The Physalis Method for Particulate Flow Simulations: Applications and Extensions

by

Daniel P. Willen

A dissertation submitted to The Johns Hopkins University
in conformity with the requirements for the degree of
Doctor of Philosophy

Baltimore, Maryland
July, 2018

Abstract

Particulate multiphase flows, in which a disperse solid phase is immersed in a continuous fluid medium, are relevant in a wide variety of natural and technological phenomena. Examples include dust storms, avalanches, and sediment transport, as well as fluidized bed reactors, grain transport, and combustion. The prediction and understanding of such flows is highly desirable, but the large range of scales involved presents a difficult challenge for researchers. For instance, dust storms can be several kilometers long but contain particles whose diameter is on the order of tens of micrometers. This makes it difficult to account for all the scales in experimental or numerical investigations and so much recent work has focused on investigating highly resolved simulations of small-scale flows with the intent of gaining insight into the microscopic interactions between the fluid and the particles. This new knowledge can then be applied to reduced-order models capable of capturing macroscopic phenomena.

In the present work, we report on several such investigations of particulate flows, with a focus on a set of simulations representing triply-periodic sedimentation of a suspension composed of solid spherical particles immersed in a fluid. Using a technique to represent the disperse particle phase as a (coarse-grained) continuous field, we identify the presence of continuity waves in the simulations and measure their wave speed. We find excellent agreement of the measured wave speed with a prediction based on a one-dimensional balance equation for the particle volume fraction, which indicates that the method used to coarse-grain the particle phase accurately captures the macroscopic features of the flow.

Additionally, we investigate elements of the suspension microstructure concerning particle collisions, diffusivities, mean free paths, pair distribution function and other features. It is found that many qualitative trends found in earlier studies continue to hold in the parameter range investigated here as well. The analysis of collisions reveals that particles interact prevalently via their flow fields rather than by direct contacts and a tendency towards particle clustering is demonstrated. The time evolution of the shape and size of particle tetrads is also examined.

The simulations used in the aforementioned results were performed using a computational fluid dynamics simulation code that incorporates the Physalis method to apply particle boundary condition to the fluid fields. The implementation, which was developed by a previous graduate student, runs on a single graphics processing unit (GPU). In order to increase the possible domain size of the simulations, which is limited by the memory space on a single GPU, we present a new implementation of the simulation code that uses distributed memory to take advantage of supercomputers with hundreds of GPUs, allowing for much larger simulations. We document changes necessary to move to a distributed memory model for both the fluid and solid phases, and benchmark the code on up to one million resolved particles in a domain size of 1920^3 on 216 GPUs at the Maryland Advanced Research Computing Center. We also present favorable strong and weak scaling results, as well as validation of the implementation using two test cases.

Thesis Committee

Dr. Andrea Prosperetti (Primary Advisor)
Homewood Professor of Engineering
Department of Mechanical Engineering
Whiting School of Engineering
Johns Hopkins University

Distinguished Professor of Mechanical Engineering
Department of Mechanical Engineering
Cullen College of Engineering
University of Houston

G. Berkhoff Professor of Applied Physics
Faculty of Science and Technology
J.M. Burgers Centre for Fluid Dynamics
University of Twente

Dr. Gretar Tryggvason
Charles A. Miller Jr. Distinguished Professor and Department Head
Department of Mechanical Engineering
Whiting School of Engineering
Johns Hopkins University

Dr. Tamer Zaki
Associate Professor
Department of Mechanical Engineering
Whiting School of Engineering
Johns Hopkins University

Acknowledgments

This work would not have been possible without the continuous support and guidance of my advisor, Dr. Andrea Prosperetti. I'd also like to acknowledge Dr. Adam Sierakowski, whose mentorship and assistance was invaluable. I will miss working with both of these tireless (and patient!) individuals.

I would like to explicitly thank Dr. Tamer Zaki and Dr. Gretar Tryggvason for reviewing this dissertation and taking part in the defense committee, as well as all of the faculty and staff in the Mechanical Engineering department who have contributed to my academic development or assisted me in any way since I first came to Hopkins. The Prosperetti research group, including Dr. Adam Sierakowski, Dr. Shigan Chu, Dr. Yayun Wang, Yuhang Zhang (who devised the Poisson matrix symmetrization procedure in Chapter 5), and Gedi Zhou, supplied many interesting and productive conversations throughout the research process. I wish to thank Dr. Amit Amritkar for some useful discussions at the beginning of the many-GPU development, as well as the University of Houston for hosting me for several months at the beginning of 2018. Much of this work was sponsored by the U.S. National Science Foundation under grant number CBET 1335965 and additional computational resources were provided by the Maryland Advanced Research Computing Center.

Finally, I would like to recognize everyone who has provided friendship and support over the years, in particular Laughlin Barker, Joel Bretheim, Alex Caffee, Malachy Duffy, Perry Johnson, Carl Shapiro, Adam Sierakowski, and Andrew Spielvogel. And, most of all,

my parents Barb and Bill, my brother Mike, as well as Michelle Chen, for their love and encouragement (and occasional reality check!).

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Disperse multiphase flows, which consist of a disperse phase immersed in a continuous fluid medium, are common in many applications. Examples in nature include the formation of rain drops, the erosion and transport of material on the Earth's surface, and the behavior of powder snow avalanches, whereas technological examples include fluidized beds and thermal spray coating. In general, these systems encompass a wide variety of physics. In addition to the hydrodynamic interactions between the disperse and continuous phases, heat transfer, electrostatics, and chemical reactions are present in many of these systems (see, e.g., [25]). As an example of the complexity that arises from the multi-physics interactions, the formation of rain droplets in clouds is still an active area of research despite its everyday occurrence [49].

Considering only mass and momentum conservation of the continuous and disperse phases simplifies the picture somewhat, but understanding and predicting these systems is still a formidable challenge. For one, many disperse flows involve a large range of scales that renders direct analysis difficult, as the microscale effects have a direct impact on the macroscopic features—a dust storm may span several kilometers, but the individual dust particles are on the order of tens of micrometers. Additionally, many disperse flows exist at a scale where both the continuum nature of the fluid and the discrete nature of the disperse particles are

important to the overall dynamics. In contrast, at atomic and astronomic scales, particles exist in a vacuum and typically interact through electrostatic and gravitational central forces, respectively. At more moderate scales, the fluid may either be ignored (e.g., granular gases [17]) or its effect linearized (e.g., Stokesian dynamics [16]). In this work, we will focus on solid spherical particles immersed in a liquid at moderate Reynolds numbers, a regime for which the full non-linear effects of the fluid need to considered.

An application of particular interest is a fluidized bed, in which a stationary bed of particles is subjected to an upward-directed fluid stream. Once the flow velocity exceeds the so-called minimum fluidization velocity (see, e.g., [56]), the drag on the particles exceeds their weight and the particles become suspended in the fluid. As the fluid velocity increases further, the mean particle concentration decreases until, when the flow velocity becomes comparable to the single-particle terminal settling velocity, they are blown out of the system. Fluidized beds find major applications in chemical engineering, the oil industry, combustion, and other areas because of the intimate contact that they promote between the solid and fluid phases, increasing the efficacy of heat and mass transfer. However, inhomogeneities in the mixture, such as "bubbles" of low particle concentration, decrease the efficiency of fluidized beds as a large amount of fluid can leave the bed without interacting with many particles.

Understanding the mechanisms behind bubble formation in fluidized beds would have a great impact on fluidized processes, but the complexity of the system makes prediction and analysis of disperse flows difficult. Theoretical results are usually limited to very high or very low Reynolds numbers, even for single particles in a fluid. Experimental approaches run into difficulties as beds can be nearly optically opaque even at low volume fractions and it is difficult to track thousands of particles and the flow fields surrounding them with traditional single-phase techniques. Macroscopic results from these methods that hold over a

large range of parameters are thus few—for example, while the Richardson-Zaki correlation (which relates the hindered settling speed of particles to the particle volume fraction) seems to be a robust feature of fluidization, the results are purely empirical [88]. Other important macroscopic considerations, particularly for designing industrial systems, include the bulk velocity, rheological properties such as the mixture viscosity, and hydrodynamic diffusion of the particles. These macroscopic features are influenced by microscopic properties such as the relative distribution of particles and the interparticle and hydrodynamic forces. Understanding these properties is crucial to improving knowledge of more complex features such as bubble formation.

Numerical investigations, while computationally expensive, are capable of simulating a wide variety of problems and provide access to all of the fluid and particle information. Several methods have been devised that are capable of modeling fluidized beds, each with their limitations and benefits. One method involves averaging the equations of motion governing the fluid and solid phases, resulting in the two-fluid equations in which the disperse particle phase is treated as continuum. Although this makes the equations more amenable to numerical simulations than accounting for the effect of each individual particle, the resulting equations are unclosed and closure has remained elusive [43]. Though the two-fluid models sacrifice physical accuracy in that the closure relations are typically empirical and the small-scale details of the fluid-particle interactions are not resolved, they are currently the only models capable of simulating industrial-scale systems [59]. Prosperetti and Tryggvason [86] provide an excellent summary of the method, along with some striking notes about its validity. In particular, the equations of motion used in many two-fluid models suffer from a lack of hyperbolicity, implying that their solution is unstable to short waves. Coupled with the result that the stability condition of the same models is independent of the wavenumber (see, e.g., [58, 85]), many two-fluid models are unstable, even for steady uniform flow. Ex-

perience suggests that this is not the case and the search for well-posed averaged equations of motion is an active area of research.

Another method for modeling particulate flows at somewhat reduced scales is the 'point-particle' method that tracks individual particles in the fluid but does not resolve the fluid-particle or particle-particle interactions [86]. This assumption is justified on the grounds that individual particles are small compared to the macroscopic flow features and, in a dilute limit, can be represented by point forces in a fluid field. Rather than determining the forces on each particle by integrating the stress exerted by the fluid on each particle's surface, the forces are parametrized based on theoretical or empirical knowledge. For example, the drag force on a spherical particle is only known analytically for very small Reynolds numbers. Although the behavior of the drag force as the Reynolds number increases is quite well studied, there is no closed-form solution and so the drag force must be parametrized. Indeed, the range of parameters (e.g., volume fraction, Reynolds number, particle-to-fluid density ratio) over which theoretical predictions are valid is quite small and therefore empirical data is inevitable with these methods. Still, much effort has been made trying to improve these models, as they may represent a compromise between physical accuracy and simulation size.

Recent improvements in computational capabilities and numerical methods have rendered possible the simulation of thousands of fully resolved particles, giving unprecedented access to detailed information on the flow in the neighborhood of each particle and on the particles response. Several of these new methods avoid body-fitted meshes in favor of regular rectilinear grids, trading the computational expense of mesh deformation and regeneration for an increase in numerical complexity associated with imposing the no-slip particle boundary conditions. Examples include the immersed boundary method, which adds a forcing term to the fluid momentum equation that enforces the no-slip condition on the particle surface, and the Physalis method, which recognizes that an analytic solution to the linearized fluid

momentum equation is valid near the particle surface and can provide accurate boundary conditions to the flow. The state of the art of immersed boundary methods for resolved particle simulations is summarized in Chapter 1.1, and a more thorough discussion of resolved particle simulation methods can be found in the recent review article by Maxey [75] The Physalis method is described in more detail in Chapter 2.

In principle, one can envisage a range of simulation techniques used to investigate different scales of motion, similar to how direct numerical simulations, large eddy simulations, and Reynolds-averaged Navier-Stokes simulations are used in turbulence. In the present case, the resolved particle simulations would be used to study the microscopic properties of the flow in order to better understand the physics underlying the phase interactions. This knowledge would be used to improve upon or create new closure relationships for use in the point-particle or two-fluid methods (see, e.g., the discussion by [84] and the recent article by [39] for an example of this in practice).

Motivated by this idea, we present in this work the results of simulations of a fluidized-bed-like solid-liquid system over a range of parameters. We describe fully resolved simulations of this type in Chapter 1.2. The simulations are performed with the Physalis method, which we summarize in Chapter 2. A complete description of the Physalis method is available in several papers including, most recently, [96].

In Chapter 3, we show how the detailed particle-level information available from the simulations can be processed to identify the presence of kinematic waves in the suspension. We compare the celerity of these waves with the theory developed in earlier averaged treatments and find a very good agreement between the two. An additional element of interest is the description of a method by which detailed, microscopic information on a complex fluid-particle system can be interrogated to extract information that concerns the average, macroscopic character of the system.

We consider in Chapter 4 the same set of simulations in more detail, drawing comparisons to prior work in overlapping parameter ranges. We study particle collisions, clustering, and the time evolution of four-particle structures, or 'tetrads'. We find that several earlier results on the two-particle distribution function, particle diffusivity, and particle velocity fluctuations extend to the parameter range considered here. Several previously identified qualitative features of the results, such as trends with increasing volume fraction, are found not to be intrinsic to the system dynamics, but dependent on the normalization used to present the results.

The aforementioned studies used an implementation of the Physalis method to simulate the fluidized-bed-like system. This implementation was originally reported in [95] and [96]. Since this legacy implementation was developed to run on a single graphics processing unit (GPU), the total domain size of the simulation (and, correspondingly, the total number of particles) was constrained by the amount of memory on a single GPU. In order to run larger simulations, we present in Chapter 5 a new implementation of the Physalis method that is able to take advantage of large GPU-enabled supercomputers.

Much of this work is adapted from original work by the author and several collaborators. Chapter 3 originally appeared as Reference [115], whereas Chapters 4 and 5 are currently under review (References [112] and [114], respectively).

Finally, we conclude by summarizing the work done and looking to the future of disperse particulate flow simulations using the many-GPU implementation. Simulations with tens or hundreds of thousands or even millions of particles are already commonplace—the only constraint is the availability of computational resources. With access to this unprecedented amount of data, analysis methods and a method of action to improve reduced-order models will become paramount. We comment on some of these issues in Chapter 6.

1.1 Resolved particle simulations methods

In general, there are two ways to apply the influence of a body embedded in a flow to the equations governing the motion of the fluid. The first is to discretize the computational domain in a body-fitted manner such that the boundary conditions on the embedded object can be applied directly on the interface. Although the boundary conditions are easy to determine, this method becomes computationally expensive when the grid needs to be regenerated, e.g. in the case where one or more bodies move in a flow field. The second method is to discretize the domain using a simple uniform structured rectilinear grid, which trades the computational expense of mesh regeneration for the challenge of imposing the boundary conditions on the body. This challenge arises because the surface of the body does not, in general, conform with the rectilinear mesh. As previously mentioned, several methods exist that overcome this geometrical mismatch. In this section, we briefly discuss the immersed boundary method, which is the method currently used by most of the cutting-edge resolved particle simulation codes. The method we use, Physalis, is described in Chapter 2; a more thorough discussion of resolved particle methods is given in [75].

The immersed boundary method applies the influence of a body embedded in the flow to a set of uniform grid nodes by determining the force locally required to satisfy the boundary conditions on the body (see, e.g., [82, 76]). This method was adapted by Uhlmann [103] for use in resolved particle simulations by tracking the surface of each particle using Lagrangian points and improved by [62] (with more accurate numerics) and [61] (with more accurate collisions and boundary condition satisfaction).

Simulations with tens or even hundreds of thousands of particles resolved using the immersed boundary method are already common in several research groups—see, for example, results from Markus Uhlmann's group at Karlsruhe Institute of Technology: [104] ($N_p \approx$ 2,500 to 20,000), [63] (N_p = 3,246 to 16,854) and [64] (N_p = 79,073 to 263,412); or from

Luca Brandt's group at KTH Royal Institute of Technology: [9] (N_p up to around 31,000), [28] (N_p = 80,000 to 640,000), [60] (N_p = 134 to 6,680), and [71] (N_p = up to 46,140). To our knowledge, the largest resolved particle simulation was performed by Kidanemariam and Uhlmann [65] with up to 1,053,648 particles on a Cartesian grid with ∼3.6 billion grid cells. These simulations allow unprecedented access to a range of physical scales, as well as providing a large number of particles to average over for statistical convergence.

The immersed boundary method has a number of advantages and disadvantages compared with the Physalis method. Unlike the Physalis method, which is only developed for spherical particles (although extensions to ellipsoidal and other particle shapes may be possible, albeit difficult), the immersed boundary method is not limited by any similar constraints on geometry and has been used to study other shapes, e.g. oblate ellipsoids in [70] and spheroidal particles in [8]. Additionally, the Physalis method relies on an analytic solution to a linearization of the Navier-Stokes equations near the particle surface; an analytic solution to the linearized equations may not exist in non-Newtonian fluids. Immersed boundary methods, on the other hand, suffer from no such limitations.

However, the immersed boundary method requires a large number of Lagrangian points (and therefore computational memory) to track the particles—around 1,000 points per particle in [62], whereas the Physalis method only requires 25 quadrature points (see the discussion on Lebedev quadrature in Chapter 2 and [96]). Additionally, since the solution used in the Physalis method is expressed as a series of spherical harmonics, the error decreases exponentially with the increase of the number of degrees of freedom used to describe each particle. This feature is in marked contrast with the algebraic error decrease of the immersed boundary method, which typically requires higher particle resolution for a similar accuracy. Furthermore, in the Physalis method, the forces and couples on the particles are given directly from the coefficients of the expansion with no need for additional calculation. Thus,

for these reasons, although the Physalis method is limited to the case of spherical particles in a Newtonian fluid, it may prove to be more computationally efficient for large simulations. Regardless, the availability of several competing tools for resolved particle simulations is beneficial to the community and provides further assurance that results obtained with both methods are accurate and reliable.

1.2 Fluidized-bed simulations

The simulations considered in Chapters 3 and 4 are performed with the single-GPU implementation of the Physalis method, described in Chapter 2 and [96, 95]. The computational domain used in the present simulations was triply periodic with a square cross section of dimension $20a \times 20a$ and a vertical extent of $60a$; all particles were assumed to have equal radius. We used eight mesh lengths per particle radius, which, on the basis of our previous experience, provides a very good accuracy in the range of Reynolds number relevant for this study. By balancing the gravitational forcing in the vertical direction with an imposed upward pressure gradient, the simulations were in effect carried out in a reference frame coincident with the mean vertical particle motion. With 500, 1000, 1500, and 2000 equal particles the mean particle volume fraction ϕ took the values 0.087, 0.175, 0.262, 0.349, respectively. We considered four different values of the particle-to-fluid density ratio, $\rho^* \equiv \rho_p/\rho_f = 2.0$, 3.3, 4.0, 5.0.

The parameters for the simulations were chosen to match experiments by Richardson & Zaki [88], who used glass spheres of radius 2.1 mm; our simulations for $\rho^* = 3.3$ correspond to the liquid with density $\rho_f = 875$ kg/m^3 and kinematic viscosity $\nu_f = 1.715 \times 10^{-5}$ m^2/s that they denote by $R_0 I$. With the exception of Young's modulus, which is softened to 0.65 MPa to avoid the very stringent time step constraint required by the use of a realistic value as explained in [96], the physical parameters used in the particle collision model (summarized

in Section 2 of this Chapter), listed in Table 1.1, were chosen to match those of glass spheres. The collision Stokes number $St_c = \rho^* Re_r/9$, with Re_r the particle Reynolds number based on the relative velocity, characterizes the strength of collisions (see, e.g., [12]). Typical values of St_c encountered in the present simulations were at most 10-20. Thus, collisions are dominated by the fluid viscosity and are too weak to result in a rebounding motion of the colliding particles (see Figure 6 in [96]). For this reason, the use of a smaller Young's modulus cannot affect the results in any significant way.

Table 1.1: Values of the parameters used in the collision model of [96]

Young's modulus, E (MPa)	0.65
Poisson ratio, σ	0.5
Dry coefficient of restitution, $e_{\rm dry}$	0.98
Coefficient of friction, μ_f	0.5

In order to characterize the balance between gravity and viscous dissipation it is convenient to use the Galilei number

$$Ga \;=\; \frac{1}{\nu}\sqrt{\left(\frac{\rho_p}{\rho_f}-1\right) d^3 g}\,, \tag{1.1}$$

in which g is the acceleration of gravity and ν the fluid kinematic viscosity; the values of Ga corresponding to the present simulations are shown in Table 1.2. By carrying out separate simulations in domains with size $20a \times 20a \times 80a$ we have calculated the terminal settling velocity w_t of single particles for the density ratios used in this study. The results are shown in the form of the single-particle Reynolds number $Re_t = dw_t/\nu$ in Table 1.2, together with the values of Re_t which satisfy the empirical relation [117]

$$Ga^2 = \begin{cases} 18Re_t[1+0.1315Re_t^{0.82-0.05\,\log_{10}Re_t}] & 0.01 < Re_t < 20 \\ 18Re_t[1+0.1935Re_t^{0.6305}] & 20 < Re_t < 260 \end{cases}\,, \tag{1.2}$$

with the Galilei numbers used in the simulations. The numerical results and the values of Re_t that satisfy this correlation are found to be in very good agreement with each other with

maximum differences of less than 3%. Similar differences were found in an earlier paper [96] and shown to be comparable with those of other recent numerical studies.

Table 1.2: Galilei number, single-particle terminal velocity w_t, and corresponding Reynolds number for the present simulations compared with an experimental result from [88] (R&Z) and simulations from [117] (Y&K).

Current Work				R&Z	Y&K
ρ_p/ρ_f	Ga	w_t m/s	Re_t	w_t m/s, Re_t	Re_t
2.0	49.7	0.177	43.27	–	44.17
3.3	75.4	0.313	76.60	0.319, 78.25	78.40
4.0	86.1	0.374	91.57	–	93.82
5.0	99.4	0.453	110.84	–	113.74

Particles were initially randomly arranged in the computational domain and, before data were recorded, allowed to reach a statistically steady state as revealed by the average values of the fluid velocity and particle velocity fluctuations. For the lower densities and volume fractions we could run the simulations used to calculate the speed of kinematic waves for dimensionless times $\nu t/d^2$ up to 24.3. However, as the density ratio and volume fraction increase, interparticle interactions become more frequent and energetic, which requires a smaller time step and more iterations for convergence. In these cases, for practical reasons, we only integrated up to $\nu t/d^2$ of about 14.2. The integration time necessary for the comparisons with the Richardson-Zaki correlation was significantly shorter (in some cases as short as $\nu t/d^2 = 0.7$) since the fluid velocity did not take much time to reach steady state. Due to computational constraints, we were unable to run simulations with larger density ratios and volume fractions for long enough to calculate the kinematic wave speed in Chapter 3 or investigate many of the properties in Chapter 4. These simulations were, however, included in the comparison to the Richardson-Zaki correlation in Chapter 3.

Chapter 2

The Physalis Method

This chapter describes the Physalis method used for the simulations previously described in Chapter 1.2. The Physalis method was developed for computationally simulating particle-resolved disperse two-phase flows. These simulations typically contain N_p spherical particles of radius a and density ρ_p moving in a fluid-filled domain while interacting with the fluid and each other through both hydrodynamic forces and direct contact. In the fluid phase, we solve the Navier-Stokes equations,

$$\partial_t \mathbf{U} + (\mathbf{U} \cdot \nabla)\,\mathbf{U} = -\frac{1}{\rho_f}\nabla p + \nu \nabla^2 \mathbf{U} + \mathbf{g}; \tag{2.1a}$$

$$\nabla \cdot \mathbf{U} = 0 \tag{2.1b}$$

for velocity $\mathbf{U}$ and pressure p with density ρ_f, kinematic viscosity ν, and an external force per unit mass $\mathbf{g}$. The fluid phase interacts with the solid phase at each particle surface, subject to a no-slip condition. We solve (2.1) on a fixed Cartesian grid, which does not conform with the spherical geometry of the particles. It is precisely this geometrical mismatch that the Physalis method is designed to overcome. We refer the reader to [96] for a precise description of the Physalis method itself, as we present a general summary of the overall method—with more detailed discussion about only the most relevant elements—in this text.

Synthetically, at each time step, the Physalis method proceeds as follows. Beginning with the solution from the previous time step, we construct a cage around each particle—the subset of discrete flow domain cells to which the particle influence is communicated—based on the particle's current position. On each particle cage we apply specially determined velocity and pressure boundary conditions before stepping to the subsequent time step by solving (2.1) using a second-order finite difference projection method on a staggered grid [18, 41].

The Physalis method determines these velocity and pressure boundary conditions, $\tilde{\mathbf{u}}$ and $\tilde{p}$ in a particle's reference frame, in such a way that the solution to (2.1) fulfills the no-slip boundary on the particle to analytical accuracy. The process for determining these internal boundary conditions comprises the heart of the Physalis method and involves the solution of a linearization of (2.1) (see, e.g., [69, 66]) in a small region local to each particle. The solution may be written in an abbreviated form as follows:

$$\tilde{\mathbf{u}}\left(t,r,\theta,\varphi\right)=\sum_{l=0}^{\infty}\sum_{m=-l}^{l}A_{lm}\left(t\right)\tilde{\mathbf{u}}_{lm}\left(r\right)Y_{l}^{m}\left(\theta,\varphi\right), \tag{2.2a}$$

$$\tilde{p}\left(t,r,\theta,\varphi\right)=\sum_{l=0}^{\infty}\sum_{m=-l}^{l}B_{lm}\left(t\right)\tilde{p}_{lm}\left(r\right)Y_{l}^{m}\left(\theta,\varphi\right), \tag{2.2b}$$

where the Y_l^m are spherical harmonics, and $\tilde{\mathbf{u}}_{lm}$ and $\tilde{p}_{lm}$ are explicitly known algebraic functions of the distance r from the particle center. Finding the time-dependent coefficients A_{lm} and B_{lm} (to which we refer as the *Lamb coefficients*) is the core of the method. These coefficients are calculated by taking a scalar product [52], which takes the form of a spherical surface integration, e.g.,

$$\left(Y_{l}^{m},\tilde{p}\right)=\int_{-\pi}^{\pi}d\varphi\int_{0}^{\pi}\sin\theta\,d\theta\,\bar{Y}_{l}^{m}\left(\theta,\varphi\right)\tilde{p}\left(t,r=R,\theta,\varphi\right), \tag{2.3}$$

where the overbar denotes complex conjugation and R is the radius of a spherical surface concentric with and slightly larger than the particle. The integration in (2.3) is performed

by Lebedev quadrature [72], which requires the interpolation of the fluid fields from the Cartesian grid to a set of 26 quadrature nodes $\boldsymbol{\psi}_i$ arranged on the integration surface in such a way that

$$\frac{1}{4\pi}\int_{-\pi}^{\pi} d\varphi \int_{0}^{\pi} \sin\theta \, d\theta \, f\left(\theta,\varphi\right) \approx \sum_{i=1}^{26} W_i f\left(\boldsymbol{\psi}_i\right), \tag{2.4}$$

where W_i are suitable weights. In practice, we retain a finite number of coefficients in the Lamb series (2.2) as the convergence is spectral; in this work, we retain 25 coefficients, which corresponds to retaining multipoles up to and including order 2.

Once the A_{lm} and B_{lm} have been determined, we apply (2.2) to the particle cage cells and solve (2.1) subject to these boundary conditions. One step of the second-order finite difference projection method that we employ requires the solution of a Poisson equation of the form

$$\nabla^2\phi = \frac{\rho_f}{\Delta t}\nabla\cdot\mathbf{U}^*, \tag{2.5}$$

where ϕ represents the pressure correction required to ensure that the solution to the first equation in (2.1) (the momentum equation) satisfies the second equation (the continuity equation); $\mathbf{U}^*$ is an intermediate velocity field; and Δt is the discrete time step size. We note this detail because of its relevance to the discussion in Chapter 5 and refer the reader to [96] for more specific information.

After solving for the fluid velocity and pressure (2.1), we determine the forces acting on each particle by the surrounding fluid and neighboring particles. The method by which we find pairs of interacting particles is described in more detail in Chapter 2.1 of this work. The hydrodynamic forces and couples are known from algebraic relations between the Lamb coefficients; for example, the hydrodynamic force on a particle are given in the particle's reference frame by

$$\hat{\mathbf{F}}_h = \pi\mu\nu(2\boldsymbol{\Phi}-\mathbf{P}) + 2\pi\mu\nu(2\boldsymbol{\Phi}+\mathbf{P}), \tag{2.6}$$

where $\mathbf{P}$ and $\boldsymbol{\Phi}$ are simple functions of the Lamb coefficients A_{lm} and B_{lm} in (2.2). The

two terms in (2.6) identify the pressure and viscous contributions, respectively. In principle, the hydrodynamic force would include the influence of nearby, but not touching, particles. However, as two particles approach each other, the number of grid nodes in the gap between the particle surfaces decreases to zero. As the gap size h approaches the grid resolution Δx, the solution of the Navier-Stokes equations fails to resolve the physics of the interaction. This feature is not limited to the Physalis method and has been discussed before in, e.g., [92]. To make up for this deficiency, we augment the hydrodynamic force with a lubrication model that captures the contribution of the fluid in the gap when $0 < h < h_{max}$; here, h_{max} is typically set to the particle radius a, but results vary little with $\frac{1}{2}a < h_{max} < a$. For two equal spheres approaching at a relative velocity $\mathbf{w}_{\alpha\beta}$, the normal component of the lubrication force is given to $O(\ln h_{max}/h)$ by

$$\mathbf{F}_n = -6\pi\mu a \left[\frac{1}{4}\left(\frac{a}{h} - \frac{a}{h_{max}}\right) + \frac{9}{40}\ln\left(\frac{h_{max}}{h}\right)\right](\mathbf{w}_{\alpha\beta}\cdot\hat{\mathbf{n}})\hat{\mathbf{n}}\ , \tag{2.7}$$

with $\hat{\mathbf{n}}$ the unit vector connecting the particle centers. Similar relations can be written for the tangential component of the lubrication force and the lubrication couple, and the same ideas can be applied to interactions between spherical particles and no-slip solid walls. More details on the short range lubrication force can be found in Section 4.1 of [96] and the references therein.

Sufficiently energetic particle interactions will result in direct particle contact that may deform the particles during the collisions ($h \leq 0$), causing an opposing contact force. However, the length and time scales involved in the deformation of the material are significantly smaller than those in the fluid [61], rendering their resolution unfeasible. Additionally, in many of the conditions for which the Physalis method is most useful (e.g., liquid-solid flows with St_c less than several hundred) the fluid viscosity plays a major role in the particle interactions. Neglecting to resolve the contact details of the solid deformation in these situations would not alter the results in a significant way. Rather, we model the particle contacts

using a non-linear damped Hertzian spring. The spring constant is determined from particle radius, Young's modulus, and Poisson ratio [102], whereas the damping coefficient is determined at the onset of each collision by the collisional Stokes number. Adjusting the damping coefficient according to St_c allows the model to accurately capture the expected wet coefficient of restitution, which is primarily a function of St_c (see, e.g., [29])

The hydrodynamic and particle interaction forces, together with the gravitational force and any additional external forces, are used to compute the particle acceleration and update the particle velocity. Ideally, this information could now be used to update the particle positions and proceed to the next time step. However, we have solved the Navier-Stokes equations (2.1) according to the particle boundary conditions (2.2), which themselves were determined from the solution to (2.1) at the previous time step. Therefore, the flow solution and boundary conditions are not expected to match. To deal with this, we iterate (within a time step) between solving (2.1), updating the boundary conditions (2.2), and determining the forces on the particle until the flow solution and the boundary conditions converge. These iterations, which we term *Lamb iterations*, are typically required between one and twenty times for each time step depending on the nature of the flow. Once converged, we have completed the Physalis method for one time step and move each particle according to the forces acting upon them. The entire process is then repeated for the next time step.

2.1 All-pairs interactions

In order to account for the lubrication and collision interaction forces, we must perform an all-pairs interaction search over all of the particles. We take advantage of the compact support of the particle-to-particle lubrication and contact effects to spatially limit the interaction search to only the nearest neighbors of each particle. To do so, we introduce a coarser Cartesian grid with a mesh length equal to the particle diameter augmented by the lubrication interaction

cutoff distance. Each particle is assigned to a cell (or 'bin') of this grid based on the position of its center. By considering the bin occupied by a given particle along with the 26 bins surrounding it, it is possible to identify all the particles that may directly (i.e., by collision or lubrication forces) interact with the given particle. Operationally, this device reduces the cost of the all-pairs interaction search from N_p^2 to $N_p \log(N_p)$ and also provides a convenient mapping from particles in physical space to particles stored in memory. This procedure is relevant to the discussion in Chapter 5; [95] and [50] provide more detailed information.

Chapter 3

Continuity Waves

Kinematic (or continuity, or density) waves are among the simplest form of nonlinear waves [111] and are widely encountered in many areas such as traffic flow, avalanches, suspensions, and many others. The denomination arises from the fact that the flux of some conserved quantity is expressed directly in terms of that quantity, thus bypassing the need for the consideration of dynamic effects. In these conditions the statement of continuity, complemented by suitable boundary and initial conditions, is sufficient to completely determine the evolution of the conserved quantity in space and time.

