Commun. Comput. Phys.
doi: 10.4208/cicp.030509.141009a

Vol. **8**, No. 1, pp. 115-142
July 2010

A Lumped Particle Modeling Framework for Simulating Particle Transport in Fluids

Omar al-Khayat[1,*], Are Magnus Bruaset[1,2] and Hans Petter Langtangen[2,3]

[1] *Computational Geosciences, CBC, Simula Research Laboratory, P.O. Box 134, NO-1325 Lysaker, Norway.*
[2] *Department of Informatics, University of Oslo, P.O. Box 1080, Blindern, NO-0316 Oslo, Norway.*
[3] *Center for Biomedical Computing, Simula Research Laboratory, P.O. Box 134, NO-1325 Lysaker, Norway.*

Received 3 May 2009; Accepted (in revised version) 14 October 2009

Available online 12 February 2010

Abstract. This paper presents a lumped particle model for simulating a large number of particles. The lumped particle model is a flexible framework in modeling particle flows, embodying fundamental features that are intrinsic in particle laden flow, including advection, diffusion and dispersion. In this paper, the particles obey a simplified version of the Bassinet-Boussinesq-Oseen equation for a single spherical particle. However, instead of tracking the individual dynamics of each particle, a weighted spatial averaging procedure is used where the external forces are applied to a "lump" of particles, from which an average position and velocity is derived. The temporal evolution of the particles is computed by partitioning the lumped particle into smaller entities, which are then transported throughout the physical domain. These smaller entities recombine into new particle lumps at their target destinations. For particles prone to the effects of Brownian motion or similar phenomena, a symmetric spreading of the particles is included as well. Numerical experiments show that the lumped particle model reproduces the effects of Brownian diffusion and uniform particle transport by a fluid and gravity. The late time scale diffusive nature of particle motion is also reproduced.

PACS: 47.55.Kf, 47.57.Gc

Key words: Particle modeling framework, particle transport, lumped particle model.

1 Introduction

Consider sand and mud particles suspended in seawater, being moved around by the fluid flow before they eventually settle on the sea floor. This process is at the heart of

*Corresponding author. *Email addresses:* omark@simula.no (O. al-Khayat), arem@simula.no (A. M. Bruaset), hpl@simula.no (H. P. Langtangen)

computational studies of how sedimentary rocks are formed by erosion and deposition. In geoscience, such studies are referred to as depositional modeling [27]. When simulating the flow of particle-laden fluids, many physical aspects must be considered; the frequency of interparticle collisions, the ambient flow configuration including the degree of turbulence, and gravitational effects, to name a few. The size of the suspended particles is also an important factor, as particles of micrometers or less will have Brownian motion as well. For instance, all of the above mentioned effects must be taken into account when studying *turbidity currents* [41], which take the form of a highly turbulent sand-laden subaquatic flow. Turbidity currents are often triggered by tsunamis, earthquakes, or underwater avalanches, and contribute significantly to the transport of sediments into deep marine areas [25].

The accurate simulation of particle transport and the correct modeling of turbulence are two central aspects in understanding turbidity currents. This paper will focus on the simulation of the particles in the flow.

Although there is great variation in the details, the simulation of particle flows can roughly be categorized into two distinct approaches: *discrete particle models* and *Eulerian continuum models*. In discrete particle models, the motion of either single particles or clusters of particles are simulated individually by applying forces as prescribed by Newton's laws. The Eulerian models, however, treat the particles as a continuum where averaged equations of motions are solved instead.

The most common of the discrete particle methods is the Lagrangian approach [23]. Here, each particle's position and velocity are obtained by integrating an equation describing the particle's motion, which is usually the Bassinet-Boussinesq-Oseen (BBO) equation [36]. This equation is coupled with variants of the Navier-Stokes equation to obtain a description of the particle flow [30]. Usually, this coupling is one-way, meaning that the particles do not influence the ambient fluid. However, two-way couplings are often needed for dense particle flows, for which the particles' effect on the fluid is modeled by either including body force terms in the Navier-Stokes equations or adding a particle density dependent fluid viscosity. The Lagrangian approach has also the advantage of being applicable to problems covering a wide range of Reynolds numbers [24, 38]. More details on Lagrangian models are available in [5] and the references therein.

For low Reynolds numbers, a widely used discrete particle method is the *Stokesian dynamics* approach [28]. Here, the linearized hydrodynamic equations are solved for the particles and the fluid [18], where both the rotational and translational aspects are studied. Forces and torques on the particles are obtained by integrating over the particles' surface, the net effect of which is then applied to the surrounding fluid as well. Stokesian dynamics have been very successful in reproducing observed results, such as Brownian motion, in addition to correctly predicting values for drag coefficients However, this method is computationally very expensive for dynamically evolving systems, since the computational effort scales as the cube of the number of particles [19]. Currently, simulations based on this method is limited to 100 or less. Stokesian dynamics is, however, a useful tool to study basic particle physics at low Reynolds numbers.

A similar approach to the Stokesian one is the use of the moving boundary approach [18] for the *lattice Boltzmann method* for fluid flow [26]. Here, the fluid is considered as consisting of fictive particles subject to consecutive propagation and collision processes over a regular lattice. Macroscopic fluid quantities like velocity and density are computed from statistical moments of the fictitious particles' lattice variables that are obtained from the *lattice Boltzmann equation* [39]. In the moving boundary approach, particles are usually treated as a set of solid lattice points absent of the fictitious fluid particles. The fluid interacts with the particles by simple collision rules that exchange momentum between the fluid and the particles. Solid sites are filled with fluid as the particles are moved through the domain. Simulating a fluid with a high Reynolds number is possible as well. For instance, recent developments have shown that the lattice Boltzmann method can simulate fluid flows for Reynolds numbers up to 20000 [4]. However, these simulations has yet to be tested with suspended solid particles.

A certain class of discrete particle methods use computational entities to represent a number of real particles. Notable examples include the *smoothed dissipative particle dynamics* models [10, 14], *stochastic rotational dynamics* models [16], and the *Lagrangian-Eulerian* methods [12]. A description of these methods goes well beyond the scope of this paper, and more details concerning this class of methods can be found in [17]. Most of these approaches stem from molecular dynamics [2], and share the inclusion of a stochastic element in the modeling of the particle dynamics. Variants of these methods have been applied to particle flows with low Reynolds numbers, and show good agreement with experiments [22]. However, these approaches have yet to be applied to flows with high Reynolds numbers. As is common with most discrete particle methods, the computational effort scales as the square of the particle number N. Although certain state-of-the-art algorithms reduce this scaling to $\mathcal{O}(N\log N)$ [17].

In many geological applications the number of particles is huge; of the order $10^8 - 10^{12}$. Hence, tracking the movement of each particle is not feasible with currently available computing power, nor can one expect to see realistic simulations of millions of particles in the foreseeable future. However, in many instances it may be argued that large amounts of particles can equally well be modeled as a continuum.

In the continuum approach, often known as Eulerian *two-fluid* or *multiphase* computational fluid dynamics (CFD), both the fluid and the particles are treated as interacting continua [6]. Usually, the Newtonian equations for the particles undergo a volume-averaging procedure that results in continuum equations [3]. A set of equations for mass and momentum conservation for each phase are then obtained. The momentum conservation equations include a term that signifies the momentum transfer between the phases, which is required to close the set of equations. These *closure equations* are often complex, and necessitate amongst other things the correct specification of *constitutive relations*, like the solid phase stress. Constitutive relations for two-phase flows are in general empirical and sometimes lack experimental validation for the conditions they are valid under [40]. However, recent advances have applied concepts from kinetic theory [13] to obtain the particle phase stress relations from net streaming and collisions of parti-

cles [21]. In the derivations, the particle collisions are assumed to be binary and instantaneous, which may not be applicable for dense flows [42]. The kinetic theory approach has as yet only been applied to gas-particle flows and show good qualitative results with experiments [9]. For a detailed review of multiphase CFD, see [40].

Depending on the physical setting studied, the solid and fluid phase equations can sometimes be simplified to variants of the advection-diffusion equation. This approximation is often used in geological settings [15], and does in some cases simplify the numerical schemes employed. Problems arise, however, in the correct estimation of model parameters requiring a great deal of fine tuning to reproduce experimental observations [33]. The resolution of boundary conditions can also pose a problem in multiphase CFD. For instance, at a wall, the usual value for the fluid velocity would be zero. This is generally not the case for particles. Various attempts have been proposed to remedy this problem, see [5] for further discussion of boundary conditions.

As mentioned earlier, many physical effects must be included in the simulation of turbidity currents. However, the continuum approaches have yet to capture all of the physical properties that are characteristic of these systems [41]. One reason for this is clearly the lack of good constitutive relations in the closure assumptions. Moreover, it is clear that the particle nature of these flows are important for turbidity currents, but discrete particle methods are still computationally expensive. Furthermore, it is also unclear to what extent the various physical processes effect the evolution of the particles [20]. We believe that numerical experiments are a way to explore these relations. Hence, to address these problems, we propose a mesoscopic hybrid continuum-particle approach in which we simulate particle transport by introducing a *lumped particle model*.

The lumped particle model described in this paper can be viewed as a flexible framework in modeling transport of particles in fluid flow. Our goal is to define an efficient and simple algorithm for the evolution of particles in space and time. In the proposed model, we include the fundamental features that are intrinsic in particle flow: advection, diffusion and dispersion of the particles. These serve to model Brownian motion and other hydrodynamic forces. Additional physical effects can readily be included into the modeling framework. However, in the lumped particle model, a certain number of particles are treated as a single entity. Instead of tracking the individual dynamics of each particle, a weighted spatial averaging procedure is used. The external forces are applied to the lump of particles, from which an average position and velocity is derived. Hence, the particles are in a sense considered as a continuum, but where the particle nature heavily influence the dynamics. As will be illuminated below, when computing the evolution of the particles, the particle lumps are partitioned into smaller entities which are then transported according to local physical effects. These smaller entities recombine into new particle lumps at the target destinations. The lumped particle model introduced here alleviates the need to track the evolution of each particle, while simultaneously keeping the granular nature of the particles.

