Algorithms in a Robust Hybrid CFD-DEM Solver for Particle-Laden Flows

Commun. Comput. Phys.
doi: 10.4208/cicp.260509.230210a

Vol. **9**, No. 2, pp. 297-323
February 2011

Algorithms in a Robust Hybrid CFD-DEM Solver for Particle-Laden Flows

Heng Xiao[1,*] and Jin Sun[2]

[1] *Institute of Fluid Dynamics, ETH Zürich, 8092 Zürich, Switzerland.*
[2] *Institute for Infrastructure and Environment, The University of Edinburgh, Edinburgh EH9 3JL, United Kingdom.*

Received 26 May 2009; Accepted (in revised version) 23 February 2010

Communicated by Boo Cheong Khoo

Available online 27 August 2010

Abstract. A robust and efficient solver coupling computational fluid dynamics (CFD) with discrete element method (DEM) is developed to simulate particle-laden flows in various physical settings. An interpolation algorithm suitable for unstructured meshes is proposed to translate between mesh-based Eulerian fields and particle-based Lagrangian quantities. The interpolation scheme reduces the mesh-dependence of the averaging and interpolation procedures. In addition, the fluid-particle interaction terms are treated semi-implicitly in this algorithm to improve stability and to maintain accuracy. Finally, it is demonstrated that sub-stepping is desirable for fluid-particle systems with small Stokes numbers. A momentum-conserving sub-stepping technique is introduced into the fluid-particle coupling procedure, so that problems with a wide range of time scales can be solved without resorting to excessively small time steps in the CFD solver. Several numerical examples are presented to demonstrate the capabilities of the solver and the merits of the algorithm.

PACS: 47.55.Kf, 47.57.Gc

Key words: Fluid-particle interaction, particle-laden flow, discrete element method, computational fluid dynamics, hybrid model.

1 Introduction

Particle-laden flows occur in many industrial and natural settings. For example, fluidized bed reactors are widely used in chemical and petroleum industries to carry out multiphase chemical reactions, where gas with high velocity is injected to beds of solid particles to achieve effective heat transfer, mass mixing, and accelerated reactions. In coastal engineering, waves mobilize and suspend sediments from beaches and transport sand

*Corresponding author. *Email addresses:* xiaoh@ethz.ch (H. Xiao), j.sun@ed.ac.uk (J. Sun)

particles with the flow. In these examples, the fluid-particle and particle-particle interactions play important roles. Fluid-particle interactions are also found in blood flows [1], sand dune evolution [2], and other geological flows.

In the systems described above, the fluids are often modeled in the Eulerian frame with Navier Stokes equations or their variants. The particles can also be modeled in the Eulerian frame, where particle volume fraction and particle flow field velocities are described and solved, and the individual particle motions are not tracked. This approach is referred to as two-fluid model [3,4], since the particle phase is also treated as a fluid. Another approach is to track the movement of each individual particle based on the forces exerted on the particle by the fluid and by other particles, which is referred to as the hybrid computational fluid dynamics-discrete element method (CFD-DEM) model. In both approaches, depending on the treatment of the fluid-particle and particle-particle interactions, the numerical methods can be categorized as one-way coupling (fluid-to-particle action only), two-way force coupling (fluid-particle mutual interactions), and four-way coupling (fluid-particle interactions and particle-particle collisions) [5]. For dilute flows with small solid volume fractions, one-way or two-way coupling are sufficient. For dense flows, which are common in fluidized-bed reactors and geological processes such as sediment transport, the solid volume fraction could be very high (above 60% in certain regions), and fluid-particle interactions and particle-particle collisions are both important. Another feature of geological flows and many other particle-laden flows is the wide range of particle sizes. We aim to model the dense flows occurring in industrial and natural processes with wide ranges of particle-size distributions, and thus the CFD-DEM approach with four-way coupling is necessary. The mathematical models and the numerical methods for these problems are the focuses of this paper.

Cundall and Strack [6] first used DEM to model granular flows without interstitial fluids in geotechnical engineering back in the 1970s. The hybrid CFD-DEM approach to model particle-laden flows was attempted later to solve industrial fluid-particle flows [7] and gained popularity in the past two decades. This approach has been used by many researchers to simulate multiphase flows in chemical engineering processes [8, 9] and other applications [10]. Wachem et al. [11] compared various formulations and closure models in the simulation of dense gas-solid systems. Kafui et al. [12] clarified the equations used in the literature for CFD-DEM modeling. On the numerical aspect, Sundaram and Collins [13] presented an efficient algorithm for their numerical model simulating dilute suspensions with fluid-particle and particle-particle interactions. However, there are still several difficulties associated with fluid-particle interactions in the dense flows that have not been addressed in previous studies. Specifically, the following challenges were encountered in our attempts to model dense particle-laden flows:

1. There is still a lack of proper schemes to translate between Eulerian fields based on unstructured meshes and Lagrangian particle quantities,

2. The fluid-particle interaction terms in the fluid momentum equation, if treated explicitly, often cause convergence difficulty in flows with high volume faction of particles.

An alternative treatment of the interactions with better stability performance but without compromising the accuracy is needed,

3. Particle beds with a wide range of diameter distributions can have multiple time scales and lead to stiff numerical problems, making conventional time stepping techniques inefficient.

In this paper, we present the numerical techniques to address these difficulties encountered in the development of a robust and efficient CFD-DEM fluid-particle solver, which is capable of simulating particle-laden flows in complex geometries with a wide range of particle sizes and volume fractions.

The mathematical models describing the fluid-particle system are presented in Section 2. The numerical methods used to solve the equations are discussed in Section 3. In Section 4, numerical examples are presented to validate the solver and to illustrate the merits of the proposed algorithm. Section 5 concludes the paper.

2 Mathematical formulation

2.1 Mathematical model for particle motion

In fluid-particle systems, the translational and rotational motions of a spherical particle can be described individually according to Newton's second law [8,14]

$$m_i \frac{\mathrm{d}\mathbf{u}_{pi}}{\mathrm{d}t} = \mathbf{f}_{ti} + \mathbf{f}_{fpi} + m_i \mathbf{g}, \tag{2.1}$$

$$I_i \frac{\mathrm{d}\boldsymbol{\Psi}_i}{\mathrm{d}t} = \boldsymbol{T}_i, \tag{2.2}$$

where i is the particle index; t is the time; $m_i, I_i, \mathbf{u}_{pi}, \boldsymbol{\Psi}_i$ are the mass, momentum of inertia, velocity, and angular velocity, respectively, of the particle; $\mathbf{f}_{ti}, \mathbf{f}_{fpi}, \boldsymbol{T}_i$ are the particle-particle contact force, fluid-particle interaction force, and torque, respectively, on particle i; $\mathbf{g}$ is the body force (gravity) per unit mass. The particles are modeled as viscoelastic spheres. The contact force between two particles is computed by assuming a linear spring-dashboard model [8]. The particle-particle and particle-wall collisions are characterized by the following physical parameters [14]: stiffness coefficient k (the elastic spring constant of the particles), restitution coefficient e (the percentage of elastic energy recovered during a particle-particle or particle-wall collision event, where $e=1$ corresponds to perfect collision with no elastic energy loss), and friction coefficient γ (the ratio between the maximum frictional force and the normal force during particle-particle or particle-wall contacts). In this study, it is assumed that the torque on a particle is solely due to the particle-particle contact, and the fluid-particle interactions do not contribute to the rotational motions of particles.

2.2 Mathematical model for fluid phase dynamics

The locally averaged governing equations for incompressible flows are used to describe the fluid phase [12, 15]

$$\frac{\partial}{\partial t}(\varepsilon_{\rm f}\rho_{\rm f})+\nabla\cdot(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f})=0, \tag{2.3}$$

$$\frac{\partial}{\partial t}(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f})+\nabla\cdot(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f}\mathbf{u}_{\rm f})=-\nabla p+\nabla\cdot\boldsymbol{\tau}_f+\varepsilon_{\rm f}\rho_{\rm f}\mathbf{g}-\mathbf{F}_{\rm fp}, \tag{2.4}$$

where $\varepsilon_{\rm f}$ is the fluid volume fraction; $\rho_{\rm f}$ is the fluid density; $\mathbf{u}_{\rm f}$ is the fluid velocity; p is the fluid pressure; $\boldsymbol{\tau}_f$ is the viscous stress of the fluid and $\nabla\cdot\boldsymbol{\tau}_f=\mu\nabla^2\mathbf{u}_{\rm f}$ for a Newtonian fluid. The fluid-particle interaction force, $\mathbf{F}_{\rm fp}$, per unit volume is

$$\mathbf{F}_{\rm fp}=\sum_{i=1}^{c_n}\mathbf{f}_{fpi}/V_c, \tag{2.5}$$

with

$$\mathbf{f}_{fpi}=-V_{pi}\nabla p+V_{pi}\nabla\cdot\boldsymbol{\tau}_f+\varepsilon_{\rm f}\mathbf{f}_{di}, \tag{2.6}$$

where c is the cell index; c_n is the number of particles in the cell with index c; V_{pi} is the volume of the particle with index i; V_c is the volume of cell c; the subscript f denotes fluid phase; $\varepsilon_{\rm f}\mathbf{f}_{di}$ (instead of $\mathbf{f}_{di}$) is the drag force applied on particle i. The particle drag force notation is adopted to be consistent with the conventions in the previous literature (see, for example, [3, 12, 16]) since $\mathbf{f}_{di}$ is the quantity usually obtained from experimental correlations.