The efficiency of fluidized beds and other types of disperse flow systems would be higher were it not for the fact that the particle-fluid mixture does not remain homogeneous. Rising regions of low particle density, referred to as "bubbles", very frequently form, particularly when there is a large density difference between the particles and the fluid, such as in a gas-solid fluidized bed. The precise mechanism producing these structures has been debated for decades, but firm conclusions resulting from these efforts are disappointingly few. A widely held belief is that bubbles form as the result of a cascade of instabilities, the first one of which affects the initial uniform state of fluidization which loses stability to vertically propagating density waves [6, 7, 13, 14, 35, 36, 38, 48, 46, 47, 99]. This is one of the reasons of interest in the study of kinematic waves in disperse fluid-particle systems. Besides fluidized beds,

other particulate systems display density waves, as first recognized by Kynch in the study of settling suspensions, and further elaborated in later studies by others [67, 97, 108].

Nearly all existing theoretical studies of kinematic waves in particulate systems are based on coarse-grained descriptions in which the particles are modeled in an average sense. The only exception is [32] in which lattice-Boltzmann simulations for conditions close to those of the experiments of [37] are described. In order to limit the complexity of the horizontal structure of the voidage waves the authors used domains with a very small horizontal extent, mostly $12a \times 12a$, with a the particle radius, finding a generally good match with the experiments of [36, 37]. They noticed the presence of voidage waves but did not attempt to interpret them as kinematic waves.

In this chapter we show how the results of the fully resolved simulations of a fluidized-bed-like solid-liquid system described in Chapter 1.2 can be analyzed so as to bring out the existence of kinematic waves and calculate their velocity. It is found that the velocity calculated in this way is in very good agreement with existing estimates based on a macroscopic, averaged formulation of fluidized beds [108]. An additional element of interest of this work is the description of a method by which detailed, microscopic information on a complex fluid-particle system can be interrogated to extract information that concerns the average, macroscopic character of the system.

3.1 Kinematic Waves

When the particle density is not too different from that of the fluid, in steady conditions dynamical effects are minor and the particle velocity is mostly determined by the hindrance that they provide to each other's motion and to the motion of the fluid in the interstitial spaces. In these conditions the primary determinant of the particle-fluid relative velocity is the particle volume fraction as has been known since the early days of studies on this subject.

A one-dimensional balance equation for the particle volume fraction may be written in the form

$$\frac{\partial \phi}{\partial t} + \frac{\partial j_p}{\partial z} = 0\,, \tag{3.1}$$

where j_p is the flux of the volume fraction ϕ and z the vertical coordinate directed upward; t is time. In the conditions envisaged here the particle flux can be expressed in terms of the volume fraction ϕ. Application of the chain rule to the second term of (3.1) then results in a first-order wave equation in which the speed c with which information about volume fraction changes propagates is given by $c = dj_p/d\phi$.

3.1.1 Richardson-Zaki Correlation

The particle flux j_p can be expressed as the sum of a component ϕj, describing the fact that the particles travel with the mixture volume flux j, and a drift-flux component j_d, which corrects for the difference between j and the actual particle flux. If the particles and fluid mean velocities are denoted by $\langle w_z \rangle$ and $\langle u_z \rangle$, we have $j = \phi\langle w_z \rangle + (1-\phi)\langle u_z \rangle$, and the particle flux relative to j is $j_d = \phi(j - \langle w_z \rangle) = \phi(1-\phi)(\langle w_z \rangle - \langle u_z \rangle)$. A well-known empirical relationship for $\langle w_z \rangle - \langle u_z \rangle$ was developed by Richardson & Zaki [88] in the form

$$\frac{\langle u_z \rangle - \langle w_z \rangle}{w_t} = \kappa(1-\phi)^{n-1}\,, \tag{3.2}$$

where the exponent n depends on the single-particle Reynolds number; a correlation describing this dependence is $(5.1-n)/(n-2.7) = 0.1 Re_t^{0.9}$ [44]. The parameter κ in (3.2) is a slowly decreasing function of Re_t and is somewhat less than 1 [117, 24, 34, 33, 11]. This circumstance reflects the fact that the mutual interference among the particles, mostly due to their slowly decaying wakes, persists even in very dilute systems; a similar effect is found in the case of rising bubble swarms [107, 120].

We can calculate $\langle u_z \rangle$ and $\langle w_z \rangle$ from our numerical results by carrying out volume and time averages over the entire computational domain and duration of the simulations and

Re_t	n	κ
43.27	3.25 ± 0.15	0.87 ± 0.04
76.60	3.03 ± 0.11	0.85 ± 0.03
91.57	2.94 ± 0.11	0.84 ± 0.03
110.84	2.94 ± 0.08	0.85 ± 0.02

Table 3.1: Fitted parameters κ and n from the current simulations with 95% confidence intervals.

fit a relation of the form (3.2) to the results, thus determining values for κ and n. The computed values are given in Table 3.1 and shown in graphical form in Figure 3.1, where they are compared with the results of [117] for κ and [88] (solid line) and [44] (dashed line) for n. Figure 3.2 compares the present simulation results for the mean liquid-particle relative velocity (symbols) with the Richardson-Zaki curve (3.2) (lines) calculated with the parameter values derived from the present simulations. These comparisons are very favorable, which provides additional confidence in the present numerical results.

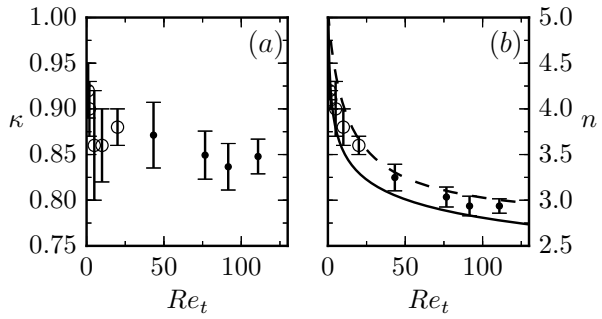


Figure 3.1: Comparison of the power-law fit coefficients n and κ in (3.2). Black dots are from this study, white circles are from [117], and the solid and dashed lines are relations for n from [88] and [44], respectively.

Upon making use of (3.2) in the expression for the particle volume flux we find

$$j_p \;=\; \phi j - \kappa w_t \phi (1-\phi)^n \,, \tag{3.3}$$

from which, upon differentiation with respect to ϕ and recognizing the fact that the total flux j is a constant (as follows immediately by adding (3.1) to its counterpart written for

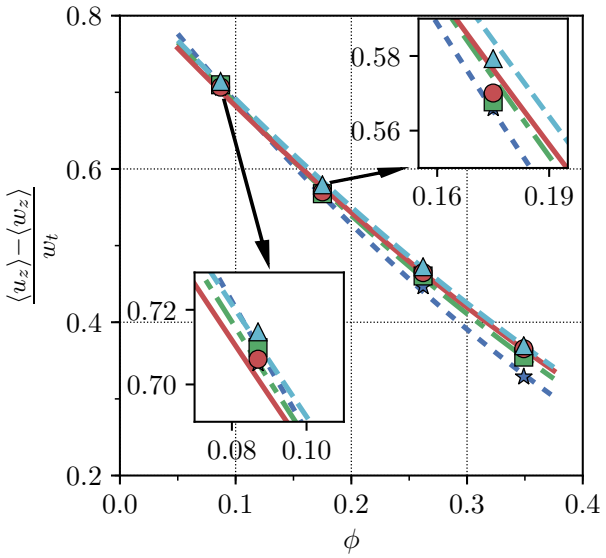


Figure 3.2: Comparison of the present simulation results for the mean fluid-particle relative velocity (symbols) with the Richardson-Zaki curve (3.2) (lines). The dotted line and stars are for $\rho_p/\rho_f = 2$; the dash-dotted line and green squares for $\rho_p/\rho_f = 3.3$; the solid line and red circles for $\rho_p/\rho_f = 4$; and the dashed line and light blue triangles for $\rho_p/\rho_f = 5$. Insets: top right: close-up of $\phi \approx 0.175$; bottom left: close-up of $\phi \approx 0.087$.

the liquid volume fraction $1 - \phi$), we find

$$c = j - \kappa w_t (1-\phi)^{n-1} \left[1 - (n+1)\phi\right] . \tag{3.4}$$

In the situation considered here the mean particle velocity vanishes so that $j = (1-\phi)\langle u_z \rangle$. Use of (3.2) then gives the well-known result [see e.g. 108, p. 189]

$$c = \kappa n \phi (1-\phi)^{n-1} w_t . \tag{3.5}$$

A point worthy of explicit notice is that, according to kinematic wave theory, the waves are nondispersive. Dispersive effects may exist of course, but they will be small in situations in which kinematic wave theory is approximately applicable. A quantification of dispersion would require a coupling of the continuity and momentum equations and, therefore, a more sophisticated theory. Another feature of this type of wave is that they are inherently nonlinear, and therefore many techniques commonly used for linear waves are not available.

3.2 Kinematic wave speed

In a system seat of waves, all the fields will exhibit a wave structure that propagates with the same velocity. In order to determine this velocity we make use of the autocorrelation of the fields on the basis of the following argument. Consider a generic field f, such as the particle number density, volume fraction or other. Since the system under consideration is statistically uniform over horizontal planes we will consider fields averaged over the horizontal variables x and y and dependent only on the vertical coordinate z and time.

A pure wave propagating in the z direction would confer to the field f a space and time dependence of the form $f(z,t) = f(z - ct)$. The space-time autocorrelation is $f(z + \Delta z, t + \Delta t)\, f(z,t) = f(z + \Delta z - c(t + \Delta t))f(z - ct)$ and, for $\Delta z = c\Delta t$, it reduces to $f^2(z,t)$ and exhibits therefore a maximum. In the present system the waves are contaminated by the randomness of the particle distribution. In order to bring out this maximum with greater clarity we average the autocorrelation over z, since the system is statistically homogeneous in the vertical direction, and over time, since we consider only the numerical results at steady state after the initial transient has faded away. Thus we focus on quantities of the type

$$R_{ff}(\Delta z, \Delta t) \;=\; \langle f(z + \Delta z, t + \Delta t)\, f(z,t) \rangle \,, \tag{3.6}$$

where angle brackets denote the space-time average, and, for brevity, we write f in place of $(f - \langle f \rangle)/\sqrt{\sigma_f}$, with σ_f the variance of f so that, in fact, R_{ff} denotes the autocorrelation function of f. In the presence of waves this quantity will exhibit a series of maxima along lines $\Delta z = c(\Delta t + NT)$ with T the wave period, $N = 0$ for the first wave, $N = 1$ for the second one, and so on. From the slope of these lines in a Δt, Δz plane we can then determine the wave speed c. Again, because of the statistical irregularities of the system, we might well expect that the line of maxima will be strongest for the first wave and gradually decay as the value of f at a certain position z_0 cannot be expected to be very strongly correlated to

the value of f at the same position several waves later. Similarly, the value of f at a certain time t_0 cannot be expected to be strongly correlated to its value at the same instant several wavelengths away.

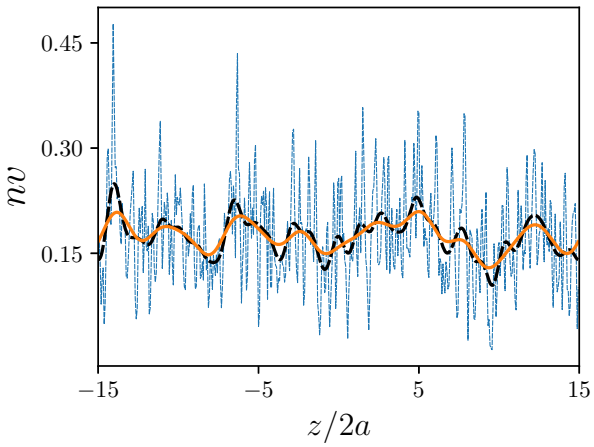


Figure 3.3: The thin dashed line is a snapshot of the "raw" number density, i.e., the number density calculated by counting the particle centers contained in each horizontal layer of cells, for $\rho_p/\rho_f = 3.3$ with 1000. The solid line and the thick dashed line are the number density reconstructed by a Fourier series with 15 and 30 coefficients, respectively.

With the present numerical data, a naive way to implement this approach is to calculate averages of the field of interest over horizontal layers of cells thus generating a discretized version of $f(z,t)$; we will use the adjective "raw" to refer to quantities obtained in this way. As an example, a snapshot of the "raw" particle number density n so calculated is shown by the thin dashed line in Figure 3.3 for the case $\rho_p/\rho_f = 3.3$ with 1000 particles (the quantity shown is vn, the particle number density normalized by the sphere volume $v = \frac{4}{3}\pi a^3$). Each data point is calculated by counting the particle centers contained in a single horizontal layer of cells and dividing by the volume of the layer. It is evident that, even after this horizontal averaging, the result is affected by a considerable amount of noise. Figure 3.4 shows the autocorrelation function R_{nn} for these parameter values calculated as explained before by averaging over z and t. While approximately parallel lines of maxima and minima

are vaguely suggested by this figure, a strong signal can be identified only for the first wave and only for very small values of Δz and Δt. It would be very difficult to deduce a precise estimation of the wave speed from results of this type.

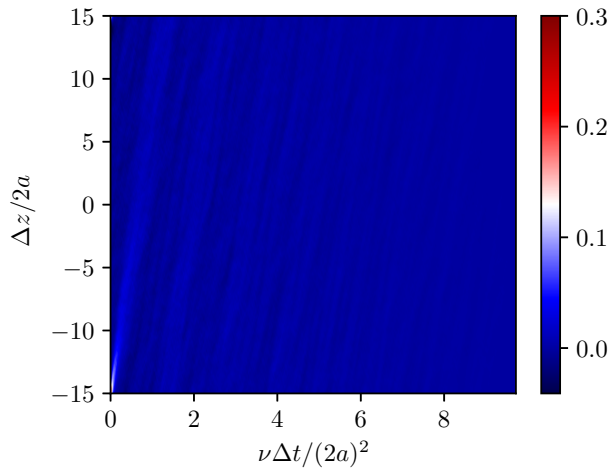


Figure 3.4: Autocorrelation function (averaged over horizontal planes and z and t) of the "raw" particle number density, i.e., the number density calculated by counting the particle centers contained in each horizontal layer of cells, for $\rho_p/\rho_f = 3.3$ with 1000 particles.

The situation is very similar if we try to use other fields. In principle the particle volume fraction can be calculated by counting the fraction of the volume of each horizontal layer of cells falling inside particles. A snapshot of the particle volume fraction obtained in this way versus z similar to that of Figure 3.3 is shown by the dashed line in Figure 3.5. A number of peaks with a width of about 1 can be discerned which, given the normalization in terms of the particle diameter, evidently correspond to fluctuations on the scale of single particles. The autocorrelation function of the volume fraction so calculated (averaged over z and t) is shown in Figure 3.6. Once again we can distinguish a series of inclined line-like features, but we encounter the same problem as before if we are interested in an accurate determination of c.

The use of additional short-time or short-space averaging might perhaps reduce the influence of statistical fluctuations, but it would make it more difficult to discern the space-time

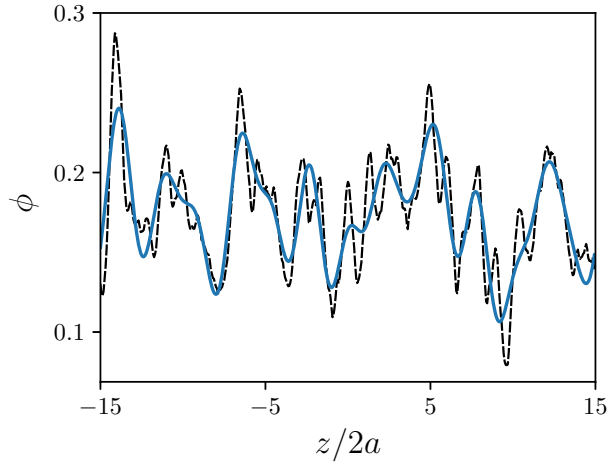


Figure 3.5: The dashed line is the "raw" particle volume fraction, i.e., the volume fraction calculated as the fraction of volume of each horizontal cell layer falling inside particles for $\rho_p/\rho_f = 3.3$ with 1000 particles. The solid line is the particle volume fraction reconstructed by a Fourier series with 15 coefficients.

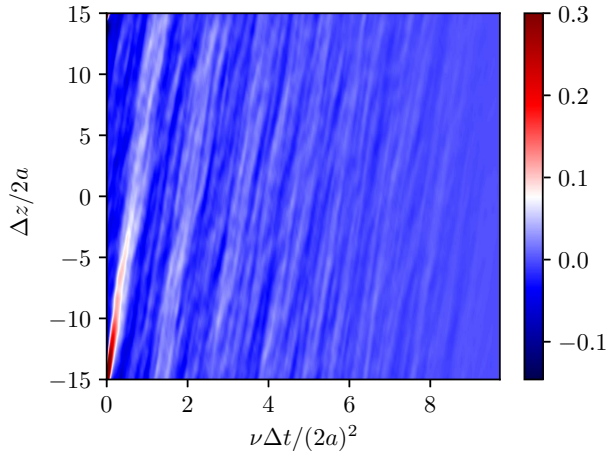


Figure 3.6: Autocorrelation of the "raw" particle volume fraction ϕ (i.e., the particle volume fraction obtained by counting the fraction of volume of each horizontal cell layer falling inside particles) for $\rho_p/\rho_f = 3.3$ with 1000 particles; the quantity shown is averaged over z and t.

variability associated with the waves by blurring their distinct spatio-temporal structure. The challenge facing this procedure is illustrated by the space-time representation of the "raw" particle volume fraction shown in Figure 3.7. The range of values represented in the figure is very limited, and averaging to eliminate noise runs the serious risk of destroying much of the signal as well. This danger would be even greater at larger volume fractions, in which the fluctuations around the mean would be even smaller.

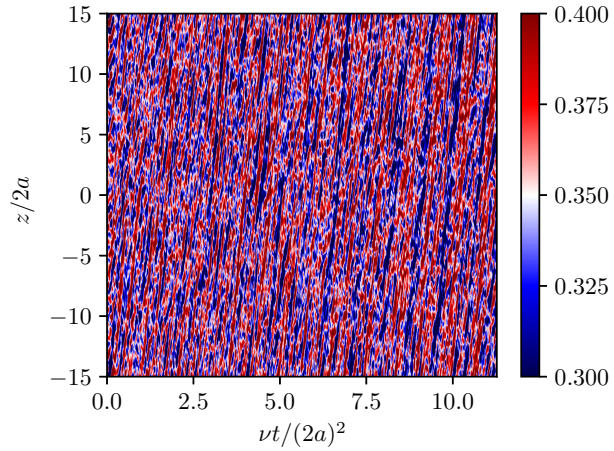


Figure 3.7: Space-time distribution of the "raw" particle volume fraction (i.e., the particle volume fraction obtained by counting the fraction of volume of each horizontal cell layer falling inside particles) for $\rho_p/\rho_f = 3.3$ with 2000 particles.

These difficulties have prompted us to develop a different way to interrogate the results of our simulations by relying on the use of a truncated Fourier series to coarse-grain the fields obtained from the resolved simulations.

3.3 Fourier Reconstruction

The numerical simulations furnish what may be called "microscopic" information on the various quantities characterizing the process under consideration. For the particle number

density such microscopic information is embodied in a field n_{micr} defined by

$$n_{micr}(\mathbf{x},t) \;=\; \sum_{\alpha=1}^{N_p} \delta(\mathbf{x}-\mathbf{x}^{\alpha}(t))\,, \tag{3.7}$$

where $\mathbf{x}^{\alpha}(t)$ is the instantaneous position of the center of the αth particle and N_p the total number of particles. This quantity can be expanded in a Fourier series as

$$n_{micr}(\mathbf{x},t) \;=\; \sum_{\mathbf{k}} n(\mathbf{k},t)\exp(i\mathbf{k}\cdot\mathbf{x})\,, \tag{3.8}$$

where the summation ranges over all the wave vectors $\mathbf{k} = 2\pi(n/L_x, m/L_y, \ell/L_z)$ in the reciprocal lattice; $L_x,\ L_y$, and L_z are the dimensions of the computational domain in the three coordinate directions and $\ell,\ m$, and n integers. The Fourier coefficients $n(\mathbf{k},t)$ are given by the scalar products

$$n(\mathbf{k},t) \;=\; \frac{1}{V}\Big(\ \exp(i\mathbf{k}\cdot\mathbf{x}), n_{micr}\Big) \;=\; \frac{1}{V}\sum_{\alpha=1}^{N_p}\exp(-i\mathbf{k}\cdot\mathbf{x}^{\alpha}(t))\,, \tag{3.9}$$

with $V = L_xL_yL_z$ the volume of the computational domain.

Retention of the infinite number of terms in the Fourier expansion (3.8) would reproduce n_{micr}, but a suitable truncation of the series will generate a coarse-grained version of n_{micr}. Furthermore, retaining only the terms of the series with wave number vectors $\mathbf{k}$ parallel to the z axis (i.e., retaining only the zeroth term of the x and y components of the wave vectors) is equivalent to averaging over horizontal planes. On the basis of these considerations we define the coarse-grained number density as

$$n(z,t) \;=\; \sum_{\ell=-N}^{N} n_{\ell}(t)\exp\left(ik_{\ell}z\right)\,, \tag{3.10}$$

with $k_{\ell} = 2\pi\ell/L_z$ and

$$n_{\ell}(t) \;=\; \frac{1}{V}\sum_{\alpha=1}^{N_p}\exp(-ik_{\ell}z^{\alpha}(t))\,. \tag{3.11}$$

A suitable value for N can be estimated by recognizing that the smallest features of the coarse-grained number density field n that it makes sense to consider in a macroscopic

framework should not be so small as to permit the identification of single particles. If we choose this shortest wavelength to be two particle diameters, we have $N = L_z/(4a) = 15$, and we find the result shown by the solid line in Figure 3.3. While this is just an estimate, we have found that the results are not very different if this number is increased up to 30 or decreased to 10, which would amount to include wavelengths as short as the particle diameter or as long as three diameters, respectively. As an example, the thick dashed line in Figure 3.3 shows the particle number density reconstructed with 30 terms. Some more detail can be identified in this line, but the 15-term reconstruction overall reproduces well the large-scale (and, therefore, properly macroscopic) features of the number density.

We can deal in a similar way with any other quantity associated with the particles. In particular, the microscopic version of the volume fraction, ϕ_{micr}, equal to one inside the particles and zero in the fluid, can be expressed as

$$\phi_{micr}(\mathbf{x},t) \;=\; \sum_{\alpha=1}^{N_p} H\left(a - |\mathbf{x} - \mathbf{x}_\alpha(t)|\right) \,, \tag{3.12}$$

with H the Heaviside step function. Its Fourier-series expansion is

$$\phi_{micr}(\mathbf{x},t) \;=\; \sum_{\mathbf{k}} \phi(\mathbf{k},t) \exp(i\mathbf{k}\cdot\mathbf{x}) \,, \tag{3.13}$$

with

$$\begin{aligned} \phi(\mathbf{k},t) \;&=\; \frac{1}{V}\Big(\, \exp(i\mathbf{k}\cdot\mathbf{x}), \phi_{micr}\Big) \;=\; \frac{1}{V}\sum_{\alpha=1}^{N_p} \int_{v^\alpha} \exp(-i\mathbf{k}\cdot\mathbf{x}) H\left(a - |\mathbf{x} - \mathbf{x}_\alpha(t)|\right) dv^\alpha \\ &=\; \frac{4\pi}{k^3}\left(\sin ka - ka\, \cos ka\right) n(\mathbf{k},t) \,. \end{aligned} \tag{3.14}$$

Here each integral in the first line is extended over the volume v^α of the αth particle and $k = |\mathbf{k}|$. It is worth noting that the last step shown here is a direct consequence of the expression for the Fourier coefficients and could not be obtained in a volume averaging context.

Again, we obtain a horizontally averaged coarse-grained volume fraction field by considering only the coefficients of order 0 in the horizontal directions and truncating the sum:

$$\phi(z,t) \;=\; \sum_{\ell=-N}^{N} \phi_\ell(t)\exp(ik_\ell z)\,. \tag{3.15}$$

This is the coarse-grained version of ϕ_{micr} which we identify with the ϕ appearing in the macroscopic theory. Since ϕ_{micr} is highly discontinuous, in order to avoid possible convergence problems caused by the Gibbs phenomenon, the sum in (3.15) is calculated according to the Cesáro summation method [20].

An example of the spatio-temporal representation of the field ϕ obtained in this way is shown in Figure 3.8. In spite of the statistical noise, the wave structure of the particle distribution is much clearer than in Figure 3.7.

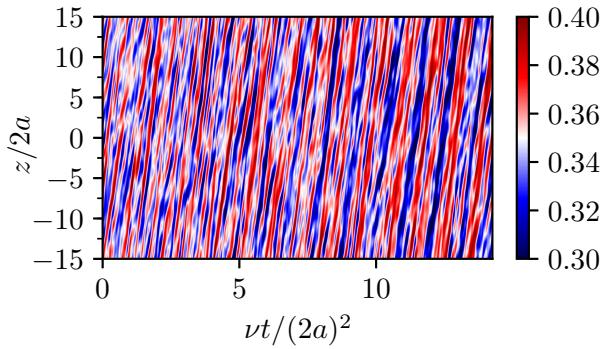


Figure 3.8: Fifteen-terms Fourier reconstruction of the volume fraction for a representative case with $\phi = 0.349$ ($N = 2000$) and $\rho_p/\rho_f = 3.3$; white corresponds to the mean volume fraction over the entire computational domain. This figure should be compared with Figure 3.7 in which the "raw", rather than the Fourier-processed, volume fraction is shown.

3.3.1 The speed of kinematic waves

The autocorrelation of the particle volume fraction reconstructed with 15 Fourier coefficients and averaged over z and t is shown in Figure 3.9. The wave structure of the particle distribution is strikingly clearer than in Figs. 3.6 and 3.4. In order to calculate the wave

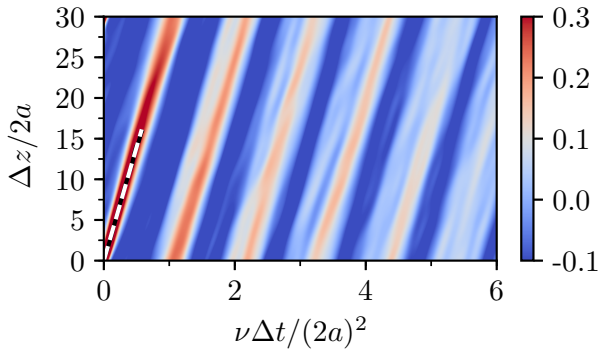


Figure 3.9: Autocorrelation of the particle volume fraction reconstructed with 15 Fourier terms for a representative case with $\phi = 0.349$ ($N = 2000$) and $\rho_p/\rho_f = 3.3$. The dashed line is obtained by a least-squares fit of the position of the maxima for each Δz. This figure may be compared with Figs. 3.6 and 3.4 which show the autocorrelation of the "raw" volume fraction and number density, respectively.

speed from these results, for each value of Δz we find the Δt corresponding to the first maximum, form the ratio $\Delta z/\Delta t$ and then average the results obtained in this way. The standard deviation of the distribution of values of $\Delta z/\Delta t$ about this average permits us to estimate the error of this determination. The results obtained in this way are shown in Figure 3.10. The lines shown in the figure are graphs of the average-equation result (3.5) constructed with the values of w_t, n, and κ calculated in this work and shown in Tables 1.2 and 3.1.

Overall, the agreement between (3.5) and the numerical results is very good. Accounting for the confidence levels of the fits shown in Figure 3.1 brings the lines within the expected range of the computational results for all but the lowest volume fraction points, where a discrepancy of less than 5% remains. A possible explanation is that, at low volume fractions, the mean-free path of the particles is long enough that dynamic effects become significant and the kinematic wave model therefore less applicable. For the larger density ratios and volume fractions the completion of all the necessary simulations would have required more time than seemed warranted by the very good agreement between macroscopic theory and simulations

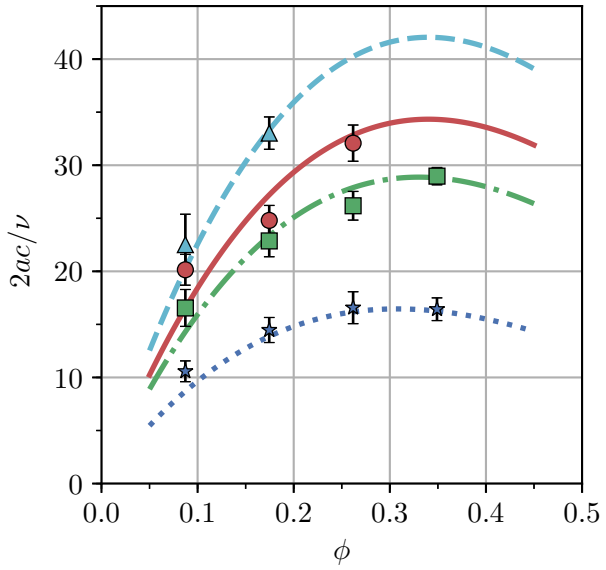


Figure 3.10: Continuity wave speed calculated from the present simulations (symbols) compared to the relationship given in (3.5) shown by the dashed lines.

displayed in the figure; we have chosen to proceed without including these results.

3.3.2 The fluid velocity field

A similar procedure can be adopted for the fields of the continuous phase. We consider the fluid velocity field in the vertical direction, $u_z(z,t)$ and write its coarse-grained version by truncating its Fourier series expansion:

$$\phi(z,t)u_z(z,t) \;=\; \sum_{\ell=-N}^{N} u_\ell(t)\exp(ik_\ell z)\,, \tag{3.16}$$

with

$$u_\ell(t) \;=\; \frac{1}{V}\int_V u_z(\mathbf{x},t)\chi_f(\mathbf{x},t)\exp(ik_\ell z)dV\,, \tag{3.17}$$

where $\chi_f(\mathbf{x},t)$ is the indicator function of the fluid phase and the integral is over the entire volume of the computational domain. A useful feature of this approach is that the volume occupied by the particles is excluded in a natural way.

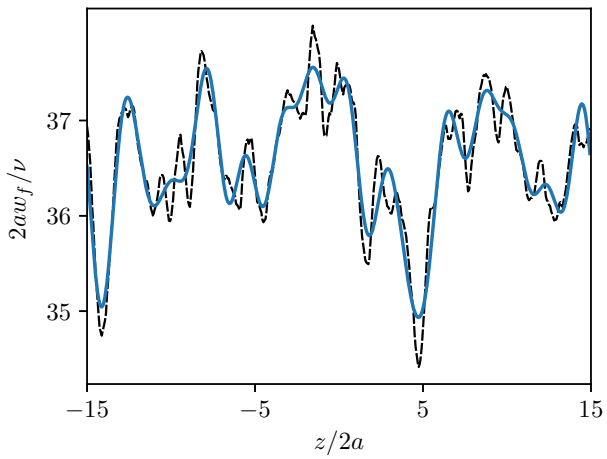


Figure 3.11: Instantaneous fluid vertical velocity vs. z as calculated from the Fourier reconstruction with $N = 15$ coefficients (solid line) and by averaging over horizontal cell layers for $\rho_p/\rho_f = 3.3$ with 1000 particles.

The relation between the instantaneous velocities calculated with this truncated Fourier series reconstruction with $N = 15$ and by averaging over horizontal cell layers is shown by the solid and dashed lines, respectively, in Figure 3.11. The effectiveness of the Fourier method to remove noise is particularly evident here.

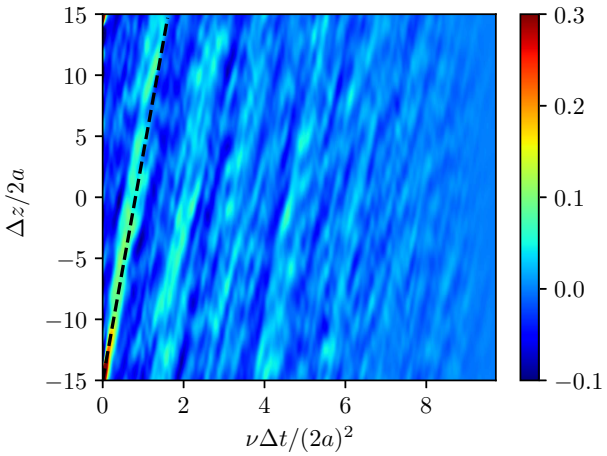


Figure 3.12: Autocorrelation function of the "raw" fluid vertical velocity, i.e., the fluid vertical velocity averaged over each horizontal cell layer; the quantity shown is averaged over z and t for $\rho_p/\rho_f = 3.3$ and 1000 particles.

The presence of kinematic waves can be detected in the fluid velocity autocorrelation in the same way demonstrated before. An example is shown in Figure 3.12. The picture is somewhat fuzzier than for the volume fraction, but the dashed line with the slope as calculated by a least square method as explained before provides a good fit to the velocity maxima. The dimensionless slope corresponding to this line is $2ac/\nu = 21.37$, which differs by about 7% from the value 22.86 shown in Figure 3.10.

