Computers and Fluids 192 (2019) 104233

Contents lists available at ScienceDirect

Computers and Fluids

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

A comparative study of immersed boundary method and interpolated bounce-back scheme for no-slip boundary treatment in the lattice Boltzmann method: Part I, laminar flows

Cheng Peng[b,*], Orlando M. Ayala[c], Lian-Ping Wang[a,b]

[a] Department of Mechanics and Aerospace Engineering, Southern University of Science and Technology, Shenzhen 518055, Guangdong, China
[b] 126 Spencer Lab, Department of Mechanical Engineering, University of Delaware, Newark, DE, 19716 USA
[c] 111A Kaufman Hall, Department of Engineering Technology, Old Dominion University, Norfolk, VA, 23529, USA

ARTICLE INFO

Article history:
Received 30 October 2018
Revised 5 February 2019
Accepted 28 June 2019
Available online 29 June 2019

Keywords:
Lattice Boltzmann method
Interpolated bounce-back schemes
Immersed boundary methods
No-slip boundary

ABSTRACT

The interpolated bounce-back schemes and the immersed boundary method are the two most popular algorithms in treating a no-slip boundary on curved surfaces in the lattice Boltzmann method. While those algorithms are frequently implemented in the numerical simulations involving complex geometries, such as particle-laden flows, their performances are seldom compared systematically over the same local quantities within the same context. In this paper, we present a systematic comparative investigation on some frequently used and most state-of-the-art interpolated bounce-back schemes and immersed boundary methods, based on both theoretical analyses and numerical simulations of four selected 2D and 3D laminar flow problems. Our analyses show that immersed boundary methods (IBM) typically yield a first-order accuracy when the regularized delta-function is employed to interpolate velocity from the Eulerian to Lagrangian mesh, and the resulting boundary force back to the Eulerian mesh. This first order in accuracy for IBM is observed for both the local velocity and hydrodynamic force/torque, apparently different from the second-order accuracy sometime claimed in the literature. Another problem of immersed boundary methods is that the local stress within the diffused fluid-solid interface tends to be significantly underestimated. On the other hand, the interpolated bounce-back generally possesses a second-order accuracy for velocity, hydrodynamic force/torque, and local stress field. The main disadvantage of the interpolated bounce-back schemes is its higher level of fluctuations in the calculated hydrodynamic force/torque when a solid object moves across the grid lines. General guidelines are also provided for the necessary grid resolutions in the two approaches in order to accurately simulate flows over a solid particle.

1. Introduction

Over the last thirty years, the lattice Boltzmann method (LBM) has been actively developed and has become a reliable tool for simulating flow problems with complex geometries, such as flow in porous media [1], fluid structure interaction [2] and particle-laden turbulent flows [3,4]. In these applications, the treatment of the no-slip boundary condition at the fluid-solid interfaces is often an important issue that affects the overall accuracy, numerical stability, and computational efficiency of the lattice Boltzmann method.

As a mesoscopic method based on the Boltzmann equation but with the goal to solve the macroscopic Navier-Stokes equations, the treatment of a no-slip boundary within LBM can be flexible as either the no-slip boundary treatment schemes used in conventional computational fluid dynamics (CFD) or the microscopic properties in the Boltzmann equation may be applied and implemented. There are mainly two categories of no-slip boundary treatment in LBM simulations. The first is the immersed boundary method (IBM). IBM is a popular no-slip boundary treatment developed in conventional CFD [5–7], but it can be easily incorporated within LBM algorithms [8,9]. The idea of IBM is to represent the effect of the no-slip condition as a boundary force applying to the neighboring region of the fluid-solid interface. In order to ensure that the no-slip condition is enforced at precisely the location of the boundary, a body-fitted Lagrangian grid is usually attached to the surface of each solid object besides the Eulerian grid covering

* Corresponding author.
E-mail addresses: cpengxpp@udel.edu (C. Peng), oayala@odu.edu (O.M. Ayala), lwang@udel.edu (L.-P. Wang).

https://doi.org/10.1016/j.compfluid.2019.06.032
0045-7930/

the whole computational domain. A regularized delta function is employed to interpolate information between the Eulerian and Lagrangian grids [5,6]. Depending on how the boundary force that enforces the no-slip condition is calculated, IBM can be grouped as penalty IBM [10] or direct-forcing IBM [11]. For problems involving only non-deformable rigid surfaces, direct-forcing IBM is preferred due to its clearer physical picture and better numerical stability.

The second category of no-slip boundary treatment in LBM is to directly construct the unknown distribution functions at the boundary nodes using the known ones while observing the hydrodynamic constraints. This type of algorithm is known as bounce-back schemes. The early bounce-back scheme such as that proposed by Ladd [12] approximates a curved surface as a staircase shaped polylines when applied to a complex geometry. The improved bounce-back schemes were developed later to address this deficiency [13–17]. While the detailed algorithms are not unique, the idea of these improved schemes are similar, which is to construct the unknown distribution functions to have at least a second-order accuracy. These schemes are typically referred to as the interpolated bounce-back (IBB) schemes. It is known that the hydrodynamic equations can be obtained from the Chapman-Enskog expansion of the Boltzmann equation, however, it is not completely clear whether IBB schemes are consistent with the Chapman-Enskog expansion at the boundary nodes. The accuracy and numerical stability of IBB schemes are typically examined only by numerical tests.

In the past, both IBM and IBB were extensively used by the LBM community in a wide range of applications. Although each method is validated in a few numerical tests on its own, systematic comparative studies between the two sets of methods are rare. Peng & Luo [18] compared performances of Bouzidi et al.'s quadratic IBB scheme [13] and Feng & Michaelides's direct-forcing IBM-LBM [8], focusing on evaluating the drag and lift coefficients of a cylinder placed at different location facing a uniform stream. They observed that while the numerical error in the integrated force evaluation generally followed a second-order convergence rate, the results from IBB scheme are much more accurate than those from IBM-LBM. As will become clearer later with the present work, although in certain cases the hydrodynamic force/torque evaluation does possess a second-order accuracy, such observation may not be generalized for arbitrary flows. Chen et al. [19] compared a few IBB schemes and IBM-LBM algorithms in simulating the acoustic waves scattering on static and moving cylinder surfaces. They reported that while IBB schemes outperformed in accuracy in static cylinder cases, IBM-LBM could be a better choice in cases with moving objects in terms of suppressing the high-frequent fluctuations (*i.e.*, the grid jitter problem) associated with objects crossing the grid mesh lines.

While these previous comparative studies are useful, a re-examination of the inter-comparison of the two treatments is still necessary, for several reasons. First, in the aforementioned studies, the benchmark results used as standards are usually from other simulations, rather than from the theory. This brings difficulty to rigorously gauge the accuracy of a method. For example, in the study of Peng & Luo [18], as will be shown, IBM-LBM method is of only first-order accuracy; it remains a puzzle that the first-order accurate IBM-LBM could lead to second-order converged drag and lift force evaluations. In many validation studies of IBM, the Taylor-Green flow without a solid-fluid interface was employed [6,20]. This validation is not so meaningful since the accurate flow field can be obtained with or without the IBM. Second, it is important to follow the recent developments in both categories of methods in order to make unbiased conclusions. For example, Breugem [7] proposed an improved IBM by retracting the locations of the Lagrangian grid points from the surface of a solid object towards the interior of the solid object. It is claimed this retraction could improve the accuracy of IBM from first-order to second-order. Zhou & Fan [21] incorporated this improvement to LBM that seemed to reach a similar conclusion. On the other hand, IBB schemes are also under further developments. A good example is the single-node second-order accurate IBB scheme by Zhao & Yong [22], which allows the second-order accurate no-slip boundary to be realized using the information only on the boundary node itself. This scheme is particularly useful for cases such as dense particle suspension where the gap between two solid surfaces is too narrow for other IBB schemes to be executed. Whether these new developments would alter the conclusions made in the previous comparative studies is yet to be examined.

In this paper, we examine the performance of several selected IBM algorithms and IBB schemes in flows with reliable benchmark results. Those IBM algorithms and IBB schemes are chosen because they have been implemented in complex simulations such as direct numerical simulations of particle-laden turbulent flows [3,4,23–25]. In order to assess the reliability of the reported results, it is important to test the accuracy and robustness of these methods in relatively simpler laminar flows that are easier to analyze. The rest of the paper is arranged as the following. In Section 2, we briefly introduce LBM and the selected IBB schemes and IBM algorithms to be examined. Then, the performances of these no-slip boundary treatments are compared in some carefully chosen two-dimensional and three-dimensional laminar flow tests in Section 3. Finally, the key observations will be summarized in Section 4.

2. The lattice Boltzmann method and its no-slip boundary treatments

The evolution equation of LBM can be viewed as a fully discrete form of the Boltzmann Bhatnagar—Gross—Krook (BGK) equation in space and time, with a selected set of particle velocities

$$f_i(\mathbf{x}+\mathbf{e}_i\delta_t, t+\delta_t) - f_i(\mathbf{x},t) = -\frac{1}{\tau}\left[f_i(\mathbf{x},t) - f_i^{(eq)}(\mathbf{x},t)\right] + F_i(\mathbf{x},t), \tag{1}$$

where f_i is the particle distribution function for the discrete velocity $\mathbf{e}_i$, $\mathbf{x}$ and t are the spatial coordinate and time, respectively. δ_t is the time step size, $f_i^{(eq)}$ is the equilibrium distribution of f_i, F_i is the term representing the body force in the Boltzmann equation. τ is the non-dimensional relaxation time, which is related to the kinematic viscosity ν as

$$\nu = (\tau - 0.5)c_s^2\delta_t, \tag{2}$$

with c_s being the speed of sound.

Eq. (1) is known as the lattice BGK equation, whose collision operator (right-hand side of Eq. (1)) contains only one relaxation time τ. Alternatively, if the collision operator is constructed in the moment space through linear transformation, different moments can be relaxed at different rates, the evolution equation of LBM can then be expressed as

$$\mathbf{f}(\mathbf{x}, \mathbf{e}_i\delta_t, t+\delta_t) - \mathbf{f}(\mathbf{x},t) = -\mathbf{M}^{-1}\mathbf{S}\left[\mathbf{m}(\mathbf{x},t) - \mathbf{m}^{(eq)}(\mathbf{x},t)\right] + \mathbf{M}^{-1}\boldsymbol{\Psi}(\mathbf{x},t). \tag{3}$$

which possesses larger flexibility in the model design. $\mathbf{f}$ is the vector expression of f_i. $\mathbf{m}$, $\mathbf{m}^{(eq)}$, and $\boldsymbol{\Psi}$ are the moment vector, equilibrium moment vector, and the forcing vector, respectively. $\mathbf{M}$ is the transform matrix that relates the moment vector and vector of distribution functions as $\mathbf{m} = \mathbf{M}\mathbf{f}$ and $\mathbf{f} = \mathbf{M}^{-1}\mathbf{m}$. LBM using Eq. (3) as the evolution equation is known as the multi-relaxation time (MRT) LBM. More details regarding Eqs. (1) and (3) can be found in the textbooks [26] and other classic articles of LBM [27,28], thus they are not repeated here.

2.1. Immersed boundary-lattice Boltzmann method

The standard LBM can be viewed as a mesoscopic alternative of the incompressible Navier-Stokes solver in the weakly compressible limit. The no-slip boundary treatments in conventional CFD may be incorporated in LBM. The most popular method that has been used widely in CFD for the no-slip boundary treatment on arbitrarily shaped surface is the immersed boundary method. The first incorporation of IBM into LBM has been achieved by Feng & Michaelides [29]. Since then, there have been many variations of the method in terms of the calculation the boundary force and the incorporation of this force into the evolution equation of LBM. The boundary force in IBM-LBM can be calculated by the penalty feedback forcing [29], direct forcing [8], and momentum exchange forcing [30]. Among these three force calculation methods, the direct forcing is the most popular one due to its simplicity and the capability to use larger CFL numbers [6]. The direct-forcing IBM has been made particularly efficient to realize the no-slip condition on rigid particle surfaces in particle-laden flows [4,23,24]. In this study, we focus our attention on the evaluation of direct-forcing IBM algorithms that has been frequently used in the three-dimensional flow simulations with a large number of particles. In these algorithms, two sets of grids, a fixed Eulerian grid is used to store the information of the flow field, and a Lagrangian grid attached to the solid surface is used to ensure the no-slip condition is enforced precisely on the physical location. Uhlmann significantly simplified the algorithm of direct-forcing IBM as five key steps [6]. First, the known velocity field stored at the Eulerian grid $\mathbf{u}^n$ is evolved to a temporary velocity field $\tilde{\mathbf{u}}$ by solving the N-S equations without considering the boundary force.

$$\rho\frac{\tilde{\mathbf{u}}-\mathbf{u}^n}{\delta t}=-\rho(\mathbf{u}\cdot\nabla)\mathbf{u}-\nabla p+\mu\nabla^2\mathbf{u}. \tag{4}$$

Next, this temporary velocity field at the Eulerian grid $\mathbf{x}$ is interpolated to the Lagrangian grid $\mathbf{X}$.

$$\tilde{\mathbf{U}}(\mathbf{X})=\sum_{\mathbf{x}}\tilde{\mathbf{u}}(\mathbf{x})\delta_h(\mathbf{x}-\mathbf{X})h^3, \tag{5}$$

where δ_h is the interpolation kernel which typically has a form of the regularized delta function [5]. By default, the four-point delta-function [5]

$$\delta_h=\frac{1}{h^3}\phi\left(\frac{x_1}{h}\right)\phi\left(\frac{x_2}{h}\right)\phi\left(\frac{x_3}{h}\right),$$
$$\phi(r)=\begin{cases}0, & |r|\geq 2,\\ \frac{1}{8}\left(5-2|r|-\sqrt{-7+12|r|-4r^2}\right), & 1\leq|r|<2\\ \frac{1}{8}\left(3-2|r|+\sqrt{1+4|r|-4r^2}\right), & 0\leq|r|<1,\end{cases} \tag{6}$$

derived by Peskin is used for all the simulations presented below, unless specified otherwise. h^3 is the volume of a Eulerian grid cell. By default, we use the uppercase letters to represent the properties on the Lagrangian grid and the lowercase letters to represent the properties on the Eulerian grid. Next, the boundary force $\mathbf{F}(\mathbf{X})$ used to enforce the no-slip condition on the Lagrangian grid should be calculated as

$$\mathbf{F}(\mathbf{X})=\frac{\mathbf{U}^d(\mathbf{X})-\tilde{\mathbf{U}}(\mathbf{X})}{\delta_t}. \tag{7}$$

Then, this boundary force is distributed back to the Eulerian grid.

$$\mathbf{f}(\mathbf{x})=\sum_{\mathbf{X}}\mathbf{F}(\mathbf{X})\delta_h(\mathbf{x}-\mathbf{X})\Delta V, \tag{8}$$

where ΔV is the control volume of a Lagrangian grid, which is typically chosen as $\Delta V\approx h^3$ [6]. At last, the obtained force field is used to update the velocity field from $\tilde{\mathbf{u}}$ to $\mathbf{u}^{n+1}$.

$$\mathbf{u}^{n+1}=\tilde{\mathbf{u}}+\mathbf{f}\delta_t, \tag{9}$$

Eq. (4) to Eq. (9) well describe a direct-forcing IBM algorithm in CFD. There are different ways to incorporate the above algorithm into the frame of LBM [9,20,31,32]. Since the interpolation via delta function only has a first-order accuracy on a general fluid-solid surface (which will be proven later) [5,32], the choice of a specific algorithm may not affect the accuracy of the simulation results that much. Of course, it is more reasonable to use the mesoscopic forcing terms in the evolution equations of LBM, *i.e.*, F_i in Eq. (1) or $\mathbf{\Psi}$ in Eq. (3), which ensures a second-order accuracy when applied to a non-uniform force field, as the boundary force redistributed back to the Eulerian grid is a non-uniform force field. When the lattice BGK equation is employed, both Guo's scheme [28] and Cheng & Li's scheme [33] possess the second-order accuracy when applied to a non-uniform force field. These two schemes are actually identical (proven in [20]). When the MRT-LBM equation is used, the forcing term can be constructed using the inverse design, as demonstrated in [34].