Because both the fluid and the solid phases are assumed to be incompressible, the following mass conservation equation, which is equivalent to Eq. (2.3), is solved in our formulation

$$\nabla\cdot(\varepsilon_{\rm f}\mathbf{u}_f+\varepsilon_{\rm s}\mathbf{u}_s)=0, \tag{2.7}$$

where $\mathbf{u}_s$ is the velocity of the solid/particle phase; $\varepsilon_{\rm s}$ is the solid volume fraction. Equation (2.7) is analogous to the continuity equation for incompressible single phase flows (that is, $\nabla\cdot\mathbf{u}=0$, with $\mathbf{u}$ being the fluid velocity). Physically Eq. (2.7) means that the volume fraction-averaged velocity field of the mixture should be divergence-free, similar to that in the single phase flow. For the momentum conservation, Eqs. (2.5), (2.6), and the relation between $\boldsymbol{\tau}_{\rm f}$ and $\mathbf{u}_{\rm f}$ are plugged into Eq. (2.4) to yield the following equation [12]

$$\frac{\partial}{\partial t}(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f})+\nabla\cdot(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f}\mathbf{u}_{\rm f})=-\varepsilon_{\rm f}\nabla p+\varepsilon_{\rm f}\mu\nabla^2\mathbf{u}_{\rm f}-\varepsilon_{\rm f}\sum_{i=1}^{c_n}\mathbf{f}_{di}/V_c+\varepsilon_{\rm f}\rho_{\rm f}\mathbf{g}. \tag{2.8}$$

In the equations above, the fluid volume fraction, $\varepsilon_{\rm f}$, is not solved but obtained from the relation

$$\varepsilon_{\rm f}=1-\varepsilon_{\rm s},$$

where ε_s is obtained by averaging the Lagrangian particle quantities from the DEM simulations. The solid field velocity, $\mathbf{u}_s$, is obtained by using a similar procedure. The term $\varepsilon_f \sum_{i=1}^{c_n} \mathbf{f}_{di}$ is the sum of drag forces from all the c_n particles in cell c. This is only the conceptual representation. The actual implementation will be discussed in Section 3.1. The drag term $\mathbf{f}_{di}$ is correlated experimentally to the fluid and particle velocities [3,17]

$$\mathbf{f}_{di} = \frac{V_{pi}}{\varepsilon_f \varepsilon_s} \beta_i (\mathbf{u}_{pi} - \mathbf{u}_{fi}) = \frac{\pi d_{pi}^3}{6} \frac{1}{\varepsilon_f \varepsilon_s} \beta_i (\mathbf{u}_{pi} - \mathbf{u}_{fi}), \tag{2.9}$$

where V_{pi}, d_{pi}, $\mathbf{u}_{pi}$ are the volume, the diameter, and the velocity, respectively, of particle i; $\mathbf{u}_{fi}$ is the fluid velocity interpolated at the location of particle i; and β_i is the correlation coefficient. To facilitate the presentation of the fluid-particle interaction algorithm in Section 3.2, Eq. (2.9) is re-written as

$$\mathbf{f}_{di} = B_i (\mathbf{u}_{pi} - \mathbf{u}_{fi}), \quad \text{with} \quad B_i = V_{pi} \beta_i / \varepsilon_f \varepsilon_s \text{ (no summation over } i). \tag{2.10}$$

The specific form of β_i depends on the drag correlations used. In this study, the drag correlation by Syamlal and O'Brien is adopted [3]

$$\beta_i = \frac{3}{4} \frac{C_d}{V_r^2} \frac{\rho_f |\mathbf{u}_{pi} - \mathbf{u}_{fi}|}{d_{pi}} \varepsilon_s \varepsilon_f, \quad \text{with} \quad C_d = \left(0.63 + 4.8 \sqrt{V_r / \mathrm{Re}}\right)^2, \tag{2.11}$$

where the particle Reynolds number Re is defined as

$$\mathrm{Re} = \rho_f d_{pi} |\mathbf{u}_{pi} - \mathbf{u}_{fi}| / \mu, \tag{2.12}$$

where V_r is the ratio of the terminal velocity of a group of particles to the terminal velocity of a single particle, obtained from the following correlation [18]

$$V_r = 0.5 \left(A_1 - 0.06\mathrm{Re} + \sqrt{(0.06\mathrm{Re})^2 + 0.12\mathrm{Re}(2\mathrm{A}_2 - \mathrm{A}_1) + \mathrm{A}_1^2} \right), \tag{2.13}$$

with

$$A_1 = \varepsilon_f^{4.14}, \qquad A_2 = \begin{cases} 0.8\varepsilon_f^{1.28}, & \text{if } \varepsilon_f \le 0.85, \\ \varepsilon_f^{2.65}, & \text{if } \varepsilon_f > 0.85. \end{cases} \tag{2.14}$$

3 Numerical methods

The particle motion equations (2.1) and (2.2) are solved using a classical molecular dynamics simulation solver, LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator), developed at the Sandia National Laboratory [19]. It has been extensively used in previous studies of granular flows [8,14].

The fluid flow solver is partly based on a two-fluid model simulator developed by Rusche [20] utilizing the library OpenFOAM (Open Field Operation and Manipulation) [21]. The finite volume method is used to discretize the equations on an unstructured mesh. All variables are stored in the cell centers, that is, co-located grid is used. The

implicit Euler scheme is used for the time integration, which has only the first order accuracy but is unconditionally stable. The convection and diffusion terms are discretized with a blend of central difference (with second-order accuracy) and upwind difference (with first order accuracy). Further details of the numerical schemes are given in [20].

In the hybrid CFD-DEM solver, the fluid phase is solved in the Eulerian frame, and the particle phase is tracked in the Lagrangian frame. The solvers for the two phases are coupled using fluid-particle interaction models accounting for various forces such as drag, buoyancy, and added mass. In this section, three specific issues in the fluid-particle interactions are discussed: (i) the translation between the Eulerian fields and Lagrangian quantities; (ii) the treatment of fluid-particle interactions; and (iii) the effects of sub-stepping. Finally, the overall algorithm of the CFD-DEM solver is summarized.

3.1 Forward and backward interpolations between Lagrangian and Eulerian fields

In Eqs. (2.7) and (2.9), solid volume fraction, ε_s, and Eulerian mesh-based particle phase velocity, $\mathbf{u}_s$, are needed. They are obtained from the particle data in the DEM simulation via averaging procedures. The problem of calculating ε_s can be stated as: given a mesh and a set of particles with known locations and diameters, the mesh-based volume fraction field is to be calculated. Lagrangian polynomial interpolation is the most widely used method for the Lagrangian-Eulerian averaging processes in previous works (e.g., [22]). However, it requires an orthogonal grid and is not applicable when a three-dimensional (3D) unstructured mesh is used. A heuristic approach is taken here to explain the procedure used in this study.

For the convenience of illustration, we consider a generic two-dimensional (2D) system with three particles (P1 to P3) and four cells (C1 to C4) as shown in Fig. 1(a). A possible method of calculating the solid volume fraction is to sum up the volume of all the particle centered in each cell and then divide by the corresponding cell volumes (see e.g., [10]). With this procedure, the solid volume fraction, ε_{sc3}, for cell 3 would be

$$\varepsilon_{sc3} = \frac{1}{V_{c3}}(V_{p2} + V_{p3}), \tag{3.1}$$

where V_{c3} is the volume of cell 3; V_{p2} and V_{p3} are the volume of particles 2 and 3, respectively. This procedure, referred to as the *sum-up procedure* hereafter, may have difficulties when several large particles happen to have centers in one cell. Fig. 1(b) shows an exaggerated scenario, where the volumes of the four large particles all contribute to cell at the center (denoted with thick edges), making it possible for ε_s of this cell to approach unity. The problem is even more pronounced when the particles have a wide range of size distribution, because they could achieve more compact packing. When ε_s approaches unity, the drag force term obtained from Eq. (2.9) becomes unbounded, which in turn destabilizes the solution. In practice, instability often occurs whenever ε_s exceeds about 0.80 during a simulation.

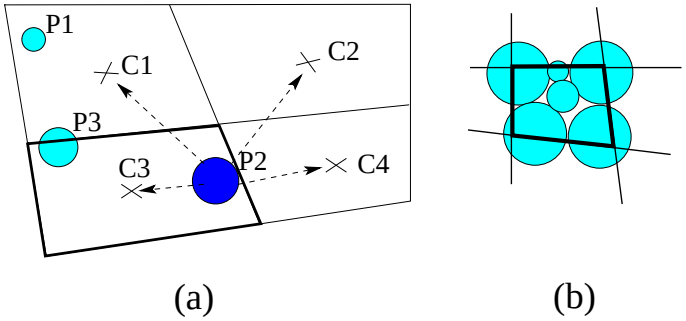

Figure 1: (a) Illustration of the averaging procedure using a generic and simple fluid-particle system with three particles and four cells. (b) An exaggerated scenario encountered when using the sum-up procedure (box-car kernel function) for the averaging. Polygons denote cells; Circles denote particles; X denote cell centers; P1-P3 and C1-C4 denote particle and cell numbering; Arrows denote particle-to-cell "contribution" of in solid volume calculation.

To overcome this difficulty, in the averaging process, the volume of each particles is distributed to all the cells according to its distance to the cell centers. Accordingly, a cell accepts "contributions" from all particles in the system. For example, the volume fraction of cell 3 is

$$\varepsilon_{sc3}=\frac{1}{V_{c3}}\Big(\omega_{1,c3}V_{p1}+\omega_{2,c3}V_{p2}+\omega_{3,c3}V_{p3}\Big), \tag{3.2}$$

where $\omega_{1,c3}$ to $\omega_{3,c3}$ are the weights of the contributions from particles 1-3 to cell 3. Clearly, the solid volume which all the cells "receive" must equal the sum of all the particle volumes, which is equivalent to requiring the weight functions to be normalized. This is ensured by choosing appropriate weighting functions for the contributions. For example, the weight $\omega_{2,c3}$ in Eq. (3.2) of the contribution from particle 2 to cell 3, can be computed as

$$\omega_{2,c3}=\frac{K\big(|\mathbf{x}_{p2}-\mathbf{x}_{c3}|/b\big)}{\sum_{m=1}^{4}K\big(|\mathbf{x}_{p2}-\mathbf{x}_{cm}|/b\big)}, \tag{3.3}$$

where K is a kernel function; $\mathbf{x}_{p2}$ and $\mathbf{x}_{c3}$ are the coordinates of particle 2 and cell 3, respectively; b is the bandwidth; and m is the general cell index. Similarly, the weights of the contributions from particle 2 to cells 1, 2, and 4 (denoted as $\omega_{2,c1}$, $\omega_{2,c2}$ and $\omega_{2,c4}$, respectively) can be obtained. Because

$$\sum_{m=1}^{4}\omega_{2,cm}=1,$$

the total solid volume contributed by particle 2 to all the cells is

$$\omega_{2,c1}V_{p2}+\omega_{2,c2}V_{p2}+\omega_{2,c3}V_{p2}+\omega_{2,c4}V_{p2}=V_{p2}. \tag{3.4}$$

Therefore, the normalization requirement above is satisfied no matter which form is chosen for the kernel function. This procedure is similar to the estimation of probability density function from discrete points, a problem frequently encountered in statistical data

analysis [23]. It is also applicable for the interpolation of other quantities such as velocity $\boldsymbol{u}_s$. It can be easily generalized to 3D cases.