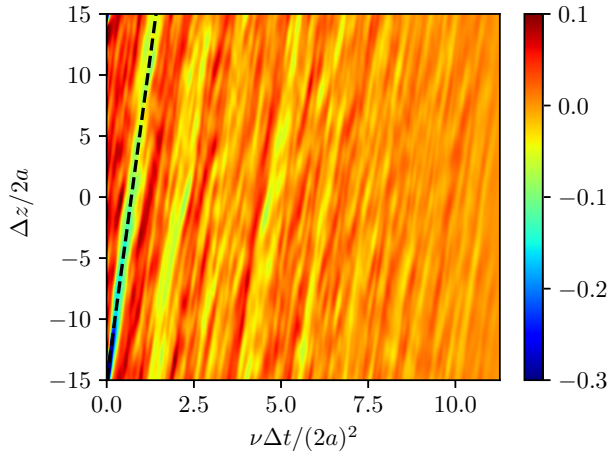


Figure 3.13: Normalized cross-correlation of volume fraction and fluid vertical velocity, each reconstructed with 15 Fourier modes, for the case $\rho_p/\rho_f = 3.3$ with 1000 particles. The slope of the dashed line is the speed of kinematic waves calculated from the reconstructed volume fraction as explained in the text.

On intuitive grounds it may be expected that fluid velocity and particle volume fraction would be oppositely correlated so that a graph of the cross-correlation function

$$R_{u\phi}(\Delta z, \Delta t) \;=\; \frac{\langle [u_z(z+\Delta z, t+\Delta t) - \langle u_z\rangle]\; [\phi(z,t) - \langle\phi\rangle]\rangle}{\sqrt{\sigma_u \sigma_\phi}}\,, \tag{3.18}$$

should exhibit a pronounced negative minimum along lines $\Delta z = c(\Delta t + NT)$. This expectation is indeed borne out by the results of Figure 3.13, where the line has the slope $2ac/\nu = 21.37$.

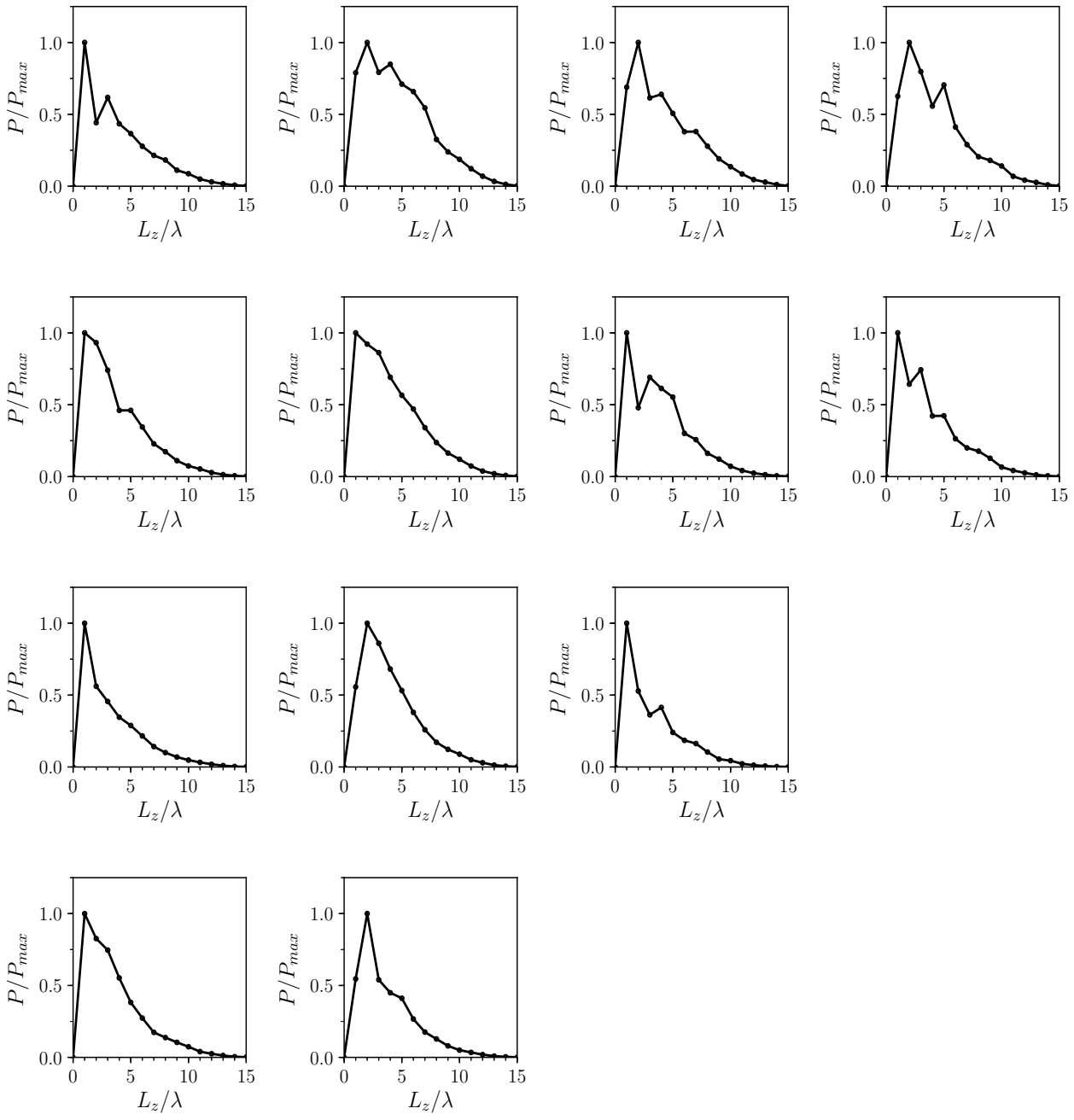


Figure 3.14: Spatial power spectra of the Fourier-reconstructed volume fraction; λ denotes the wavelength of the Fourier components. From left to right, the columns correspond to ϕ = 0.087, 0.175, 0.262, 0.349; from top to bottom the rows are for ρ_p/ρ_f = 2, 3.3, 4 and 5.

3.3.3 Power spectra

The squares of the Fourier coefficients plotted as functions of the wave number give directly the power spectra of the waves. These spectra, calculated from the Fourier-reconstructed volume fraction, are shown in Figure 3.14 for all the parameter values simulated in this work as functions of L_z/λ_ℓ, with $\lambda_\ell = 2\pi/k_\ell$. In each case the spectra are normalized by the maximum value.

Several spectra exhibit a dominant peak for a wavelength equal to half the height of the computational domain, while in other cases the dominant peak is for a wavelength equal to the height of the domain. This is an indication that, in these latter cases, the computational domain used is too short to include the entire wave. Nevertheless, since the wave speed is independent of the wavelength as noted after (3.5), the present results for the wave speed are unaffected by this limitation. Indeed, upon comparing with the graph of the wave speeds in Figure 3.10, it is seen that velocities corresponding to cases with peaks at $\lambda = L_z$ and $\lambda = \frac{1}{2}L_z$ exhibit a comparable agreement with the kinematic wave theory.

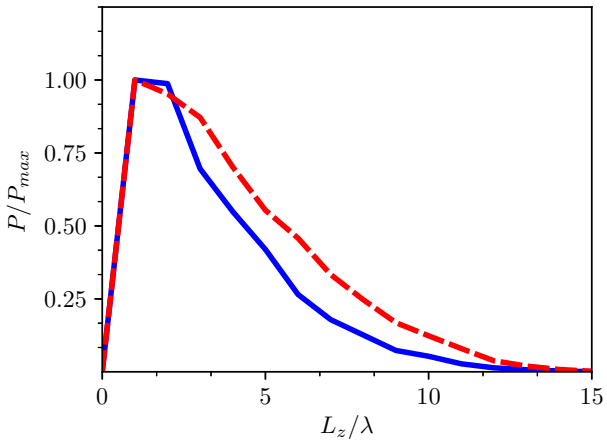


Figure 3.15: Spatial power spectra of the particle volume fraction (solid line) and of the fluid vertical velocity, both reconstructed with 15 Fourier coefficients, vs the normalized wave number for $\rho_p/\rho_f = 3.3$ with 1000 particles.

A comparison between the spectra calculated from the Fourier-reconstructed fluid velocity

and volume fraction is shown for one case in Figure 3.15. The two spectra are quite similar as expected.

The temporal spectra can be found by expanding the Fourier-reconstructed signal (e.g., the volume fraction) in a Fourier series in time at each spatial point and then averaging over space. An example of the results obtained in this way is shown in Figure 3.16 for $\rho_p/\rho_f = 3.3$ and 1000 particles. The two nearly superposed lines are the results found from the reconstructed volume fraction and vertical fluid velocity. There is a prominent peak at $f_* \equiv (2a)^2 f/\nu \simeq 0.71$. Forming the product $(\lambda/2a)f_* = 2ac/\nu$ with $\lambda = L_z$ we find $2ac/\nu \simeq 21.3$, to be compared with the value 22.86 from Figure 3.10. The difference between the two values is likely due to the need to omit, in the calculation of the spectrum, data corresponding to the initial transient, the end of which is somewhat ill-defined. We have found that the position of the peak frequency moves slightly as the fraction of omitted data is changed. Figure 3.16 also exhibits another strong peak at the first harmonic $f_* \simeq 1.5$, another much weaker one at the second harmonic $f_* \simeq 2$ and other smaller ones as well.

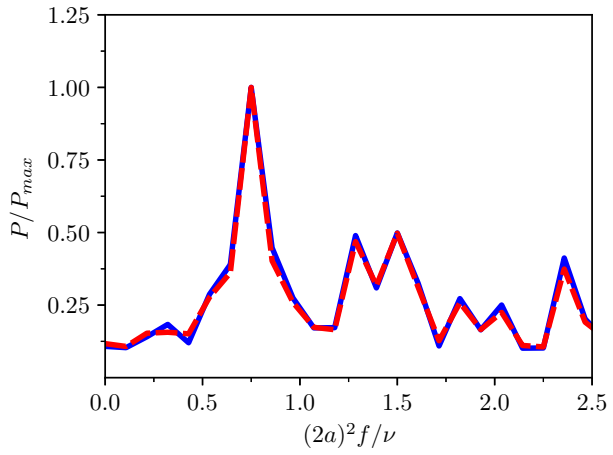


Figure 3.16: Temporal power spectra of the particle volume fraction (solid line) and of the fluid vertical velocity, both reconstructed with 15 Fourier coefficients, vs. the normalized frequency, for $\rho_p/\rho_f = 3.3$ with 1000 particles.

3.4 Discussion

In this Chapter, we demonstrated a method to extract average properties from the results of direct numerical simulations of particulate flows. We used the method to reconstruct the coarse-grained volume fraction and fluid velocity field in the fully resolved simulation of a fluidized-bed-like system in which equal spheres are suspended in an upward flow of a fluid with comparable, but smaller, density. In particular, we have shown that the coarse-grained fields show the presence of kinematic waves propagating upward in the bed. The celerity of these waves is very close to that produced by existing macroscopic description of such systems.

To compare these results to prior work, we consider the work of Duru et al. [37], who describe a study of volume fraction waves forced by the oscillations of the distributor at the bottom of a liquid-filled tube for particles with a density ratio comparable to the present one and a Galileo number slightly larger than ours. In the volume fraction range of our simulations they characterize their observations as being a "turbulent regime" and were unable to identify clear wavelike structures by their method. A visual inspection of the particle motion computed in this work (see the video uploaded with the Supplemental Material [2] in [115], which refers to 2000 particles with $\rho^* = 3.3$) confirms this disorderly appearance. Nevertheless, in spite of their convoluted appearance, the volume fraction isosurfaces given by the filtered three-dimensional Fourier reconstruction, shown on the right side of the video, convey the clear impression of upward moving structures. These waves emerge therefore as a very robust feature of the dynamics of the system investigated in spite of the complex horizontal structure of the numerical results as well as, in all likelihood, of experiment.

Chapter 4

Sedimenting Suspensions

The importance of fluids with suspended settling particles in natural and engineering systems such as sedimentation, fluidized beds, sediment transport and many others has motivated numerous experimental, theoretical and computational studies. Theoretical progress is hampered by the inherent great complexity of the phenomenon, especially when effects due to a finite particle Reynolds number become significant. In this situation, particle-resolved numerical simulations and a detailed analysis of the results thereby obtained offer hope to gain the insight necessary for progress in the modeling of such systems.

Table 4.1 summarizes the parameter range covered by several experimental and numerical studies of sedimenting particles. The earlier papers mainly focused on the Stokes regime. The advent of new numerical methods and more powerful computers has opened the way to the study of Reynolds number effects. By and large, in the parameter range where studies overlap, findings have been consistent. For example, the volume fraction dependence of the average settling speed has been found to be well correlated by the Richardson-Zaki expression [88, 44] modified by a pre-factor as suggested in [117]. A larger velocity fluctuation amplitude in the vertical rather than the horizontal direction is reported by many authors (see, e.g., [27, 119]). Connections of these phenomena with the pair distribution function and its dependence on the particle volume fraction and particle Reynolds number have also

been pointed out (see, e.g., [55]). The vexed question of the divergence of the velocity fluctuations with vessel size [19] has received much attention especially in the dilute, low-Reynolds-number regime (see [51] for a good summary). The matter has been resolved by showing that the predicted divergence only occurs with a hard-sphere particle distribution that in practice does not persist due to the evolution of the suspension microstructure, in particular with increasing Reynolds number.

These results are very helpful in that they begin to flesh out a picture of the dynamics of these systems. However, as can been from the Table, the region of parameter space covered by these studies is still limited, especially in view of the range relevant for many applications. In the present work we use the Physalis method to examine many of the issues studied by previous investigators in extended parameter ranges, in particular by considering moderately dense suspensions at single-particle Reynolds numbers up to 114 using the simulations that were described in Chapter 1.2. We also study particle collisions, clustering and the time evolution of tetrads, four-particle structures. We find that several earlier results on the two-particle distribution function, particle diffusivity and particle velocity fluctuations extend to the parameter range considered here. Several previously identified qualitative features of the results, such as trends with increasing volume fraction, are found not to be intrinsic to the system dynamics, but dependent on the normalization used to present the results. For brevity, some of the plots in this chapter are only shown for a few simulations—the remainder of the plots can be found in Appendix B.1.

4.1 Results

The present simulations are conducted in the frame of reference in which the mean vertical particle velocity $\langle w_z \rangle$ vanishes at each time step; here the angle brackets denote the average over the particles. Thus, the frame of reference used here is appropriate for the description

Table 4.1: Parameter ranges addressed by some previous experimental and numerical studies of settling particles in a fluid; Ga is the Galilei number, ρ_p/ρ_f the particle-to-fluid density ratio, ϕ the particle volume fraction and Re_t the single-particle terminal Reynolds number; when not explicitly given in the original reference, the Galilei number was calculated from (1.2).

Experimental					
Reference	Year	Ga	ρ_p/ρ_f	ϕ (%)	$Re_t = dw_t/\nu$
Ham and Homsy [53]	1988	~ 0.04	2.24	2.5 – 10	$< 10^{-4}$
Nicolai et al. [79]	1995	~ 0.1	2.53	0 – 40	$< 10^{-3}$
Segrè et al. [93]	1997	~ 0.05	~ 1	0.1 – 5	1.2×10^{-4}
Segrè et al. [94]	2001	0.04	–	5 – 50	10^{-4}
Chehata Gómez et al. [22]	2009	~ 0.03	2.6 – 4.2	0.1 – 0.8	$\sim 4 \times 10^{-5}$
Snabre et al. [98]	2009	0.17	1.34	10 – 55	1.6×10^{-3}
Numerical					
Reference	Year	Ga	ρ_p/ρ_f	ϕ (%)	$Re_t = dw_t/\nu$
Ladd [68]	1993	–	–	5 – 45	0
Climent and Maxey [27]	2003	1.4 – 17.9	0.9 – 1.5	1 – 12	0.1 – 10
Yin and Koch [117]	2007	4.5 – 28.5	2.0	0.5 – 40	1 – 20
Yin and Koch [118]	2008	1.9 – 28.5	2.0	1 – 20	0.2 – 20
Hamid et al. [54]	2013	2.3	5	1 – 50	0.28
Hamid et al. [55]	2014	1.0 – 17.9	5	1 – 40	0.05 – 10
Uhlmann and Doychev [105]	2014	121, 178	1.5	0.5	141, 233
Zaidi et al. [119]	2015	0.4 – 54.3	2.5 – 2.7	≤ 40	0.01 – 50
Fornari et al. [42]	2016	145	1.02	0.5 – 1	188
Present work	2018	50 – 99	2.0 – 5.0	8.7 – 34.9	44 – 114

Table 4.2: Calculated vertical mean fluid velocity $\langle u_z \rangle$ normalized by the reference velocity (4.1) and dimensionless particle characteristic time defined in (4.3) for the present simulations.

ρ_p/ρ_f	ϕ (%)	$\langle u_z \rangle/\tilde{u}$	$\tau_p \tilde{u}/d$	ρ_p/ρ_f	ϕ (%)	$\langle u_z \rangle/\tilde{u}$	$\tau_p \tilde{u}/d$
2.0	8.7	0.842	1.568	4.0	8.7	0.820	4.830
2.0	17.5	0.820	1.413	4.0	17.5	0.779	4.556
2.0	26.2	0.802	1.246	4.0	26.2	0.761	4.180
2.0	34.9	0.753	1.088	4.0	34.9	0.734	3.790
3.3	8.7	0.828	3.610	5.0	8.7	0.824	6.635
3.3	17.5	0.784	3.384	5.0	17.5	0.782	6.303
3.3	26.2	0.768	3.079	5.0	26.2	0.760	5.844
3.3	34.9	0.731	2.780	5.0	34.9	0.724	5.380

of a fluidized-bed-like system. The mean vertical fluid velocity $\langle u_z \rangle$ (with angle brackets here denoting the time and volume average over the fluids phase) calculated in this system is readily converted to the mean particle settling velocity in a sedimentation set-up in which the mean overall volumetric flux of the mixture vanishes (see, e.g., [117]). Upon effecting this change of axes we can compare our results with numerous others available in the literature. We have done so in Chapter 3 finding good agreement with the Richardson-Zaki correlation as modified in [117].

In principle, the physically relevant velocity to be used in the scaling of the numerical results is the mean fluid-particle relative velocity which, however, is itself a result of the calculations. For convenience we will therefore use a reference fluid-particle relative velocity $\tilde{u}$ defined by

$$\frac{\tilde{u}}{w_t} = (1-\phi)^{n-1}\,, \tag{4.1}$$

in which n is the Richardson-Zaki exponent for which Garside and Al-Dibouni [44] give the relation

$$\frac{5.1-n}{n-2.7} = 0.1 Re_t^{0.9}\,. \tag{4.2}$$

The reference velocity (4.1) has the advantage of being easy to calculate from a knowledge

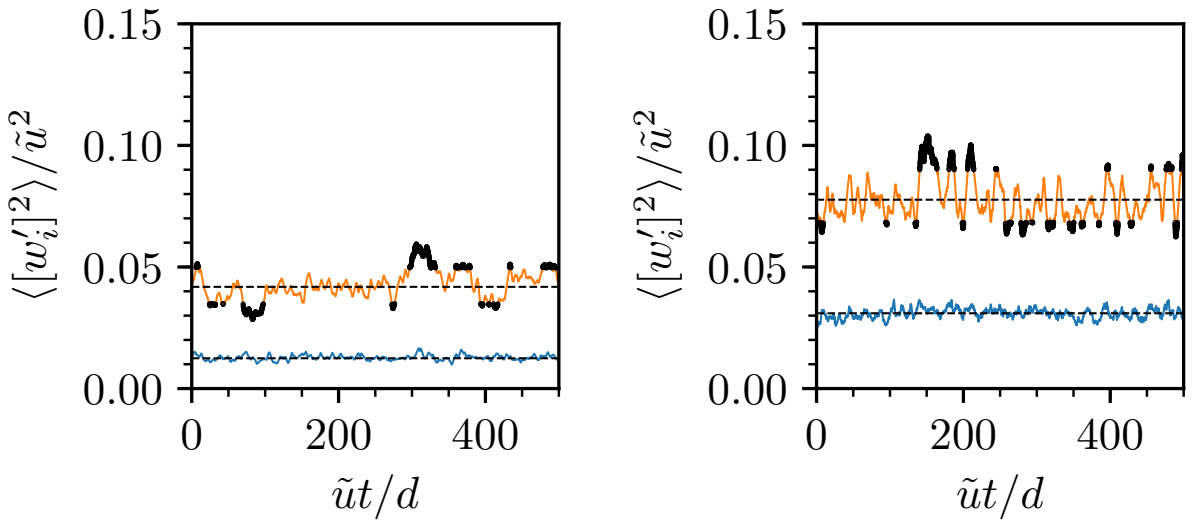


Figure 4.1: Two examples of the time dependence of the instantaneous fluctuations of the square of the particle velocity in the vertical (upper lines) and horizontal directions with respect to the time-mean values indicated by the horizontal lines; the left panel is for $\phi = 8.7\%$ and the right panel $\phi = 34.9\%$, both with $\rho_p/\rho_f = 3.3$. The highlighted portions of the lines correspond to the fraction of values in the top and bottom 10% of the computed values. The time interval shown is a fraction of the total length of the simulation. The reference velocity $\tilde{u}$ is defined in (4.1).

of w_t for which reliable correlations are available in the literature (see, e.g., [26]).

The relation between the calculated vertical mean fluid-particle velocity and the reference velocity (4.1) is shown in Table 4.2, which confirms the close quantitative relationship of the two quantities. The Table also includes the normalized characteristic particle relaxation time τ_p defined by

$$\tau_p = \frac{d^2}{18\nu}\frac{\rho_p}{\rho_f}\left[1 + 0.15\left(\frac{d\langle u_z\rangle}{\nu}\right)^{0.687}\right]^{-1} . \tag{4.3}$$

4.1.1 Clustering

The quantity

$$\frac{\langle [w_i'(t)]^2\rangle}{\langle u_z^2\rangle} = \frac{\langle [w_i(t) - \langle w_i\rangle]^2\rangle}{\langle u_z^2\rangle} , \tag{4.4}$$

represents the instantaneous fluctuation of the square of the particle velocity in the vertical ($i = z$) and horizontal ($i = x$, y) directions. The mean values of this quantity in the two directions are represented by the horizontal lines in the two panels of Figure 4.1. The upper lines in these two panels show examples of the time dependence of the vertical component of this quantity for a low- (left) and a high-concentration case, $\phi = 8.7\%$ and $\phi = 34.9\%$, respectively, with $\rho_p/\rho_f = 3.3$. The highlighted portions of the lines correspond to the fraction of values in the top and bottom 10% of computed values. The lower lines represent the analogous horizontal velocity fluctuations. While the latter exhibit small statistical oscillations around their mean value, a striking feature of the squared vertical particle velocity fluctuations is the presence of long-lived prominent peaks above and minima below the mean value. For both volume fractions the peaks can be as large as 35% above the mean value and are interspersed by smaller-amplitude fluctuations. For the higher-concentration case the peaks are more frequent and have a shorter duration. In both cases the lifetime of the peaks is longer than the particle integral time scale defined in (4.13) and shown later in Figure 4.10.

The presence of these long-lived peaks strongly suggests the formation of particle clusters in which a number of particles sufficient to have a significant impact on the mean value move downward or upward faster than the average. The presence of such clusters is visually evident in the movies referenced in Appendix B.1. Nicolai et al. [79] observed the existence of clusters in their experiments, as did Ladd [68] in his simulations, with both studies in the low-Reynolds number regime. The more recent study by Uhlmann and Doychev [105] at two values of the Galilei number, $Ga = 121$ and 178, also showed the presence of clusters at $Ga = 178$, but not as clear for $Ga = 121$. These authors explained their results on the basis of known features of single-particle settling wakes, which are certainly important at the very dilute conditions that they considered, $\phi = 0.5\%$. These considerations, however, are

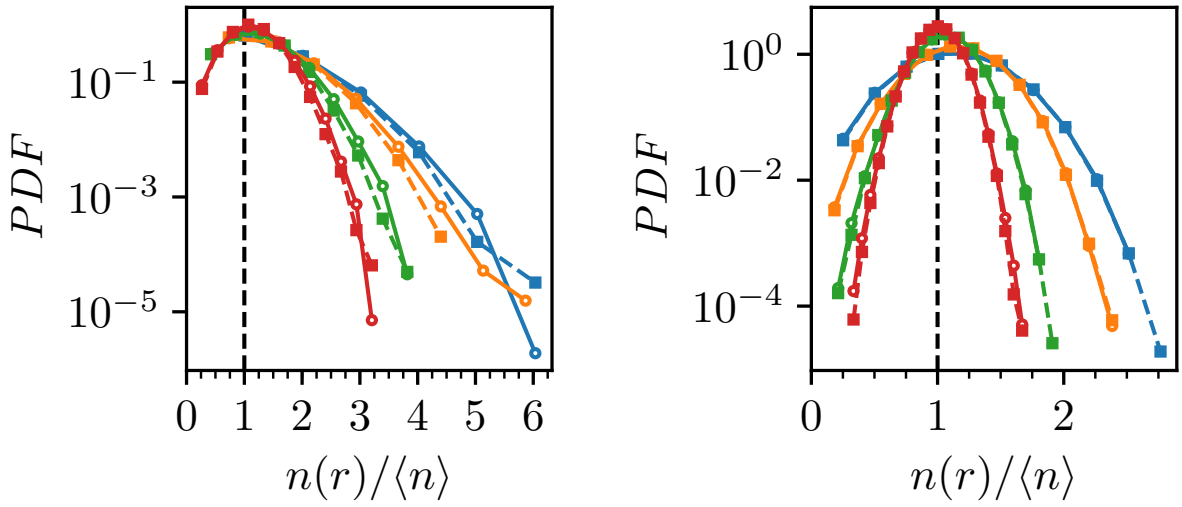


Figure 4.2: $n(r)$ is the normalized number density of particles with center in spherical shells of different radii r centered on each particle during the periods of time highlighted in Figure 4.1. The figure shows the PDF of $n(r)/\langle n\rangle$, with $\langle n\rangle$ the mean number density over the computational domain, for $r/a = 2.25$ (blue), 2.50 (orange), 3 (green) and 3.50 (red) and the same conditions as in the previous figure. The solid and dashed lines are the values of $n(r)/\langle n\rangle$ during the high- and low-velocity intervals, respectively. For the low-concentration case on the left note the higher probability of larger values of $n(r)/\langle n\rangle$ during the higher-velocity periods highlighted in the previous figure. For the denser case, this effect is only apparent for $r = 3.5\,a$, and it is also present for $n(r) < \langle n\rangle$, in which case the local mixture is buoyant with respect to the average.

probably not very relevant for the present range of concentrations and, besides, the study at two values of Ga did not permit them to establish whether the clustering that they observed has a gradual or an abrupt onset.

Uhlmann and Doychev [105] established their results on clustering by means of an elaborate analysis involving the Voronoi tessellation of their computational domain. Here it is sufficient to proceed more simply by calculating $n(r)$, the normalized number density of particles with center in spherical shells of different radii r centered on each particle (referred to as "test particle" in the literature) during the periods of time highlighted in Figure 4.1; the test particle is included in the count. Figure 4.2 shows the PDF of this number di-

vided by the mean number density, $n(r)/\langle n\rangle$, for $r/a = 2.25$, 2.50, 3 and 3.50; the vertical dashed line shows the average number density over the entire computational domain. The solid and dashed lines are the normalized number densities of particles during the high- and low-velocity intervals, respectively. For the low-concentration case, $\phi = 8.7\%$, (left panel in Figure 4.2) there is a significant probability to find a normalized number density during the peak events larger than that during the periods of low values of $\langle w_z^2\rangle$. The only exception are the points in the right "tail" around $n(r = 2.25) \simeq 5\langle n\rangle$, which is probably a statistical fluctuation due to the very small number of such events. This result confirms that the peak periods of $\langle w_z^2\rangle$ correspond to the formation of relatively large particle clusters which extend at least as far as $r = 3.5\,a$.

The right panel of Figure 4.2 is a similar graph for the case $\phi = 34.9\%$ (note the difference in the horizontal scale). Here the results for the smaller shells are nearly indistinguishable, but a slightly increased probability of a local number density greater than $\langle n\rangle$ during high-velocity intervals emerges for $r = 3.5\,a$. The effect appears smaller than for the lower-concentration case, which is not surprising in view of the smaller space available to the particles, but its presence is still suggested by the results. Interestingly, the effect appears to be present not only for $n(r) > \langle n\rangle$, in which the particle cluster surrounding the test sphere is heavier than the average, but also for $n(r) < \langle n\rangle$, in which the particle number is smaller and therefore the mixture is locally buoyant with respect to the average. Since, for r fixed, there can be many more particles near the test particle in the dense as compared with the dilute case, there is a significantly larger probability to encounter spherical shells with $n(r) < \langle n\rangle$.

An examination of the probability density function of w_z (not shown) reveals that high-velocity events with $w_z > 0$ and $w_z < 0$ occur in approximately the same number, although the frequency of the events with $w_z > 0$ tends to slightly increase with the volume fraction.

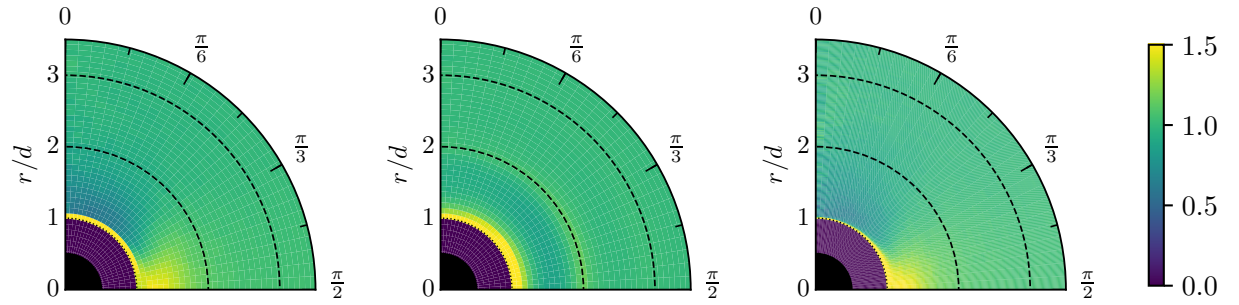


Figure 4.3: The pair distribution function $g(r, \theta)$ for $\phi = 8.7\%$ (left) and 34.9% (center) for $\rho_p/\rho_f = 2$ ($Re_t = 43.27$); the image on the right is for $\phi = 8.7\%$ and $\rho_p/\rho_f = 5$ ($Re_t = 110.8$).

4.1.2 Two-particle distribution function

Several investigators have studied the two-particle distribution function $g(r, \theta)$ in systems of the type investigated here. Yin and Koch [117] and Hamid et al. [55] considered single-particle Reynolds numbers up to 20 and 10, respectively, while the Reynolds number range considered in [119] extends up to 50; all these authors considered volume fractions comparable with ours.

For low concentrations, all these studies agree in reporting an anisotropic distribution with more particles near the test particle in the horizontal direction, $\theta \sim \pi/2$, and fewer particles in the vertical direction, $\theta \sim 0$. Yin and Koch [117] find the greatest anisotropy for $\phi = 1\%$ and $Re_t = 10$, a regime in which the mixture behavior is dominated by the anisotropic wake interactions. As the volume fraction increases, the anisotropy decreases. Increased inertia from $Re_t = 1$ to 10 enhances the anisotropy [117, 55].

The Reynolds number range in the present work goes up to 110 but the results found by previous investigators for smaller inertia are essentially confirmed. Figure 4.3 shows the two-particle distribution function for $\phi = 8.7\%$ and 34.9% with $\rho_p/\rho_f = 2$ ($Re_t = 43.27$), as well as $\phi = 8.7\%$ with $\rho_p/\rho_f = 5$ ($Re_t = 110.8$); additional results of this type are shown in Appendix B.1. A comparison between the first two panels (constant Re_t, increasing ϕ) shows

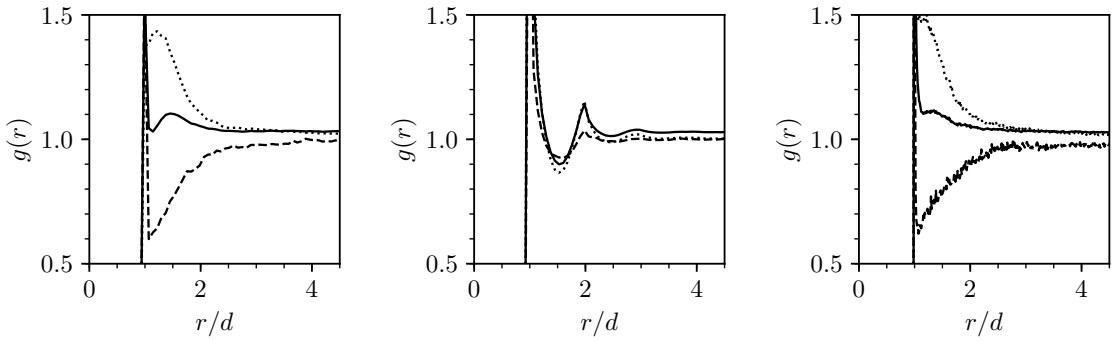


Figure 4.4: Angular averages of the pair distribution functions shown in the previous figure. The solid lines are the average over the entire range $0 \leq \theta \leq \pi/2$; the dashed lines are the average over a vertical sector $0 \leq \theta \leq \pi/12$ and the dotted lines over a horizontal sector $5\pi/12 \leq \theta \leq \pi/2$.

the fading of the anisotropy with increasing concentration. A comparison between the first and last panels shows the enhanced particle number in the horizontal direction and closer to the test particle caused by the increased inertia. A different view of the same information is provided in Figure 4.4, where the lines show three angular averages of $g(r, \theta)$. The solid lines are the average over the entire range $0 \leq \theta \leq \pi/2$, the dashed lines are the average over a vertical sector $0 \leq \theta \leq \pi/12$ and the dotted lines over a horizontal sector $5\pi/12 \leq \theta \leq \pi/2$. A comparison of the first two panels ($\phi = 8.7\%$ and 34.9%, $Re_t = 43.27$), shows that the anisotropy strongly decreases as the concentration increases, although it is not completely removed. The peaks around $r/2a = 2$ indicate the gradual build-up a "cage" of particles around the test particle, with a slightly stronger effect in the horizontal direction.