In the direct-forcing IBM algorithm described in Eq. (4) to Eq. (9), the boundary force is defined as a correction force that brings the fluid velocity to target one at the next time step $n+1$. The IB-LBM algorithm that corresponds to this algorithm is the implicit velocity correction based IB-LBM developed by Wu & Shu [9]. In this algorithm, Guo's forcing scheme [28] is used. Kang & Hassan [20] developed a similar algorithm using Cheng & Li's forcing scheme [33]. The only concern about these correction-based IB-LBM algorithms is whether they are fully consistent with the Chapman-Enskog expansion. When Guo's forcing scheme is used, half of the force is added when calculating the velocity field from the distribution functions [28]. However, in the correction based IB-LBM, this half force is absent in order to calculate an "unforced" velocity field. The same issue can be identified in Kang & Hassan's algorithm with Cheng & Li's forcing scheme. The implicit force field that should be added right after the propagation of the distribution functions is postponed after the update of the hydrodynamic properties (density, velocity, etc.) as a correction [20]. A more consistent algorithm of the correction-based IB-LBM may be the one developed by Zhang et al. [32] recently. In this algorithm, the implicit force field added after the propagation of the distribution functions is obtained through iterations [32]. However, in simulations with a large number of solid surfaces, the iteration is usually undesired.

The specific IB-LBM algorithm we examine in this paper is a relatively simple one. At each step, prior to the evolution of distribution functions, the boundary force is fist calculated as Eq. (7). This boundary force is then distributed to the Eulerian grid, and used for evolving the distribution functions according to Eq. (1) or Eq. (3). The boundary force in this algorithm is therefore a force responding to the presence of a solid force at the current time, rather than a force that enforce the no-slip condition at the next time step.

2.2. Interpolated bounce-back schemes

The essence of bounce-back schemes is to directly construct the unknown distribution functions from the known ones and the hydrodynamic constraints at the boundary nodes. With the boundary configuration in Fig. 1, a simple bounce-back scheme can be written as [12]

$$f_i\left(\mathbf{x}_f,t+\delta_t\right)=f_{\bar{i}}^*\left(\mathbf{x}_f,t\right)+2\rho_0 w_i\frac{\mathbf{e}_i\cdot\mathbf{u}_w}{c_s^2}, \tag{10}$$

where $f_i\left(\mathbf{x}_f,t+\delta_t\right)$ and $f_{\bar{i}}^*\left(\mathbf{x}_f,t\right)$ are the bounce-back distribution function and the incident distribution function, both locate at the boundary node $\mathbf{x}_f$ and with $\mathbf{e}_i=-\mathbf{e}_{\bar{i}}$. $\mathbf{u}_w$ is the velocity at the wall location $\mathbf{x}_w$. The last term on the right-hand side is used to ensure the no-slip condition when the solid boundary is moving. Eq. (10) means that the post-collision particles traveling towards a

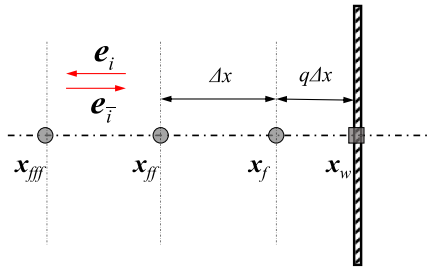

Fig. 1. A sketch of a fluid-solid interface in a LBM simulation.

wall return back along the same location after bouncing back from the wall, thus the scheme obtained its name. Since a distribution function travels precisely one grid spacing from t to $t+\delta_t$, particles start from $\mathbf{x}_f$ can end precisely at the same location only when $\mathbf{x}_f$ is half a grid spacing from the wall. In fact, when this condition is not satisfied, the bounce-back scheme of Eq. (10) only has a first-order accuracy, which restricts its application on an arbitrarily shaped surface.

In order to ensure that the second-order spatial accuracy in a bounce-back process for more general cases, interpolation is usually required. Since the number of unknown distribution functions is usually larger than the number of hydrodynamic constraints, strategies to design interpolated bounce-back schemes are not unique. Two representative interpolated bounce-back schemes are the conditional scheme proposed by Bouzidi et al. [13], and the unified scheme by Yu et al. [16]. In Bouzidi et al.'s scheme, when the relative distance from the boundary node point to the wall location, *i.e.*, $q=|\mathbf{x}_f-\mathbf{x}_w|/|\mathbf{x}_f-\mathbf{x}_b|$, is smaller than 0.5, a virtual distribution function is interpolated first at $\mathbf{x}_i$ so that the molecules represented by this virtual distribution function ends precisely at $\mathbf{x}_f$ after the bounce back from the wall. Apparently, $\mathbf{x}_i$ locates between $\mathbf{x}_f$ and the neighboring fluid node $\mathbf{x}_{ff}$, thus the virtual distribution function can be interpolated from the corresponding distribution functions at $\mathbf{x}_f$, $\mathbf{x}_{ff}$, and $\mathbf{x}_{fff}$. On the other hand, when $q\geq 0.5$, $\mathbf{x}_i$ locates between $\mathbf{x}_f$ and $\mathbf{x}_w$, the interpolation becomes extrapolation, which could result in numerical instability. To avoid this, the streaming is proceeded first, *i.e.*, the distribution function at $\mathbf{x}_f$ first bounce-back from the wall and ends at a temporary location $\mathbf{x}_t$. Then the unknown distribution function at $\mathbf{x}_f$ is interpolated with the corresponding distribution functions at $\mathbf{x}_t$, $\mathbf{x}_{ff}$, and $\mathbf{x}_{fff}$. Bouzidi et al.'s interpolated bounce-back scheme can be summarized as

$$\begin{aligned} f_i\left(\mathbf{x}_f,t+\delta_t\right) &= q(2q+1)f_{\bar{i}}^*\left(\mathbf{x}_f,t\right)+(1+2q)(1-2q)f_{\bar{i}}^*\left(\mathbf{x}_{ff},t\right)\\ &\quad -q(1-2q)f_{\bar{i}}^*\left(\mathbf{x}_{fff},t\right)\\ &\quad +2\rho_0 w_i\frac{\mathbf{e}_i\cdot\mathbf{u}_w}{c_s^2}, \qquad q<0.5, \end{aligned} \tag{11a}$$

$$\begin{aligned} f_i\left(\mathbf{x}_f,t+\delta_t\right) &= \frac{1}{q(2q+1)}\left[f_{\bar{i}}^*\left(\mathbf{x}_f,t\right)+2\rho_0 w_i\frac{\mathbf{e}_i\cdot\mathbf{u}_w}{c_s^2}\right]\\ &\quad +\frac{2q-1}{q}f_i\left(\mathbf{x}_{ff},t+\delta_t\right)\\ &\quad -\frac{2q-1}{1+2q}f_i\left(\mathbf{x}_{fff},t+\delta_t\right), \qquad q\geq 0.5. \end{aligned} \tag{11b}$$

Alternatively, Yu et al. designed a unified IBB scheme for all values of q from 0 to 1. Their idea is straightforward. First, a virtual distribution function is interpolated between $\mathbf{x}_f$ and $\mathbf{x}_{ff}$, which ends exactly at the wall location after streaming a grid spacing towards the wall, *i.e.*,

$$\begin{aligned} f_{\bar{i}}(\mathbf{x}_w,t+\delta_t) &= \frac{q(q+1)}{2}f_{\bar{i}}^*\left(\mathbf{x}_f,t\right)+(1+q)(1-q)f_{\bar{i}}^*\left(\mathbf{x}_{ff},t\right)\\ &\quad -\frac{q(1-q)}{2}f_{\bar{i}}^*\left(\mathbf{x}_{fff},t\right). \end{aligned} \tag{12}$$

Next, an instantaneous bounce-back happens right after the virtual distribution function arrives at the wall location

$$f_i(\mathbf{x}_w,t+\delta_t)=f_{\bar{i}}(\mathbf{x}_w,t+\delta_t)+2\rho_0 w_i\frac{\mathbf{e}_i\cdot\mathbf{u}_w}{c_s^2} \tag{13}$$

At last, the unknown distribution function $f_i\left(\mathbf{x}_f,t+\delta_t\right)$ is interpolated from $f_i(\mathbf{x}_w,t+\delta_t)$, $f_i\left(\mathbf{x}_{ff},t+\delta_t\right)$ and $f_i\left(\mathbf{x}_{fff},t+\delta_t\right)$ as

$$\begin{aligned} f_i\left(\mathbf{x}_f,t+\delta_t\right) &= \frac{2}{(1+q)(2+q)}f_i(\mathbf{x}_w,t+\delta_t)\\ &\quad +\frac{2q}{1+q}f_i\left(\mathbf{x}_{ff},t+\delta_t\right)-\frac{q}{2+q}f_i\left(\mathbf{x}_{fff},t+\delta_t\right). \end{aligned} \tag{14}$$

In practice, it is more efficient to combine the above three steps into a single equation involving up to five distribution functions, which reads

$$\begin{aligned} f_i\left(\mathbf{x}_f,t+\delta_t\right) &= \frac{q}{2+q}f_{\bar{i}}^*\left(\mathbf{x}_f,t\right)+\frac{2(1-q)}{1+q}f_{\bar{i}}^*\left(\mathbf{x}_{ff},t\right)\\ &\quad -\frac{(1-q)q}{(1+q)(2+q)}f_{\bar{i}}^*\left(\mathbf{x}_{fff},t\right)\\ &\quad +\frac{2q}{1+q}f_i\left(\mathbf{x}_{ff},t+\delta_t\right)\\ &\quad -\frac{q}{2+q}f_i\left(\mathbf{x}_{fff},t+\delta_t\right)\\ &\quad +\frac{4}{(1+q)(2+q)}\rho_0 w_i\frac{\mathbf{e}_i\cdot\mathbf{u}_w}{c_s^2}. \end{aligned} \tag{15}$$

While these two schemes are constantly used in LBM for no-slip boundary treatment on curved surfaces. A potential issue is that they require not only the information at the boundary node itself, *i.e.*, $\mathbf{x}_f$, but also the distribution functions at $\mathbf{x}_{ff}$ and $\mathbf{x}_{fff}$ to process the interpolation. When two solid surfaces sit very close, which frequently happens in particle-laden flows with dense particle suspensions, Eq. (10) has to be used instead, where the overall accuracy of the boundary treatment might be contaminated. This potential issue is resolved with the recently proposed single-node second-order bounce-back scheme by Zhao & Yong [22], which reads

$$\begin{aligned} f_i\left(\mathbf{x}_f,t+\delta_t\right) &= \frac{2q}{1+2q}f_i^*\left(\mathbf{x}_f,t\right)+\frac{1}{1+2q}f_{\bar{i}}\left(\mathbf{x}_f,t\right)\\ &\quad +\frac{2}{1+2q}\rho_0 w_i\frac{\mathbf{e}_i\cdot\mathbf{u}_w}{c_s^2}. \end{aligned} \tag{16}$$

Unlike the previous two IBB schemes that construct $f_i\left(\mathbf{x}_f,t+\delta_t\right)$ purely from the post-collision distribution functions. Zhao & Yong's scheme utilize both the pre-collision and post-collision distribution functions to fulfill the "interpolation". The second-order accuracy of this scheme can be rigorously proven by an asymptotic analysis [22]. It is also worth mentioning that an alternative single-node second-order bounce-back scheme was recently proposed by Tao et al. [35]. We were made aware of this scheme quite late thus it is not included in our comparisons shown below.

These three IBB schemes, *i.e.*, Bouzidi et al.s scheme, Yu et al.s scheme, as well as Zhao & Yong's scheme will be used in the numerical examinations in Section 3. With the use of bounce-back schemes, the natural way to calculate the hydrodynamic force and torque acting on a solid surface is the momentum exchange method (MEM) [12,36,37]. Although the combinations of bounce-back schemes and MEM do not ensure Galilean invariance in instantaneous force and torque calculation [38], their accuracy has

been proven to be sufficient in most simulations [39,40]. In particular, the Galilean invariant momentum exchange method (GIMEM) proposed by Wen et al. [37],

$$\mathbf{F}\delta_t = \sum_{\mathbf{x}_f,i} \left[f_{\bar{i}}^*\left(\mathbf{x}_f, t\right)\left(\mathbf{e}_{\bar{i}} - \mathbf{u}_w\right) - f_i\left(\mathbf{x}_f, t+\delta_t\right)\left(\mathbf{e}_i - \mathbf{u}_w\right)\right], \tag{17a}$$

$$\mathbf{T}\delta_t = \sum_{\mathbf{x}_f,i} \left(\mathbf{x}_w - \mathbf{Y}_c\right) \times \left[f_{\bar{i}}^*\left(\mathbf{x}_f, t\right)\left(\mathbf{e}_{\bar{i}} - \mathbf{u}_w\right) - f_i\left(\mathbf{x}_f, t+\delta_t\right)\left(\mathbf{e}_i - \mathbf{u}_w\right)\right], \tag{17b}$$

will be used in the subsequent numerical examinations to reduce the "grid locking" ("grid locking" will be discussed in detail later) [39]. This is different from the original MEM [12,36].

3. Numerical examinations

Appropriately chosen validation cases help us better evaluate the performance of the boundary treatment schemes. In the earlier investigations, the accuracy of IBM was often validated in the flow of a two-dimensional Taylor-Green vortex flow. These tests, in our view, are not so meaningful since the accurate flow field can be obtained with or without the boundary forcing. The only information one may obtain from these tests is perhaps that IBM does not contaminate the second-order accuracy of LBM when it is applied to a smooth flow field[1]. Unfortunately, the velocity across a real solid-fluid interface is usually not smooth [5]. Another often used test flow is a uniform flow passing a 2D cylinder or 3D sphere at finite Reynolds number. In this case, since the analytic solution is unavailable, while it is safe to validate whether a boundary treatment method is generating reasonable results, it is difficult to assess rigorously the accuracy and compare the results among different methods.

In this paper, we choose four test flows to benchmark the performances of IBB schemes and IBM algorithms. The two-dimensional circular Couette flow and the three-dimensional laminar pipe flow are chosen since analytic solutions are available that can help benchmarking the accuracy of each boundary treatment when an actual curved wall presents. A case of two-dimensional cylinder settling in a quiescent flow is used to examine the performance of each boundary treatment in predicting the motion of the objects in a viscous fluid. At last, a case of a uniform flow passing a static sphere is employed to assess the grid resolution requirement for each boundary treatment in order to obtain reliable hydrodynamic force acting on a spherical particle at different Reynolds numbers.

3.1. Transient circular Couette flow

The purpose of the present study is to assess the performance of the boundary treatment schemes in general cases with curved and moving boundaries. For this purpose, the circular Couette flow, or Taylor-Couette flow between two concentric cylinders is employed. A sketch of this flow is shown in Fig. 2. The analytic solution of this flow is available in [41]. We repeat it in Appendix A simply for readers' convenience.