In summary of the illustration above, the volume fraction of each cell can be calculated as

$$\varepsilon_{\rm sc} \equiv \varepsilon_{\rm s}(\mathbf{x}_c) = \frac{1}{V_c}\sum_{i=1}^{N_p}\omega_i V_{pi}, \tag{3.5a}$$

with

$$\omega_i = \frac{K\big(|\mathbf{x}_i - \mathbf{x}_c|/b\big)}{\sum_{m=1}^{N_c} K\big(|\mathbf{x}_i - \mathbf{x}_m|/b\big)}, \tag{3.5b}$$

where $\varepsilon_{\rm sc}$ is the solid volume fraction defined at the center of cell c; $\mathbf{x}$ is the coordinate of cells or particles; i is the general index for particles, which runs from 1 to N_p (the total number of particles in the system); The cell index m runs from 1 to N_c (the total number of cells); ω_i is the weight of the contribution from particle i, where the cell number is omitted from the subscript since it is clear that cell c is of concern in Eq. (3.5); The Gaussian function is a reasonable candidate for the kernel function K:

$$K(\zeta) = \exp[-\zeta^2],$$

where $\zeta = |\mathbf{x}_i - \mathbf{x}_c|/b$. However, due to efficiency considerations, the following kernel function is used in the implementation

$$K(\zeta) = \begin{cases} [1-\zeta^2]^4, & \text{if } |\zeta| < 1, \\ 0, & \text{if } |\zeta| \geq 1, \end{cases} \tag{3.6}$$

which is very close to the Gaussian function for ζ between 0 and 1, as shown in [24]. It can be seen that the sum-up procedure mentioned above is equivalent to using a box-car kernel function in Eq. (3.5) with bandwidths equal to the local cell sizes, which may not be globally uniform. Similarly, the solid phase velocity is obtained via the following averaging procedure

$$\mathbf{u}_{\rm sc} = \frac{1}{\varepsilon_{sc} V_c}\sum_{i=1}^{N_p}[\omega_i \mathbf{u}_i V_{pi}], \tag{3.7}$$

with ω_i given by (3.5b). In a similar approach, the drag force on the flow field per unit volume of mixture (or on cell c when discretized) is computed from drag forces from individual particles as follows:

$$\mathbf{f}_{dc} \equiv \mathbf{f}_d(\mathbf{x}_c) = \frac{1}{V_c}\sum_{i=1}^{N_p}[\omega_i \mathbf{f}_{di}], \tag{3.8}$$

where ω_i given by (3.5b), in lieu of the conceptual representation, $\mathbf{f}_{dc}=\sum_{i=1}^{c_n}\mathbf{f}_{di}/V_c$, in Eq. (2.8).

In Eqs. (3.5), (3.7), and (3.8), Eulerian fields are obtained from Lagrangian particle quantities through averaging procedure. This operation is called averaging or backward interpolation. Another operation required in the drag calculation is to calculate the fluid velocity, $\mathbf{u}_{fi}$, at the particle location, $\mathbf{x}_i$, given the fluid velocity field based on an unstructured mesh. This operation, referred to as forward interpolation, can be conducted using the same kernel function

$$\mathbf{u}_{fi}\equiv\mathbf{u}_{\mathrm{f}}(\mathbf{x}_i)=\sum_{c=1}^{N_c}\omega_c\mathbf{u}_c, \tag{3.9a}$$

with

$$\omega_c=\frac{K\big(|\mathbf{x}_i-\mathbf{x}_c|/b\big)}{\sum\limits_{m=1}^{N_c}K\big(|\mathbf{x}_i-\mathbf{x}_m|/b\big)}, \tag{3.9b}$$

where ω_c is the weight for cell c.

The interpolation operations in Eqs. (3.5) and (3.7)-(3.9) share a similar form

$$q=\sum_{j=1}^{N}\omega_j q_j, \tag{3.10}$$

where q is the quantity obtained via interpolation (scalar or vector); q_j is the source data points, either particle quantities or fluid phase quantities; ω_j is the weight for each source data point. For the convenience of notation, we use $\widetilde{\sum}_i$ to denote the *weighted sum* of the quantities from all the particles, that is,

$$\widetilde{\sum}_i q_i\equiv\sum_{i=1}^{N_p}\omega_i q_i. \tag{3.11}$$

For forward interpolations (from Eulerian fields to Lagrangian particle quantities), the index i in Eq. (3.11) is replaced with c. With the new notation, Eqs. (3.5), (3.7) (3.8), and (3.9) are written in a unified framework as

$$\varepsilon_{\mathrm{sc}}=\frac{1}{V_c}\widetilde{\sum}_i V_{pi},\quad \mathbf{u}_{\mathrm{sc}}=\frac{1}{\varepsilon_{sc}V_c}\widetilde{\sum}_i\mathbf{u}_i V_{pi},\quad \mathbf{f}_{\mathrm{dc}}=\frac{1}{V_c}\widetilde{\sum}_i\mathbf{f}_{di},\quad \text{and}\quad \mathbf{u}_{fi}=\widetilde{\sum}_c\mathbf{u}_c, \tag{3.12}$$

respectively. When the box-car kernel function is used, the weighted sum degenerates to the regular sum.

Although theoretically the interpolation procedure in Eq. (3.12) has a computational complexity of the order of $\mathcal{O}(N_pN_c)$, since all the particles and cells are examined at each step. In practice, however, the kernel functions have local supports, and thus only the

neighboring particles or cells need to be examined. Since the kernel function, K, depends on the particle-cell distance, $|\mathbf{x}_i-\mathbf{x}_c|$, which is a frame-independent norm, the results do not directly depend on the frame orientation and the mesh sizes.

In the simulations using the hybrid CFD-DEM solver, the Eulerian mesh has to be fine enough for the fluid simulation to achieve grid convergence and reasonable accuracy. On the other hand, in order to reduce the statistical error during the particle field averaging procedure, each fluid cell has to contain enough particles. To satisfy both requirements, a large number of particles may be needed in the system, which could make the computation very expensive, since the DEM simulations are computationally intensive. Introducing the kernel functions into the interpolation procedure can achieve small statistical errors with relatively small number of particles in each cell, and thus minimize mesh-dependence while keeping the computation cost low. In addition, in the scenarios where the cells are of nonuniform sizes, using the sum-up procedure would assume different bandwidths (equal to local cell sizes) at different locations, while the kernel-based averaging procedure is free from this limitation. Therefore, overall speaking, the averaging procedure presented above is better than the sum-up procedure, although the later has slightly lower computation cost, and is more consistent on the boundaries.

Although the bandwidth b in the kernel function can take any value theoretically, our experience shows that taking b between 1.5Δ and 2.5Δ (where Δ is the average cell dimension) gives the best results. Computational expenses increase significantly when the bandwidth is too large, while using very small bandwidth makes the procedure degenerate to the sum-up procedure and thus lose its advantages.

3.2 Robust treatment of fluid-particle interactions

In most of the previous studies of fluid-particle interactions using CFD-DEM approaches, the dynamic effects of the particles on the flow field are represented by a drag force term in the fluid momentum equation (see [9], [11], and the review in [25]). The drag term is often treated explicitly in the numerical solution. This method of handling fluid-particle interactions was originally proposed by Crowe et al. [26], where the algorithm was termed particle-source-in-cell model. Note that the inter-particle interactions were not accounted for in [26]. As reported in the literature (e.g., [27]) and according to our experience, convergence difficulty often occurs in dense particle-laden flows, because of the large external momentum source term ($\sum_{i=1}^{c_n}\mathbf{f}_{di}/V_c$ in Eq. (2.8)). In this study, this issue is addressed with a semi-implicit treatment of drag terms as described below.

Using the drag relation in Eqs. (2.9) and (2.10), the momentum equation (2.8) can be written as

$$\frac{\partial}{\partial t}(\varepsilon_{\mathrm{f}}\rho_{\mathrm{f}}\mathbf{u}_{\mathrm{f}})+\nabla\cdot(\varepsilon_{\mathrm{f}}\rho_{\mathrm{f}}\mathbf{u}_{\mathrm{f}}\mathbf{u}_{\mathrm{f}})=-\varepsilon_{\mathrm{f}}\nabla p+\varepsilon_{\mathrm{f}}\mu\nabla^2\mathbf{u}_{\mathrm{f}}+\frac{\varepsilon_{\mathrm{f}}}{V_c}\widetilde{\sum_i}B_i(\mathbf{u}_{pi}-\mathbf{u}_{fi})+\varepsilon_{\mathrm{f}}\rho_f\mathbf{g}, \tag{3.13}$$

where B_i is defined in Eq. (2.10). The drag force which the particles exert on the flow field (that is, the third term on the right-hand-side of Eq. (3.13)), is nonlinear in that the