2 Algorithmic overview

Imagine a set of identical, spherical particles suspended in a fluid. Each individual particle has a number of physical properties, such as its velocity and its position relative to a given reference frame. Our aim is to calculate the evolution of these quantities over time as external forces are applied. Interparticle collisions are not considered in this paper. Such collisions become important only when the particle concentration in the fluid becomes high, usually 30% or higher. However, this value depends in general on the flow configuration [37].

The particle lumping procedure is as follows: Consider a discretization of the spatial area where both particles and fluid reside. In this paper, we will restrict our attention to a two-dimensional regular lattice, but most of what is presented can also be used with other grid types and in three dimensions. Hence, we partition the computational domain into a regular lattice, with physical spacing parameters Δx and Δy. Time is discretized into increments of Δt. As shown in Fig. 1, the lattice defines a set of grid cells, each having a center point $\mathbf{x}_p$. The distribution $N(\mathbf{x}_p,t)$, and the velocity $\mathbf{V}(\mathbf{x}_p,t)$ are defined as the number of particles inside the cell and the average velocity of these particles, respectively. The proposed algorithm calculates how these variables change during a time increment Δt.

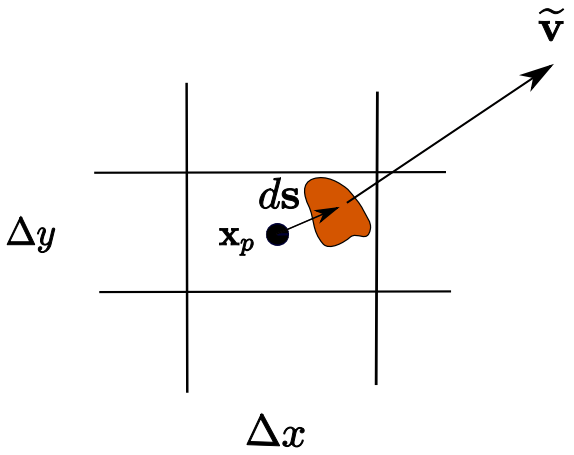


Figure 1: The lumped representation of particles. Particles within a grid cell are treated as a single entity. Here, $\mathbf{V}$ is the average velocity and $d\mathbf{s}$ is the offset from the particle centroid to the cell center.

The computation of the particle lump's dynamical properties consists of three distinct steps which replace the conventional full particle-tracking approach. The first step is a *dispersion* step, where the particle lump is split into smaller parts, called *quasi-particles*. Each of these quasi-particles have a *dispersion velocity* $\mathbf{c}_i$, which quantifies the direction of the quasi-particle movement. It is in this step that the kinematics of these quasi-particles is calculated. The second step is the *recombination* step, where quasi-particles are recombined to form a new particle lump at the destination sites. The third step is a *diffusion*

step, which enables the modeling of Brownian motion and similar phenomena. An short account of these steps will be given below.

Dispersion step

In this step, we compute the force acting on the particle lump in each grid cell. Using the average velocity as a basis, an acceleration is calculated numerically from Newton's second law applied to the particle lump. This acceleration is then applied to the quasi-particles within a grid cell, thereby changing their dispersion velocities $\mathbf{c}_i$. As a result, the quasi-particles are transported to their target cells, corresponding to traveling with their respective velocities in the given time increment Δt, see Fig. 2.

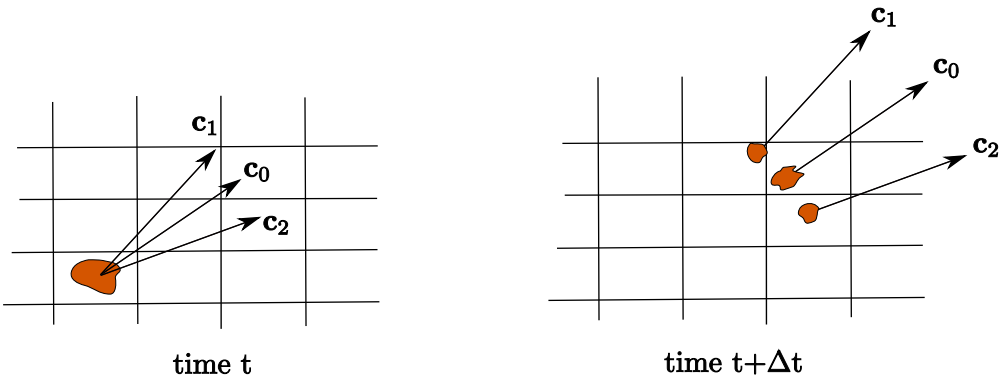


Figure 2: Dispersion of the particle lump. The particle lump is split up into smaller parts, known as quasi-particles. These are transported according to the dispersion velocities, represented by the vectors $\mathbf{c}_0$, $\mathbf{c}_1$ and $\mathbf{c}_2$.

However, the distance the quasi-particle travels does not usually correspond to an integer multiple of one grid cell. Such a general displacement would lead to an error in the computation of the particles' position. As a means to track this error, an *error correction* vector $d\mathbf{s}(\mathbf{x}_p)$ is introduced. This vector field quantifies the offset of the centroid of the particle lump relative to $\mathbf{x}_p$, and can serve as a measure of positional error. Each quasi-particle's displacement is modified by the error correction, possibly choosing a different grid cell to become the destination. Once the quasi-particles reach their destinations, new error measures are calculated.

Recombination step

As shown in Fig. 3, the recombination step causes the creation of a new lumped particle from the incoming quasi-particles. A new mean velocity $\mathbf{V}$ and error measure $d\mathbf{s}$ are calculated as the averages of the respective velocities and error measures of the quasi-particles. To compensate for the possibly varying velocities of the quasi-particles, a set of dispersion velocities $\mathbf{c}_i$ are found. These are derived from a *momentum balance* calculation, where each direction is weighted according to its fraction of the total momentum within

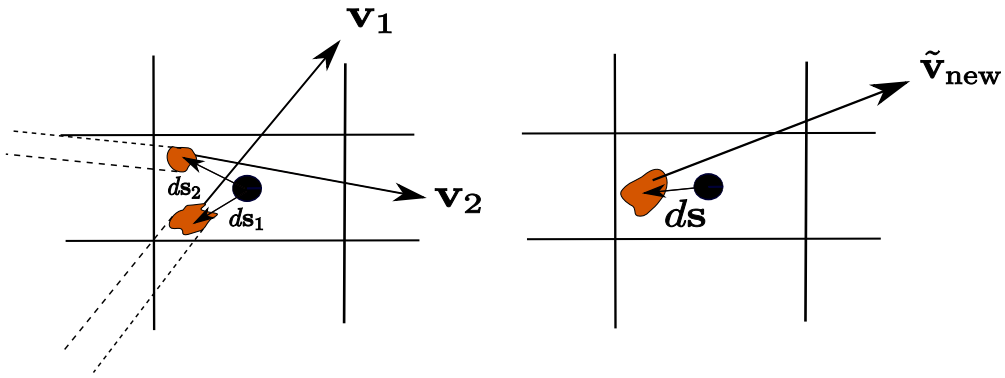


Figure 3: Quasi-particles recombine into a particle lump. Two quasi-particles with different velocities enter a grid cell. Their respective velocities $\mathbf{v}_1$ and $\mathbf{v}_2$, and error measures $d\mathbf{s}_1$ and $d\mathbf{s}_2$ are averaged. This averaging gives a velocity $\mathbf{V}_{\text{new}}$ and error measure $d\mathbf{s}$ of the new particle lump.

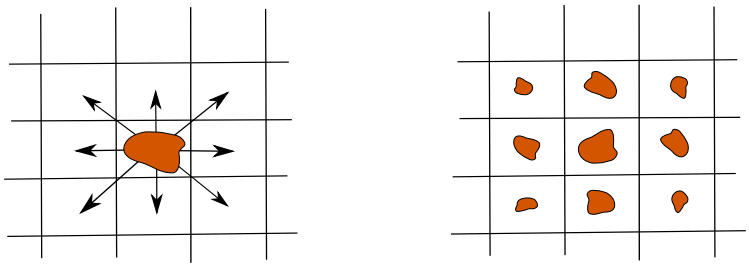


Figure 4: The diffusion procedure. Particles in each cell are symmetrically redistributed, to reproduce diffusion as described by the continuum advection-diffusion equation.

the grid cell. That is, in preparation for the next dispersion step, these weights determine the number of particles each quasi-particle will consist of.

Diffusion step

The two preceding steps are sufficient to model non-Brownian motion of particles. To be able to reproduce Brownian motion and similar phenomena, a diffusion phase is included in our model. This step is very similar to the other two phases, but with some modifications. Brownian motion in its simplest form, is characterized by the random movement of microscopic particles suspended in a fluid, caused by elastic collisions with fluid molecules. As the number of particles increases to infinity, Brownian movement can be approximated to good accuracy by homogeneous diffusion in a continuum sense [8]. If Brownian motion is relevant for the case in question, a new dispersion and recombination step is added. Here, the particle lumps are partitioned into 9 quasi-particles for two spatial dimensions, and 15 for three dimensions. And, as shown in Fig. 4, these quasi-particles are given spherically symmetric dispersion directions, although other symmetries can be applied as well. Additionally, the dispersion weights are chosen from a probability distribution with a similar symmetry. Hence, the diffusion phase is just a symmet-

ric redistribution of the particle lumps. It will be shown in Section 6 that our approach is equivalent to the continuum advection-diffusion equation under certain circumstances.

3 The Bassinet-Boussinesq-Oseen equation for a single spherical particle

We start by considering the individual particles that constitute the particle lump. These are assumed to be small spherical particles suspended in a fluid. The Bassinet-Boussinesq-Oseen (BBO) equation describes the temporal evolution of a spherical particle in an unsteady flow [36], which is a form of Newton's second law of motion. One version of this equation is given by

$$m_p \frac{\mathrm{d}\mathbf{v}_p}{\mathrm{d}t} = -\frac{\pi \mu d_p \mathrm{Re}_p}{8} C_D \left(\mathbf{v}_p - \mathbf{u}\right) + \left(1 - \frac{\varrho_f}{\varrho_p}\right) m_p \mathbf{g}. \tag{3.1}$$

By solving the above equation, the particle position $\mathbf{x}$ can be found by solving

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v}_p. \tag{3.2}$$

In Eq. (3.1), $\mathbf{v}_p$ and $\mathbf{u}$ are the particle and fluid velocities respectively, while d_p is the particle diameter, m_p is the particle mass and $\mathbf{g}$ is the gravitational acceleration. Furthermore, ϱ_f and ϱ_p are the fluid and particle densities respectively, and μ is the dynamic fluid viscosity.