In the simulations presented below, the inner cylinder is fixed while the outer cylinder rotates with an angular velocity that defines the flow Reynolds number $Re = (R_2 - R_1)\Omega_2 R_2/\nu = 45$. The ratio of the outer to inner cylinder radius, γ, is set to 2. The simulations are conducted using the D2Q9 MRT collision model but run with a single relaxation time, *i.e.*, the equilibrium and the body force terms are defined in the moment space but all the relaxation times in matrix **S** are identical.

[1] the smooth flow field is defined as a field where the velocity gradient normal to the interface is continuous, see Peskin [5].

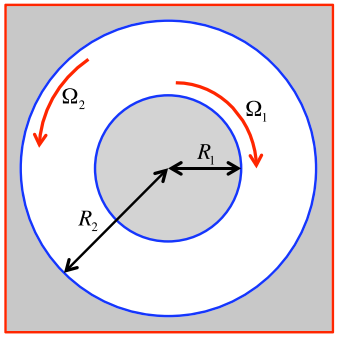

Fig. 2. A sketch of a Taylor-Couette flow.

Unlike in LBM-IBB simulations where the boundary treatment is purely determined by the information from the fluid region (white region in Fig. 2), in LBM-IBM simulations the whole domain, including the solid region (gray region in Fig. 2), is filled with the same fluid and the flows outside and inside the solid region may be inter-connected through the N-S equations. Therefore, how the flow in the solid region is treated may affect the flow within the fluid region. Specifically, in the Circular Couette flow, appropriate treatment of the boundary of the computational domain (red solid lines in Fig. 2) plays an important role in ensuring the correctness of the results, especially when the outer cylinder is rotating. To demonstrate this point, we present the velocity profiles at different non-dimensional times ($t^* = t\nu/(R_2 - R_1)^2$) from two LBM-IBM simulations, both use Breugem's IBM [7] with a retraction distance $0.3\delta x$ to treat the two no-slip conditions on the cylinder surfaces, but with Dirichlet boundary (a) $u_r = 0, u_\theta = 0$, (b) $u_r = 0, u_\theta = \Omega_2 r$ on the edges of the computational domain. The grid resolution used for the simulations is $R_1 = 25\delta x$. The profiles are generated by averaging the velocity at the grid nodes sitting in 25 equal-width bins with the width of $dr = (R_2 - R_1)/25$. Obviously, with setting (a), the velocity profiles of the simulation deviate from the theoretical solution, while with setting (b), the velocity profiles match the theoretical solution quite well. This observation leads to the first remark that cautions must be given to the treatment of flow in the solid region when IBM is used. As we shall observe later inyes Fig. 6, even setting (b) can result in a significant error in the hydrodynamic force evaluation. Unfortunately, the treatment on the edges of the computational domain is usually irrelevant to the physical description of the flow. LBM-IBB simulations, on the other hand, do not suffer from the same problem. The construction on the unknown distribution functions at the boundary nodes purely depends on the information in the fluid region. The velocity profiles of the LBM-IBB simulation with Bouzidi et al.'s quadratic interpolation scheme are in good agreement with the theory, as shown in Fig. 3(c).

To quantify the numerical error of the results in a simulation, the L1- and L2-errors, defined as

$$\varepsilon_{L1} = \frac{\sum_{\mathbf{x}} \|Q_s(\mathbf{x}) - Q_t(\mathbf{x})\|}{\sum_{\mathbf{x}} \|Q_t(\mathbf{x})\|}, \quad \varepsilon_{L2} = \frac{\sqrt{\sum_{\mathbf{x}} [Q_s(\mathbf{x}) - Q_t(\mathbf{x})]^2}}{\sqrt{\sum_{\mathbf{x}} [Q_t(\mathbf{x})]^2}}, \tag{18}$$

are calculated, where Q_s and Q_t are the numerical result and theoretical result, respectively. The convergence rates of L2-errors of the velocity fields at the steady state are presented in Fig. 4, for three LBM-IBM simulations, *i.e.*, with the IBM scheme of by Uhlmann ("LBM-IBM-Uhlmann"), and with the IBM scheme proposed by Breugem with two different retraction distances, $0.3\delta x$ and $0.4\delta x$ ("LBM-IBM-Breugem, $R_d = 0.3$" and "LBM-IBM-Breugem, $R_d = 0.4$"), as well as three LBM-IBB simulations, using the quadratic interpolation schemes by Bouzidi et al. ("LBM-IBB-Bouzidi") and Yu et al. ("LBM-IBB-Yu"), and the single-node

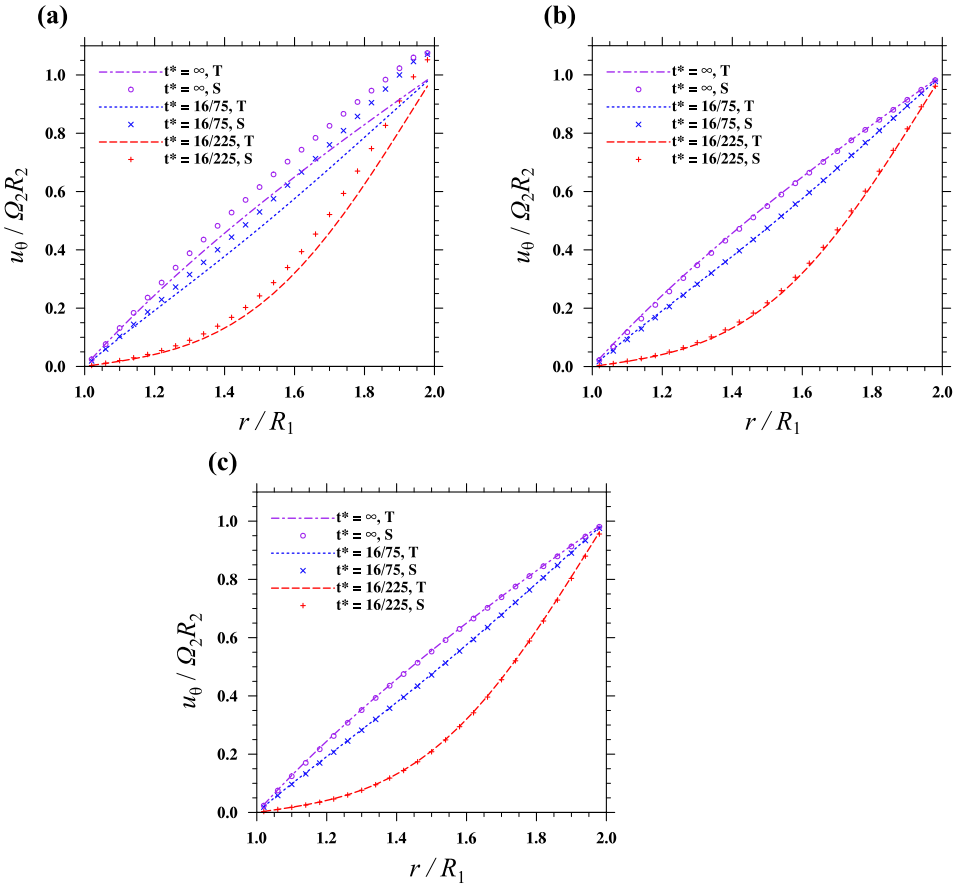

Fig. 3. Velocity profiles of a transient Taylor-Couette flow: (a) LBM-IBM simulation with Breugem's IBM scheme with a retraction distance of $0.3\delta x$, the velocity on the edges of the computational domain is set as $u_r = 0, u_\theta = 0$; (b) same as (a), except that the velocity on the edges of the computational domain is set as $u_r = 0, u_\theta = \Omega_2 r$; (c) LBM-IBB simulation with Bouzidi *et al.*'s quadratic interpolated bounce-back scheme. "S" and "T" in the legend of each plot represent "simulation result" and "theoretical result", respectively.

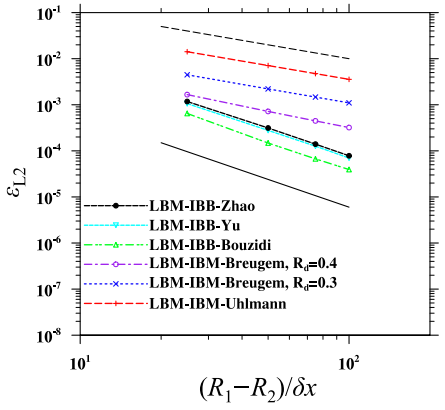

Fig. 4. Error convergence rates of the velocity field in the LBM-IBM and LBM-IBB simulations. The dash line and the solid line are references of slop -1 and -2, respectively. The same applied to all figures in the rest of the paper.

bounce-back scheme by Zhao & Yong ("LBM-IBB-Zhao"). The boundary force in the three LBM-IBM simulations are iterated for 5 times to ensure the representation of the no-slip boundary on the Lagrangian points is sufficiently accurate. As clearly demonstrated in Fig. 4, the velocity fields in the three LBM-IBM simulations are always of first-order accuracy, while these from all the three LBM-IBB simulations are of second-order accuracy.

The first-order accuracy of the LBM-IBM is a result of the fact that the delta-function used to interpolate information between the Eulerian and Lagrangian grids possesses second-order accuracy only for a smooth interface where the velocity gradient normal to the interface is continuous [5,6,42]. While this remark is already quite well-known in IBM, we here provide a theoretical proof in the Appendix B. The idea of this proof is to assume the velocity prior to the boundary forcing process is exact, and examine what is the order of the error generated in the boundary forcing process.

Although the retraction of Lagrangian grid does not improve the order of accuracy of the velocity calculation in the LBM-IBM simulation, it does significantly reduce the magnitudes of the error at all resolutions (the results labeled Uhlmann in Fig. 4 is equal to the case with zero retraction distance). Breugem examined the effect of the retraction distance in a few flow examples, such as a uniform flow passing a fixed sphere and the laminar pipe flow, and suggested that $R_d = 0.3\delta x$ was the general optimal retraction distance. Zhou & Fan also confirmed such observation in LBM-IBM that an optimized retraction distance should be $0.3\delta x \le R_d \le 0.4\delta x$. The three-point delta-function of Roma et al. [43] was adopted in both studies to draw this conclusion. Intuitively, since the physical fluid-solid interface is diffused at different levels by different delta-functions, the optimal retraction distance to offset such diffusion should be delta-function dependent.

To confirm this point, we simulate the same TC flow with different combinations of three delta-functions, *i.e.*, the four-point piecewise delta-function used above, the three-point piecewise delta-function by Roma et al., and the two-point linear delta-function, and five retraction distances $R_d = 0$, $R_d = 0.1\delta x$, $R_d = 0.2\delta x$, $R_d = 0.3\delta x$, and $R_d = 0.4\delta x$. It should be noted that the three-point

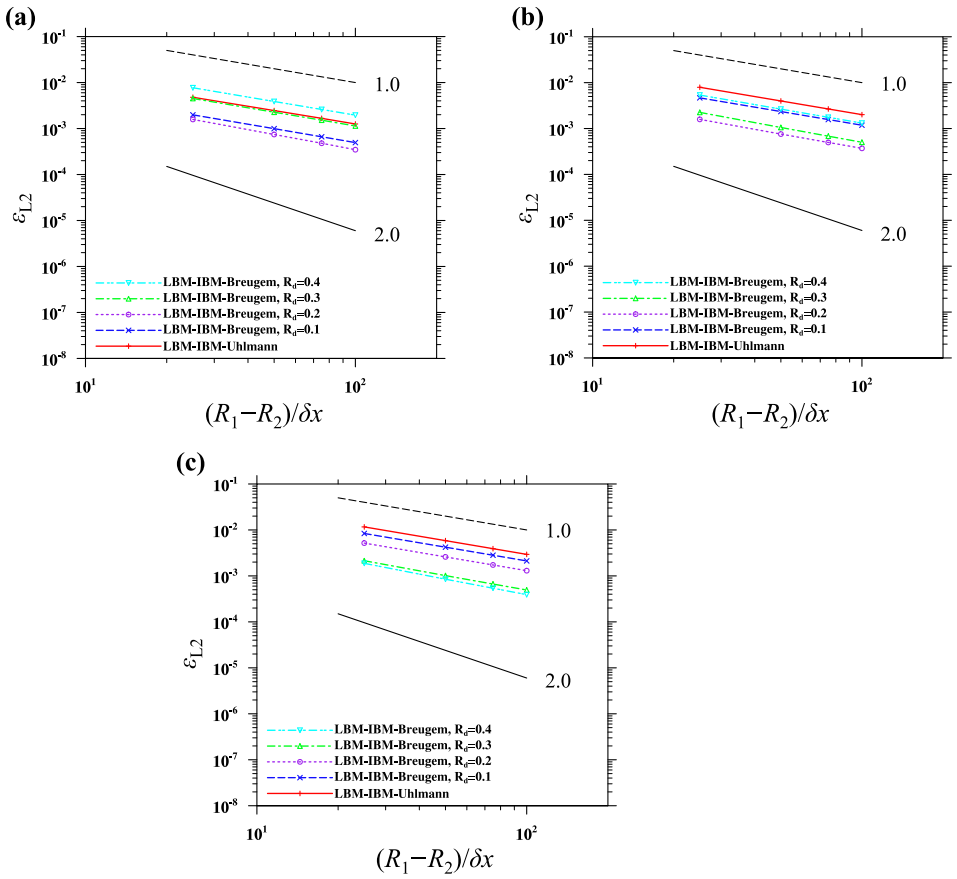

Fig. 5. The convergence rates of velocity field with different delta-functions, (a) two-point linear delta-function, (b) three-point delta-function, (c) four-point delta-function.

piecewise delta-function and the two-point linear delta-function diffuse the physical fluid-solid interface less than their four-point counterpart, which may bring a negative impact on the numerical stability. In fact, with all the other simulation setup parameters being identical to what were used earlier, switching to the three-point and two-point delta-functions made the code diverge. To ensure numerical stability with all combinations, a smaller flow Reynolds number $Re = (R_2 - R_1)\Omega_2 R_2/\nu = 15$ is used instead. The convergence rates of the steady state velocity fields in different cases are shown in Fig. 5. In each simulation, the boundary force is still iterated for 5 times to ensure better no-slip boundary representation. As shown in Fig. 5, when the two-point linear delta-function is used, the retraction distances of $R_d = 0.1\delta x$ and $R_d = 0.2\delta x$ result in the most accurate velocity field. With more diffusive delta-functions, the optimized retraction distance becomes larger in magnitude. With the three-point delta-function, the optimized retraction distance is between $R_d = 0.2\delta x$ and $R_d = 0.3\delta x$; while with the four-point delta-function, the best result is observed when $R_d = 0.3\delta x$ and $R_d = 0.4\delta x$. Another observation worth mentioning is that, for the current Reynolds number $Re = 15$, with four-point delta function (Fig. 5(c)), the retraction distance of $R_d = 0.4\delta x$ only results in slightly more accurate velocity field than the retraction distance of $R_d = 0.3\delta x$. However, as shown in Fig. 4, when the Reynolds number increases to $Re = 45$, the results improve much more significantly when R_d is increased from 0.3 to 0.4. A Reynolds number dependence of the optimized retraction distance may also be expected.

Unlike LBM-IBM, the interpolated bounce-back schemes can preserve the second-order spatial accuracy when curved no-slip surfaces are present. This is because the interpolation schemes ensure the construction of the unknown distribution functions at the boundary grid points is of second- or higher-order spatial accuracy. Particularly, the single-node bounce-back scheme by Zhao & Yong is able to achieve a second-order accuracy using only the information at the boundary node itself. This scheme is useful when simulating flow in porous media, or flows with dense particle suspensions, where narrow gaps can form between two solid surfaces that disables multiple-point interpolations. With the contribution of Zhao & Yong's bounce-back scheme, the no-slip boundary treatment via IBB should possess second-order accuracy in any situation.

