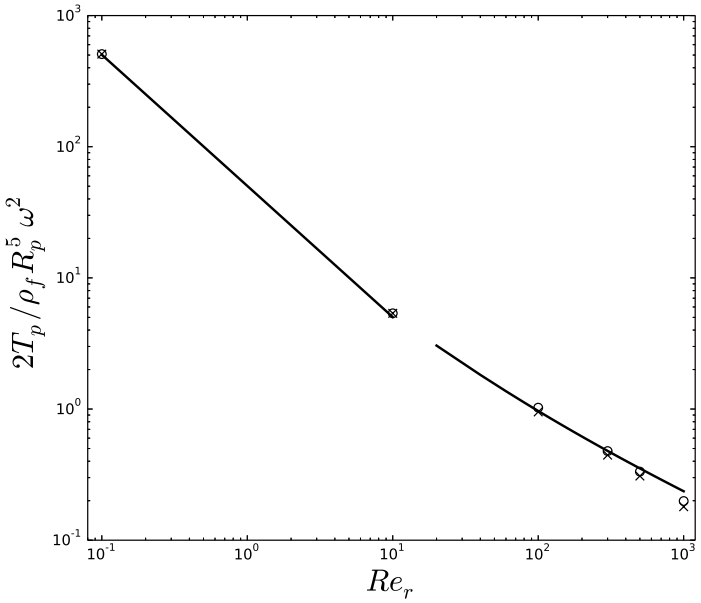


Figure 14: Results of non-dimensional hydrodynamic torque as a function of Re_r: Eq. 55 (—); $R_p/h = 10$ (×); $R_p/h = 20$ (∘).

For the torque on a sphere at higher Reynolds number, Sawatzki [42] performed experimental measurements. Dennis et al. [43] obtained an empirical

correlation which matches very well with the experiments of Sawatzki [42] in the range of rotation Reynolds number 20 to 1000. The expression for the non-dimensional torque, K ($\frac{2T_p}{\rho_f R_p^5 \omega^2}$) is:

$$\begin{aligned} &(Re_r < 1) && K = \frac{16\pi}{Re_r} \\ &(20 < Re_r < 1000) && K = \frac{6.45}{\sqrt{Re_r}} + \frac{32.1}{Re_r} \end{aligned} \tag{55}$$

The numerical results obtained with the current IBM are compared in Fig. 14 with the expression presented in Eq. 55. The comparison shows an excellent match with the reference data, except for $Re_r = 1000$. This discrepancy is caused by a thinner boundary layer associated with an increasing Reynolds number. Therefore, a finer grid is required to accurately capture the velocity gradients in the boundary layer around a sphere which directly influences the torque computation.

4.2.4. ***Drag on an oscillating sphere***

For sharp interface IBM approaches, it is a challenge to achieve smoothness of the pressure force in case of moving particles. Mittal et al. [17] showed that their ghost cell approach leads to spurious oscillations for the pressure force, which is caused by a local error in mass conservation. This problem was remedied using the cut-cell approach by Seo and Mittal [30]. Lee et al. [29] also reported the issue of spurious oscillations related to the spatial and temporal discontinuity in the flow variables in case of freshly cleared or dead cells. In the current IBM, the pressure equation is solved in the whole computational domain, including the cells inside the solid phase. No special

treatment is performed for the cells near the solid surface while solving the pressure Poisson equation. No boundary condition for pressure is applied on the solid surface and mass conservation is achieved in the whole computational domain at each time step. Hence, the converged solution for pressure gives a continuous distribution in time and space for these variables in case of moving particles as well. To verify the smoothness of the pressure force, a test case of an oscillating sphere is simulated. The simulation parameters are the same as chosen by Seo and Mittal [30].

$$x_c(t) = x_c(0) + X_0\,[1 - cos(2\pi ft)]\,; \quad u_c(t) = U_0\,sin(2\pi ft) \tag{56}$$

The sphere is resolved with 16 grid cells ($D_p/h = 16$) and the domain is of size $(4D_p)^3$. The position and velocity of the particle is governed by Eq. 56. The amplitude of oscillation is $X_0 = D_p/8$ and the frequency is $f = 1\,s^{-1}$. The period of an oscillation is resolved with 100 time steps which corresponds to CFL = 0.125. The Reynolds number, $Re = U_0 D_p/\nu_f$ is 78.54 and the Strouhal number, $St = fD_p/U_0$ is 1.2732. Free-slip boundary conditions are applied on x-walls and no-slip condition is applied for all the remaining walls. Neumann boundary condition for pressure is used on all the confining walls.

The time series data of a pressure-drag coefficient is plotted in Fig. 15. It can be observed that the drag coefficient profile is much smoother compared to the profile of Mittal et al. [17] and comparable to the results of Seo and Mittal [30]. It is important to note that the current IBM does not have spurious oscillations and maintains the simplicity and efficiency of the method due to continuity in pressure and velocity. This feature of the method will be

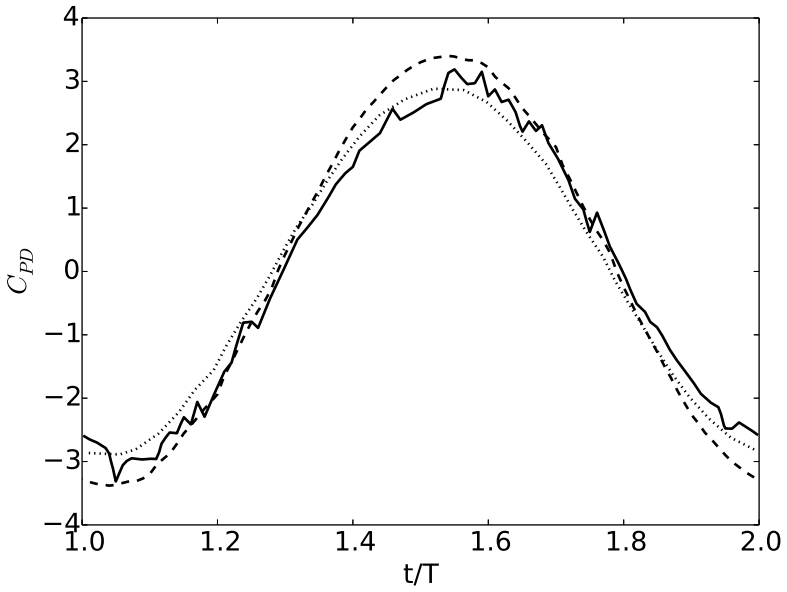


Figure 15: Variation of the pressure drag coefficient with time: Mittal et al. [17] (—); Seo and Mittal [30] (· · ·); Current IBM (- - -).

computationally advantageous in case of a multiparticle system. Moreover, the stability of the method for a coarse grid will enable us to simulate larger domains while saving computational costs.

4.2.5. ***Sedimentation of a buoyant particle*** *(*$Re_p < 35$*)*

Hereafter, test cases with freely moving particles are simulated. In this first test case, the sedimentation of a single buoyant particle in a closed container is simulated. The simulation parameters are same as the experiments of Ten Cate et al. [44]. This test case has been simulated by many researchers to check the accuracy of numerical codes [44, 45, 18, 20]. The simulation parameters are presented in Table 6 along with the Reynolds numbers reported in experiments[44]. The particle diameter(D_p) is 15 mm with a density (ρ_p)

of 1120 kg/m^3. The particle is resolved with $D_p/h = 15$. All walls of the domain are treated as no-slip boundaries. The CFL used for all the cases is around 0.4.