For the largest Reynolds number (last panel) the peak in the horizontal direction is higher and it moves closer to the test particle. The average in the vertical sector extends further out from the test particle and is slightly decreased. A plausible explanation of these results is that, at higher Re_t, the vertical orientation of particle pairs is less stable as the couple that causes the broad-side rotation is stronger and the fluid dynamic force that tends to separate horizontal pairs is less effective due to the particle inertia.

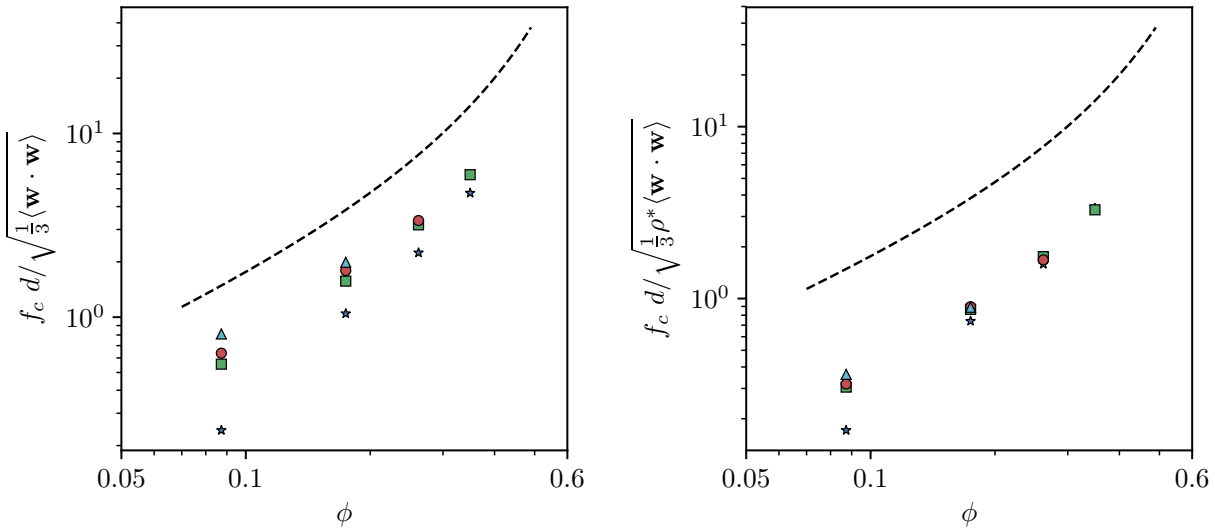


Figure 4.5: Particle collision frequency normalized as in (4.5) vs. volume fraction for the systems simulated in this study; the symbols denote the particle-to-fluid density ratio: $\rho^* = \rho_p/\rho_f = 2$ (asterisks), 3.3 (squares), 4 (circles) and 5 (triangles). The dashed lines are the predictions from the theory of granular flows.

4.1.3 Particle collisions

A specific phenomenon caused by larger inertia, particularly at moderate to large volume fractions, is an increased frequency of particle collisions. As described in [96], the collision algorithm used in the simulations embodies a nonlinear Hertzian contact model which is activated whenever the distance between two particle centers becomes equal to a diameter. This model is complemented by a lubrication interaction when the distance between the particle surfaces is less than a radius. Figure 4.5 shows our results for the volume fraction dependence of the normalized collisional frequency f_*,

$$f_* = f_c d \left(\frac{1}{3} \langle \mathbf{w} \cdot \mathbf{w} \rangle \right)^{-1/2} , \tag{4.5}$$

with f_c the computed value and the angle brackets denoting the particle average as before. The results shown are the average number of actual Hertzian contacts per particle per unit time. Normalization by the particle number is useful in that it admits proper accounting of

multi-particle collisions. The dashed line is the Enskog collision frequency for elastic hard spheres (see, e.g., [106, 17])

$$f_E = 4g_2(\phi)d^2n\sqrt{\frac{\pi}{3}\langle \mathbf{w}\cdot\mathbf{w}\rangle}\,, \tag{4.6}$$

normalized in the same way; here $g_2(\phi)$ is the angle-averaged two-particle distribution function at contact, approximated by the Carnahan-Starling formula

$$g_2(\phi) = \frac{1}{2}\frac{2-\phi}{(1-\phi)^3}\,. \tag{4.7}$$

Given the presence of hydrodynamic resistance, kinetic theory over-predicts the computed collisional frequency, although it provides a good match for the volume fraction dependence except for the lightest particles, which are most affected by the fluid. The collisional frequency increases with the particle mass, as expected from the fact that, with larger inertia, hydrodynamic forces become less and less able to prevent a close approach. The division of the collisional frequency by $\sqrt{\rho_p/\rho_f}$ produces an approximate collapse of the results as shown in the right panel of Figure 4.5. This result might reflect a role of the added mass in determining the relevant mean particle kinetic energy $\langle \mathbf{w}\cdot\mathbf{w}\rangle$ for the purposes of scaling some aspects of the system dynamics; a similar collapse is found below for the particle diffusivity.

4.1.4 Particle diffusion coefficient

The connection between the particle velocity correlation tensor $\langle w_i(t+\tau)w_j(t)\rangle$ and the particle diffusivity D_p is well known:

$$D_p^{(ij)} = \lim_{t\to\infty}\frac{1}{2t}\langle r_i(t)r_j(t)\rangle = \lim_{t\to\infty}\int_0^t \langle w_i(\tau)w_j(0)\rangle d\tau\,, \tag{4.8}$$

in which $r_i(t) = x_i(t) - x_i(0) - \int_0^t\langle w_i(\tau)\rangle d\tau$ is the displacement of the test particle from the initial position corrected for the mean displacement of all the particles. While in principle $D_p^{(ij)}$ is a tensorial quantity, we find that the off-diagonal components are very small so that

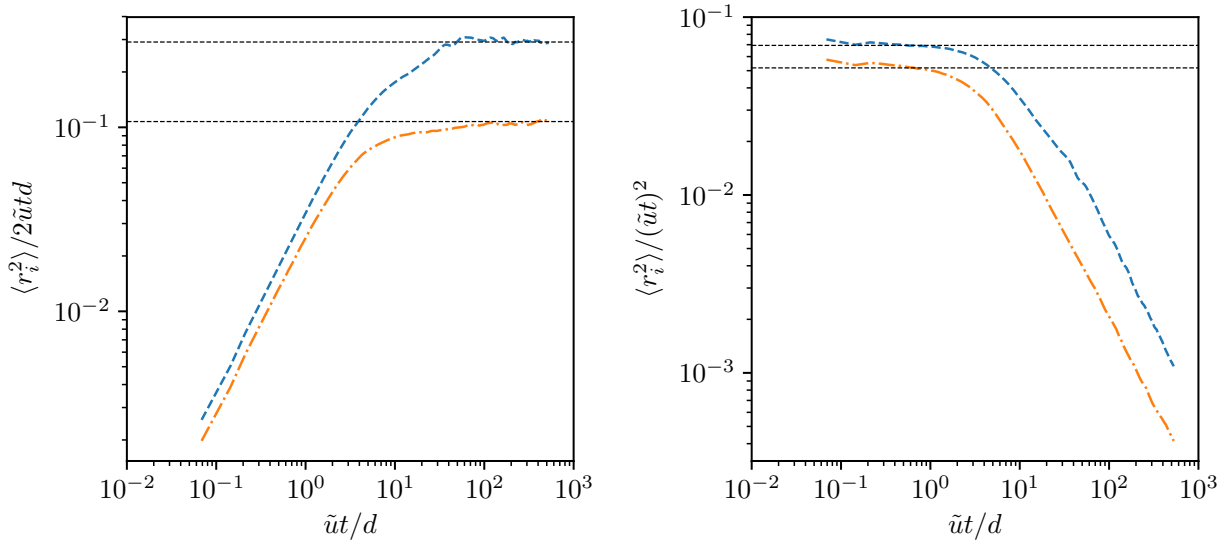


Figure 4.6: Two examples of the time dependence of the mean squared particle displacement in the vertical, $\langle r_z^2(t)\rangle/2\tilde{u}td$ (upper lines), and horizontal, $\langle r_\perp^2(t)\rangle/2\tilde{u}td$, directions. In the right panel the displacement is normalized by $\tilde{u}^2t^2$. The horizontal lines are the long-time and short-time mean values.

we limit ourselves to presenting results for the diffusivity in the vertical and in the horizontal directions, D_p^z and $D_p^\perp$, respectively, the latter calculated in the horizontal plane.

The time dependence of $\langle r_z^2(t)\rangle/2t$ and $\langle r_\perp^2(t)\rangle/2t$ is shown in the left panel of Figure 4.6, where the asymptotic approach to a constant for both the vertical and horizontal directions is evident. The right panel shows instead $\langle r_z^2(t)\rangle/t^2$ and $\langle r_\perp^2(t)\rangle/t^2$. The constant values of these quantities for short times indicate the prevalence of the so-called ballistic regime, during which the particle velocity maintains a correlation with itself.

The two sides of (4.8) provide two alternative ways to calculate the particle diffusivity, one from the mean-square displacement, the other from the integral of the velocity correlation. The results for D_p^z and $D_p^\perp$ calculated from the mean-square displacement as shown in the left panel of Figure 4.6 are shown in Figure 4.7, in which the dashed line is the prediction from the kinetic theory of granular gases, corrected for the factor of 3 used in the definition

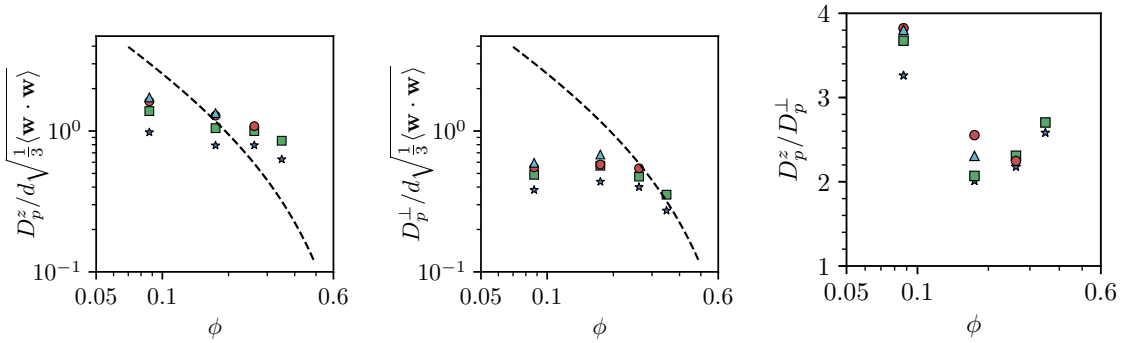


Figure 4.7: Particle diffusivities in the vertical (left) and horizontal (center) directions normalized as in (4.9); the last panel is the ratio of the two diffusivities and illustrates the marked anisotropy of the diffusion process in the system investigated. The dashed lines are the predictions from the theory of granular flows (4.9). The symbols denote the different particle-to-fluid density ratio: $\rho_p/\rho_f = 2$ (asterisks), 3.3 (squares), 4 (circles) and 5 (triangles).

of the diffusion coefficient in that theory [17]:

$$D_E = \frac{9}{8}\frac{1}{d^2 g_2(\phi) n}\sqrt{\frac{1}{3}\langle \mathbf{w}\cdot\mathbf{w}\rangle}\,. \tag{4.9}$$

The results are made dimensionless by division by $\sqrt{\frac{1}{3}\langle \mathbf{w}\cdot\mathbf{w}\rangle}$. The computed and kinetic theory results are comparable in magnitude, but there are significant differences in their concentration dependence. In the vertical direction, for $\phi = 8.7\%$, kinetic theory overpredicts the diffusivity, probably as a result of the hydrodynamic force on the particles that hinders their random motion. However, the trend reverses with increasing ϕ, for which kinetic theory predicts a much stronger decrease than the found in the simulations. A likely explanation is that the flowing fluid enhances the mobility of the test particle by breaking up the "cages" formed by the surrounding particles thus favoring its escape. While the flowing fluid enhances the vertical particle displacements, it has a much smaller effect in the horizontal direction. Thus, the horizontal diffusivity is lower than the kinetic theory prediction at low volume fractions due to hydrodynamic resistance, while it is found to follow the kinetic theory prediction at higher concentration for which the absence of a mean flow

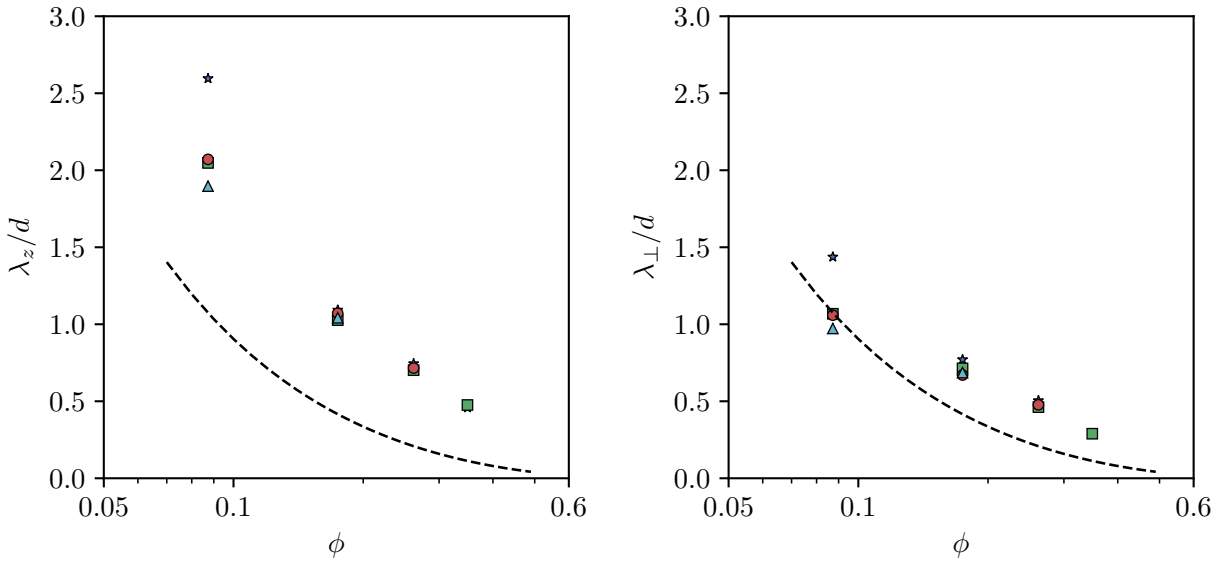


Figure 4.8: Normalized particle mean free path defined in (4.10) in the vertical (left) and horizontal direction; the dashed line is the kinetic theory prediction (4.11). The symbols denote the different particle-to-fluid density ratio: $\rho_p/\rho_f = 2$ (asterisks), 3.3 (squares), 4 (circles) and 5 (triangles).

does not enhance particle mobilities. This explanation is in accordance with the build-up of the cage structure mentioned before and with the radial distribution function mentioned at the end of Chapter 4.1.2. Interestingly, as shown in Appendix B.1, a division of both D_p^z and $D_p^\perp$ by $\sqrt{\rho_p/\rho_f}$ provides an excellent collapse of both diffusivities similarly to what was found for the collisional frequency. The last panel in Figure 4.7 is the ratio $D_p^z/D_p^\perp$. One observes a marked anisotropy with a minimum around $\phi \simeq 26.2\%$.

Following the approach of kinetic theory, we can obtain an estimate of the particle mean free path λ as

$$\frac{\lambda_{ii}}{d} = \sqrt{\frac{D_p^{ii}}{f_c d^2}}\,. \tag{4.10}$$

The kinetic theory prediction for this quantity is

$$\frac{\lambda_E}{d} = \frac{1}{6\sqrt{2}\,\phi g_2(\phi)}\,. \tag{4.11}$$

Figure 4.8 compares these two quantities. In the vertical direction, our calculated mean free

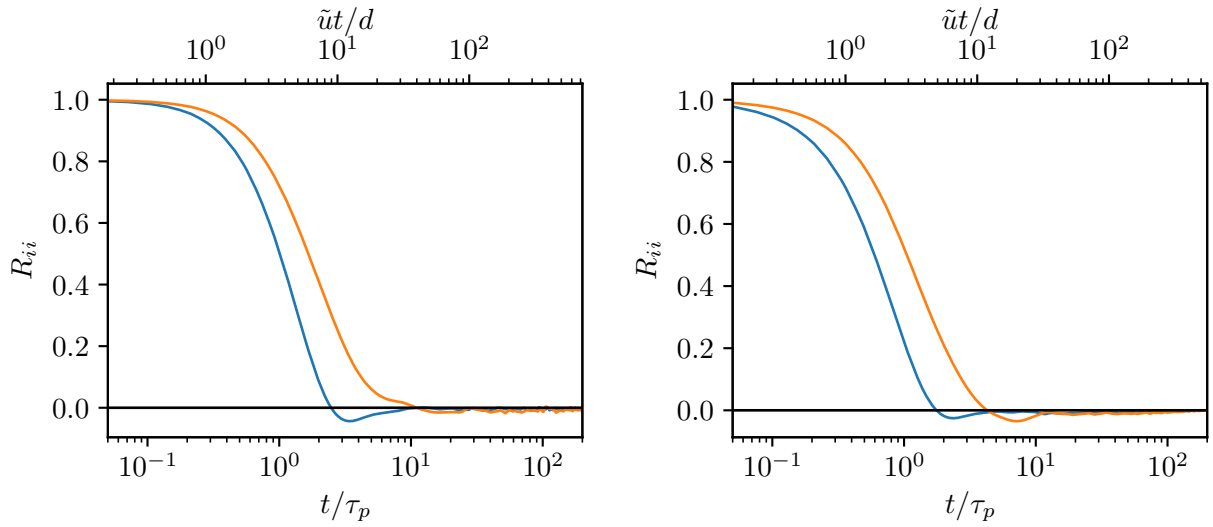


Figure 4.9: Two examples of the velocity autocorrelation (4.12) for $\phi = 8.7\%$ (left) and $\phi = 34.9\%$, both for $\rho_p/\rho_f = 3.3$. The upper and lower lines are for the horizontal and vertical velocity, respectively. On the lower horizontal axis time is normalized by the particle relaxation time τ_p defined in (4.3). On the upper axis time is normalized by the particles settling time scale $\tilde{u}t/d$.

path is longer than the kinetic theory prediction in agreement with the enhanced mobility previously demonstrated by the results for D_p^z. In addition, the flow caused by a particle tends to displace the particles toward which it moves, a process that cannot happen in a granular gas without interstitial fluid. This effect is analogous to that of a repulsive inter-particle force in kinetic theory, which is also known to increase the diffusivity [21]. In the horizontal direction kinetic theory under-predicts the mean free path as well, but by a significantly smaller margin. Again we see here the effect of the absence of a mean horizontal fluid velocity resisted by a force comparable to gravity.

We now turn to the second way to estimate the diffusivity, namely by integrating the normalized velocity correlation

$$R_{ii} = \frac{\langle w_i(t) w_i(0) \rangle}{\langle w_i^2 \rangle} . \tag{4.12}$$

It may be noted that the denominator is the same quantity shown by the horizontal lines in

Figures 4.1 and 4.6. Figure 4.9 shows two examples of the dependence of R_{ii} on time for $\rho_p/\rho_f = 3.3$ and $\phi = 8.7\%$ and 34.9% with two different normalizations, the particle relaxation time on the lower horizontal axis and the particle settling time on the upper one. Comparison of these two scales reveals that the velocity remains correlated for a time during which particles fall by about three diameters. As found by several earlier investigators [79, 27, 55], the velocity de-correlates much faster in the horizontal than in the vertical direction. The correlation lasts longer at low volume fraction due to the weaker particle-particle interactions caused by the larger mean separation.

To examine the importance of direct collisions on the correlation we can examine the integral time scale defined by

$$\mathcal{T}_{ii} = \lim_{t\to\infty} \int_0^t \frac{\langle w_i(\tau) w_i(0)\rangle}{\langle w_i^2\rangle}\, d\tau\,, \tag{4.13}$$

the average of which is shown in Figure 4.10 normalized by $d/\tilde{u}$. This time scale shows a strong decrease with increasing ϕ. If direct collisions were the major factor affecting the velocity correlations, one would expect that the product $\mathcal{T}_{ii} f_c$ would be approximately constant. This, however, is not the case as shown in Figure 4.11, where this product is found to undergo a marked increase with ϕ, primarily due to the increase of the collisional frequency. This result indicates that direct collisions are mostly weak, as already remarked in Chapter 1.2. The conclusion that must be drawn is that the most significant agents causing the de-correlation of a particle velocity are the flow fields caused by the particles that it encounters rather than direct contacts as in a granular gas.

The result for the particle diffusivity calculated from the velocity correlation integral (open symbols) is compared with that calculated with the mean-square displacement shown earlier (solid symbols) in Figure 4.12. Here, unlike Figure 4.7, we normalize D_p by $\tilde{u}d$ to bring out the presence of a maximum around $\phi \sim$ 10-15%, which was also found e.g. in [55]. The effect of this normalization is due to the rapid decrease of the mean fluid-particles relative

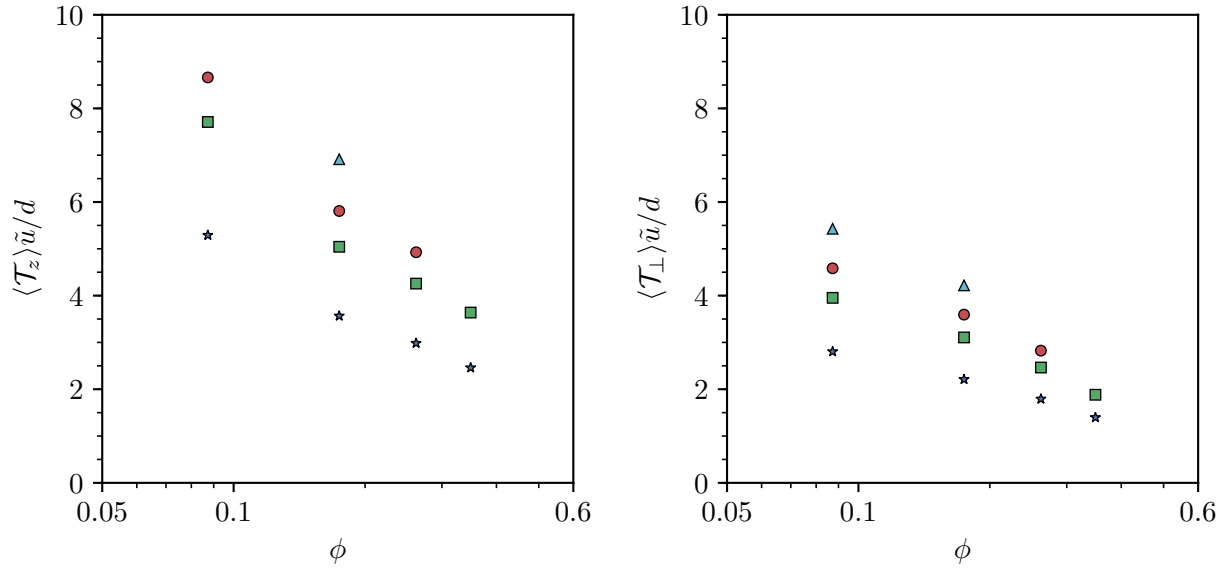


Figure 4.10: Normalized integral time scale defined in (4.13) in the vertical (left) and horizontal direction. The symbols denote the different particle-to-fluid density ratio: $\rho_p/\rho_f = 2$ (asterisks), 3.3 (squares), 4 (circles) and 5 (triangles).

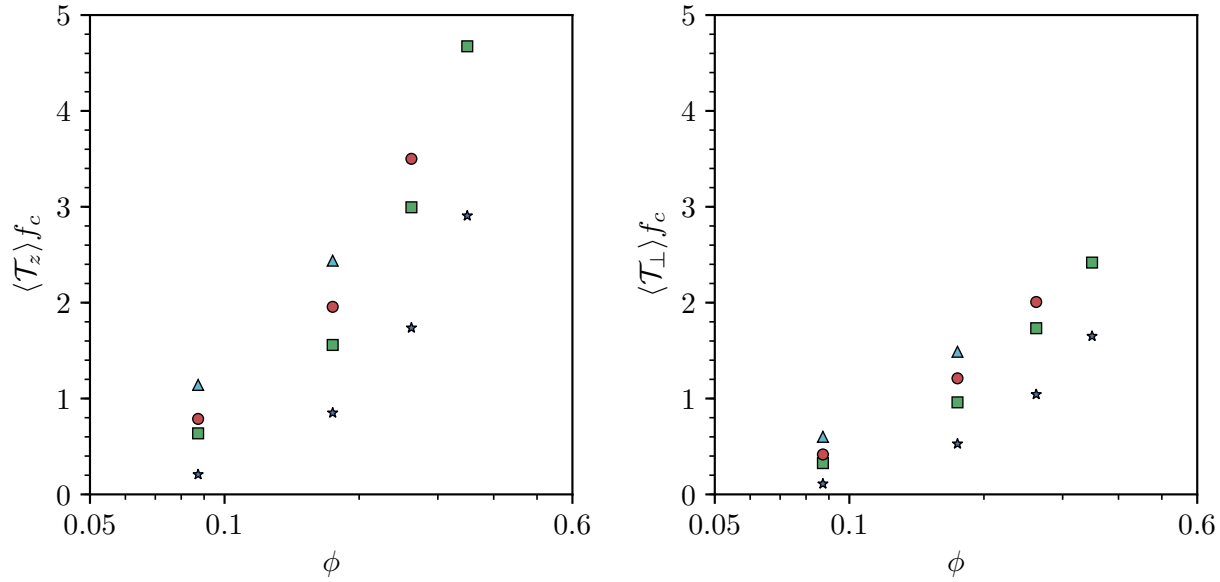


Figure 4.11: Integral time scale defined in (4.13) in the vertical (left) and horizontal direction normalized by the collision frequency.

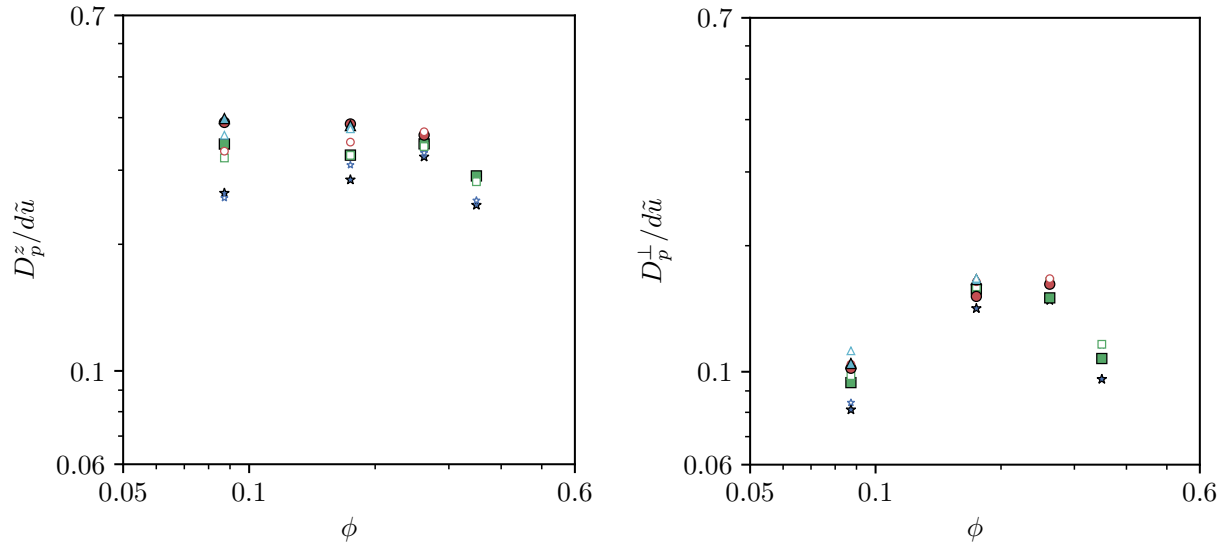


Figure 4.12: The open symbols show the particle diffusivities in the vertical (left) and horizontal directions as obtained from integration of the velocity correlation normalized by $d\tilde{u}$; the closed symbols are the same as shown in Figure 4.7 with this different normalization. The symbols denote the different particle-to-fluid density ratio: $\rho_p/\rho_f = 2$ (asterisks), 3.3 (squares), 4 (circles) and 5 (triangles).

velocity with particle concentration, as represented by $\tilde{u}$, which is faster than the decrease of $\sqrt{\langle \mathbf{w} \cdot \mathbf{w} \rangle}$. This second normalization, however, appears to be less justified by the physics of the particle diffusion, which is directly dependent on the particle velocity fluctuations, incorporated in the earlier normalization by the use of $\sqrt{\langle \mathbf{w} \cdot \mathbf{w} \rangle}$. The two ways to calculate D_p have an average difference of 5%. In view of the difficulty of accurately estimating the velocity correlation and its integral, reliance on the mean-square displacement method is probably justified.

4.1.5 Velocity fluctuations

Velocity fluctuations in sedimenting suspensions have been investigated by many authors motivated by the predicted divergence with container size at low Reynolds numbers. The current understanding of this matter is summarized in the review [51]. Since our domain

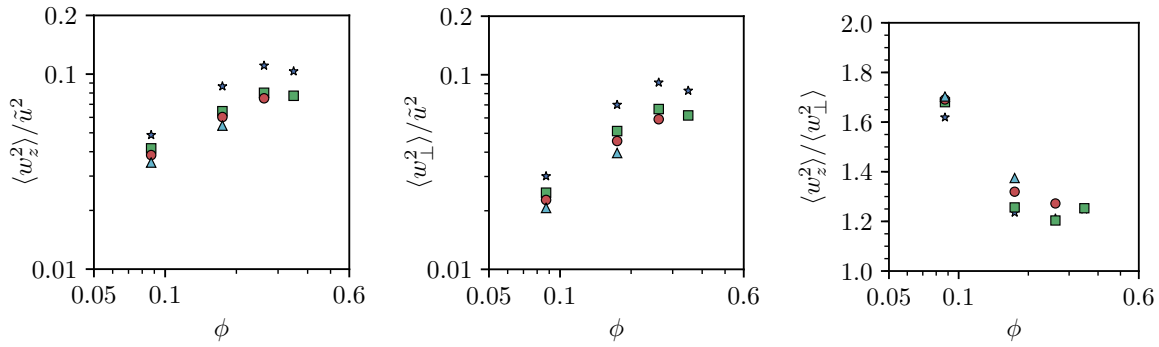


Figure 4.13: Particle velocity fluctuations in the vertical (left) and horizontal (center) directions; the last panel is the ratio of the two and indicates a marked anisotropy. The symbols denote the different particle-to-fluid density ratio: $\rho_p/\rho_f = 2$ (asterisks), 3.3 (squares), 4 (circles) and 5 (triangles).

size is fixed, we cannot comment on this aspect.

Our results for the particle velocity fluctuations normalized by the particle terminal velocity w_t (not shown) confirm those presented in [55] in their common Reynolds number range. We find a peak at our second lowest volume fraction ($\phi = 17.5\%$) and a decrease thereafter. Figure 4.13 shows the present results normalized by $\tilde{u}$ rather than w_t. With this normalization the fluctuations increase with ϕ and appear to saturate at our largest volume fraction $\phi = 34.9\%$. The different trends obtained with the two normalizations depend on the fact that, while $\tilde{u}$ decreases with concentration, w_t is independent of the particle concentration. At the larger Reynolds numbers they studied, $Re_t = 20$, [118] found a very slow decay of $\langle w_z^2 \rangle / w_t^2$. We find similar results in our higher Reynolds number range for all the volume fractions we simulated.

The last panel in Figure 4.13 is the ratio of the vertical and horizontal velocity fluctuations. There is a marked anisotropy with a minimum around $\phi = 26.2\%$ which, not coincidentally, occurs at the same volume fraction where the anisotropy of the diffusivities is also a minimum.

