Study of Particle Rotation Effect in Gas-Solid Flows using Direct Numerical Simulation with a Lattice Boltzmann Method

Final Technical Report

10/1/2011 - 9/30/2014

Kyung C Kwon[1], Liang-Shih Fan[2], Qiang Zhou[2], and Hui Yang[2]

December 2014

DOE Award Number:DE-FE0007520

1. Chemical Engineering Department, Tuskegee University, 514 Luther H Foster Hall, University Ave, Tuskegee, AL 36088,

2. Department of Chemical and Biomolecular Engineering, Ohio State University, 125A Koffolt Laboratories, 140 West 19th Ave, Columbus, OH 43210

DISCLAIMER:

ABSTRACT

A new and efficient direct numerical method with second-order convergence accuracy was developed for fully resolved simulations of incompressible viscous flows laden with rigid particles. The method combines the state-of-the-art immersed boundary method (IBM), the multi-direct forcing method, and the lattice Boltzmann method (LBM). First, the multi-direct forcing method is adopted in the improved IBM to better approximate the no-slip/no-penetration (ns/np) condition on the surface of particles. Second, a slight retraction of the Lagrangian grid from the surface towards the interior of particles with a fraction of the Eulerian grid spacing helps increase the convergence accuracy of the method.

An over-relaxation technique in the procedure of multi-direct forcing method and the classical fourth order Runge-Kutta scheme in the coupled fluid-particle interaction were applied. The use of the classical fourth order Runge-Kutta scheme helps the overall IB-LBM achieve the second order accuracy and provides more accurate predictions of the translational and rotational motion of particles. The preexistent code with the first-order convergence rate is updated so that the updated new code can resolve the translational and rotational motion of particles with the second-order convergence rate. The updated code has been validated with several benchmark applications.

The efficiency of IBM and thus the efficiency of IB-LBM were improved by reducing the number of the Lagragian markers on particles by using a new formula for the number of Lagrangian markers on particle surfaces. The immersed boundary-lattice Boltzmann method (IB-LBM) has been shown to predict correctly the angular velocity of a particle. Prior to examining drag force exerted on a cluster of particles, the updated IB-LBM code along with the new formula for the number of Lagrangian markers has been further validated by solving several theoretical problems. Moreover, the unsteadiness of the drag force is examined when a fluid is accelerated from rest by a constant average pressure gradient toward a steady Stokes flow. The simulation results agree well with the theories for the short- and long-time behavior of the drag force.

Flows through non-rotational and rotational spheres in simple cubic arrays and random arrays are simulated over the entire range of packing fractions, and both low and moderate particle Reynolds numbers to compare the simulated results with the literature results and develop a new drag force formula, a new lift force formula, and a new torque formula. Random arrays of solid particles in fluids are generated with Monte Carlo procedure and Zinchenko's method to avoid crystallization of solid particles over high solid volume fractions.

A new drag force formula was developed with extensive simulated results to be closely applicable to real processes over the entire range of packing fractions and both low and moderate particle Reynolds numbers. The simulation results indicate that the drag force is barely affected by rotational Reynolds numbers. Drag force is basically unchanged as the angle of the rotating axis varies.

A new lift force formula was developed with simulation results comprehensively over the rotational Reynolds numbers of 0.1 to 500, and solid volume fractions up to the close-packed limits, and low and moderate particle Reynolds numbers. The simulation results indicate that the lift force produced by the rotation movement is directly proportional to rotational Reynolds numbers defined by the angular velocity perpendicular with the flow direction. The lift force is very insignificant for ordered arrays and random arrays at the rotational Reynolds number below 1. However, the lift force at especially low solid volume fractions can be larger than the drag force as the rotational Reynolds number increases.

The torque exerted on spheres by a fluid phase was investigated to fully understand and quantify particle-fluid interactions. The torque is also very essential to advance the angular momentum equation for solid particles in discrete particle simulations (DPM). A formula for the torque exerted on spheres in random arrays was developed with simulation results over the solid volume fraction range of effective zero up to the close-packed limits, and low to intermediate particle Reynolds numbers. The normalized torque keeps nearly constant with low rotational Reynolds numbers, whereas slight deviation is observed at high rotational Reynolds numbers.

The proposed formulas for drag forces, Magnus lift forces, and torques exerted on solid particles in random arrays were installed in the open source software Multiphase Flow with Interphase eXchanges (MFiX) (version 2014-1). Both the discrete element model (DEM) and the interpolation suite in it need to be invoked to use the proposed formulas. The interpolation suite is used to calculate the drag force on each particle based on particle location rather than cell averages. Simulations for a bubbling fluidized bed are performed with the proposed formulas through the MFiX.

TABLE OF CONTENTS

TABLE OF CONTENTS – Continued 1

TABLE OF CONTENTS – Continued 2

LIST OF FIGURES

LIST OF FIGURES – Continued 1

LIST OF FIGURES – Continued 2

LIST OF FIGURES – Continued 5

EXECUTIVE SUMMARY

A new direct numerical method with second-order convergence rate was developed for fully resolved simulations of incompressible viscous flows laden with rigid particles. The method was developed by combining the state-of-the-art immersed boundary method (IBM), the multi-direct forcing method, and the lattice Boltzmann method (LBM). Previously, the combination of IBM and LBM could only achieve first-order accuracy, though LBM is a second-order method. The IBM was recently improved based on the traditional solver of incompressible Navier-Stokes equations.

In this study, the multi-direct forcing method is adopted in the improved IBM to better approximate the no-slip/no-penetration (ns/np) condition on the surface of particles, and a slight retraction of the Lagrangian grid from the surface towards the interior of particles with a fraction of the Eulerian grid spacing helps increase the convergence rate of the direct numerical method. The method is further improved by an over-relaxation technique in the procedure of multi-direct forcing method and an implementation of the classical fourth order Runge-Kutta scheme in the coupled fluid-particle interaction. The over-relaxation technique is demonstrated to yield higher orders of convergence.

The main difficulty in combining LBM and fourth order Runge-Kutta scheme is that the flow information such as density and velocity cannot be obtained directly at a fractional time step from LBM, since LBM only provides the flow information at integer time step. To overcome this problem, the flow field around a particle at a fractional time step is obtained by simply extrapolating the known flow field at the previous integer time step. The extrapolation is only implemented locally in cubic computational domains that circumscribe particles without significant computational efforts.

The classical fourth order Runge-Kutta scheme helps the overall IB-LBM achieve the second order accuracy and provides more accurate predictions of the translational and rotational motion of particles. A proper choice of the retraction distance allows the direct numerical method to reach a super-convergence of around fourth order.

The preexistent first-order accurate code applicable to only translational motion of particles has been updated by incorporating these new improvements into the preexistent code.

The updated new code, which can resolve the translational and rotational motion of particles with the second-order accuracy, has been validated with several benchmark applications.

The efficiency of IB-LBM can be easily improved by reducing the number of the Lagragian markers on particles. A new formula is proposed to determine the number of Lagrangian markers on particle surfaces. The new formula is designed to prevent overlapping force exerted on Eulerian grid points around particle surfaces. Less Lagrangian markers are needed with the new Lagrangian formula than that used in the previous literature to achieve a desired computational accuracy. Second-order convergence rates of numerical solutions can be achieved by slightly retracting Lagrangian markers from the surface towards the interior of particles with a fraction of the Eulerian grid spacing.

The immersed boundary-lattice Boltzmann method (IB-LBM) has been shown to predict correctly angular velocity of a particle. Prior to examining the drag force exerted on a cluster of particles, the updated IB-LBM code containing the new formula for the number of Lagrangian markers has been further validated by solving several theoretical problems. A set of simulations with low Reynolds numbers are executed to calculate the drag force on spheres in simple cubic arrays. The simulation results are found in good agreement with theoretical predictions. Moreover, the unsteadiness of the drag force is examined when a fluid is accelerated from rest by a constant average pressure gradient toward a steady Stokes flow. The simulation results agree well with the theories for the short- and long-time behavior of the drag force.

Flows through non-rotational and rotational spheres in simple cubic arrays and random arrays are simulated over the entire range of packing fractions, and both low and moderate particle Reynolds numbers to compare the simulated results with the literature results and develop a new drag force formula, a new lift force formula, and a new torque formula. Random arrays of solid particles in fluids are generated with Monte Carlo procedure (Metropolis N. et al. (1953)) and Zinchenko's method (Zinchenko, A. Z. (1994)) to avoid crystallization of solid particles over high solid volume fractions. Particle Reynolds number are kept very low to ensure flows of fluids around solid particles in the Stokes regime.

The simulated drag force exerted on the non-rotational spheres shows excellent agreement with the existing theories. Simulated drag forces of non-rotational particles as well as rotational particles are found to follow well the drag law proposed by Van Der Hoef et al. (2005)

except at the packing-limit solid volume fractions. A new drag force law is developed with extensive simulated results to be closely applicable to real processes over the entire range of packing fractions, and both low and moderate particle Reynolds numbers. The simulation results indicate that the drag force is barely affected by rotational Reynolds numbers. Drag force is basically unchanged as the angle of the rotating axis varies.

A new lift force formula was developed with comprehensive simulated results as a function of arbitrary rotational Reynolds numbers over the entire range of packing fractions, and low and moderate particle Reynolds numbers. The lift force also is very insignificant at rotational Reynolds numbers below 1. The lift force can be larger than the drag force as the rotational Reynolds numbers get higher especially at low solid volume fractions. Lift force exerted on rotational solid spheres is proportional to rotational Reynolds numbers defined by the angular velocity perpendicular with the flow direction.

The torque exerted on spheres by a fluid phase was investigated as another important factor to fully understand and quantify particle-fluid interactions. The torque is also very essential to advance the angular momentum equation for solid particles in discrete particle simulations (DPM). The torque exerted on spheres in random arrays is determined over the solid volume fraction range of effective zero up to the close-packed limits, and both low and intermediate particle Reynolds numbers. A formula for the torque exerted on spheres in random arrays was developed with simulation results over the solid volume fraction range of effective zero up to the close-packed limits and low to intermediate particle Reynolds numbers. The normalized torque keeps nearly constant with low rotational Reynolds numbers, whereas slight deviation is observed at high rotational Reynolds numbers.

The most recent version of the MFiX code (version 2014-1) has been downloaded and installed on a personal computer. The proposed formulas for drag forces, Magnus lift forces, and torques exerted on solid particles in random arrays are installed in the open source software Multiphase Flow with Interphase eXchanges (MFiX). Both the discrete element model (DEM) and the interpolation suite in it need to be invoked to use the proposed formulas. The interpolation suite is used to calculate the drag force on each particle based on particle location rather than cell averages.

Simulations for a bubbling fluidized bed are performed, using the proposed new formulas through the MFiX. The proposed new drag formula for the bubbling fluidized bed with lower superficial gas velocity gives similar results compared to the previous drag laws such as Gidaspow and BVK, whereas the proposed new drag formula for a bubbling bed with higher superficial gas velocity predicts better void fraction at one side of the bubbling bed than the other side, and the void fraction profiles produced by the present drag formula appears to be asymmetric. The better prediction of the void fractions at one side of the bed is very promising. More comprehensive numerical studies are needed to fully understand the performance of the proposed formulas.

INTRODUCTION

Particulate flows are involved in a great number of engineering applications. The direct numerical simulation (DNS) method has been demonstrated to be a capable and popular approach in particulate flows. The conventional DNS methods, such as the finite volume (FVM) and the finite element methods (FEM) are not very efficient in simulations of particulate flows with a large number of particles. The main obstacle with these methods comes from the frequent need of generating new and geometrically adapted grids at very advancing step. This is a very time-consuming task especially in three-dimensional flows.

Ladd [1, 2] was believed to be the first one who successfully adopted the lattice Boltzmann method (LBM) for the DNS study of particulate flows. In his study, a fixed Eulerian grid system is implemented to represent the flow field. The "bounce-back" rule [2] is applied to realize the no-slip condition on the solid-fluid interface. Therefore, the need of generating new adapted grids at very time step is eliminated. However, based on the "bounce-back" rule, the boundary of a particle is captured in a step-wise way, making the solid-fluid interface rough. To overcome this problem, Peskin [3] developed the immersed boundary method (IBM). The basic idea of the IBM is to employ a fixed Cartesian grid for the discretization of the fluid phase and to resolve the solid-fluid interface by adding additional force terms to the governing equations.

Feng & Michaelides [4] combined desired elements of the immersed boundary method, the direct forcing method [5], and the lattice Boltzmann method. They added a forcing term in the momentum equation to enforce the no-slip condition on the boundary of a moving particle. The method was demonstrated to generate a smooth boundary for particles and to be capable of achieving higher Reynolds number flows. Uhlmann [6] also presented an improved method through the use of IBM in a traditional incompressible viscous flow solver. The main idea is to incorporate Peskin's regularized delta function approach [7] into a direct formulation of the fluid-solid interaction force in order to allow for a smooth transfer between Eulerian and Lagrangian representations. This technique was implemented in a finite-difference and fractional-step context.

Recently, Kempe & Frohlich [8] proposes several enhancements of IBM which considerably improve accuracy and extend the range of applicability. An important step is a

simple low-cost iterative procedure for the Euler–Lagrange coupling yielding a substantially better imposition of boundary conditions at the interface, even for large time steps. The procedure they adopted is indeed the multi-forcing method addressed by Luo et al. [9]. Furthermore, they designed an efficient integration step for the artificial flow field inside the particles, making the accessible ratios of particle density and fluid density down to 0.3 from around 1.0.

Breugem [10] demonstrated that accuracy of the immersed boundary method could be increased to second order by adopting several new developed techniques. The method is based on the computationally efficient direct-forcing method adopted by Uhlmann [6]. Specifically, the original IBM was improved by a multi-direct forcing scheme, a slight retraction of the Lagrangian grid from the surface towards the interior of particles with a fraction of the Eulerian grid spacing, and a new procedure to lower the accessible particle–fluid mass density ratios by a direct account of the inertia of the fluid contained within the particles. The numerical examples performed by Breugem [10] have shown that the retraction distance (r_d) has a strong influence on the effective particle diameter and little influence on the error in the no-slip/no-penetration (ns/np) condition, while exactly the opposite holds for the number of iterations (N_s). The choice of $r_d=0.3\Delta x$ was found to yield second-order accuracy compared to first-order accuracy of the original method that corresponds to $r_d=0$.

Our original code only allows a particle to translation in a fluid phase with the first-order accuracy. Our code was upgraded by incorporating the state-of-the-art improvements into our old code. Two quality papers addressing the most recent improvements are published by Kempe & Frohlich [8] and Breugem [10] in 2012, where the IBM is proven to be second-order accuracy. We would like to mainly refer to the work of Breugem for the code update since the IBM, for the first time, was proven to be second-order accurate by Breugem [10]. In his study, the IBM is combined with a traditional incompressible solver. This type of solver can be executed in a fractional-step context using high-order time advancing techniques, such as the Runge-Kutta scheme. However, the fractional-step technique cannot be directly embedded in the framework of LBM. We developed a novel approach to combine the IBM and LBM. Our upgraded code is able to accommodate the translational and rotational movements of particles. Besides, an over-relaxation technique is adopted in the procedure of multi-direct forcing method to further

improve the accuracy of the combined IB-LBM. The upgraded code is capable of achieving the second-order accuracy that was verified by solving several bench-mark problems. Moreover, a novel finding in this study is that the retraction allows the direct numerical method to reach a super-convergence of around fourth order over a proper range of the retraction distance.

To examine the drag force exerted on clusters of particles, the efficiency of the present immersed boundary – lattice Boltzmann method (IB-LBM) became a concern since lots of particles are involved in the simulation. The LBM has been updated and tested over decades, and its computational efficiency has been established. However, new techniques for IBM still appear in recent years. In this study, it is found the efficiency of IBM and thus the efficiency of IB-LBM can be easily improved by reducing the number of the Lagragian markers on particles. The number of the Lagragian markers on particles is determined by a new formula, which is designed to avoid overlapping the force on Eulerian grid points around particle surfaces.

The IB-LBM code along with the new formula for the number of Lagrangian markers has been validated by solving several theoretical problems such as simple cubic lattice of fixed spheres, rotation of a sphere in linear shear flow, and a freely moving sphere in plane Poiseuille flow. Simulations under the Stokes flow regime are executed to calculate the drag force on spheres in simple cubic arrays. The simulation results are found in good agreement with theoretical predictions. Moreover, the unsteadiness of the drag force is examined when a fluid is accelerated from rest by a constant average pressure gradient toward a steady Stokes flow. The simulation results agree well with the theories for the short- and long-time behavior of the drag force. Furthermore, the code has been upgraded even further to be capable of simulating simultaneously multiple particles. The parallelization of the code has also been done. The accuracy of the parallel code has been fully validated by comparing the results to those generated by the original serial code.

Steady Stokes flows through non-rotational and rotational multiple spheres in ordered arrays are examined extensively to simulate drag force and lift force exerted on spheres at various solid volume fractions up to 0.6 and the rotational Reynolds numbers of 0.1, 1, 10, 50 and 100, verify simulation results by comparing our simulation results with literature results, develop a formula for the drag force and a formula for the lift force, and examine simulation capabilities and efficiencies of the upgraded code. The rotational Reynolds number described

with the angular velocity (ω) and the diameter (D) of rotational spheres, and kinematic viscosity (ν) of a fluid around spheres, defined as $Rer = \omega D^2/\nu$, is used to characterize rotational movement of spheres. Steady Stokes flows through non-rotational and rotational multiple (93 - 186) spheres in various random arrays generated with Monte Carlo procedure (Metropolis N. et al. (1953)) [11] are examined to simulate drag force and lift force exerted on spheres.

Richardson extrapolation method [12] is adopted to obtain the final simulation results for each simulation case, based on the results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain final simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate simulation results, though the computation of each configuration is time consuming.

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration to develop the final proposed formulas. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$, $144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144. The combination of Monte Carlo (MC) procedure and Zinchenko's method (1994) [13] for the final proposed formulas is employed to generate random configurations of spheres. Zinchenko's method generates random sphere configurations as close as possible to the close-packed limit, which is around 0.637. The radius of a sphere in a packed configuration is first decreased and then MC equilibration steps are applied to obtain small solid volume fractions. The configuration can be randomized without possible crystallization of small solid spheres by using this procedure.

Steady Stokes flows through non-rotational and rotational multiple (144) spheres in random arrays generated with Zinchenko's method (Zinchenko, A. Z. (1994)) [13] are examined to simulate drag force and lift force exerted on spheres over the entire range of packing fractions and the rotational Reynolds numbers of 1, 10 and 100. The particle Reynolds number is kept small to ensure the flow of a fluid under the Stokes regime in simulating drag forces and lift forces of non-rotational and rotational spheres in the fluid.

The drag forces exerted on non-rotational spheres show an excellent agreement with the existing theory. Drag force is barely affected by the rotational Reynolds number. However, drag force also decreases as rotational Reynolds number increases at low solid volume fractions. The

drag force is basically unchanged as the angle of the rotating axis varies. The simulation results indicate that the lift force produced by the rotation movement is directly proportional to rotational Reynolds numbers defined by the angular velocity perpendicular with the flow direction. Lift forces are affected by the angle between the rotating axis of solid spheres in ordered arrays and the flow direction of fluids. Lift force exerted on rotational spheres in the ordered array and random arrays is very insignificant at rotational Reynolds numbers below 0.1 and 1, respectively. Lift force exerted on rotational solid spheres in ordered and random arrays can be larger than the drag force at high rotational Reynolds numbers and especially low solid volume fractions.

Drag force, Magnus lift force, and torque exerted on solid particles in both simple cubic arrays and random arrays are simulated for solid volume fractions up to the close-packed limits over the low to intermediate particle Reynolds number range, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The proposed final formulas for drag force, Magnus lift force, and torque exerted on solid particles in simple cubic arrays were developed, based on simulation results, and compared with literature formulas and simulation data.

Simulated drag forces exerted on non-rotational solid spheres as well as rotational solid spheres are found to follow well the drag law proposed by Van Der Hoef et al (2005) [14] except at the packing-limit solid volume fractions. A new drag formula is proposed to fit drag force of spheres in random arrays over the entire range of packing fraction. The new drag force formula for sold particles in random arrays was developed with extensive simulated results to be closely applicable to real processes over the entire range of packing fractions and low and moderate particle Reynolds numbers.

Lift forces are simulated at the rotational Reynolds numbers of 0.1, 1, 10, 50, 100, 200 and 500, and various solid volume fractions and various particle Reynolds numbers. A new lift force formula for solid particles in random arrays was developed with comprehensive simulated results as a function of arbitrary rotational Reynolds numbers over the entire range of packing fractions and low and moderate particle Reynolds numbers.

The torque exerted on spheres by a fluid phase was investigated as another important factor to fully understand and quantify particle-fluid interactions in addition to the drag force and

the Magnus lift force exerted on solid spheres. The torque is also very essential to advance the angular momentum equation for solid particles in discrete particle simulations (DPM). The torque exerted on spheres in random arrays is determined over the solid volume fraction range of effective zero up to the close-packed limits and both low and intermediate particle Reynolds numbers. A formula for the torque exerted on solid particles in random arrays was developed with simulation data. The normalized torque keeps nearly constant with low rotational Reynolds numbers, whereas slight deviation is observed at high rotational Reynolds numbers.