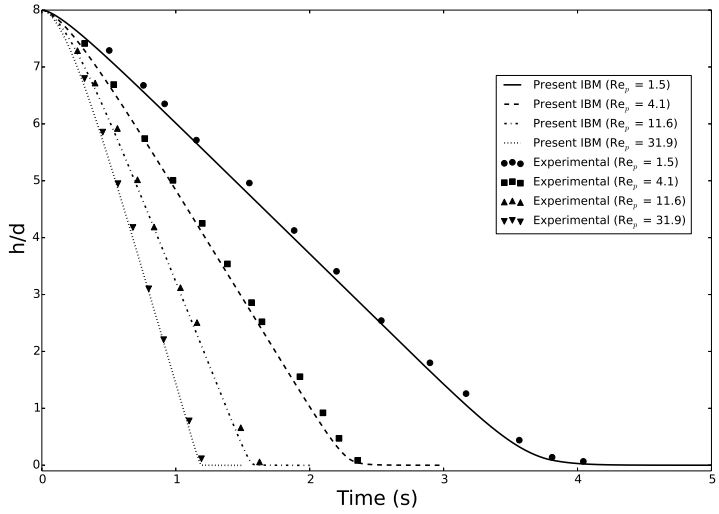
Table 6: Simulation parameters for a single sedimenting sphere ($Re_p < 35$).

Parameter	Case 1	Case 2	Case 3	Case 4
ρ_f	970	965	962	960
μ_f	0.373	0.212	0.113	0.058
$\mathbf{x}_p$	$(2.67D_p, 3.33D_p, 3.33D_p)$			
Ω	$10.67D_p \times 6.67D_p \times 6.67D_p$			
$\mathbf{g}$	$(9.81, 0, 0)$			
Re_p	1.5	4.1	11.6	31.9

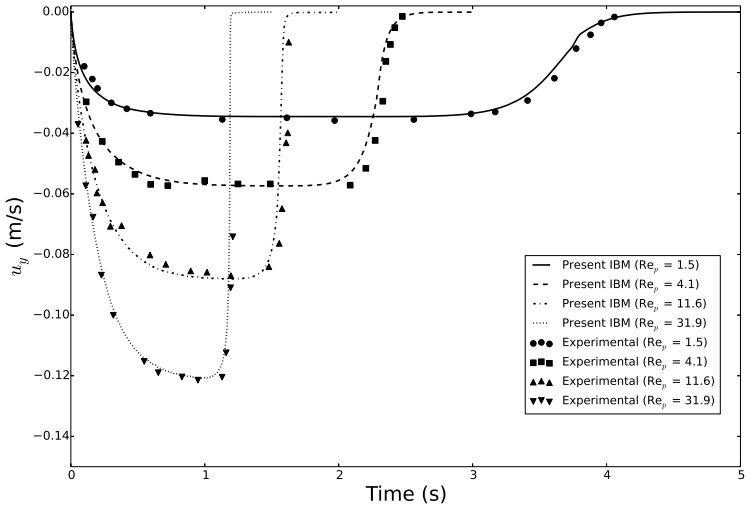
The time series data of particle location and velocity obtained by the simulations is compared with the experimental results of [44] as shown in Fig. 16. The simulation data show a very good match with the experimental data for both position and velocity. It should be emphasized that the present IBM does not require any calibration of a diameter like in [44], nor any special treatment because the density ratio is close to one like in [20]. Moreover, the grid resolution required for the present IBM ($D_p/h = 15$) is much lower than the one used by [20] ($D_p/h \approx 40$).

4.2.6. ***Sedimentation of a buoyant particle ($Re_p > 35$)***

This test case is similar to the previous one except that the Reynolds number is higher. It should be noted that the accuracy of the sharp-interface



(a)



(b)

Figure 16: Particle location(a) and velocity (b) for sedimenting buoyant sphere compared with the experimental data of Ten Cate et al. [44].

IBM for freely moving particles at higher Reynolds number has not been tested in previous studies. The simulation results obtained with the present IBM are compared with the experimental data of Mordant and Pinton [46]. The simulation parameters are chosen to have the same non-dimensional numbers (Fr, ρ_p/ρ_f, Re_p) as the experiments [46]. This approach is similar to the work of [19, 20].

$$Re_p = \frac{u_p D_p}{\nu_f} \tag{57}$$

$$Fr = \frac{u_p}{\sqrt{|\mathbf{g}| D_p}} \tag{58}$$

The simulation parameters for three different cases is presented in Table 7. The dimensions of the container for Case 1 and 3 are 11.0 m $\times$ 1.28 m $\times$ 1.28 m whereas for Case 2 it is 22.0 m $\times$ 1.28 m $\times$ 1.28 m. The walls of the container are treated with free-slip boundary conditions. A particle of diameter ($D_p = 0.167$ m) is released from rest at t = 0 with the gravitation acceleration of 9.81 m/s^2 in the x-direction. The particle is resolved with $D_p/h = 16.7$. The density of the fluid is 1000 kg/m^3 and the variable fluid and particle properties are given in Table 7.

It is observed that for the higher Reynolds number cases (Case 2 and 3), the higher order interpolation (like QUICK, min-mod etc.) for convection terms cause the oscillations in the particle motion and significant drifting of a particle to the one side of the wall. The oscillations in the force values are not observed for the test cases with the static rotating/non-rotating particle.

Although the higher order convection interpolation is stable for low Reynolds number case (Case 1), first order upwind scheme is used in the present IBM for all the moving particle simulations for the stability purpose. The reason for the oscillations could be because of not using the interpolation near the solid particle for convection terms. However, the detailed investigation of this issue is left as a future extension of this method.

Table 7: Simulation parameters for a single sedimenting sphere ($Re_p > 35$).

Parameter	Case 1	Case 2	Case 3
ν_f	0.00543	0.00268	0.00104
ρ_p/ρ_f	2.56	7.71	2.56
$\mathbf{g}$		$(9.81, 0, 0)$	

A comparison of our numerical results with the experimental results of Mordant and Pinton [46] shows an excellent agreement (Fig. 17) for all cases. The presented plots are for CFL = 0.05. However, Table 8 shows that acceptable results can be found also for higher CFL values. The deviation in the numerical results for lower and higher CFL is within 2% whereas the maximum error compared to the experiments is 3%. Moreover, the simulation with the refined grid ($D_p/h = 25.05$) is performed for Case 3 and the deviation in terminal velocity with the coarse grid is within 2%. This test case confirms that the present IBM is able to predict the dynamic behaviour of freely moving particles quite well for a wide range of Reynolds numbers. Moreover, it is important to note that accurate results can be obtained with a coarse grid and relatively high CFL values for higher Reynolds numbers as

well.

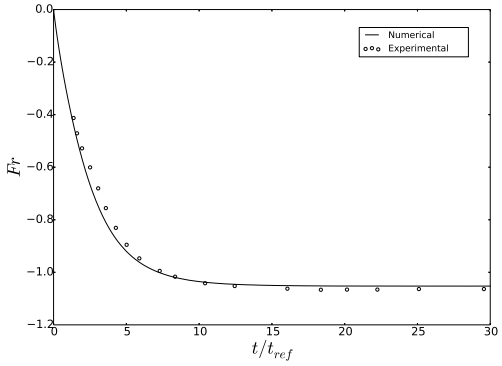
Table 8: Results for dimensionless terminal velocity (Fr) for different CFL values compared to experiments.