4.2 Tetrads

Some interesting information on the behavior of the system under consideration can be obtained by a study of the time evolution of the shape and orientation of groups of four particles – particle tetrads. This study enables us to probe the small-scale dynamics of the particulate phase beyond the one- and two-particle information considered up to this point, as well as to determine the three-dimensional structure of particle arrangements. Studies of this type have been carried out e.g. in polymer science [101, 10], in the theory of random walks [90], single-phase turbulence (e.g., to define a large scale velocity gradient tensor) [23, 15, 116, 87] and other areas [74].

4.2.1 Tetrad geometry

Several different ways of investigating tetrad geometry have been proposed (see, e.g., [87]). We use an approach common in the polymer literature [101, 90] in view of its intuitive appeal.

At each instant of time, we define a coarse-grained velocity gradient tensor M_{ji} around the instantaneous center of each tetrad by minimizing the quantity[87]

$$K = \sum_{n=1}^{4}\sum_{i=1}^{3}\left[(w_i^n - \overline{u}_i) - \sum_{j=1}^{3}(x_j^n - \overline{x}_j)\mathsf{M}_{ji}\right]^2 , \tag{4.14}$$

in which x_i^n and w_i^n represent the i^{th} component of position and velocity of the n^{th} particle in the tetrad and

$$\overline{x}_i \;=\; \frac{1}{4}\sum_{n=1}^{4} x_i^n \,, \qquad \overline{w}_i \;=\; \frac{1}{4}\sum_{n=1}^{4} w_i^n \,, \tag{4.15}$$

are the position and velocity of the center of the tetrad. The velocity gradient tensor M_{ji} that minimizes K is the solution of the linear system

$$\sum_{k=1}^{3}\mathsf{G}_{ik}\mathsf{M}_{kj} = \mathsf{W}_{ij} \,, \tag{4.16}$$

in which G_{ik} is the shape (or gyration) tensor [89, 10, 101]

$$\mathsf{G}_{ij} = \frac{1}{4}\sum_{n=1}^{4}(x_i^n - \overline{x}_i)(x_j^n - \overline{x}_j)\,, \tag{4.17}$$

and

$$\mathsf{W}_{ij} = \frac{1}{4}\sum_{n=1}^{4}(x_i^n - \overline{x}_i)(w_j^n - \overline{w}_j)\,. \tag{4.18}$$

The eigenvectors of the shape tensor

$$\mathsf{G}\,\mathbf{v}_k \;=\; \lambda_k \mathbf{v}_k\,, \tag{4.19}$$

with λ_k the respective eigenvalues, identify the principal axes of the tetrad orientation. For an isotropic tetrad, $\lambda_1 = \lambda_2 = \lambda_3$. The normalized eigenvalues

$$I_k \;=\; \frac{\lambda_k}{3\overline{\lambda}}\,, \tag{4.20}$$

are defined in terms of the mean value $\overline{\lambda} \;=\; \frac{1}{3}\,(\lambda_1 + \lambda_2 + \lambda_3)$. The deviation of the normalized eigenvalues from 1/3 provides a measure of the anisotropy of the shape tensor.

The antisymmetric part of the velocity gradient tensor describes the rotation of the tetrad, while the symmetric part $\mathsf{S}_{ij} = \frac{1}{2}\,(\mathsf{M}_{ij} + \mathsf{M}_{ji})$, on which we focus, carries information on its deformation. The eigenvectors $\mathbf{s}_k$ of S are the directions of the principal axes of strain.

The primary measure of the tetrad size is the radius of gyration R_G defined by

$$R_G^2 = \mathrm{Tr}(\mathsf{G}_{ij}) \;=\; \lambda_1 + \lambda_2 + \lambda_3 \;=\; 3\overline{\lambda}\,. \tag{4.21}$$

The shape of the tetrad can be further characterized by two dimensionless parameters, the shape variance Δ, also called "relative shape anisotropy," and the shape factor S. The shape variance is defined as

$$\Delta \;=\; \frac{3}{2}\frac{\mathrm{Tr}(\hat{\mathsf{G}}_{ij}^2)}{(\mathrm{Tr}\mathsf{G}_{ij})^2}\,, \tag{4.22}$$

where $\mathrm{Tr}(\hat{\mathsf{G}}_{ij}^2) \;=\; \sum_{k=1}^{3}(\lambda_k - \overline{\lambda})^2$ is proportional to the variance of the eigenvalues of the deviatoric part of G_{ij}, defined by $\hat{\mathsf{G}}_{ij} = \mathsf{G}_{ij} - \overline{\lambda}\delta_{ij}$. It can be shown that $0 \le \Delta \le 1$ [10]. For

a spherically symmetric particle arrangement the shape variance vanishes, while it reaches its maximum value 1 when the centers of the four particles fall on a straight line; for a right tetrad with three isosceles triangles $\Delta = 1/9$. The shape factor S is defined by

$$S = 27\frac{\det(\hat{\mathsf{G}}_{ij})}{(\mathrm{Tr}(\mathsf{G}_{ij}))^3}\,, \tag{4.23}$$

where $\det(\hat{\mathsf{G}}_{ij}) = \prod_{k=1}^{3}(\lambda_k - \overline{\lambda})$. It can be shown that $-\frac{1}{4} \leq S \leq 2$ [10]. For a prolate particle arrangement $\lambda_1 > \overline{\lambda} > \lambda_2$, λ_3 and S is positive while, for an oblate arrangement with λ_1, $\lambda_2 > \overline{\lambda} > \lambda_3$, S is negative.

4.2.2 Tetrad Initialization

At the initial instant of each realization we select four-particle groups with an initial distance between any two particle centers between slightly less than $2a$ (to account for the very small overlap occurring during collisions) and $2.5\,a$. To ensure a reasonable degree of isotropy in the initial tetrad configuration we only use tetrads with an initial shape variance of $\Delta \leq 0.15$. These choices strike a balance between the number of tetrads sufficient for statistical convergence and the removal of a significant contamination by anisotropic structures.

In view of the triply periodic nature of the simulation, we prefer not to follow the tetrads much beyond the time when the average radius of gyration exceeded $20a$, the shortest dimension of our computational domain. Since this criterion was applied to the average radius of gyration rather than to each individual tetrad, one would expect that the radius of gyration of some tetrads would have exceeded $20a$. However, this is not a serious concern because, as will be seen below, tetrads tend to elongate in the vertical direction, in which the domain size is $60a$ and, furthermore, one can account for a particle exiting the computational domain simply by adding the width of the domain to the position of its image inside the domain. Since the time to reach $\langle R_G \rangle = 20a$ was shorter than the duration of the simulations, we carved up each simulation into several portions that were used to populate the ensemble

Table 4.3: Number of tetrads tracked for each case

ϕ	ρ^*	# of Tetrads
0.087	[2, 3.3, 4, 5]	[1376, 2871, 3569, 4332]
0.175	[2, 3.3, 4, 5]	[34327, 52686, 61903, 67910]
0.262	[2, 3.3, 4, −−]	[104230, 174862, 197487, −−]
0.349	[2, 3.3, −−, −−]	[84339, 232426, −−, −−]

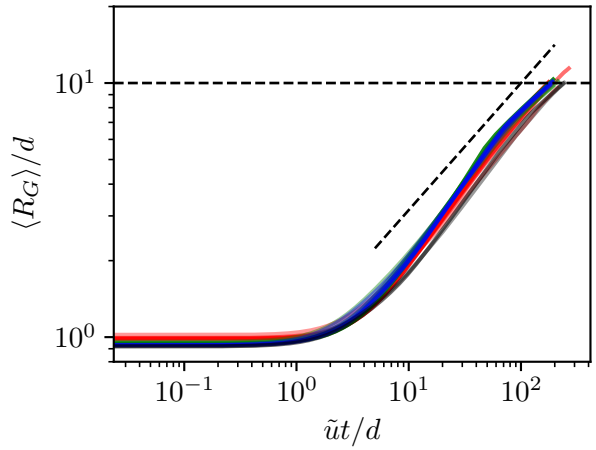


Figure 4.14: Time dependence of the normalized mean tetrad radius of gyration defined in (4.21); $\rho_p/\rho_f = 2$ (gray), 3.3 (green), 4 (red) and 5 (blue).

over which averages were calculated. The time necessary for a doubling of $\langle R_G \rangle$ was used to define the interval between the starts of successive portions. The number of tetrads tracked in this way for each set of parameters is presented in Table 4.3.

4.2.3 Shape evolution

Results from simulations with different volume fractions are represented by different colors (red for $\phi = 8.7\%$, green for $\phi = 17.5\%$, blue for $\phi = 26.2\%$, and gray for $\phi = 34.9\%$).

Mean

The radius of gyration averaged over all tetrads is shown as a function of a dimensionless time $\tilde{u}t/2a$ in Figure 4.14. Here the horizontal black dashed line identifies the length of

the smallest dimension of the computational domain and the other dashed line has a slope 1/2. The time scale $d/\tilde{u}$ does a good job of collapsing the curves, while it is found that a characteristic time based on the granular temperature does not (not shown). The tetrads maintain a size close to the initial one for a short time indicating that the constituent particles are still experiencing a similar flow environment and similar hydrodynamics forces. This corresponds to the ballistic regime in the right panel of Figure 4.6. At $\tilde{u}t/d \sim 10$, at which time R_G has doubled to approximately $2d$, the radius of gyration enters a diffusive growth regime. This is approximately the time when the diffusivity approaches a constant value as shown in the left panel of Figure 4.6.

The normalized eigenvalues (4.20) averaged over all the tetrads are shown in Figure 4.15. After a short initial time $\tilde{u}t/d \sim 1$, during which the tetrads remain close to their initial shape, the normalized eigenvalues start evolving until they reach approximately steady values around $\tilde{u}t/d \sim 10$. The curves corresponding to different Galilei numbers collapse very well. However, they do not collapse as well for different ϕ. The numerical values of the $\langle I_k \rangle$'s show that the initially-regular tetrads evolve into thin elongated structures at long times. Around $\tilde{u}t/d \sim 10^2$, we begin to see a departure from the steady state that corresponds to $\langle R_G \rangle$ approaching the shortest dimension of the computational domain.

The behavior of the shape variance Δ and shape factor S, shown in Figure 4.16, are compatible with this interpretation. The asymmetry parameter Δ reaches a value of approximately 0.6, which is compatible with a very small third eigenvalue, i.e., with the particles forming a thin tetrad. The shape factor reaches a steady state value of $S \approx 0.75$, indicating prolate shapes, as also demonstrated by the normalized eigenvalues.

The shape and symmetry of the tetrads remain approximately constant up to $\tilde{u}t/d \sim 1$. Over the following decade of time, the tetrads change shape rapidly, ultimately reaching approximately steady values around $\tilde{u}t/d \sim 10$. The appearance of all these figures changes

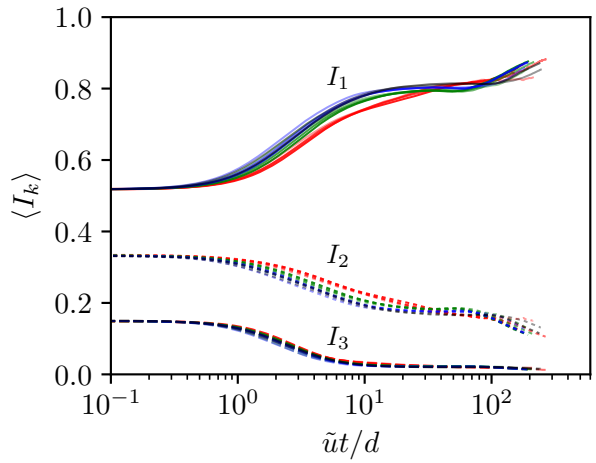


Figure 4.15: Time dependence of the normalized eigenvalues of the shape tensor, (4.20); $\rho_p/\rho_f = 2$ (gray), 3.3 (green), 4 (red) and 5 (blue).

from $\tilde{u}t/d \sim 100$ onward, possibly an effect of the tetrad size approaching or exceeding the dimensions of the computational domain.

Shape Alignment

The alignment of the tetrad principal axes with the direction of gravity $\mathbf{e}_z$, $\mathbf{e}_z \cdot \mathbf{v}_k(t)$, is shown in Figure 4.17. The initial value of this quantity, close to 0.5, indicates that the initialization of the tetrads is not preferentially aligned with any direction. With time, the tetrads' largest principal axis tends to become preferably aligned with gravity and the cosine of the angle that the intermediate and minor axes form with gravity decreases indicating a tendency toward a horizontal arrangement.

Figure 4.18 shows the alignment of the principal shape axes with the initial principal strain axes. At initialization, the tetrad shape is uncorrelated with the local particle field strain direction. As the particles in the tetrad adapt to local conditions, however, the major, intermediate, and minor principal axes begin to align with their respective counterparts of the principal strain axes. Eventually, near the onset of the diffusive behavior, the principal directions and strain become unaligned as the local conditions that generated the initial

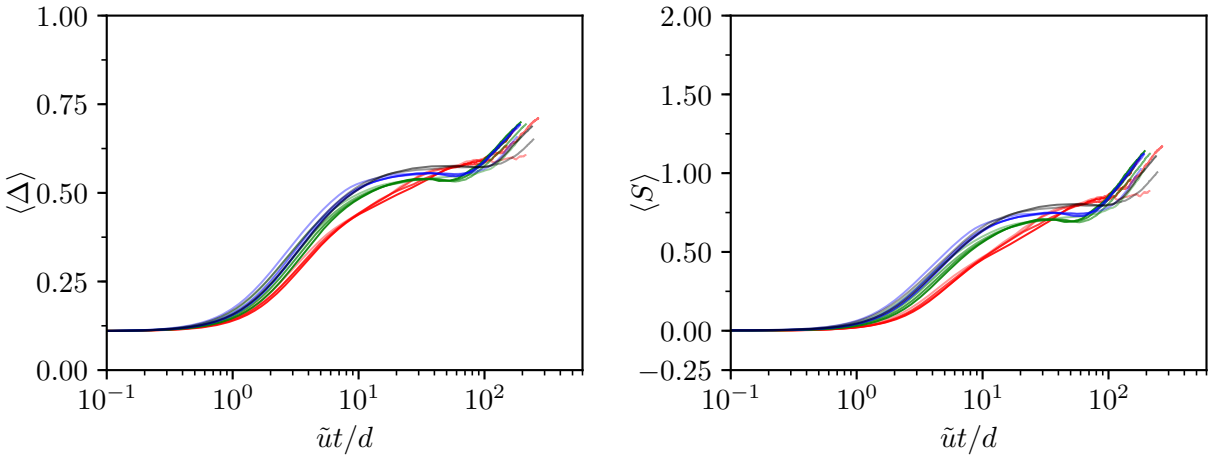


Figure 4.16: Time dependence of the shape variance Δ, (4.22), and shape factor S, (4.23); $\rho_p/\rho_f = 2$ (gray), 3.3 (green), 4 (red) and 5 (blue). Increasing hue saturation corresponds to increasing particle volume fraction (see Appendix B.1).

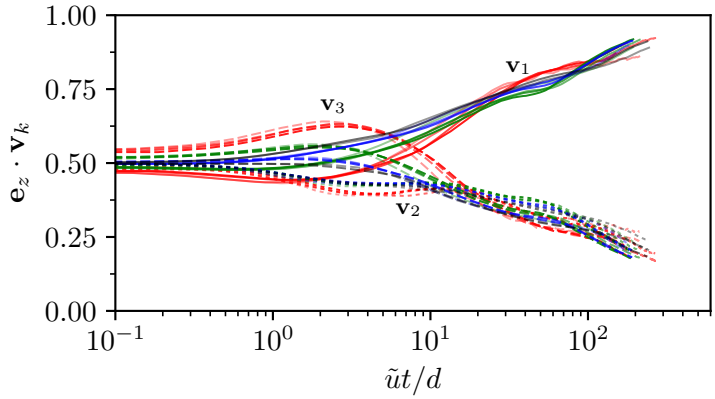


Figure 4.17: Time dependence of the alignment of the tetrad principal axes with gravity. Colors refer to different particle-to-fluid density ratios: $\rho_p/\rho_f = 2$ (gray), 3.3 (green), 4 (red) and 5 (blue). Increasing hue saturation corresponds to increasing particle volume fraction (see Appendix B.1).

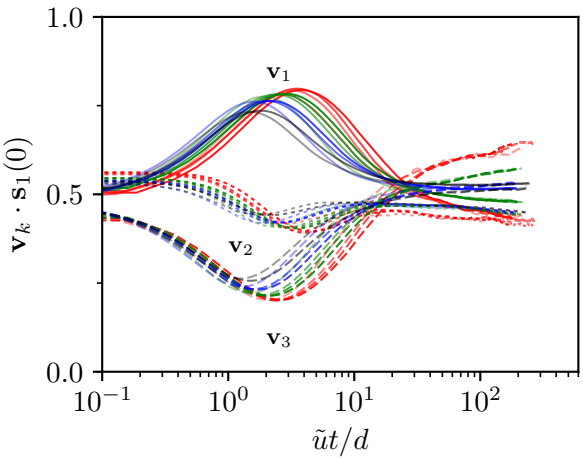


Figure 4.18: Time dependence of the alignment of the tetrad principal axes with the principal rate of strain at the initial instant corresponding to the largest eigenvalue. Colors refer to different particle-to-fluid density ratios: $\rho_p/\rho_f = 2$ (gray), 3.3 (green), 4 (red) and 5 (blue). Increasing hue saturation corresponds to increasing particle volume fraction (see Appendix B.1).

strain change.

4.3 Summary

This Chapter compared the results of our simulations with those of several previous investigators. By and large our results agree with previous ones for the quantities investigated earlier and the parameter ranges where there is an overlap. A new feature identified in this study is the tendency toward particles clustering as revealed by the large excursions above and below the mean value of the square of the mean particle velocity fluctuations. We also studied particle collisions, comparing the results with those of the kinetic theory of granular gases. Another feature of this work that has not been considered before is the time evolution of tetrads, arrangements of four particles. Initially the size and orientation of the tetrads is little affected by the fluid motion, but eventually they evolve into thin, elongated structures preferentially aligned with the direction of gravity. This result is in agreement

with the markedly anisotropic nature of the particle diffusivity and velocity fluctuations, which are larger in the vertical than in the horizontal direction. The transition toward an anisotropic structure occurs on the same time scale as the initial ballistic regime of the particles displacement is replaced by a diffusive regime.

Chapter 5

Many-GPU Implementation

The Physalis implementation used for the simulations studied in Chapters 3 and 4 was originally designed to use a graphics processing unit (GPU) to perform all of the computations required in the numerical solution (see Sierakowski and Prosperetti [96] and Sierakowski [95]). In particular, the authors took a 'GPU-centric' approach, where the GPU is used as the primary compute unit and the simulation proceeds exclusively on the GPU without any memory transfers to the central processing unit (CPU) except for file input and output. The GPU-centric model contrasts with the GPU-accelerated model, where the CPU offloads memory to the GPU for computationally intensive portions of the code. We note that the previous implementation of the method was limited to a single GPU, which constrained the total simulation size to that which would fit in the GPU device memory (a computational domain with around 300^3 grid cells containing several thousand particles for an Nvidia K-40 with 12 GB of device memory). In order to run larger simulations that capture a broader range of physical scales, the memory must be distributed among multiple GPUs; this in turn makes interprocess communication a requirement. The small GPU communication bandwidth is usually viewed as a bottleneck in distributed memory GPU simulations, especially in the solution of the Poisson equation for the fluid pressure (see, e.g., [4]). In this work, however, we demonstrate that this communication is not more expensive than computations

even at the largest test cases with 216 GPUs, which is sufficient for the simulation of, in total, a domain with 1920^3 grid cells and one million resolved particles.

We present in this chapter a distributed memory, GPU-parallelized, and GPU-centric extension of the Physalis method (described in [96] and [95]); we will use the term 'many-GPU' to encompass all of these characteristics. The implementation is open-source and available online at `http://PhysalisCFD.org`. In Chapter 2, we summarized the elements of the Physalis method that are necessary to set the stage for understanding the many-GPU implementation; more detailed information about the Physalis method can be found in the references. Sections 5.2 and 5.3 of this chapter describe the modifications to the original formulation necessary for the present many-GPU implementation. Finally, we validate and benchmark the many-GPU implementation in Section 5.4 of this chapter.

In the current implementation, we take advantage of several recent improvements to GPU technology. We employ Cuda-aware MPI to pass GPU device memory directly to the Message Passing Interface (MPI) implementation, allowing MPI to optimize communication based on the underlying network hardware. These optimizations include the Nvidia GPUDirect technologies that increase both the intranode and internode communication bandwidth (see, e.g., [109, 81]). We note that the use of GPUs has seen a dramatic rise in recent years, particularly in the area of machine learning, and both hardware (e.g., Nvidia, Mellanox Infiniband) and software (e.g., Nvidia, various MPI implementations) vendors are aware of the communication limitations between GPUs. It may be expected that future technological advances will further improve the communication efficiency of the methods that we present in this chapter.

5.1 The Physalis implementation

The relevant details of the Physalis method are presented in Chapter 2. To summarize, the method is implemented using the following algorithm, omitting any aspects related to distributing the particles among GPUs:

Step 1. Solve for the fluid fields in (2.1) and interpolate to the Lebedev quadrature nodes.

Step 2. Calculate the scalar products (2.3) using Lebedev quadrature (2.4).

Step 3. Solve for the Lamb coefficients in (2.2).

Step 4. Apply hydrodynamic forces and couples to the particles using the Lamb coefficients.

Step 5. Apply any lubrication and contact forces associated with particle interactions.

Step 6. Apply boundary conditions to the flow field using (2.2).

Step 7. Repeat until the Physalis solution (2.2) and global Navier-Stokes solution (2.1) have converged.

Step 8. Integrate the particle positions forward in time to finish the time step.

5.1.1 Poisson problem matrix symmetrization

The Physalis method supplies analytic particle boundary conditions for both the velocity and pressure fields. The addition of particle boundary conditions to the pressure Poisson equation changes the diagonal structure of the left-hand side of the discretized pressure Poisson equation (2.5). Sierakowski and Prosperetti [96] showed in their Figure A11 that the inclusion of the particle boundary conditions renders the matrix asymmetric and consequently they used the biconjugate-gradient stabilized algorithm to solve the system of linear equations. A colleague has recognized that the matrix can in fact be written symmetrically

[121], allowing the use of the simpler and more efficient conjugate gradient method. This is especially beneficial with the limited GPU device memory, as it uses less memory during the iterative process, thus increasing the total domain size possible.

Figure 5.1 modifies the work of [96] with the changes necessary to make the matrix symmetric. Previously, the matrix was only modified at the pressure nodes where the particle boundary conditions were applied. All diagonals for a given node were set to zero (shown in red in the figure) except the main diagonal which was set to one (shown in yellow in the figure); the right-hand side was then set to the analytically-calculated pressure boundary condition. In this work, we modify the rows of nodes adjacent to those at which an analytic boundary condition is known. The off-diagonal terms that would be multiplied by the boundary condition are zeroed (shown in green in the figure) and their influence is moved to the right-hand side of the equation. This has the effect of making the matrix symmetric.

5.2 Many-GPU flow solver implementation

As previously mentioned, the domain size in [96, 95] was limited by the amount of device memory on a single GPU. In order to run larger simulations, we need to distribute the data associated with the fluid fields across multiple GPUs. Distributed memory flow solvers using CPUs have existed for many years; however, there are few GPU implementations in the literature since memory communication bottlenecks are thought to be prohibitively expensive. Some examples of computational fluid dynamics codes implemented on GPUs include [78, 57, 122], though we note that almost all of these either do not solve the full Navier-Stokes equations, were developed before the advent of Cuda-aware MPI and GPUDirect technologies, are not fully utilizing a GPU-centric paradigm, or do not use a three-dimensional domain decomposition.

In developing the flow solver, we allow for a three-dimensional regular decomposition of

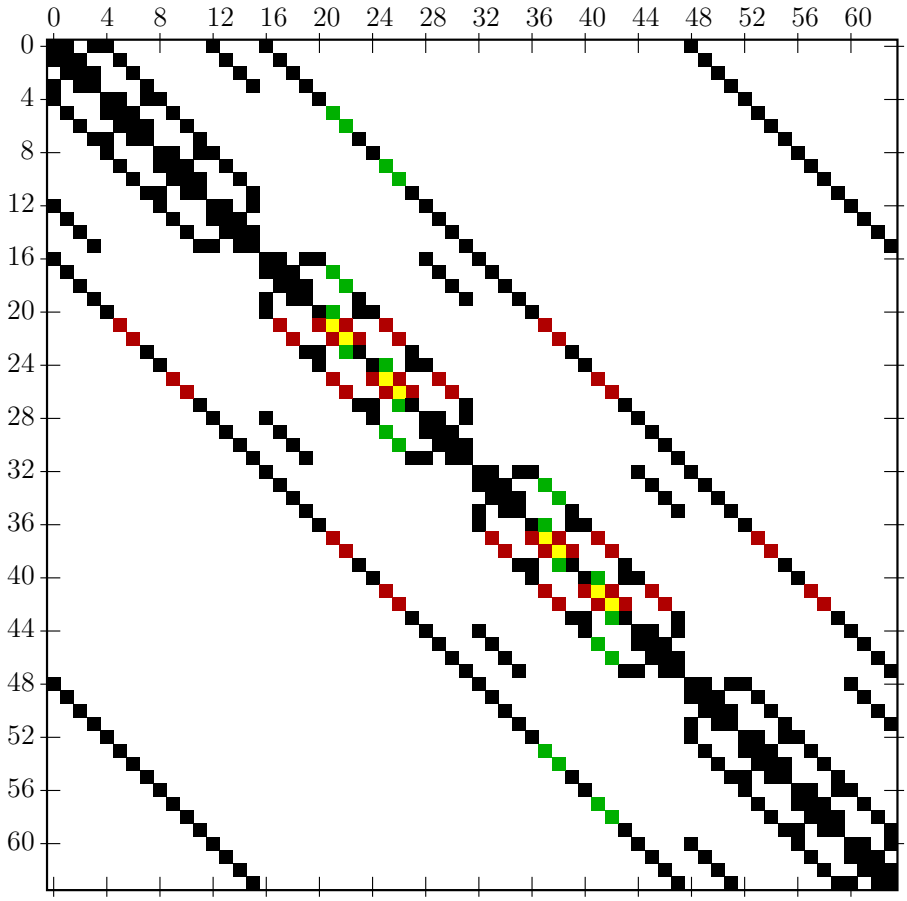


Figure 5.1: The pressure matrix for a triply-periodic domain of size $4 \times 4 \times 4$ with one (under-resolved) particle in the center. Black nodes correspond to the typical second-order finite difference stencil, yellow nodes are set equal to one, and red and green nodes are set equal to zero. The key difference between this and Figure A11 of [96] is the zeroing of the green nodes, which makes the matrix symmetric. The information at the green nodes is known from the particle boundary condition application at adjacent cells and is moved to the right-hand side of the equation.

the computational domain among the GPUs such as the one exemplified in two dimensions in Figure 5.2. The implementation is affected in such a way as to allow a regular rectilinear decomposition of the domain, including 'slab' or 'pencil' decompositions. Additionally, the relative arrangement of subdomains on the underlying hardware will very likely have an effect on the communication efficiency, but we have not taken any steps to automate this optimization.

5.2.1 Domain decomposition

We would like to draw attention to a few specific features of the decomposition of the computational domain into subdomains. As shown in Figure 5.2, we use three different levels of subdomain indexing to map the domain decomposition to the underlying computational hardware. Each subdomain has a set of Cartesian indices (black numbers in the figure) that describe its position in the overall decomposition and is mapped to a global MPI rank (blue number) used for MPI communication. The regular grid simplifies the MPI communication since each subdomain only communicates with one other in any given direction. Additionally, the MPI ranks are further mapped to a compute-node-local GPU rank (green number) that specifies which GPU on a compute node is used as the primary compute resource for the subdomain.

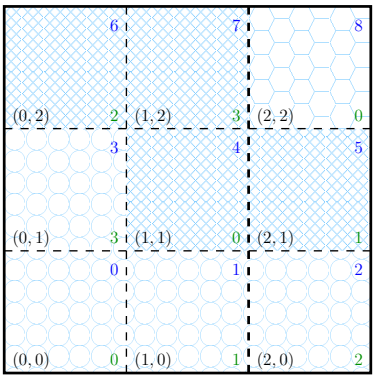


Figure 5.2: A two-dimensional abstraction of our three-dimensional domain decomposition in which the subdomains are drawn as dashed boxes in the global domain. In this example, we show a hypothetical set of compute nodes that each contain four GPUs and are denoted by various shadings. The black indices in the bottom-left of each subdomain represent the Cartesian indices of the subdomains in the global decomposition. These are mapped to the blue indices (upper-right), which identify the global MPI rank and are used for MPI communication. Each MPI rank is assigned a compute-node-local GPU rank (green indices, lower-right), which is used to specify which GPU on a node is used as the primary compute resource for the subdomain.

In order to properly perform certain finite difference operations associated with the Pois-

son equation (2.5), adjacent subdomains have to share fluid field data. For example, the one-dimensional second-order central finite difference approximation to the second derivative of p evaluated at x_i is

$$\left.\frac{\partial^2 p}{\partial x^2}\right|_i = \frac{p_{i-1} - 2p_i + p_{i+1}}{(\Delta x)^2}, \tag{5.1}$$

where Δx is the discretization length. Each subdomain is responsible for evaluating the stencil for all discrete locations i that lie within the subdomain. However, at the edge of the subdomain, one of p_{i+1} or p_{i-1} will be unknown since it belongs to an adjacent subdomain and so needs to be communicated to complete the stencil calculation. To facilitate this communication, we augment each subdomain by a one-cell-thick halo of ghost cells that contain copies of the necessary information in the adjacent subdomain. These 'internal' ghost cells are updated at every iteration of the conjugate gradient method.

5.2.2 Solution of the Poisson equation

As mentioned in Section 2 of this chapter, the matrix arising from the discretization of the flow domain is symmetric and positive definite. This allows us to employ the conjugate gradient algorithm to solve the Poisson equation (2.5). While algorithms for applying the conjugate gradient method to distributed memory matrices have existed for several decades (see [73, 31, 30]), to our knowledge very little work has been done implementing these algorithms in a GPU-centric manner. We present a brief overview of the conjugate gradient method, which will become useful in discussing our many-GPU implementation.

The discretization of (2.5) leads to a linear system of equations of the form

$$\mathbf{M}^{-1}\mathbf{A}\boldsymbol{\phi} = \mathbf{M}^{-1}\boldsymbol{b}, \tag{5.2}$$

where $\mathbf{M}$ is the preconditioner. In the current work, we use the Jacobi preconditioner, $\mathbf{M} = \text{diag}(\mathbf{A})$, which improves performance and is simple to generate. The conjugate gradient

method takes the following form (see, e.g., [91]). Note that in this algorithm, the discrete scalar product is defined by $(\boldsymbol{a}, \boldsymbol{b}) = \sum_i a_i b_i$.

Algorithm 1 Conjugate gradient

1: $\boldsymbol{\phi}_0 \leftarrow 0$ ▷ Set initial guess
2: $\boldsymbol{r}_0 \leftarrow \boldsymbol{b}$ ▷ Set initial residual
3: $\boldsymbol{z}_0 \leftarrow \mathbf{M}^{-1}\boldsymbol{r}_0$ ▷ Set auxiliary variable
4: $\boldsymbol{p}_0 \leftarrow \boldsymbol{z}_0$ ▷ Set descent direction
5: $q \leftarrow 0$ ▷ Set iteration count
6: **while** $(\boldsymbol{r}_q, \boldsymbol{z}_q) \geq R^2(\boldsymbol{b}, \boldsymbol{b})$; **do** ▷ Iterate to convergence at desired residual R
7: $\texttt{num1} \leftarrow (\boldsymbol{r}_q, \boldsymbol{z}_q)$ ▷ Reused from previous iteration
8: $\texttt{spmv} \leftarrow \mathbf{A}\boldsymbol{p}_q$ ▷ Sparse matrix-vector product
9: $\texttt{denom} \leftarrow (\boldsymbol{p}_q, \texttt{spmv})$ ▷ *Requires global reduction*
10: $\alpha \leftarrow \texttt{num1}/\texttt{denom}$
11: $\boldsymbol{\phi}_{q+1} \leftarrow \boldsymbol{\phi}_q + \alpha\boldsymbol{p}_q$
12: $\boldsymbol{r}_{q+1} \leftarrow \boldsymbol{r}_q - \alpha * \texttt{spmv}$
13: $\boldsymbol{z}_{q+1} \leftarrow \mathbf{M}^{-1}\boldsymbol{r}_q$
14: $\texttt{num2} \leftarrow (\boldsymbol{r}_{q+1}, \boldsymbol{z}_{q+1})$ ▷ *Requires global reduction*
15: $\beta \leftarrow \texttt{num2}/\texttt{num1}$
16: $\texttt{num1} \leftarrow \texttt{num2}$ ▷ Store for next iteration
17: $\boldsymbol{p}_{q+1} \leftarrow \boldsymbol{z}_{q+1} + \beta\boldsymbol{p}_q$ ▷ *Requires ghost-cell exchange*
18: $q \leftarrow q + 1$
19: **end while**

When implementing this algorithm for many GPUs, we must account for the unique characteristics of the hardware, namely the high throughput of floating-point operations and limited memory space. We choose to decrease memory usage at the expense of increased floating-point operations by applying a matrix-free sparse matrix-vector product. The matrix-free method calculates the diagonals of $\mathbf{A}$ as needed, rather than storing them in memory. This allows larger subdomains on each GPU and has been shown to be faster on GPUs than methods that store the matrix diagonals due to reduced global memory access [77].