The most recent version of the MFiX code (version 2014-1) has been downloaded and installed on a personal computer. The proposed formulas for drag forces, Magnus lift forces, and torques exerted on solid particles in random arrays were installed in the open source software Multiphase Flow with Interphase eXchanges (MFiX). Both the discrete element model (DEM) and the interpolation suite in it need to be invoked to use the proposed formulas. The interpolation suite is used to calculate the drag force on each particle based on particle location rather than cell averages.

Simulations for a bubbling fluidized bed are performed with the proposed formulas through the MFIX. The proposed new drag formula for the bubbling fluidized bed with lower superficial gas velocity gives similar results compared to the previous drag laws, whereas the proposed new drag formula for a bubbling bed with higher superficial gas velocity predicts better void fraction at one side of the bubbling bed than the other side, and the void fraction profiles produced by the present drag formula appears to be asymmetric. The better prediction of the void fractions at one side of the bed is very promising. More comprehensive numerical studies are needed to fully understand the performance of the proposed formulas.

EXPERIMENTAL METHODS

Several computers and software were utilized to validate the proposed IB-LBM and develop a new drag force formula, a new Magnum lift formula, and a new torque formula with extensive simulation data. The most recent version of the open source software Multiphase Flow with Interphase eXchanges (MFiX) code (version 2014-1) has been downloaded and installed on a personal computer. Simulations for a bubbling fluidized bed were performed with the proposed formulas through the MFIX.

THEORY

1. Governing equations

The governing equations for the fluid-particle composite read:

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\bullet\nabla\boldsymbol{u}\right)=\mu\nabla^2\boldsymbol{u}-\nabla p-\nabla p_e+\boldsymbol{f} \tag{1a}$$

$$\frac{\partial \rho}{\partial t}+\nabla\bullet(\rho\boldsymbol{u})=0\ (\text{or, }\nabla\bullet\boldsymbol{u}=0), \tag{1b}$$

$$\boldsymbol{f}(\boldsymbol{x},t)=\int_{\Gamma}\boldsymbol{F}(\boldsymbol{s},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{s},t))d\boldsymbol{s} \tag{1c}$$

and

$$\frac{\delta \boldsymbol{X}}{\delta t}=\int_{\Omega}\boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{s},t))d\boldsymbol{x} \tag{1d}$$

Nomenclature is as usual, with $\boldsymbol{u}=(u,v,w)^T$ designating the velocity vector in Cartesian components, i.e., along the Cartesian coordinates, *x*, *y*, *z*, while p represents the fluid pressure, ρ the fluid density and μ the fluid viscosity. It is noted that p_e is the contribution to the total pressure from a constant pressure gradient that is possibly imposed to drive a flow. The particle surface force density and the fluid body force density are referred to as $\boldsymbol{F}(\boldsymbol{s},t)$ and $\boldsymbol{f}(\boldsymbol{x},t)$, respectively. The boundary surface of a particle is denoted by Γ with the Lagrangian parametric coordinate $\boldsymbol{s}$. The immersed domain of the particle is denoted by Ω, represented by the Eulerian coordinates $\boldsymbol{x}$. Any position on the particle surface can be written as $\boldsymbol{x}=\boldsymbol{X}(\boldsymbol{s},t)$. The no-slip boundary condition is satisfied by enforcing the velocity at all boundaries to be equal to the velocity of the fluid at the same location:

$$\frac{\delta \boldsymbol{X}(\boldsymbol{s},t)}{\delta t}=\boldsymbol{u}(\boldsymbol{X}(\boldsymbol{s},t),t) \tag{2}$$

To solve the fluid field with a body force density $\boldsymbol{f}(\boldsymbol{x},t)$ (if a constant pressure gradient ∇p_e exists, the term ∇p_e can be simply included in $\boldsymbol{f}(\boldsymbol{x},t)$), the fundamental LBM equation is modified by adding a term to the collision function and becomes as follows:

$$m_i(\boldsymbol{x}+\boldsymbol{e_i},t+1)-m_i(\boldsymbol{x},t)=-\frac{1}{\tau}\left[m_i-m_i^{(eq)}\right]+F_{\boldsymbol{f},i} \tag{3}$$

where, $\boldsymbol{e_i}$ is the lattice directions, t is the lattice simulation time, τ is the relaxation time, $m_i(\boldsymbol{x},t)$ is fluid particle distribution function in the $\boldsymbol{i}$th direction, $m_i^{(eq)}$ is the equilibrium distribution function, and $F_{\boldsymbol{f},i}$ is the added force term, whose relation with $\boldsymbol{f}(\boldsymbol{x},t)$ can be found in (Luo et al. 2007) [9]. Luo et al. (2007) [9] addressed the detailed procedure to incorporate the force density $\boldsymbol{f}(\boldsymbol{x},t)$ into the LBM. The LB equation with their procedure has been proven to give the accurate recovery of the incompressible Navier-Stokes equations. Therefore, their procedure is adopted in this work to account for the solid-fluid forces.

In this report, a three dimensional D3Q19 model is used. For the details of this lattice model, the reader can refer to Ref. (Sui et al. 2008) [15].

2. Direct forcing scheme

To discuss the general concepts of the direct forcing scheme, we first write the time-discretized Eq. (1a) in the following form:

$$\rho^n\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^n}{\Delta t}=rhs^n+\boldsymbol{f}^{n+1} \tag{4}$$

where,

$$rhs=\mu\nabla^2\boldsymbol{u}-\nabla p-\nabla p_e-\rho\boldsymbol{u}\bullet\nabla\boldsymbol{u} \tag{5}$$

Following the procedure proposed by Uhlmann (2005) [6], we evaluate the force term on any Lagrangian marker $\boldsymbol{X}_l^{(i)}$ ($\boldsymbol{X}_l^{(i)}$ denotes the location of the $\boldsymbol{l}$th marker on the $\boldsymbol{i}$th particle), by

$$\boldsymbol{F}^{n+1}=\rho^n\frac{\boldsymbol{U}_p^{n+1}-\boldsymbol{U}^n}{\Delta t}-RHS^n \tag{6}$$

In Eq. (6) and henceforth the upper-case letters for quantities are evaluated at the locations of the Lagrangian marker $\boldsymbol{X}_l^{(i)}$. The desired velocity $\boldsymbol{U}_p$ of the Lagrangian marker is given by the rigid-body motion of the particle:

$$\boldsymbol{U}_p(\boldsymbol{X}_l^{(i)})=\boldsymbol{u}_c^i+\boldsymbol{\omega}_c^i\times(\boldsymbol{X}_l^{(i)}-\boldsymbol{x}_c^{(i)}) \tag{7}$$

where $\boldsymbol{u}_c^i, \boldsymbol{\omega}_c^i, \boldsymbol{x}_c^{(i)}$ are the translational and rotational velocity and center coordinates of the $\boldsymbol{i}$th particle, respectively.

The terms on the right-hand side of Eq. (6) can be collected as

$$\widetilde{\boldsymbol{U}}_p^{n+1} = \rho^n \boldsymbol{U}_p^{n+1} \tag{8}$$

and

$$\widetilde{\boldsymbol{U}}^{n+1} = \rho^n \boldsymbol{U}^n + rhs^n \Delta t \tag{9}$$

In Eq. (9), $\widetilde{\boldsymbol{U}}^{n+1}$ corresponds to a preliminary velocity obtained without applying a force term. Now, Eq. (6) has been reduced to

$$\boldsymbol{F}^{n+1} = \frac{\widetilde{\boldsymbol{U}}_p^{n+1} - \widetilde{\boldsymbol{U}}^{n+1}}{\Delta t}. \tag{10}$$

Once $\boldsymbol{F}^{n+1}$ has been obtained, it can be spread to Eulerian nodes to get $\boldsymbol{f}^{n+1}$. For the spreading procedure, we use the regularized Dirac delta function δ_d of Roma et al. (1999) [16], due to its compact support and computational efficiency. The detailed spreading procedure of ($\tilde{\boldsymbol{u}}$, $\widetilde{\boldsymbol{U}}$) and ($\boldsymbol{f}^{n+1}, \boldsymbol{F}^{n+1}$) between Lagrangian and Eulerian locations can be found in Ref. (Uhlmann 2005) [6].

Feng & Michaelides (2005) [4] adopted the direct forcing scheme of Eq. (6) for their Proteus code. They reported that usage of this scheme, along with the spreading technique, only gives the IB-LBM the first-order accuracy.

3. Improvements on the state-of-the-art immersed boundary - lattice Boltzmann method (IB-LBM)

3.1 Formulation to obtain flow information at fractional steps

In the framework of LBM, the flow information such as fluid density ρ and fluid velocity vector $\boldsymbol{u}$ cannot be obtained directly in the fractional time step. However, in order to achieve higher accuracy of the overall IB-LBM, higher order time schemes with fractional steps, such as Runge-Kutta schemes, are required. To overcome this problem, the flow field around a particle is advanced by

$$\rho^{n+\alpha} = \rho^n - \nabla\bullet(\rho\boldsymbol{u})\alpha\Delta t \tag{11}$$

and

$$(\rho\boldsymbol{u})^{n+\alpha} = \rho^n\boldsymbol{u}^n + (rhs^n + \boldsymbol{f}^{n+1})\alpha\Delta t\,. \tag{12}$$

Note, Eqs. (11) and (12) can be readily derived from Eqs. (1a) and (1b). These two equations are only implemented locally in cubic computational domains that circumscribe particles. For convenience, the influenced cubes are placed with their edges parallel with the Eulerian grid lines. The center of every cube collapses with that of the circumscribing particle. The length of every cube is set to be eight lattice units wider than the particle diameter to make sure that some differentiating operations in Eqs. (11) and (12) can be executed with enough neighbor nodes.

3.2 Relaxation technique for multi-direct forcing scheme

It has been demonstrated that the use of a Dirac delta function for the interpolation and spreading operations results in a diffuse distribution of the IBM force around the interface of a particle [9, 10]. This brings errors to the desired particle velocity at the Lagrangian grid points. The basic idea of a multi-direct forcing scheme is to iteratively determine the IBM forces to the involved Eulerian grid points so that the no-slip condition is better satisfied on the particle surface. A typical multi-direct forcing scheme reads (Luo et al. 2007 [9], Breugem 2012 [10]),

do**s**=1, $N_{\mathbf{s}}$

$$\widetilde{\boldsymbol{U}}_l^{s-1} = \sum\tilde{\mathrm{u}}^{s-1}\delta_d(\boldsymbol{x} - \boldsymbol{X}_l^{n+1})\Delta x\Delta y\Delta z \tag{13a}$$

$$\boldsymbol{F}_l^{n+1,s} = \boldsymbol{F}_l^{n+1,s-1} + \frac{\tilde{\boldsymbol{U}}_{p,l}^{n+1} - \tilde{\boldsymbol{U}}_l^{n+1,s-1}}{\Delta t} \tag{13b}$$

$$\boldsymbol{f}^{n+1,s} = \sum\boldsymbol{F}_l^{n+1,s}\delta_d(\boldsymbol{x} - \boldsymbol{X}_l^{n+1})\Delta V_l\,, \tag{13c}$$

$$\tilde{\boldsymbol{u}}^s = \tilde{\boldsymbol{u}}^0 + \Delta t\boldsymbol{f}^{n+1,s} \tag{13d}$$

enddo

where N_s is the total number of force iterations and $\tilde{\boldsymbol{u}}^0$ is the velocity at the current time step. In order to accelerate the convergence rate of IBM forces, we embed a relaxation technique in the multi-direct forcing scheme. The relaxed multi-direct forcing scheme is basically the same as the original one, except that the Eq. (13d) is updated to

$$\tilde{\boldsymbol{u}}^{s} = \tilde{\boldsymbol{u}}^{0} + \Delta t(\boldsymbol{f}^{n+1,s-1} + \omega(\boldsymbol{f}^{n+1,s} - \boldsymbol{f}^{n+1,s-1})) \quad (13e)$$

where, ω is the relaxation coefficient. Generally, $\omega = 1.4$ or $\omega = 1.5$ gives appreciable convergence acceleration of IBM forces. The values greater than 1.5 may make the simulation unstable. As one can see, the modification for relaxed multi-direct forcing scheme is rather simple. It does not incur any significant increase of computation time.

The relaxation coefficient was not adopted in extensive simulations to obtain the drag force, Magnus lift force and the torque in ordered and random arrays of spheres (i.e., $\omega = 1$ is used through these simulatiosn), though the relaxation coefficient was developed, reported and has been found to be effective. The Runge-Kutta scheme applied to combine the IBM and the LBM is already sufficiently accurate for these computations.

3.3 Combination of IBM and LBM using a Runge-Kutta scheme

The Newton equations for particles are advanced using the classical four-stages, fourth-order Runge-Kutta scheme. In order to couple the particle movements with the flow field of a fluid phase resolved by LBM, Eqs. (11) and (12) are adopted to retrieve a flow information from the fluid phase. Considering that a constant pressure gradient ∇p_e also exists in the fluid phase, the advancement of the solution from the time step *n* to the time step *n* + 1 reads:

$$\boldsymbol{f} = \nabla p_e \quad (14a)$$

do$\boldsymbol{q}$=1,4

$$\bar{n} = n + \alpha_q \quad (14b)$$

$$\underline{n} = n + \alpha_{q-1} \quad (14c)$$

$$\rho^{\bar{n}} = \rho^{n} - \nabla\bullet(\rho\boldsymbol{u})\,\alpha_q \Delta t \quad (14d)$$

$$(\rho\boldsymbol{u})^{\bar{n}} = \rho^{n}\boldsymbol{u}^{n} + (rhs^{n} + \boldsymbol{f}^{\underline{n}+1})\alpha_q \Delta t \quad (14e)$$

$$\tilde{\boldsymbol{u}}^{\bar{n}+1} = (\rho\boldsymbol{u})^{\bar{n}} + rhs^{n}\Delta t = \rho^{n}\boldsymbol{u}^{n} + (rhs^{n} + \boldsymbol{f}^{\underline{n}+1})\alpha_q \Delta t + rhs^{n}\Delta t \quad (14f)$$

$$\tilde{\boldsymbol{U}}_{l}^{n+1} = \Sigma\tilde{\boldsymbol{u}}^{n+1}\delta_d(\boldsymbol{x} - \boldsymbol{X}_{l}^{n+1})\Delta x\Delta y\Delta z \quad (14g)$$

$$\boldsymbol{F}_{l}^{\bar{n}+1} = \frac{\tilde{\boldsymbol{U}}_{p,l}^{\bar{n}+1} - \tilde{\boldsymbol{U}}_{l}^{\bar{n}+1}}{\Delta t} \quad (14h)$$

$$\boldsymbol{f}^{\bar{n}+1} = \sum \boldsymbol{F}_l^{\bar{n}+1} \delta_d(\boldsymbol{x} - \boldsymbol{X}_l^{\bar{n}+1}) \Delta V_l \tag{14i}$$

$$\tilde{\boldsymbol{u}}^{\bar{n}+1,0} = \tilde{\boldsymbol{u}}^{\bar{n}} + \Delta t \boldsymbol{f}^{\bar{n}+1} \tag{14j}$$

dos=1,N_s

$$\boldsymbol{U}_l^{\bar{n}+1,s-1} = \sum \boldsymbol{u}^{\bar{n}+1,s-1} \delta_d(\boldsymbol{x} - \boldsymbol{X}_l^{\bar{n}+1}) \Delta x \Delta y \Delta z \tag{14k}$$

$$\boldsymbol{F}_l^{\bar{n}+1,s} = \boldsymbol{F}_l^{\bar{n}+1,s-1} + \frac{\tilde{\boldsymbol{U}}_{p,l}^{\bar{n}+1} - \tilde{\boldsymbol{U}}_l^{\bar{n}+1,s-1}}{\Delta t}, \tag{14l}$$

$$\boldsymbol{f}^{\bar{n}+1,s} = \sum \boldsymbol{F}_l^{\bar{n}+1,s} \delta_d(\boldsymbol{x} - \boldsymbol{X}_l^{\bar{n}+1}) \Delta V_l \tag{14m}$$

$$\boldsymbol{f}^{\bar{n}+1,s} = \boldsymbol{f}^{\bar{n}+1,s-1} + \omega(\boldsymbol{f}^{\bar{n}+1,s} - \boldsymbol{f}^{\bar{n}+1,s-1}) \tag{14n}$$

$$\tilde{\boldsymbol{u}}^{\bar{n}+1,s} = \tilde{\boldsymbol{u}}^{\bar{n}} + \Delta t \boldsymbol{f}^{\bar{n}+1,s} \tag{14o}$$

enddo

$$\boldsymbol{f} = \boldsymbol{f} + \beta_q \boldsymbol{f}^{\bar{n}+1,N_s} \tag{14p}$$

enddo

$$m_i(\boldsymbol{x} + \boldsymbol{e}_i, t+1) - m_i(\boldsymbol{x}, t) = -\frac{1}{\tau}\left[m_i - m_i^{(eq)} \right] + F_{\boldsymbol{f},i}(\boldsymbol{f}) \tag{14q}$$

$$\rho^n \to \rho^{n+1}, \boldsymbol{u}^n \to \boldsymbol{u}^{n+1}, p^n \to p^{n+1} \tag{14r}$$

where $\alpha_0 = 0$, $\alpha_1 = 0$, $\alpha_2 = 1/2$, $\alpha_3 = 1/2$, $\alpha_4 = 1$, $\beta_1 = 1/6$, $\beta_2 = 1/3$, $\beta_3 = 1/3$, and $\beta_4 = 1/6$.

The governing equations for each solid particle are as follows,

$$d\boldsymbol{x}_c / dt = \boldsymbol{u}_c \tag{15a}$$

$$d\boldsymbol{p} / dt = \boldsymbol{F} \tag{15b}$$

$$d\boldsymbol{L} / dt = \boldsymbol{T} \tag{15c}$$

$$d\boldsymbol{q} / dt = \boldsymbol{\varpi}_c \boldsymbol{q} / 2 \tag{15d}$$

where, $\boldsymbol{p}$ and $\boldsymbol{L}$ denote the linear momentum ($M_p \boldsymbol{u}_c$) and the angular momentum ($\boldsymbol{I}_p \boldsymbol{\varpi}_c$) of the particle, respectively, and $\boldsymbol{F}$ and $\boldsymbol{T}$ represent the force and the torque exerted on the particle. In Eq. (15d), $\boldsymbol{q}$ is a unit quaternion used to record the three-dimensional orientation of the particle

(for details, refer to Ref. (Diebel 2006) [17]), and $\boldsymbol{\varpi}$ represents the angular velocity of the particle. It is remarked that there is no need to calculate $\boldsymbol{q}$ for spheres, since in this case, rotating the Lagrangian markers along with the rotation of the particle is not necessary. We only need to impose appropriate velocities on these markers. The detailed advancing procedure of each particle with the same Runge-Kutta scheme, following directly after Eq. (14p) in the do loop, is shown as follows,

$$\boldsymbol{F} = -\sum_l \boldsymbol{F}_l^{\bar{n}+1,N_s}\Delta V_l + \left(\left\{ \int_{V_p} \rho \boldsymbol{u} dV \right\}^{\bar{n}+1} - \left\{ \int_{V_p} \rho \boldsymbol{u} dV \right\}^{\bar{n}} \right) / \Delta t + (1-\frac{\rho_f}{\rho_p}) M_p g \tag{16a}$$

$$\boldsymbol{T} = -\sum_l \boldsymbol{r}_l^{\bar{n}+1} \times \boldsymbol{F}_l^{\bar{n}+1,N_s}\Delta V_l + \left(\left\{ \int_{V_p} \boldsymbol{r} \times \rho \boldsymbol{u} dV \right\}^{\bar{n}+1} - \left\{ \int_{V_p} \boldsymbol{r} \times \rho \boldsymbol{u} dV \right\}^{\bar{n}} \right) / \Delta t \tag{16b}$$

$$\tilde{n} = n + 1 + \alpha_{q+1} \tag{16c}$$

$$\boldsymbol{x}_c^{\tilde{n}} = \boldsymbol{x}_c^{n+1} + \alpha_{q+1}\boldsymbol{u}_c \Delta t \tag{16d}$$

$$\boldsymbol{p}^{\tilde{n}} = \boldsymbol{p}^{n+1} + \alpha_{q+1}\boldsymbol{F} \Delta t \tag{16e}$$

$$\boldsymbol{L}^{\tilde{n}} = \boldsymbol{L}^{n+1} + \alpha_{q+1}\boldsymbol{L} \Delta t \tag{16f}$$

$$\boldsymbol{q}^{\tilde{n}} = \boldsymbol{q}^{n+1} + \alpha_{q+1}\boldsymbol{\varpi}_c^{n+1}\boldsymbol{q}^{n+1} \Delta t / 2 \tag{16g}$$

$$\boldsymbol{u}_c^{\tilde{n}} = \boldsymbol{p}^{\tilde{n}} / M_p \tag{16h}$$

$$\boldsymbol{\varpi}_c^{\tilde{n}} = \boldsymbol{I}_p^{-1}\boldsymbol{L}^{\tilde{n}} \tag{16i}$$

$$\widetilde{\boldsymbol{U}}_{p,l}^{\tilde{n}} = \boldsymbol{u}_c^{\tilde{n}} + \boldsymbol{\varpi}_c^{\tilde{n}} x \left(\boldsymbol{X}_l^{\tilde{n}} - \boldsymbol{x}_c^{\tilde{n}} \right) \tag{16j}$$

where, V_p, ρ_p, M_p and $\boldsymbol{I}_p$ are the particle's volume, density, mass and rotational inertia, respectively. It is remarked that ρ_f represents the constant formal density of the fluid phase. We use it to get the buoyancy of the particle. This is because ρ, the instantaneous density of the incompressible fluid phase, always keeps changing around ρ_f, even though very slightly. This is determined by the intrinsic property of LBM, which solves a flow with very low compressibility flow to mimic the incompressible flow. The use of ρ_f to obtain the buoyancy simplifies the

simulation and does not incur any appreciable accuracy loss since the solving flow is incompressible.