coefficient B_i depends on the particle velocity $\mathbf{u}_{pi}$ and interpolated fluid velocity, $\mathbf{u}_{fi}$. This term is first linearized so that $\mathbf{u}_{pi}$ and $\mathbf{u}_{fi}$ from previous step is used to calculate B_i. The particle velocity $\mathbf{u}_{pi}$ is not a function of fluid velocity, and thus the contribution from the particle velocities toward the drag term can only be treated explicitly, along with other external source terms such as $\mathbf{g}$. However, the interpolated fluid velocity, $\mathbf{u}_{fi}$, at the particle location is a function of the fluid velocity field, $\mathbf{u}_{\rm f}$. Specifically, it is a linear combination of the fluid velocities at all the cell locations where $\mathbf{u}_{\rm f}$ is discretized. Therefore, it is possible to treat this term implicitly. To simplify the implementation, we use the cell closest to the particle to approximate the fluid velocity at the particle location instead of a weighted average of the fluid velocity from several (or all) cells. This simplification is for the convenience of implementation and is not theoretically essential. Separating the explicitly and implicitly treated terms, the fluid momentum equation can be written as

$$\frac{\partial}{\partial t}(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f})+\nabla\cdot(\varepsilon_{\rm f}\rho_{\rm f}\mathbf{u}_{\rm f}\mathbf{u}_{\rm f})=-\varepsilon_{\rm f}\nabla p+\underbrace{\varepsilon_{\rm f}\rho_f\mathbf{A}_c+\varepsilon_{\rm f}\rho_f\mathbf{g}}_{\text{Explicit treatment}}+\underbrace{\varepsilon_{\rm f}\mu\nabla^2\mathbf{u}_{\rm f}-\varepsilon_{\rm f}\rho_f\Omega_c\mathbf{u}_{fc}}_{\text{Implicit treatment}}, \tag{3.14}$$

with

$$\mathbf{A}_c=\frac{1}{\rho_{\rm f}V_c}\widetilde{\sum_i}B_i\mathbf{u}_{pi},\quad \Omega_c=\frac{1}{\rho_{\rm f}V_c}\sum_{i=1}^{c_n}B_i, \tag{3.15}$$

where $\mathbf{u}_{fc}$ is the fluid velocity of the cell of concern, that is, the discretized version of the fluid velocity field, $\mathbf{u}_f$. The term $\mathbf{A}_c$ is the acceleration on the fluid cell caused by the explicitly treated drag force, that is, the discretized version of the explicit drag acceleration field. This field is referred to as *explicit drag* for simplicity, but note that it has the dimension of acceleration. The term Ω_c is the coefficient for implicit drag and has the dimension of frequency (viz. [1/s]).

The semi-implicit treatment of the drag terms ensures good stability performance, easy implementation, and minimum mesh-dependence. Although this algorithm is not unconditionally stable, our experience suggests that in all the cases simulated in this study, the stability performance of the algorithm has been excellent. This is a major improvement over the fully explicit treatment in the current literature. Of course, a fully implicit treatment of the drag term would have even better stability performance (probably unconditionally stable). However, this cannot be easily implemented because of the difficulty in obtaining particle velocities for the current step.

Other components in the fluid-particle interactions include added masses, which are treated implicitly, and lift forces, which are treated explicitly. Added mass is important only when the fluid density is comparable to the particles density (e.g., sand particles in water).

The momentum equation (3.14) together with the mass conservation equation (2.7) are solved with a modified PISO algorithm. The original PISO algorithm with Rhie-Chow interpolation was developed to prevent pressure-velocity decoupling in fluid flow

simulations on co-located grids [28]. It was later modified for transient flow simulations [29,30] and two-fluid models [20]. The algorithm of Rusche [20] is used in this study with some modifications to accommodate the CFD-DEM coupling (refer to Section 3.4). At individual time step for the fluid flow, the PISO algorithm can be summarized as follows [20,30]:

(i) Momentum prediction: solve the fluid momentum equation using the pressure field and the external forcing from the previous step,

(ii) Pressure solution: solve the pressure equation, which can be obtained from Eq. (2.7), for the updated pressure field,

(iii) Velocity correction: correct the velocity field using the updated pressure field with the momentum equation,

(iv) Steps (ii) and (iii) are repeated until convergence.

The drag forces in Eq. (3.14) are computed based on each particles and then interpolated to the Eulerian field. An alternative approach is to average the particle quantities to mesh-based fields first, based on which the drag is then computed. However, to conserve momentum during the interaction, the computed drag needs to be distributed to the particles in the cell, which makes it difficult to study some phenomena such as segregation of poly-dispersed particles, since the choice of drag distributions scheme inevitably influences the results of the predictions [24]. The treatment presented here does not require drag distribution and addresses this difficulty well.

3.3 Particle relaxation time and sub-stepping

In the current algorithm, the drag force is constant for each fluid time step. Consider a scenario where a particle with zero velocity is released in a stream with constant velocity, U_e. Assuming the drag force computed according to initial relative velocity is applied on the particle without decrease due to the decrease of the slip velocity (defined as $\mathbf{u}_{ri}=\mathbf{u}_{pi}-\mathbf{u}_{fi}$), the relaxation time, τ_p, of a particle can be easily computed as follows using the drag correlation in Eq. (2.9)

$$\tau_p=\frac{U_e}{\varepsilon_{\mathrm{f}} f_{di}/m_i}=\frac{\rho_s\varepsilon_{\mathrm{s}}}{\beta_i}, \tag{3.16}$$

where ρ_s is the particle density; f_{di} is computed according to Eq. (2.9), with the slip velocity taken as U_e since the particle is stationary initially. The relaxation time above has a similar interpretation as the particle characteristic time in the definition of Stokes number. Inserting the expression for β_i in Eq. (2.11) into (3.16) yields

$$\tau_p=\frac{\rho_s\varepsilon_{\mathrm{s}}}{\frac{3}{4}\frac{C_d}{V_r^2}\frac{\rho_{\mathrm{f}}U_e}{d_{pi}}\varepsilon_{\mathrm{f}}\varepsilon_{\mathrm{s}}}=\frac{4\rho_s d_{pi}V_r^2}{3C_d U_e\rho_{\mathrm{f}}\varepsilon_{\mathrm{f}}}. \tag{3.17}$$

Table 1: Exemplary relaxation time (in seconds) for different volume fractions, fluids, fluid velocities, and particle diameters. Parentheses indicate cases which may need sub-stepping.

Volume fraction	$\varepsilon_s = 0.2$		$\varepsilon_s = 0.6$		Unit
Diameter	0.1	1	0.1	1	mm
Water (3 m/s)	(7.0×10^{-5})	(1.0×10^{-3})	(3.0×10^{-5})	(4.0×10^{-4})	s
Water (1 m/s)	(1.5×10^{-4})	3.0×10^{-3}	(6.7×10^{-5})	(1.0×10^{-3})	s
Air (3 m/s)	3.0×10^{-2}	0.72	1.3×10^{-2}	0.29	s
Air (1 m/s)	3.9×10^{-2}	1.5	1.9×10^{-2}	0.66	s

The actual relaxation time is a few times larger than the value in Eq. (3.16) because of the exponential decay of the drag force as the particle velocity approaches fluid velocity.

Relaxation time for several different flow and particle conditions are presented in Table 1, considering two solid volume fractions, $\varepsilon_s = 0.2$ and 0.6. Other drag correlations [17, 31] are tested and similar results are obtained. When the relaxation time is smaller than or close to the fluid time step, velocity oscillations or even instabilities may occur. One way to circumvent this problem without decreasing the fluid step is to update the particle velocity and drag force on the particle within one fluid step. This procedure is called *sub-stepping*. Fig. 2 illustrates the fluid-particle interaction procedure, particularly the drag update procedure, with and without sub-stepping.

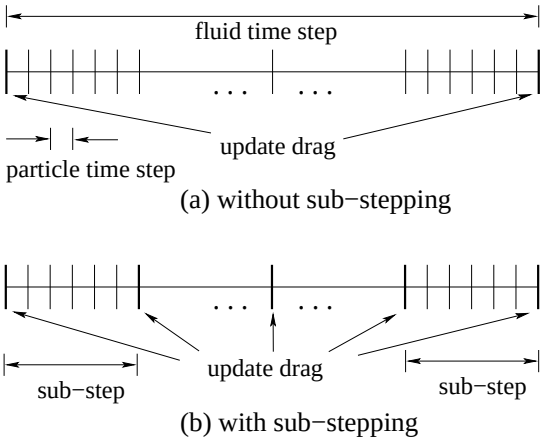

Figure 2: Schematic of particle time step (separated by vertical thin lines), sub-steps (separated by vertical thick lines), and fluid time step (the whole horizontal line), as well as the drag update in algorithms (a) without sub-stepping and (b) with sub-stepping.

For fluid flows with properties in the range of interest and the mesh sizes used in this study, the typical order of magnitude of time steps for the CFD simulation is $\Delta t_f = 10^{-3}$s, considering the Courant-Friedrichs-Lewy (CFL) condition and the possible destabilizing effects caused by the fluid-particle interactions. From Table 1, it is seen that when simulating particle-laden water flows (usually with smaller Stokes numbers than gas flows), sub-stepping is generally necessary if the particle sizes are smaller than approximately

1mm. The cases which may need sub-stepping are indicated with parentheses in Table 1, if the fluid step size is to be kept of the order of 10^{-3}s.