We next examine the accuracy of simulated hydrodynamic force in LBM-IBM and LBM-IBB. In LBM-IBM, the boundary force and torque have already been calculated at each Lagrangian grid, obtaining the total hydrodynamic force and torque acting on the solid objects simply amounts to summing up the contributions over all the Lagrangian grid points. When LBM-IBB is used, the hydrodynamic force and torque are calculated with Eq. (17). A slight difference to note is that when LBM-IBB is used, the force calculated with the momentum exchange method contains a hydrostatic pressure contribution in the wall-normal direction since there is no fluid inside the solid domain. On the other hand, the force calculated in LBM-IBM contains only the viscous stress, as fluid exists on both sides of the boundary. Fig. 6 shows the torques acting on the inner and outer cylinders for the same cases shown in Fig. 3(b) and (c). The two solid black lines represent the analytic torque on the inner and outer cylinders at the steady state. While the torque results of LBM-IBB match well with the analytic results on both cylinders, the result of LBM-IBM has a significant derivation from the theory on the outer cylinder. This is again due to the poor treatment on edges of the computational domain. Rather than setting the Dirichlet boundary $u_r = 0, u_\theta = \Omega_2 r$, a better boundary

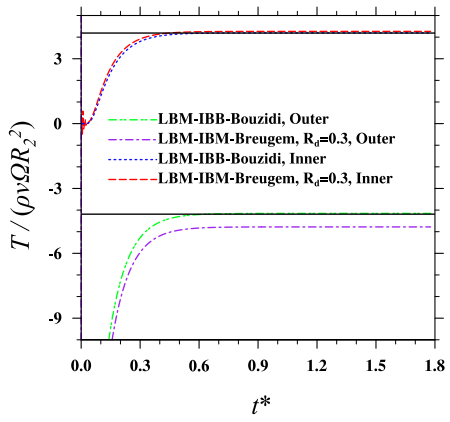

Fig. 6. The time-dependent torque acting on the inner and outer cylinders.

condition potentially reduces the error. However, this information is not available in the physical problem description. The accuracy of torque evaluation in different simulations is presented in Fig. 7. The results of the torque on the outer cylinder in LBM-IBM simulations are no longer included. Again, the torque evaluations in the three LBM-IBM simulations are still first-order accurate, with or without retracting the Lagrangian grid. This observation seems to conflict with the conclusion reported in the literature that the retraction of Lagrangian grid in IBM results in a second-order accurate total force/torque. As shown in Appendix B, the local velocity fields in IBM have only first-order accuracy, which constrains the accuracy of local force evaluation to be the first order. Whether the first-order error at each Lagrangian grid point can be canceled out to result in a second-order accurate total force/torque depends on the specific flow patterns. In a Taylor-Couette flow, the flow is azimuthally independent, which means the local error of hydrodynamic force calculation at each Lagrangian grid point should be the same. In this case, the first-order local errors cannot be cancelled out, as such the total hydrodynamic force remains to have only a first-order accuracy. On the other hand, in cases of a uniform flow passing a fixed cylinder or sphere, symmetric flow pattern may form around the cylinder/sphere. In such cases, the first-order local error contributed by each Lagrangian point may cancel out precisely to yield a second-order accuracy for the total force. The latter observation has been widely reported in the literature [7,18,21], and also confirmed by our own simulation in Section 3.4. We emphasize that the hydrodynamic force/torque calculation in IBM cannot reach the second-order accuracy in general. On the contrary, the torques calculated with momentum exchange method

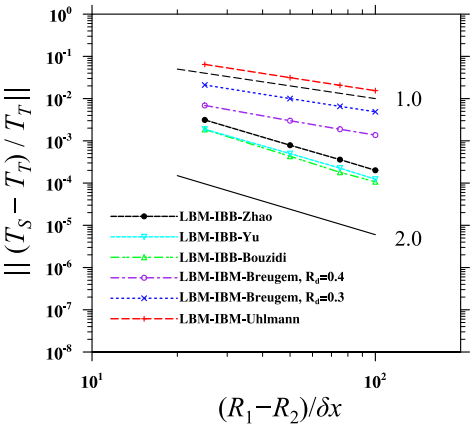

Fig. 7. The convergence rates of the torque evaluation error.

in the LBM-IBB simulations are always second-order accurate. This is because the bounced-back distribution function in Eq. (17) are of second-order accuracy, same as the accuracy of the interpolated bounce-back schemes.

We last examine the calculation of the dissipation rate in different LBM-IBM and LBM-IBB simulations. The dissipation rate is an important quantity in turbulent flows that affects the energy budget in a flow. In turbulent flows, the dissipation rate is often defined as $\varepsilon = 2\nu s'_{ij}s'_{ij}$, where s_{ij} is the velocity strain rate tensor, " $'$ " indicates its fluctuation part in the Reynolds decomposition [44]. Here in the laminar flow, the velocity is not decomposed and the dissipation rate is defined as $\varepsilon = 2\nu s_{ij}s_{ij}$ instead. In the framework of LBM, there are two different ways to calculate the strain rate tensor $s_{ij} = 0.5\left(\partial u_i/\partial x_j + \partial u_j/\partial x_i\right)$. The first way is to use a finite-difference approximation, as adopted in conventional CFD. To preserve the accuracy, a second- or higher-order finite-difference scheme is usually required. Alternatively, s_{ij} in LBM can be calculated directly as a moment of the non-equilibrium distribution functions. According to the Chapman-Enskog expansion and taking the D2Q9 MRT collision operator used in the simulation, the three components in s_{ij} can be calculated as

$$\frac{\partial u}{\partial x} = -\frac{s_e}{4\rho_0\delta t}\epsilon m_e^{(1)} - \frac{3s_n}{4\rho_0\delta t}\epsilon m_n^{(1)} - \frac{1}{4\rho_0}\left[\left(\frac{s_e}{2-s_e}\right)\psi_e + \left(\frac{3s_n}{2-s_n}\right)\psi_n\right], \tag{19a}$$

$$\frac{\partial v}{\partial y} = -\frac{s_e}{4\rho_0\delta t}\epsilon m_e^{(1)} + \frac{3s_n}{4\rho_0\delta t}\epsilon m_n^{(1)} - \frac{1}{4\rho_0}\left[\left(\frac{s_e}{2-s_e}\right)\psi_e - \left(\frac{3s_n}{2-s_n}\right)\psi_n\right], \tag{19b}$$

$$\frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) = -\frac{3s_c}{2\rho_0\delta t}\epsilon m_c^{(1)} - \frac{1}{2\rho_0}\left(\frac{3s_c}{2-s_c}\right)\psi_c, \tag{19c}$$

where s_e, s_n and s_c are the relaxation parameters for the energy, normal stress and shear stress moments, respectively. $\epsilon m_e^{(1)} \approx m_e - m_e^{(eq)}$, $\epsilon m_n^{(1)} \approx m_n - m_n^{(eq)}$, $\epsilon m_c^{(1)} \approx m_c - m_c^{(eq)}$ are their corresponding leading-order non-equilibrium part. ψ_e, ψ_n, ψ_c are the corresponding components in the mesoscopic forcing term $\mathbf{\Psi}$ in Eq. (3), whose definition can be found in [34]. Compared to the finite-difference approximation, the mesoscopic method of calculating the strain rate tensor from the non-equilibrium moments (or distribution functions if LBGK collision operator is used) ensures a second-order accuracy even when the velocity field in the LBM simulation is of the same second-order accuracy [45,46], which makes it generally preferred.

The profiles of dissipation rate in the two simulations shown in Fig. 3(b) and 3(c) are exhibited in Fig. 8(a). For the LBM-IBM simulation, the dissipation rate is calculated in three different ways, *i.e.*, 1) with the second-order central finite-difference scheme (FD1), 2) use the second-order central difference scheme in the bulk fluid region, but replace with a second-order upwind scheme near the two solid surfaces to exclude the grid points in the solid region from the calculation (FD2), and 3) from the non-equilibrium moments (ME). In the LBM-IBB simulation, for the sake of simplicity, only the mesoscopic method is employed. As shown in Fig. 8, no matter which method is employed to calculate the dissipation rate in the LBM-IBM simulation, the results are always significantly smaller than the theory. This is probably because IBM smooths out the sharp fluid-solid interfaces and reduces the local velocity gradient in the interface region. Excluding the grid points inside the solid volume improves the accuracy of dissipation rate calculation near the boundary but a large part of the error still remains. In the fluid region away from boundary ($1.2 \le r/R_1 \le 1.8$), the dissipation rate results of the LBM-IBM simulations become acceptable, with only a slight over-prediction of the dissipation rate. The local dis-

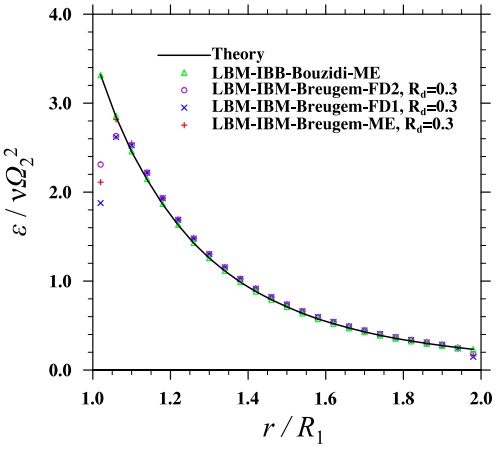

Fig. 8. The profiles of the dissipation rate.

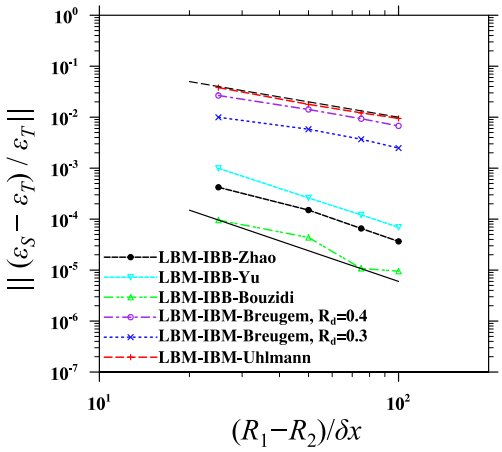

Fig. 10. The convergence rates of the total dissipation rate in the whole fluid domain.

sipation rate result in LBM-IBB, on the other hand, is in excellent agreement with the theory. In the fluid region away from boundary ($1.2 \le r/R_1 \le 1.8$), the calculated dissipation rate from LBM-IBM is acceptable, but it is still worse than that in LBM-IBB. This indicates that the overall accuracy of IBB in terms of no-slip boundary treatment is much better than that in IBM.

The convergence rates of dissipation rates calculation in the LBM-IBM and LBM-IBB are presented in Fig. 9. Since the non-uniform distributions of the error in the LBM-IBM simulations (see Fig. 8) tend to amplify L2 errors, only L1 errors are presented. For conciseness, only the dissipation rates calculated by the mesoscopic way are presented. Clearly, the dissipation rate calulation in LBM-IBM is of only a first-order accuracy, while the dissipation rate calculation in LBM-IBB is of a second-order accuracy. The L1 error in the latter is about one to two orders of magnitude smaller. While using IBM to treat the no-slip boundary can lead to significant numerical errors in dissipation rate results near the fluid-solid interfaces, results of the total dissipation summing over the whole fluid domain are more acceptable. Corresponding results are shown in Fig. 10. This is because the underestimated dissipation rates near the fluid-solid interfaces due to the diffused boundary are offset by their overestimated counterparts away from the interfaces, which can be seen in Fig. 8. Nevertheless, the above comparisons indicate that the regular definition of dissipation rate may need to be improved in order to account for the diffused boundary effect in IBM. This aspect receives little attention in the past, thus further investigations are certainly required.

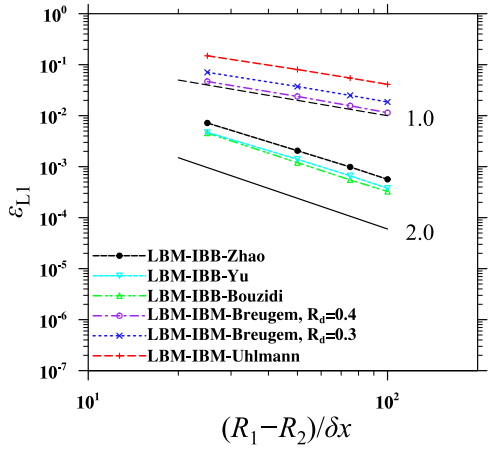

Fig. 9. The convergence rates of the dissipation rate calculation.

3.2. Sedimentation of a cylinder in a vertical channel

Next we compare the performance of the interpolated bounce-back schemes and immersed boundary methods in calculating the force/torque on a moving solid object. For a solid object immersed in a viscous fluid, the governing equations for its translational motion and angular rotation read

$$\rho_p V_p \frac{d\vec{v}}{dt} = \oint_{\partial s} \left(\vec{\tau}\cdot\vec{n}\right) ds + \left(\rho_p - \rho_f\right) V_p \vec{g} + \vec{F}_{int} + \cdots, \tag{20a}$$

$$I_p \frac{d\vec{\omega}}{dt} = \oint_{\partial s} \vec{r} \times \left(\vec{\tau}\cdot\vec{n}\right) ds + \vec{T}_{int} + \cdots, \tag{20b}$$

where ρ_p and ρ_f are the densities of the solid and fluid phases, respectively. V_p is the volume of the solid object, $\vec{g}$ is the gravitational acceleration, $\vec{F}_{int}$ and $\vec{T}_{int}$ are the force and torque due to the interaction with other solid objects. Other sources of force and torque may also be included. In LBM-IBB simulations, the hydrodynamic force and torque are computed from the amount of momentum/angular momentum exchanges. On the other hand, in LBM-IBM simulations, since the solid object is also filled with fluid, the fluid inertia inside the solid object appears in the momentum/angular momentum balances as

$$\oint_{\partial s} \left(\vec{\tau}\cdot\vec{n}\right) ds = \frac{d}{dt}\int_{\partial V} \vec{u} dV - \int_{\partial V} \vec{f} dV, \tag{21a}$$

$$\oint_{\partial s} \vec{r} \times \left(\vec{\tau}\cdot\vec{n}\right) ds = \frac{d}{dt}\int_{\partial V} \left(\vec{r}\times\vec{u}\right) dV - \int_{\partial V} \left(\vec{r}\times\vec{f}\right) dV. \tag{21b}$$

where ∂S and ∂V are the surface and volume of a solid object. When calculating the hydrodynamic force/torque, the treatment of the fluid inertia inside the particle clearly plays an important role. A straightforward treatment is to assume the fluid inside the solid object is following rigid body motion, as did by Uhlmann [6]. With such assumption, Eq. (20) becomes

$$\left(\rho_p - \rho_f\right) V_p \frac{d\vec{v}}{dt} = -\int_{\partial V} \vec{f} dV + \left(\rho_p - \rho_f\right) V_p \vec{g} + \vec{F}_{int} + \cdots, \tag{22a}$$

$$I_p \left(1 - \frac{\rho_f}{\rho_p}\right) \frac{d\vec{\omega}}{dt} = -\int_{\partial V} \left(\vec{r}\times\vec{f}\right) dV + \vec{T}_{int} + \cdots. \tag{22b}$$

An obvious problem of Eq. (22) is that the left-hand sides vanish when $\rho_p \approx \rho_f$. When the density ratio ρ_p/ρ_f is below a limit, simulations employing Eq. (22) are not stable. To overcome such stability deficiency, Feng & Michaelides [47] proposed a specific time discretization of Eq. (22) as