In the advancement of the particle's motion, the techniques such as the determination of the Lagrangian marker number, the direct account of fluid inertia within particle, and inward retraction of Lagrangian markers are adopted. To make this report more concise, we do not brief them here. The reader can refer to the Ref. (Breugem 2012) [10] for the details.

Eqs.(14a-14r) and Eqs.(16a-16j) compose the complete procedure that advances the flow field from time step n to n+1, and also upgrades simultaneously the motion of a particle from time step n+1 to n+2. This is a fully explicit coupling of the LB equation and Newton-Euler equations. It is expected that this coupling will result in increased accuracy of the overall IB-LBM.

4. A new formula for the number of Lagragian markers on a particle surface

Breugem [10] examined the relation between the Eulerian grids and the Lagrangian markers (note, Lagrangian markers can also be referred as Lagrangian grid cells). In order to ensure the resolutions match each other, Breugem designed the following rules for the volume ΔV_l of the Lagrangian markers:

(a) ΔV_l is as close as possible equal to Δx^3;

(b) the number N_L of Lagrangian markers over the surface of a sphere is an integer number;

(c) the radial thickness of the Lagrangian grid cells is equal to Δx.

Following those rules, N_L and ΔV_l are defined as

$$N_L = \left[\frac{(R - r_d + \Delta x/2)^3 - (R - r_d - \Delta x/2)^3}{3\Delta x^3/(4\pi)} \right] \tag{17}$$

$$\Delta V_l = \frac{(R - r_d + \Delta x/2)^3 - (R - r_d - \Delta x/2)^3}{3N_L/(4\pi)} \tag{18}$$

where R and r_d are the radius of the particle and the retraction distance from the surface towards the interior of the particle, respectively. The square brackets in Eq. (17) denote the nearest integer value of the enclosed expression. It should be noted that the method of Leopardi [18] is employed to produce the "evenly" distribution of the Lagrangian markers on particle surfaces.

The three rules proposed by Breugem [10] were examined closely. It's hard to find some solid theories that support the first rule from the aspects of computational flow dynamics and numerical stability. Therefore, the first rule may be relaxed without affecting the simulation results. Generally, the resolution of particle motions strongly depends on the resolution of a flow field. If the flow field is not well resolved on Eulerain grids, there is no way to predict the motion of the immersed body accurately. This actually indicates that the Eulerian grids need to be relatively finer than the Lagrangian markers distributed on surfaces of particles. This principle gives $\Delta V_l > \Delta x^3$. It is noted that, ensuring $\Delta V_l > \Delta x^3$ is very essential if a deformable surface is solved. The relative movement between Lagrangian markers on a deformable surface would result in $\Delta V_l < \Delta x^3$ occasionally, that will introduce numerical errors into the quantities generated on the first or the second spatial derivatives of the distances between Lagrangian markers. The simulation will become unstable due to the numerical errors accumulated by the differentiation of very short distances between Lagrangian markers. Here, there's no such numerical instability problem for the rigid particle. Still, the principle that a good prediction of the motion of an immersed body relies on the well resolved fluid flow should be applied.

The other important aspect leading us to reduce the number of Lagrangian markers on the surface of a particle is that the forcing overlap on Eulerain grids should be avoided. The problem of the forcing overlap has been investigated by both Breugem [10] and Luo et al. [9]. They pointed out that the same Eulerian grid points are involved to force the velocity on different Lagrangian markers. At these Eulerian grid points the forcing required for the desired particle velocity at one of the Lagrangian grid points is perturbed by the forcing needed for the other and vice versa. Due to this overlap in forcing, the distribution of the IBM force around the interface of the particle may not very well enforce the desired particle velocity on the Lagrangian markers. Luo et al. [9] first proposed a multidirect forcing scheme as a remedy for this problem.

The basic idea is to do the direct forcing scheme iteratively to reduce the errors of the desired velocities on Lagrangian markers. Actually this problem can be mitigated or even completely circumvented by a very simple way. Evidently, the reason that causes the overlap in the forcing procedure is that the Lagrangian markers are too close to each other. Therefore, the direct solution for this forcing overlap is to increase the distances between the Lagrangian markers. This can be done easily by increasing the volume ΔV_l of the Lagrangian markers. The volume of the Lagrangian markers cannot be arbitrarily large, since in that case, some Eulerian grid points in the neighborhood of the particle surface will never be used, which would be a waste of the computation efforts for the flow field. In order to make best use of the flow information obtained by LB simulation on the Eulerain grid points, the volume of the Lagrangian markers should not be larger than the domain that covered by the forcing procedure. At this stage, the first rule proposed by Breugem [10] can be modified so that the ideal ΔV_l allows the Lagrangian markers to fully use the Eulerian grid points around particle surfaces without any forcing overlap.

Since the size of the domain covered by the forcing procedure is determined by the spreading function used in this procedure, the property of the spreading function should be examined carefully. For most recent IBM, the regularized Dirac delta function developed by Roma et al. [16] serves as the spreading function. It reads

$$\varphi = \begin{cases} \left(5 - 3\,\|r\| - \sqrt{-3(1-\|r\|)^2 + 1}\right)/6, & 0.5 \le \|r\| \le 1.5 \\ \left(1 + \sqrt{-3\,\|r\|^2 + 1}\right)/3, & \|r\| \le 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

where r is the distance between Lagrangian markers and Eulerian grid points normarlized by Eulerian grid spacing Δx. A salient quality of this delta function is that it only has a width of three grid cells, which means the radius of influence of a Lagrangian marker is only $1.5\Delta x$. Note, the function φ vanishes when the radius of influence reaches $1.5\Delta x$. Owing to this small radius of influence, this delta function gives a thinner porous shell around the actual particle surface than other delta functions.

To determine ΔV_l for the Lagrangian markers, the distance between two adjacent Lagrangian markers needs to be examined. For simplicity, the distribution of Lagrangian markers on a flat surface is examined. This is equivalent to the case with the particle of very large radius. Figure 1(a) shows one special case of the closest displacement of two adjacent Lagrangian markers. It can be seen that the shortest distance to avoid the use of the same Eulerian grid points by two adjacent Lagrangian markers is $2\Delta x$. The coordinates of the markers *a* and *b* shown in the figure are (*n-1.5, m*) and (*n+0.5, m*), respectively. If the markers are moved even closer, for example, marker *a* is moved towards marker *b*, the Eulerian grid point (*n,m*) used in the forcing procedure for marker *b* will also be used for marker *a*. This means the forcing overlap occurs on the Eulerian grid point (*n,m*) and the disturbed forcing on (*n,m*) may not well enforce the desired particle velocity on the markers *a* and *b*.

Figure 1(b) shows the possible longest distance ($3.5\Delta x$) to make the two adjacent Lagrangian markers fully use the Eulerian grid points between them. In this case, the Eulerian grid point (*n ,m*) is critically used by marker *b*. If marker *b* is moved slightly away from marker *a*, the point (*n ,m*) will be left unused by either marker.

Figure 1(c) shows the case with a distance of $3\Delta x$. Note that, the distance $3\Delta x$ is determined inherently by the delta function shown in Eq. (19). This case is very desirable because it ensures that the two adjacent Lagrangian markers can fully use the Eulerian grid points between them without any overlap. This case leaves one Eulerian grid point critically unused only when both marker *a* and marker *b* are in the half way of Eulerian grid points, as shown in Figure 1(d).

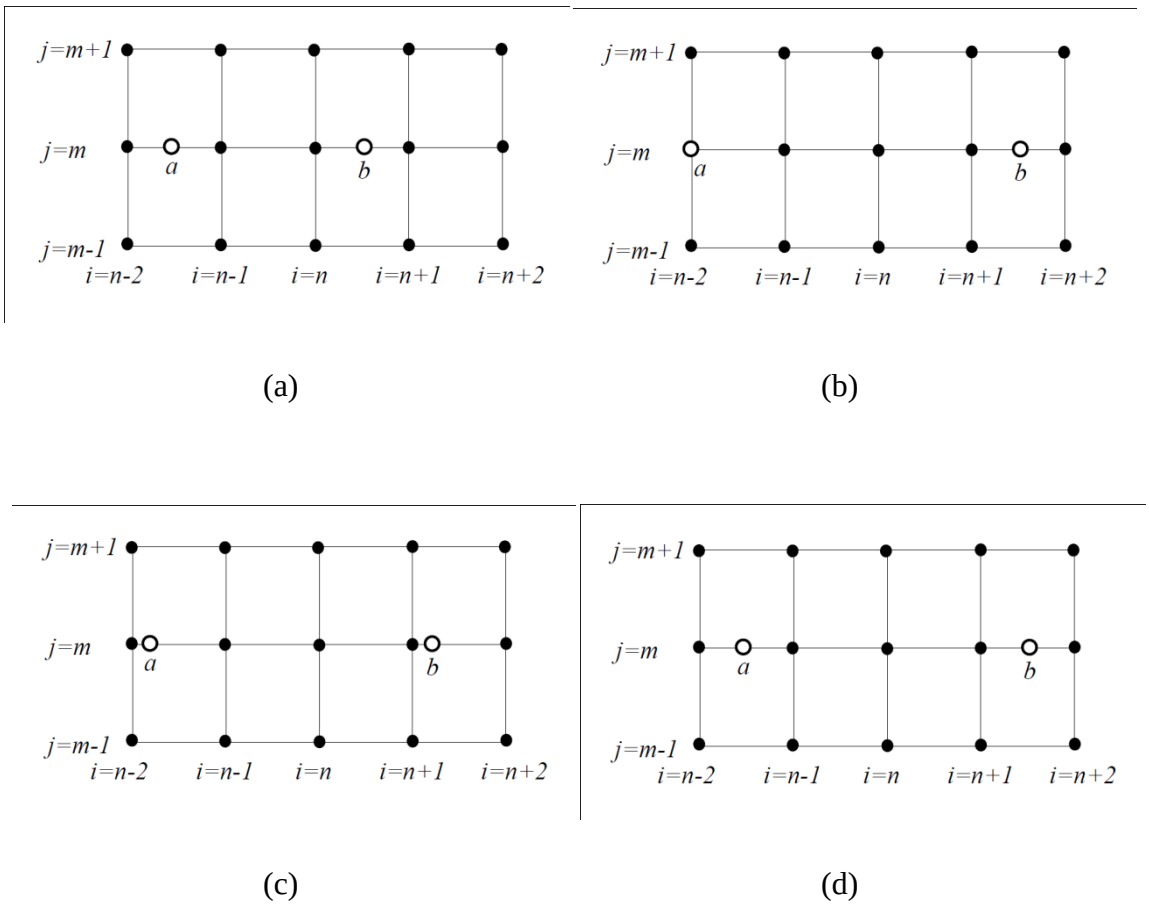


Figure 1. The schematic diagrams for the distribution of Lagrangian markers on a plane surface. Solid symbols denotes the Eulerian grid points, while *a* and *b* denote two adjacent Lagrangian markers. (a) the shortest distance ($2\Delta x$) allowed to avoid the use of the same Eulerian grid points by two adjacent Lagrangian markers; (b) the longest distance ($3.5\Delta x$) allowed to make the two adjacent Lagrangian markers to fully use the Eulerian grid points between them. (c) the distance ($3\Delta x$) that ensures the two adjacent Lagrangian markers to fully use the Eulerian grid points between them; (d) a special case of (c), in this case, the Eulerian grid point (*n*,*m*) is critically not used by either marker *a* or marker *b*.

The above analysis shows that the horizontal distance between two adjacent Lagrangian markers on a flat surface is larger than $2\Delta x$, and smaller than $3.5\Delta x$. The distance of $3\Delta x$ is the most desirable one since it ensures that the two adjacent Lagrangian markers fully use the Eulerian grid points between them without any overlap. If the Lagrangian markers are applied to the horizontal direction and the vertical direction of the flat plane, the predicted area influenced by a Lagrangian marker can be simply $(3\Delta x)^2$. Now, extending the analysis on a sphere surface,

it is assumed that the volume occupied by a Lagrangian marker satisfy $\Delta V_l = \gamma^2 \Delta x^3$, provided the thickness of the surface is Δx (the third rule proposed by Breugem [**10**]). This gives the new formula for determining the number of the Lagrangian marker number, as shown in Equation 20.

$$N_L = \left[\frac{(R - r_d + \Delta x/2)^3 - (R - r_d - \Delta x/2)^3}{3\gamma^2 \Delta x^3/(4\pi)} \right] \tag{20}$$

One can estimate that $2.0 < \gamma < 3.5$ based on previous analysis on a flat plane. The exact γ would depend on the radius of the sphere and on the distribution of the Lagrangian markers on the sphere surface. Therefore, it is hard to give a theoretical prediction for γ. However, the value of γ can be obtained through extensive computational tests. The γ value of 2.824 works very well for all the tested applications according to our computational test. The γ value of 2.824 can reduce the error between the actual velocity and the desired velocity to around 1×10^{-9} in the test of Darcy problem, whereas, the previous $\gamma = 1$ used by Breugem [10] can only reduce the error to around 1×10^{-3}. The efficiency of the overall IB-LBM can also be improved significantly since the number of the Lagrangian marker is only 1/8 of that used in previous simulation. The detailed data of these tests will be given elsewhere since this report is focused on the validation of the code.

This technique using less Lagrangian markers is another technique that is developed, reported but not adopted in the extensive simulations obtaining the drag force, Magnus lift force and the torque in both ordered and random arrays of spheres (still Eq.(17) is used). Certainly, this technique will benefit simulations in which very large spheres are involved, e.g., simulations for poly-dispersed suspensions in which spheres with greater size become extremely large when spheres with smaller size are sufficiently resolved.

5. Existing analytical equations and simulation data on drag forces exerted on solid spheres in simple cubic arrays over low particle Reynolds numbers at various solid volume fractions.

From the fundamental periodic solution of the Stokes equations, Hasimoto [19] calculated the drag force exerted on spheres in dilute simple cubic, body-centred cubic and face-

centred cubic arrays. For simple cubic arrays, the non-dimensional drag force is given by Equation 21.

$$F = (1-c)[1-1.7601c^{1/3} + c - 1.5593c^2 + O(c^{8/3})]^{-1} \quad (21)$$

Sangani and Acivos [20] modified Hasimoto's method and calculated drag force in cubic arrays over the complete porosity range. Their extended equation for simple cubic lattices is shown in Equation 22.

$$F = (1-c)[1-1.7601c^{1/3} + c - 1.5593c^2 + 3.9799c^{8/3} - 3.0734c^{10/3} + O(c^{11/3})]^{-1} \quad (22)$$

Zick & Homsy [21] calculated the drag force on the spheres in ordered arrays at solid volume fractions up to their respective close-packed limits. It is believed that the error in their computations for close-packed arrays was less than 2%. Therefore, their results are used as the benchmark for the comparison with those of our lattice-Boltzmann simulations at largest solid volume fractions.

6. Existing analytical formulas on the drag force exerted on solid spheres in random arrays for comparison with our simulated results with those from previous studies over low particle Reynolds numbers

Existing analytical formulas on the drag force are selected to compare our simulated results with those from previous studies. The Carman equation [22] described the drag force of spheres in terms of practical dense packing, as shown in Equation 23.

$$F_D = 10\frac{c}{(1-c)^2} \ . \quad (23)$$

Koch & Sangani (1999) [23] proposed the following expression for the drag force in the entire range of solid volume fractions:

$$F_D = \begin{cases} \dfrac{(1-c)(1+\dfrac{3}{\sqrt{2}}c^{1/2}+\dfrac{135}{64}c\ln c+16.14c)}{1+0.681c-8.48c^2+8.16c^3}, & c<0.4 \\ 10\dfrac{c}{(1-c)^2}, & c>0.4 \end{cases} \tag{24}$$

Recently, the most accurate formula on the drag force is proposed by Van Der Hoef et al. (2005) [14] based on their extensive LBM simulations, as shown in Equation 25.

$$F_D = 10\frac{c}{(1-c)^2}+(1-c)^2(1+1.5\sqrt{c}) \tag{25}$$

7. Calculations of torque on spheres

The torque $\overline{\boldsymbol{T}}$ exerted on a rotating sphere by a fluid is simulated. The torque $\overline{\boldsymbol{T}}$ is very desirable in practical numerical simulations such as computational fluid dynamics-discrete element method **(**CFD-DEM), because it is needed along with torques from other origins to advance the equations that govern the rotational motion of individual particles. Here, $\overline{\boldsymbol{T}}$ in stage q of a time step can be calculated, as shown in Equation 26.

$$\overline{\boldsymbol{T}}^{q} = -\sum_{l} \boldsymbol{r}_l^{q-1} \times \boldsymbol{F}_l^{q,N_s} \Delta V_l + \frac{d}{dt}\left(\int_{V_p} \boldsymbol{r} \times (\rho \boldsymbol{u}) dV \right), \tag{26}$$

where V_p is the sphere volume, $\boldsymbol{u}$ is the velocity vector of the flow, ΔV_l is the volume for the l th Lagrangian marker, $\overline{\boldsymbol{F}}_l^{q,N_s}$ is the force experienced by the l th Lagrangian marker, N_s represents the total number of the force iterations in the multi-direct forcing method, $\boldsymbol{r}$ is the position vector relative to the sphere centroid. The final value of $\overline{\boldsymbol{T}}$ is obtained in the following expression,

$$\overline{\boldsymbol{T}} = \sum_{k=1}^{i} \beta_{ik} \overline{\boldsymbol{T}}^{k}, \tag{27}$$

where β_{ik} denotes the coefficients of the Runge-Kutta scheme. Here, $i = 4$ since the four stages of the fourth order Runge-Kutta scheme is used.

The torque $\overline{\boldsymbol{T}}$ due to particle rotation for the near zero solid volume fraction at low Reynolds numbers has been analytically obtained by Kirchhoff (1876) [24] and Rubinow & Keller (1961) [25]. It reads

$$\boldsymbol{T}_{RK} = -\pi\mu d^3\boldsymbol{\omega}. \quad (28)$$

where $\boldsymbol{\omega}$ is the angular velocity of the sphere, d is the diameter of the sphere, and μ is the dynamic viscosity of the fluid. It is appropriate to normalize the torque $\boldsymbol{T}$ with the magnitude of $\boldsymbol{T}_{RK}$. Hence, the normalized torque $\boldsymbol{T}$ can be calculated as

$$\boldsymbol{T} = T\boldsymbol{n}_T = \boldsymbol{T}/\pi\mu d^3|\boldsymbol{\omega}|, \quad (29)$$

where T represents the magnitude of the normalized torque $\boldsymbol{T}$ and $\boldsymbol{n}_T$ is the unit direction vector of $\boldsymbol{T}$. In low and intermediate particle Reynolds number flows, the simulations show that the direction of the torque $\boldsymbol{T}$ is indistinguishable from the opposite direction of the angular velocity $\boldsymbol{\omega}$ even when the rotational Reynolds number reaches up to $O(10^2)$. Therefore, $\boldsymbol{n}_T$ can be simply defined by the following expression

$$\boldsymbol{n}_T = -\boldsymbol{\omega}/\sqrt{\boldsymbol{\omega}\cdot\boldsymbol{\omega}}. \quad (30)$$

RESULTS AND DISCUSSIONS

1. Validation of the proposed IB-LBM

1.1 Simple cubic lattice of fixed spheres

The accuracy of the present IB-IBM is demonstrated for both fixed and freely moving spheres. Here, only flows containing just one sphere are considered. The extension to simulating flows with multiple spheres can be done easily without any significant difficulties. The influence of the relaxation coefficients in the multi-direct forcing scheme and of the retraction distance (r_d) on the numerical accuracy is discussed. In all the following simulations, the total multi-direct forcing loop number is set to N_s=2 to give better no penetration/no slip (np/ns) condition without incurring too much computational efforts.