We note that the conjugate gradient algorithm requires two global reductions to calculate scalar products and one neighbor ghost cell exchange, as shown in Steps 9, 14, and 17 of

Algorithm 1. Since the method is iterative, the notion is widespread in the literature that these communications are expensive and so have discouraged researchers from using GPUs for distributed memory flow solvers. However, the benchmark results in Section 5.4.3 of this chapter show that communication does not exceed computational time even for the largest simulations considered with 216 GPUs. At this point, we are able to simulate one million resolved particles in a domain size of 1920^3, which represents a significant increase in the capabilities of the original single-GPU implementation presented in [96]. It should be noted that these conclusions refer to a code, which, in its present form, does not overlap communication and computation. The addition of this feature will result in further improvements, a step that we leave for the future.

5.3 Many-GPU particle implementation

Many of the computational hardware constraints that informed our methods for decomposing the fluid domain also affect the solid phase. Unlike the fluid, which is a continuum, the solid phase takes the form of dispersed particles and so requires some additional considerations. For efficient memory usage and communication minimization, we distribute the particles spatially so that they reside in the GPU memory space of the fluid subdomain in which they are located. We draw special attention to the fact that these particles have finite size and they can therefore straddle subdomain boundaries. The first two subsections below discuss the mechanisms used to distribute and communicate particle data between subdomains.

Revisiting the Physalis description in Section 5.1 of this chapter, we identify three steps that are affected by the distribution of particles across GPUs. The final three subsections below discuss these issues in detail. For a particle that straddles subdomain boundaries, the interpolation of the fluid fields to the Lebedev nodes in Step 1 presents a problem (see, e.g., Figure 5.5). Second, the particle-particle interaction forces must be accumulated on

each particle near the subdomain boundaries after Step 5. Finally, when the solution has converged, the particle positions must be updated on all subdomains; subsequently, particles need to be added or removed from subdomains as appropriate after Step 8.

5.3.1 Particle distribution

When deciding precisely how to distribute the particles between subdomains, we aim to keep particle operations local to each GPU in order to limit communication and take full advantage of the GPU hardware. Were we to distribute particles according to, e.g., the location of their center, particle data would have to be communicated repeatedly in the course of the Lamb iteration, especially when determining particle-particle interactions. Due to MPI overhead, frequent communication of small amounts of data over the course of a time step should be avoided because it is less efficient than the communication of a larger amount of data once. Further, the efficiency of GPU memory access patterns would be limited by communications of this kind. We circumvent these issues by introducing ghost particles analogous to the ghost cells used to support stencil calculations in the fluid domain.

The longest length scale necessary to account for particle-particle interactions is the particle interaction length $\ell = 2a + \varepsilon$, with ε the lubrication force cutoff; ℓ is also used as the bin size in Chapter 2. These bins provide a convenient organizational structure to introduce ghost particles since they naturally identify the particles from adjacent subdomains that may be required for interactions with particles in a given subdomain. Figure 5.3 shows the bins overlaid on a coarse computational grid and Figure 5.4 demonstrates the criteria by which ghost particles are identified and communicated using ghost bins.

5.3.2 General particle communication method

Although there are several types of data associated with each particle (e.g., position, velocity, Lamb coefficients), each of them can be communicated in the same general manner. First,

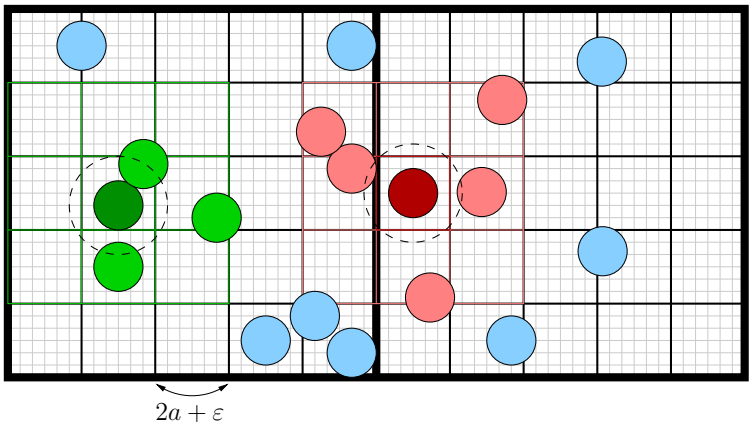


Figure 5.3: Bins (medium-thickness black lines) overlaid on a computational grid (thin gray lines). The thick black lines mark subdomain boundaries. Particle-particle interactions for a given particle (e.g., the dark green or dark red particle) only take place within the particle's bin and the 8 (in two dimensions, 26 in three dimensions) surrounding bins, which are highlighted in the particle's color. The dashed circles around the dark green and red particles show the interaction radii. The lighter green and red particles are considered for interaction with their darker counterparts.

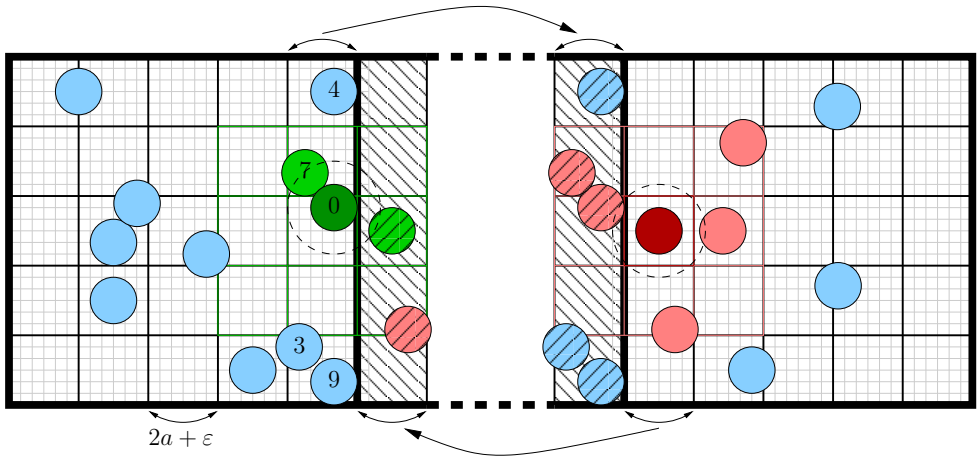


Figure 5.4: Example domain decomposition with the internal ghost bins, ghost particles, and ghost cells shown. The ghost particles and bins are marked by hashed patterns. The general particle communication algorithm for the numbered particles is described in Section 5.3.2 of this chapter.

we determine the amount of data that must be communicated and notify adjacent subdomains. Second, we pack into a contiguous array the unstructured particle data that needs to be communicated. Finally, this array is passed via MPI to the adjacent subdomain and

unpacked into the appropriate location. The full utilization of GPU hardware complicates these steps since they must be performed in parallel.

Data determination Once the particles are binned according to their position (see, e.g., [95, 50]), we perform a reduction (in parallel; see (A.1) in Appendix A.1) over the number of particles in each of the bins to be communicated to determine the total number of particles to be communicated. In the example in Figure 5.4, the numbered particles are transferred from the rightmost column of bins in the left subdomain to the column of (hashed) ghost bins in the right subdomain. The number of particles involved in the communication can be determined locally except for when the particle positions are updated, at which point the number must be communicated from the adjacent subdomain. In Figure 5.4, the number of particles that needs to be communicated from left to right is five.

Data packing Once the amount of data to be sent is determined, a contiguous sending array of the appropriate size is allocated. The particle data is then 'packed' into this array in such a way that each piece of data corresponding to a single particle is copied in parallel to the array without conflicting memory accesses. We accomplish this by performing an exclusive prefix scan (in parallel; see (A.2) and (A.3) in Appendix A.1) over the number of particles in each of the bins to be communicated. This operation results in a unique 'packing index' for each bin to which the bin can copy data without conflicting accesses from other bins.

In Figure 5.4, the exclusive prefix scan over the rightmost bin column in the left subdomain returns, starting from the bottommost bin, $[0, 2, 2, 3, 4]$. These are the packing indices for each bin; for example, the bin containing particle 7 will copy data to the sending array starting at an offset of 3 (the sending array is zero-indexed). If, in this example, the particle data is just the number of each particle, the sending array will contain the particle numbers

in the following order: $[9, 3, 0, 7, 4]$.

Data transfer and unpacking We now use MPI to communicate the sending array to the receiving subdomain, where it is copied into the correct memory location using the *Data packing* procedure above. At the end of this step, the ghost particles data (hashed) in the right subdomain matches that of the original (numbered) particles in the left subdomain in Figure 5.4.

5.3.3 Lebedev quadrature

To determine the Lamb coefficients A_{lm} and B_{lm} in (2.2), we interpolate the fluid fields to the Lebedev quadrature nodes $\boldsymbol{\psi}_i$ in (2.4). When a particle is very close to or straddles a subdomain boundary, however, some of the quadrature nodes may fall inside an adjacent subdomain, as illustrated in Figure 5.5. This presents a problem specifically because the quadrature nodes that fall outside the subdomain have no fluid data to interpolate. To deal with this situation, we split the full summation in (2.4) into partial summations that are performed on subdomain into which the nodes fall. The resulting partial sums are accumulated in the manner described in Section 5.3.2 of this chapter in such a way that the particle and all of its corresponding ghost particles have the correct full sum. This step is beneficial because it limits the amount of data communicated by performing the sum of the interpolated values before communication.

5.3.4 Interaction forces

Although we handle particle-particle interactions locally for each particle inside the subdomains, ghost particles need special consideration. For example, the solid dark green particle in the left subdomain in Figure 5.4 interacts with all its (light green) neighbors including the (hashed) ghost particle from the right subdomain. However, the solid dark red particle

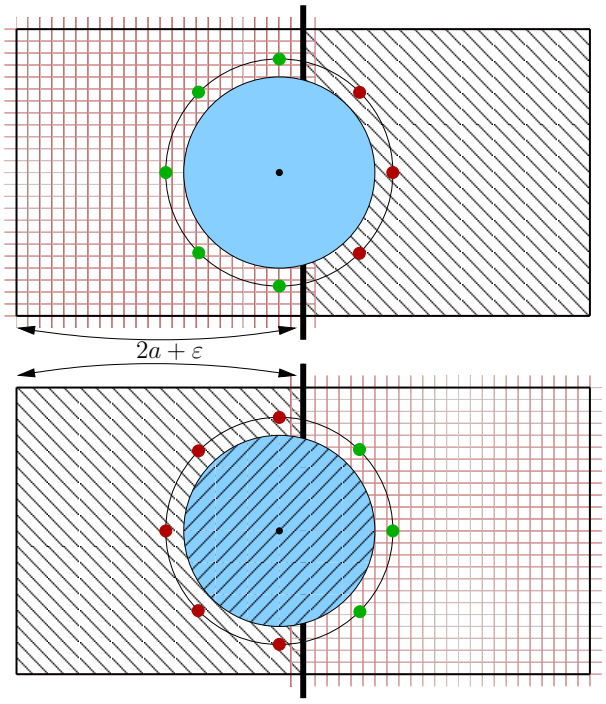


Figure 5.5: A two-dimensional abstraction of the Lebedev quadrature partial summation (note that the node locations are not representative of the actual positions used). The particle is assigned to the left subdomain (top image), but only five of the eight nodes are located within the subdomain. The remaining three nodes are located in the adjacent subdomain to the right (bottom image). The Lebedev quadrature sum (2.4) is divided into two contributions calculated in the two adjacent subdomains. The resulting partial sums are then accumulated between the particle and its ghost image.

in the right subdomain (which is a ghost particle in the left subdomain) also interacts with a particle to its right. This final interaction is not reflected in the calculations local to the left subdomain, so after the interaction forces are calculated on each particle within the subdomain, they must be communicated to the ghost particles.

5.3.5 Updating particle positions

At the end of each time step, the particle positions are integrated forward in time based on their new velocities, which are determined from the hydrodynamic and particle interaction forces. If a particle enters (or leaves) the outer bin planes, it must be added to (or removed

from) the appropriate adjacent subdomain. To do this, we perform the same general communication as above but pack all particle data (i.e., positions, velocities, Lamb coefficients, etc.) into the contiguous sending array. When the ghost particles arrive at their destination, we allocate a new particle array for all particles in the subdomain, taking into account the number of particles that have entered (or left) and fill the new array with the appropriate particles. Rather than communicating directly across subdomain edges and corners, we communicate in each of the three Cartesian directions sequentially. This method has the effect of moving particles to any of the 26 neighboring subdomains without needing explicit communication for the edge- and corner-adjacent subdomains.

5.4 Validation and benchmarking

We present validation of the new implementation of the many-GPU methods described above [113]. The first two subsections below use a turbulent channel flow and a three-dimensional Taylor-Green vortex to confirm that the distributed flow solver gives correct results. We investigate the strong and weak scaling properties of the flow and particle implementations in the following two subsections and benchmark the new implementation against the single-GPU version from [96] and [95] in the final subsection.

All of the computational resources in this section were provided by the Maryland Advanced Research Computing Center. This system has 72 GPU nodes with two Nvidia K-80s (usable as four Nvidia K-40s) and two 12-core 2.5 GHz Intel Haswell CPUs per node, and FDR-14 Infiniband interconnects. OpenMPI 2.1.2 was used for MPI communication [100].

5.4.1 Turbulent channel flow

A turbulent channel flow is a canonical problem in fluid dynamics, with many theoretical, experimental, and numerical results agreeing on the properties of the flow. The available

results provide a convenient means with which to validate our distributed flow solver implementation. The channel flow is driven in the periodic streamwise direction (x) by an applied pressure gradient, with solid no-slip walls in the wall-normal direction (y) and periodic boundary conditions in the spanwise direction (z). We use a non-equispaced grid in the three directions, with the smallest mesh length in the wall-normal direction and a somewhat longer mesh length in the other two directions. Details on the simulation parameters are shown in Table 5.1.

ρ_f	1
$-\partial p/\partial x$	1
$L_y/2$	1
$\nu = 1/Re_\tau$	0.005556
Domain size, L_x, L_y, L_z	4π, 2, 2π
Grid size, N_x, N_y, N_z	192, 360, 160
# of GPUs (decomposition)	4 ($2 \times 2 \times 1$)
Re_τ	180

Table 5.1: Parameters used for the turbulent channel flow simulation. The turbulent Reynolds number is defined by $Re_\tau \equiv u_\tau L_y/2\nu$, with $u_\tau = \sqrt{\tau_w/\rho_f}$ the friction velocity and $\tau_w = L_y(-\partial p/\partial x)/2$ the wall shear stress.

In wall units, we expect the mean velocity profile to be [83]:

$$u^+ = \begin{cases} y^+, & y^+ < 5; \\ \frac{1}{\kappa}\ln y^+ + B, & y^+ > 30,\ 2y/L_y < 0.3. \end{cases} \tag{5.3}$$

We initialize the flow field to the logarithmic profile in (5.3) with additional random solenoidal noise with a magnitude equal to 40% of the centerline velocity in a laminar channel flow with the same applied pressure gradient. After the dimensionless wall shear stress reaches a statistically steady state around unity according to the simulation parameters, we collect data for ten non-dimensional time units $tL_y/2u_\tau$ to determine the mean profile. Figure 5.6 shows our results; a curve fit to our data at $y^+ > 30$ gives $\kappa = 0.40$ and $B = 5.40$, which is in good agreement with accepted values (see e.g., [83]).

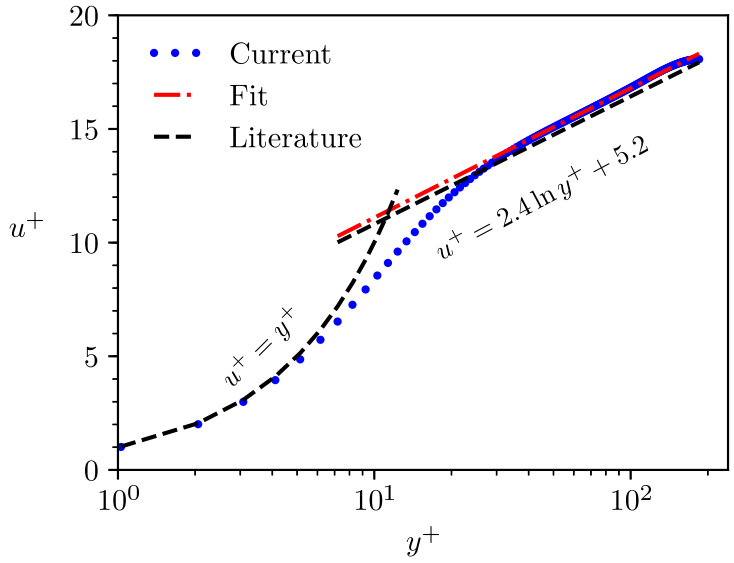


Figure 5.6: Comparison of current results (blue dots) with (5.3) (dashed black line and text). A fit at $y^+ > 30$ (red dashed-dot line) yields coefficients of $\kappa = 0.40$ and $B = 5.40$ in (5.3).

5.4.2 Three-dimensional Taylor-Green vortex

We further validate the distributed flow solver against a three-dimensional Taylor-Green vortex. This set-up is commonly used to evaluate the accuracy and performance of different numerical methods for solving the Navier-Stokes equations (2.1) by comparing against existing time-dependent computational results [3].

This validation test involves the viscous decay of the initial condition at $t = 0$:

$$u_0 = V_0 \sin\left(\frac{x}{L}\right) \cos\left(\frac{y}{L}\right) \cos\left(\frac{z}{L}\right), \tag{5.4a}$$

$$v_0 = -V_0 \cos\left(\frac{x}{L}\right) \sin\left(\frac{y}{L}\right) \cos\left(\frac{z}{L}\right), \tag{5.4b}$$

$$w_0 = 0, \tag{5.4c}$$

$$p_0 = \frac{\rho V_0^2}{16} \left(\cos\left(\frac{2x}{L}\right) + \cos\left(\frac{2y}{L}\right)\right) \left(\cos\left(\frac{2z}{L}\right)\right), \tag{5.4d}$$

where V_0 is a characteristic velocity scale and $-\pi L \leq x, y, z \leq \pi L$ defines the size of the cubic domain. The boundary conditions are periodic in all directions and the Reynolds

number $Re \equiv V_0 L/\nu$ is set to 1600. Several important parameters related to the decay of (5.4) include the temporal evolution of the kinetic energy,

$$E = \frac{1}{\Omega} \int_\Omega \tfrac{1}{2}\, \mathbf{u} \cdot \mathbf{u}\, d\Omega, \tag{5.5}$$

with Ω the volume of the computational domain, the temporal evolution of the kinetic energy dissipation rate calculated from the kinetic energy, dE/dt, and the temporal evolution of the enstrophy,

$$\varepsilon = \frac{1}{\Omega} \int_\Omega \tfrac{1}{2}\, \boldsymbol{\omega} \cdot \boldsymbol{\omega}\, d\Omega, \tag{5.6}$$

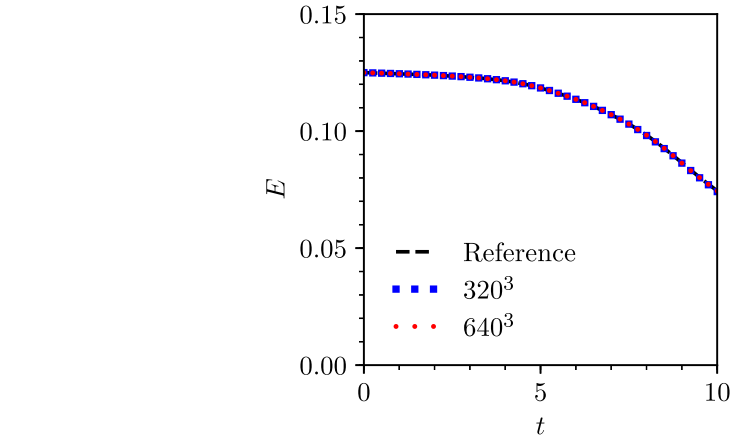
with $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ the vorticity. The kinetic energy dissipation rate is related to the enstrophy by $2\nu\varepsilon = -dE/dt$. Note that the enstrophy is expected to be more difficult to resolve than the kinetic energy dissipation rate [3].

Figure 5.7 compares our results using two different grid resolutions (320^3 and 640^3) to spectrally-accurate data. Both grids were decomposed into a $2 \times 2 \times 2$ arrangement of GPUs with each subdomain assigned to a different GPU. Overall, we see excellent agreement with the reference data, in particular with the finer grid in Figure 5.7c, which essentially coincide with the reference values. The maximum error in the enstrophy on the coarser grid is similar to the error on comparable grids reported in [3].

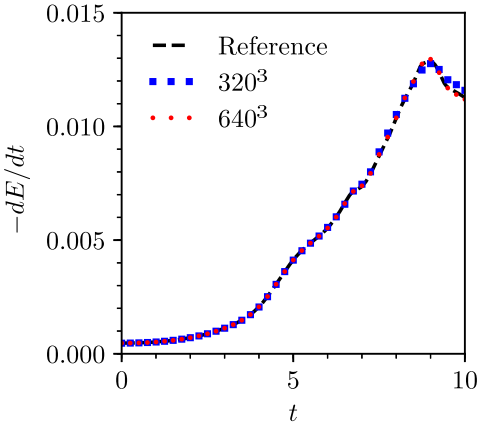
5.4.3 Distributed flow solver benchmark

In the previous subsections, we demonstrated that the implementation described in Sections 5.2 and 5.3 of this chapter gives accurate results for the test cases investigated. Here, we examine the strong and weak scaling behavior of the flow solver without the addition of particles. All simulations below are based on the three-dimensional Taylor-Green vortex validation case described in Section 5.4.2 of this chapter.

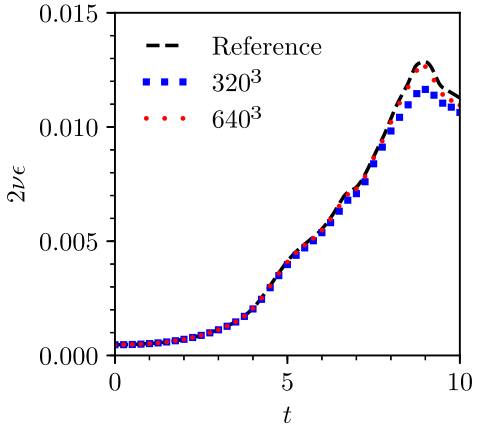
Strong scaling benchmarks quantify the effects of adding more processors (not necessarily GPUs) to a simulation with a constant total size with the expectation that the simulation



(a) Total kinetic energy E



(b) Dissipation dE/dt



(c) Enstrophy $2\nu\varepsilon = -dE/dt$

Figure 5.7: Results for the three-dimensional Taylor-Green vortex validation showing the reference data from [3] (black dashed line) and the data from the 320^3 and 640^3 grid resolutions (blue squares and red dots, respectively).

completes faster since the work per processor decreases. On the other hand, weak scaling benchmarks investigate the performance as the simulation grows in size proportional to the number of processors with the expectation that larger simulations can be performed in the same amount of time since the work per processor stays constant. Ideally, strong scaling tests should achieve an inversely proportional relationship between the number of processors and the time to solution, while weak scaling tests should result in a constant execution time as the simulation size and number of processors increase. Realistically, increased communication between processors causes deviations from these theoretical limits as their number increases.

Flow strong scaling

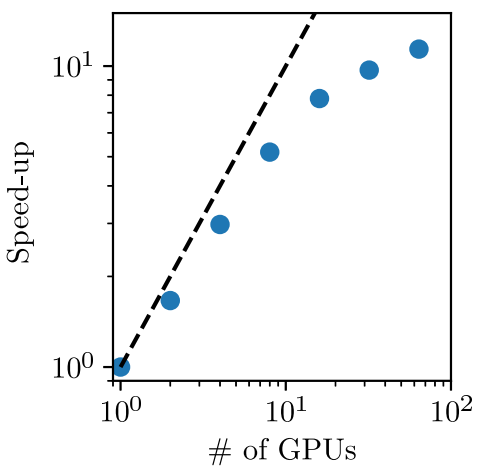
The simulations considered are shown in Table 5.2. In order to measure the speed-up relative to a single GPU, we average the time for each Poisson iteration over approximately 100 time steps for each simulation and plot the results in Figure 5.8a. As expected, adding more GPUs results in an overall speed-up proportional to the number of GPUs. The results show good strong scaling up to 16 GPUs, after which the scaling worsens. There are two factors contributing to the loss of performance, as illustrated in Figure 5.8b. The first is an increase in communication costs (hollow squares), which become increasingly significant as the number of GPUs increases. The second is that the computational speed-up (solid squares) sees diminishing returns. The GPU algorithms we use are based on those in [95] and were optimized to fully utilize the GPU hardware for large subdomain sizes. As such, they do not perform as well as the subdomain size decreases; improving these algorithms is a topic of future work.

Flow weak scaling

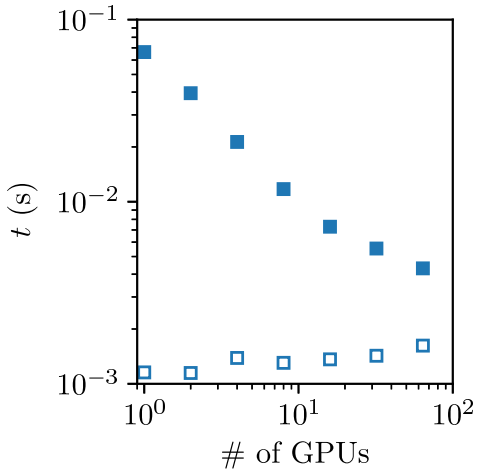
For weak scaling, we calculate the average time for each Poisson iteration as we expect that the time-per-iteration should stay constant except for an increase due to communication.

Number of GPUs	Domain decomposition	$N_x \times N_y \times N_z$ per GPU
1	$1 \times 1 \times 1$	$320 \times 320 \times 320$
2	$2 \times 1 \times 1$	$160 \times 320 \times 320$
4	$2 \times 1 \times 2$	$160 \times 320 \times 160$
8	$2 \times 2 \times 2$	$160 \times 160 \times 160$
16	$4 \times 2 \times 2$	$80 \times 160 \times 160$
32	$4 \times 2 \times 4$	$80 \times 160 \times 80$
64	$4 \times 4 \times 4$	$80 \times 80 \times 80$

Table 5.2: Simulations used in the strong scaling analysis. The global domain size is kept fixed at 320^3, which very nearly fills the 12 GB memory space on a single Nvidia K-40 GPU.



(a) Speed-up of the flow solver (solid circles) compared to perfect speed-up (dashed line).



(b) Comparison of the time spent communicating (hollow squares) with that spent computing (solid squares).

Figure 5.8: Strong scaling speed-up results for the flow solver. Increasing the number of GPUs decomposes the global domain in smaller portions, reducing the computation on each GPU and increasing the communication.

The simulations we perform are listed in Table 5.3 and the results are shown in Figure 5.9. As expected, the time spent computing is roughly constant over the range of GPUs used since the amount of work per GPU stays the same. The communication time increases with the number of GPUs used, but is still several times less than the computing time even for the largest simulations. The non-monotonic behavior of the communication time between 27 and 216 GPUs may be related to mapping of MPI processes to the underlying hardware. Since each compute node only has four GPUs, some nodes are not fully utilized for the simulations with 27 and 125 GPUs.

Number of GPUs	Domain decomposition	Global $N_x \times N_y \times N_z$
1	$1 \times 1 \times 1$	$320 \times 320 \times 320$
2	$2 \times 1 \times 1$	$640 \times 320 \times 320$
4	$2 \times 1 \times 2$	$640 \times 320 \times 640$
8	$2 \times 2 \times 2$	$640 \times 640 \times 640$
27	$3 \times 3 \times 3$	$960 \times 960 \times 960$
64	$4 \times 4 \times 4$	$1280 \times 1280 \times 1280$
125	$5 \times 5 \times 5$	$1600 \times 1600 \times 1600$
216	$6 \times 6 \times 6$	$1920 \times 1920 \times 1920$

Table 5.3: Parameters used for the weak scaling analysis. The global domain length and the grid size are kept constant with 320^3 grid cells per GPU.

5.4.4 Distributed particle implementation benchmark

We now study the behavior of the code when particles are added to the fluid. We arrange fixed particles (i.e., no translation or rotation) at the vertices of a simple cubic lattice; the number of particles N_p was chosen to be as close as possible to one of three desired volume fractions, $\phi = 0.01, 0.15$, and 0.30. Fixing the particles neither affects the scaling results nor reflects a limitation of the implementation. The particle communications described in Section 5.3.2 of this chapter occur during each time step without regard to whether the particles have moved or not, as the logic to do otherwise would inhibit the GPU parallelization. Additionally, we

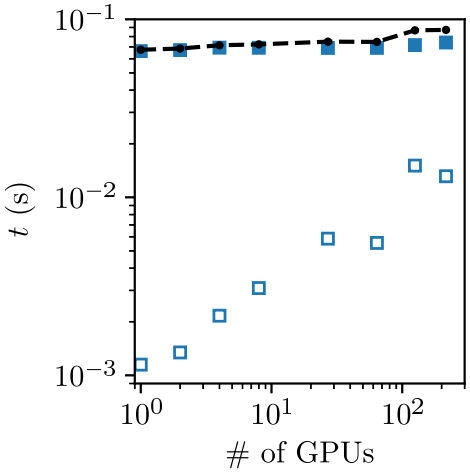


Figure 5.9: Comparison of the time spent communicating (hollow squares) with that spent computing (solid squares) for weak scaling of the flow solver. The total execution time is shown by the black dashed line with black circles.

have compared the number of particles expected to be involved in the communication (based on a random distribution of particles) with an estimate of the number actually involved in the communication and found that the two numbers are usually not significantly different. An exception is at the lowest volume fractions for small numbers of GPUs (less than eight), where the particle spacing is such that very few, if any, particles are communicated. In this range of parameters, however, particle communications are not expected to be a significant component of the total execution time.

We impose periodic boundary conditions on the computational domain and apply a pressure gradient with magnitude $a^3(\partial p/\partial z)/\mu\nu = 10$ to the fluid that causes the fluid to accelerate. The particle Reynolds number $Re_p \equiv 2au/\nu$ can be constructed using the characteristic velocity $u \sim \ell^2(\partial p/\partial z)/\mu$ based on the interparticle spacing ℓ. For the volume fractions $\phi = 0.01, 0.15$, and 0.30, the Reynolds number is approximately 55, 10, and 4, respectively. The particles are resolved with 8 grid cells per particle radius, which is a typical resolution demonstrated to be accurate by this method in the past [96, 95].

Particle strong scaling

For strong scaling tests, we begin with a domain size of 320^3 and decompose it as shown in Table 5.4. We place the particles in the domain as shown in Table 5.5.

Number of GPUs	Domain decomposition	$N_x \times N_y \times N_z$ per GPU
1	$1 \times 1 \times 1$	$320 \times 320 \times 320$
2	$2 \times 1 \times 1$	$160 \times 320 \times 320$
4	$2 \times 1 \times 2$	$160 \times 320 \times 160$
8	$2 \times 2 \times 2$	$160 \times 160 \times 160$
16	$4 \times 2 \times 2$	$80 \times 160 \times 160$
32	$4 \times 2 \times 4$	$80 \times 160 \times 80$
64	$4 \times 4 \times 4$	$80 \times 80 \times 80$

Table 5.4: Domain decomposition used in the particle strong scaling tests. The global domain size is kept fixed at 320^3.

	Array size	N_p	ϕ
$\phi \approx 0.01$	5^3	125	0.0082
$\phi \approx 0.15$	15^3	2197	0.1438
$\phi \approx 0.30$	17^3	4813	0.3216

Table 5.5: Particle simulation parameters for the particle strong scaling tests. The size of the cubic particle array, the number of particles N_p, and the actual volume fraction ϕ are shown.

The quantity of interest for strong scaling with particles is the execution time per Lamb iteration, which is comprised of a complete flow and particle solve (see Chapter 2 and [96]). This is shown as a speed-up relative to a single GPU in Figure 5.10. We see similar behavior as for the flow solver strong scaling in Section 5.4.3 of this chapter. An interesting difference is the continued improvement in speed-up for larger numbers of GPUs (at least 16 to 32) as the volume fraction increases. This occurs because the particle computations continue to scale well at larger number of GPUs, as opposed to the flow computations which were optimized for large subdomain sizes.