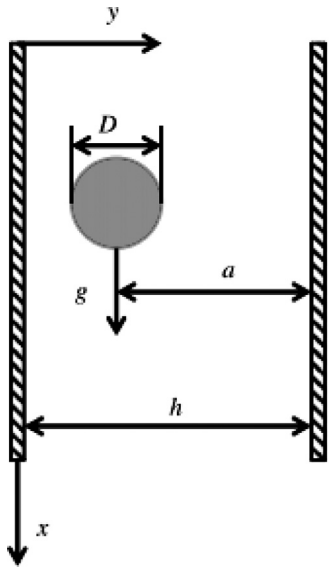

Fig. 11. A sketch of a cylinder settling in a quiescent flow.

$$\rho_p V_p \frac{\vec{v}^{n+1}-\vec{v}^{n}}{\delta t} = \rho_f V_p \frac{\vec{v}^{n}-\vec{v}^{n-1}}{\delta t} - \int_{\partial V} \vec{f} dV + \left(\rho_p - \rho_f\right) V_p \vec{g} + \vec{F}_{int} + \cdots, \tag{23a}$$

$$I_p \frac{\vec{\omega}^{n+1}-\vec{\omega}^{n}}{\delta t} = I_p \frac{\rho_f}{\rho_p} \frac{\vec{\omega}^{n}-\vec{\omega}^{n-1}}{\delta t} - \int_{\partial V} \left(\vec{r}\times\vec{f}\right) dV + \vec{T}_{int} + \cdots. \tag{23b}$$

Alternatively, one can directly compute the fluid inertia inside the solid object to avoid singularity when the density ratio is close to unity. Kempe et al. [48] used a level set functions to compute such terms as

$$\int_{\partial V} \vec{u} dV = \sum_{1}^{n_x}\sum_{1}^{n_y}\sum_{1}^{n_z} \vec{u}_{i,j,k} h^3 \alpha_{i,j,k},$$
$$\int_{\partial V} \vec{r}\times\vec{u} dV = \sum_{1}^{n_x}\sum_{1}^{n_y}\sum_{1}^{n_z} \vec{r}_{i,j,k}\times\vec{u}_{i,j,k} h^3 \alpha_{i,j,k}, \tag{24}$$

where

$$\alpha_{i,j,k} = \frac{\sum_{l=1}^{8} -\phi_l H(-\phi_l)}{\sum_{l=1}^{8} \|\phi_l\|}, \tag{25}$$

ϕ_l is a signed distance function,

$$\phi_l = \sqrt{\frac{\left(\vec{x}_{i,j,k}-\vec{x}_c\right)^2}{a^2} + \frac{\left(\vec{y}_{i,j,k}-\vec{y}_c\right)^2}{b^2} + \frac{\left(\vec{z}_{i,j,k}-\vec{z}_c\right)^2}{c^2}} - 1 \tag{26}$$

where $(\vec{x}_c, \vec{y}_c, \vec{z}_c)$ is the center location of a particle, a, b, c are the lengths of the three axes of an ellipsoidal shaped particle. Apparently, one should obtain $\phi_l > 0$ outside and $\phi_l < 0$ inside the particle, $H(\phi_l)$ is the Heaviside function. The summation is over the 8 corners of a three-dimensional grid cell, or 4 corners of a two-dimensional grid cell. In the following test, both Eqs. (23) and (24) will be examined in LBM-IBM simulations of moving particles in viscous flows.

The benchmark case chosen here is a cylinder settling in a vertical channel. A sketch of the flow is shown in Fig. 11. The parameters in physical units are chosen as $D = 0.1cm$, $L = 4cm$, $H = 0.4cm$, $a = 0.324cm$, $g = 980cm^2/s^2$, and the density ratio $\rho_p/\rho_f = 1.03$ to match the arbitrary Lagrangian Eulerian (ALE) simulation performed by Hu et al. [49]. First, the two ways of considering the inertia of fluid inside the cylinder, *i.e.*, Eqs. (23) and (24) are compared in the LBM-IBM simulations. Eq. (22) is not included as it results in instability with the density ratio 1.03. The trajectory, angular velocity, vertical and horizontal translational velocities of the cylinder are presented in Fig. 12(a), (b), (c), and (d), respectively. The results are obtained with a grid resolution of $D = 30\delta x$. The results are not sensitive to how the inertia of the fluid in the cylinder is treated. Assuming the fluid inside the two-dimensional cylinder follows the rigid body motion appears to be safe. Compared to Uhlmann's IBM with zero retraction distance, Breugem's IBM with a retraction distance of $r_d = 0.4\delta x$ clearly improves the accuracy of simulating the particle motion. Particularly, the terminal velocity of the cylinder with Uhlmann's IBM is smaller than the benchmark result. This is because the diffused fluid-solid interface creates a larger effective hydraulic radius that over predicts the drag force. The retraction of Lagrangian grid points helps to offset such over prediction [7].

We next compare the performance of LBM-IBB and LBM-IBM in simulating the particle motion. The verticel velocity of the cylinder with Bouzidi et al.'s interpolated bounce-back scheme, and Breugem's IBM with a retraction distance of $0.4\delta x$ are presented in Fig. 13(a) and (b), respectively. Here we simulate the same flow with different grid resolutions from $D = 10\delta x$ to $D = 45\delta x$. The results of LBM-IBB simulation almost converge at the grid resolution of $D = 10\delta x$, while the results of LBM-IBM simulation reach the same accuracy from the grid resolution of $D = 15\delta x$. This is mainly due to the advantage of the second-order accuracy in IBB compared to the first-order accuracy in IBM. Assuming the ALE benchmark results are accurate, the grid-independent numerical error of the LBM-IBB simulation is slightly larger than that of the LBM-IBM. This benefit is likely brought by the adjustable retraction distance r_d in the latter.

At the end of this case, the level of "grid locking" in the hydrodynamic force/torque evaluations is examined. The "grid locking" means when a solid object crosses over the grid mesh, the calculated instantaneous hydrodynamic force/torque exerted on the solid object have a slight dependence on the configuration of the grid mesh and the solid object, and not being strictly Galilean invariant [7]. The calculated instantaneous force and torque therefore present a high-frequency fluctuation which restores its initial value when the solid object displaces exactly one grid spacing. The term "grid locking" was dubbed by Breugem but the phenomenon was discovered much earlier in IBM, *e.g.*, in [6,50]. LBM-IBB simulations also suffer from the same problem, as discussed by Lallemand & Luo [51], Peng et al. [39] and Tao et al. [40]. Essentially, both the interpolation in IBB and the boundary diffusion in IBM have made the realization of no-slip condition on the sharp interface depending on the information of multiple grid points around, which helps to suppress the fluctuation in force and torque evaluation. In Fig. 14(a) the effects of the two schemes, *i.e.*, Feng & Michaelides's scheme (Eq. (23)) and Kempe et al.'s scheme (Eq. (24)), for treating the fluid inertia inside the solid volume in IBM are compared. At the initial stage, the scheme of Feng & Michaelides presents a lower level of force fluctuation that the scheme of Kempe et al., but the two schemes eventually lead to the same prediction of the force, as shown in the inserted zoom-in plot in Fig. 14(a). Fig. 14(b) shows the comparison of force evaluation among four simulations, two LBM-IBB simulations with the quadratic interpolation scheme of Bouzidi et al. (Eq. (11)) and the single-node bounce-back scheme of Zhao & Yong (Eq. (16)), and two LBM-IBM simulations with Uhlmann's IBM and Breugem's IBM with a retraction distance of $r_d = 0.4\delta x$. The two LBM-IBM simulations use Kempe et al.'s scheme to directly consider the inertia of fluid inside particle region. Compared to IBB schemes, IBM results clearly better suppress the force fluctuation. This benefit is a result that the delta-functions employed in IBM diffuse the sharp

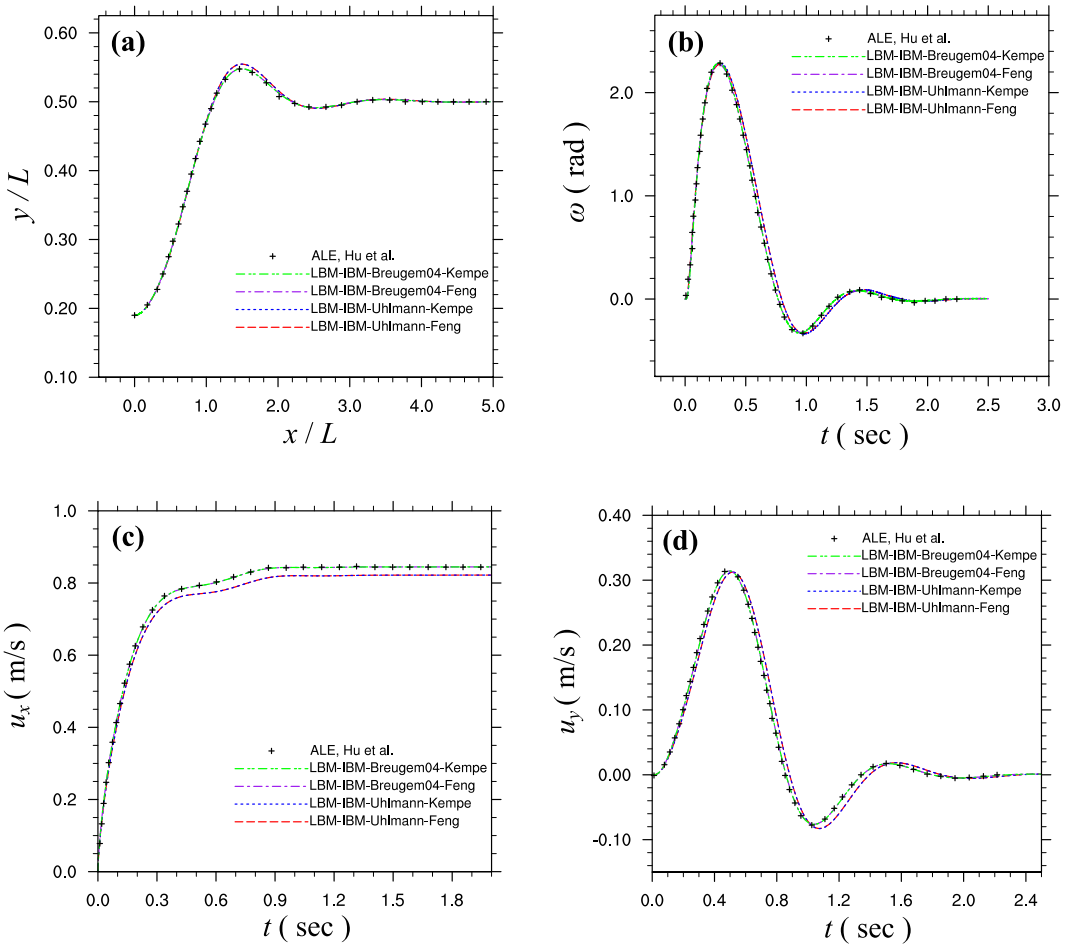

Fig. 12. The effects of treatment of inertia of fluid inside cylinder on (a) particle trajectory, (b) particle angular velocity, (c) vertical translational velocity, (d) horizontal translational velocity.

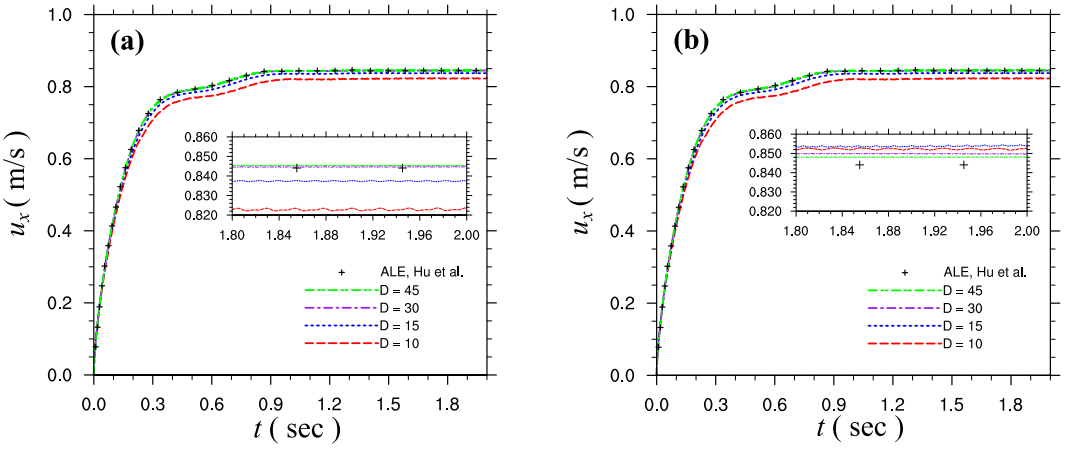

Fig. 13. The results of particle vertical translational velocity with different grid resolutions, (a) LBM-IBM with Breugem's IBM, $r_d = 0.4\delta x$, (b) LBM-IBB with Bouzidi et al.'s quadratic interpolated bounce-back scheme.

interface more in IBM than in the interpolation schemes used in IBB. In IBM, the force contributed from a single Lagrangian node depends the information from a maximum 4×4 (in 2D) subdomain of the Eulerian mesh. On the contrary, in IBB, the force contributed by a single boundary link depends on the information from no more than three node points. The latter system therefore has a much less inertia to suppress the high-frequency fluctuations. The force fluctuation in an IBM simulation might be further suppressed by using more diffusive delta-functions with larger spans, as suggested in Ref. [50]. However, those more diffusive delta-functions will introduce larger numerical viscosity and further reduce the accuracy of the IBM simulation in terms of averaged quantities. If the instantaneous force/torque computation is not of particular importance and the simulation has sufficient

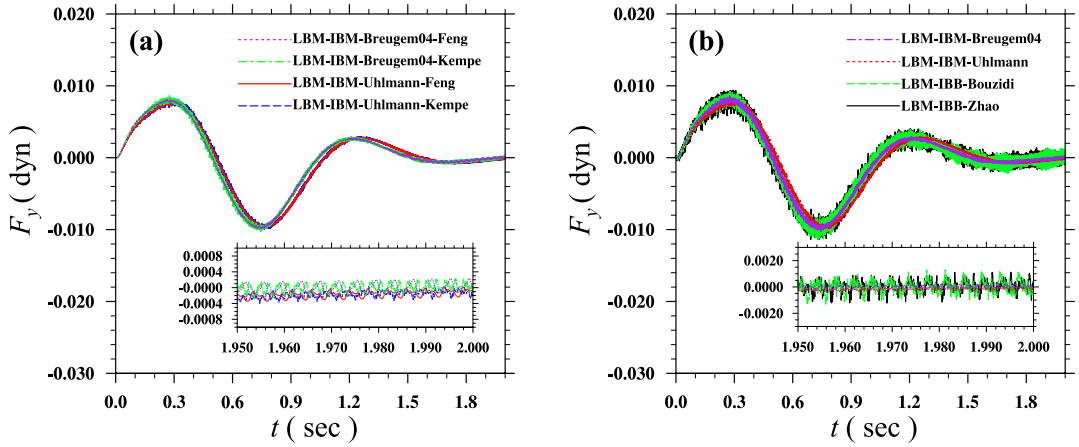

Fig. 14. The horizontal component of the hydrodynamic force acting on the particle, $D = 30$, (a) the effects of treatments on the inertia of fluid inside the solid region, (b) the comparison between LBM-IBB and LBM-IBM.

numerical stability, less diffusive delta-functions should be recommended. We notice that many finite-volume based IBM studies (*e.g.*, Ref. [6,7]) recommended the three-point delta-function by Roma et al. [43], perhaps due to the balance between its ability to suppress the force fluctuation and acceptable boundary diffusion. In our LBM based IBM simulations, however, we found the use of three-point delta-function leads to larger vulnerability for numerical instability (see Section 3.1). We therefore recommend the four-point delta-function by Peskin [5] instead.