Without sub-stepping, the momentum exchange between the two phases during fluid step n is $\varepsilon_{\rm f}\mathbf{f}_{di}^{n}\Delta t_f$, with $\varepsilon_{\rm f}\mathbf{f}_{di}^{n}$ being the drag force on particle i calculated at the beginning of fluid step n. If sub-stepping is used such that $\Delta t_{\rm sub}=\Delta t_f/N_s$, with $t_{\rm sub}$ being the time interval after which the drag forces on the particles are updated (sub-stepping size) and N_s being the number of sub-steps per fluid time step, then the momentum exchange ($\mathbf{I}_{di}$) between particle i and the fluid is

$$\mathbf{I}_{di}=\sum_{k=1}^{N_s}\varepsilon_{\rm f}\mathbf{f}_{di}^{n,k}\Delta t_{\rm sub}=\frac{1}{N_s}\sum_{k=1}^{N_s}\frac{V_{pi}\beta_i\mathbf{u}_{ri}^{k}}{\varepsilon_{\rm s}}\Delta t_f \tag{3.18}$$

$$=\frac{1}{N_s}\sum_{k=1}^{N_s}\frac{V_{pi}\beta_i(\mathbf{u}_{fi}-\mathbf{u}_{pi}^{k})}{\varepsilon_{\rm s}}\Delta t_f=\frac{V_{pi}\beta_i\Delta t_f}{\varepsilon_{\rm s}}\Big(\mathbf{u}_{fi}-\frac{1}{N_s}\sum_{k=1}^{N_s}\mathbf{u}_{pi}^{k}\Big), \tag{3.19}$$

where n is the fluid time step index and k is the sub-step index during fluid step n. The average value for $\mathbf{f}_{di}$ during the fluid step n is

$$\mathbf{f}_{di}^{n}=\frac{V_{pi}\beta_i}{\varepsilon_{\rm s}\varepsilon_{\rm f}}\Big(\mathbf{u}_{fi}-\frac{1}{N_s}\sum_{k=1}^{N_s}\mathbf{u}_{pi}^{k}\Big). \tag{3.20}$$

Comparing Eq. (3.20) with (2.9), the average particle velocity from all the sub-steps (viz. $\bar{\mathbf{u}}_{pi}^{n}=\sum_{k=1}^{N_s}\mathbf{u}_{pi}^{k}/N_s$) should be used in place of $\mathbf{u}_{pi}$ when calculating the drag force term $\mathbf{A}_c$ in Eq. (3.15), in order to conserve momentum in the fluid-particle interactions. The average velocity $\bar{\mathbf{u}}_{pi}^{n}$ can be considered as the ensemble average of the particle velocities based on all the sub-steps during the $n^{\rm th}$ fluid step.

It is emphasized that if all particles in the fluid are of very small Stokes number, the particles can be modeled as passive tracers because the particles follow fluid motion closely without dynamics effects, and thus the sub-stepping procedure or even the DEM simulation for the particles is unnecessary. The sub-stepping procedure is intended for fluid-particles systems with a *wide range of Stokes numbers* (and thus relaxation time scales) caused by the presence of some very fine particles. An example of such case is shown in Section 4.4. It is apparent that reducing fluid time step according to the requirement specified by Eq. (3.17) would certainly yield stable solution. However, it is often not desirable to use excessively small time step and CFL number for the fluid simulation due to accuracy considerations. Note that saving computational expense in the fluid simulations is not the major benefit of using the sub-stepping scheme. Taking for example Case 4 in Section 4.4, the saving of computational time by using sub-stepping is only 3%, which is relatively insignificant. This is because the DEM simulation is much more computationally intensive than the CFD simulation, and thus the major computational expense is associated with the DEM simulation. Therefore, reducing the computation time spent on the CFD simulation has only minor effects on the overall computational expense.

To ensure stability, the number of sub-steps N_s should be chosen as approximately $\Delta t_f/\tau_{p,\min}$, where $\tau_{p,\min}$ is the smallest relaxation time associated with all the particles. Usually the particles are of the same density and thus $\tau_{p,\min}$ is the relaxation time of the smallest particles.

3.4 Overall algorithm of fluid-particle interactions with CFD-DEM coupling

In summary, the overall algorithm for the fluid-particle interactions is presented as follows (refer to the pseudo-code in Algorithm 3.1 for the numbered steps mentioned below):

Algorithm 3.1: Overall fluid-particle interaction algorithm with sub-stepping

Initialize drag coefficient terms (Ω_c^0 and $\mathbf{A}_c^0$);

for $n=1$ *to* N_t **do**

1. Evolve fluid fields from $\mathbf{u}_f^{n-1}$, p^{n-1} to $\mathbf{u}_f^n$, p^n (using PISO);

2. Calculate drag terms Ω_c^n and $\mathbf{A}_c^n$;

for $k=1$ *to* N_s **do**

3. Compute fluid drag ($\mathbf{f}_{di}^{n,k}$) on each particle;

4. Evolve particle states with DEM simulator;

end

5. Obtain Eulerian fields ($\mathbf{u}_s^n$ and ε_s^n) for fluid step n via averaging.

end

N_t is the number of fluid steps; n is the fluid step index; N_s is the number of sub-steps per fluid step; k is the sub-step index. Further details of each step are explained in Section 3.4.

Given the initial fluid fields ($\mathbf{u}_f^0$, p^0) and particle state (positions $\mathbf{x}_p^0$ and velocities $\mathbf{u}_p^0$), along with appropriate boundary conditions, the objective is to evolve the fluid-particle system by certain amount of time (N_t steps of size Δt_f). At each fluid step, the fluid fields are first solved with the modified PISO algorithm, and the fluid fields are evolved from state $n-1$ to n (see Step 1 in Algorithm 3.1). Solving the fluid equation requires the information of the particle drag on the fluid fields. The drag information from previous fluid step, Ω_c^{n-1} and $\mathbf{A}_c^{n-1}$, is used.

In Step 2 (refer to Algorithm 3.1), based on the newly obtained fluid fields (u^n and p^n) and current particle states (position $\mathbf{x}_p^{n-1}$ and velocity $\mathbf{u}_p^{n-1}$), the particle drag terms are computed according to Eq. (3.15), and thus Ω_c^n and $\mathbf{A}_c^n$ are obtained. The calculation involves backward interpolation, which is detailed in Section 3.1. In addition, as mentioned in Section 3.3, when computing Ω_c and $\mathbf{A}_c$, the ensemble particle velocity ($\bar{\mathbf{u}}_{pi}^n$) from the previous fluid step should be used in lieu of the current particle velocity. The evolution of particle states is conducted in N_s sub-steps.

In each sub-step, the drag forces are computed according to fluid fields at current time step n and particle state at current sub-step k (refer to Step 3 in Algorithm 3.1). With the information of fluid particle interaction forces, the particle states are evolved from ($\mathbf{u}_p^{n-1}$, $\mathbf{x}_p^{n-1}$) to ($\mathbf{u}_p^n$, $\mathbf{x}_p^n$) with the DEM simulator in N_s sub-steps, where each sub-step consists of a certain number of particle steps (refer to Step 4).

Finally in Step 5, (that is, after all the sub-steps and at the end of each fluid step), the Eulerian fields (ε_s and $\mathbf{u}_s$) are obtained from the particle quantities ($\mathbf{u}_p^n$ and $\mathbf{x}_p^n$) with the averaging procedures detailed in Section 3.1. At the end of the fluid step, all the quantities and fields are all advanced to the new state, including the flow field ($\mathbf{u}_f^n$ and $\mathbf{p}^n$), the particle state ($\mathbf{u}_p^n$ and $\mathbf{x}_p^n$), and the Eulerian particle fields ($\mathbf{u}_s^n$ and ε_s^n).

4 Numerical examples

Four numerical examples are presented in this section. In Case 1, a mono-dispersed bed is fluidized by upward air flow with gradually increasing velocity. This case is used to verify the numerical model. In Case 2, a similar bed consisting of bi-dispersed particles is fluidized by air flow at a constant velocity. The influence of kernel functions on the mesh-dependence is studied by comparing the results with those obtained using the sum-up procedure. Case 3 illustrates the effects of sub-stepping with the motion of one particle in a constant water stream. Case 4 is an example of the transport and segregation of natural sand by groundwater flows. The particle properties for all the cases are presented in Table 2. Particle-particle and particle-wall collision parameters are as follows: particle stiffness coefficient $k = 5000$ N/m, restitution coefficient $e = 0.7$, and friction coefficient between particles $\gamma = 0.15$. The walls are assumed to be smooth and rigid. The fluid domain dimensions are specified in each case. Pseudo-3D simulations are conducted for all the cases. Specifically, the particle simulations are conducted in 3D, while the fluid simulations are conducted in 2D. This is justified by the fact that the variation of fluid motion in the thickness direction is small due to the small thickness of the bed. Note that with the presence of very small particles (compared to the domain thickness) such as in Case 4, the flow field is not strictly 2D. However, the dynamic effects on the flow field due to the presence of the particles also decrease with decreasing particle sizes. Therefore, the flow field variation in the thickness dimension is not considered. In all the simulations except Case 3 (where there is only one particle), the particles are initially settled at the bottom of the bed, with particles of all types randomly packed and uniformly mixed. Unless noted otherwise, the fluid mesh has $15\times75\times1$ cells (in x, y, and z directions, respectively; same convention hereafter), the fluid (CFD) time step is 10^{-3}s, and particle (DEM) time step is 10^{-6}s.

The dynamic viscosity and density of the air are $\mu_a = 1.8\times10^{-5}$ [Pa s] and $\rho_a = 1.2$ [kg/m^3], respectively. For water, we use $\mu_w = 1.0\times10^{-3}$ [Pa s] and $\rho_w = 1\times10^3$ [kg/m^3]. The gravity constant is $g = 9.8$m/s^2. The added mass is considered when the interstitial fluid is water, and an added mass coefficient 0.5 is used. Lift forces are neglected.

Table 2: Particle properties for the computational Cases 1, 2, 3, and 4.

Model validation (Case 1)		
d_p	1.5mm	
ρ_s	2000kg/m^3	
U_{mf} (theoretical*)	84cm/s	
Number of particles	2160	
Mesh-dependence study (Case 2)		
	Small particles	Large particles
d_p	1.52mm	2.49mm
ρ_s	2523kg/m^3	2526kg/m^3
U_{mf} (theoretical**)	$\sim$92cm/s	$\sim$130cm/s
Number of particles	1783	377
Sub-stepping study (Case 3)		
d_p	0.083mm	
ρ_s	2000kg/m^3	
Number of particles	1	
Groundwater seepage problem (Case 4)		
d_p	0.083 — 2.5mm	
ρ_s	2000kg/m^3	
U_{mf} (theoretical**)	0.01 — 2.5cm/s	
Number of particles	2160	

* Obtained with drag correlation in Eq. (2.11) assuming $\varepsilon_s = 0.554$, the median value of the initial solid volume fraction field.

** Assuming $\varepsilon_s = 0.583$. This is the measured value in the experiment [32] according to which this case is composed. Note that the bed with bi-dispersed particles in this case has higher volume fraction than the mono-dispersed bed in Case 1.