Case #	CFL = 0.05	CFL = 0.1	CFL = 0.2	CFL = 0.4	Experiments [46]
1	1.06	1.05	1.06	1.08	1.06 ± 0.01
2	3.53	3.47	3.46	3.48	3.56 ± 0.03
3	1.79	1.76 ($D_p/h = 16.7$) 1.78 ($D_p/h = 25.05$)	1.76	1.76	1.80 ± 0.01

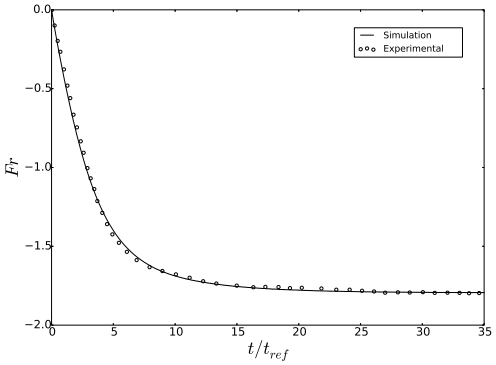
To test the suitability of the current method for lighter than fluid particles, simulation of the rise of light particles in a viscous fluid is performed. The fluid properties and particle size are kept the same as for case 1 of the validation with Mordant and Pinton [46]. The results are presented in Fig. 18. It can be observed that the simulations are stable down to $\rho_p/\rho_f = 0.3$. Below this value oscillations in the velocity field are observed. These oscillations arise due to the explicit coupling of the fluid and solid phase. Although the current method does not suffer from the strict restrictions on the density ratio (ρ_p/ρ_f) as observed in the method of Uhlmann [19], a limit of $\rho_p/\rho_f \approx 0.3$ exists which is comparable to the limit reported by an improved version of Uhlmann's method [23].

4.2.7. ***Sedimentation of two equal particles***

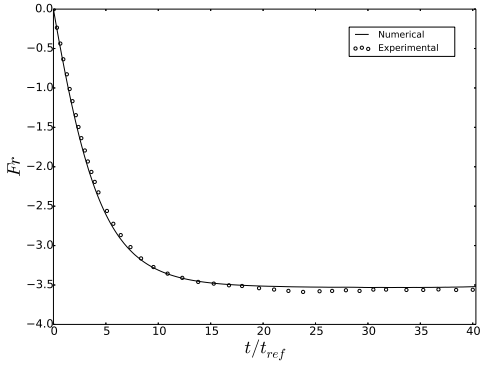
In this test case, two equal spheres are freely falling under gravity in a closed box filled with the viscous fluid. This problem is a test case for checking the stability and accuracy of a numerical method in case of inter-



(a) Case 1 ($Re_p = 41$)



(b) Case 2 ($Re_p = 280$)



(c) Case 3 ($Re_p = 360$)

Figure 17: Comparison of a dimensionless velocity of sedimenting buoyant sphere with experiments [46] ($t_{ref} = \sqrt{(D/|g|)}$).

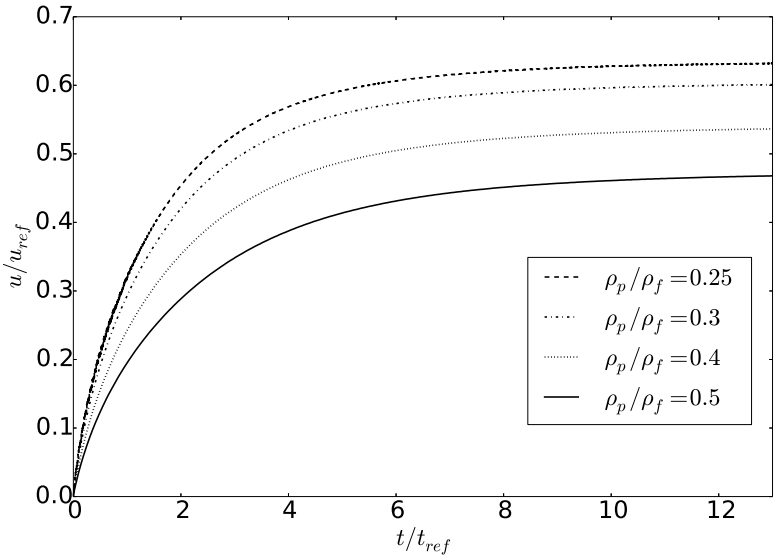
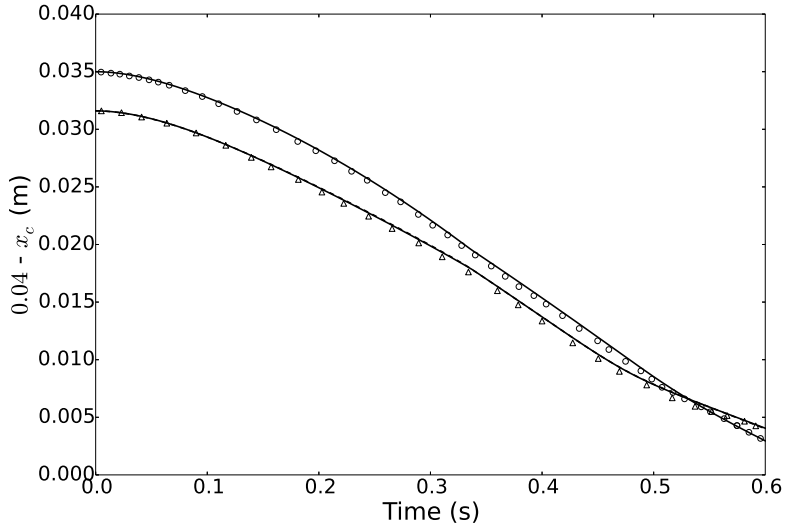


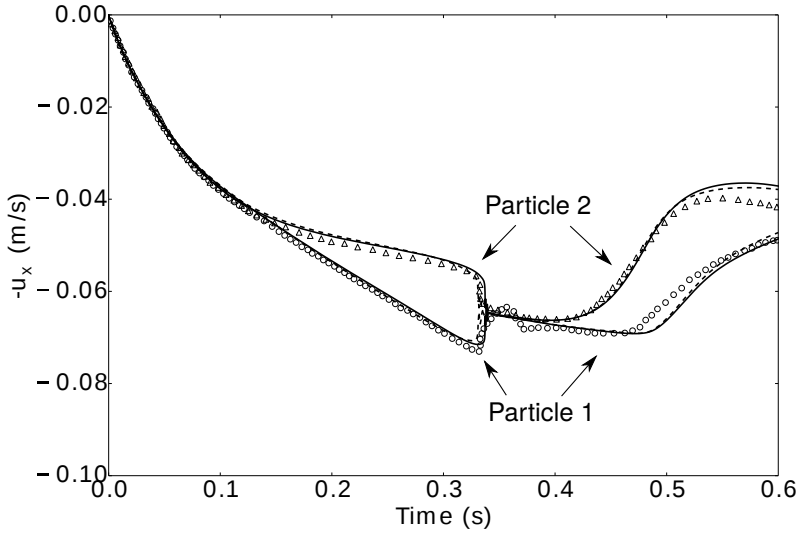
Figure 18: Rise velocity of light particles in a viscous fluid.

acting particles. This test case has been simulated previously by several researchers [23, 47, 48] to validate their numerical models. The simulation parameters used in this study are similar as in the previous studies and are presented below.