The left hand side of Eq. (3.1) is the well known particle acceleration term. Moreover, the first term on the right hand side is the Stokes drag on the particle, which tends to drive the particle velocity towards the fluid velocity. The second term represents the combined effect of gravity and buoyancy. Observe that in this paper we have neglected a few physical effects, like the virtual mass force [36] and rotational forces. Observe that it is usual in the literature to reserve the tern BBO for when these terms are included. We will, however, for simplicity use the term "BBO-like" in this paper. The omitted terms can be important in some applications [35], but are not relevant for the particle sizes considered in this paper.

The parameter C_D is called the *drag coefficient* and Re_p is the *particle Reynolds number* defined as

$$\mathrm{Re}_p = \frac{\varrho_f |\mathbf{v}_p - \mathbf{u}| d_p}{\mu}. \tag{3.3}$$

For low particle Reynolds number, that is $\mathrm{Re}_p < 1000$, the Stokes drag can be approximated by

$$C_D \simeq \frac{24}{\mathrm{Re}_p}(1 + 0.15\mathrm{Re}_p^{0.687}). \tag{3.4}$$

With this approximation, Eq. (3.1) can be written as

$$\frac{\mathrm{d}\mathbf{v}_p}{\mathrm{d}t} = -\frac{1}{\tau_p}\left(\mathbf{v}_p - \mathbf{u}\right) + \left(1 - \frac{\varrho_f}{\varrho_p}\right)\mathbf{g}. \tag{3.5}$$

Here, we have defined

$$\tau_p = \frac{m_p}{3\pi d_p \mu}\frac{1}{1+0.15\mathrm{Re}_p^{0.687}},$$

where the particle *relaxation time* τ_p is a measure of the particle's response to a changing fluid velocity. We will assume that the particle Reynolds number is of the order $\sim 10^{-4}$, which is valid for many applications. That is, we will only consider a constant particle relaxation time

$$\tau_p \simeq \frac{m_p}{3\pi d_p \mu}. \tag{3.6}$$

4 The dispersion step

Consider now a group of small spherical particles embedded in a fluid or gas, obeying the BBO Eq. (3.5). These particles are assumed to have the same average size and shape. Recall from Section 2 that $N(\mathbf{x}_p,t)$ is the number of particles within a grid cell with center point $\mathbf{x}_p$. Additionally, we defined $\mathbf{V}(\mathbf{x}_p,t)$ to be the average particle velocity of the particles in that grid cell. In dense flows, the presence of many particles will change the effective drag coefficient C_d for each particle [11]. Since we are in this paper only considering dilute flows, we will assume that the BBO-equation remain unchanged when applied to each particle in the particle lump. In principle, this means that the BBO-equation for the lumped particle is

$$\sum_k \frac{\mathrm{d}\mathbf{v}_{p,k}}{\mathrm{d}t} = -\frac{1}{\tau_p}\sum_k\left(\mathbf{v}_{p,k} - \mathbf{u}\right) + k\left(1 - \frac{\varrho_f}{\varrho_p}\right)\mathbf{g}, \tag{4.1}$$

where k is the number of particles in a given grid cell.

In this section, we will describe the method for transporting the particle lump. Recall from Section 2 that the particle lump is divided into smaller entities, *quasi-particles*, with a specific dispersion velocity, shown in Fig. 2. Exactly how the lump is partitioned into quasi-particles will be detailed in Section 5. The BBO Eq. (4.1) is applied to each particle the particle lump, and a new average acceleration is calculated. This acceleration is then applied to the quasi-particles.

4.1 A numerical scheme for the BBO equation

The particle evolution is computed for a sequence of time steps with uniform spacing Δt. We will need to compute the acceleration for the particle lump, which we then can apply to the quasi-particles. The computation requires a discrete solution of the BBO Eq. (3.5),

which is obtained by an explicit forward Euler method in time. Let $t^\ell = \ell\Delta t$ with ℓ being the time step index. A temporal discretization of Eq. (3.5) becomes

$$\frac{\Delta \mathbf{v}^\ell}{\Delta t} = -\frac{1}{\tau_p}\left(\mathbf{V}^{\ell-1} - \mathbf{u}^{\ell-1}\right) + \left(1 - \frac{\varrho_f}{\varrho_p}\right)\mathbf{g}, \tag{4.2}$$

where $\Delta \mathbf{v}^\ell / \Delta t$ is the acceleration of each individual particle in the current grid cell. Hence, we will apply this acceleration to the quasi-particles. The average velocity $\mathbf{V}^{\ell-1}$ is used as the basis for the force calculation, modeling the overall drift of the particle lump. Whereas the fluid velocity $\mathbf{u}^{\ell-1}$ is assumed to be known, either analytically or as a numerical approximation generated by a separate solver for fluid flow.

The displacement $\Delta \mathbf{x}^\ell$ of the quasi-particles will depend on the dispersion velocities. Since the k'th quasi-particle attains the new velocity $\mathbf{c}_k + \Delta \mathbf{v}^\ell$, we can use the Trapezoidal rule to integrate Eq. (3.2). This integration gives

$$\Delta \mathbf{x}_k^\ell = \int_{t^{l-1}}^{t^l} \mathbf{v}_p \mathrm{d}t = \left(\mathbf{c}_k + \frac{\Delta \mathbf{v}^\ell}{2}\right)\Delta t. \tag{4.3}$$

Note that Eq. (3.1) can also be solved using an implicit scheme such as the Newmark method. This is especially important when the particle relaxation time becomes small compared to the fluid time scale.

4.2 The quasi-particle displacement algorithm

In order to update the position of the quasi-particles at each time step, we need a procedure to calculate both the new position in the grid and the error correction d$\mathbf{s}$. Recall that the error correction is designed to keep track of the average position of the lumped particle within a grid cell. The calculation will be motivated by studying the movement of *one* particle in *one* spatial dimension. In this case the error correction d$\mathbf{s}$ becomes a scalar ds and reduces to the particles' relative position to the grid cell center. Hence, the algorithm can be viewed as a mapping from the particle position in space to the grid position. Once the core algorithm is derived, generalization to higher dimensions will be straightforward. The many-particle extension of the procedure will also be accounted for.

Consider one particle at some position along the x-axis. To be able to measure the position of this particle on the grid, a set of grid cells I_j equal in size Δx is defined. The particle position can be unambiguously quantified by the offset ds from the cell center x_j, as shown in Fig. 5. In general, the particle's physical position will not coincide with a grid center. Therefore, the offset ds can be viewed as a measure for the positional error, induced by using x_j as the particle's grid position. Using this convention, our goal is to find the particle's new location at the next time step.

During the current time step, we assume that the particle will travel a distance s, a value obtained from Eq. (4.3). This movement would correspond to a displacement of

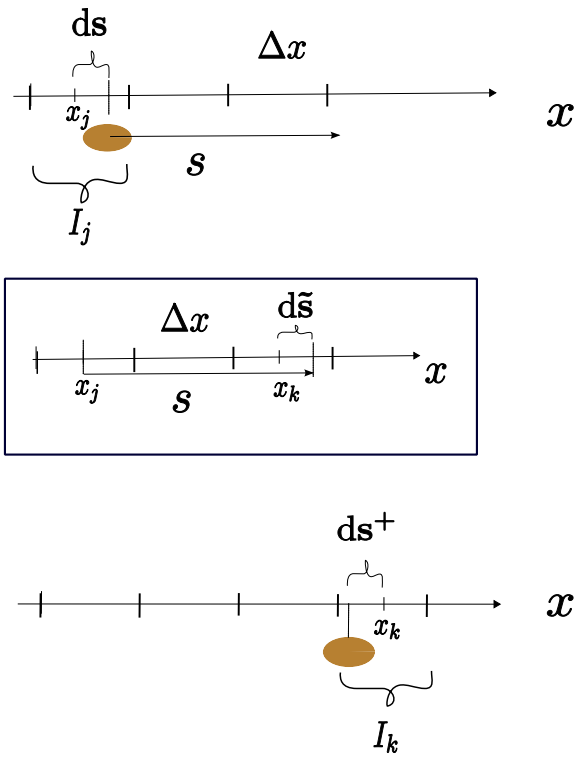


Figure 5: The displacement algorithm of a single particle.

$s/\Delta x$ units on the grid. At first glance, one would then assume that the target grid cell I_k at the next time step would be given by $k=j+[s/\Delta x]$, where $[\cdot]$ denotes the closest integer value. But this would be wrong because $s/\Delta x$ is rarely an integer, and the particle's initial position is not exactly x_j. Indeed, if this method is used at each time step, the inferred error would grow in time. We therefore propose an *error correction* as a remedy for this problem.

The basic idea of the error correction is to compensate for the error in both the dynamics and the position when moving the particle. That is, first we calculate the number of units that the particle has traveled $[s/\Delta x]$. Then we calculate the *dynamical error* $\mathrm{d}\tilde{s}$, which is the error in the displacement if the particle would have been placed at the grid cell center x_j, Moreover, we adjust the particle grid cell I_k according to the errors in the position and displacement. Finally, the new position offset $\mathrm{d}s^+$ for the next time step can be estimated.

As shown in the middle box of Fig. 5, the difference between the particle displacement and the number of grid cells transversed is given by $\mathrm{d}\tilde{s}_x = s-[s/\Delta x]\Delta x$. From this value, we calculate the new target grid cell index $k=j+[s/\Delta x]+[(\mathrm{d}s+\mathrm{d}\tilde{s})/\Delta x]$, where the dynamical and positional error has been included. Finally, to complete the procedure, the new offset value $\mathrm{d}s^+$ is calculated by $\mathrm{d}s^+ = \mathrm{d}s+\mathrm{d}\tilde{s}-[\mathrm{d}s+\mathrm{d}\tilde{s}/\Delta x]\cdot\Delta x$.