Another source for the larger fluctuation in IBB is due to the requirement that the distribution functions at a new grid point when it is uncovered by the cylinder need to be initialized (known as "refilling"), since no distribution function is assigned to nodes inside the solid region when IBB is used. Proper refilling schemes may reduce fluctuation but its contribution cannot be removed [39]. LBM-IBM, on the other hand, avoids the refilling process, as the whole computational domain is filled with fluid and assigned with distribution functions.

3.3. Transient laminar pipe flow

We now move to the two three-dimensional problems, with the purpose of further supporting the remarks that have been made using the two-dimensional flows discussed above. The first 3D flow is the transient laminar pipe flow. Strictly speaking, this flow is a two-dimensional flow, but is run on three-dimensional Cartesian grids. Under a constant driving force, the flow that is initially static in a circular pipe accelerates and reaches a steady state. The governing equation of this axi-symmetric flow reads as

$$\frac{\partial u_z}{\partial t} = \nu \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u_z}{\partial r}\right) + g, \tag{27}$$

where u_z is the streamwise velocity in a cylindrical coordinate system (r, θ, z), g is the constant body force driving the flow. Applying periodic boundary condition in the streamwise direction and no-slip condition on the pipe wall, the above governing equation can be solved theoretically to obtain as Wang and Du [52]

$$u_z(r,t) = u_0\left[\left(1-\frac{r^2}{R^2}\right) - \sum_{n=1}^{\infty}\frac{8J_0(\lambda_n r/R)}{\lambda_n^3 J_1(\lambda_n)} exp\left(-\frac{\lambda_n^2 \nu t}{R^2}\right)\right], \tag{28}$$

where $u_0 = gR^2/4\nu$ is the centerline velocity at the steady state, J_0 and J_1 are the Bessel function of the first kind J_α for integer orders $\alpha = 0$ and $\alpha = 1$, λ_n is the nth root for J_0.

First, the velocity contours and profiles at the steady state with (a) Zhao & Yong's bounce-back, (b) Breugem's IBM with $R_d = 0.4$, where the driving body force is applied only to the fluid domain, (c) same as (b) but the driving body force is applied to the whole computational domain are shown in Fig. 15. The Reynolds number of the flow $Re = 2u_0R/\nu$ is 100, the radius of the pipe $R = 30\delta x$. The pipe is contained in a computational domain of $n_x \times n_y \times n_z = 72 \times 72 \times 16$. While the velocity profiles with Zhao & Yong's bounce-back collapse well with the theoretical solutions at different times, those profiles with Breugem's IBM have slight visible deviations from the theory at later times. Comparing the velocity profiles in case (b) and case (c), we observe that applying the driving force in the fictitious fluid domain (physical domain occupied by solid phase) leads to larger derivations than restricting the driving force in the physical fluid domain. This again suggests that how to appropriately treat the flow in the fictitious domain affects the accuracy of flow in the physical fluid domain, as the two parts can directly exchange information via advection and diffusion through the N-S equations.

In order to better quantify the numerical errors in the laminar pipe flow simulations, the convergence rates of L1 and L2 errors of the steady state velocity are calculated and presented in Fig. 16. Here we examine six boundary treatment schemes, the linear interpolated bounce-back schemes of Bouzidi et al. and Yu et al., Zhao & Yong's bounce-back scheme, Uhlmann's IBM, Breugem's IBM with retraction distances of $R_d = 0.3\delta x$ and $0.4\delta x$. The boundary forces in three simulations with IBM are iterated for 5 times. Similar to the case of circular Couette flow, numerical errors in the three cases with IBM generally have first-order convergence rate, in contrast to the second-order convergence rates in the three IBB cases. While retracting the Lagrangian grid to the solid side significantly reduces the magnitude of the numerical error, the convergence rate is only slightly improved, i.e., from 1.0 to 1.2 with the retraction distance of $R_d = 0.4\delta x$. According to our derivation in Appendix B, as long as the actual boundary is diffused more than the retraction distance by the delta-function, the first-order error induced by the interpolation should always involve the flow in both the fluid and solid regions. Therefore, the improved order of accuracy claimed in previous IBM studies, *e.g.*, in [7], may not be generalizable, as we are unable to reproduce the second-order accuracy with the LBM-IBM here.

3.4. Uniform flow passing a fixed sphere

The last case we examine is a uniform stream passing a fixed sphere in an unbounded domain. A spherical particle with a diameter D is fixed at $(x, y, z) = (6D, 6D, 6D)$ in a cuboid domain of a size $(Lx, Ly, Lz) = (24D, 12D, 12D)$. A uniform unidirectional up-

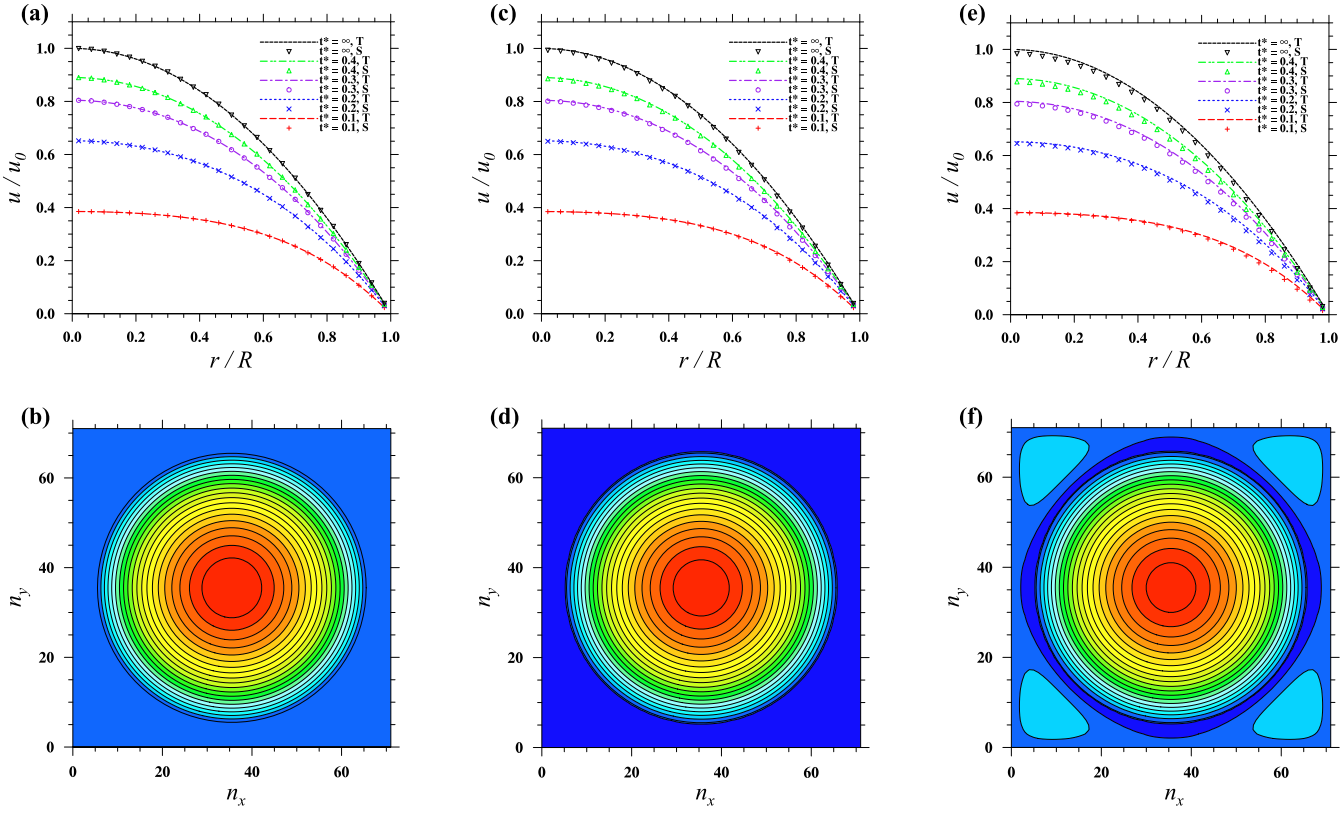

Fig. 15. The velocity contours and profiles of a laminar pipe flow at its steady state: (a) and (d), with Zhao & Yong's bounce-back; (b) and (e), with Breugem's IBM with a retraction distance of $R_d = 0.4$, the driving force only applied to the fluid region, (c) and (f), same as (b) and (e), but the driving force applied to the whole computational domain.

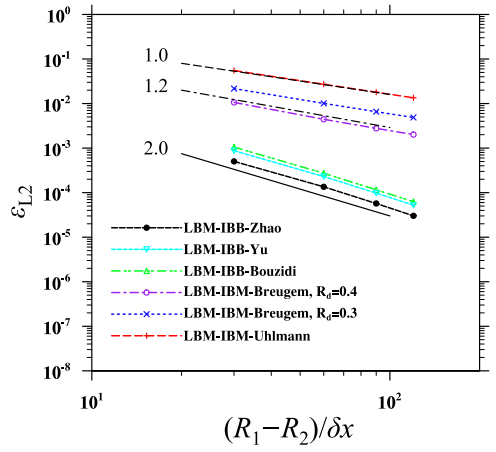

Fig. 16. The convergence rates of the velocity in a laminar pipe flow.

stream flow with $\vec{u} = (u_x, u_y, u_z) = (U_i, 0, 0)$ enters the inlet ($x = 0$) of the domain and passes over the fixed particle. The four sides are set to be stress-free, *i.e.*, $u_z = 0$, $\partial u_x/\partial z = \partial u_y/\partial z = 0$ at $z = 0$ and $z = Lz$, and $u_y = 0$, $\partial u_x/\partial y = \partial u_z/\partial y = 0$ at $y = 0$ and $y = Ly$, to mimic the boundary condition in an infinitely large domain. The flow exits the domain with the following outflow boundary condition, $\partial(\rho_0\vec{u})/\partial t + U_o\partial(\rho_0\vec{u})/\partial x = 0$, where U_o is the streamwise velocity at the outlet [53].

The drag coefficients under three different particle Reynolds numbers, $Re_p = U_i D/\nu = 20$, 50, and 150 are examined. With these Reynolds numbers, the flow after the sphere is steady and axisymmetric with closed recirculating wake [54]. At each Reynolds number, we vary the grid resolution, *i.e.*, $D/\delta x$ from 8 to 48, and investigate the drag coefficient $C_D = 8F_D/(\rho_f Re_p^2 \pi \nu^2)$, with kinematic viscosity ν fixed when varying the grid resolution. The results of drag coefficient at $Re_p = 20$, 50, and 150 are presented in Figs. 17–19, respectively. The vertical solid lines in each figure indicate the grid resolution gives an error of 1% using the result of the current boundary treatment with the highest grid resolution as benchmark. At all three Reynolds numbers, Zhao & Yong's bounce-back scheme always reaches the converged drag coefficient with the coarsest grid resolution among all four boundary treatments. This is perhaps due to the fact that Zhao & Yong's bounce-back scheme has a second-order accuracy while the immersed boundary algorithms only have first-order accuracy. The retraction of Lagrangian grid points again results in much more accurate results compared to zero retraction distance. $R_d = 0.4\delta x$ always appears to be the optimal retraction distance when the four-point delta-function is employed. According to our results, We recommend that Breugem's Lagrangian grid retraction be used when IBM is used for no-slip boundary treatment.

Finally, if we define a "sufficient" grid resolution as the grid resolution that gives 1% relative error from the converged result, the sufficient grid resolutions for Zhao & Yong's bounce-back at $Re_p = 20$, 50, and 150 are about $D\delta x = 14.3$, 16.7 and 15.9. The same quantities are 37.1, 38.0, 36.3 with Uhlmann's IBM, 29.5, 30.7, 25.0 with Breugem's IBM with $R_d = 0.3\delta x$, and 25.0, 26.3, 20.8 with Breguem's IBM with $R_d = 0.4\delta x$. These results may provide a criterion to assess whether a grid resolution is fine enough to ensure trustworthy results when a certain scheme is adopted for no-slip boundary treatment in a three-dimensional particle-laden flow simulation, at the similar particle Reynolds number.

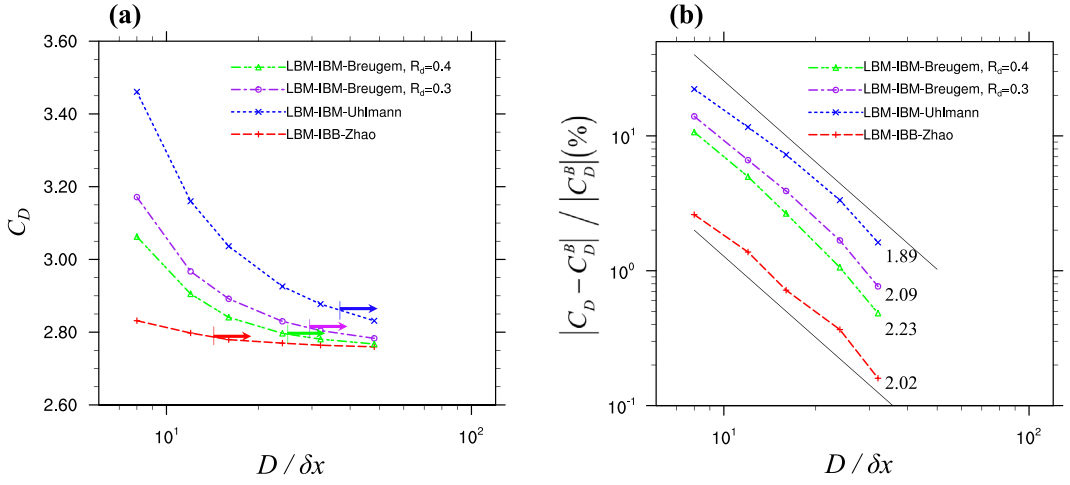

Fig. 17. The drag coefficients of a uniform flow passing a fixed sphere at $Re_p = 20$.

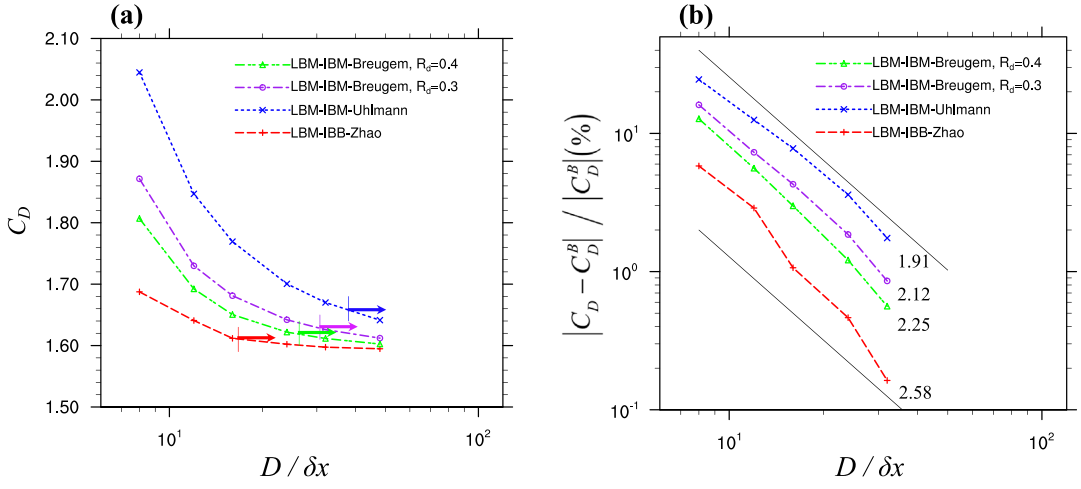

Fig. 18. The drag coefficients of a uniform flow passing a fixed sphere at $Re_p = 50$.