The container is of size 0.04 m $\times$ 0.01 m $\times$ 0.01 m. The gravitational acceleration is 9.81 m/s^2 in the x-direction. The fluid density (ρ_f) is 1000 kg/m^3 and dynamic viscosity (μ_f) is 10^{-3} kg/ms. The density of spherical particles (ρ_p) is 1140 kg/m^3 and the diameter (D_p) is 0.00167 m. Initially, the particles are at rest at a location $x_{c,1} = 0.005$ m and $x_{c,2} = 0.0084$ m whereas $y_{c,1} = z_{c,1} = 0.00503$ m; $y_{c,2} = z_{c,2} = 0.00497$ m. The slight offset in the y and z location of particles is introduced to instigate the drafting-kissing-tumbling phenomenon [49]. The time step is chosen such that the



(a)



(b)

Figure 19: (a) Particle distance from the bottom wall and (b)velocity for sedimentation of two equal buoyant spheres case: - - $D_p/h = 16.7$; — $D_p/h \approx 56$; $\triangle$,$\circ$Breugem [23].

CFL is 0.4 for the simulations performed in this section. The walls of the container are treated as no-slip boundaries.

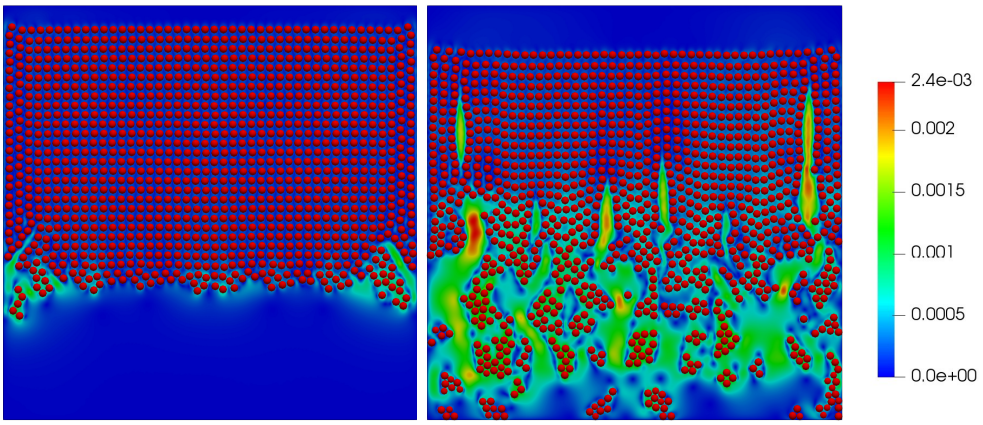
In the simulations, the soft-sphere collision model of Cundall and Strack [35] is used with the input of collision time $(t_{col}) = 8\Delta t_f$ and dry restitution of coefficient $(e_{dry}) = 0.98$. Similar to Breugem [23], the torque encountered due to collision $(\mathbf{T}_{col})$ is ignored in this test case. As mentioned earlier, when two particles approach each other with only one fluid grid cell in between, extrapolation/interpolation based on the two grid points is not possible for the drag calculation. For all such cells, no viscous or pressure drag is computed, but the sub-grid lubrication model is invoked to model the enhanced drag force in case of close approach of two particles or particle-wall.

Fig. 19 shows the evolution of position and velocity of the particles as a function of time in comparison with the results of Breugem [23]. The comparison shows a very good quantitative and qualitative agreement before the kissing phase (where particles touch each other). The results are qualitatively similar after the kissing phase but quantitative differences appear. This could be because of differences in the collision models used in the present study and that by Breugem [23]. Breugem [23] uses a collision model of Glowinski et al. [47] in which the collision force is active even before the particles actually touch each other. The results obtained with the simulations performed on two different grids ($D_p/h = 16.7$ $D_p/h \approx 56$) are also presented in Fig. 19 and do not show significant differences. This test case, furthermore, confirms the effectiveness of the present IBM on a coarser grid.

4.2.8. ***Sedimentation of 1232 spheres***

In this section, sedimentation of 1232 identical spherical particles in a narrow box is simulated using the current IBM. The fluid properties, particle size and initial particle arrangement is the same as that used by Feng and Michaelides [50]. The particles are arranged in 28 horizontal rows with each row consisting of 44 particles. The diameter of a particle is 0.625 mm and is resolved with 8 grid cells. The computational domain consists of a uniform Cartesian grid with $400 \times 400 \times 12$ grid cells. The fluid density is 1000 kg/m^3 and the viscosity is 0.001 kg/ms. The particle to fluid density ratio is 1.01. No-slip boundary conditions are used on all container walls. The difference compared to [50] is in the collision model and lubrication model used. We have implemented the soft-sphere collision model whereas [50] uses the improved version of a repulsive potential model. The collision parameters used here are: normal and tangential coefficient of restitution is 0.97 and 0.33, respectively, and the friction coefficient is 0.1 for particle-particle collisions and 0.2 for particle-wall collisions.

Initially, particles on the side of the container are found to settle faster, as observed in Fig. 20a. Later, a Rayleigh-Taylor type instability is observed which is in accordance with the literature [50, 47]. The effect is however found to be less prominent compared to the 2D simulation of Feng and Michaelides [51] where two eddies are formed in the lower part of the domain. In the current simulation, as observed in Fig. 20b, the rising fluid is observed to break through the particle array separating the neighbouring particle columns. Particles take around 60 seconds to sediment completely



(a) t = 7 s (b) t = 17 s

Figure 20: Particle positions and fluid velocity magnitude (m/s) during sedimentation process.

compared to around 55 seconds observed by [50]. This test case demonstrates that the current IBM performs quite well for multiparticle systems as well.

4.2.9. ***Fluidization in a pseudo-2D bed***

In this section, a pseudo-2D gas-solid fluidized bed is simulated using the current IBM and results are compared with the experimental and numerical results of Tang et al. [52]. Tang et al. [52] used a diffuse interface IBM with a diameter calibration. In this test case, the superficial inlet gas velocity is u_g = 2.6 m/s, which corresponds to 1.95 times u_{mf}, the minimum fluidization velocity. 5000 spherical particles are used with a normal coefficient of restitution e_n = 0.97 and a tangential coefficient of restitution e_t = 0.33. The friction coefficient is 0.1 for particle-particle collisions and 0.2 for particle-wall collisions. The remaining parameters used for the simulations and for the experiments are given in Table 9. We use identical simulation parameters

to that of Tang et al. [52].

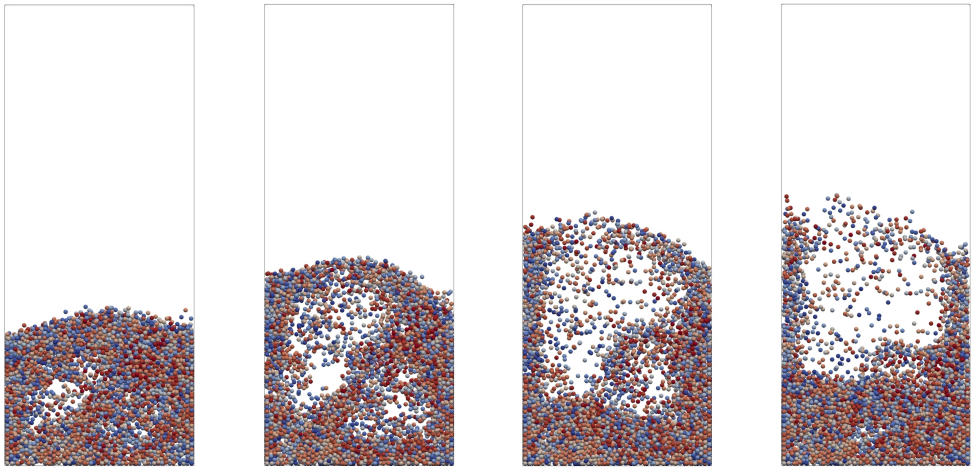


Figure 21: Snapshots of the fluidized bed obtained with an interval of 0.05 s.