1.1.1 Effects of retraction distances on convergence order rate

To the best of our knowledge, this numerical problem has been tested, for the first time, by Breugem [10]. This problem describes a laminar flow going through a simple cubic lattice of fixed spheres. Therefore, it suffices to simulate a single sphere positioned in the center of a fully periodic cubical flow domain. Since the sphere is held fixed in space, the Newton-Euler

equations do not need to be solved. The desired velocity at the position of the Lagrangian grid points is simply zero. For the detailed description of this problem, refer to the Ref. [10].

The simulations with $D/\Delta x$ = 16, 24 and 36 are executed to get the convergence rate of the current IB-LBM, where D denotes the diameter of the simulated sphere. The error percentage in Darcy number D_a is calculated based on the computational results. From Figure 2, we can see that the simulation with no retraction (r_d=0.0) only gives a 1.17 order accuracy. As the retraction distance increases, the convergence rate increases first, and then decreases dramatically after the distance goes beyond 0.35.

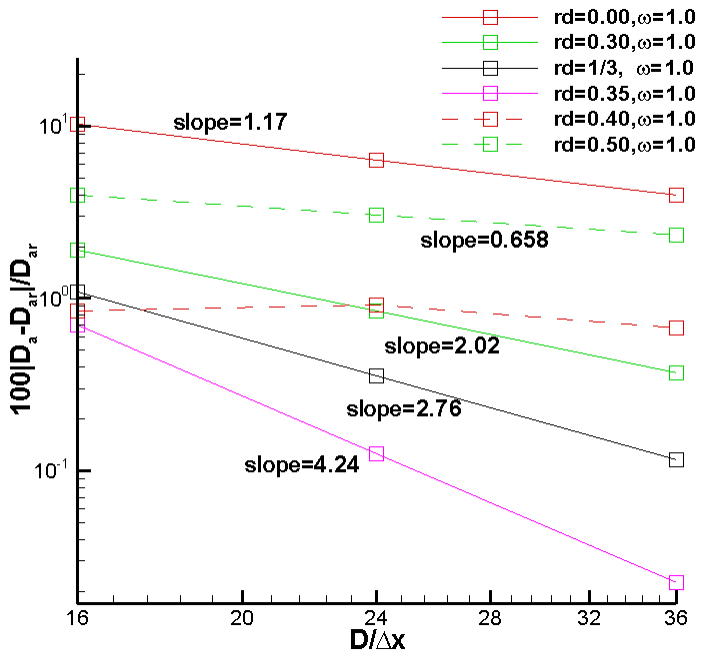


Figure 2, Error percentages in Darcy number D_a as function of the grid resolution at different retraction distances. The error is relative to the value of the Darcy number D_{ar} obtained from Richardson extrapolation [12] using the data points of $D/\Delta x$ = 16, 24 and 36.

It shows that, when r_d is between 0.30 and 0.35, the numerical methods become more than second order accurate. Very surprisingly, r_d=0.35 yields a convergence rate higher than fourth order. It is remarked that, this super convergence will not persist if we refine our Eulerain grid further. This is because the LBM only predicts the flow field with the second-order accuracy. However, the super convergence can be very useful since it gives more accurate

results with the same Eulerain grids. Breugem [10] has shown that the retraction distance r_d=0.30 yields the second-order results. This is confirmed by our simulation.

1.1.2 Effects of relaxation coefficients on convergence order accuracy

In order to evaluate the influence of the relaxation coefficient, the simulations with different relaxation coefficients are also performed. Figure 3 shows that the convergence rate of the simulation increase slightly as the relaxation coefficient goes from 1.0 to 1.6. This result indicates that a value greater than 1 of the relaxation coefficient does help accelerate the multi-direct forcing procedure. Our simulations also show that the value larger than 1.6 tends to bring in numerical instabilities to the simulation. Therefore, for general use, $\omega = 1.4$ or $\omega = 1.5$ is readily recommended.

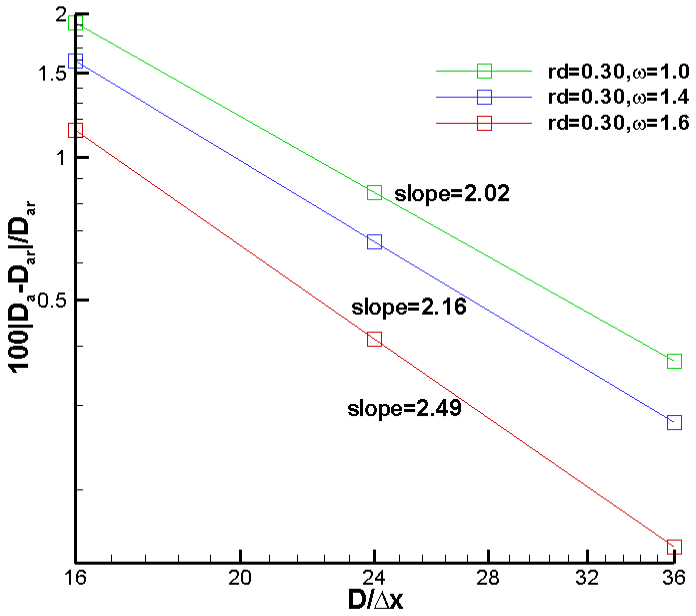


Figure 3. Error percentages in Darcy number D_a as function of the grid resolution at the same retraction distance (r_d=0.3) with different relaxation coefficients in the multi-direct forcing scheme. The error is relative to the value of the Darcy number D_{ar} obtained from Richardson extrapolation [12] using the data points of $D/\Delta x$ = 16, 24 and 36.

1.2 Rotation of a sphere in linear shear flow

This numerical problem has been tested by Bagchi & Balachandar [26] and Kempe & Frohlich [8]. It presents the rotational motion of a spherical particle in a linear shear flow given by

$$u(y) = U_0 + Sy \tag{31}$$

The position of the particle is fixed. The particle is free to rotate around the fixed position of its center, experiencing only torque generated by the surrounding fluid. There are two following essential dimensionless numbers controlling the flow.

$$\mathrm{Re}_s = \frac{SD^2}{\nu} \tag{32}$$

$$G = SD/U_c \tag{33}$$

Re_s and G are the Reynolds number and the shear parameter, respectively. In the following simulations, the Reynolds number and the shear parameter are kept 4 and 0.2, respectively. The retraction distance is fixed to r_d=0.3 in the simulations.

Figure 4 shows that the second-order accuracy of resolving the rotational motion of particles is achieved with the present IB-LBM. We can also see a slight decrease of the accuracy when the relaxation coefficient increases from 1.0 to 1.4. However, the simulation with $\omega = 1.4$ still outperforms that with $\omega = 1.0$ (no relaxation), in the sense that the former gives smaller error at the same grids.

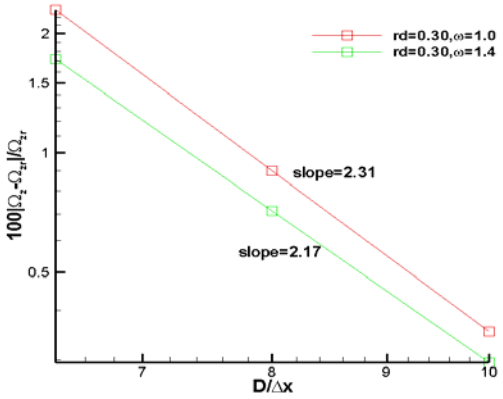


Figure 4. Error percentages in the angular velocity Ω_z of the particle as function of the grid resolution at the same retraction distance (r_d=0.3) with different relaxation coefficients in the multi-direct forcing scheme. The error is relative to the value of the angular velocity Ω_z obtained from Richardson extrapolation [12] using the data points of $D/\Delta x$ = 6.4, 8 and 10.

1.3 Freely moving sphere in plane Poiseuille flow

This problem, a freely moving sphere in an upward plane Poiseuille flow, has been tested by Uhlmann [27] and also by Breugem [10]. This problem serves the good purpose of demonstrating the accuracy of the present IB-LBM for freely moving particles. To save space, the description of this problem is omitted here. For details, one can refer to Ref. [10] or [8].

The convergence rates of the simulations on different grids with two different relaxation coefficients are shown in Figure 5. We can see that both relaxation coefficients give the numerical methods the third-order accuracy, demonstrating the success of the present combination of IB-LBM. Here the influence of the relaxation coefficient is very marginal.

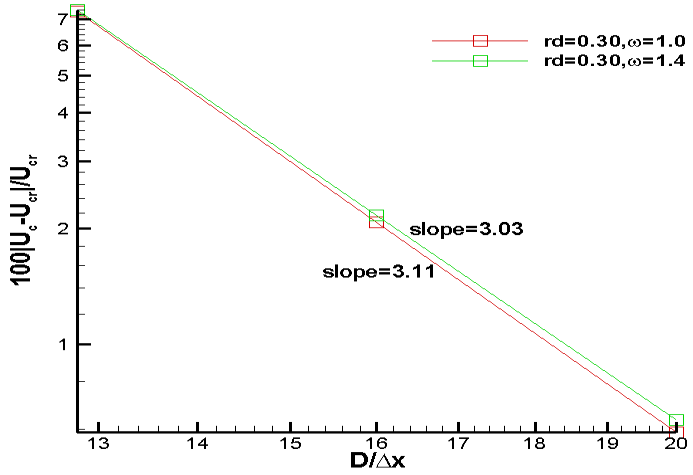


Figure 5. Error percentages in the linear velocity U_c of the particle as function of the grid resolution at the same retraction distance (r_d=0.3) with different relaxation coefficients in the multi-direct forcing scheme. The error is relative to the value of the linear velocity U_c obtained from Richardson extrapolation [12] using the data points of $D/\Delta x$ = 12.8, 16, and 20.

2. Final Formulas

2.1. Drag Force

2.1.1 The drag force exerted on solid spheres in simple cubic arrays over low particle Reynolds numbers and arbitrary solid volume fractions up to the clos-packed limits (π/6)

Drag force exerted on solid particles in simple cubic arrays is simulated for solid volume fractions up to the close-packed limits in the low particle Reynolds number range, using a

second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formulas for drag force exerted on solid particles in simple cubic arrays were developed, based on the simulation results.

The drag force of a solid particle in simple cubic arrays only needs to be calculated for a single direction at a constant pressure gradient. Hasimoto (1959) [19] calculated the drag force on the spheres in dilute ordered arrays. The formula for the normalized drag force exerted on the spheres in simple cubic arrays was developed as a function of solid volume fractions c, as shown in Equation 21. Sangani & Acrivos (1982) [28] extended Hasimoto's results to a higher-order approximation, as shown in Equation 22.

Almost at the same time, Zick & Homsy (1982) [21] calculated the drag force on spheres in simple cubic arrays at solid volume fractions up to the close-packed limit. The maximum error in their computations for around close-packed arrays was believed to be less than 2%, making their results the benchmark to which the numerical simulation results are often compared especially at high solid volume fractions (Hill et al. 2001) [29].

The simulations of the drag force exerted on solid particles are performed for each solid volume fraction at the grid resolutions L/h of 32, 48, 72 and 108, where L denotes the dimension of the computational domain and h is the grid interval. Simulated drag forces exerted on solid particles are determined by using the Richardson extrapolation method. The difference between the simulated drag force results generated at the grid resolutions L/h of 32, 48 and 72 and those at the grid resolutions L/h of 48, 72 and 108 is only around 0.1%. These simulated drag forces show that simulations on even finer meshes are not necessary. The simulated results generated from $L/h =$ 48, 72 and 108 are described in this section. The normalized drag force on the spheres in a simple cubic array is shown in Figure 6. Our simulation results are in excellent agreement with the theory of Zick & Homsy. The differences between our simulation results and those of Zick & Homsy are 0.7%, 0.5%, 0.8%, 0.5%, 1.3%, 1.1%, 1.3% and 1.6% at the solid volume fractions of 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.45 and 0.5, respectively. These simulation results indicate that the difference between our simulation results and those of Zick & Homsy [21] increases as the solid volume fraction increases. However, all the differences are below 2%, which is the maximum error percentage claimed in Zick & Homsy's results. It is believed that

the present results are more accurate than Zick & Homsy's at high solid volume fractions, where their analytical computations suffer from limited order approximations. The computational results from Hill et al. (2001) [29] also are shown in Figure 6. Their data scatter around the results of Zick & Homsy [21] due to the insufficient grid resolutions adopted in their simulations.

Zick & Homsy (1982) [21] did not formulate an explicit formula as did Hasimoto (1959) [19] and Sangani & Acrivos (1982) [28], though they calculated the drag force for the entire range of the solid volume fraction.

A simple formula for the drag force over the entire packing range and the low particle Reynolds number range is proposed with the present simulation data for high solid volume fractions and literature theoretical data for low solid volume fractions, as shown Equation 34.

$$F_D = \begin{cases} (1-c)[1-1.7601c^{1/3} + c - 1.5593c^2 + 3.9799c^{8/3} - 3.0734c^{10/3}]^{-1} & c < 0.2 \\ 2.812 + 2.621c + 47.99c^2 + 16.99c^3 & 0.2 \le c \le \pi/6 \end{cases} \quad (34)$$

where the part for $c < 0.2$ is from Sangani & Acrivos's theory and $\pi/6 \approx 0.524$ represent the theoretical packing limit of simple cubic arrays of spheres. It can be seen that the proposed formula well reproduces the drag force in the entire packing range, as shown in Figure 6. The slight deviation of the proposed formula from the dashed line that interpolates the results from Zick and Homsy is visible for high solid volume fractions. This deviation, as mentioned above, comes from the limited order approximations adopted in their analytical calculations.

Equation 34, described the drag force as a function of solid volume fractions, still be applicable to evaluating drag force exerted on rotational solid particles regardless of the values of rotational Reynolds numbers Re_r and rotating axis directions θ, since the drag force is not sensitive with these rotation-related parameters.

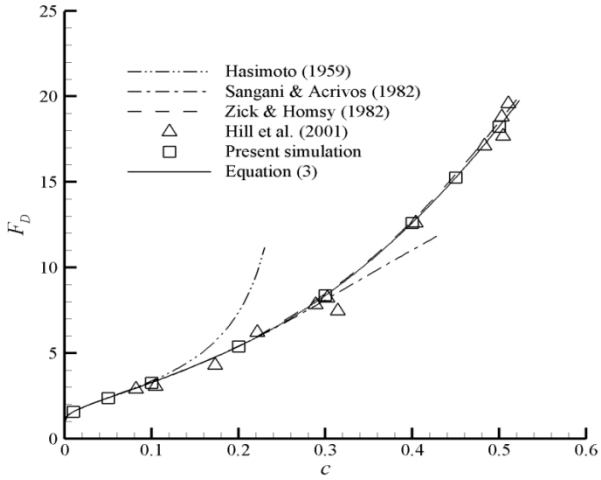


Figure 6. The normalized Stokes-flow drag force on non-rotational spheres in simple cubic arrays as a function of the solid volume fraction at low particle Reynolds numbers. The simulation result of Hill et al. (2001) [29] and the results from the theories of Hasimoto (1959) [19] and Sangani & Acrivos (1982) [28] are also shown. The dashed line interpolates the discrete results of Zick and Homsy (1982) [21]. The solid line represents the fomula proposed based on the present simulation data and literature theories.

2.1.2. The drag force on non-rotational solid particles in random arrays over wide-range packing fractions and low particle Reynolds numbers.

Drag forces exerted on solid particles in random arrays are smulated for solid volume fractions up to the close-packed limits over the low particle Reynolds number range, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formulas for drag force exerted on non-rotational solid particles in random arrays were developed, based on the simulation results.

The combination of Monte Carlo (MC) [11] procedure and Zinchenko's method (1994) [13] is employed to generate random configurations of spheres. Zinchenko's method generates random sphere configurations as close as possible to the close-packed limit, which is around 0.637. The radius of a sphere in a packed configuration is first decreased and then MC equilibration steps are applied to obtain small solid volume fractions. The configuration can be randomized without possible crystallization of small solid spheres by using this procedure.

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$,

$144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144. Richardson extrapolation method is adopted to obtain the final simulation results for each simulation case, based on the results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain final simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate simulation results, though the computation of each configuration is time consuming.

The solid volume fractions (c) chosen for the simulation of the drag force of solid spheres are 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.615, 0.63, 0.6340 and 0.6345.

Several existing formulas on drag force are chosen to compare our simulated results with literature studies. The Carman equation [22] describes the drag force for spheres in practical dense packing, as shown in Equation 23. Koch & Sangani (1999) [23] proposed the following formula for the drag force in the entire range of solid volume fractions, as shown in Equation 24. Recently, Van Der Hoef et al. (2005) [14] has developed the most accurate formula for the drag force from their extensive LBM simulation data, as shown in Equation 25.

The normalized Stokes-flow drag force F_D exerted on non-rotational spheres in random arrays at various solid volume fractions (c) is shown in Figure 7. The present simulation results are compared to several sets of numerical data from the literatures. Our simulation results agree well with the formula proposed by Van Der Hoef et al. (2005) [14]. The present simulation at the close-packing limit gives slight smaller drag compared to that predicted by Van Der Hoef et al.'s formula. The simulated drag forces for dense systems actually tend to get close to the experimental results reported by Ergun (1952) [33].

A new drag formula is proposed to fit the present simulation results over low Reynolds numbers and full-range packing fractions, as shown in Equation 35.

$$F_D\big|_{Re_p=0}=\begin{cases}9.9c/(1-c)^2+(1-c)^3(1+3c^{0.6}) & c<0.55\\ 5.87\sin\left((c/0.637)^{1.75}\,\pi/2\right)/(1-c)^2 & 0.55\le c\le 0.637\\ (5.87/0.637)c/(1-c)^2 & c>0.637\end{cases} \tag{35}$$

where 0.637 is the packing limit of the randomly packed bed of monodisperse spheres in experiments (Scott & Kilgour,1969) [34]. In the range of $c<0.55$, the present expression takes

a similar form of Van Der Hoef et al.'s equation. It is found that the maximum difference between Van Der Hoef et al.'s equation and the present data is less than 3%. The third line in Equation 35 indicates that $F_{D|Rep=0}$ is equal to $(5.87/0.637)c/(1-c)^2$ for very large solid volume fraction c above 0.637. This line is added since the solid volume fraction may be beyond the packed limit for perfectly spherical particles in practical simulation.

Figure 7 shows that the proposed drag law formula, shown in Equation 35, fits our simulation results very well. The drag force exerted on rotational particles still follows the drag force formula described in Equation 35, since the drag force is almost unaffected by the rotational motion of the spheres.

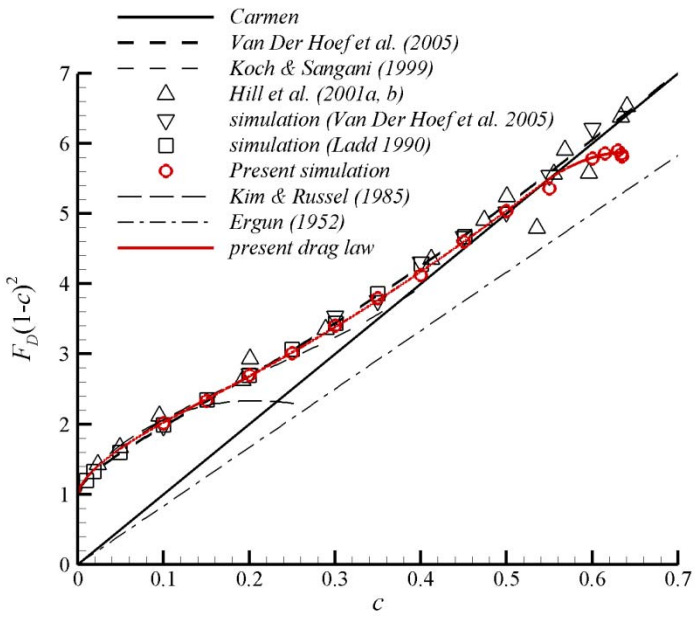


Figure 7. The normalized Stokes-flow drag force F_D (multiplied by the porosity squared) on the non-rotational spheres in random arrays as a function of the solid volume fraction over low Reynolds numbers. The simulation results of Ladd (1990) [31], Hill et al. (2001a,b) [30, 32] and Van Der Hoef et al. (2005) [14] are represented by symbols. The formula proposed by Van Der Hoef et al. (2005) is represented by a dash dotted line. The results from the theories of Carmen [22] and Koch & Sangani (1999) [23] are also shown.

2.1.3 The drag force on solid particles in random arrays at arbitrary variables such as particle Reynolds numbers, rotational Reynolds numbers and packing fractions

Drag forces exerted on solid particles in random arrays are simulated for solid volume fractions up to the close-packed limits over the particle Reynolds number range of low to intermediate, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formula for drag force was developed, based on the simulation results.

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$, $144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144.