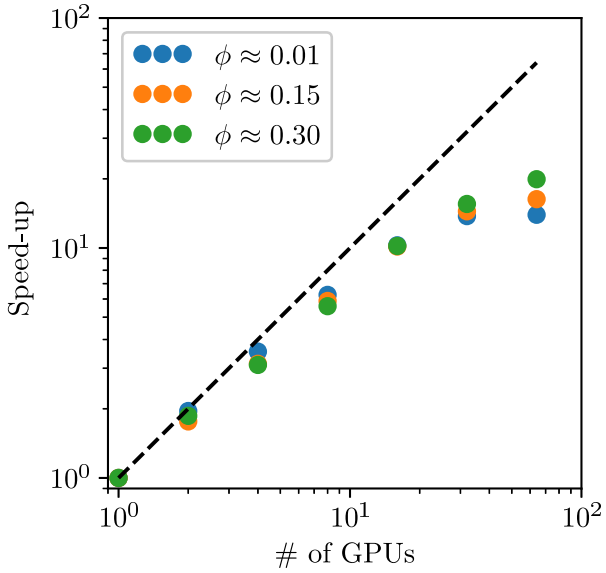


Figure 5.10: Strong scaling particle speed-up for three different volume fractions.

We can find more detail by looking at the profiling results in Figure 5.11, which shows the absolute time taken by each of the four major components of the simulation, namely communication and computation for both particle and flow operations. The most time-consuming component influencing the scaling is the flow computation (filled squares) for low volume fraction and the particle computation (filled circles) for high volume fraction. An unexpected feature is the decrease in time spent on flow computation as the volume fraction increases. This is not the result of the many-GPU implementation, but rather inherent in the nature of the Physalis algorithm. Indeed, imposing the Dirichlet condition on the particle surface for the pressure Poisson solver makes the Poisson equation easier to solve and so takes fewer iterations to converge to the solution. This also decreases the total amount of flow communications, which is reflected in the figure. Note that the flow computation profiling follows the same trend as discussed in Section 5.4.3 of this chapter.

The total time for both flow and particle communication does not exceed the total time for computation, although for larger numbers of GPUs the two are within an order of mag-

nitude. This is expected since the computational time decreases with increasing number of GPUs, whereas the communication time is roughly constant, although it is non-monotonic. There are several factors that may contribute to the non-monotonic behavior, including the network topology (e.g., due to an allocation of nodes that are 'far' from each other in the computer) and latent network traffic (due to other jobs on the machine using bandwidth on the interconnects), both of which were out of our control for the majority of these tests. Given their relative magnitude, we are not presently concerned with these fluctuations and instead focus on the overall trends. We additionally note that the time spent in particle communication, especially for the low volume fraction cases, approaches the microsecond resolution limit of the timing function used.

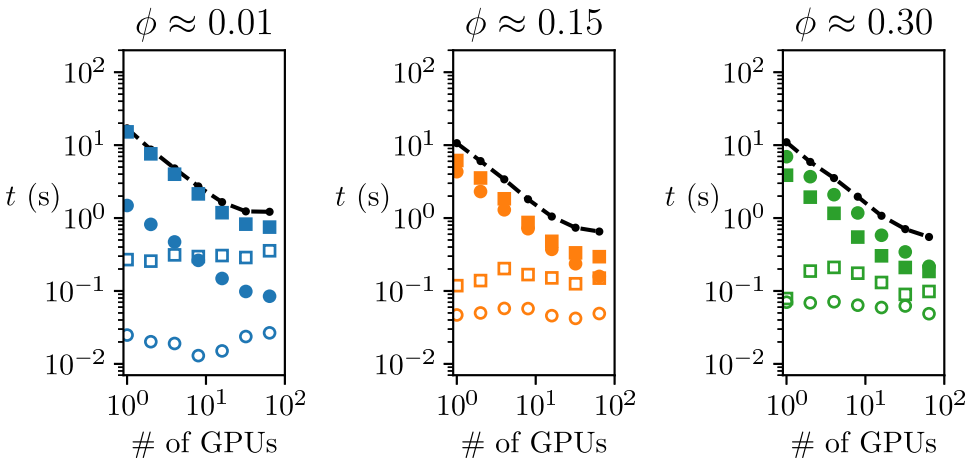


Figure 5.11: Profiling of the execution time for each of the calculation components in the particle strong scaling cases. The filled markers represent computation and the empty markers represent communication. Squares indicate flow operations and circles indicate particle operations. The total execution time is shown by the black dashed line with black circles.

Particle weak scaling

For weak scaling with particles, we perform simulations with a constant subdomain size of 320^3. The subdomain decompositions used are given in Table 5.6. Aside from the number of

particles and global domain size, we use the same physical parameters and applied pressure gradient as for the particle strong scaling tests.

	$\phi \approx 0.01$		$\phi \approx 0.15$		$\phi \approx 0.30$	
GPUs; decomp	N_p	ϕ	N_p	ϕ	N_p	ϕ
1; $1 \times 1 \times 1$	5^3	0.0082	13^3	0.144	17^3	0.322
2; $2 \times 1 \times 1$	2×5^3	0.0082	2×13^3	0.144	2×17^3	0.322
4; $2 \times 1 \times 2$	4×5^3	0.0082	4×13^3	0.144	4×17^3	0.322
8; $2 \times 2 \times 2$	11^3	0.0109	26^3	0.144	33^3	0.294
27; $3 \times 3 \times 3$	16^3	0.0099	40^3	0.155	50^3	0.303
64; $4 \times 4 \times 4$	21^3	0.0095	53^3	0.152	66^3	0.294
216; $6 \times 6 \times 6$	32^3	0.0099	79^3	0.149	100^3	0.303

Table 5.6: Simulations used in the particle weak scaling tests.

We again use the duration of a Lamb iteration as the basic unit of measure, although we note that the number of Poisson iterations per Lamb iteration varies with domain size and volume fraction. The execution time per Lamb iteration for each of the calculation components is shown in Figure 5.12, where we see that the time spent in both flow and particle computations is nearly constant as the number of GPUs increases. As for the communications, the time required is still small compared to the total time spent computing, although it increases with additional GPUs and is expected to eventually become more expensive than the computations.

5.4.5 Comparison to previous implementation

The single-GPU implementation was found to run approximately 60 times faster than a legacy serial CPU implementation [95]. To properly benchmark the many-GPU implementation, we begin by comparing it to the single-GPU implementation. Table 5.7 shows the results of three instances of the triply-periodic sedimentation simulations investigated in [115] using a particle-to-fluid density ratio of 5 and a volume fraction of 35% (2,000 particles). Using one GPU in the current implementation, each Lamb iteration requires 87% of the

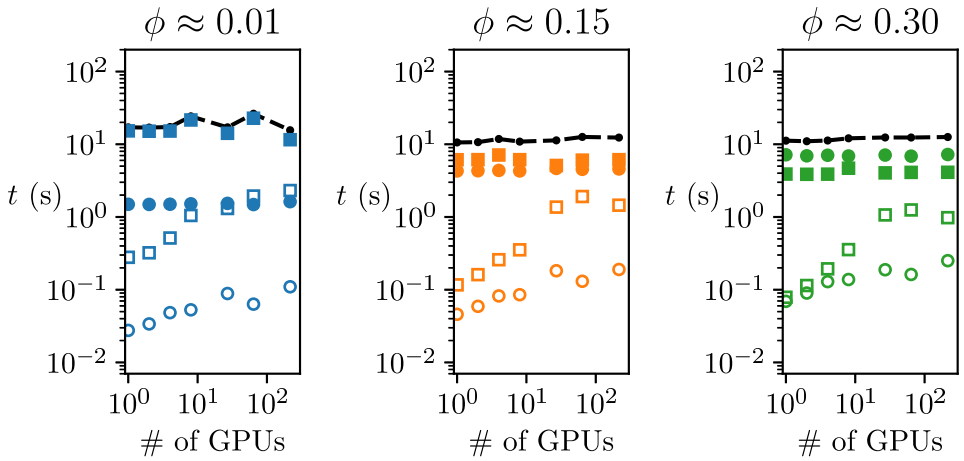


Figure 5.12: Profiling of the execution time for each of the calculation components in the particle weak scaling cases. The filled markers represent computation and the empty markers represent communication. Squares indicate flow operations and circles indicate particle operations. The total execution time is shown by the black dashed line with black circles.

time compared to the previous single-GPU implementation, indicating that even a one-to-one comparison shows modest improvement. This small difference is likely due to the switch from the biconjugate gradient stabilized algorithm to the conjugate gradient algorithm to solve the pressure Poisson equation. A single Lamb iteration for the same simulation using eight GPUs runs in 15% of the time compared to the previous single-GPU implementation and 18% of the time compared to the present GPU implementation using one GPU.

Implementation	# of GPUs	t (s)
Legacy	1	6.46
Current	1	5.59
Current	8	0.99

Table 5.7: Comparison of the execution time for one Lamb iteration with the legacy implementation [96, 95], the current implementation using a single GPU, and the current implementation using eight GPUs in a $2 \times 2 \times 2$ arrangement.

5.5 Summary

In this Chapter, we developed a procedure to distribute both fluid and particle memory across many GPUs with the aim of being able to simulate larger domain sizes with greater numbers of particles. To communicate between subdomains, we used internal ghost cells for the fluid fields and ghost particles for particles that are near to or straddle subdomain boundaries. An efficient method to package, transfer, and communicate ghost particle data across subdomains was described and developed.

The benchmarking results presented in this Chapter demonstrate that the new computational procedure achieves favorable weak scaling results for up to one million resolved particles in a domain size of 1920^3 using 216 GPUs, which was the largest number of GPUs available to us. We have shown that the communication associated with the fluid and particle operations is not prohibitively expensive at the domain sizes studied. Furthermore, the new implementation also achieves favorable strong scaling results.

Based on our reading of the literature, we approached the development with tempered expectations for the scaling results. However, the results indicate that it is very feasible to achieve encouraging performance for a distributed memory GPU-centric implementation even without carrying out detailed and extensive optimizations. Although there are several enhancements that could improve the scaling—e.g., communication and computation overlap, the optimization of the mapping between GPUs and MPI ranks, and the choice of MPI implementation and communication type—the current work represents a significant enhancement in the path towards larger resolved-particle simulations of fluid-particle systems using the Physalis method.

Chapter 6

Summary and Outlook

In this work, we have described simulations of fluidized-bed-like systems using the Physalis method. These simulations were described in detail in Chapters 3 and 4. Additionally, we extended a legacy single-GPU implementation of the Physalis method in Chapter 5 to take advantage of large many-GPU-enabled supercomputers, demonstrating scaling up to 216 GPUs including 1,000,000 resolved particles.

It may be hoped that results of Chapters 3 and 4 will help the formulation of useful reduced descriptions of particle-fluid systems. We have taken a small step in this direction in Chapter 3 in which we have shown that the averaged-equation theory for kinematic waves can be recovered from the results of resolved simulations. Much work remains to be done however. The smoothening out of the statistical fluctuations of resolved simulations, while leaving the effect of the most important physical processes intact, is still an unsolved problem. If an efficient and accurate method can be developed for this purpose, for example, it will be possible to examine in detail the proposed closure relations of averaged-equation models and judge their validity. By gradually increasing the particle density with respect to that of the fluid one may hope to gain an understanding of the nature of the instabilities affecting gas-particles fluidized beds and the manner in which these instabilities are contained in the reduced equations models. A valuable outcome of these efforts would be an understanding of

the correct way in which the many non-hyperbolic reduced-description models in existence should be rendered well-posed.

Further progress in this area would also be accomplished by performing simulations larger systems that contain more scales of the relevant physics. To this end, we described several modifications to the Physalis implementation originally presented in [96, 95] that enable the algorithm to be extended to more than a single GPU. The new computational procedure achieved favorable strong and weak scaling results and represents a significant improvement of the capabilities of the original single-GPU version.

6.1 Future Work

As mentioned in Chapter 1.1, resolved particle simulations using the immersed boundary method with tens or hundreds of thousands of particle are already common in the literature With the work performed in Chapter 5, the Physalis method is now capable of simulations of this size (provided adequate computational resources are available!), providing an alternative to the immersed boundary method. Preliminary examples of simulations of this scale using the Physalis method are shown in Figure 6.1. The left half shows 10,000 fixed particles immersed in a pressure-gradient driven flow; the fluid is colored according to the velocity magnitude. The right half of the Figure shows 50,000 particles sedimenting under gravity very soon after the settling starts; the color shows the fluid pressure. Already, 'rafts' of particles clusters can be seen, identified by an increased fluid pressure underneath the particles.

The Physalis method has also been extended to include resolved heat transfer between particles and fluid using the lumped capacitance approximation for the particles [110]. As Wang et al. [110] mention in their conclusion, the method used is easily extensible to other important quantities, such as mass diffusion involving dissolving or absorbent particles. Al-

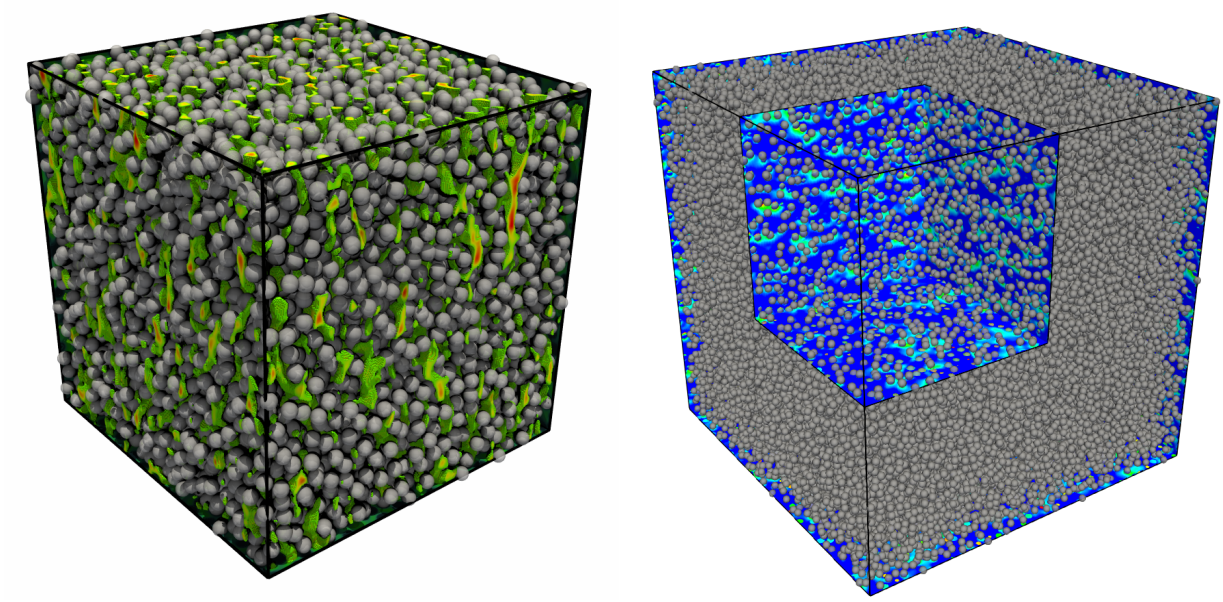


Figure 6.1: Many-GPU Physalis examples. The left image shows 10,000 fixed particles in a pressure-gradient driven flow, whereas the right image shows 50,000 particles sedimenting under gravity.

though neither of these physical scenarios have been integrated into the many-GPU implementation, the ability to simulate particle flows with the inclusion of heat and mass transfer would be a major step forward in understanding the complexity of these flows with multiple physics at play. A long-term goal would be the creation of a community of researchers based around using the code, with various groups running simulations and investigating the results and other groups involved in adding new features and improving the compute performance of the existing code. This type of community exists in several other scientific fields, e.g. the HOOMD-blue code used in molecular dynamics [5, 45].

The pace of scientific progress derived from simulations of particulate flows is, in some sense, directly related to simulation time. This was the initial motivation for the many-GPU development in Chapter 5—faster simulations would increase the turnaround time between conception of an idea and analysis of the results, while also allowing larger simulations in a reasonable amount of time. The near-term future of supercomputing, in particular

in relation to GPUs, is promising in both of these regards. For example, Nvidia recently announced the HGX-2 server, which includes 16 Nvidia V100 GPUs with 32GB of memory each, all connected with Nvidia NVSwitch interconnects [80]. This essentially acts as one large GPU, as the bandwidth between GPUs is estimated to be 2.4 TB/s. With the current many-GPU implementation, this would allow for simulations with close to 1.4 billion grid points ($\sim 1,118^3$) with little concern for communication requirements—at a volume fraction of 35% with 8 grid cells per radius, this corresponds to close to 230,000 particles. At a larger scale, Oak Ridge National Laboratory recently announced the Summit supercomputer, which has 4,608 compute nodes each with 6 Nvidia V100 GPUs with 16GB of memory per GPU, connected with 100 GB/s Infiniband interconnects [40]. Ignoring the effort needed to scale our implementation to run on all 27,648 GPUs, the realities of securing such a large reservation for long enough time to generate useful results, and storing the massive amounts of resulting data, such a computer would allow simulations with $\sim 10,650^3$ grid points. Based on the simulation parameters in Chapter 1.2, this would represent a cubic box almost 3 meters per side, which is a significant amount of physical space.

Of course, simulations of this size may not be practical for several reasons, including the feasibility of performing analysis of flow fields of this size. Rather, as previously mentioned, one could focus on improving the two-fluid models. As an example, we finish with an application of the Fourier reconstruction method introduced in Chapter 3. There, we reconstructed the particle volume fraction field and discovered evidence of continuity waves propagating vertically through the domain. Considering the one-dimensional continuity equation,

$$\frac{\partial \phi}{\partial t} + \frac{\partial (\phi w_z)}{\partial z} = 0 \ , \tag{6.1}$$

we can apply the same method by reconstructing the quantity ϕw_z. We show the left- and right-hand terms of (6.1) calculated with second order finite-differences of the expanded quantities in Figure 6.2. It can be seen that the fields are essentially the negative of each

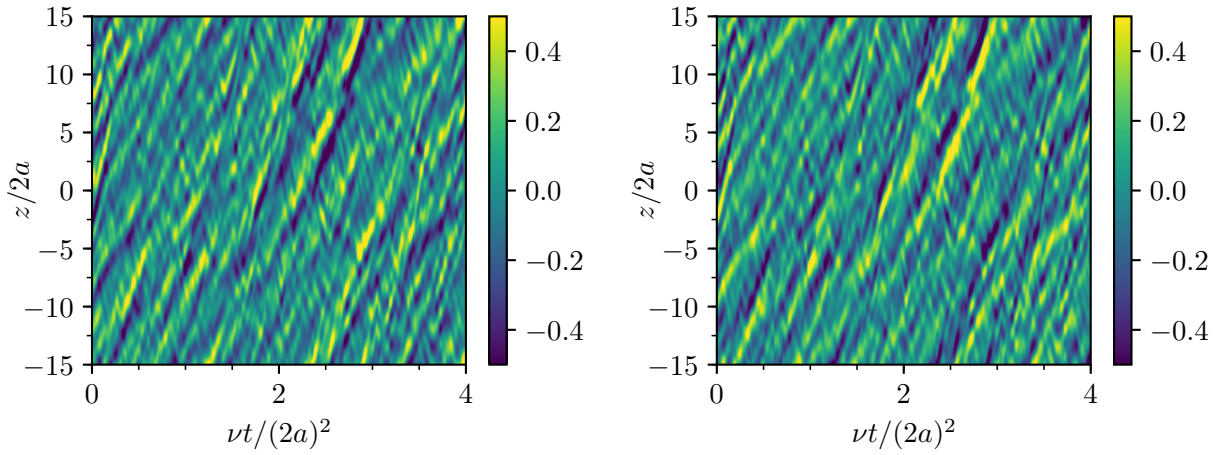


Figure 6.2: Fourier reconstruction of the one-dimensional continuity equation for the volume fraction for the simulation with $N_p = 500$ and $\rho^* = 3.3$. The left image shows the reconstruction of the first term of (6.1) using a second-order finite difference in time to determine the derivative. The right image shows the reconstruction of the second term in (6.1) using a second-order finite difference in space to determine the derivative. The relative root-mean-square error is 1.2%

other, as expected from (6.1). The relative root-mean-square error in reproducing (6.1) in this manner is 1.2%. Alternatively, since the z-dependence of the reconstruction is known analytically (in the forms of sines and cosines), we can take a derivative of the expansion for ϕw_z and determine the right-hand term in (6.1) without need for finite differences calculations for the spatial derivative. The accuracy reconstructing the continuity equation in this manner gives hope that more complex terms in the momentum equation can be examined in a similar fashion, allowing study of the closure terms needed to improve existing models.

Appendix A

A.1 Parallel Algorithms

The parallel reduction and exclusive prefix scan used in Chapter 5.3.2 are both performed using the Thrust [1] parallel algorithms library. Here, we give a brief overview of these algorithms.

A.1.1 Reduction

A reduction transforms the elements of an array into a single result using some predefined operator. In our work we use the addition operator, so the reduction is simply the sum of the elements. Consider a sequence of numbers a_i with n elements. The reduction of a_i is

$$\texttt{reduce}(a_i) = \sum_{i=1}^{n} a_i. \tag{A.1}$$

A.1.2 Exclusive prefix scan

Consider a sequence of numbers a_i with n elements. Then, the exclusive prefix scan of a_i is another sequence b_i of the same size defined by

$$b_j = \sum_{i=1}^{j-1} a_i \tag{A.2}$$

for $1 < j \leq n$; b_1 is 0. This can be written recursively as

$$b_i = b_{i-1} + a_i. \tag{A.3}$$

Appendix B

B.1 Additional Results for Chapter 4

Here we present some additional material to complement that presented in the main text of Chapter 4. In particular, we show graphs for all the quantities that we have calculated, only samples of which are given in Chapter 4.

B.2 Videos

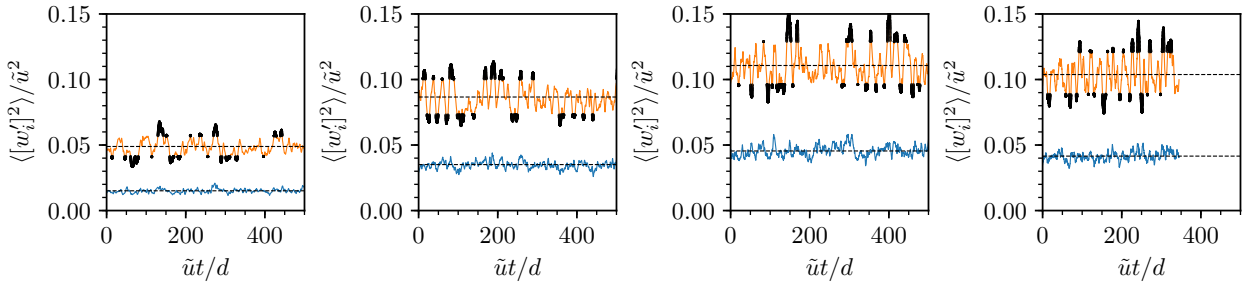
The video `500-rho33-full.avi` is available upon publication of [112] or by request to the author. The video `2000-rho33.avi` is available in the Supplementary Material of [115], see [2].

- `500-rho33-full.avi`: Sequence illustrating the motion of 500 spheres ($\phi = 8.7\%$) with a particle-to-fluid density $\rho_p/\rho_f = 3.3$. Color indicates the particle velocities; $t =$ 1 s corresponds to $t/\tau_p = 17.7$ and $\tilde{u}t/d = 63.8$. The total duration is $t/\tau_p = 377$ or $\tilde{u}t/d = 1{,}362.3$. The movie shows the entire simulation starting at $t = 0$. The initial transient lasts from $t = 0$ to $t/\tau_p = 7.1$ or $tildeut/d = 25.5$.

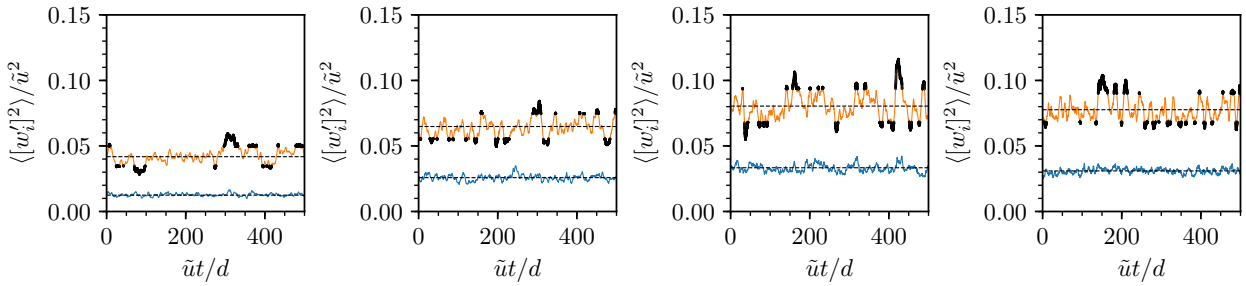
- `2000-rho33.avi`: Sequence illustrating the motion of 2000 spheres ($\phi = 34.9\%$) with a particle-to-fluid density $\rho_p/\rho_f = 3.3$; $t = 1$ s corresponds to $t/\tau_p = 13.0$ and $\tilde{u}t/d =$ 36.1. The movie shows a portion of the simulation lasting $t/\tau_p = 13$ or $\tilde{u}t/d = 36.1$

after steady state has been reached. The total duration of the simulation is $t/\tau_p = 197.4$ or $\tilde{u}t/d = 549.2$. Steady state is reached at $t/\tau_p = 7.1$ or $\tilde{u}t/d = 19.9$.

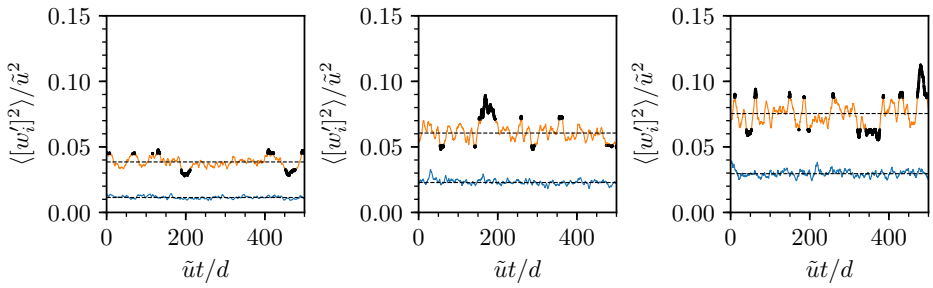
Particles appear and disappear across the boundaries of the computational domain due to the use of periodicity boundary conditions.



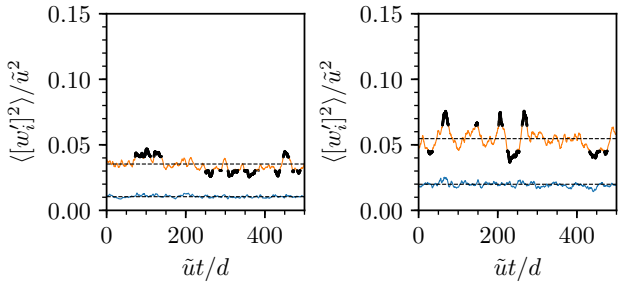
(a) $\phi = 0.087, \rho^* = 2.0$ (b) $\phi = 0.175, \rho^* = 2.0$ (c) $\phi = 0.262, \rho^* = 2.0$ (d) $\phi = 0.349, \rho^* = 2.0$



(e) $\phi = 0.087, \rho^* = 3.3$ (f) $\phi = 0.175, \rho^* = 3.3$ (g) $\phi = 0.262, \rho^* = 3.3$ (h) $\phi = 0.349, \rho^* = 3.3$

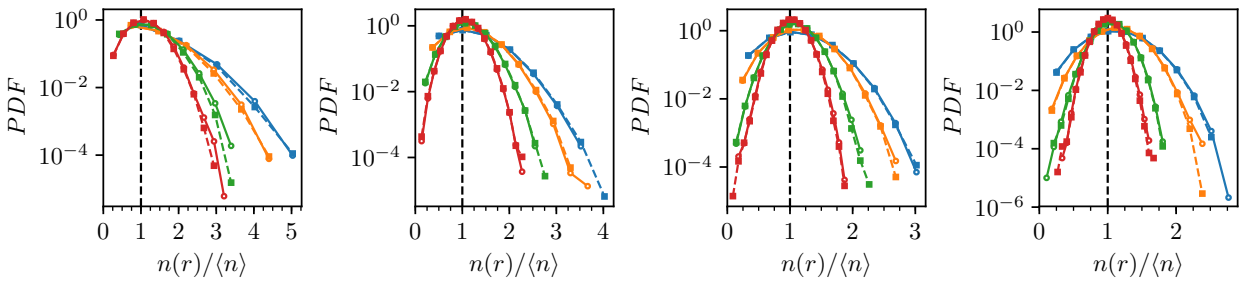


(i) $\phi = 0.087, \rho^* = 4.0$ (j) $\phi = 0.175, \rho^* = 4.0$ (k) $\phi = 0.262, \rho^* = 4.0$

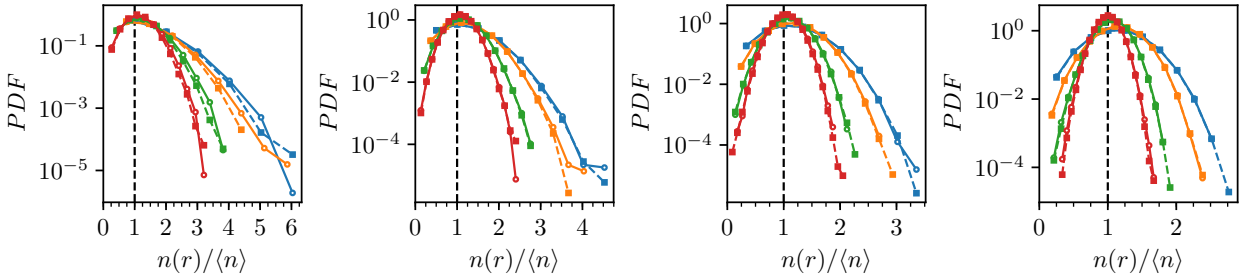


(l) $\phi = 0.087, \rho^* = 5.0$ (m) $\phi = 0.175, \rho^* = 5.0$

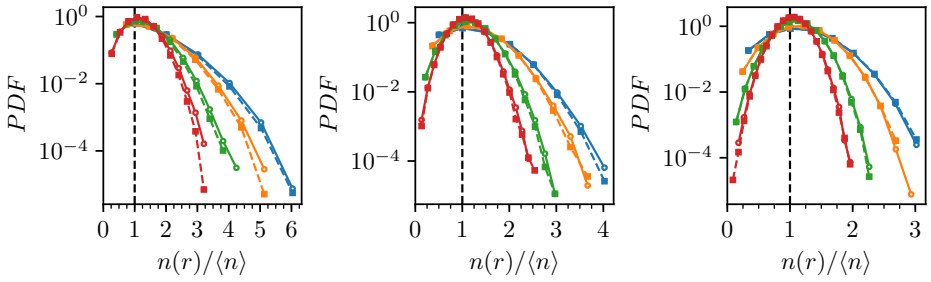
Figure B.1: Complete set of results for the square of the velocity fluctuations analogous to Figure 4.1 in Chapter 4.