It is important to mention that an imaginary wall is used for particle-wall collisions, located one particle diameter above the inlet wall. The reason is that at the inlet boundary a velocity is imposed and when a particle would collide with the inlet wall, the difference between particle and inlet velocities causes problems in obtaining a divergence-free flow field.

To quantify solids volumetric flux (ϕ), whole computational domain is divided into finite parts ($32 \times 80 \times 4$) and Eq. 59 is used to get the value of ϕ at a given location and time. $n_{p,cell}$ is total number of particles in a selected cell, V_p is volume of a particle, V_{cell} is a volume of the cell. Eq. 60 is used to get the time averaged solids volumetric flux ($\langle\phi\rangle$).

$$\phi(\mathbf{x}, t) = \frac{\sum_{i=1}^{n_{p,cell}} V_p \mathbf{u_p}}{V_{cell}} \tag{59}$$

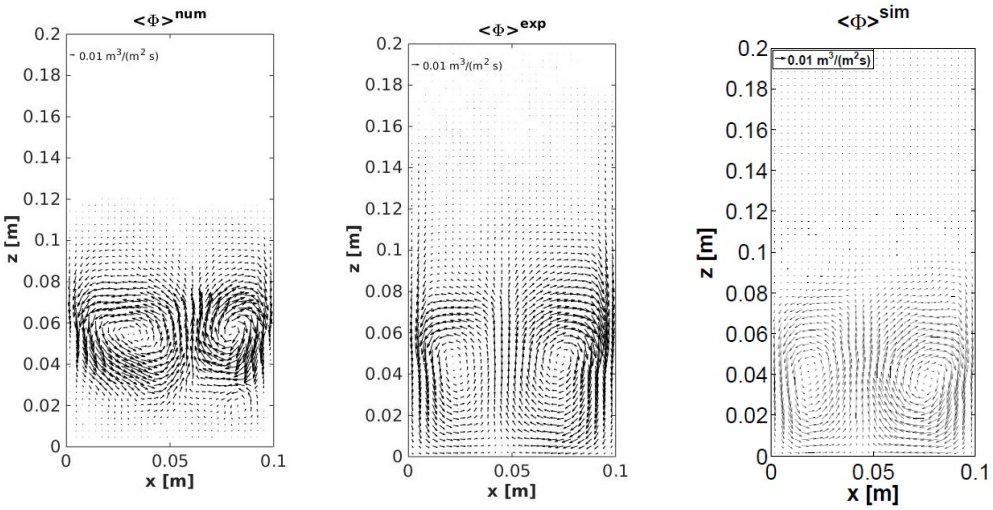
$$\langle \phi \rangle (\mathbf{x}) = \frac{\int_{t_1}^{t_2} \phi(\mathbf{x}, t) dt}{t_2 - t_1} \tag{60}$$

Table 9: Parameters used for experiments [52] and the IBM simulation of fluidized bed.

Parameters	Experiments	Simulation
Domain size	$100 \times 1000 \times 15$ mm^3	$100 \times 250 \times 15$ mm^3
ρ_f	air	1.2 kg/m^3
μ_f	air	1.8 $\times 10^{-5}$ kg/m.s
Inlet velocity, u_g	2.6 m/s	2.6 m/s
D_p	2.5 mm	2.5 mm
ρ_p	2526 kg/m^3	2526 kg/m^3
Δ	-	0.5 mm

Snapshots of the fluidized bed with an equal interval of 0.05 seconds are shown in Fig. 21. The simulations show that, initially, bubbles are created at the bottom of the bed and reach the top wall carrying particles with them. Once bubbles exit the bed, particles come down along the side walls. After the initial few bubble eruptions, the bed reaches a periodic state with a constant bed expansion height for each period. The average volumetric solids flux obtained from the simulations after reaching a quasi-steady state is compared with the numerical and experimental results [52] in Fig. 22. It shows the first 200 mm measured from the bottom of the bed. The existence of the two vortices is well predicted using the IBM simulations. However, the symmetry is not as pronounced as observed in the experimental results. This could be due to the smaller simulation time of around 4 s, compared to the 40 s of experiments. The location of the vortices is a bit higher in our

simulations than observed in the experiments but still in reasonable agreement whereas the location of vortices is a bit lower in the simulation results of Tang et al. [52].



(a) Current IBM (b) Experiment [52] (c) IBM [52]

Figure 22: Time-averaged solids volumetric flux at a central plane.

Lateral profiles of the average volumetric solids flux at different heights is compared in Fig. 23. Results for profile at $y = 0.085$ m match quite well with the experiments and provide a better prediction compared to the results of Tang et al. [52] in near wall region. For $y = 0.055$ m, results are quite accurate for the left half of the bed. However, a peak is observed in the simulations on the right side of the bed which is absent in the experiments. This could again be caused by insufficient time of averaging for the simulations since in the experiments the location of eruption of the bubble shifts from right to left and vice versa in consecutive cycles and is averaged over sufficiently many cycles.

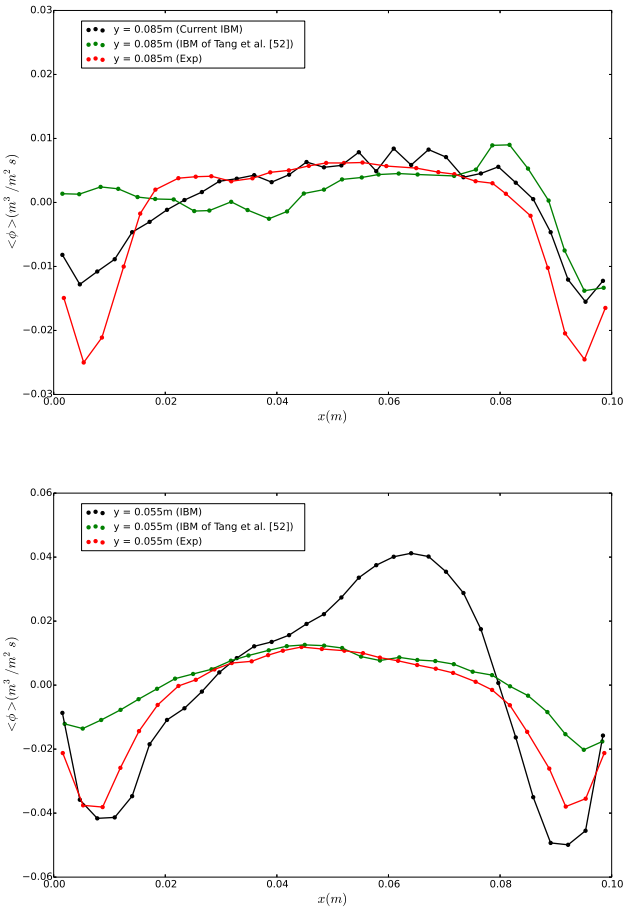


Figure 23: Profiles of axial volumetric solids flux at different bed heights

Moreover, the numerical value of $\langle\phi\rangle$ averaged over a cross-section plane at a given y-position is very close to zero indicating an overall balance of a particle motion. The results of Tang et al. [52] at $y = 0.055$ match well in the bulk region however big discrepancy is observed in the near wall region indicating a less circulation compared to experiments. Mohaghegh and Udaykumar [53] reported that the the diffuse interface IBM gives accurate results compared

to sharp interface IBM at a coarse grid resolution. This test case emphasizes that our sharp interface IBM with accurate force and torque computation based on a proposed projected area computation gives comparable results to that of diffuse interface IBM involving diameter calibration. More detailed analysis of the fluidized bed can be performed, however, the aim of the current test case is to demonstrate that reasonable accuracy can be achieved by the current IBM in reproducing important characteristics of the fluidized bed, even for a coarse resolution of $D_p/\Delta = 5$.