As mentioned earlier, the error correction is easily generalized to three dimensions. Let $I_{\mathbf{j}}$ be the initial grid cell and $I_{\mathbf{k}}$ be the target grid cell, where $\mathbf{k}$ and $\mathbf{j}$ are grid cell index vectors. If $\mathrm{d}s_i$ and k_i denotes the i'th component of $\mathrm{d}\mathbf{s}$ and the grid cell index vector respectively, then these three are updated at each time step by

$$\mathrm{d}\tilde{s}_i = s_i - \left[\frac{s_i}{\Delta x_i}\right]\Delta x_i\,, \tag{4.4}$$

$$k_i = j_i + \left[\frac{s_i}{\Delta x_i}\right] + \left[\frac{\mathrm{d}s_i + \mathrm{d}\tilde{s}_i}{\Delta x_i}\right], \tag{4.5}$$

$$\mathrm{d}s_i^+ = \mathrm{d}s_i + \mathrm{d}\tilde{s}_i - \left[\frac{\mathrm{d}s_i + \mathrm{d}\tilde{s}_i}{\Delta x_i}\right]\Delta x_i\,. \tag{4.6}$$

Here, $\mathrm{d}\tilde{s}_i$ is i'th component of the dynamical error, s_i is the displaced distance in the i'th grid direction, and Δx_i is the grid coarseness in the i'th direction.

In the many-particle case, the interpretation of $\mathrm{d}\mathbf{s}$ returns to the offset between the grid cell center and the centroid position of the lumped particle. The update rule for the position and the centroid $\mathrm{d}\mathbf{s}$ will as in the single-particle case be Eqs. (4.4)-(4.6). But now, it is applied on all the quasi-particles instead.

5 Recombination of quasi-particles

Like the particle lump is split as described in Section 4, quasi-particles *recombine* into a new particle lump, as illustrated in Fig. 3. Hence, the velocity and position of the particle lump will be averaged values based on the properties of the incoming quasi-particles. Let us examine one grid cell where several quasi-particles have entered. The i'th quasi-particle consists of N^i particles and has a velocity $\mathbf{v}_i$. There are many different ways of calculating averages, but we will choose the average velocity to be the *root mean square* (rms) of the quasi-particles' velocities. Assuming that there are m quasi-particles entering the grid cell, the magnitude of the average velocity for the next time step becomes

$$\tilde{v}^+ = \sqrt{\frac{N^1\mathbf{v}_1^2 + N^2\mathbf{v}_2^2 + N^3\mathbf{v}_3^2 + \cdots + N^m\mathbf{v}_m^2}{\sum N^i}}. \tag{5.1}$$

To find the direction, we first calculate the weighted average velocity vector

$$\hat{\mathbf{v}} = \frac{N^1\mathbf{v}_1 + N^2\mathbf{v}_2 + N^3\mathbf{v}_3 + \cdots + N^m\mathbf{v}_m}{\sum N^i}, \tag{5.2}$$

which implies the new velocity

$$\mathbf{V} = \tilde{v}^+\frac{\hat{\mathbf{v}}}{|\hat{\mathbf{v}}|}. \tag{5.3}$$

This averaging approach has the distinct advantage of conserving the total kinetic energy of the particles. That is, the kinetic energy of the lumped particle is the direct sum of the

individual particles' energy. As for many dense particle flows, dissipation effects can be added to our model a posteriori.

The position of the lump is quantified by the offset from the grid cell center to the centroid of the particle lump. The offset $\mathrm{d}\mathbf{s}$ is calculated as the weighted average of the incoming quasi-particles' offsets

$$\mathrm{d}\mathbf{s} = \frac{N^1\mathrm{d}\mathbf{s}_1 + N^2\mathrm{d}\mathbf{s}_2 + N^3\mathrm{d}\mathbf{s}_3 + \cdots + N^m\mathrm{d}\mathbf{s}_m}{\sum N^i}, \tag{5.4}$$

Recall that each $\mathrm{d}\mathbf{s}_i$ is calculated from Eqs. (4.4)-(4.6). Moreover, the number of particles constituting the new lump is $N(\mathbf{x}_p, t+\Delta t) = \sum N^i$.

5.1 Dispersion directions

To reach a complete model, we also need to address the effects due to the varying velocities of the lumps' quasi-particles. The use of *dispersion velocities* $\mathbf{c}_i$ can be one way to compensate for the lumped treatment of the particles. The basic idea is to assign certain velocities along which new quasi-particles of the lump are transported. We will now describe a procedure for calculating the number of particles that will move in each different direction. This approach serves to simulate the mixing of the particles in the fluid.

For simplicity, we continue the discussion in the context of two spatial dimensions. The generalization of the procedure to three dimensions is straightforward. To quantify the general spread of particle velocities within the grid cell, we define the *standard deviation* $\boldsymbol{\sigma} = (\sigma_x, \sigma_y)$ from the mean value $\hat{\mathbf{v}} = (\hat{v}_x, \hat{v}_y)$ as defined by Eq. (5.2). Recall that $\mathbf{v}_i = (v_x^i, v_y^i)$ is the velocity of the i'th quasi-particle. The standard deviation is then

$$\sigma_x = \sqrt{\frac{N^1(\hat{v}_x - v_x^1)^2 + N^2(\hat{v}_x - v_x^2)^2 + N^3(\hat{v}_x - v_x^3)^2 + \cdots + N^m(\hat{v}_x - v_x^m)^2}{\sum N^i}}, \tag{5.5}$$

$$\sigma_y = \sqrt{\frac{N^1(\hat{v}_y - v_y^1)^2 + N^2(\hat{v}_y - v_y^2)^2 + N^3(\hat{v}_y - v_y^3)^2 + \cdots + N^m(\hat{v}_y - v_y^m)^2}{\sum N^i}}. \tag{5.6}$$

We will restrict the number of dispersion velocities for the two-dimensional case to three, even though higher numbers are possible. The three most natural choices of directions are

$$\tilde{\mathbf{c}}_0 = \hat{\mathbf{v}}, \tag{5.7}$$

$$\tilde{\mathbf{c}}_1 = \hat{\mathbf{v}} + \boldsymbol{\sigma}, \tag{5.8}$$

$$\tilde{\mathbf{c}}_2 = \hat{\mathbf{v}} - \boldsymbol{\sigma}. \tag{5.9}$$

To calculate how many particles that are dispersed in the respective directions, we use a momentum balance approach in which the above vectors with large magnitudes are weighted higher than the ones of smaller magnitude. The *fractional number distribution* q_i

quantifies the fraction of particles at each grid cell that are transported in direction $\mathbf{c}_i$. So, q_i is given by

$$q_i = \frac{\tilde{\mathbf{c}}_i^2}{\sum_j \tilde{\mathbf{c}}_j^2}. \tag{5.10}$$

Furthermore, the magnitude of the dispersion directions must be adjusted to conserve the total kinetic energy within the grid cell. This property is guaranteed by setting the magnitude equal to the rms velocity $\mathbf{V}$, which defines the dispersion vectors

$$\mathbf{c}_0 = v^+ \frac{\hat{\mathbf{v}}}{|\hat{\mathbf{v}}|}, \tag{5.11}$$

$$\mathbf{c}_1 = v^+ \frac{\hat{\mathbf{v}}+\boldsymbol{\sigma}}{|\hat{\mathbf{v}}+\boldsymbol{\sigma}|}, \tag{5.12}$$

$$\mathbf{c}_2 = v^+ \frac{\hat{\mathbf{v}}-\boldsymbol{\sigma}}{|\hat{\mathbf{v}}-\boldsymbol{\sigma}|}. \tag{5.13}$$

Here, v^+ is given by Eq. (5.1). In three spatial dimensions, two additional vectors are needed,

$$\mathbf{c}_3 = v^+ \frac{\hat{\mathbf{v}}+(\boldsymbol{\sigma}\times\hat{\mathbf{v}})}{|\hat{\mathbf{v}}+(\boldsymbol{\sigma}\times\bar{\mathbf{v}})|}, \tag{5.14}$$

$$\mathbf{c}_4 = v^+ \frac{\hat{\mathbf{v}}-(\boldsymbol{\sigma}\times\hat{\mathbf{v}})}{|\hat{\mathbf{v}}-(\boldsymbol{\sigma}\times\hat{\mathbf{v}})|}. \tag{5.15}$$

Five dispersion velocities are the minimum set of vectors needed to simulate the dispersion of the particles in three dimensions.

6 Diffusion

In this section, we present a detailed account of the diffusion procedure. Diffusion is a frequently occurring phenomenon in Nature, and is of great importance to particle-laden fluid flow. Particle diffusion can have different physical causes, but can usually be categorized into two distinct classes. *Brownian diffusion* is characterized by elastic, random collision of the particles with the water molecules. The other type of particle diffusion is non-Brownian in nature and is often called *hydrodynamic self-diffusion*. This type of diffusion embodies the long time scale effect of inter-particle collisions and transport [7]. The simplest way to model both these physical systems is to use the advection-diffusion equation. Since advection and dispersion of particles are already dealt with in the previous sections of this paper, we will concentrate solely on diffusion in this section. In two spatial dimensions, the diffusion equation is given by

$$\frac{\partial \rho}{\partial t} = D_x \frac{\partial^2 \rho}{\partial x^2} + D_y \frac{\partial^2 \rho}{\partial y^2}. \tag{6.1}$$

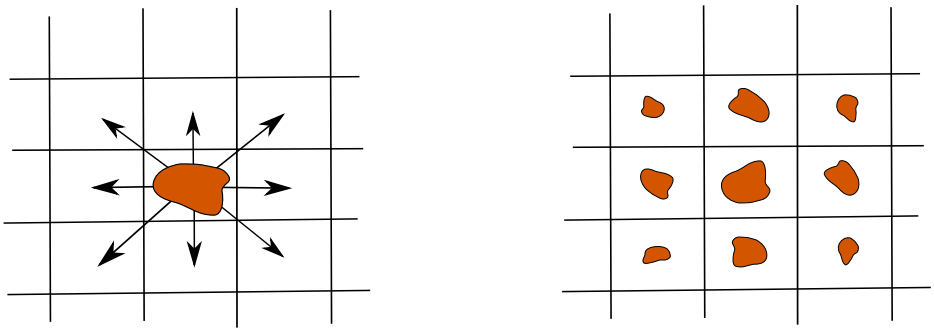


Figure 6: Illustration of the diffusion procedure. Particles within a grid cell are dispersed to adjacent grid cells.

Here, ρ is the particle density, and D_x and D_y are referred to as diffusion constants.