As an overview of the computational demand of the solver, the CPU time required to simulate 2000 particles (Cases 1, 2, and 4) for a physical time of 20 s is about 6 hours on one processor (AMD Opteron® 2.3GHz with 4 GB memory). Most of the CPU time is consumed by the DEM simulations, particularly when the particles are densely packed, because more particle-particle collisions need to be handled.

4.1 Case 1: model validation

In this numerical example, air flow is injected into the bottom of a particle bed with gradually increasing velocity. The domain geometry as well as the initial and boundary conditions are shown in Fig. 3 and Table 3. The fluid pressure and velocity are monitored at two heights: $y_1 = 0.2$ cm and $y_2 = 15$ cm, as indicated in Fig. 3. It is observed that the flow fields are approximately uniform along the horizontal (x) and depth (z) directions, and thus the x and y coordinates of the monitored locations are not of concern.

The inlet air velocity linearly increases from 0m/s to 1.5m/s, and then linearly decreases to 0m/s, as shown in Fig. 4. The pressure drop keeps increasing as the inlet air velocity increases until the bed is fluidized. After that, the pressure drop keeps fluc-

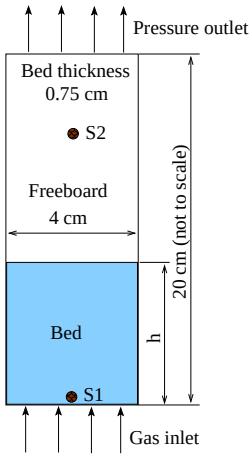

Figure 3: Geometry of the pseudo 3D computational domain and boundary conditions for all the simulations. The two filled circles indicate the locations of "virtual sensors" where pressure and velocities are monitored in Cases 1 and 2.

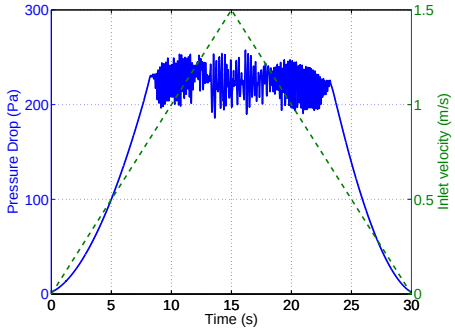

Figure 4: Verification study of the numerical solver (Case 1): Inlet velocity (dashed line) and the corresponding pressure drop (solid line) through the bed as measured by the pressure difference of the two points (S1 and S2) in Fig. 3.

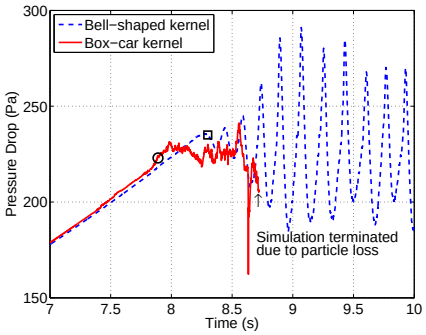

Figure 5: Performance comparison of the box-car and the bell-shaped kernel functions in the simulation of bed fluidization. The time series of inlet velocity is the same as that shown in Fig. 4. The points corresponding to the bed fluidization on the results from box-car and bell-shaped kernel functions are marked with "◯" and "□", respectively. The point where the simulation using box-car kernel function terminated due to particle loss is annotated with an arrow.

Table 3: Computational domain sizes and initial and boundary conditions for Cases 1, 2, and 4.

Geometry	
Height of domain (L_y)	20cm
Width of domain (L_x)	4cm
Depth of domain (L_z)	0.75cm
Initial bed height (h in Fig. 3)	
model validation (Case 1)	$\sim$ 2.5cm
mesh dependence (Case 2)	$\sim$ 4cm
Groundwater seepage (Case 4)	$\sim$ 2cm
Boundary conditions	
Uniform fluid inflow	Specified velocity
Pressure	Zero gradient at outlet

tuating around a constant value but with increasing fluctuation amplitude as the inlet flow velocity increases. The inlet velocity at which the pressure drop stops increasing is the minimum fluidization velocity of the bed. The pressure drop at this point should approximately balance the gravity of the particle bed (air weight is ignored here due to its low density). When the inlet flow velocity decreases below the fluidization velocity again, the pressure drop decreases. However, the decreasing and the increasing paths are of hysteresis, because the fluidization/de-fluidization process is history-dependent. The minimum fluidization velocity and the corresponding pressure drop are $U_{\rm mf}=83$cm/s and $\Delta p=235$Pa, respectively. The theoretical prediction of $U_{\rm mf}$ depends on the solid volume fraction, $\varepsilon_{\rm s}$. It is estimated to be $U_{\rm mf}=84$cm/s assuming $\varepsilon_{\rm s}=0.554$, which is the median value of the initial solid volume fraction field. The numerical prediction is very close to the theoretical value. The pressure drop across the fluidized bed can be theoretically estimated according to

$$\Delta p=N_p m_i g/(L_x L_z),$$

where $N_p m_i$ is the total mass of all the particles. The estimation gives 254Pa according to the particle and domain properties in Tables 2 and 3. This is close to the prediction from the simulation. The remaining discrepancy (less than 10%) is possibly explained by the jam effects from the wall.

To evaluate the performance of the box-car and bell-shaped kernel functions, the simulation described above is repeated using both kernel functions. In this comparison, a finer mesh with $25\times100\times1$ cells is used, because the differences between the two kernel functions are more pronounced for fine meshes, and fine meshes are often required to achieve better accuracy in CFD simulations. The time series of the pressure drop, Δp, obtained by using box-car kernel function is shown in Fig. 5, and is compared with the results obtained using the bell-shaped kernel function. Fig. 5 is zoomed in to show the time when the fluidization occurs. The points corresponding to the bed fluidization on the results from box-car and bell-shaped kernel functions are marked with a circle and a square, respectively.

Table 4: Comparison of predictions of minimum fluidized velocity ($U_{\rm mf}$) and pressure drop (Δp) in the simulation of Case 1, using the box-car and the bell-shaped kernel functions.

	Box-car	Bell-shaped	Theoretical prediction *
$U_{\rm mf}$	79 cm/s	83 cm/s	84 cm/s
Δp	223 Pa	235 Pa	254 Pa

* Note: Using the median value of the initial solid volume fraction ($\varepsilon_{\rm s}$) field, 0.554.

The minimum fluidization velocity, $U_{\rm mf}$, and the corresponding pressure drop, Δp, obtained using the two kernel functions are presented in Table 4, and are compared to the theoretical predictions. It can be seen that the bell-shaped kernel function gives better agreements with the theoretical values in terms of both $U_{\rm mf}$ and Δp. In particular, it is noted that when the box-car kernel function is utilized in the averaging procedure, the predicted minimum fluidization velocity, $U_{\rm mf}$, is lower than the numerical predictions from the bell-shaped kernel function and the theoretical value. This is because the sum-up procedure inevitably gives unphysically high solid volume fractions in some cells (see Fig. 1(b) for a more detailed explanation) and thus leads to artificially early fluidization. During the simulation, when the computed solid volume fraction in any of the cells is too high (e.g., larger than 0.80), particles in these cells may gain very large velocities, which leads to instability of the whole system and these particles may even leave the simulation domain. In these scenarios, the simulation has to be terminated, which is indicated in Fig. 5 for the box-car kernel function case with arrow.

According to our experiences during this study, when box-car kernel functions are used, simulations often fail due to this reason. This drawback makes the box-car kernel function not a desirable choice for robust and stable solvers. Summarizing the comparison presented in Fig. 5 and Table 4, the bell-shaped kernel functions have better robustness and accuracy than the box-car kernel functions in general, and are thus preferred. Another desirable feature for a robust solver is the mesh independence, which is investigated in Section 4.2 for both kernel functions.

4.2 Case 2: mesh-dependence study

In this example, a similar fluidized bed as in Case 1 is studied, except that the bed consists of two types of bi-dispersed particles and the inlet air velocity is constant (1m/s). The domain geometry as well as the initial and boundary conditions are shown in Fig. 3 and Table 3. To study the mesh-dependence, simulations are conducted with two meshes. The fine mesh has $12\times60\times1$ cells of size 3.33 mm $\times$3.33 mm $\times$7.5 mm, and the coarse mesh has $8\times40\times1$ cells of size 5 mm$\times$5 mm $\times$7.5 mm . The bandwidth for the kernel function is 10 mm for both the fine mesh and the coarse mesh.

The pressure drops are monitored in the same way as in the verification case above. The results obtained with the fine mesh and those with the coarse mesh are compared in Fig. 6(a). As a comparison, the simulation results obtained on the same meshes but with

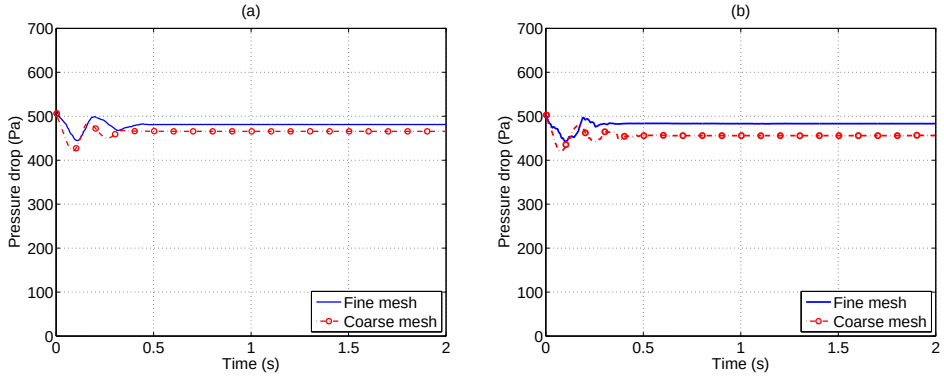

Figure 6: Mesh-dependence study (Case 2): Simulated pressure drop through the bed with (a) bell-shaped kernel function and (b) box-car kernel function (sum-up procedure). Results obtained with a fine mesh and those with a coarse mesh are compared.