5. Conclusion

In the present work, the sharp-interface IBM based on a ghost cell approach of Deen et al. [5] is improved by increasing the stability and numerical accuracy of the method. The developed IBM has been verified and tested for wide range of flow problems with static as well as *freely* moving particle simulations. Along with single particle cases, multiple particle sedimentation and fluidization simulations are also performed confirming the robustness of the current method for multiparticle system. Another important aspect of the current method is that accurate results can be obtained with a coarse grid resolution, which is important for the efficient simulation of multiparticle systems that have large domain sizes.

As reported in literature, the reason for spurious oscillations in sharp interface methods has its origins in the spatial and temporal discontinuity of the flow variables and mass conservation errors. In the current method, pressure and velocity are solved over the whole domain including solid nodes.

This approach leads to spatial as well temporal continuity for the flow variables. Mass conservation automatically follows for each cell, including the cells near solid surfaces which generally are the cells where highest mass conservation problems are encountered. This approach is quite straightforward since no special treatment of a cut-cell approach or field extension approach is required. The effectiveness of the method is verified with the oscillating sphere problem where a smooth time-dependent drag force is observed.

In this work, an approach to accurately compute torques and forces on particles is presented. As a first step, projected areas are computed numerically by considering the missed cells and an excellent match is found. Next, the torque on a rotating sphere in the Stokes regime is computed and compared with its analytical solution. The original approach of Deen et al. [18] ignores the contribution of transpose terms of the velocity gradient field, which leads to a big discrepancy with the exact solution. The modified approach is found to give accurate results even for a coarse grid resolution. The results are successfully validated with the experimental results of rotating sphere case at higher Reynolds number.

Another advantage of the current method is the sharp representation of the particle surface, which avoids the requirement of any calibration based on the volume fraction and Reynolds number. Moreover, the current method works very well for lower density ratios down to $\rho_p/\rho_f \sim 0.3$ without needing any extra steps in the algorithm. Moreover, it is straightforward to implement collision models in the current IBM compared to the Uhlmann type

of methods in which the overlapping Lagrangian marker points between two particle have to be modified during collision [20]. This paper includes multi-particle simulations of Drafting-Kissing-Tumbling phenomenon, sedimentation of multiple spheres showing Rayleigh-Taylor instability and behaviour of fluidized bed. The results are found to give satisfactory match with the experimental and/or previous simulation results.

Last, we want to point out that the current method can easily be extended to include non-Newtonian fluids, polydisperse system of particles and/or non-spherical particles.

Acknowledgements

This work is supported by the programme 'Computational Sciences for Energy Research (CSER)' of the Foundation for Fundamental Research on Matter (FOM) which is now part of the Netherlands Organisation for Scientific Research Institutes (NWO-I). This research is also co-financed by Shell Global Solutions International B.V. This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.

Appendix A. Computation of projected area formed by connecting n points in 3D space

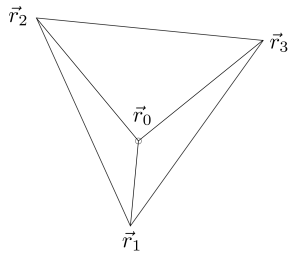


Figure A.1: Representation of a triangle.

The area vector of a triangle formed by the points ($\vec{r_1}$, $\vec{r_2}$, $\vec{r_3}$) as shown in Fig. A.1 is given as:

$$\vec{A} = \frac{1}{2}(\vec{r}_2 - \vec{r}_1) \times (\vec{r}_3 - \vec{r}_1) \tag{A.1}$$

$$\vec{A} = \frac{1}{2}(\vec{r}_1 \times \vec{r}_2 + \vec{r}_2 \times \vec{r}_3 + \vec{r}_3 \times \vec{r}_1) \tag{A.2}$$

$$\vec{A} = \frac{1}{2}\sum_{i=1}^{3}(\vec{r}_i \times \vec{r}_{i+1}) \tag{A.3}$$

If $\vec{r}_0$ is a vector representation of a random point in the space then the area of a triangle can also be computed by the summation of the triangles which has $\vec{r}_0$ as a common vertex (Eq. A.4).

$$\vec{A} = \frac{1}{2}\sum_{i=1}^{3}[(\vec{r}_i - \vec{r}_0) \times (\vec{r}_{i+1} - \vec{r}_0)] \tag{A.4}$$

Similarly, the area of a polygon with n vertices can also be computed as an addition of the areas of all triangles (Fig. A.2).

$$\vec{A} = \frac{1}{2}\sum_{i=1}^{n}[(\vec{r}_i - \vec{r}_c) \times (\vec{r}_{i+1} - \vec{r}_c)] \tag{A.5}$$

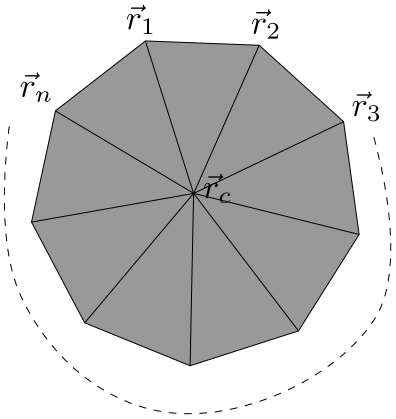


Figure A.2: Representation of a n-sided polygon.

$$\vec{A} = \frac{1}{2}\sum_{i=1}^{n}\left[(\vec{r}_i \times \vec{r}_{i+1}) - \vec{r}_c \times (\vec{r}_{i+1} - \vec{r}_i)\right] \tag{A.6}$$

$$\vec{A} = \frac{1}{2}\sum_{i=1}^{n}(\vec{r}_i \times \vec{r}_{i+1}) - \frac{1}{2}\vec{r}_c \times \cancelto{0}{\sum_{i=1}^{n}(\vec{r}_{i+1} - \vec{r}_i)} \tag{A.7}$$

$$\vec{A} = \frac{1}{2}\sum_{i=1}^{n}(\vec{r}_i \times \vec{r}_{i+1}) \tag{A.8}$$

These equations are valid for a polygon connected by straight lines. If two points are connected by curved edges, a correction term has to be added. In this derivation, it is assumed that the equation of a circle of which the given curved edge is part of, is known.

$$\vec{A}_c = \vec{A}_s + \vec{A}_{corr} \tag{A.9}$$

The corrected area as per the Fig A.3:

$$\vec{A}_{corr} = \vec{A}_{sector} - \vec{A}_{\Delta(\vec{r}_c,\vec{r}_1,\vec{r}_2)} \tag{A.10}$$

$$\left|\vec{A}_{sector}\right| = \frac{1}{2}R^2\theta \tag{A.11}$$

$$\left|\vec{A}_{\Delta(\vec{r_c},\vec{r_1},\vec{r_2})}\right| = \frac{1}{2}R^2 sin(\theta) \tag{A.12}$$

The meaning of θ is shown in Fig. A.3. Finally, we have the expression:

$$\vec{A}_{corr} = \vec{A}_{\Delta(\vec{r_c},\vec{r_1},\vec{r_2})}\left(\frac{\theta}{sin(\theta)} - 1\right) \tag{A.13}$$

Therefore,

$$\vec{A}_c = \vec{A}_s + \vec{A}_{\Delta(\vec{r_c},\vec{r_1},\vec{r_2})}\,(M-1) \tag{A.14}$$

Here, $M = \theta/sin(\theta)$

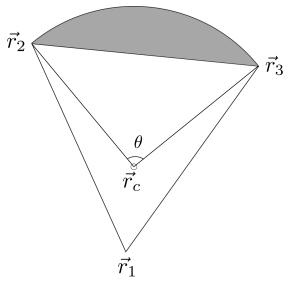


Figure A.3: Representation of a triangle with one curved side.