The diffusion constants are principal in specifying the time and length scales of the physical problem. For instance, in Brownian motion [8] one observes $D\sim 0.1-0.8\mu\text{m}^2/\text{s}$. However, in depositional modeling [32], these parameters are usually of the order $\sim 10^4 m^2/\text{year}$. We will focus on Brownian-like diffusion in this paper, even though the methods developed here can be applied to other settings as well. Brownian-like motion have traditionally been simulated with random walks [31], where each particle is displaced to certain directions, depending on the outcome of a random toss. When the particle number becomes large, the sum of all these random walks are shown to reproduce diffusion according to (6.1).

In our approach, particles are moved to specific grid cells according to a deterministic spreading procedure. This strategy mimics the net effect of a large number of stochastic collisions. Fig. 6 illustrates the basic concept. The particle lump within each grid cell is split up into smaller quasi-particles which then are transported to neighboring cells. Therefore, the diffusion velocity assigned to each quasi-particle will be a priori given, and does not depend on the flow configuration. Let $\mathbf{C}_l$ be the diffusion velocities of quasi-particle l. In two dimensions, the set of vectors that transport the quasi-particles to neighboring grid cells are given by

$$\begin{aligned}\mathbf{C}_l=\{&(0,0),(1,0),(0,1),(-1,0),\\ &(0,-1),(1,1),(-1,1),(-1,-1),(1,-1)\},\end{aligned} \tag{6.2}$$

which is expressed in grid cell units. Note the inclusion of the null vector, which allows a fraction of particles to remain stationary. Similar to what we did in Section 5.1, we us a fractional number distribution Q_l to quantify the fraction of particles at each grid cell that are transported in direction $\mathbf{C}_l$. Consequently the quasi-particle being advected with diffusion velocity $\mathbf{C}_l$ will consist of $Q_l N(x,y)$ particles. Many different distributions are possible for Q_l, as long as $\sum_l Q_l=1$. The even distribution $Q_l=\frac{1}{9}$ is at first glance the most natural choice. However, this choice gives rise to non-physical spatial grid polarization of the particle distribution. This may be remedied slightly with higher resolutions, at the expense of a larger computational cost. A better strategy would be to compensate for the

fact that diagonally moving particles travel further than those moving parallel with the grid lines. One distribution satisfying these criteria is

$$Q_l = \left\{ \frac{4}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{9}, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, \frac{1}{36} \right\}. \tag{6.3}$$

A distribution similar to Eq. (6.3) is commonly used in discrete fluid simulation to obtain Galilean invariant models, as for instance in the Lattice Boltzmann method [39]. This distribution is proven to practically eliminate grid polarization effects.

We define $N(x\pm\Delta x, y\pm\Delta y, t) \equiv N_{\pm\pm}$ and $N(x,y,t) \equiv N_{00}$. Furthermore, we set $N(x\pm\Delta x, y, t) = N_{\pm 0}$ and $N(x, y\pm\Delta x, t) = N_{0\pm}$. Moreover, let q_{ij} be a matrix representation of Q_l. For instance, $q_{00} = 4/9$, $q_{+0} = 1/9$ and $q_{--} = 1/36$, and so forth. Using these definitions, we can then express the diffusion procedure with

$$N(x,y,t+\Delta t) = \sum_{ij} q_{ij} N_{ij}. \tag{6.4}$$

Velocity and error correction vectors in the target grid cells are updated in the same way as described previously in Section 4. This also requires a recalculation of the dispersion directions as described in Section 5.

We have presented the basic concept of a procedure that can model diffusion phenomena within our lumped particle framework. In the appendix we show that this approach is indeed equivalent to the diffusion equation. For the distribution given by Eq. (6.3), we get

$$D_x = \frac{1}{6}\frac{\Delta x^2}{\Delta t}, \quad D_y = \frac{1}{6}\frac{\Delta y^2}{\Delta t}. \tag{6.5}$$

In three spatial dimensions, more velocities are required. Usually, a set of 19 symmetric velocities is sufficient to reproduce the three dimensional diffusion equation. The diffusion procedure in three spatial dimensions is given by

$$N(x,y,z,t+\Delta t) = \sum_{ijk} q_{ijk} N_{ijk}, \tag{6.6}$$

where q_{ijk} is a multi-dimensional representation of Q_l and $N_{\pm\pm\pm} \equiv N(x\pm\Delta x, y\pm\Delta y, z\pm\Delta z, t)$, with similar definitions for $N_{00\pm\pm}$ and so forth.

7 Numerical experiments

In the preceding sections we have presented our lumped particle approach to the simulation of particle flows. This paper is focused on the particle motion alone, and not does not include the dynamics of the ambient fluid. Therefore, model verification through comparisons with experimental data is not possible. Such comparisons will be left to future work where the two-way particle-fluid coupling will be studied in detail. Instead, we will present selected numerical experiments that illustrate different aspects of the framework. Whenever possible, the computational results are compared with analytic solutions.

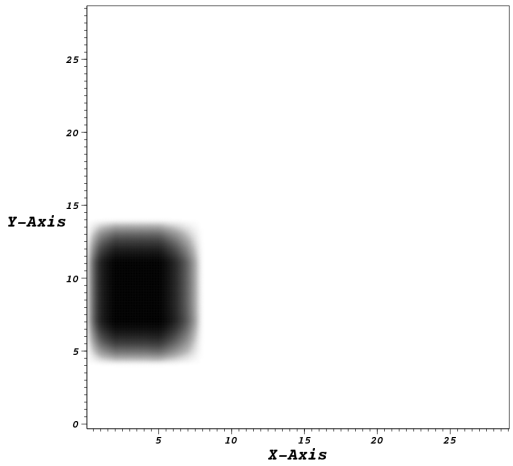


Figure 7: Illustration of the pure advection case. Particles are distributed evenly in a rectangular area. The domain is filled with a fluid that exhibits a drag force on the particles.

7.1 Pure advection with constant fluid velocity

These series of tests are designed to validate the advection part of the lumped model. In these cases, we use the physical domain $[0.0,5.0]\times[0.0,5.0]$, which is partitioned in 30×30 grid cells. The particles, of which there are 5828, are initially at rest and are distributed evenly in a rectangular area as shown in Fig. 7. The fluid is set to have a constant velocity $\mathbf{u}=(u_{\max},0)$ and the gravitational acceleration g is set to 0. Moreover, the relaxation time is set to $\tau_p=0.053$ and the time length of the time steps is $\mathrm{d}t=0.005$, with about 40 time steps being simulated. All the particles will experience the same acceleration since the drag force is the same in all grid cells. An analytical solution is available in this case, given by

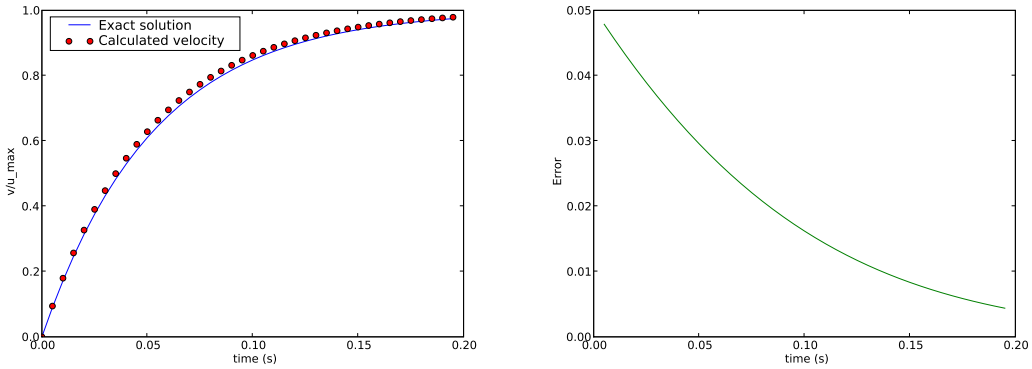
$$v(t)=u_{\max}\left(1-\exp\left[-\frac{t}{\tau_p}\right]\right), \tag{7.1}$$

where $v(t)$ is the average speed of the particles.

Fig. 8 shows the calculated average velocity compared to the analytic solution. Even with this relatively coarse grid and time stepping, we see that the simulated result is close to the analytic solution. In addition, test cases including the effects of gravity show similar results as described here, but with the limiting velocity becoming

$$\mathbf{v}_\infty=\left(u_{\max},(1-\varrho_f/\varrho_p)g\tau_p\right). \tag{7.2}$$

Finally turning off both gravity and the fluid drag force, the particles evolve freely. By construction, the kinetic energy of the particles are conserved throughout the computation. The late time scale behavior of the particles are studied in Section 7.3.



(a) Average velocity (b) Relative error

Figure 8: Pure advection: The average velocity as a function of time. The analytic solution (7.1) is compared with the calculated result from our model. The relative error is at most 0.05 and decreases steadily to 0.

7.2 Comparisons with the advection-diffusion equation

To validate the application of our model to diffusion, we will compare the computed results to an exact solution of the advection-diffusion equation, which in two spatial dimensions is given by

$$\frac{\partial \rho}{\partial t}+\frac{\partial (v_x\rho)}{\partial x}+\frac{\partial (v_y\rho)}{\partial y}=D_x\frac{\partial^2\rho}{\partial x^2}+D_y\frac{\partial^2\rho}{\partial y^2}. \tag{7.3}$$

Here, ρ is the particle density, $\mathbf{v}=(v_x,v_y)$ is the average velocity, and D_x and D_y are the diffusion constants in the x- and y-directions, respectively. Consider M particles normally distributed in a rectangular domain. The continuum density distribution $\rho(x,y)$ is defined as the number of particles within the grid cell divided by the cell's size.