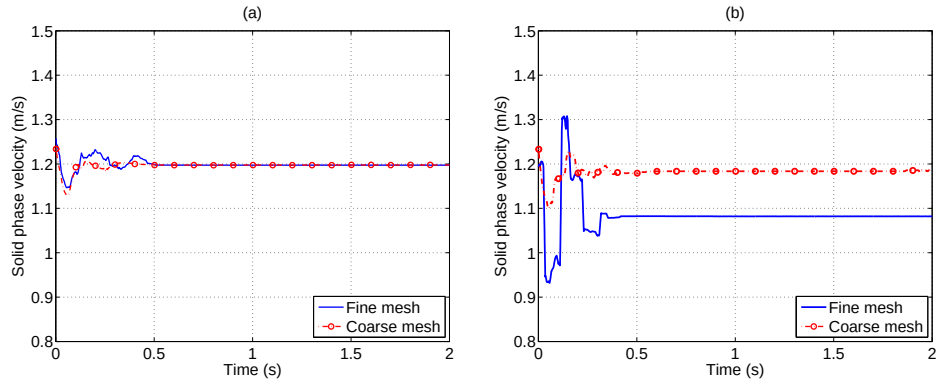

Figure 7: Mesh-dependence study (Case 2): Vertical component of the Eulerian mesh-based particle phase velocity at S1 (see Fig. 3) with (a) bell-shaped kernel function and (b) box-car kernel function (sum-up procedure). Results obtained with a fine mesh and those with a coarse mesh are compared.

box-car kernel function are shown in Fig. 6(b). The grid-convergence performance for the bell-shaped kernel function is better than that for the sum-up procedure, as shown by the differences between the fine mesh results and coarse mesh results. The Eulerian particle phase velocity, $\mathbf{u}_s$, at point S1 (refer to Fig. 3) is also monitored. The results obtained with bell-shaped kernel function and those from box-car kernel function are compared in the same way as for the pressure drop. Again, the fine mesh results and the coarse mesh results are compared for both kernel functions, and the differences between them indicate the performance of mesh-dependence. It can be seen that mesh-dependence of the particle phase velocity is much larger in the box-car kernel function results than that in the bell-shaped kernel function results. The difference in grid convergence is more significant for the particle phase velocity because the particle phase quantities are directly influenced by the averaging procedure, while the fluid field quantities such as the pressure are only indirectly influenced.

Another observation is that with bell-shaped kernel functions, finer mesh could be used to gain better accuracy for the flow field without causing the instability described in Section 3.1. In other words, we could have fewer particles in each cell. According to our experience, when using the sum-up procedure (box-car kernel function), instability often occurs and it is more difficult to obtain reasonable results. Considering that the DEM is a computationally intensive method, this advantage of the proposed averaging procedure is of practical significance when the available computational resources are limited compared to the demands of the problems.

4.3 Case 3: sub-stepping study

In this example, we consider a single particle in a water flow of constant velocity $U_e = 0.05\text{m/s}$. The domain geometry is shown in Fig. 3, with the same dimensions as in other cases but with different boundary and initial conditions so that uniform flow is achieved. The particle has an initial velocity of zero and is accelerated due to the fluid drag. For simplicity, other external forces such as gravity, buoyancy, and added mass are neglected in this case, and the box-car kernel function is used. The particle has a diameter of 0.083mm and the cell size in the mesh is $2\text{mm}\times 2\text{mm}\times 2\text{mm}$. The influence of the particle to the overall flow field is negligible.

The particle velocity is described by

$$\frac{\mathrm{d}u_{pi}}{\mathrm{d}t} = F_{\mathrm{d}i}, \tag{4.1}$$

where $F_{di} = \varepsilon_{\mathrm{s}} f_{di}$ is the drag force on the particle; f_{di} is a nonlinear function of u_{pi} as computed from Eq. (2.9), and $\varepsilon_{\mathrm{s}} = 0.524$ is constant. Both the force f_{di} and the velocity u_{pi} are scalars in this context since unidirectional flow is considered. If a constant force computed according the initial slip velocity is applied on the particle, the time it takes to accelerate the particle to the steam velocity is $\tau_p = 0.5\times 10^{-3}$ s in this case, where τ_p is the critical time step. The forward Eulerian time-stepping is used to solve Eq. (4.1) for the particle motion, and the results obtained with different time step sizes are presented in Fig. 8(a). The result obtained with MATLAB function ODE45, which uses a higher order method, is considered as the exact solution for reference.

It can be seen that when the time step size Δt is smaller than or equal to τ_p, the solution is stable. When Δt is between τ_p and $2\tau_p$, some oscillations occur but the particle approach the flow velocity eventually. If Δt is larger than $2\tau_p$, the solution diverges exponentially. Fig. 8(a) shows the results obtained from the numerical simulation with the hybrid solver, with fluid time step size $\Delta t_f = 1.1\times 10^{-3}$s. The particle movement in the hybrid solver is similar to solving Eq. (4.1) except that the force term F_{di} (a nonlinear function of u_{pi}) is linearized in the solver. The result obtained without sub-stepping is compared to those with 2 and 50 sub-steps per fluid step. It suggests that the sub-stepping effectively renders the algorithm stable. When 50 sub-steps are used, the results is almost identical to the exact solution (MATLAB ODE45 result in Fig. 8(b)).

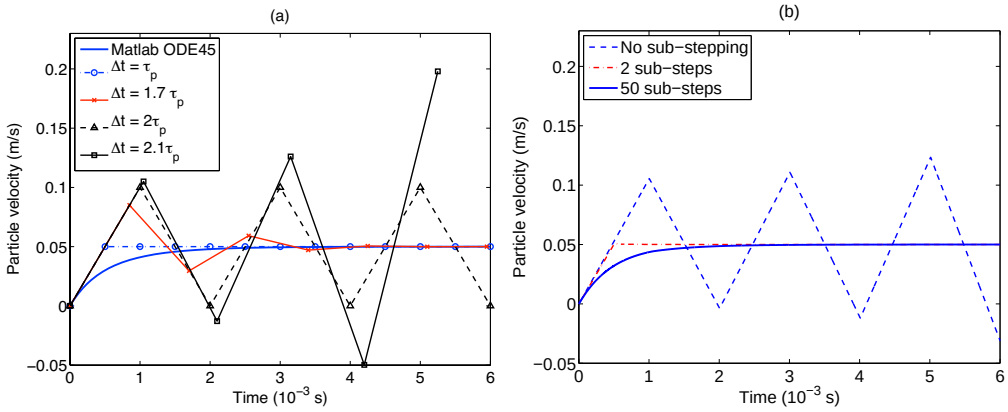

Figure 8: Sub-stepping study (Case 3): Particle velocity from (a) theoretical prediction with different time step sizes and from (b) numerical simulations using the hybrid solver with and without sub-stepping.

4.4 Case 4: groundwater seepage-induced particle transport

The transport of fine particles in porous media is of interest to researchers in various disciplines. For example, in geotechnical engineering, rapid groundwater flow caused by dewatering can carry away the fine sand from the soil, leading to mass loss and eventually the collapse of soil foundations. An example of seepage-induced transport is presented here to show the capabilities of the solver. The domain geometry as well as the initial and boundary conditions are shown in Fig. 3 and Table 3. The setup is similar to Cases 1 and 2, except that the interstitial fluid is water, with an inlet velocity of 3.5cm/s. The particle diameter has a Gaussian distribution with a mean of 1.25mm and a standard deviation of 0.5mm. The minimum and maximum particle diameters are 0.083mm and 2.5mm, respectively. The cumulative mass distribution curve shown in Fig. 9. For the convenience of visualization, the particles are categorized into three groups: small (0.083mm to 0.83mm), median (0.83mm to 1.67mm), and large (1.67mm to 2.5mm). The simulation is conducted for 4.1s, with particle time step size $\Delta t_p = 6\times 10^{-8}$s and 25 substeps per fluid step.

The snapshots at three time instances ($t = 0$s, 0.6s, and 4.1s) are shown in Fig. 10. Particles of different size groups are shown in different colors. The snapshot at $t = 0$ s in Fig. 10(a) shows the initial configuration where particles of all sizes are stationary and uniformly mixed. During the initial period of approximately 1s, the bed expands gradually and meanwhile the very small particles start to rise to the top. The snapshot at $t = 0.6$s is shown in Fig. 10(b) where many small particles are observed at the top surface of the bed. The segregation of median and large particles is not obvious from the snapshot. Afterward, some small particles start to be transported upward and leave the bed. Meanwhile, the size segregation among all particle groups continues. The snapshot at $t = 4.1$s in Fig. 10(c) shows the bed configuration after the segregation process, where smaller particles are concentrated at the top and the larger particles are settled at

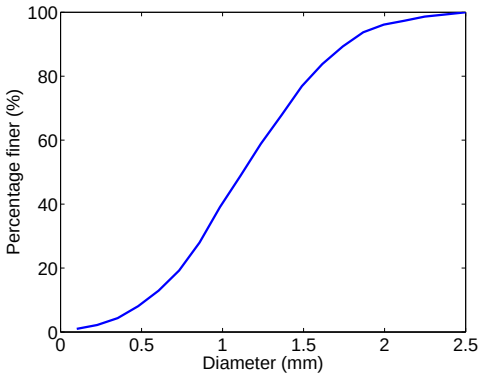

Figure 9: Cumulative distribution of particle diameters in Case 4.