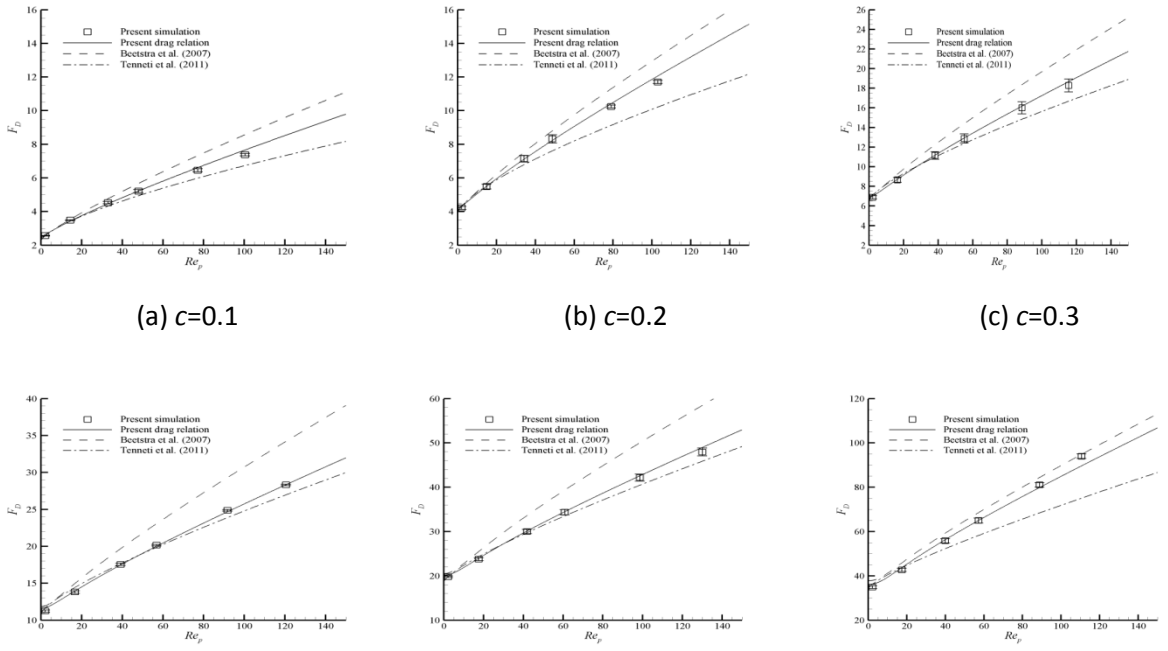
Richardson extrapolation method is adopted to obtain the final simulation results for each simulation case, based on the results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain credible simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate simulation results, though the computation of each configuration is time consuming.

The drag force formula at arbitrary Reynolds numbers was developed with extensive simuation results, as shown in Equation 36.

$$F_D(c,Re_p)=F_D\big|_{Re_p=0}+\frac{(0.256+1.41c-5.61c^2+6.04c^3)Re_p}{24(1-c)^2}\left[\frac{(1-c)^{-1}+3c(1-c)+8.4Re_p^{-0.343}}{1+10^{3c}Re_p^{-(1+4c)/2}}\right] \quad (36)$$

where $F_D\big|_{Re_p=0}$ is the drag force relation at low Reynolds numbers, as shown in Equation 35.

The normalized drag force F_D is simulated over the particle Reynolds number range of effectively zero to approximately 100, as shown in Figure 8. The proposed drag formula agrees well with the present simulation results. The drag formulas proposed by Beetstra et al.(2007) [35] and Tenneti et al.(2011) [36] are compared with the proposed drag formula, as shown in Figure 8.



(a) c=0.1 (b) c=0.2 (c) c=0.3

(d) c=0.4 (e) c=0.5 (f) c=0.6

Figure 8. The normalized drag force F_D over the particle Reynolds number range of effectively zero to approximately 100. The drag laws proposed by Beetstra et al. (2007) [35] and Tenneti et al. (2011) [36] are also shown for comparison.

The drag forces from Beetstra et al.'s law (2007) [40] over-predict those from the proposed drag force formula, whereas the drag forces from Tenneti et al.'s law (2011) [36] under-predict those from the proposed drag force formula, as shown in Equation 36.

Figure 9 shows the normalized drag force F_D on the spheres in random arrays as a function of the particle Reynolds number Re_P at various solid volume fractions. The error bars represent the standard deviations in F_D. The solid lines from bottom to top are computed for c =0.1, 0.2, 0.3, 0.4, 0.5 and 0.6, respectively, using Equation 36.

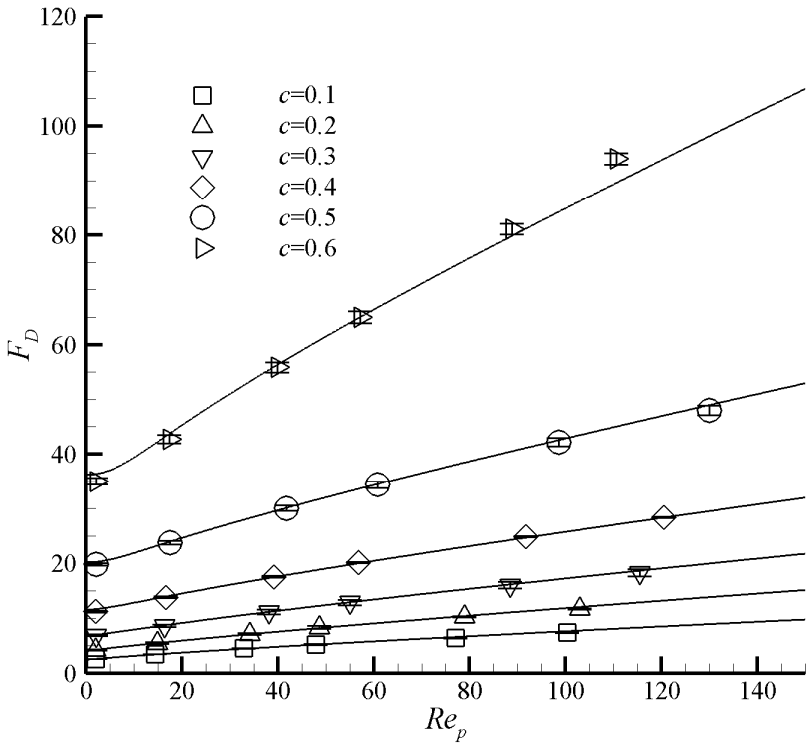


Figure 9. The normalized drag force F_D on the spheres in random arrays as a function of the particle Reynolds number Re_P at various solid volume fractions. The error bars represent the standard deviations in F_D. The solid lines from bottom to top are computed for c =0.1, 0.2, 0.3, 0.4, 0.5 and 0.6, respectively, using Equation 36.

2.2. Magnus Lift Force

2.2.1 The Magnus lift force on rotational solid spheres in simple cubic arrays at low particle Reynolds numbers, arbitrary rotational Reynolds numbers, and arbitrary packing fractions up to the close-packed limits

Magnus lift force exerted on solid particles in simple cubic arrays is simulated for solid volume fractions up to the packed limit over the low particle Reynolds number range, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formula for Magnus lift force exerted on solid particles in simple cubic arrays was developed, based on the simulation results.

The Magnus lift force F_L on rotational solid spheres for various solid volume fractions c was developed solely at low rotational Reynolds numbers due to the linear dependence of the lift forces on the rotational Reynolds number Re_r. Figure 10(a) depicts the lift force F_L as a function of the solid volume fraction c at Re_r=0.1. Figure 10(b) depicts F_L(1-c)^2 (lift force multiplied by the porosity squared) as a function of the solid volume fraction c at Re_r=0.1. The data points are computed at $Re_r = 0.1$, which provides accurate proportionalities for high Re_r. Using the standard least-squares algorithm, a tentative fit of the data points up to third order approximation can be developed, as shown in Equation 37.

$$F_L(1-c)^2 / Re_r = (-0.252c^3 + 0.197c^2 - 0.0827c + 0.0412). \quad (37)$$

Equation (37), developed solely based on the simulated data for the discrete solid volume fractions, has a good agreement with the theoretical result developed by Rubinow & Keller (1961) [25], as shown in Equation 38, as the solid volume fraction c is approaching zero, where the error is as small as 1.1%, as shown in Equation 38.

$$F_L / Re_r = 1/24. \quad (38)$$

This good agreement indeed demonstrates that the accuracy of the present numerical methods is excellent. Our simulation result is compared with the theoretical result of Rubinow & Keller in Figure 10. This is simply because no results of the Magnus lift force over low particle Reynolds numbers can be found at non-zero solid volume fractions in the literature and all previous empirical formulass (e.g., Oesterle & Dinh 1998; Loth 2008) [37] for the Magnus lift force were not develop over low particle Reynolds numbers.

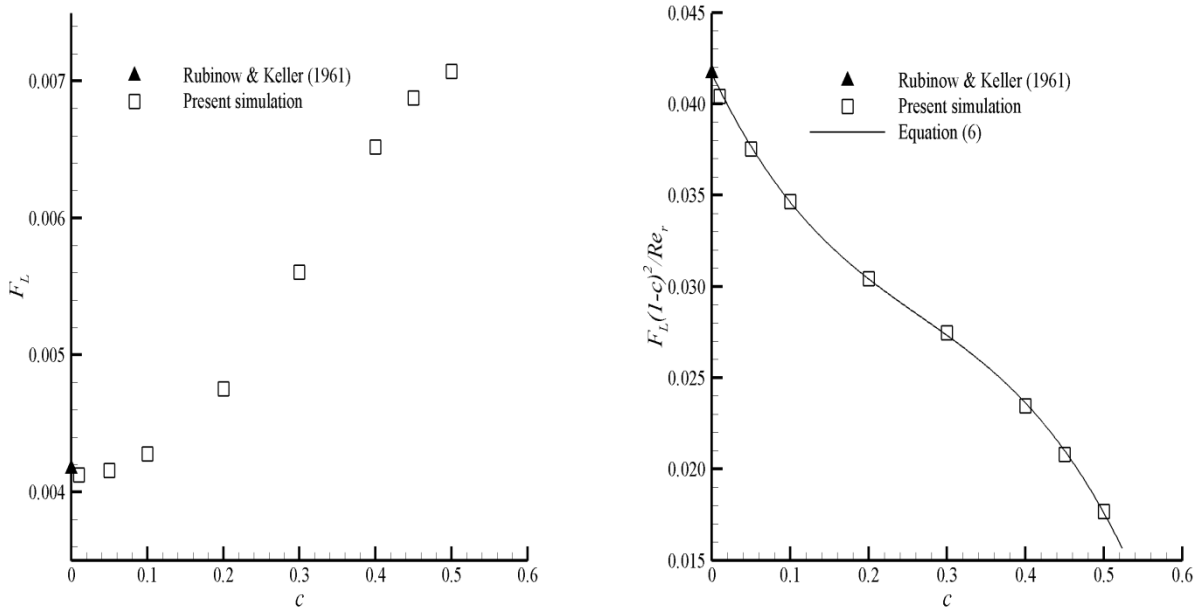
Equation 37 is modified in order to best fit the present simulation data with the theoretical result of Rubinow & Keller and also have the correct limiting behavior for $c \to 0$, as shown in Equation 39.

$$F_L(1-c)^2 / Re_r = (-0.287c^3 + 0.228c^2 - 0.0904c + 1/24). \quad (39)$$

This proposed formula naturally produces the correct results at the limit of vanishing c and the largest deviation of this formula with the present numerical data is less than 1%. Thus, Equation 50 can be used in the entire range of c from zero up to the packed limit of simple cubic arrays, which is $\pi/6 \approx 0.524$.

For arbitrary rotating axis directions, arbitrary rotational Reynolds numbers, and arbitrary solid volume fractions over low particle Reynolds number range, the dependence of the Magnus lift force on the rotational Reynolds number can be well described as

$$F_L(Re_r;\theta;c) = Re_r \sin(\theta)(-0.287c^3 + 0.228c^2 - 0.0904c + 1/24)/(1-c)^2 \quad c \le \pi/6 . \quad (40)$$



(a) (b)

Figure 10. (a) The normalized Magnus lift force F_L on the rotational spheres in simple cubic arrays as a function of the solid volume fraction at $Re_r = 0.1$; (b) $F_L(1\text{-}c)^2/Rer$ of the rotational spheres in simple cubic arrays as a function of the solid volume fraction. The solid line represents the best fit to the present data at $Re_r = 0.1$. The theoretical result calculated by Rubinow & Keller (1961) [25] at zero solid volume fraction also is shown in Figure 10 (a) and (b).

2.2.2. The lift force exerted on rotational spheres in random arrays at arbitrary variables such as rotational Reynolds numbers, particle Reynolds numbers, and packing fractions

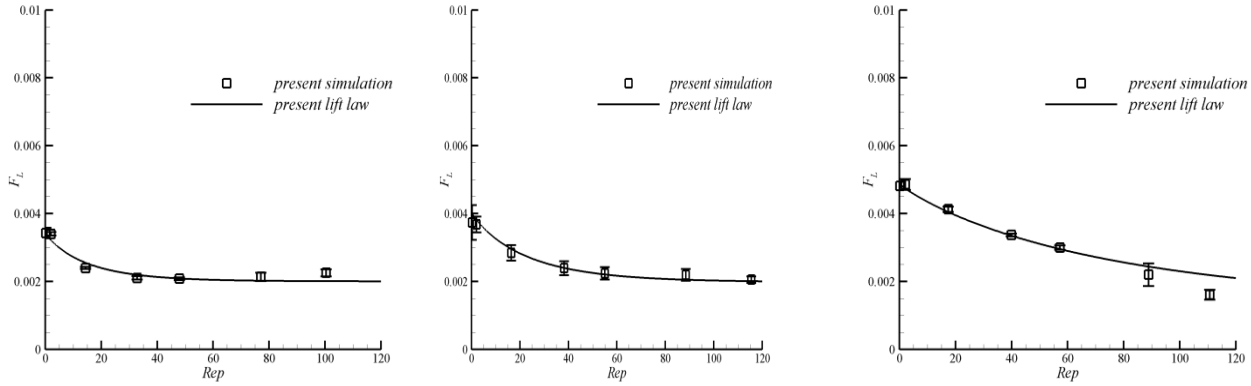
Magnus lift forces exerted on solid particles in random arrays are simulated for solid volume fractions up to the close-packed limits over the particle Reynolds number range of low to intermediate, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). A new formula for lift force was developed, based on the simulation results.

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$, $144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144.

Richardson extrapolation method is adopted to obtain the final results for each simulation case, based on the results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain credible simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate simulation results, though the computation of each configuration is time consuming.

The lift force of rotational spheres with the solid volume fractions of 0.1, 0.3 and 0.6 is simulated over the rotational Reynolds number ranges of 0.1 - 500. The lift force exerted on rotational spheres for each solid volume fraction is simulated at a rotational Reynolds number of 0.1. The lift force exerted on solid spheres is simulated at the particle Reynolds number range of effective zero to approximately 100 to explore inertia effects on the lift force.

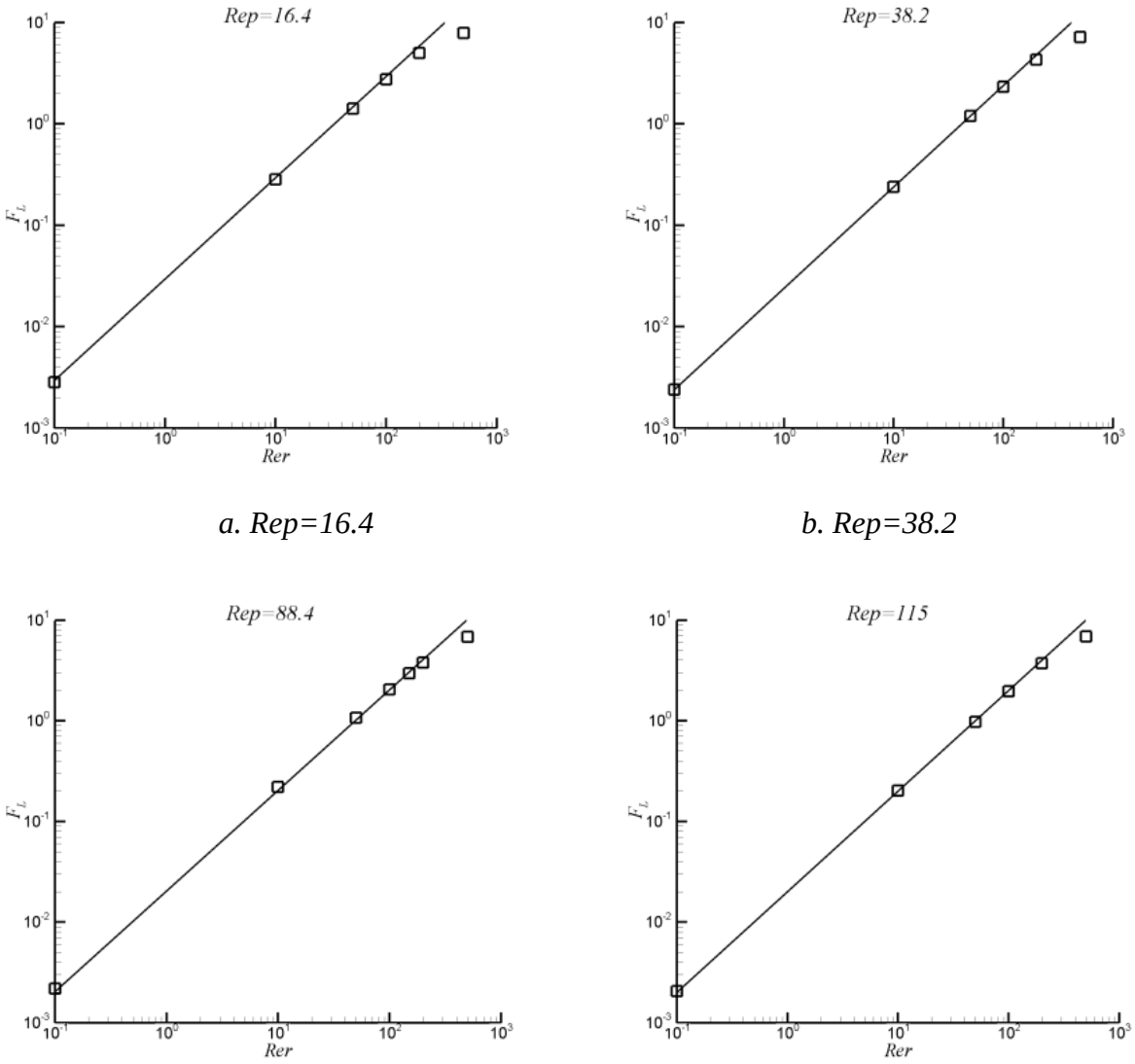
Figure 11 depicts the normalized lift force F_L exerted on the rotational spheres in random arrays as a function of particle Reynolds numbers at the rotational Reynolds number of 0.1. The lift force decreases progressively at a fixed rotational Reynolds number as the particle Reynolds number increases. The lift force at the lower solid volume fraction 0.1 tends to approach a steady value at the particle Reynolds number above 50, whereas the lift force at the larger solid volume fraction 0.6 decreases even at the particle Reynolds number above 100.



a. c=0.1 *b. c=0.3* *c. c=0.6*

Figure 11. The normalized lift force F_L exerted on the rotational spheres in random arrays as a function of particle Reynolds numbers at the rotational Reynolds number of 0.1.

The lift forces were simulated at the solid volume fraction 0.3 and various particle Reynolds numbers over the rotational Reynolds number range of 0.1 - 500 to investigate the dependence of the lift force on the rotational Reynolds number, as shown in Figure 12.



a. Rep=16.4 *b. Rep=38.2*

c. Rep=88.4 *d. Rep=115*

Figure 12. The normalized lift force F_L exerted on the rotational spheres in random arrays as a function of rotational Reynolds numbers at the solid volume fraction of 0.3 and various particle Reynolds numbers.

The lift force produced by the rotation movement is directly proportional to the rotational Reynolds number. The lift force is very insignificant at the rotational Reynolds number below than 1. However, the lift force can be larger than the drag force as the rotational Reynolds number becomes larger at especially low solid volume fractions.

2.2.3 The Magnus lift force on solid spheres in random arrays at arbitrary variables such as particle Reynolds numbers, rotational Reynolds numbers and packing fractions

Magnus lift forces exerted on solid particles in random arrays were simulated for solid volume fractions up to the close-packed limits over the particle Reynolds number range of low to intermediate, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formulas for Magnus lift force was developed with the simulation results..

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$, $144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144.

Richardson extrapolation method is adopted to obtain the final simulation results for each simulation case, based on the results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain credible simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate simulation results, though the computation of each configuration is time consuming.

Figure 13 depicts $F_L(1-c)^2/Re_r$ of solid spheres in random arrays as a function of the solid volume fraction c over low particle Reynolds numbers at $Re_r = 0.1$. Also shown in Figure 13 are the error bars that represent the standard deviations of the data points. This figure shows that the dependence of $F_L(1-c)^2/Re_r$ on c is approximately linear.