The general expression for the n-sided polygon with the curved edges being part of the circle with the centre $\vec{r_c}$ is:

$$\vec{A}_c = \frac{1}{2}\sum_{i=1}^{n}\left[(\vec{r}_i - \vec{r}_c)\times(\vec{r}_{i+1} - \vec{r}_c)\right] + \frac{1}{2}\sum_{i=1}^{n}(M_i - 1)\left[(\vec{r}_i - \vec{r}_c)\times(\vec{r}_{i+1} - \vec{r}_c)\right] \tag{A.15}$$

$$\vec{A}_c = \frac{1}{2}\sum_{i=1}^{n}(\vec{r}_i \times \vec{r}_{i+1}) + \frac{1}{2}\sum_{i=1}^{n}(M_i - 1)\left[(\vec{r}_i - \vec{r}_c) \times (\vec{r}_{i+1} - \vec{r}_c)\right] \tag{A.16}$$

$$\vec{A}_c = \vec{A}_s + \frac{1}{2}\sum_{i=1}^{n}(M_i - 1)\left[(\vec{r}_i - \vec{r}_c) \times (\vec{r}_{i+1} - \vec{r}_c)\right] \tag{A.17}$$

$$M_i = \begin{cases} 1 & \text{straight edges} \\ \frac{\theta}{sin(\theta)} & \text{curved edges} \end{cases}$$

Appendix B. Computational details

In our code, compute intensive part is the solution of the implicit part of the momentum equation and of the pressure Poisson equation. Assuming that the domain is cubical and N is the number of fluid cells in one direction, number of operations involved in solving fluid part will scale as $\mathcal{O}(N^3)$ since we use a linearly scalable multigrid preconditioner. For the particle phase, the drag force is computed by computing pressure and shear rate at the intersection point on the solid surface. Intersection points are obtained by traversing from a fluid node neighbouring a solid surface in a given spatial dimension. Number of intersection points(N_i) will be same as neighbouring fluid cells, $N_i \sim \mathcal{O}((D/h)^2)$. Solving particle phase for N_p particles would require $\mathcal{O}(N_p N_i)$ operations. It can be easily shown that even for a fully packed system, the fluid phase would dominate the time spent in a simulation. It should be noted that the scaling of operations required for the current method is same as that reported by Uhlmann[19].

To measure the simulation time of current method, problem of sedimentation of multiple spheres is considered. Table B.1 shows the comparison of time taken by one simulation time step for our method in comparison with

Table B.1: Simulation time for one time step for current method on 2.6 GHz Intel Xeon Processor E5-2690 v3 CPUs

Case #	$N_x \times N_y \times N_z$	N_p	nproc	t_{fluid}, s (%)	t_{solid}, s (%)	t_{total}, s	t^*, s[19]
1	$512 \times 512 \times 512$	1024	64	34.85 (90.63 %)	3.60 (9.37 %)	38.45	48.65
2	$512 \times 1024 \times 512$	1024	128	37.14 (90.98 %)	3.68 (9.02 %)	40.82	61.30
3	$512 \times 1024 \times 512$	2048	128	38.80 (91.15 %)	3.77 (8.85 %)	42.57	62.36

$N_x \times N_y \times N_z$: number of fluid cells, N_p: number of particles, nproc: number of processors, t_{fluid}: time spent in solving fluid phase, t_{solid}: time taken in solving particle phase, t_{total}: total time spent in one time step, t^*: time reported by Uhlmann[19] with scaling factor (=1.1/2.6) based on a difference in a clock frequency of processors

the method of Uhlmann[19]. We scaled the times reported by [19] based on the difference in our hardware. Although this is not really a direct comparison of methods since times will be dependent on the optimization of a code, tolerance used for solving linear system of equations and divergence criteria etc., it gives an idea of time taken by these methods on modern hardware.

In the code, fluid phase is solved in parallel using domain decomposition method. Comparison between case 1 and 2 indicates a very good weak-scaling for MPI-parallelized fluid part. Force computation on particles and solid phase flag assignment on fluid nodes is also carried out in parallel. However,particle-particle interaction is executed serially on a root processor. From Table B.1, it can be observed that the time increase in a simulation from case 2 $\rightarrow$ 3 due to increased number of particles is not significant. However, serially solved particle-particle part will limit the scalability of the current code significantly according to Amdahl's law if even more number of particles are used in the simulation.

References

[1] C. S. Peskin, Numerical analysis of blood flow in the heart, Journal of computational physics 25 (3) (1977) 220–252.

[2] C. S. Peskin, The immersed boundary method, Acta numerica 11 (2002) 479–517.

[3] G. Iaccarino, R. Verzicco, Immersed boundary technique for turbulent flow simulations, Applied Mechanics Reviews 56 (3) (2003) 331–347.

[4] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 239–261.

[5] N. G. Deen, E. A. J. F. Peters, J. T. Padding, J. A. M. Kuipers, Review of direct numerical simulation of fluid–particle mass, momentum and heat transfer in dense gas–solid flows, Chemical Engineering Science 116 (2014) 710–724.

[6] D. Goldstein, R. Handler, L. Sirovich, Modeling a no-slip flow boundary with an external force field, Journal of Computational Physics 105 (2) (1993) 354–366.

[7] E. Saiki, S. Biringen, Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method, Journal of Computational Physics 123 (2) (1996) 450–465.

[8] J. Mohd-Yusof, Combined immersed boudaries/ b-splines methods for simulations in complex geometries, ctr annual research briefs, nasa ames (1997).

[9] E. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, Journal of computational physics 161 (1) (2000) 35–60.

[10] J. Kim, D. Kim, H. Choi, An immersed-boundary finite-volume method for simulations of flow in complex geometries, Journal of Computational Physics 171 (1) (2001) 132–150.

[11] S. Majumdar, G. Iaccarino, P. Durbin, Rans solvers with adaptive structured boundary non-conforming grids, Annual Research Briefs, Center for Turbulence Research, Stanford University (2001) 353–466.

[12] Y.-H. Tseng, J. H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, Journal of computational physics 192 (2) (2003) 593–623.

[13] S. De, S. Das, J. A. M. Kuipers, E. A. J. F. Peters, J. T. Padding, A coupled finite volume immersed boundary method for simulating 3d viscoelastic flows in complex geometries, Journal of Non-Newtonian Fluid Mechanics 232 (2016) 67–76.

[14] S. Das, N. G. Deen, J. A. M. Kuipers, Immersed boundary method (ibm) based direct numerical simulation of open-cell solid foams: Hydrodynamics, AIChE Journal 63 (3) (2017) 1152–1173.