With a gaussian initial condition on an infinite domain with 0 at infinity shown in Fig. 9, the exact solution ρ_e is given by

$$\rho_e(x,y,t)=\frac{M}{2\pi}\frac{1}{\sqrt{\sigma_x^2+2D_xt}\sqrt{\sigma_y^2+2D_yt}}\exp\left[-\frac{(x-\bar{x})^2}{2\sigma_x^2+4D_xt}-\frac{(y-\bar{y})^2}{2\sigma_y^2+4D_yt}\right], \tag{7.4}$$

where $\bar{x}=x_0-v_xt$ and $\bar{y}=y_0-v_yt$, with $(x_0,\ y_0)$ being the position of the maximum value. Furthermore, σ_x and σ_y are the standard deviations of the initial gaussian in the respective directions and should not be confused with the dispersion vectors in Section 5.1. In all of the following examples we have chosen $M=9999880$, which is a sufficiently high particle number to ensure a smooth distribution. Recall from Section 6 that the

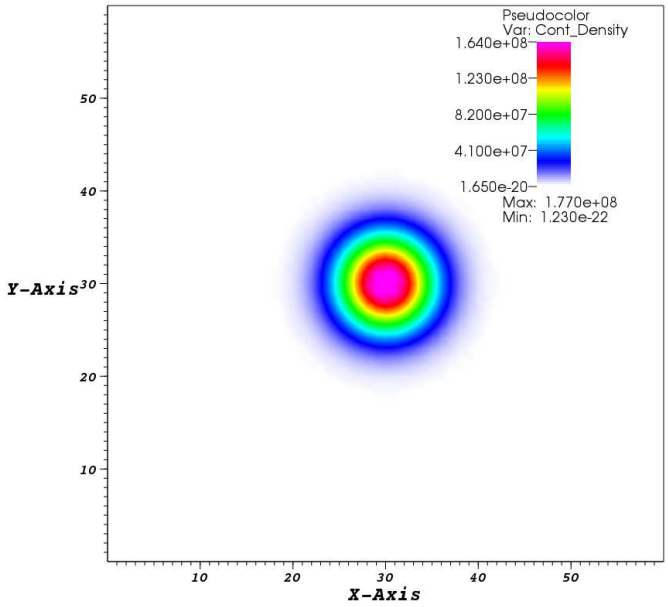


Figure 9: Initial distribution of particles. This figure shows the density of particles within the domain at $t=0$.

diffusion coefficients are calculated to be

$$D_x = \frac{1}{6}\frac{\Delta x^2}{\Delta t}, \tag{7.5}$$

$$D_y = \frac{1}{6}\frac{\Delta y^2}{\Delta t}. \tag{7.6}$$

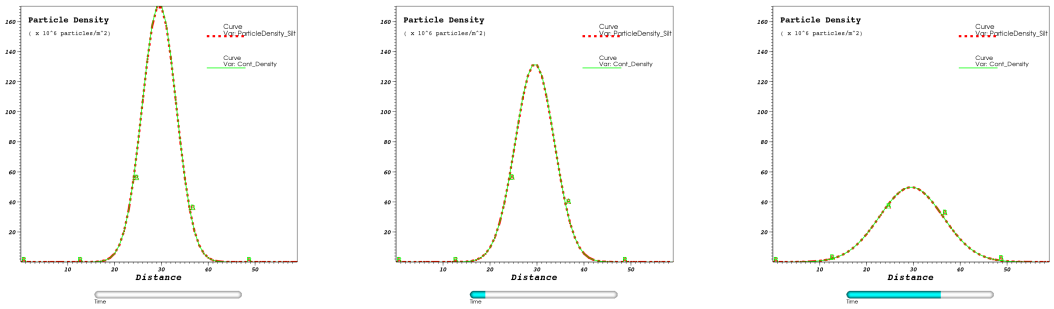
The density function $N(x,y)$, which is the number of particles within each grid cell, is set equal to the initial Gaussian distribution at $t=0$. This is accomplished by the calculation

$$N(x,y) = \left[\int_{x-\Delta x}^{x+\Delta x}\int_{y-\Delta y}^{y+\Delta y}\rho(x,y)\mathrm{d}x\mathrm{d}y\right], \tag{7.7}$$

where $[Y]$ is the closest integer to Y. To view the distribution more clearly, a slice along the line $y=30$ has been extracted. Fig. 10 shows this slice, where both the exact solution (7.4) and the simulated solution are displayed. Here, the dashed red line represents the simulated result, and the solid green line indicates the exact solution. Note that small roundoff errors are expected from the integer operator. This can be seen in Fig. 11, where we have magnified the top part of the distribution. The discrepancy is small, and is only relevant in the first time step. To quantify the error in the simulation in this case, the relative L_2 norm is used:

$$\epsilon = \sqrt{\frac{\sum(\rho_c-\rho_e)^2}{\sum\rho_e^2}}, \tag{7.8}$$

where the sum runs over all grid cells. Here, ρ_c is the simulated result.



(a) Initial (b) Intermediate (c) Final state

Figure 10: Evolution of the particle distribution.

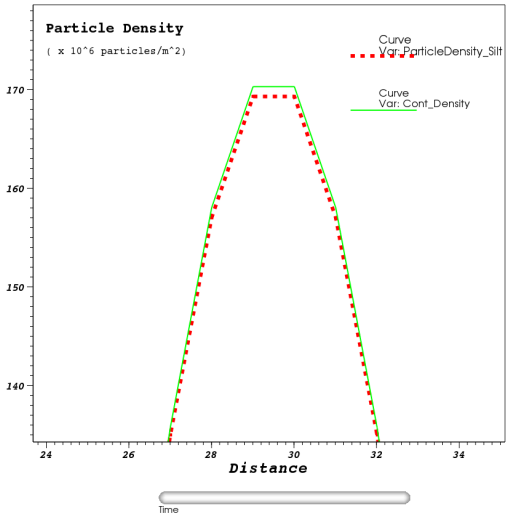


Figure 11: Advection-Diffusion comparison: Initial distribution of particles. This figure shows a magnification of the initial density of particles within the domain. The green curve is represents the exact solution, and the red curve is calculated from our model.

Diffusion

We now set both the particle and fluid velocities to 0. To capture the effect of a decreasing error as the computational grid is refined, three spatial resolutions have been used; $\Delta x = \Delta y = 0.083$, $\Delta x = \Delta y = 0.062$ and $\Delta x = \Delta y = 0.05$. These values obviously correspond to increasing the number of grid points along each direction, where each grid consists of 61×61 , 81×81, and 101×101 grid points. To compensate for decreasing diffusion speed with grid resolution, the time increments $\Delta t = 0.01$, $\Delta t = 0.015$, and $\Delta t = 0.02$ has been used for the respective grid.

Fig. 10 shows the temporal evolution of the distribution with $\Delta x = \Delta y = 0.083$. We observe that the difference between the distributions is so small that the graphs seem to coincide at each point. This can be seen more clearly by studying the L_2 norm of the

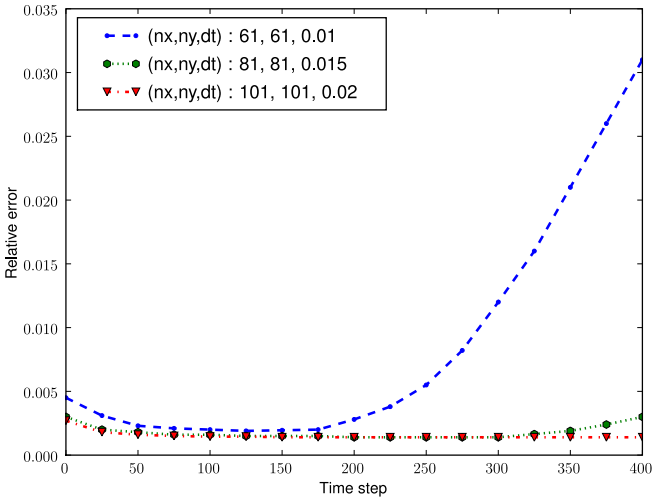


Figure 12: The relative error between the calculated distribution and the exact solution. Three spatial resolutions have been used: $\Delta x=\Delta y=0.083$ (blue curve), $\Delta x=\Delta y=0.062$ (green curve), and $\Delta x=\Delta y=0.05$ (the red curve).

difference between the two distributions. Fig. 12 shows the relative L_2 norm of the difference between the simulated and exact solutions for all three grid resolutions. Overall, it is clear that the error is small and reduces as the grid resolution increases. Notice that from $t=0$ to about $t\sim 50-70$, the error decreases from 0.5% to about 0.1%. This is due to the roundoff errors from the integer operator being evened out. It is also apparent that this error diminishes as resolution increases.

Advection-diffusion

Choosing a linear non-zero velocity for the fluid $\mathbf{u}=(u_0,0)$, we initialize all the particles to this velocity. Hence, there is no acceleration of the particles. For simplicity, we choose $u_0=\Delta x/\Delta t$ to avoid grid resolution effects. Moreover, we use periodic boundary conditions such that particles leaving the domain are inserted back at the opposite side of the domain. Until these particles reach the backwards diffusing ones, no significant differences from the pure diffusion case is observed.

In conclusion, we have shown that our model reproduces an advection-diffusion process that is consistent with (6.1). However, one reason for concern is that there does seem to be evidence for a long time divergent behavior. Indeed, in the blue curve of Fig. 12, the small errors seem to accumulate and steadily increase. This is also present in the green curve, although to a lesser degree. The error accumulation is not visible in the red curve, implying that if this effect is there, it would only become significant at very late times. For our applications, we do not see the error accumulation effect as a substantial problem, since we are interested in physical effects that occur over relatively short time spans.

7.3 Long time scale behavior of freely moving particles

In this section we explore the behavior of a set a freely evolving particles in a fluid at long time scales. Usually, the averaged dynamics of particles such as these approaches a diffusion-like state with time [29]. Our aim is to show that the averaging procedure of the lumped particle model recreates a diffusion-like behavior when the BBO-equation has little effect on the particles' movement. This situation typically occurs at late states of the simulation when the particles attain a velocity equal to the fluid's velocity.

We use the physical domain $[0.0,7.0]\times[0.0,7.0]$, which is partitioned in 351×351 grid cells. This division corresponds to a grid spacing of $\Delta x=\Delta y=0.02$. The drag force and the gravitational acceleration are both set to zero. Moreover, the relaxation time is $\tau_p=5.3$, and the length of the time steps is chosen to be $\mathrm{d}t=0.5$, with about 90 time steps being simulated. The particles, of which there are 100000, are distributed evenly in the rectangular area $[1.5,2.5]\times[1.5,2.0]$ within the domain. This is similar to what is shown in Fig. 7 concerning the case in Section 7.1. But here, we are using a higher number of particles, in addition to a higher grid resolution. Each particle is initialized with a velocity

$$\mathbf{v}_p=v_0(\cos\eta,\sin\eta), \tag{7.9}$$

where η is a uniformly distributed random variable in $[-\pi/3,\pi/3]$ and $v_0=0.1$. From the particle velocities, the average velocity $\mathbf{V}$ and the dispersion directions are calculated as described in Section 5. The particles are then evolved using our model, but omitting the diffusion step described in Section 6.