The best fit to describe this dependence is proposed, as shown in Equation 41.

$$F_L(1-c)^2/Re_r = -0.0398c + 0.0317 \quad c \le 0.637 . \tag{41}$$

The standard least-squares algorithm is adopted to calculate the coefficient in the new formula from extensive simulation results. The maximum difference in the entire range of packing fractions between Equation 41 and the simulation data is only around 5%, providing a good

support of the observed linear dependence. It is noted that the Magnus lift force exerted on solid particles in random arrays is relatively smaller than that in simple cubic arrays at all solid volume fractions. This is because the random positioning allows some of spheres to stay closer to each other, and making spheres hidden behind others experiences smaller flow velocity and consequently produce less Magnus lift force.

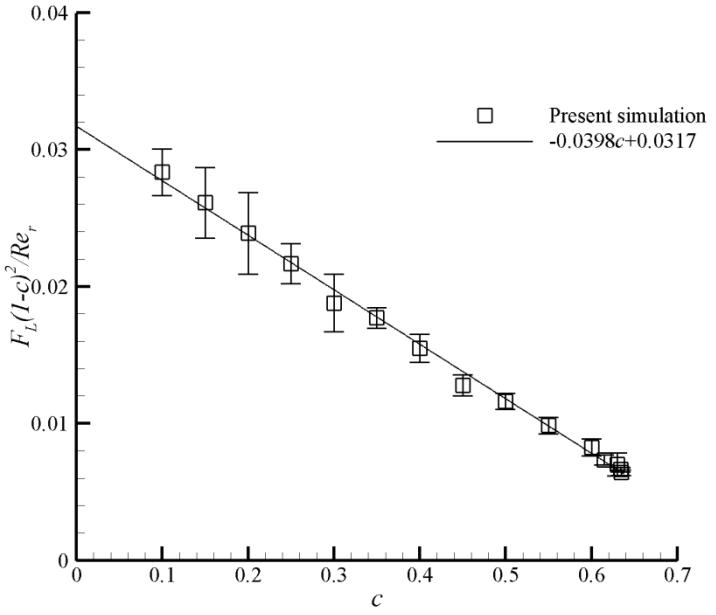


Figure 13. The normalized Magnus lift force F_L (multiplied by the porosity squared over the rotational Reynolds number) on the rotational spheres in random arrays as a function of the solid volume fraction over low particle Reynolds numbers. The error bars represent the standard deviations ΔF_L in F_L. The simulation data of F_L come from the simulations at $Re_r = 0.1$. A line is shown to fit the simulation results in the entire range of packing fractions.

A formula for the Magnus normalized lift force was proposed with extensive simulation results, as shown in Equation 42.

$$F_L = 0.02Re_r + (F_L|_{Re_p=0} - 0.02Re_r)\exp((-0.106 + 0.132c)Re_p^{0.9}), \quad (42)$$

where Re_r is the rotational Reynolds number and $F_L|_{Re_p=0}$ is the Magnus lift force at low particle Reynolds numbers, as shown in Equation 41.

Figure 14 compares the proposed formula with the simulation results, where the lift force values from the proposed formula are fairly agreeable with those from simulated results. The normalized Magnus lift force F_L at each solid volume fraction in Figure 14 approaches a constant value of 0.02 as the particle Reynolds number increases.

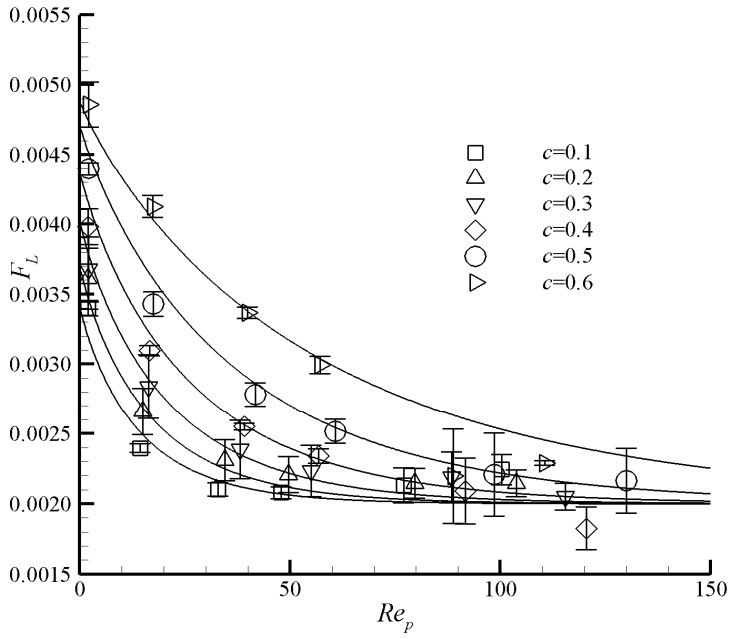


Figure 14. The normalized Magnus lift force F_L on the rotational spheres in random arrays as a function of the particle Reynolds number at $Re_r = 0.1$. The error bars represent the standard deviations in F_L. The solid lines for c =0.1, 0.2, 0.3, 0.4, 0.5 and 0.6 are generated from Equation 42.

The above normalized Magnus lift force formulas as shown in Equations 41 and 42 were proposed when the rotating axis direction of particles is perpendicular with the flow direction. For arbitrary rotating axis directions, our simulation results show that the dependence of the Magnus force on the rotational Reynolds number can be well described by simply changing the Re_r to $Re_r \sin(\theta)$, where θ is the angle between the rotating axis direction and the flow direction, as shown in Equations 43 and 44. It needs to be mentioned that, the change of θ does not significantly affect the expressions for the normalized torque and drag force.

$$F_L(Re_r;\theta;c) = Re_r \sin(\theta)(-0.0398c + 0.0317)/(1-c)^2 \quad c \le 0.637 . \tag{43}$$

Therefore, for arbitrary particle Reynolds number and packing fractions, the dependence of the lift force on the rotational Reynolds number can be well described as

$$F_L(Re_r;\theta;c) = \sin(\theta)\,[\,0.02Re_r + \left(F_L|_{Re_p=0} - 0.02Re_r\right)\exp\left((-0.106+0.132c)Re_p^{0.9}\right)] \quad (44)$$

2.3. The ratio of the Magnus lift force to the drag force

The ratio of F_L to F_D is calculated in the entire range of solid volume fractions c based on the simulation results as well as the proposed formulas to clearly identify the importance of the Magnus lift force F_L relative to the drag force F_D on spheres in random arrays,.

The values of F_L at $\theta = \pi/2$ only are used for simplicity. The ratios of F_L to F_D of solid partices in random arrays at low particle Reynolds numbers are shown in Figure 15. The line in Figure 15 is calculated from dividing the lift force formula (see Equation 41) by the drag force formula (see Equation 35). The ratio of F_L to F_D first decreases steeply in the range of c<0.1, and then gradually, as c further increases. Most of the simulation results agree well with the solid line data calculated from proposed formulas except those at $Re_r > 50$ for low solid volume fractions, which is due to the nonlinearity of the flow at $Re_r > 50$. Figure 15 demonstrates that the lift force can be very significant and even greater than the drag force when Re_r is up to $O(10^2)$, which is still in the practical range. Close observation reveals that the lift-to-drag ratio will be over unity in the solid volume fraction range of $c <$ approximately 0.17 at the rotational Reynolds number Re_r of 100.

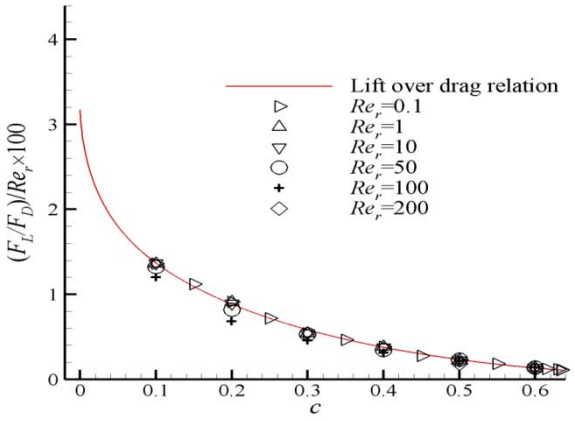


Figure 15. The ratio of the normalized Magnus lift force F_L to the normalized drag force F_D on the rotational spheres in random arrays over low particle Reynolds numbers as a function of the solid volume fraction. The solid lines are calculated from the proposed relations for F_L and F_D.

The lift force can be safely ignored in the high solid volume fraction range of $c > 0.3$ and the rotational Reynolds number range of $Re_r < 10$, where the lift-to-drag ratio is generally less than 0.07. However, the lift force of rotating spheres is significant at high rotation rates. For instance, the lift-drag-ratios for $c = 0.3$ and 0.5 at the rotational Reynolds number $Re_r = 100$ are around 0.58 and 0.23 respectively.

Figure 16 shows the lift-to-drag ratios at various solid volume fractions in the particle Reynolds (Re_P) number range of low to intermediate. The lines in Figure 16 are calculated from dividing the lift force formula (see Equation 42) by the drag force formula (see Equation 36). Ratios of the lift force to the drag force decrease with increased particle Reynolds numbers. However, the lift force still can be very significant when the rotational Reynolds number is around $O(10^2)$. The lift-to-drag ratios with the rotational Reynolds number Re_r=100 and the particle Reynolds number Re_p = 10 are 0.85 at c=0.1 and 0.11 at c=0.6, while the lift-to-drag ratios with the particle Reynolds number Re_p = 100 and the particle Reynolds number Re_p = 100 are 0.26 at c=0.1 and 0.03 at c=0.6.

Overall, it is believed that the Magnus lift force appearing in a direction perpendicular to the drag force with a non-negligible magnitude may appreciably affect and complicate the particle fluid dynamics in practical flow systems. Thus, the inclusion of the Magnus lift force in flow simulations is practically important.

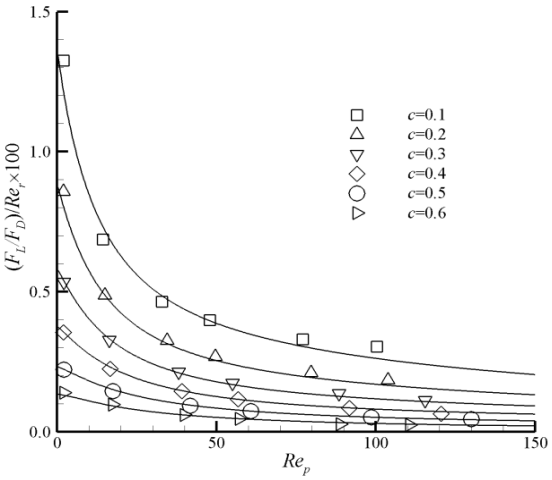


Figure 16. The ratio of the normalized Magnus lift force F_L to the normalized drag force F_D on the rotational spheres in random arrays as a function of the solid volume fraction in the particle Reynolds number (Re_P) range of low to intermediate. The solid lines are calculated with the proposed formulas for F_L (Equation 42) and F_D (Equation 36).

2.4. Torque

2.4.1 The normalized torque on solid spheres in simple cubic arrays over low particle Reynolds numbers.

Torque exerted on solid particles in simple cubic arrays was simulated for solid volume fractions up to the close-packed limits over the particle Reynolds number range of low to intermediate, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formula for torque force was developed, based on the simulation results.

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$, $144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144.

Richardson extrapolation method is adopted to obtain the final simulation results for each simulation case, based on the results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain credible simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate simulation results, though the computation of each configuration is time consuming.

The simulation results of the normalized torque exerted on spheres in simple cubic arrays are shown in Figure 17. The simulated torque value extrapolated to the zero solid volume fraction agrees with that predicted by Rubinow & Keller's theory [25]. This observation further validates the present numerical method. A formula for the normalized torque acting on the sphere in simple cubic arrays is proposed based on the present simulation results, as shown in Equation 45.

$$T = (-14.09c^4 + 10.26c^3 - 2.916c^2 - 0.7480c + 1)^{-1}. \quad (45)$$

This formula naturally produces correct torque values at the limit of vanishing c. The torque formulas of Rubinow & Keller (1961) [25] and Kirchhoff (1876) [24] predict that the torque value T is 1 at the zero solid volume fraction. The largest deviation of this formula with the present numerical simulation results is less than 1%. Though this formula is developed based on

the simulation torque data at the rotational Reynolds number $Re_r = 0.1$, it is valid for Re_r up to $O(10^2)$ due to the simple linear dependence of the torque on the rotational Reynolds number.

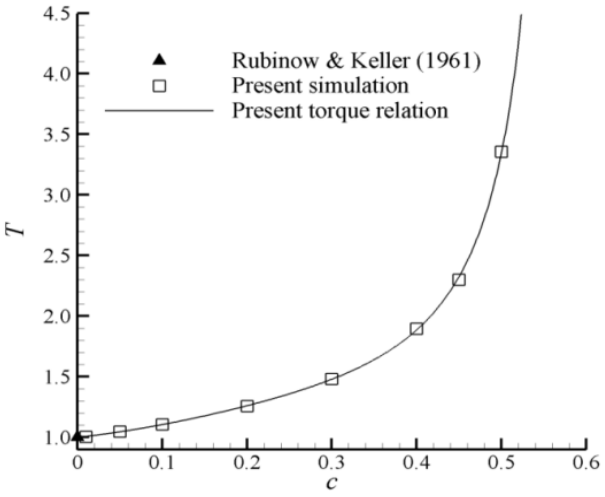


Figure 17. The normalized torque T exerted on the rotational spheres in simple cubic arrays as a function of the solid volume fraction c. The solid line represents the formula (Eq. (45)) to the present simulation results at $Re_r = 0.1$. Also shown is the theoretical result calculated by Rubinow & Keller (1961) [25] at zero solid volume fraction.

2.4.2 The normalized torque exerted on rotational spheres in random arrays over low and intermediate particle Reynolds numbers

Torque exerted on solid particles in random arrays was simulated for solid volume fractions up to the close-packed limits of random arrays in the particle Reynolds number range of low to intermediate, using a second-order accurate immerse boundary - lattice Boltzman method (IB-LBM). The formula for torque was developed, based on the simulation results.

A series of simulations (at least three) with increasing sphere diameters are conducted for each configuration. The simulated grid resolutions are $64\times64\times64$, $80\times80\times80$, $108\times108\times108$, $144\times144\times144$, and $192\times192\times192$. The numbers of the spheres simulated in the computational domain are all fixed to 144.

Richardson extrapolation method is adopted to obtain the final simulation results for each simulation case, based on the simulation results on the three finest meshes. The simulation results on 5-8 configurations at each solid volume fraction are usually averaged to obtain

credible simulation results. It is still desirable to simulate at least five configurations at each solid volume fraction to obtain accurate results, though the computation of each configuration is time consuming.

The normalized torque T on the rotational spheres in random arrays as a function of the solid volume fraction over low particle Reynolds numbers is depicted in Figure 18. The simulation results of T shown in Figure 18 also come from the simulations at $Re_r = 0.1$. It is remarked that the normalized torque T keeps nearly constant with increasing Re_r up to $O(10^2)$ and only slight deviation is observed at large Re_r. Therefore, the simulation results of T at a small Re_r such as $Re_r = 0.1$ are good enough to represent those of T at Re_r up to $O(10^2)$ for each solid volume fraction. The following formula of normalized torque T over low particle Reynolds numbers is proposed to best fit simulation results for the entire packing fractions

$$T = (-13.19c^4 + 14.41c^3 - 4.291c^2 - 0.9747c + 0.9465)^{-1} \tag{46}$$

The largest deviation of this fit from the present numerical simulation results is less than 1% for $c \le 0.5$ and around 5% for $c > 0.5$. The larger deviation at high c apparently is related to the sharp increase of T in that range (see Figure 18), which essentially is caused by the dramatically shrinking space among spheres as c approaches the packing limit.

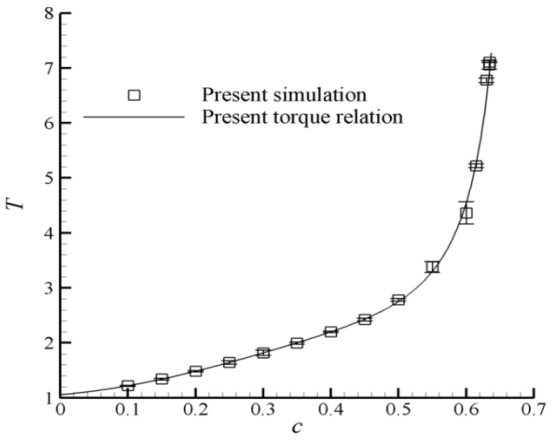


Figure 18. The normalized torque T on the rotational spheres in random arrays as a function of the solid volume fraction over the low particle Reynolds number range at $Re_r = 0.1$. The error bars represent the standard deviations in T. The solid line (Eq. 46) is shown to fit the simulation results in the entire range of packing fractions.

For intermediate particle Reynolds numbers, the normalized torque T on the rotational spheres in random arrays as a function of the particle Reynolds number are shown in Figure 19. The following expression is proposed to fit the torque results simulated at various solid volume fractions and particle Reynolds numbers,

$$T = g(c) + (T|_{Re_p=0} - g(c))\exp(f(c)Re_p^{0.9}), \tag{47}$$

where

$$g(c) = \exp(3.01c + 0.137), \tag{48}$$

$$f(c) = -0.0462 + 0.174c - 0.184c^2, \tag{49}$$

and $T|_{Re_p=0}$ represents the torque at low particle Reynolds numbers, which is from Equation 46. The agreement between Equation 58 and the simulation results is fairly good, as shown in Figure 19.

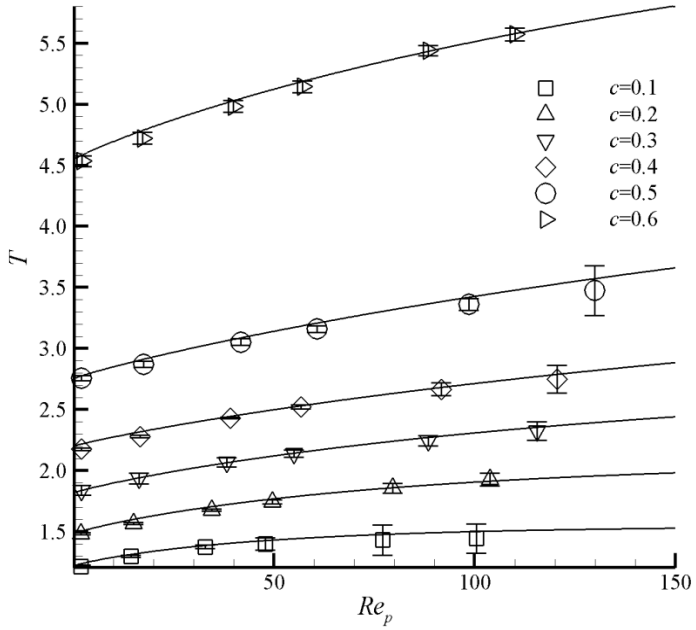


Figure 19. The normalized torque T on the rotational spheres in random arrays is shown as a function of particle Reynolds numbers. The error bars represent the standard deviations in T. The solid lines for c =0.1, 0.2, 0.3, 0.4, 0.5 and 0.6 are generated from Equation 47.

3. Installation of the proposed formulas in MFiX

Three new subroutines must be created to install the proposed formulas for the drag force, Magnus lift force, and the torque in the open source software MFiX. The three new subroutines in the file drag_gs.f are added to the framework of the original MFiX. The details of the three subroutines are listed, as shown in the Appendix.

4. Bubbling bed simulated with proposed formulas through MFiX

This numerical example has been tested by many researchers, such as Muller et al. (2009) [38] and Li et al. (2012) [39]. The parameter used can be found in detail in Li et al. (2012), thus, is omitted here.

Figure 20 depicts the lateral void fraction ($\varepsilon = 1 - c$) at 31.4 mm above the distributor and the superficial gas velocity of 0.6 m/s. Li et al. [44] also used the MFIX but with DES_INTERP_ON turned off. It can be seen that, after turning DES_INTERP_ON on, the present results and Gidaspow drag law (Gidaspow, 1994) [40] gives better prediction at the two sides of the bed but worse prediction at the middle particle of the bed. It can also be seen that, the present drag law gives similar results to those from Gidaspow drag law.