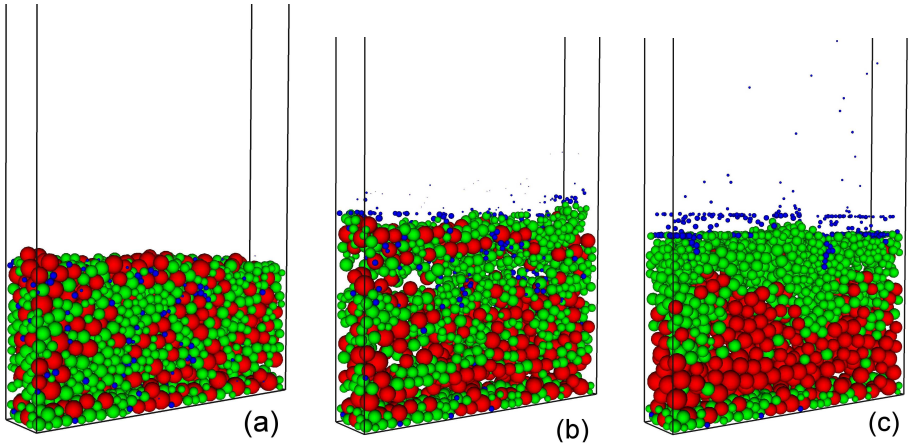

Figure 10: Snapshot at different time instances: (a) $t=0$s (initial particle state); (b) $t=0.6$s; and (c) $t=4.1$s. The box indicates the fluid domain. Only bottom half of the domain is shown. Different particle size groups (small, median, and large) are shown in different colors for the convenience of visualization.

the bottom. The segregation is obvious in this snapshot. To illustrate the segregation process quantitatively, the mean height of the three groups of particles (y_{small}, y_{medium}, and y_{large} for small, median, and large particles, respectively) are shown in Fig. 11(a). The ratio between the mean height of the small particles and that of the large particles ($y_{\text{small}}/y_{\text{large}}$) is shown in Fig. 11(b). The three groups of particles have the same mean height initially, confirming the uniform initial mixing. The bed expansion and particle segregation processes as explained above can be clearly seen in Fig. 11.

In this case, the particle time step size is very small because of the presence of the very fine particles (0.083mm minimum). The fluid step size is kept the same as in Cases 1 and 2 ($\Delta t_f = 10^{-3}$s). The sub-stepping is important for this case, otherwise either excessively small fluid step size is needed or the small particle would leave the domain with unphysically large velocity. This case also shows that the numerical model is able to capture the particle size segregation process and the transport of fine particles in a poly-

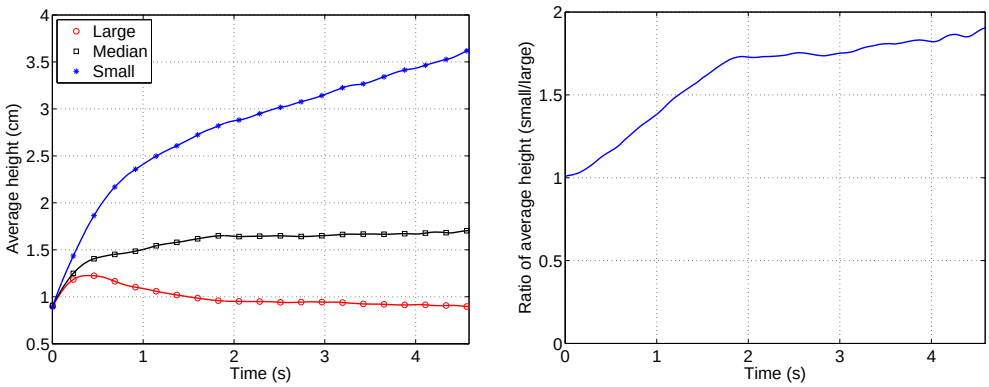

Figure 11: Quantitative description of transport and particle size segregation caused by upward seepage flows (Case 4). (a) Mean height of the three particle size groups (large, median, and small). (b) Ratio between the mean height of small particles and that of large particles.

dispersed bed. A major advantage of hybrid CFD-DEM models over two fluid models is their ability to simulate beds with continuous particle size distribution as in this case [4].

5 Conclusions

In this work, a robust hybrid CFD-DEM solver has been developed to simulate dense particle-laden flows with a wide distribution of particle sizes. A kernel-based interpolation procedure between Eulerian and Lagrangian fields suitable for 3D unstructured meshes is proposed. This procedure has been shown to be able to reduce the mesh-dependence of the interpolation processes and to increase the robustness of the solver. A semi-implicit treatment of the fluid-particle interaction terms in the fluid momentum equation is proposed, leading to improved stability performance over the explicit treatment. Finally, for flows with a wide range of Stokes numbers and thus multiple time scales, numerical examples suggest that excessively small step sizes for the fluid are needed. A momentum-conserving sub-stepping procedure is proposed to eliminate the stringent time step requirement for the fluid solver. Numerical simulations have been conducted to demonstrate the capabilities of the solver and the merits of the algorithm.

References

[1] Y. Liu, L. Zhang, X. D. Wang and W. K. Liu, Coupling of Navier-Stokes equations with protein molecular dynamics and its application to hemodynamics, Int. J. Numer. Meth. Fl., 46 (2004), 1237–1252.

[2] F. Osanloo, M. R. Kolahchi, S. McNamara and H. J. Herrmann, Sediment transport in the saltation regime, Phys. Rev. E., 78 (2008), 011301.

[3] M. Syamlal, W. Rogers and T. O'Brien, MFIX Documentation: Theory Guide, Technical report, National Energy Technology Laboratory, Department of Energy, 1993, See also URL `http://www.mfix.org`.

[4] J. Sun and F. Battaglia, Hydrodynamic modeling of particle rotation for segregation in bubbling gas-fluidized beds, Chem. Eng. Sci., 61 (5) (2006), 1470–1479.
[5] C. T. Crowe, T. R. Troutt and J. N. Chung, Numerical models for two-phase turbulent flows, Annu. Rev. Fluid. Mech., 28 (1) (1996), 11–43.
[6] P. A. Cundall and D. L. Strack, A discrete numerical model for granular assemblies, Géotechnique, 29 (1979), 47–65.
[7] Y. Tsuji, T. Kawaguchi and T Tanaka, Discrete particle simulation of two-dimensional fluidized bed, Powder. Technol., 77 (1993), 79–87.
[8] J. Sun, F. Battaglia and S. Subramaniam, Dynamics and structures of segregation in a dense, vibrating granular bed, Phys. Rev. E., 74 (6) (2006), 061307(13).
[9] Y. Q. Feng and A. B. Yu, Microdynamic modelling and analysis of the mixing and segregation of binary mixtures of particles in gas fluidization, Chem. Eng. Sci., 62 (1-2) (2007), 256–268.
[10] U. E. Shamy and M. Zeghal, Coupled continuum-discrete model for saturated granular soils, J. Engrg. Mech., 131 (4) (2005), 413–426.
[11] B. G. M. Van Wachem, J. C. Schouten, C. M. Van den Bleek, R. Krishna and J. L. Sinclair, CFD modelling of gas-fluidized beds with a bimodal particle mixture, AIChE. J., 47 (6) (2001), 1292–1301.
[12] K. D. Kafui, C. Thornton and M. J. Adams, Discrete particle-continuum fluid modelling of gas-solid fluidised beds, Chem. Eng. Sci., 57 (13) (2002), 2395–2410.
[13] S. Sundaram and L. R. Collins, Numerical considerations in simulating a turbulent suspension of finite-volume particles, J. Comput. Phys., 124 (1996), 337–350.
[14] L. E. Silbert, D. Ertas, G. S. Grest, T. C. Halsey, Levine D. and S. J. Plimpton, Granular flow down an inclined plane: bagnold scaling and rheology, Phys. Rev. E, 64 (5) (2001), 051302.
[15] T. B. Anderson and R. Jackson, A fluid mechanical description of fluidized beds: Equations of motion, Ind. Che. Engi. Fund., 6 (1967), 527–534.
[16] J. Sun, F. Battaglia and S. Subramaniam, Hybrid two-fluid DEM simulation of gas–solid fluidized beds, J. Fluid. Eng., 129 (2007), 1394–1403.
[17] C. Wen and Y. Yu, Mechanics of fluidization, Chem. Eng. Prog., Symp. Ser., 62 (1966), 100–111.
[18] J. Garside and M. R. Al-Dibouni, Velocity-voidage relationships for fluidization and sedimentation, Ind. Eng. Chem. Proc. Dd., 16 (1977), 206–214.
[19] J. Plimpton, Fast parallel algorithms for short-range molecular dynamics, J. Comp. Phys., 117 (1995), 1–19.
[20] H. Rusche, Computational Fluid Dynamics of Dispersed Two-Phase Flows at High Phase Fractions, PhD thesis, Imperial College London, UK, 2002.
[21] OpenCFD, OpenFOAM User Guide, 2008, See also `http://www.opencfd.co.uk/openfoam`.
[22] D. M. Snider, An incompressible three-dimensional multiphase particle-in-cell model for dense particle flows, J. Comput. Phys., 170 (2001), 523–549.
[23] R. A. Carmona, Statistical Analysis of Financial Data in S-Plus, Springer, 2004.
[24] J. Sun, H. Xiao and D. H. Gao, Numerical study of segregation using multiscale models, Int. J. Comput. Fluid. D., 23 (2) (2009), 81–92.
[25] H. P. Zhu, Z. Y. Zhou, R. Y. Yang and A. B. Yu, Discrete particle simulation of particulate systems: theoretical developments, Chem. Eng. Sci., 62 (2007), 3378–3396.
[26] C. T. Crowe, M. P. Sharma and D. E. Stock, The particle-source-in-cell (psi-cell) model for gas-droplet flow, J. Fluid. Eng., 99 (1977), 325–332.
[27] H. H. Hu, D. D. Joseph and M. J. Crochet, Direct simulation of fluid particle motions, Theor.

Comp. Fluid. Dyn., 3 (1992), 285–306.
[28] C. M. Rhie and W. L. Chow, A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation, AIAA., 21 (11) (1983), 1525–1532.
[29] R. I. Issa, Solution of the implicitly discretised fluid flow equations by operator-splitting, J. Comput. Phys., 62 (1986),40–65.
[30] H. Jasak, Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows, PhD thesis, Imperial College London, UK, 1996.
[31] S. Ergun, Fluid flow through packed columns, Chem. Eng. Prog., 43 (2) (1952), 226–231.
[32] M. Goldschmidt, Hydrodynamic Modelling of Fluidised Bed Spray Granulation, PhD thesis, Twente University, The Netherlands, 2001.