To quantify the average displacement of the particles, we use the average distance from the centroid (x_0,y_0) of the particles at $t=0$. The centroid is given by

$$x_0=\frac{1}{M}\sum_i x_i, \tag{7.10}$$

$$y_0=\frac{1}{M}\sum_i y_i, \tag{7.11}$$

where (x_i,y_i) is the position of particle i and $M=100000$ is the total number of particles. We define the average displacements in the x- and y-direction by

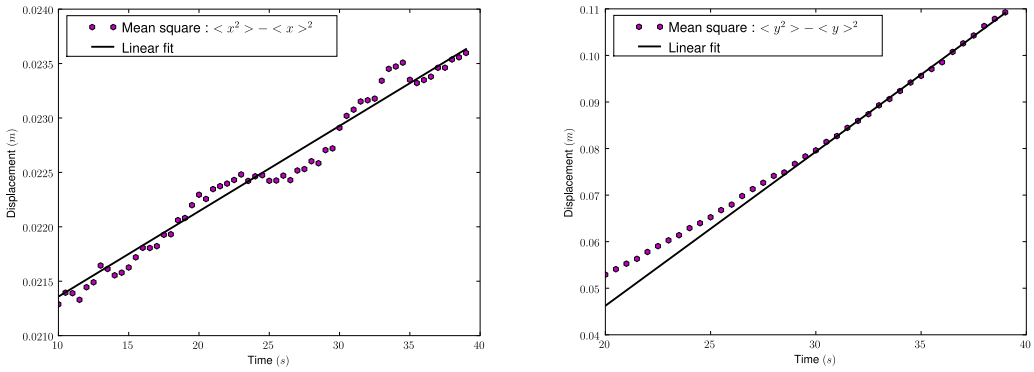
$$<x>=\frac{1}{M}\sum_i(x_i-x_0), \tag{7.12}$$

$$<y>=\frac{1}{M}\sum_i(y_i-y_0). \tag{7.13}$$

Similarly, we define

$$<x^2>=\frac{1}{M}\sum_i(x_i-x_0)^2, \tag{7.14}$$

$$<y^2>=\frac{1}{M}\sum_i(y_i-y_0)^2. \tag{7.15}$$



(a) Mean square displacement in the x-direction. (b) Mean square displacement in the y-direction.

Figure 13: The mean square displacement of freely evolving particles.

This enables us to calculate the mean square displacement $<x^2>-<x>^2$ and $<y^2>-<y>^2$ at each time step in the simulation.

The particle motion is diffusive if, as time passes, the average square displacement of the particles depends linearly on time [1]. This means that

$$<x^2>-<x>^2 \propto t, \tag{7.16}$$

$$<y^2>-<y>^2 \propto t. \tag{7.17}$$

We compute the least square linear fit of the mean square displacements for $t>10$ for the x-direction, and $t>30$ for the y-direction. The specific time values are chosen to be the values for which a linear profile becomes visible, and are in general dependent on the initial velocity distribution. The results from the calculations are plotted in Fig. 13(a) and Fig. 13(b). In these figures, we can observe the linear behavior of the average square displacement. To be able to quantify the degree of linearity of the data set, we calculate the *correlation coefficient* [34] given by

$$r_l = \frac{\sum_k (t_k-\bar{t})(x_{l,k}-\bar{x}_l)}{[\sum_k (t_k-\bar{t})^2 \sum_k (x_{l,k}-\bar{x}_l)^2]^{1/2}}, \tag{7.18}$$

where $\{t_k\}$ is the time values and $\{x_{l,k}\}$ is the mean square displacement in direction l. Moreover, $\bar{t}$ and $\bar{x}_l$ are the mean values of $\{t_k\}$ and $\{x_{l,k}\}$, respectively. A value of $r=1$ implies a perfect linear relationship between the datasets. For the mean square displacements, we obtain

$$r_x = 0.988 \quad \text{and} \quad r_y = 0.997. \tag{7.19}$$

This leads us to conclude that our model correctly predicts the diffusive nature of the particles' motion at late times.

8 Conclusion

In this paper we have described a novel lumped particle framework for modeling transport of a large number of particles suspended in a fluid. Our model considers a group of particles within a certain volume as a single entity. The overall distribution of the lumped particle is updated by using the Bassinet-Boussinesq-Oseen equation combined with a method for particle dispersion. A diffusion phase can be included as well if Brownian movement is relevant. We have showed that our model includes properties inherent in both continuum and discrete approaches, and that we correctly reproduce advection and diffusion phenomena. Furthermore, the averaging procedure of the model is shown to reproduce long time scale behavior of simple flows of particles. The assumptions we currently have made is that the particles are all of the same size and that the effects of interparticle collisions are negligible. Furthermore, only the drag force and gravitational effects are included in the Bassinet-Boussinesq-Oseen equation.

Acknowledgments

The presented work was funded by a research grant from StatoilHydro. The work has been conducted at Simula Research Laboratory as part of CBC, a Center of Excellence awarded by the Research Council of Norway.

Appendix: Derivation of the continuum limit of the diffusion procedure

In this appendix, we will show that the discrete diffusion model proposed in this paper reproduces a standard continuum-based diffusion equation is reproduced. Throughout this derivation, we will assume that the average velocity $\mathbf{V}$ and the error correction d$\mathbf{s}$ are zero. Additionally, the particle density function $N(x,y,t)$ is assumed to be twice differentiable. This condition is usually satisfied when there are a few thousand particles in each grid cell.

In Section 6, we explained that the diffusion procedure could be expressed as

$$N(x,y,t+\Delta t)=\sum_{ij} q_{ij}N_{ij}, \tag{A.1}$$

where we defined $N(x\pm\Delta x,y\pm\Delta y,t)\equiv N_{\pm\pm}$ and $N(x,y,t)\equiv N_{00}$. Moreover, q_{ij} is a matrix representation of Q_l. In this derivation, we require that the velocities in Q_l is axis-symmetric, such that diagonally moving particles have the same distribution, and the horizontal and vertical moving particles as well. That is, $Q_1=\cdots=Q_4\equiv q_a$ and $Q_5=\cdots=$

$Q_8 \equiv q_d$. Examining the right hand side of Eq. (A.1), we see that

$$\begin{aligned}\sum_{ij} q_{ij}N_{ij} &= q_{00}N_{00} + \sum_{ij\neq 0,0} q_{ij}N_{ij} \\ &= q_{00}N_{00} + q_a \underbrace{(N_{+0}+N_{0+}+N_{0-}+N_{-0})}_{\mathbf{I}} \\ &\quad + q_d \underbrace{(N_{++}+N_{-+}+N_{+-}+N_{-+})}_{\mathbf{II}}.\end{aligned} \tag{A.2}$$

The terms labeled '**I**' represent particles entering the grid cell from vertical and horizontal directions, while the terms labeled '**II**' are the diagonal contributions. We will now expand these terms as Taylor series'. For convenience, we write $\partial/\partial t = \partial_t$, $\partial/\partial x = \partial_x$ and $\partial/\partial y = \partial_y$, with double derivatives taking on the form $\partial^2/\partial x^2 = \partial_{xx}$ and so on. Consider first the terms N_{+0}, N_{0+}, N_{0-} and N_{-0}. A second order expansion of these quantities leads to

$$\begin{aligned}N_{+0} &= N_{00} + \partial_x N_{00}\Delta x + \frac{1}{2}\partial_{xx}N_{00}\Delta x^2 + \mathcal{O}\left(\Delta x^3\right), \\ N_{0+} &= N_{00} + \partial_y N_{00}\Delta y + \frac{1}{2}\partial_{yy}N_{00}\Delta y^2 + \mathcal{O}\left(\Delta y^3\right), \\ N_{0-} &= N_{00} - \partial_y N_{00}\Delta y + \frac{1}{2}\partial_{yy}N_{00}\Delta y^2 + \mathcal{O}\left(\Delta y^3\right), \\ N_{-0} &= N_{00} - \partial_x N_{00}\Delta x + \frac{1}{2}\partial_{xx}N_{00}\Delta x^2 + \mathcal{O}\left(\Delta x^3\right).\end{aligned} \tag{A.3}$$

Adding these together gives the terms labeled '**I**'

$$4N_{00} + \partial_{yy}N_{00}\Delta y^2 + \partial_{xx}N_{00}\Delta x^2 + \mathcal{O}\left(\Delta x^3, \Delta y^3\right). \tag{A.4}$$

Expanding the diagonal terms ('**II**') we get

$$\begin{aligned}N_{++} &\simeq N_{00} + \partial_x N_{00}\Delta x + \partial_y N_{00}\Delta y \\ &\quad + \frac{1}{2}\left(\partial_{xx}N_{00}\Delta x^2 + 2\partial_{xy}N_{00}\Delta x\Delta y + \partial_{yy}N_{00}\Delta y^2\right), \\ N_{-+} &\simeq N_{00} - \partial_x N_{00}\Delta x + \partial_y N_{00}\Delta y \\ &\quad + \frac{1}{2}\left(\partial_{xx}N_{00}\Delta x^2 - 2\partial_{xy}N_{00}\Delta x\Delta y + \partial_{yy}N_{00}\Delta y^2\right), \\ N_{+-} &\simeq N_{00} + \partial_x N_{00}\Delta x - \partial_y N_{00}\Delta y \\ &\quad + \frac{1}{2}\left(\partial_{xx}N_{00}\Delta x^2 - 2\partial_{xy}N_{00}\Delta x\Delta y + \partial_{yy}N_{00}\Delta y^2\right), \\ N_{--} &\simeq N_{00} - \partial_x N_{00}\Delta x - \partial_y N_{00}\Delta y \\ &\quad + \frac{1}{2}\left(\partial_{xx}N_{00}\Delta x^2 + 2\partial_{xy}N_{00}\Delta x\Delta y + \partial_{yy}N_{00}\Delta y^2\right),\end{aligned} \tag{A.5}$$

where all terms of the order $\mathcal{O}(\Delta x^m, \Delta y^n)$ with $m+n>2$ have been omitted. Adding all these terms together, we see that the diagonal terms becomes

$$4N_{00}+2\partial_{yy}N_{00}\Delta y^2+2\partial_{xx}N_{00}\Delta x^2+\mathcal{O}(\Delta x^m,\Delta y^n). \tag{A.6}$$