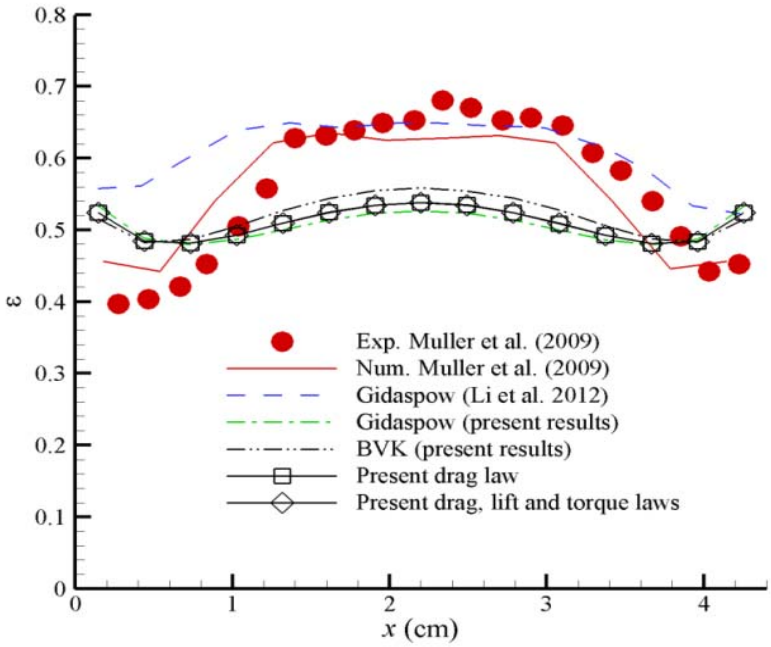


Figure 20. Lateral void fraction ($\varepsilon = 1 - c$) profiles at 31.4 mm above the distributor and the superficial gas velocity of 0.6 m/s. *x*: vertical distance from the wall of the boiling bed.

The BVK (Beetstra et al. 2007) [35] drag law is also tested. It also produces similar results compared to Gidaspow drag law and the present drag law. The Magnus lift law and the torque law are also simulated with the present drag law. However, no visible difference is produced compared to the case without invoking the Magnus lift law and the torque law. It might be because the current collision model cannot produce high spinning rates for particles and hence the lift force is very insignificant compared to the drag force.

Figure 21 depicts the lateral void fraction ($\varepsilon = 1 - c$) at 31.4 mm above the distributor and the superficial gas velocity of 0.9 m/s. The present drag law predicts best the void fraction at one side of the bubbling bed, and underestimates void fractions in the middle of the bed in comparison with the experimental data (Muller et al. 2009) [38]. However, the void fraction profiles produced by the present drag law appear to be asymmetric. Nevertheless, the better prediction of void fractions at one side of the bed is very promising. Comprehensive numerical studies are needed to fully understand the performance of the proposed formulas.

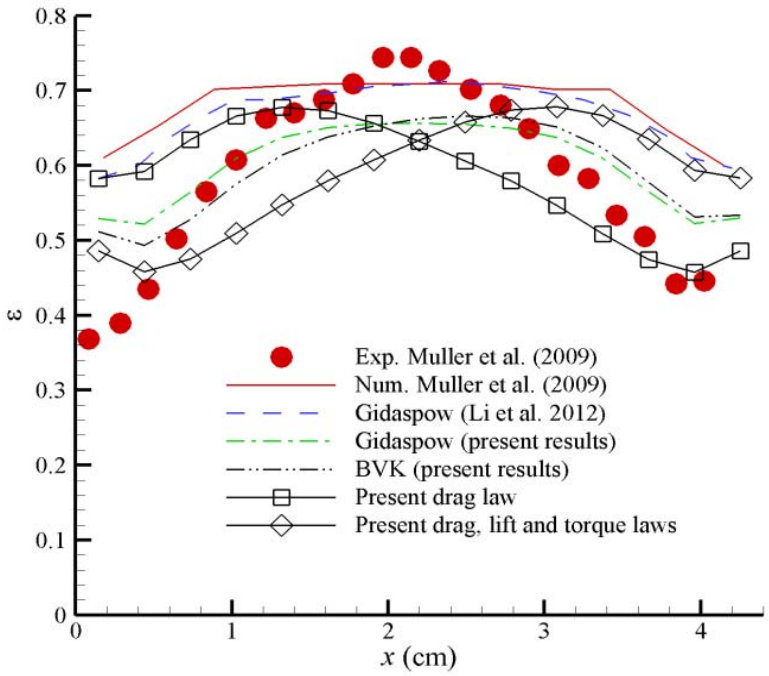


Figure 21. Lateral void fraction ($\varepsilon = 1 - c$) profiles at 31.4 mm above the distributor and the superficial gas velocity of 0.9 m/s. x: vertical distance from the wall of the boiling bed.

CONCLUSIONS

The multi-direct forcing method is adopted in the improved IBM to better approximate the no-slip/no-penetration (ns/np) condition on the surface of particles, and a slight retraction of the Lagrangian grid from the surface towards the interior of particles with a fraction of the Eulerian grid spacing helps increase the convergence rate of the direct numerical method. The method is further improved by an over-relaxation technique in the procedure of multi-direct forcing method and an implementation of the classical fourth order Runge-Kutta scheme in the coupled fluid-particle interaction. The over-relaxation technique is demonstrated to yield higher orders of convergence when the retraction distance is fixed.

An over-relaxation technique in the procedure of multi-direct forcing method and the classical fourth order Runge-Kutta scheme in the coupled fluid-particle interaction were applied to upgrade our old code by improving the accuracy of the combined IB-LBM. The use of the classical fourth order Runge-Kutta scheme helps the overall IB-LBM achieve the second order accuracy and provides more accurate predictions of the translational and rotational motion of particles. The old code with the first-order convergence rate is updated so that the updated new code can resolve the translational and rotational motion of particles with the second-order convergence rate. The updated code has been validated with several benchmark applications.

The new IB-LBM code has been further improved by the adoption of a new formula for the number of Lagrangian markers. Less Lagrangian markers are needed with the new formula than those used in the previous literature to achieve a desired computational accuracy. The simulation results with less Lagrangian markers in this study are in good agreement with literature results with more Lagrangian markers. Second-order convergence rate of numerical solutions can be achieved by slightly retracting Lagrangian markers from the surface towards the interior of particles with a fraction of the Eulerian grid spacing. In order to accomplish the major task of examining the drag force exerted on a cluster of particles, the IB-LBM code along with the new formula for the number of Lagrangian markers has been further validated by solving a set of theoretical problems.

The code has been upgraded even further to be capable of simulating simultaneously multiple particles. Besides that, the parallelization of the overall IB-LBM, which is very crucial

to speed up the further simulations, has also been done. The accuracy of the parallel code has been fully validated by comparing the results to those generated by the original serial code.

Flows through non-rotational and rotational spheres in simple cubic arrays and random arrays are simulated over the entire range of packing fractions, and both low and moderate particle Reynolds numbers to compare the simulated results with the literature results and develop a new drag force formula, a new lift force formula, and a new torque formula. Random arrays of solid particles in fluids are generated with Monte Carlo procedure and Zinchenko's method to avoid crystallization of solid particles with high solid volume fractions. Particle Reynolds number are kept very low to ensure flows of fluids around solid particles in the Stokes regime.

The simulated drag force exerted on the non-rotational spheres shows excellent agreement with the existing theories. Simulated drag forces of non-rotational particles as well as rotational particles are found to follow well the drag law proposed by Van Der Hoef et al except at the packing-limit solid volume fractions. A new drag force law is developed with extensive simulated results to be closely applicable to real processes over the entire range of packing fractions and both low and moderate particle Reynolds numbers. The simulation results indicate that the drag force is barely affected by rotational Reynolds numbers. Drag force is basically unchanged as the angle of the rotating axis varies.

A new lift force formula was developed with comprehensive simulated results as a function of arbitrary rotational Reynolds numbers over the entire range of packing fractions, and low and moderate particle Reynolds numbers. The lift force exerted on the rotational spheres is directly proportional to the rotational Reynolds number. The lift force also is very insignificant at rotational Reynolds numbers below 1. The lift force can be larger than the drag force as the rotational Reynolds numbers get higher especially at low solid volume fractions. In previous simulations, effects of the particle rotation on lift forces are not considered significant and thus the lift force is totally ignored in their drag law. This study demonstrated that the lift force caused by the particle rotation can be very significant compared to the drag force and must be considered in further study on two-fluid simulations.

The torque exerted on spheres by a fluid phase was investigated as another important factor to fully understand and quantify particle-fluid interactions in addition to the drag force and

the Magnus lift force exerted on solid spheres. The torque is also very essential to advance the angular momentum equation for solid particles in discrete particle simulations (DPM). A formula for the torque exerted on spheres in random arrays was developed with simulation results over the solid volume fraction range of effective zero up to the close-packed limits and low to intermediate particle Reynolds numbers. The normalized torque keeps nearly constant with low rotational Reynolds numbers, whereas slight deviation is observed at high rotational Reynolds numbers.

Simulations for a bubbling fluidized bed were performed, using the proposed new formulas through the MFiX. The proposed new drag formula for the bubbling fluidized bed with lower superficial gas velocity gives similar results compared to the previous drag laws such as Gidaspow and BVK, whereas the proposed new drag formula for a bubbling bed with higher superficial gas velocity predicts better void fraction at one side of the bubbling bed than the other side, and the void fraction profiles produced by the present drag formula appears to be asymmetric. The better prediction of the void fractions at one side of the bed is very promising. More comprehensive numerical studies are needed to fully understand the performance of the proposed formulas.

REFERENCES

[1]. Ladd A.J.C., Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part I. Theoretical foundation, J. Fluid Mech. 271 (1994) 285–310.

[2]. Ladd A.J.C., Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part II: Numerical results, J. Fluid Mech. 271 (1994) 311–339.

[3]. Peskin C.S..Numerical analysis of blood flow in the heart.Journal of Computational Physics. 25 (1977) 220-252.

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

[5]. Mohd-Yusof J., Combined immersed boundaries/B-splines methods for simulations of flows in complex geometries, Annual Research Briefs, Center for Turbulence Research, Stanford University, 1997.

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

[7]. Peskin C..The immersed boundary method, Acta Numerica 11 (2002) 1-39.

[8]. Kempe T., FrohlichJ.. An improved immersed boundary method with direct forcing for the simulation of particle laden flows. Journal of Computational Physics 231 (2012) 3663-3684.

[9]. Luo K., Wang Z., Fan J., Cen K., Full-scale solutions to particle-laden flows: Multi-direct forcing and immersed boundary method, Phys. Rev. E 76 (2007).

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

[11] Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H., Teller E.. 1953 Equation of state calculations by fast computing machines. J. Chem. Phys. 21, 1087.

[12]. Ferziger J.H., Peric M..Computational Methods for Fluid Dynamics, Springer-Verlag, Berlin, 2002.

[13] Zinchenko, A. Z. 1994 Algorithm for random close packing of spheres with periodic boundary conditions. J. Comput. Phys. 114, 298-306.

[14] Van der Hoef M. A., Beetstra R., Kuipers J. A. M., Lattice-Boltzmann simulations of low-Reynolds-number flow past mono- and bidisperse arrays of spheres: results for the permeability and drag force. J. Fluid Mech. (2005) vol. 528, pp. 233-254

[15]. Sui Y., Chew Y.T., Roy P., Low H.T., A hybrid method to study flow-induced deformation of three-dimensional capsules, Journal of Computational Physics 227(2008), 6351–6371

[16]. Roma A.M., Peskin C.S., Berger M.J., An adaptive version of the immersed boundary method, Journal of Computational Physics 153 (1999) 509-534.

[17]. Diebel J., Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors. Technical report, Stanford University, California, USA, Tech. Rep., 2006.

[18] Leopardi P., A partition of the unit sphere into regions of equal area and small diameter, Electron. Trans. Numer. Anal. 25 (2006) 309-327.

[19] Hasimoto, H. On the periodic fundamental solution of the Stokes equations and their application to viscous flow past a cubic array of spheres. J. Fluid Mech. (1959) 5, 317-328.

[20] Sangani, A. S. & Acrivos, A. Slow flow through a periodic array of spheres. Intl J. Multiphase Flow (1982) 8, 343-360.

[21] Zick, A. A. & Homsy, G. M. 1982 Stokes flow through periodic arrays of spheres. J. Fluid Mech. 115, 13-26.

[22] Carman, P. C.. 1937 Fluid flow through a granular bed. Trans. Inst. Chem. Engrs Lond. 15, 150-156.

[23] Koch, D. L. & Sangani, A. S. 1999 Particle pressure and marginal stability limits for homogeneous monodisperse gas fluidized bed: kinetic theory and numerical simulations. J. Fluid Mech. 400, 229.

[24] Kirchhoff, G. 1876 Vorlesungen ober mathematische Physik: Mechanik, ch. 26, Teubner, Leipzig.

[25] Rubinow S. I., Keller J. B.. 1961 The transverse force on a spinning sphere moving in a viscous fluid, J. Fluid Mech. 11(03) 447-459.

[26]. Bagchi P., Balachandar S..Effect of free rotation on the motion of a solid sphere in linear shear flow at moderate Re. Phys. Fluids 14 (2002) 2719-2737.

[27]. Uhlmann M., Experience with DNS of particulate flow using a variant of the immersed boundary method, in: P. Wesseling, E. Onate, J. Periaux (Eds.), Proceedings of the European Conference on Computational Fluid Dynamics (ECCOMAS CFD), TU Delft, The Netherlands, 2006.

[28] Sangani, A. S. & Acrivos, A. Slow flow through a periodic array of spheres. Intl J. Multiphase Flow (1982) 8, 343-360.

[29] Hill R.J., Koch D.L., Ladd A.J.C., The first effects of fluid inertia on flows in ordered and random arrays of spheres. J. Fluid Mech., (2001) 448: 213-241.

[30] Hill, R. J., Koch, D. L. & Ladd, A. J. C. 2001a The first effects of fluid inertia on flows in ordered and random arrays of spheres. J. Fluid Mech. (2001) 448, 243-278.

[31] Ladd, A. J. C. 1990 Hydrodynamic transport coefficients of random dispersions of hard spheres. J. Chem. Phys. 93, 3484.

[32] Hill, R. J., Koch, D. L. & Ladd, A. J. C. 2001b Moderate-Reynolds-number flows in ordered and random arrays of spheres. J. Fluid Mech. 448, 243.

[33] Ergun, S. 1952 Fluid flow through packed columns. Chem. Engng Proc. 48, 89

[34] Scott, G. D. & Kilgour, D. M. 1969 The density of random close packing of spheres. Brit. J. Appl. Phys. Ser. 2, 2, 863-866.

[35] Beetstra, R., Van Der Hoef, M.A., Kuipers, J.A.M., 2007. Drag force of intermediate Reynolds number flows past mono- and bidisperse arrays of spheres. AIChE J. 53, 489-501.

[36] Tenneti, S., Garg, R., Subramaniam S. 2011 Drag law for monodisperse gas-solid systems using particle-resolved direct numerical simulation of flow past fixed assemblies of spheres. International Journal of Multiphase Flow 37 (2011) 1072-1092.

[37] Oesterle, B. & Dinh, T. B. 1998 Experiments on the lift of a spinning sphere in a range of intermediate Reynolds numbers. Experiments in Fluids 25, 16-22.

[38] Muller C.R., Scott S.A., Holland D.J., Clarke B.C., Sederman A.J., Dennis J.S., et al., 2009 Validation of a discrete element model using magnetic resonance measurements, Particuology 7 (4) 297-306.

[39] Li T., Garg R., Galvin J., Pannala S. 2012 Open-source MFIX-DEM software for gas-solids flows: Part II-Validation studies. Powder Technology 220 (2012) 138-150.

[40] Gidaspow D. 1994, Multiphase Flow and Fluidization: Continuum and Kinetic Theory Descriptions, Academic Press, Boston

PUBLICATION

We plan to publish the proposed new formulas in a timely manner. These new formulas can be accessed and tested by the public. Readers of this report are also encouraged to explore the features with freedom. All the new files for the installation of the three laws into MFiX will be provided upon the request.

APPENDIX

Three new subroutines were created to install the proposed formulas for the drag force, Magnus lift force, and the torque in the open source software MFiX. The three new subroutines in the file drag_gs.f were added to the framework of the original MFiX. The details of the three subroutines are listed

```
!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC
!                                                                      C
!  Subroutine: DRAG_NEW                                                C
!  Purpose: Calculate the gas-solids drag coefficient                  C
!                                                                      C
!                                                                      C
!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC

      SUBROUTINE DRAG_NEW(lDgA, EPg, Mug, ROPg, VREL, &
                 DPM, DPA, PHIS)

!-----------------------------------------------
! Modules
!-----------------------------------------------
      USE param
      USE param1
      USE constant, only : PI
      IMPLICIT NONE
!-----------------------------------------------
! Dummy arguments
!-----------------------------------------------
! drag coefficient
      DOUBLE PRECISION, INTENT(OUT) :: lDgA
! gas volume fraction
      DOUBLE PRECISION, INTENT(IN) :: EPg
! gas laminar viscosity
      DOUBLE PRECISION, INTENT(IN) :: Mug
! gas density*EP_g
      DOUBLE PRECISION, INTENT(IN) :: ROPg
! magnitude of gas-solids relative velocity
      DOUBLE PRECISION, INTENT(IN) :: VREL
! particle diameter of solids phase M or
      DOUBLE PRECISION, INTENT(IN) :: DPM
! average particle diameter
      DOUBLE PRECISION, INTENT(IN) :: DPA
! total solids volume fraction of solids phases
      DOUBLE PRECISION, INTENT(IN) :: PHIS
!-----------------------------------------------
! Local variables
!-----------------------------------------------
! Reynolds number
      DOUBLE PRECISION :: RE
! Stokes Drag Force
      DOUBLE PRECISION :: F_STOKES
! dimensionless drag force F
      DOUBLE PRECISION :: F
!-----------------------------------------------

      IF(Mug > ZERO) THEN
! Note the presence of gas volume fraction in ROPG
         RE = DPA*VREL*ROPg/Mug        ! use DPA
      ELSE
```

```
         RE = LARGE_NUMBER
      ENDIF

      F_STOKES = 18D0*Mug*EPg/DPM**2    ! use DPM

      IF(phis <= 0.55D0) THEN
        F = 9.9d0*phis/EPg**2 + EPg**3*(ONE+3d0*(phis)**0.6d0)
      ELSEIF(phis <= 0.637D0)THEN
        F = 5.87d0*sin((phis/0.637d0)**1.75d0*PI/2.d0)/EPg**2
      ELSE
        F = 5.87d0/EPg**2
      ENDIF

      F = F + (0.256d0+1.41d0*phis-5.61d0*phis**2+6.04d0*phis**3)*RE/(24.d0*EPg**2) * &
             (ONE/EPg + 3d0*EPg*phis + 8.4d0/RE**0.343d0) / &
             (ONE+10.d0**(3d0*phis)/RE**(0.5d0+2.d0*phis))

      lDgA = F*F_STOKES
      IF (RE == ZERO) lDgA = ZERO

      RETURN
      END SUBROUTINE DRAG_NEW

!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC
!                                                                      C
!  Subroutine: MLIFT_NEW                                               C
!  Purpose: Calculate the Magnus lift force coefficient due to         C
!  particle rotation                                                   C
!                                                                      C
!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC

      SUBROUTINE MLIFT_NEW(lDgA,EPg,Mug,ROPg,VREL,MOMEGA,&
                  sinVOMEGAtheta,DPM,DPA,PHIS)

!-----------------------------------------------
! Modules
!-----------------------------------------------
      USE param
      USE param1
      IMPLICIT NONE
!-----------------------------------------------
! Dummy arguments
!-----------------------------------------------
! drag coefficient
      DOUBLE PRECISION, INTENT(OUT) :: lDgA
! gas volume fraction
      DOUBLE PRECISION, INTENT(IN) :: EPg
! gas laminar viscosity
      DOUBLE PRECISION, INTENT(IN) :: Mug
! gas density*EP_g
      DOUBLE PRECISION, INTENT(IN) :: ROPg
! magnitude of gas-solids relative velocity
```

```
      DOUBLE PRECISION, INTENT(IN) :: VREL
! magnitude of particle angular velcoity
      DOUBLE PRECISION, INTENT(IN) :: MOMEGA
! sin of the angle between gas-solids relative velocity and particle angular velcoity
      DOUBLE PRECISION, INTENT(IN) :: sinVOMEGAtheta
! particle diameter of solids phase M or
      DOUBLE PRECISION, INTENT(IN) :: DPM
! average particle diameter
      DOUBLE PRECISION, INTENT(IN) :: DPA
! total solids volume fraction of solids phases
      DOUBLE PRECISION, INTENT(IN) :: PHIS
!-----------------------------------------------
! Local variables
!-----------------------------------------------
! Reynolds number
      DOUBLE PRECISION :: RE
! Rotational Reynolds number
      DOUBLE PRECISION :: RER
! Stokes Drag Force
      DOUBLE PRECISION :: F_STOKES
! dimensionless drag force F
      DOUBLE PRECISION :: F
!-----------------------------------------------

      IF(Mug > ZERO) THEN
! Note the presence of gas volume fraction in ROPG
         RE = DPA*VREL*ROPg/Mug        ! use DPA
         RER =sinVOMEGAtheta*DPA**2*MOMEGA*ROPg/Mug
      ELSE
         RE = LARGE_NUMBER
         RER = LARGE_NUMBER
      ENDIF

      F_STOKES = 18D0*Mug*EPg/DPM**2    ! use DPM

      F = (-0.0398d0*phis+0.0317d0)/EPg**2        *RER

      F = RER*(0.02d0+(F-0.02d0)*exp((-0.106d0+0.132d0*phis)*RE**0.9d0))

      lDgA = F*F_STOKES*VREL !direction is from V cross OMEGA
      IF (RE == ZERO.or.RER == ZERO) lDgA = ZERO

      RETURN
      END SUBROUTINE MLIFT_NEW

!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC
!                                                                      C
!  Subroutine: TORQUE_NEW                                              C
!  Purpose: Calculate the Torque coefficient due to particle rotation  C
!                                                                      C
!                                                                      C
!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC
```

```
      SUBROUTINE TORQUE_NEW(lDgA,EPg,Mug,ROPg,VREL,MOMEGA,&
                  DPM,DPA,PHIS)

!-----------------------------------------------
! Modules
!-----------------------------------------------
      USE param
      USE param1
      USE constant, only : PI
      IMPLICIT NONE
!-----------------------------------------------
! Dummy arguments
!-----------------------------------------------
! drag coefficient
      DOUBLE PRECISION, INTENT(OUT) :: lDgA
! gas volume fraction
      DOUBLE PRECISION, INTENT(IN) :: EPg
! gas laminar viscosity
      DOUBLE PRECISION, INTENT(IN) :: Mug
! gas density*EP_g
      DOUBLE PRECISION, INTENT(IN) :: ROPg
! magnitude of gas-solids relative velocity
      DOUBLE PRECISION, INTENT(IN) :: VREL
! magnitude of particle angular velcoity
      DOUBLE PRECISION, INTENT(IN) :: MOMEGA
! particle diameter of solids phase M or
      DOUBLE PRECISION, INTENT(IN) :: DPM
! average particle diameter
      DOUBLE PRECISION, INTENT(IN) :: DPA
! total solids volume fraction of solids phases
      DOUBLE PRECISION, INTENT(IN) :: PHIS
!-----------------------------------------------
! Local variables
!-----------------------------------------------
! Reynolds number
      DOUBLE PRECISION :: RE
! Rotational Reynolds number
      DOUBLE PRECISION :: RER
! Stokes Torque
      DOUBLE PRECISION :: Torque_STOKES
! dimensionless toque Tor
      DOUBLE PRECISION :: Tor
! dimensionless function for torque
      DOUBLE PRECISION :: Torg,Torf
!-----------------------------------------------

      IF(Mug > ZERO) THEN
! Note the presence of gas volume fraction in ROPG
         RE = DPA*VREL*ROPg/Mug        ! use DPA
         RER =DPA**2*MOMEGA*ROPg/Mug
      ELSE
```

```
         RE = LARGE_NUMBER
         RER = LARGE_NUMBER
      ENDIF

      Torque_STOKES = PI*Mug*DPM**3    ! use DPM

      Tor = 1.d0/(-13.19d0*phis**4+14.41d0*phis**3-4.291d0*phis**2-&
           0.9747d0*phis+0.9465d0)
      Torg=exp(3.01d0*phis+0.317d0)
      Torf=(-0.0462d0+0.174d0*phis-0.184d0*phis**2)
      Tor =Torg+(Tor-Torg)*exp(Torf*RE**0.9d0)

      lDgA = Tor*Torque_STOKES
      IF (RER == ZERO) lDgA = ZERO

      RETURN
      END SUBROUTINE TORQUE_NEW
```