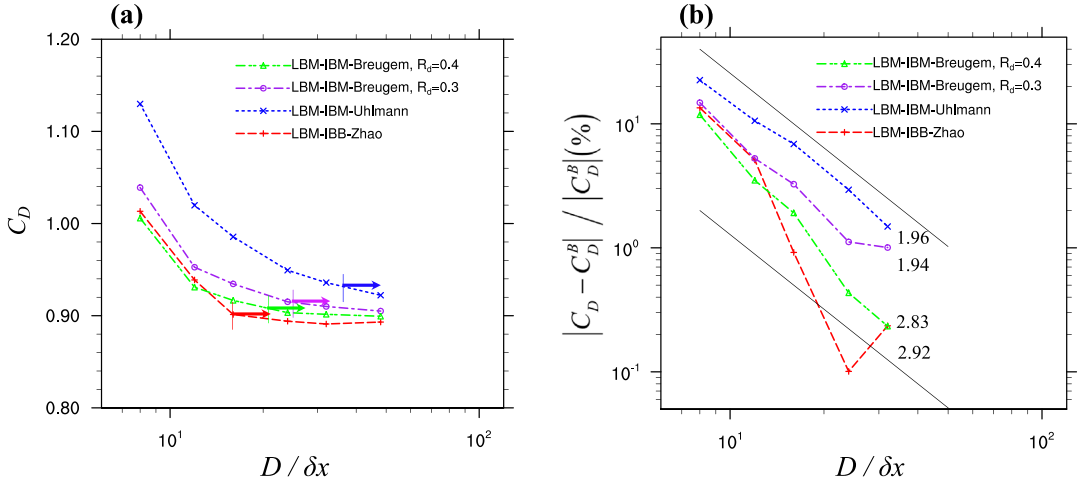

Fig. 19. The drag coefficients of a uniform flow passing a fixed sphere at $Re_p = 150$.

4. Conclusions and remarks

In this work, we systematically assessed two categories of no-slip boundary treatment methods, which are the interpolated bounce-back schemes and the immersed boundary method, on an arbitrarily shaped surface in the context of the lattice Boltzmann method. Three representative interpolated bounce-back schemes, including a recently proposed single-node second-order bounce-back scheme [22], and two popular immersed boundary algorithms are selected. Their performances, especially the accuracy of resulting velocity, hydrodynamic force/torque, and the viscous dissipation rate are carefully benchmarked in four selected flows. In all the flows examined in the present study, the interpolated bounce-back schemes always lead to much more accurate results of velocity, force/torque, and dissipation rate than the immersed boundary algorithms. The immersed boundary algorithms, on the other hand, outperform in suppressing the fluctuations of the calculated hydrodynamic force/torque compared to the interpolated bounce-back schemes. The specific major observations of this present study are summarized as follows.

- With immersed boundary algorithms, cautions should be taken to the treatment of the flow in the virtual fluid region, especially when the flow of interest is surrounded by solid objects, such as the Taylor-Couette flow and the circular pipe flow. Unfortunately, the information of how to specify the flow in the virtual fluid region may not be available in the description of physical problems.
- Our simulations confirm that the immersed boundary algorithms using the regularized delta-functions to interpolate information between the Eulerian and Lagrangian grids have only the first-order accuracy in flow velocity calculation. This conclusion holds no matter whether the Lagrangian grid points are retracted towards the solid phase or not. We prove with a theoretical analysis that information exchange between the Eulerian and Lagrangian grids via the regularized delta-function always induces a first-order error term as long as the velocity gradient is discontinuous across the solid-fluid interface. On the other hand, the interpolated bounce-back schemes can ensure a second-order accuracy of the simulated velocity. The magnitudes of the velocity errors in the interpolated bounce-back schemes are also much smaller than their counterparts from immersed boundary algorithms.
- The *local* hydrodynamic force and torque calculated with immersed boundary algorithms are only first-order accurate. These local first-order errors may cancel out to result in an apparent second-order accurate integral force/torque, as shown in Section 3.4. However, this cancellation may not be generalized. In the Taylor-Couette flow, the integral force is still first-order accurate. The forces calculated with interpolated bounce-back schemes and momentum exchange methods have a second-order accuracy in all the flows examined.
- The most serious problem we find for the immersed boundary method is that the local dissipation rate can be significantly underestimated. This is because the sharp fluid-solid interface is diffused by the regularized delta-functions, which results in a smaller velocity gradient near the interface. The same problem is not present with the interpolated bounce-back schemes, which can be viewed as a sharp-interface treatment for no-slip boundary.
- For moving particle problems, the high-frequency fluctuations in the force/torque are better suppressed in the immersed boundary methods than the interpolated bounce-back schemes. When the particle/fluid density ratio is close to unity, both Feng & Michaelides's scheme and Kempe et al.'s scheme are suitable to update the particle motion.
- We present convergence studies to find out the sufficient grid resolution associated with each boundary treatment method for 2D circular and 3D spherical particles. Since the interpolated bounce-back schemes have better accuracy than the immersed boundary method, their grid resolution requirements for a converged result are lower. For 2D circular particles, the sufficient grid resolutions for using interpolated bounce-back schemes and the immersed boundary method are $D/\delta x = 10$ and $D/\delta x = 15$, respectively. For 3D spherical particles, the sufficient grid resolutions become $D/\delta x = 15$ and $D/\delta x = 25$, respectively, for particle Reynolds number between 20 to 150. Here the immersed boundary method refers to Breugem's IBM with an appropriate retraction distance. When Uhlmann's IBM with zero retraction distance is used, the sufficient grid resolution should be further increased.

Acknowledgments

This work has been supported by the National Natural Science Foundation of China (91852205 & 91741101), and by the U.S. National Science Foundation (NSF) under grants CNS1513031 and CBET-1706130. Computing resources are provided by Center for Computational Science and Engineering of Southern University of Science and Technology and by National Center for Atmospheric Research through CISL-P35751014, and CISL-UDEL0001.

Appendix A. Analytic solution of the Taylor-Couette flow

The analytic solution of the transient Taylor-Couette flow was derived by He [41]. Here we just repeat He's derivation for readers' convenience. The simplified N-S equations for the Taylor-Couette flow is written as

$$\frac{\partial u_\theta}{\partial t} = \nu \frac{\partial}{\partial r}\left[\frac{1}{r}\frac{\partial}{\partial r}(r u_\theta)\right], u_\theta(t, R_1) = \Omega_1 R_1,$$
$$u_\theta(t, R_2) = \Omega_2 R_2, \quad u_\theta(t = 0, r) = 0, \tag{29}$$

where u_θ is the flow velocity in the angular direction, ν is the kinematic viscosity of the fluid, t and r are the time and radius coordinate, respectively. R_1 and R_2 are the radius of the inner and outer cylinders confining the flow, Ω_1 and Ω_2 are the corresponding angular velocities, respectively. The time-dependent solution of this flow can be expressed as [41]:

$$u_\theta(r, t) = u_\theta^S + \sum_{n=1}^{\infty} A_n e^{-\frac{\nu \lambda_n^2 t}{(R_2 - R_1)^2}} \left[J_1\left(\frac{\lambda_n r}{R_1}\right) - \frac{J_1(\lambda_n)}{Y_1(\lambda_n)} Y_1\left(\frac{\lambda_n r}{R_1}\right)\right] \tag{30}$$

where u_θ^S is the steady state solution

$$u_\theta^S = \frac{1}{r}\frac{\Omega_1 - \Omega_2}{R_1^{-2} - R_2^{-2}} + \frac{\Omega_2 R_2^2 - \Omega_1 R_1^2}{R_2^2 - R_1^2} r, \tag{31}$$

J_1 and Y_1 are the first-order Bessel function of the first and the second kind, λ_n is the nth root satisfies

$$J_1(\lambda_n) Y_1(\lambda_n \gamma) - J_1(\lambda_n \gamma) Y_1(\lambda_n) = 0,$$
$$0 < \lambda_1 < \lambda_2 < \lambda_3 < \cdots < \lambda_n < \cdots \to \infty. \tag{32}$$

where $\gamma = R_2/R_1$ is the radii ratio between the outer and the inner cylinder. The nth coefficient of the series A_n is

$$A_n = \frac{\int_1^\gamma \left(-u_\theta^S\right) \frac{r}{R_1}\left[J_1\left(\frac{\lambda_n r}{R_1}\right) - \frac{J_1(\lambda_n)}{Y_1(\lambda_n)} Y_1\left(\frac{\lambda_n r}{R_1}\right)\right] dr}{\int_1^\gamma \frac{r}{R_1}\left[J_1\left(\frac{\lambda_n r}{R_1}\right) - \frac{J_1(\lambda_n)}{Y_1(\lambda_n)} Y_1\left(\frac{\lambda_n r}{R_1}\right)\right]^2 dr} = \frac{\text{top}}{\text{bottom}}. \tag{33}$$

The integrals in Eq. (33) can be calculated as

$$\begin{aligned}\text{top} = -\frac{1}{\lambda_n^2}[&c_2 J_0(\lambda_n)\lambda_n - 2c_1 J_1(\lambda_n) + c_1 J_0(\lambda_n)\lambda_n + 2\alpha c_1 Y_1(\lambda_n)\\ &-\alpha c_1 Y_0(\lambda_n)\lambda_n - c_2\alpha Y_0(\lambda_n)\lambda_n - c_2 J_0(\lambda_n\gamma)\lambda_n + 2c_1\gamma J_1(\lambda_n\gamma)\\ &-c_1 J_0(\lambda_n\gamma)\lambda_n\gamma^2 - 2\alpha c_1\gamma Y_1(\lambda_n\gamma) + \alpha c_1 Y_0(\lambda_n\gamma)\lambda_n\gamma^2\\ &+c_2\alpha Y_0(\lambda_n\gamma)\lambda_n] \end{aligned} \tag{34a}$$

$$\begin{aligned}\text{bottom} = -\frac{1}{2\lambda_n}\{&\lambda_n[J_1(\lambda_n)]^2 - 2J_0(\lambda_n)J_1(\lambda_n) + [J_0(\lambda_n)]^2\lambda_n\\ &-2\alpha J_1(\lambda_n)Y_1(\lambda_n)\lambda_n + 2\alpha Y_0(\lambda_n)J_1(\lambda_n)\\ &-2\alpha Y_0(\lambda_n)J_0(\lambda_n)\lambda_n\\ &+2\alpha J_0(\lambda_n)Y_1(\lambda_n) + \alpha^2[Y_1(\lambda_n)]^2\lambda_n - 2\alpha^2 Y_0(\lambda_n)Y_1(\lambda_n)\\ &+\alpha^2[Y_0(\lambda_n)]^2\lambda_n - \gamma^2[J_1(\lambda_n\gamma)]^2\lambda_n + 2\gamma J_0(\lambda_n\gamma)J_1(\lambda_n\gamma)\\ &-\gamma^2[J_0(\lambda_n\gamma)]^2\lambda_n + 2\alpha\gamma^2 J_1(\lambda_n\gamma)Y_1(\lambda_n\gamma)\lambda_n\\ &-2\gamma\alpha Y_0(\lambda_n\gamma)J_1(\lambda_n\gamma)\\ &+2\alpha\gamma^2 Y_0(\lambda_n\gamma)J_0(\lambda_n\gamma)\lambda_n - 2\gamma\alpha J_0(\lambda_n\gamma)Y_1(\lambda_n\gamma)\\ &-\alpha^2\gamma^2[Y_1(\lambda_n\gamma)]^2\lambda_n\\ &2\gamma\alpha^2 Y_0(\lambda_n\gamma)Y_1(\lambda_n\gamma) - \alpha^2\gamma^2[Y_0(\lambda_n\gamma)]^2\lambda_n\} \end{aligned} \tag{34b}$$

where

$$c_1 = \frac{\Omega_2 R_2\gamma - \Omega_1 R_1}{\gamma^2 - 1}, \quad c_2 = \frac{\Omega_1 R_1\gamma^2 - \Omega_2 R_2\gamma}{\gamma^2 - 1}, \quad \alpha = \frac{J_1(\lambda_n)}{Y_1(\lambda_n)}. \tag{35}$$

Appendix B. A theoretical examination on the order of accuracy of immersed boundary method

A configuration of Lagrangian–Eulerian grid system for a one-dimensional fluid-solid interface is sketched in Fig. 20, where x_1 and x_2 are two Eulerian grid points on the fluid side and the solid side of the Lagrangian grid point X. The interpolation of unforced velocity from the Eulerian grid to the Lagrangian grid and the re-distribution of the boundary force the other way around take place at x_1, x_2, and X. Let us assume the velocity field prior to applying the boundary forcing is accurate, *i.e.*,

$$\vec{u}(x_1) = \vec{u}_{exact}(x_1), \qquad \vec{u}(x_2) = \vec{u}_{exact}(x_2). \tag{36}$$

The boundary force on the Lagrangian point $\vec{\mathcal{F}}(X)$ would be

$$\begin{aligned}\vec{\mathcal{F}}(X)\delta t &= \rho\big[\vec{U}(X) - \vec{u}(X)\big]\\ &= \rho\big[\vec{U}(X) - \phi_1\vec{u}_{exact}(x_1) - \phi_2\vec{u}_{exact}(x_2)\big], \end{aligned} \tag{37}$$

where $\phi_1 = \phi(x_1 - X)$ and $\phi_2 = \phi(x_2 - X)$ are the weighting factors obtained from the delta-function, $\phi_1 + \phi_2 = 1$. The boundary forces distributed back to x_1 and x_2 are

$$\vec{\mathcal{F}}(x_1)\delta t = \phi_1\vec{\mathcal{F}}(X)\delta t = \phi_1\rho\big[\vec{U}(X) - \phi_1\vec{u}_{exact}(x_1) - \phi_2\vec{u}_{exact}(x_2)\big], \tag{38a}$$

$$\vec{\mathcal{F}}(x_2)\delta t = \phi_2\vec{\mathcal{F}}(X)\delta t = \phi_2\rho\big[\vec{U}(X) - \phi_1\vec{u}_{exact}(x_1) - \phi_2\vec{u}_{exact}(x_2)\big], \tag{38b}$$

After applying the boundary forcing, the velocity at x_1 and x_2 are

$$\tilde{\vec{u}}(x_1) = \vec{u}_{exact}(x_1) + \frac{1}{\rho}\vec{\mathcal{F}}(x_1)\delta t, \qquad \tilde{\vec{u}}(x_2) = \vec{u}_{exact}(x_2) + \frac{1}{\rho}\vec{\mathcal{F}}(x_2)\delta t. \tag{39}$$

Since the velocity field before applying the boundary forcing is already exact, the errors introduced by the boundary force at x_1 and x_2 are simply

$$\begin{aligned}\Delta\vec{u}_1 &= \tilde{\vec{u}}(x_1) - \vec{u}_{exact}(x_1)\\ &= \phi_1\rho\big[\vec{U}(X) - \phi_1\vec{u}_{exact}(x_1) - \phi_2\vec{u}_{exact}(x_2)\big], \end{aligned} \tag{40a}$$

$$\begin{aligned}\Delta\vec{u}_2 &= \tilde{\vec{u}}(x_2) - \vec{u}_{exact}(x_2)\\ &= \phi_2\rho\big[\vec{U}(X) - \phi_1\vec{u}_{exact}(x_1) - \phi_2\vec{u}_{exact}(x_2)\big]. \end{aligned} \tag{40b}$$