Inserting the expressions (A.4) and (A.6) into Eq. (A.2), we obtain

$$\begin{aligned}\sum_{ij} q_{ij}N_{ij} =& q_{00}N_{00}+q_a\left(4N_{00}+\partial_{yy}N_{00}\Delta y^2+\partial_{xx}N_{00}\Delta x^2\right)\\ &+q_d\left(4N_{00}+2\partial_{yy}N_{00}\Delta y^2+2\partial_{xx}N_{00}\Delta x^2\right)+\mathcal{O}(\Delta x^m,\Delta y^n)\\ =& N_{00}+(q_a+2q_d)\partial_{xx}N_{00}\Delta x^2+(q_a+2q_d)\partial_{yy}N_{00}\Delta y^2+\mathcal{O}(\Delta x^m,\Delta y^n).\end{aligned} \tag{A.7}$$

where we have used that $q_{00}+4q_a+4q_d=1$ and $m+n>2$. The left hand side of (A.1) can similarly be expanded in the variable t

$$N(x,y,t+\Delta t)=N_{00}+\Delta t\partial_t N_{00}+\mathcal{O}\left(\Delta t^2\right). \tag{A.8}$$

Finally, the combination of (A.7) and (A.8) with Eq. (A.1) finally leads to

$$\begin{aligned}N_{00}+\Delta t\partial_t N_{00} &= N_{00}+(q_a+2q_d)\partial_{xx}N_{00}\Delta x^2+(q_a+2q_d)\partial_{yy}N_{00}\Delta y^2\\ &\Downarrow\\ \partial_t N_{00} &= \frac{(q_a+2q_d)\Delta x^2}{\Delta t}\partial_{xx}N_{00}+\frac{(q_a+2q_d)\Delta y^2}{\Delta t}\partial_{yy}N_{00}.\end{aligned} \tag{A.9}$$

We can then define

$$D_x=(q_a+2q_d)\frac{\Delta x^2}{\Delta t},\quad D_y=(q_a+2q_d)\frac{\Delta y^2}{\Delta t}, \tag{A.10}$$

which finally gives

$$\partial_t N(x,y,t)=D_x\partial_{xx}N(x,y,t)+D_y\partial_{yy}N(x,y,t). \tag{A.11}$$

This argument shows that the proposed diffusion procedure is a consistent approximation in that more accurate results are obtained as Δx, Δy and Δt tends to 0. Note that q_{00} has only indirect effect on the value of the diffusion constants. For the distribution given by Eq. (6.3), we obtain

$$D_x=\frac{1}{6}\frac{\Delta x^2}{\Delta t},\quad D_y=\frac{1}{6}\frac{\Delta y^2}{\Delta t}. \tag{A.12}$$

References

[1] T. Ala-Nissila, R. Ferrando, and S. C. Ying. Collective and single particle diffusion on surfaces. Advances in Physics, 51(3), 2002.

[2] B. J. Alder and T. E. Wainwright. Studies in molecular dynamics. i. general method. The Journal of Chemical Physics, 31(2):459–466, 1959.

[3] T. B. Anderson and Roy Jackson. Fluid mechanical description of fluidized beds. equations of motion. Industrial & Engineering Chemistry Fundamentals, 6(4):527–539, 1967.
[4] Santosh Ansumali, Shyam Sunder Chikatamarla, Christos Emmanouil Frouzakis, and Konstantinos Boulouchos. Entropic lattice boltzmann simulation of the flow past square cylinder. International Journal of Modern Physics C, 15:435, 2004.
[5] C. T. Crowe, T. R. Troutt, and J. N. Chung. Numerical models for two-phase turbulent flows. Annual Review of Fluid Mechanics, 28(11), 1996.
[6] Jennifer Sinclair Curtis and Berend van Wachem. Modeling particle-laden flows: A research outlook. AIChE Journal, 50(11):2638–2645, 2004.
[7] E. C. Eckstein, D. G. Bailey, and A. H. Shapiro. Self-diffusion of particles in shear flow of a suspension. Journal of Fluid Mechanics, 79:191–208, 1977.
[8] A. Einstein. Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen. Annalen der Physik, 17:549–560, 1905.
[9] H. Enwald, P. Mege, and A. Almstedt. Eulerian two-phase flow theory applied to fluidization. Journah of Multiphase Flow, 22:21–66, 1996.
[10] Pep Español and Mariano Revenga. Smoothed dissipative particle dynamics. Phys. Rev. E, 67(2):026705, Feb 2003.
[11] A. V. Filippov. Drag and torque on clusters of n arbitrary spheres at low reynolds number. Journal of Colloid and Interface Science, 229(1):184 – 195, 2000.
[12] R. Garg, C. Narayanan, D. Lakehal, and S. Subramaniam. Accurate numerical estimation of interphase momentum transfer in lagrangian-eulerian simulations of dispersed two-phase flows. International Journal of Multiphase Flow, 33(12):1337 – 1364, 2007.
[13] Tamas I. Gombosi. Gaskinetic Theory. Cambridge University Press, 1994.
[14] P. J. Hoogerbrugge and J. M. V. A. Koelman. Simulating microscopic hydrodynamic phenomena with dissipative particle dynamics. EPL (Europhysics Letters), 19(3):155–160, 1992.
[15] K. Hutter, B. Svendsen, and D. Rickenmann. Debris flow modeling: A review. Continiuum Mech. Thermodynamics, 8:1–35, 1996.
[16] T. Ihle and D. M. Kroll. Stochastic rotation dynamics. i. formalism, galilean invariance, and green-kubo relations. Phys. Rev. E, 67(6):066705, Jun 2003.
[17] Petros Koumoutsakos. Multiscale flow simulations using particles. Annu. Rev. Fluid Mech., 37:457–87, Feb 2005.
[18] A.J.C Ladd. Hydrodynamic transport coefficients of random dispersions of hard spheres. Journal of Chemical Engineering, 93:3484, 1990.
[19] M. Loewenberg and E. Hinch. Numerical simulation of a concentrated emulsion in shear flow. Journal of Fluid Mechanics, 321:395–419, 1996.
[20] Tore M. Løseth. Submarine Massflow Sedimentation - Computer Modelling and Basin-Fill Stratigraphy. Springer, Lecture Notes in Earth Sciences, 1999.
[21] C. Lun, S. Savage, D. Jefferey, and N. Chepurniy. Kinetic theories for granular flow: inelastic particles in couette flow and slightly inelastic particles in a general flowfield. Journal of Fluid Mechanics, 140:223–256, 1984.
[22] N.S. Martys. Study of a dissipative particle dynamics based approach for modeling suspensions. Journal of Rheology, 49(2):401–424, 2005.
[23] MR Maxey and JJ Riley. Equation of motion for a small rigid sphere. Physical Fluids, 24(4):883–889, 1983.
[24] JB. Mclaughlin. Numerical computation of particle-turbulence interaction. International Journal of Multiphase Flow, 20:211–232, 1994. Supplement.
[25] Paul McLeod, Steven Carey, R. Stephen, and J. Sparks. Behaviour of particle-lade flows into

the ocean: experimental simulation and geological implications. Sedimentation, 46:523–536, 1999.
[26] G. McNamara and G. Zanetti. Use of the Boltzmann equation to simulate Lattice Gas Automata. Phys. Rev. Lett., 61:2332–2335, 1988.
[27] G.V. Middelton. Sediment deposition from turbidity currents. Annual Review of Earth Planetary Science, 21:89–114, 1994.
[28] Ante Munjiza. The Combined Finite-Discrete Element Method. Wiley, 2004.
[29] H. Nicolai, B. Herzhaft, E. J. Hinch, L. Oger, and E. Guazzelli. Particle velocity fluctuations and hydrodynamic self-diffusion of sedimenting non-brownian spheres. Physics of Fluids, 7(1):12–23, 1995.
[30] Yarko Nino and Marcelo Garcia. Using lagrangian particle saltation observations for bedload sediment transport modelling. Hydrological Processes, 12(8):1197–1218, 1998.
[31] E.S. Oran, C.K. Oh, and B.Z. Cybyk. Direct simulation monte carlo: Recent advances and applications1. Annual Review of Fluid Mechanics, 30(1):403–441, 1998.
[32] Chris Paola. Quantitative models of sedimentary basin filling. Sedimentology, 47(Suppl. 1):121–178, 2000.
[33] G. Parker, Y. Fukushima, and H. Pantin. Self-accelerating turbidity currents. Journal of Fluid Mechanics, 171:145–181, 1986.
[34] Joseph Lee Rodgers and W. Alan Nicewander. Thirteen ways to look at the correlation coefficient. The American Statistician, 42(1):59–66, 1988.
[35] Huang Shehua, Li Wei, and Cheng Liangjun. On equation of discrete solid particles' motion in arbitrary flow field and its properties. Applied Mathematics and Mechanics, 21:297–310, 2000.
[36] J.S. Shirolkar, C.F.M Coimbra, and M. Queroz McQuay. Fundemental aspects of modeling turbulent particle dispersion in dilute flows. Progress In Energy Combustion Science, 22:363–399, 1996.
[37] Jonathan J. Stickel and Robert L. Powell. Fluid mechanics and rheology of dense suspensions. Annual Review of Fluid Mechanics, 37(1):129–149, 2005.
[38] DE. Stock. Freeman scholar lecture: Particle dispersion in gases. Journal of Fluids Engineering, 118(1), 1995.
[39] Sauro Succi. The Lattice Boltzmann Equation : For Fluid Dynamics and Beyond. Clarendon Press, 2001.
[40] B. G. M. van Wachem and A. E. Almstedt. Methods for multiphase computational fluid dynamics. Chemical Engineering Journal, 96(1-3):81 – 98, 2003.
[41] R.G. Walker. The origin and significance of the internal sedimentary structures of turbidites. Proceedings of the Yorkshire Geological Society, 35:1–32, 1965.
[42] D. Zhang and R. Rauenzahn. A viscoelastic model for dense granular flows. Journal of Rheology, 41:1275–1298, 1997.