To use the new formulas, modification must be made in the subroutine named DES_DRAG_GP, which is in the file drag_fgs.f. The new subroutine of DES_DRAG_GP is pasted as follows,

```
!vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvC
!                                                                      C
!  Subroutine: DES_DRAG_GP                                             C
!  Purpose: Calculate the gas-particle drag coefficient using          C
!           the gas velocity interpolated to the particle position     C
!           and the particle velocity.                                 C
!           Invoked from des_drag_gs and calc_des_drag_gs              C
!                                                                      C
!  Comments: The BVK drag model and all drag models with the           C
!            polydisperse correction factor (i.e., suffix _PCF)        C
!            require an average particle diameter. This has been       C
!            loosely defined for discrete particles based on their     C
!            solids phase                                              C
!                                                                      C
!  Variables referenced:                                               C
!  Variables modified:                                                 C
!  Local variables:                                                    C
!                                                                      C
!^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^C

      !SUBROUTINE DES_DRAG_GP(LL, FLUID_VEL, PARTICLE_VEL)
      SUBROUTINE DES_DRAG_GP(LL, FLUID_VEL, PARTICLE_VEL,PARTICLE_OMEGA)

!-----------------------------------------------
! Modules
!-----------------------------------------------
```

```
      USE param
      USE param1
      USE fldvar
      USE geometry
      USE indices
      USE physprop
      USE run
      USE constant
      USE compar
      USE drag
      USE sendrecv
      USE discretelement
      USE ur_facs

      IMPLICIT NONE
!-----------------------------------------------
! Dummy arguments
!-----------------------------------------------
! particle number id.
      INTEGER , INTENT(IN) :: LL
! fluid velocity interpolated to particle position
      DOUBLE PRECISION, DIMENSION(3), INTENT(IN) :: FLUID_VEL
! particle velocity
      DOUBLE PRECISION, DIMENSION(3), INTENT(IN) :: PARTICLE_VEL
! particle angular velocity
      DOUBLE PRECISION, DIMENSION(3), INTENT(IN) :: PARTICLE_OMEGA
!-----------------------------------------------
! Local variables
!-----------------------------------------------
! indices, associated with current particle
      INTEGER :: IJK
! solids phase index, associated with current particle
      INTEGER :: M
! magnitude of gas-solids relative velocity
      DOUBLE PRECISION :: VREL
! components of gas-solids relative velocity
      DOUBLE PRECISION, DIMENSION(3) ::VRELcom
! magnitude of relative particle angular velocity
      DOUBLE PRECISION :: MOMEGA
! components of the cross production between gas-solids relative velocity
! and relative particle angular velocity
      DOUBLE PRECISION, DIMENSION(3) ::VcrossOMEGA
! magnitude of the above cross production
      DOUBLE PRECISION :: MVcrossOMEGA
!sin of the angle between gas-solids relative velocity and particle angular velcoity
      DOUBLE PRECISION :: sinVOMEGAtheta
! gas laminar viscosity redefined here to set viscosity at pressure
! boundaries
      DOUBLE PRECISION :: Mu
! drag coefficient
      DOUBLE PRECISION :: DgA
! Lift coefficient
```

```
      DOUBLE PRECISION :: MDgA
! Torque coefficient
      DOUBLE PRECISION :: TDgA
! current value of F_gs (i.e., without underrelaxation)
      DOUBLE PRECISION F_gstmp
! current value of F_gsM (i.e., without underrelaxation)
      DOUBLE PRECISION F_gsMtmp
! indices of solids phases (continuous, discrete)
      INTEGER :: CM, DM, L
! temporary shift of total number of solids phases to account for both
! discrete and continuous solids phases used for the hybrid mdoel
      INTEGER :: MAXM
! tmp local variable for the particle diameter of solids
! phase M (continuous or discrete)
      DOUBLE PRECISION :: DP_loc(2*DIM_M)
! tmp local variable for the solids volume fraction of solids
! phase M (continuous or discrete)
      DOUBLE PRECISION :: EPs_loc(2*DIM_M)
! tmp local variable for the particle density of solids
! phase M (continuous or discrete)
      DOUBLE PRECISION :: ROs_loc(2*DIM_M)
! correction factors for implementing polydisperse drag model
! proposed by van der Hoef et al. (2005)
      DOUBLE PRECISION :: F_cor, tmp_sum, tmp_fac
! average particle diameter in polydisperse systems
      DOUBLE PRECISION :: DPA
! diameter ratio in polydisperse systems
      DOUBLE PRECISION :: Y_i
! total solids volume fraction
      DOUBLE PRECISION :: phis
! aliases for void fraction, gas density, gas bulk density,
! solids volume fraction, particle diameter, particle density
      DOUBLE PRECISION :: EPG, ROg, ROPg, EP_SM, DPM, ROs
!-----------------------------------------------
! Include statement functions
!-----------------------------------------------
      INCLUDE '../function.inc'
      INCLUDE '../ep_s1.inc'
      INCLUDE '../ep_s2.inc'
!-----------------------------------------------

! values based on current particle
      IJK = PIJK(LL,4)
! solids phase index of current particle
      M = PIJK(LL,5)

! Assign local variables DP_loc, EPs_loc, and MAXM.  These
! represent arrays for the particle diameter, solids volume
! fraction, and number of particle types (i.e., phases).
      IF (.NOT.DES_CONTINUUM_HYBRID) THEN
         MAXM = DES_MMAX
         DO DM = 1,MAXM
```

```
            DP_loc(DM) = DES_D_p0(DM)
            EPs_loc(DM) = DES_ROP_S(IJK,DM)/DES_RO_S(DM)
            ROs_loc(DM) = DES_RO_S(DM)
         ENDDO
      ELSE   ! des_continuum_hybrid branch
! For the hybrid model the diameters and solids volume fractions of
! of both discrete and continuous are stored in this single quantity.
! Any loops of solids phases will include all solids phases (discrete
! and continuum)
         MAXM = SMAX + DES_MMAX
! populate DP, EPS starting with discrete phases
         DO DM = 1,DES_MMAX
            DP_loc(DM) = DES_D_p0(DM)
            EPs_loc(DM) = DES_ROP_S(IJK,DM)/DES_RO_S(DM)
            ROs_loc(DM) = DES_RO_S(DM)
         ENDDO
         DO CM = 1,SMAX
            L = DES_MMAX + CM
            DP_loc(L) = D_P(IJK,CM)
            EPs_loc(L) = EP_S(IJK,CM)
            ROs_loc(L) = RO_S(IJK,CM)
         ENDDO
      ENDIF   ! end if/else (.not.des_continuum_hybrid)


! magnitude of gas-particle relative velocity
 !     IF(NO_K)THEN
 !        VREL = SQRT((FLUID_VEL(1) - PARTICLE_VEL(1))**2 +&
 !                    (FLUID_VEL(2) - PARTICLE_VEL(2))**2)
 !     ELSE
 !        VREL = SQRT((FLUID_VEL(1) - PARTICLE_VEL(1))**2 +&
 !                    (FLUID_VEL(2) - PARTICLE_VEL(2))**2 +&
 !                    (FLUID_VEL(3) - PARTICLE_VEL(3))**2)
 !     ENDIF
 !
       IF(NO_K)THEN
         VRELcom(1)=FLUID_VEL(1) - PARTICLE_VEL(1)
         VRELcom(2)=FLUID_VEL(2) - PARTICLE_VEL(2)
         VRELcom(3)=ZERO
      ELSE
         VRELcom(:)=FLUID_VEL(:) - PARTICLE_VEL(:)
      ENDIF
         VREL =SQRT(VRELcom(1)**2+VRELcom(2)**2+VRELcom(3)**2)
! magnitude of particle anguler velocity
         MOMEGA=SQRT(PARTICLE_OMEGA(1)**2+PARTICLE_OMEGA(2)**2+&
                     PARTICLE_OMEGA(3)**2)
! cross product between gas-particle relative velocity and particle anguler velocity
         VcrossOMEGA(1)=VRELcom(2)*PARTICLE_OMEGA(3)-&
                        VRELcom(3)*PARTICLE_OMEGA(2)
         VcrossOMEGA(2)=VRELcom(3)*PARTICLE_OMEGA(1)-&
                        VRELcom(1)*PARTICLE_OMEGA(3)
         VcrossOMEGA(3)=VRELcom(1)*PARTICLE_OMEGA(2)-&
```

```
                        VRELcom(2)*PARTICLE_OMEGA(1)
! magnitude of the cross product
         MVcrossOMEGA=SQRT(VcrossOMEGA(1)**2+VcrossOMEGA(2)**2+&
                           VcrossOMEGA(3)**2)
!sin of the angle between gas-solids relative velocity and particle angular velcoity
         sinVOMEGAtheta=MVcrossOMEGA/(VREL*MOMEGA+SMALL_NUMBER)

! Laminar viscosity at a pressure boundary is given the value of the
! fluid cell next to it. This applies just to the calculation of the
! drag, in other routines the value of viscosity at a pressure boundary
! always has a zero value.
! This will never happen since this subroutine is currently only called
! for fluid_at cells (does not include flow boundaries)
! This points to an inconsitency in calculation of drag between
! continuum and discrete models that is probably not addressed in the
! solution of the gas phase momentum balances
      IF (P_OUTFLOW_AT(IJK)) THEN
         IF( FLUID_AT(EAST_OF(IJK) )) THEN
            Mu = MU_G(EAST_OF(IJK))
         ELSE IF ( FLUID_AT(WEST_OF(IJK)) ) THEN
            Mu = MU_G(WEST_OF(IJK))
         ELSE IF ( FLUID_AT(NORTH_OF(IJK)) ) THEN
            Mu = MU_G(NORTH_OF(IJK))
         ELSE IF ( FLUID_AT(SOUTH_OF(IJK)) ) THEN
            Mu = MU_G(SOUTH_OF(IJK))
         ELSE IF ( FLUID_AT(TOP_OF(IJK)) ) THEN
            Mu = MU_G(TOP_OF(IJK))
         ELSE IF ( FLUID_AT(BOTTOM_OF(IJK)) ) THEN
            Mu = MU_G(BOTTOM_OF(IJK))
         ENDIF
      ELSE
         Mu = MU_G(IJK)
      ENDIF

! calculate the total solids volume fraction
      phis = ZERO
      DO L = 1, MAXM
! this is slightly /= one-ep_g due to round-off
         phis = phis + EPs_loc(L)
      ENDDO

! calculate the average paricle diameter and particle ratio
      DPA = ZERO
      tmp_sum = ZERO
      tmp_fac = ZERO
      DO L = 1, MAXM
         IF (phis .GT. ZERO) THEN
            tmp_fac = EPs_loc(L)/phis
            tmp_sum = tmp_sum + tmp_fac/DP_loc(L)
          ELSE
            tmp_sum = tmp_sum + ONE/DP_loc(L) ! not important, but will avoid NaN's in empty cells
          ENDIF
```

```
      ENDDO
      DPA = ONE / tmp_sum
      Y_i = DP_loc(M) * tmp_sum

! assign variables for short dummy arguments
      EPg = EP_G(IJK)
      ROg = RO_G(IJK)
      ROPg = ROP_G(IJK)
      EP_SM = EPs_loc(M)
      DPM = DP_loc(M)
      ROs = ROs_loc(M)

! determine the drag coefficient
      IF (EP_SM <= ZERO) THEN
! this won't happen in DEM case since routine is performed over
! particles not cells as in continuum case
         DgA = ZERO
      ELSEIF (EPg == ZERO) THEN
! this case will already be caught in most drag subroutines whenever
! RE==0 (for correlations in which RE includes EPg). however, this will
! prevent potential divisions by zero in some models by setting it now.
         DgA = ZERO
      ELSE
! determine the drag coefficient
         SELECT CASE(DRAG_TYPE_ENUM)
         CASE (SYAM_OBRIEN)
            CALL DRAG_SYAM_OBRIEN(DgA, EPG, Mu, ROg, VREL, DPM)
         CASE (GIDASPOW)
            CALL DRAG_GIDASPOW(DgA, EPg, Mu, ROg, ROPg, VREL, DPM)
         CASE (GIDASPOW_PCF)
            CALL DRAG_GIDASPOW(DgA, EPg, Mu, ROg, ROPg, VREL, DPA)
         CASE (GIDASPOW_BLEND)
            CALL DRAG_GIDASPOW_BLEND(DgA, EPg, Mu, ROg, ROPg, VREL, DPM)
         CASE (GIDASPOW_BLEND_PCF)
            CALL DRAG_GIDASPOW_BLEND(DgA, EPg, Mu, ROg, ROPg, VREL, DPA)
         CASE (WEN_YU)
            CALL DRAG_WEN_YU(DgA, EPg, Mu, ROPg, VREL, DPM)
         CASE (WEN_YU_PCF)
            CALL DRAG_WEN_YU(DgA, EPg, Mu, ROPg, VREL, DPA)
         CASE (KOCH_HILL)
            CALL DRAG_KOCH_HILL(DgA, EPg, Mu, ROPg, VREL, DPM, DPM, phis)
         CASE (KOCH_HILL_PCF)
            CALL DRAG_KOCH_HILL(DgA, EPg, Mu, ROPg, VREL, DPM, DPA, phis)
         CASE (BVK)
            CALL DRAG_BVK(DgA, EPg, Mu, ROPg, VREL, DPM, DPA, phis)
!our new drag
         CASE (OURNEW)
            CALL DRAG_NEW(DgA, EPg, Mu, ROPg, VREL, DPM, DPA, phis)

         CASE (USER_DRAG)
            CALL DRAG_USR(IJK, M, DgA, EPg, Mu, ROg, VREL, DPM, ROs)
         CASE DEFAULT
```

```
            CALL START_LOG
            IF(DMP_LOG) WRITE (*, '(A,A)') &
               'Unknown DRAG_TYPE: ', DRAG_TYPE
            WRITE (UNIT_LOG, '(A,A)') 'Unknown DRAG_TYPE: ', DRAG_TYPE
            CALL END_LOG
            CALL mfix_exit(myPE)
         END SELECT   ! end selection of drag_type
      ENDIF   ! end if/elseif/else (ep_sm <= zero, ep_g==0)


      IF(DES_MAGNUS_LIFT)THEN
            IF (EP_SM <= ZERO) THEN
            MDgA = ZERO
            ELSEIF (EPg == ZERO) THEN
            MDgA = ZERO
            ELSE
                  CALL MLIFT_NEW(MDgA, EPg, Mu, ROPg, VREL, MOMEGA, &
                   sinVOMEGAtheta, DPM, DPA, phis)
            ENDIF
      ELSE
        MDgA = ZERO
      ENDIF


      IF(DES_FLUID_TORQUE)THEN
            IF (EP_SM <= ZERO) THEN
            TDgA = ZERO
            ELSE
                  CALL Torque_NEW(TDgA, EPg, Mu, ROPg, VREL, MOMEGA, &
                   DPM, DPA, phis)
            ENDIF
      ELSE
        TDgA = ZERO
      ENDIF


! Modify drag coefficient to account for possible corrections and
! for differences between Model B and Model A
      IF(DRAG_TYPE_ENUM == GIDASPOW_PCF .OR. &
         DRAG_TYPE_ENUM == GIDASPOW_BLEND_PCF .OR. &
         DRAG_TYPE_ENUM == WEN_YU_PCF .OR. &
         DRAG_TYPE_ENUM == KOCH_HILL_PCF .OR. &
         DRAG_TYPE_ENUM == BVK .OR. &
         DRAG_TYPE_ENUM == OURNEW ) THEN
! see erratum by Beetstra et al. (2007) : the correction factor differs
! for model A versus model B.
! application of the correction factor for model A is found from
! the correction factor for model B and neglects the Y_i**3 term
         IF(Model_B) THEN
            IF (M == 1) THEN
               F_cor = (EPg*Y_i + phis*Y_i**2)
            ELSE
```

```
               F_cor = (EPg*Y_i + phis*Y_i**2 + &
                  0.064d0*EPg*Y_i**3)
            ENDIF
         ELSE
            F_cor = Y_i
         ENDIF
         DgA = ONE/(Y_i*Y_i) * DgA * F_cor !
        MDgA = ONE/(Y_i*Y_i) * MDgA * F_cor !byZhou assume similar as DgA
        TDgA = ONE/(Y_i*Y_i) * TDgA * F_cor
      ENDIF

! Calculate the drag coefficient (Model B coeff = Model A coeff/EP_g)
      IF(MODEL_B) THEN
         F_gstmp = DgA * PVOL(LL)/EP_G(IJK)
         F_gsMtmp = MDgA * PVOL(LL)/EP_G(IJK)
      ELSE
         F_gstmp = DgA * PVOL(LL)
         F_gsMtmp = MDgA * PVOL(LL)
      ENDIF
!does torque need multiply volume?? no
! Determine drag force coefficient accounting for any under relaxation
! f_gp() =  single particle drag excluding vector(v_g - v_p)
      F_gp(LL) = (ONE - UR_F_gs) * F_gp(LL) + UR_F_gs * F_gstmp
!byZhou the relaxation is done with the value at previous time step
      F_gpM(LL) = (ONE - UR_F_gs) * F_gpM(LL) + UR_F_gs * F_gsMtmp
      T_gp(LL) = (ONE - UR_F_gs) * T_gp(LL) + UR_F_gs *  TDgA
      !F_gpM(LL)=ZERO
      !T_gp(LL)=ZERO
      RETURN
      END SUBROUTINE DES_DRAG_GP
```

In this section, only the major changes to the MFIX are provided. Many other minor changes, which are necessary to make the modified MFIX compile successful, will be provided upon the request. To invoke the proposed drag law, the keywords DES_CONTINUUM_COUPLED and DES_INTERP_ON must be assigned with .T.. To invoke the proposed Magnus lift formula and the torque formula, two new keywords, DES_MAGNUS_LIFT and DES_FLUID_TORQUE also are needed to attain the value of .T..