Performing a Taylor expansion for $\vec{u}_{exact}(x_1)$ and $\vec{u}_{exact}(x_2)$ with respect to X, *i.e.*,

$$\begin{aligned}\vec{u}_{exact}(x_1) &= \vec{U}(X) + \frac{d\vec{u}}{dx}|_{fluid}\,(x_1 - X) + O\big(\Delta x^2\big),\\ \vec{u}_{exact}(x_2) &= \vec{U}(X) + \frac{d\vec{u}}{dx}|_{solid}\,(x_2 - X) + O\big(\Delta x^2\big), \end{aligned} \tag{41}$$

Substitute Eq. (41) to Eq. (40), we shall obtain

$$\begin{aligned}\Delta\vec{u}_1 &= \phi_1\left\{-\phi_1\left[\frac{d\vec{u}}{dx}|_{fluid}\,(x_1 - X) + O\big(\Delta x^2\big)\right]\right.\\ &\quad\left.-\phi_2\left[\frac{d\vec{u}}{dx}|_{solid}\,(x_2 - X) + O\big(\Delta x^2\big)\right]\right\},\\ \Delta\vec{u}_2 &= \phi_2\left\{-\phi_1\left[\frac{d\vec{u}}{dx}|_{fluid}\,(x_1 - X) + O\big(\Delta x^2\big)\right]\right.\\ &\quad\left.-\phi_2\left[\frac{d\vec{u}}{dx}|_{solid}\,(x_2 - X) + O\big(\Delta x^2\big)\right]\right\}. \end{aligned} \tag{42}$$

Note that the delta-function should have the property $\sum(x - X)\phi(x - X) = 0$[2], *i.e.*, $\phi_1(x_1 - X) + \phi_2(x_2 - X) = 0$. To simplify the notation, denote $\phi_1(x_1 - X) = c$, $\phi_2(x_2 - X) = -c$, $c \sim O(\Delta x)$, Eq. (42) becomes

$$\begin{aligned}\Delta\vec{u}_1 &= -c\phi_1\left(\frac{d\vec{u}}{dx}|_{fluid} - \frac{d\vec{u}}{dx}|_{solid}\right) + O\big(\Delta x^2\big),\\ \Delta\vec{u}_2 &= -c\phi_2\left(\frac{d\vec{u}}{dx}|_{fluid} - \frac{d\vec{u}}{dx}|_{solid}\right) + O\big(\Delta x^2\big). \end{aligned} \tag{43}$$

Therefore, only when the velocity gradient is continuous, according to the Taylor expansion

$$\frac{d\vec{u}}{dx}|_{solid} = \frac{d\vec{u}}{dx}|_{fluid} + O(\Delta x), \tag{44}$$

$\Delta\vec{u}_1$ and $\Delta\vec{u}_2$ in Eq. (43) can then have a second-order accuracy. When the velocity gradient is discontinuous, the boundary

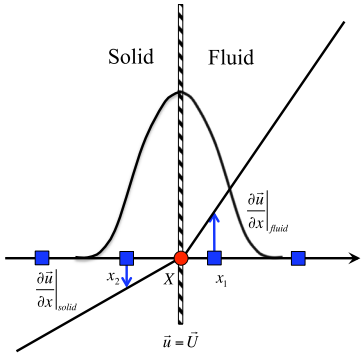

Fig. 20. The grid arrangement at a fluid-solid interface.

[2] The 4-point cosine delta-function employed frequently does not possess this property strictly, but it does not affect the argument.

forcing process shown above always induces a first-order error to the velocity field. Generally speaking, the velocity gradient is, unfortunately, not continuous across a fluid-solid interface, thus IBM using the delta-function degrades the accuracy to the first-order.

In IBM, The hydrodynamic force $\vec{F}$ on a solid object is calculated as

$$\vec{F}\delta t = \sum_l \vec{\mathcal{F}}(X_l)\delta t \Delta V_l, \tag{45}$$

where X_l is the location of the lth Lagrangian grid point, ΔV_l is the control volume of X_l. Similarly, we can define the exact hydrodynamic force as

$$\vec{F}_{exact}\delta t = \sum_l \vec{\mathcal{F}}_{exact}(X_l)\delta t \Delta V_l, \tag{46}$$

The error of hydrodynamic force in IBM is simply the difference between the two, *i.e.*,

$$\begin{aligned}\Delta\vec{F}\delta t &= \vec{F}\delta t - \vec{F}_{exact}\delta t = \sum_l \left[\vec{\mathcal{F}}(X_l) - \vec{\mathcal{F}}_{exact}(X_l)\right]\delta t \Delta V_l \\ &= \sum_l \left\{\left[\vec{U}(X_l) - \phi_1\vec{u}(x_1) - \phi_2\vec{u}(x_2)\right]\right. \\ &\quad \left. - \left[\vec{U}(X_l) - \phi_1\vec{u}_{exact}(x_1) - \phi_2\vec{u}_{exact}(x_2)\right]\right\}\Delta V_l \\ &= -\sum_l [\phi_1\Delta\vec{u}(x_1) + \phi_2\Delta\vec{u}(x_2)]\Delta V_l. \end{aligned} \tag{47}$$

Substituting $\Delta\vec{u}(x_1)$ and $\Delta\vec{u}(x_2)$ obtained in Eq. (43) results in

$$\Delta\vec{F}\delta t = -\sum_l \left[\left(c\phi_1^2 + c\phi_2^2\right)\left(\frac{d\vec{u}}{dx}\,|_{fluid} - \frac{d\vec{u}}{dx}\,|_{solid}\right) + O\left(\Delta x^2\right)\right]\Delta V_l, \tag{48}$$

where $c\phi_1^2 + c\phi_2^2$ is always positive (or negative) and on the order of Δx. Evidently, for the local first-order error in Eq. (48) to cancel out in the summation, the difference of the velocity derivatives across the fluid-solid interface must be follow certain patterns, or at least being positive on some Lagrangian nodes and being negative on the others. While we do observe this situation in the case of a uniform flow passing a fixed sphere, which has also been reported in the literature [7,18,21], this observation may not be generalized. In the case of Taylor-Couette flow, the hydrodynamic force calculation with IBM is only first-order accurate.

References

[1] Pan C, Luo L-S, Miller CT. An evaluation of lattice Boltzmann schemes for porous medium flow simulation 2006;35(8–9):898–909.
[2] Tian F-B, Luo H, Zhu L, Liao JC, Lu X-Y. An efficient immersed boundary-lattice Boltzmann method for the hydrodynamic interaction of elastic filaments. J Comput Phys 2011;230(19):7266–83.
[3] Wang L-P, Peng C, Guo Z, Yu Z. Lattice Boltzmann simulation of particle-laden turbulent channel flow. Comput Fluid 2016;124:226–36.
[4] Eshghinejadfard A, Abdelsamie A, Hosseini SA, Thévenin D. Immersed boundary lattice Boltzmann simulation of turbulent channel flows in the presence of spherical particles. Int J Multiph Flow 2017;96:161–72.
[5] Peskin CS. The immersed boundary method. Acta Numer 2002;11:479–517.
[6] Uhlmann M. An immersed boundary method with direct forcing for the simulation of particulate flows. J Comput Phys 2005;209(2):448–76.
[7] Breugem W-P. A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows. J Comput Phys 2012;231(13):4469–98.
[8] Feng Z-G, Michaelides EE. Proteus: a direct forcing method in the simulations of particulate flows. J Comput Phys 2005;202(1):20–51.
[9] Wu J, Shu C. Implicit velocity correction-based immersed boundary-lattice Boltzmann method and its applications. J Comput Phys 2009;228(6):1963–79.
[10] Goldstein D, Handler R, Sirovich L. Modeling a no-slip flow boundary with an external force field. J Comput Phys 1993;105(2):354–66.
[11] Fadlun E, Verzicco R, Orlandi P, Mohd-Yusof J. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. J Comput Phys 2000;161(1):35–60.
[12] Ladd AJC. Numerical simulations of particulate suspensions via a discretized Boltzmann equation. part 1. theoretical foundation. J Fluid Mech 1994;271(1):285–309.
[13] Bouzidi M, Firdaouss M, Lallemand P. Momentum transfer of a Boltzmann-lattice fluid with boundaries. Phys Fluid 2001;13(11):3452–9.
[14] Mei R, Shyy W, Yu D, Luo L-S. Lattice Boltzmann method for 3-D flows with curved boundary. J Comput Phys 2000;161(2):680–99.
[15] Guo Z, Zheng C, Shi B. An extrapolation method for boundary conditions in lattice Boltzmann method. Phys Fluid 2002;14(6):2007–10.
[16] Yu D, Mei R, Luo L-S, Shyy W. Viscous flow computations with the method of lattice Boltzmann equation. Prog Aerosp Sci 2003;39(5):329–67.
[17] Ginzburg I, dHumières D. Multireflection boundary conditions for lattice Boltzmann models. Phys Rev E 2003;68(6):066614.
[18] Peng Y, Luo L-S. A comparative study of immersed-boundary and interpolated bounce-back methods in LBE. Prog Comput Fluid Dyn Int J 2008;8(1–4):156–67.
[19] Chen L, Yu Y, Lu J, Hou G. A comparative study of lattice Boltzmann methods using bounce-back schemes and immersed boundary ones for flow acoustic problems. Int J Numer Method Fluid 2014;74(6):439–67.
[20] Kang SK, Hassan YA. A comparative study of direct-forcing immersed boundary-lattice Boltzmann methods for stationary complex boundaries. Int J Numer Method Fluid 2011;66(9):1132–58.
[21] Zhou Q, Fan L-S. A second-order accurate immersed boundary-lattice Boltzmann method for particle-laden flows. J Comput Phys 2014;268:269–301.
[22] Zhao W, Yong W-A. Single-node second-order boundary schemes for the lattice Boltzmann method. J Comput Phys 2017;329(6):1–15.
[23] Uhlmann M. Interface-resolved direct numerical simulation of vertical particulate channel flow in the turbulent regime. Phys Fluid 2008;20(5):053305.
[24] Picano F, Breugem W-P, Brandt L. Turbulent channel flow of dense suspensions of neutrally buoyant spheres. J Fluid Mech 2015;764:463–87.
[25] Peng C. Study of turbulence modulation by finite-size solid particles with the lattice Boltzmann method. the University of Delaware; 2018. Ph.d. dissertation.
[26] Guo Z, Shu C. Lattice Boltzmann method and its applications in engineering, 3. World Scientific; 2013.
[27] Lallemand P, Luo L-S. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability. Phys Rev E 2000;61(6):6546.
[28] Guo Z, Zheng C, Shi B. Discrete lattice effects on the forcing term in the lattice Boltzmann method. Phys Rev E 2002;65(4):046308.
[29] Feng Z-G, Michaelides EE. The immersed boundary-lattice Boltzmann method for solving fluid–particles interaction problems. J Comput Phys 2004;195(2):602–28.
[30] Niu X, Shu C, Chew Y, Peng Y. A momentum exchange-based immersed boundary-lattice Boltzmann method for simulating incompressible viscous flows. Phys Lett A 2006;354(3):173–82.
[31] Dupuis A, Chatelain P, Koumoutsakos P. An immersed boundary–lattice-Boltzmann method for the simulation of the flow past an impulsively started cylinder. J Comput Phys 2008;227(9):4486–98.
[32] Zhang C, Cheng Y, Zhu L, Wu J. Accuracy improvement of the immersed boundary–lattice Boltzmann coupling scheme by iterative force correction. Comput Fluid 2016;124:246–60.
[33] Cheng Y, Li J. Introducing unsteady non-uniform source terms into the lattice Boltzmann model. Int J Numer Method Fluid 2008;56(6):629–41.
[34] Min H, Peng C, Guo Z, Wang L-P. An inverse design analysis of mesoscopic implementation of non-uniform forcing in MRT lattice Boltzmann models. Comput Math Applica 2016.
[35] Tao S, He Q, Chen B, Yang X, Huang S. One-point second-order curved boundary condition for lattice Boltzmann simulation of suspended particles. Comput Math Applica 2018.
[36] Mei R, Yu D, Shyy W, Luo L-S. Force evaluation in the lattice Boltzmann method involving curved geometry. Phys Rev E 2002;65(4):041203.
[37] Wen B, Zhang C, Tu Y, Wang C, Fang H. Galilean invariant fluid–solid interfacial dynamics in lattice Boltzmann simulations. J Comput Phys 2014;266:161–170.
[38] Peng C, Geneva N, Guo Z, Wang L-P. Issues associated with Galilean invariance on a moving solid boundary in the lattice Boltzmann method. Phys Rev E 2017;95(1):013301.
[39] Peng C, Teng Y, Hwang B, Guo Z, Wang L-P. Implementation issues and benchmarking of lattice Boltzmann method for moving rigid particle simulations in a viscous flow. Comput Math Applica 2016;72(2):349–74.
[40] Tao S, Hu J, Guo Z. An investigation on momentum exchange methods and refilling algorithms for lattice Boltzmann simulation of particulate flows. Comput Fluid 2016;133:1–14.
[41] He Z. High order smoothed particle hydrodynamic methods for slightly compressible bounded flow. the University of Delaware; 2015. Ph.D. dissertation.
[42] Lai M-C, Li Z. A remark on jump conditions for the three-dimensional Navier-Stokes equations involving an immersed moving membrane. Appl Math Lett 2001;14(2):149–54.
[43] Roma AM, Peskin CS, Berger MJ. An adaptive version of the immersed boundary method. J Comput Phys 1999;153(2):509–34.
[44] Tennekes H, Lumley JL. A first course in turbulence. MIT press; 1972.
[45] Yong W-A, Luo L-S. Accuracy of the viscous stress in the lattice Boltzmann equation with simple boundary conditions. Phys Rev E 2012;86(6):065701.
[46] Peng C, Guo Z, Wang L-P. Lattice Boltzmann model capable of mesoscopic vorticity computation. Phys Rev E 2017;96:053304.

[47] Feng Z-G, Michaelides EE. Robust treatment of no-slip boundary condition and velocity updating for the lattice-Boltzmann simulation of particulate flows. Comput Fluid 2009;38(2):370–81.
[48] Kempe T, Fröhlich J. An improved immersed boundary method with direct forcing for the simulation of particle laden flows. J Comput Phys 2012;231(9):3663–84.
[49] Hu HH, Patankar NA, Zhu M. Direct numerical simulations of fluid-solid systems using the arbitrary Lagrangian-Eulerian technique. J Comput Phys 2001;169(2):427–62.
[50] Yang X, Zhang X, Li Z, He G-W. A smoothing technique for discrete delta functions with application to immersed boundary method in moving boundary simulations. J Comput Phys 2009;228(20):7821–36.
[51] Lallemand P, Luo L-S. Lattice Boltzmann method for moving boundaries. J Comput Phys 2003;184(2):406–21.
[52] Wang L-P, Du MH. Direct simulation of viscous flow in a wavy pipe using the lattice Boltzmann approach. Int J Eng Syst Model Simul 2008;1(1):20–9.
[53] Lou Q, Guo Z, Shi B. Evaluation of outflow boundary conditions for two-phase lattice boltzmann equation. Phys Rev E 2013;87(6):063301.
[54] Jones D, Clarke D. Defense Science and Technology Organization Victoria (Australia) Maritime Platforms Div; 2008. Tech. Rep.