[15] H. V. Patel, S. Das, J. A. M. Kuipers, J. T. Padding, E. A. J. F. Peters, A coupled volume of fluid and immersed boundary method for simulating 3d multiphase flows with contact line dynamics in complex geometries, Chemical Engineering Science 166 (2017) 28–41.

[16] H. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface cartesian grid method for simulating flows with complex moving boundaries, Journal of Computational Physics 174 (1) (2001) 345–380.

[17] R. Mittal, H. Dong, M. Bozkurttas, F. Najjar, A. Vargas, A. von Loebbecke, A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries, Journal of computational physics 227 (10) (2008) 4825–4852.

[18] N. G. Deen, S. H. Kriebitzsch, M. A. van der Hoef, J. Kuipers, Direct numerical simulation of flow and heat transfer in dense fluid–particle systems, Chemical Engineering Science 81 (2012) 329–344.

[19] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, Journal of Computational Physics 209 (2) (2005) 448–476.

[20] T. Kempe, J. Fröhlich, An improved immersed boundary method with direct forcing for the simulation of particle laden flows, Journal of Computational Physics 231 (9) (2012) 3663–3684.

[21] S. Schwarz, T. Kempe, J. Fröhlich, A temporal discretization scheme to compute the motion of light particles in viscous flows by an immersed boundary method, Journal of Computational Physics 281 (2015) 591–613.

[22] S. Tschisgale, T. Kempe, J. Fröhlich, A non-iterative immersed boundary method for spherical particles of arbitrary density ratio, Journal of Computational Physics 339 (2017) 432–452.

[23] W.-P. Breugem, A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows, Journal of Computational Physics 231 (13) (2012) 4469–4498.

[24] Z. Yu, X. Shao, A direct-forcing fictitious domain method for particulate flows, Journal of computational physics 227 (1) (2007) 292–314.

[25] Y. Tang, S. Kriebitzsch, E. Peters, M. Van der Hoef, J. Kuipers, A methodology for highly accurate results of direct numerical simulations: drag force in dense gas–solid flows at intermediate reynolds number, International journal of multiphase flow 62 (2014) 73–86.

[26] K. Luo, Z. Wang, J. Fan, K. Cen, Full-scale solutions to particle-laden flows: Multidirect forcing and immersed boundary method, Physical Review E 76 (6) (2007) 066709.

[27] S. Kriebitzsch, M. Van der Hoef, H. Kuipers, Direct numerical simulations of gas-particle flows using an immersed boundary method, in: Proceedings of the Academy Colloquium Immersed Boundary Methods: Current Status and Future Research Directions (KNAW, Amsterdam, The Netherlands, 15–17 June 2009), 2010.

[28] A. Wachs, A. Hammouti, G. Vinay, M. Rahmani, Accuracy of finite volume/staggered grid distributed lagrange multiplier/fictitious domain simulations of particulate flows, Computers & Fluids 115 (2015) 154–172.

[29] J. Lee, J. Kim, H. Choi, K.-S. Yang, Sources of spurious force oscillations

from an immersed boundary method for moving-body problems, Journal of computational physics 230 (7) (2011) 2677–2695.

[30] J. H. Seo, R. Mittal, A sharp-interface immersed boundary method with improved mass conservation and reduced spurious pressure oscillations, Journal of computational physics 230 (19) (2011) 7347–7363.

[31] J. Yang, E. Balaras, An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries, Journal of Computational Physics 215 (1) (2006) 12–40.

[32] J. Yang, F. Stern, A simple and efficient direct forcing immersed boundary framework for fluid–structure interactions, Journal of Computational Physics 231 (15) (2012) 5029–5061.

[33] J. Lee, D. You, An implicit ghost-cell immersed boundary method for simulations of moving body problems with control of spurious force oscillations, Journal of Computational Physics 233 (2013) 295–314.

[34] R. D. Falgout, U. M. Yang, hypre: A library of high performance preconditioners, in: International Conference on Computational Science, Springer, 2002, pp. 632–641.

[35] P. A. Cundall, O. D. Strack, A discrete numerical model for granular assemblies, geotechnique 29 (1) (1979) 47–65.

[36] W.-P. Breugem, A combined soft-sphere collision/immersed boundary method for resolved simulations of particulate flows, in: Proceedings of the ASME, 2010, p. 11.

[37] R. B. Bird, W. E. Stewart, E. N. Lightfoot, Transport phenomena, John Wiley & Sons, 2007.

[38] M. Baltussen, Bubbles on the cutting edge: direct numerical simulations of gas-liquid-solid three-phase flows, Ph.D. thesis, Eindhoven University of Technology (2015).

[39] A. Vreman, A staggered overset grid method for resolved simulation of incompressible flow around moving spheres, Journal of Computational Physics 333 (2017) 269–296.

[40] A. Zick, G. Homsy, Stokes flow through periodic arrays of spheres, Journal of fluid mechanics 115 (1982) 13–26.

[41] P. Bagchi, S. Balachandar, Effect of free rotation on the motion of a solid sphere in linear shear flow at moderate re, Physics of Fluids (1994-present) 14 (8) (2002) 2719–2737.

[42] O. Sawatzki, Das strömungsfeld um eine rotierende kugel, Acta Mechanica 9 (3-4) (1970) 159–214.

[43] S. Dennis, S. Singh, D. Ingham, The steady flow due to a rotating sphere at low and moderate reynolds numbers, Journal of Fluid Mechanics 101 (2) (1980) 257–279.

[44] A. Ten Cate, C. Nieuwstad, J. Derksen, H. Van den Akker, Particle imaging velocimetry experiments and lattice-boltzmann simulations on a single sphere settling under gravity, Physics of Fluids (1994-present) 14 (11) (2002) 4012–4025.

[45] S. V. Apte, M. Martin, N. A. Patankar, A numerical method for fully resolved simulation (frs) of rigid particle–flow interactions in complex flows, Journal of Computational Physics 228 (8) (2009) 2712–2738.

[46] N. Mordant, J.-F. Pinton, Velocity measurement of a settling sphere, The European Physical Journal B-Condensed Matter and Complex Systems 18 (2) (2000) 343–352.

[47] R. Glowinski, T. Pan, T. Hesla, D. Joseph, J. Periaux, A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: application to particulate flow, Journal of Computational Physics 169 (2) (2001) 363–426.

[48] N. Sharma, N. A. Patankar, A fast computation technique for the direct numerical simulation of rigid particulate flows, Journal of Computational Physics 205 (2) (2005) 439–457.

[49] A. F. Fortes, D. D. Joseph, T. S. Lundgren, Nonlinear mechanics of fluidization of beds of spherical particles, Journal of Fluid Mechanics 177 (1987) 467–483.

[50] Z.-G. Feng, E. E. Michaelides, Proteus: a direct forcing method in the simulations of particulate flows, Journal of Computational Physics 202 (1) (2005) 20–51.

[51] Z.-G. Feng, E. E. Michaelides, The immersed boundary-lattice boltzmann method for solving fluid–particles interaction problems, Journal of Computational Physics 195 (2) (2004) 602–628.

[52] Y. Tang, Y. Lau, N. Deen, E. Peters, J. Kuipers, Direct numerical simulations and experiments of a pseudo-2d gas-fluidized bed, Chemical Engineering Science 143 (2016) 166–180.

[53] F. Mohaghegh, H. Udaykumar, Comparison of sharp and smoothed interface methods for simulation of particulate flows i: Fluid structure interaction for moderate reynolds numbers, Computers & Fluids 140 (2016) 39–58.