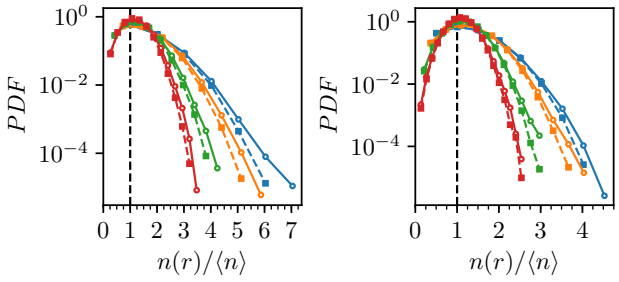
(a) $\phi = 0.087, \rho^* = 2.0$ (b) $\phi = 0.175, \rho^* = 2.0$ (c) $\phi = 0.262, \rho^* = 2.0$ (d) $\phi = 0.349, \rho^* = 2.0$



(e) $\phi = 0.087, \rho^* = 3.3$ (f) $\phi = 0.175, \rho^* = 3.3$ (g) $\phi = 0.262, \rho^* = 3.3$ (h) $\phi = 0.349, \rho^* = 3.3$

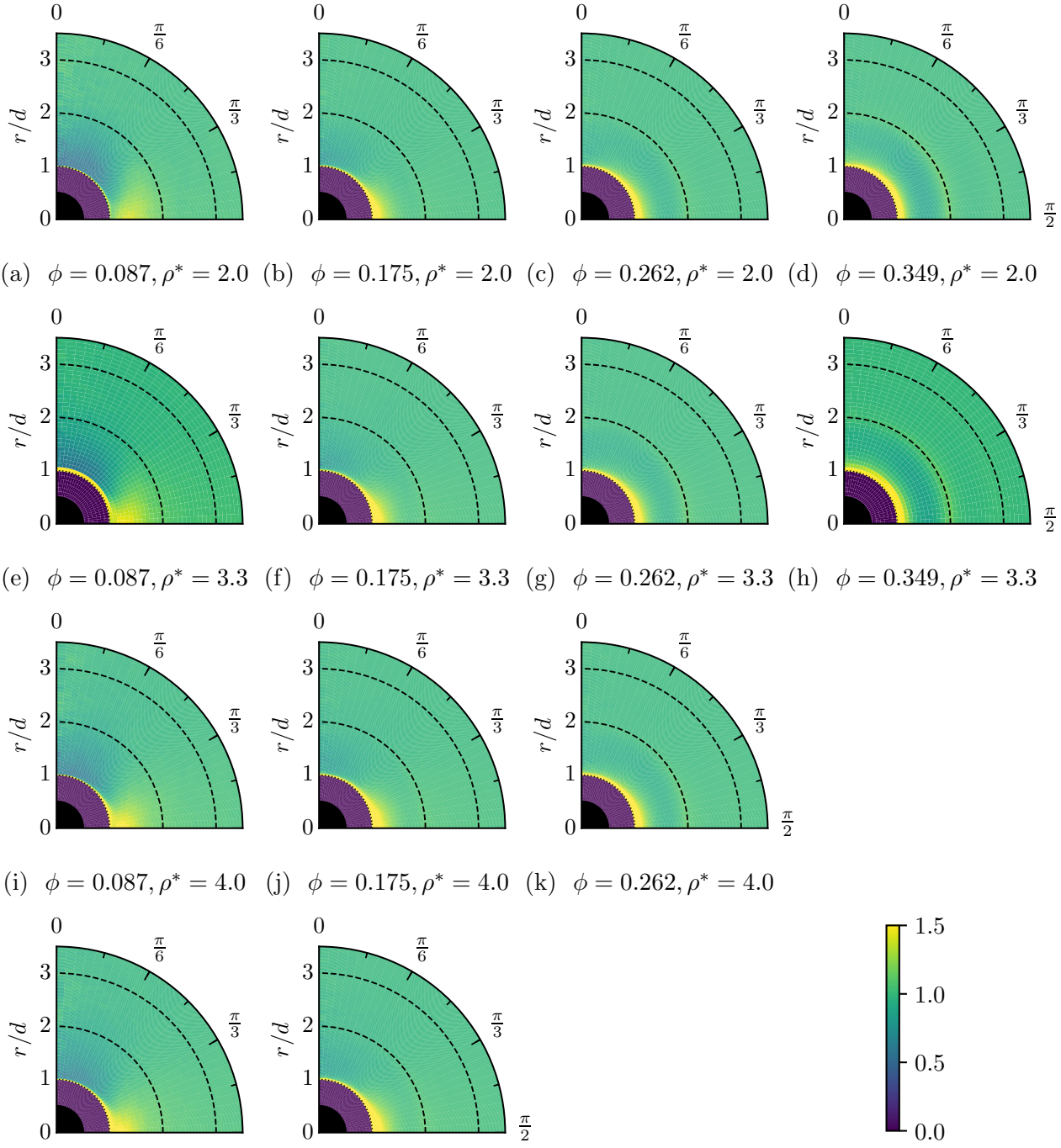


(i) $\phi = 0.087, \rho^* = 4.0$ (j) $\phi = 0.175, \rho^* = 4.0$ (k) $\phi = 0.262, \rho^* = 4.0$



(l) $\phi = 0.087, \rho^* = 5.0$ (m) $\phi = 0.175, \rho^* = 5.0$

Figure B.2: Complete set of results for the coordination number $n(r)/\langle n \rangle$ analogous to Figure 4.2 in Chapter 4.



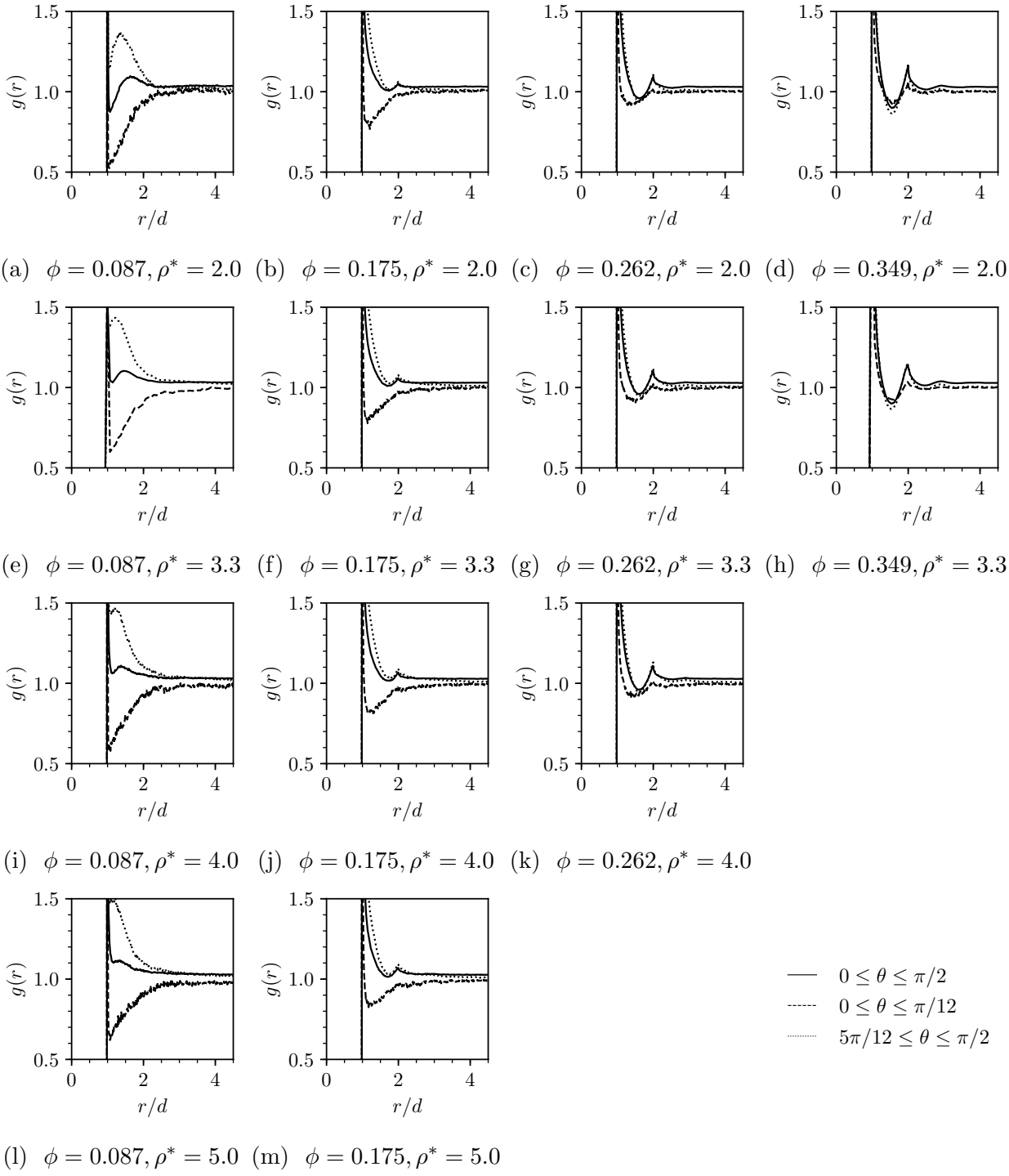
(a) $\phi = 0.087, \rho^* = 2.0$ (b) $\phi = 0.175, \rho^* = 2.0$ (c) $\phi = 0.262, \rho^* = 2.0$ (d) $\phi = 0.349, \rho^* = 2.0$

(e) $\phi = 0.087, \rho^* = 3.3$ (f) $\phi = 0.175, \rho^* = 3.3$ (g) $\phi = 0.262, \rho^* = 3.3$ (h) $\phi = 0.349, \rho^* = 3.3$

(i) $\phi = 0.087, \rho^* = 4.0$ (j) $\phi = 0.175, \rho^* = 4.0$ (k) $\phi = 0.262, \rho^* = 4.0$

(l) $\phi = 0.087, \rho^* = 5.0$ (m) $\phi = 0.175, \rho^* = 5.0$

Figure B.3: Pair distribution function $g(r, \theta)$ for all the simulations. The volume fraction increases from left-to-right and the particle-to-fluid density ratio from top-to-bottom increases. The color map legend is shown in the bottom right.



(a) $\phi = 0.087, \rho^* = 2.0$ (b) $\phi = 0.175, \rho^* = 2.0$ (c) $\phi = 0.262, \rho^* = 2.0$ (d) $\phi = 0.349, \rho^* = 2.0$

(e) $\phi = 0.087, \rho^* = 3.3$ (f) $\phi = 0.175, \rho^* = 3.3$ (g) $\phi = 0.262, \rho^* = 3.3$ (h) $\phi = 0.349, \rho^* = 3.3$

(i) $\phi = 0.087, \rho^* = 4.0$ (j) $\phi = 0.175, \rho^* = 4.0$ (k) $\phi = 0.262, \rho^* = 4.0$

(l) $\phi = 0.087, \rho^* = 5.0$ (m) $\phi = 0.175, \rho^* = 5.0$

Figure B.4: Radial distribution function $g(r)$ for all the simulations. The solid lines represent the average over the entire range $0 \leq \theta \leq \pi/2$, whereas the dotted and dashed lines represent averages over the ranges $0 \leq \theta \leq \pi/12$ and $5\pi/12 \leq \theta \leq \pi/2$, respectively. The volume fraction increases from left-to-right and the particle-to-fluid density ratio from top-to-bottom increases.

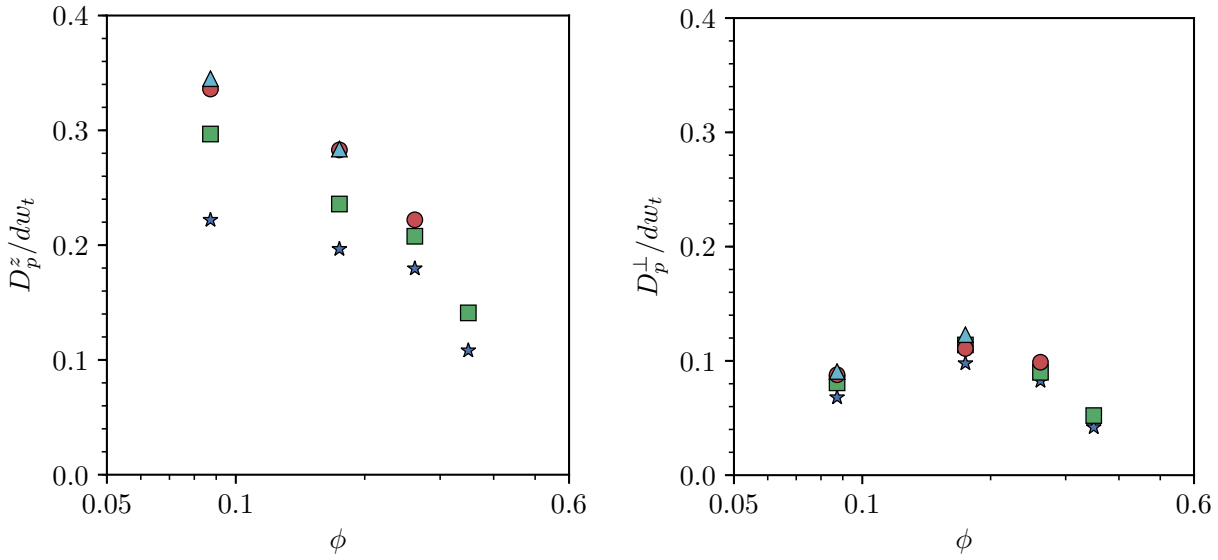


Figure B.5: The particle diffusivities shown in Figure 4.7 of Chapter 4 normalized by dw_t rather than $d\sqrt{\frac{1}{3}\mathbf{w}\cdot\mathbf{w}}$.

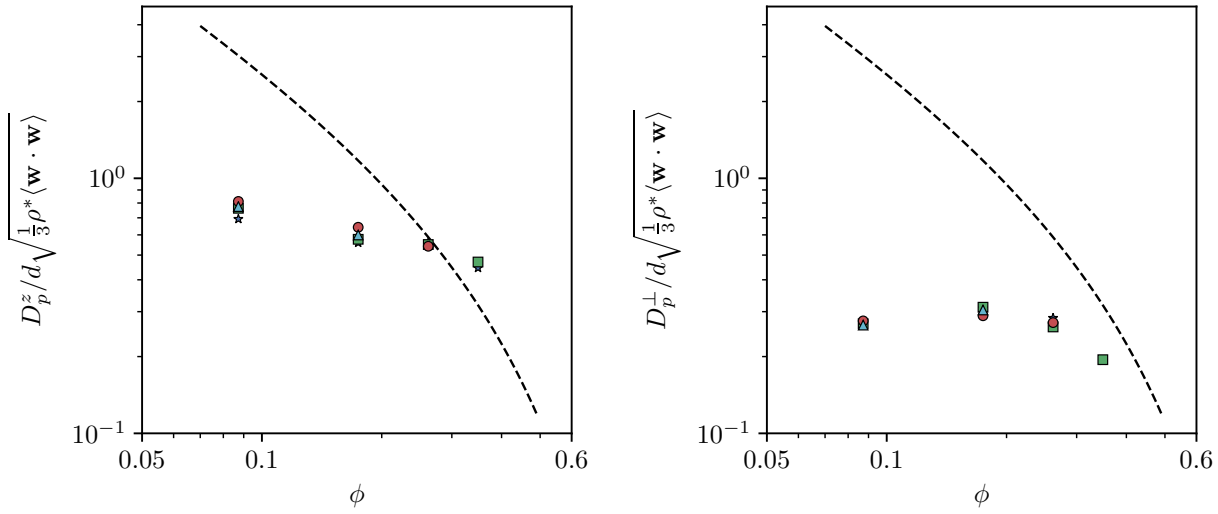
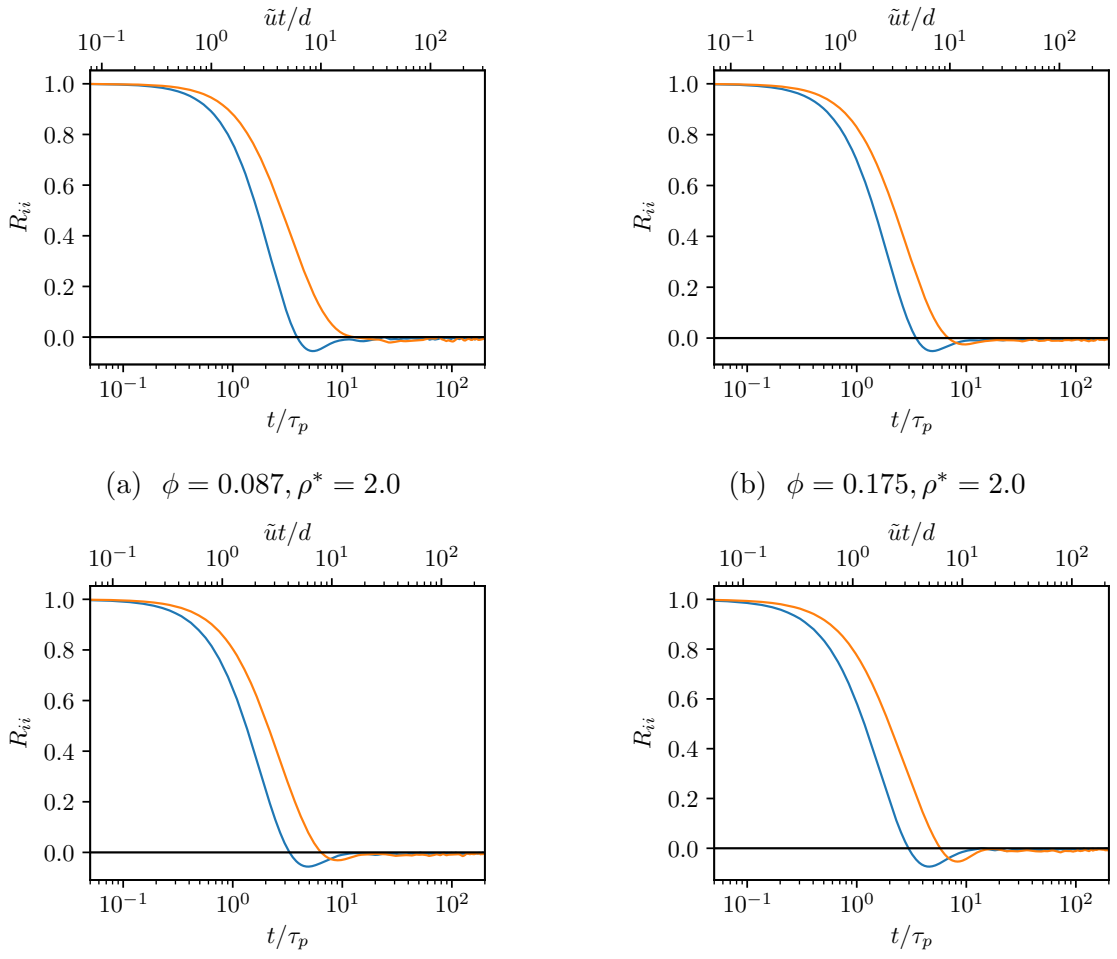


Figure B.6: The particle diffusivities normalized by division by $d\sqrt{\frac{1}{3}\rho^*\mathbf{w}\cdot\mathbf{w}}$ rather than $d\sqrt{\frac{1}{3}\mathbf{w}\cdot\mathbf{w}}$ as in Figure 4.7 of Chapter 4; here $\rho^* = \rho_p/\rho_f$.



(a) $\phi = 0.087, \rho^* = 2.0$

(b) $\phi = 0.175, \rho^* = 2.0$

(c) $\phi = 0.262, \rho^* = 2.0$

(d) $\phi = 0.349, \rho^* = 2.0$

Figure B.7: Particle velocity autocorrelation for a few cases beyond those shown in Figure 4.9 of Chapter 4. The information necessary to produce converged statistics was not available for the missing cases.

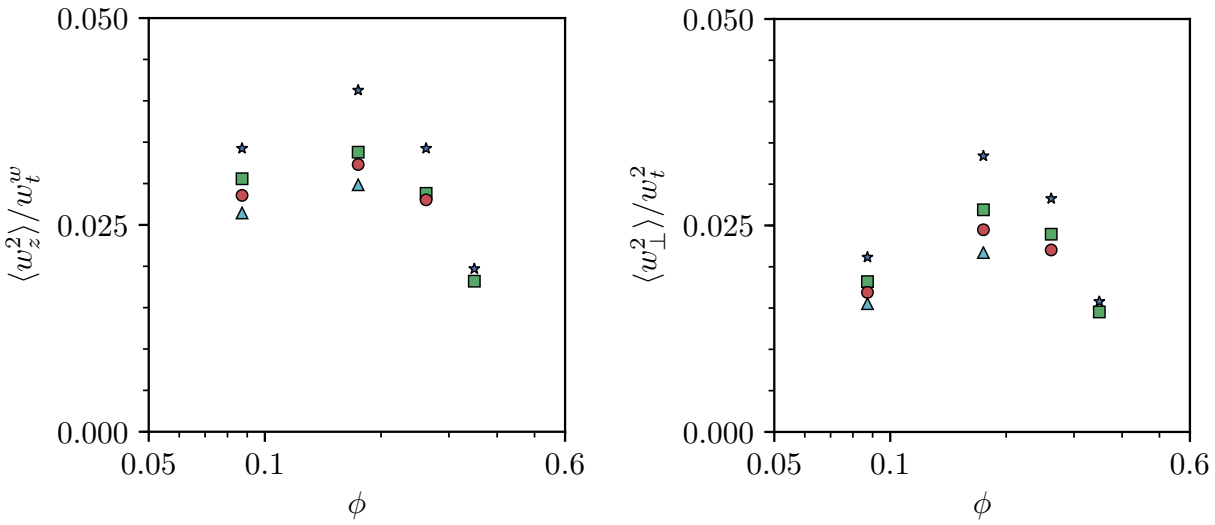


Figure B.8: The velocity fluctuations shown in Figure 4.13 of Chapter 4 normalized by the single-particle settling velocity w_t rather than $\tilde{u}$.

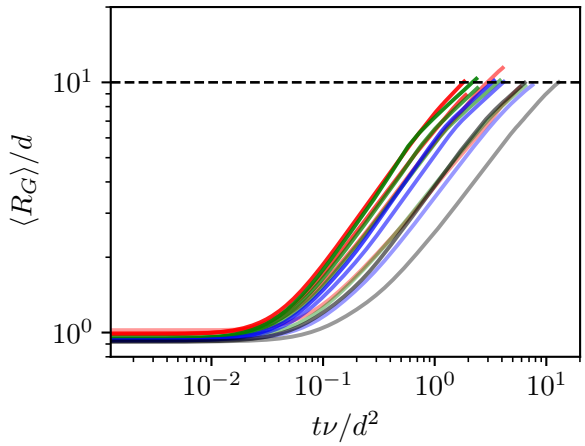


Figure B.9: Evolution of the average radius of gyration of the tetrads as a function of the dimensionless time $\nu t/d^2$ rather than $\tilde{u}t/d$ s in Figure 4.14 of Chapter 4. Colors refer to different particle-to-fluid density ratios: $\rho_p/\rho_f = 2$ (gray), 3.3 (green), 4 (red) and 5 (blue). The saturation of the hue increases with the particle volume fraction: lightest for $\phi = 8.9\%$, darkest for $\phi = 34.9\%$.

Bibliography

[1] Thrust, 2015. [Online]. Available: https://thrust.github.io.

[2] 2017. See http://link.aps.org/supplemental/10.1103/PhysRevFluids.2.114305 for a movie with two parallel sequences. The one on the left shows the results of the simulation; the sequence on the right shows the volume-fraction iso-surfaces corresponding to volume factions lower (blue) and higher (red) than the average, obtained with a Fourier reconstruction including 15 terms in the vertical direction and 5 terms in each one of the horizontal directions. The simulations are for 1000 particles with a density ratio of 3.3.

[3] *Problem C3.5 Direct Numerical Simulation of the Taylor-Green Vortex at Re = 1600*, 2012. AIAA First International Workshop on High-Order CFD Methods. [Online]. Available: http://dept.ku.edu/∼cfdku/hiocfd.html.

[4] Ament, M., Knittel, G., Weiskopf, D., and Strasser, W. A parallel preconditioned conjugate gradient solver for the poisson problem on a multi-GPU platform. In *2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing*, pages 583–592, 2010.

[5] Anderson, J. A., Lorenz, C. D., and Travesset, A. General purpose molecular dynamics simulations fully implemented on graphics processing units. *J. Comput. Phys.*, 227(10): 5342–5359, 2008.

[6] Anderson, K., Sundaresan, S., and Jackson, R. Instabilities and the formation of bubbles in fluidized beds. *J. Fluid Mech.*, 303:327, 1995.

[7] Anderson, T. and Jackson, R. A fluid mechanical description of fluidized beds: Comparison of theory and experiment. *Ind. Eng. Chem. Fund.*, 8:137, 1969.

[8] Ardekanio, M. N., Costa, P., Breugem, W. P., and Brandt, L. Numerical study of the sedimentaion of spheroidal particles. *Int. J. Multiphase Flow*, 87:16–34, 2016.

[9] Ardekanio, M. N., Asmar, L. A., Picano, F., and Brandt, L. Numerical study of heat transfer in laminar and turbulent pipe flow with finite-size spherical particles. *Int. J. Heat Fluid Fl.*, 71:189–199, 2018.

[10] Aronovitz, J. A. and Nelson, D. R. Universal features of polymer shapes. *J. Physique*, 47:1445–1456, 1986.

[11] Barnea, E. and Mizrahi, J. A generalized approach to the fluid dynamics of particulate systems: Part 1: General correlation for fluidization and sedimentation in solid multiparticle systems. *Chem. Eng. J.*, 5:171, 1973.

[12] Barnocky, G. and Davis, R. Elastohydrodynamic collision and rebound of spheres: experimental verification. *Phys. Fluids*, 31:1324, 1988.

[13] Batchelor, G. A new theory of the instability of a uniform bed. *J. Fluid Mech.*, 193: 75, 1988.

[14] Batchelor, G. Secondary instability of a gas-fluidized bed. *J. Fluid Mech.*, 257:357, 1993.

[15] Biferale, L., Boffetta, G., Celani, A., Devenish, B. J., Lanotte, A., and Toschi, F. Multiparticle dispersion in fully developed turbulence. *Phys. Fluids*, 17:111701, 2005.

[16] Brady, J. F. and Bossis, G. Stokesian dynamics. *Annu. Rev. Fluid Mech.*, 20:111–157, 1988.

[17] Brillantov, N. V. and Pöschel, T. *Kinetic Theory of Granular Gases*. Oxford University Press, 2004.

[18] Brown, D. L., Cortez, R., and Minion, M. L. Accurate projection methods for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 168(2):464–499, 2001.

[19] Caflisch, R. E. and Luke, J. H. C. Variance in the sedimenting speed of a suspension. *Phys. Fluids*, 28:759–760, 1985.

[20] Canuto, C., Hussaini, M., Quarteroni, A., and Zang, T. *Spectral Methods: Fundamentals in Single Domains*. Springer, Berlin, 2006.

[21] Chapman, S. and Cowling, S. T. G. *The Mathematical Theory of Non-Uniform Gases*. Cambridge University Press, 1970.

[22] Chehata Gómez, D., Bergougnoux, L., Guazzelli, É. J., and Hinch, E. J. Fluctuations and stratification in sedimentation of dilute suspensions of spheres. *Phys. Fluids*, 21: 093304, 2009.

[23] Chertkov, M., Pumir, A., and Shraiman, B. I. Lagrangian tetrad dynamics and the phenomenology of turbulence. *Phys. Fluids*, 11:2394–2410, 1999.

[24] Chong, Y., Ratkowsky, D., and Epstein, N. Effect of particle shape on hindered settling in creeping flow. *Powder Technol.*, 23:55, 1979.

[25] Clayton, C. T. *Multiphase Flow Handbook*. Taylor & Francis, 2006.

[26] Clift, R., Grace, J. R., and Weber, M. F. *Bubbles, Drops, and Particles*. Academic Press; reprinted by Dover, NY, 2005.

[27] Climent, E. and Maxey, M. R. Numerical simulations of random suspensions at finite Reynolds numbers. *Int. J. Multiphase Flow*, 29(4):579–601, 2003.

[28] Costa, P., Picano, F., Brandt, L., and Breugem, W. P. Effects of the finite particle size in turbulent wall-bounded flows of dense suspensions. *J. Fluid Mech.*, 843:450–478, 2018.

[29] Davis, R. H., Serayssol, J. M., and Hinch, E. J. The elastohydrodynamic collision of two spheres. *J. Fluid Mech.*, 163:479–497, 1986.

[30] D'Azevedo, E. and Romine, C. Reducing communication costs in the conjugate gradient algorithm on distributed memory multiprocessors. Technical Report ORNL/TM-12192, Oak Ridge National Lab, 09 1992.

[31] de Sturler, E. and van der Vorst, H. A. Reducing the effect of global communication in GMRES(m) and CG on parallel distributed memory computers. *Appl. Numer. Math.*, 18(4):441–459, 1995.

[32] Derksen, J. and Sundaresan, S. Direct numerical simulations of dense suspensions: wave instabilities in liquid-fluidized beds. *J. Fluid Mech.*, 587:303, 2007.

[33] di Felice, R. The sedimentation velocity of dilute suspensions of nearly monosized spheres. *Int. J. Multiphase Flow*, 25:559, 1999.

[34] di Felice, R. and Parodi, E. Wall effects on the sedimentation velocity of suspensions in viscous flow. *AlChE J.*, 42:927, 1996.

[35] Didwania, A. and Homsy, G. Flow regimes and flow transitions in liquid fluidized beds. *Int. J. Multiphase Flow*, 7:563, 1981.

[36] Duru, P. and Guazzelli, É. Experimental investigation on the secondary instability of liquid-fluidized beds and the formation of bubbles. *J. Fluid Mech.*, 470:359, 2002.

[37] Duru, P., Nicolas, M., Hinch, E. J., and Guazzelli, É. Constitutive laws in liquid-fluidized beds. *J. Fluid Mech.*, 452:371, 2002.

[38] El-Kaissy, M. and Homsy, G. Instability waves and the origin of bubbles in fluidized beds: Part I: Experiments. *Int. J. Multiphase Flow*, 2:379, 1976.

[39] Esteghamatian, A., Euzenat, F., Hammouti, A., Lance, M., and Wachs, A. A stochastic formulation for the drag force based on multiscale numerical simulation of fluidized beds. *Int. J. Multiphase Flow*, 99:363 – 382, 2018.

[40] Facility, O. R. N. L. L. C. Summit, June 2018. [Online]. Available: https://www.olcf.ornl.gov/summit/.

[41] Ferziger, J. H. and Perić, M. *Computational Methods for Fluid Dynamics*. Springer, 3rd edition, 2002.

[42] Fornari, W., Picano, F., and Brandt, L. Sedimentation of finite-size spheres in quiescent and turbulent environments. *J. Fluid Mech.*, 788:640–669, 2016.

[43] Fox, R. O. Large-eddy-simulation tools for multiphase flows. *Annu. Rev. Fluid Mech.*, 44:47–76, 2012.

[44] Garside, J. and Al-Dibouni, M. Velocity-voidage relationships for fluidization and sedimentation in solid-liquid systems. *Ind. Eng. Chem., Process Des. Dev.*, 16:206, 1977.

[45] Glaser, J., Nguyen, T. D., Anderson, J. A., Liu, P., Spiga, F., Millan, J. A., Morse,

D. C., and Glotzer, S. C. Strong scaling of general-purpose molecular dynamics simulations on GPUs. *Comput. Phys. Commun.*, 192:97–107, 2015.

[46] Glasser, B., Kevrekidis, I., and Sundaresan, S. One- and two-dimensional travelling wave solutions in gas-fluidized beds. *J. Fluid Mech*, 306:183, 1996.

[47] Glasser, B., Kevrekidis, I., and Sundaresan, S. Fully developed travelling wave solutions and bubble formation in fluidized beds. *J. Fluid Mech*, 334:157, 1997.

[48] Göz, M. Transverse instability of plane voidage wavetrains in gas-fluidized beds. *J. Fluid Mech.*, 303:55, 1995.

[49] Grabowski, W. W. and Wang, L. P. Growth of cloud droplets in a turbulent environment. *Annu. Rev. Fluid Mech.*, 45:293–324, 2013.

[50] Green, S. Particle simulation using Cuda. Technical report, Nvidia SDK Documentation, 2013.

[51] Guazzelli, É. and Hinch, E. J. Fluctuations and instability in sedimentation. *Annu. Rev. Fluid Mech.*, 43:97–116, 2011.

[52] Gudmundsson, K. and Prosperetti, A. Improved procedure for the computation of Lamb's coefficients in the Physalis method for particle simulation. *J. Comput. Phys.*, 234:44–59, 2013.

[53] Ham, J. M. and Homsy, G. M. Hindered settling and hydrodynamic dispersion in quiescent sedimenting suspensions. *Int. J. Multiphase Flow*, 14(5):533–546, 1988.

[54] Hamid, A., Molina, J. J., and Yamamoto, R. Sedimentation of non-Brownian spheres at high volume fractions. *Soft Matter*, 9:10056–10068, 2013.

[55] Hamid, A., Molina, J. J., and Yamamoto, R. Direct numerical simulations of sedimenting spherical particles at non-zero Reynolds number. *RSC Adv.*, 4:53681–53693, 2014.

[56] Jackson, R. *The Dynamics of Fluidized Particles.* Cambridge University Press, Cambridge, UK, 2000.

[57] Jacobsen, D., Thibault, J., and Senocak, I. An MPI-Cuda implementation for massively parallel incompressible flow computations on multi-GPU clusters. In *48th AIAA Aerospace Sciences Meeting*, 2010.

[58] Jones, A. V. and Prosperetti, A. On the suitability of first-order differential models for two-phase flow prediction. *Int. J. Multiphase Flow*, 11:133–148, 1985.

[59] Joshi, J. B. and Nandakumar, K. Computational modeling of multiphase reactors. *Annu. Rev. Chem. Biom. Eng.*, 6:347–378, 2015.

[60] Kazerooni, H. T., Fornari, W., Hussong, J., and Brandt, L. Inertial migration in dilute and semidilute suspensions of rigid particles in laminar square duct flow. *Phys. Rev. Fluids*, 2:084301, 2017.

[61] Kempe, T. and Frölich, J. Collision modelling for the interface-resolved simulation of spherical particles in viscous fluids. *J. Fluid Mech.*, 709:445–489, 2012.

[62] Kempe, T. and Frölich, J. An improved immersed boundary method with direct forcing for the simulation of particle laden flows. *J. Comput. Phys.*, 231:3663–3684, 2012.

[63] Kidanemariam, A. G. and Uhlmann, M. Interface-resolved direct numerical simulation of the erosion of a sediment bed sheared by laminar channel flow. *Int. J. Multiphase Flow*, 67:174–188, 2014.

[64] Kidanemariam, A. G. and Uhlmann, M. Direct numerical simulation of pattern formation in subaqueous sediment. *J. Fluid Mech.*, 750:R2, 2014.

[65] Kidanemariam, A. G. and Uhlmann, M. Formation of sediment patterns in channel flow: minimal unstable systems and their temporal evolution. *J. Fluid Mech.*, 818: 716–743, 2017.

[66] Kim, S. and Karilla, S. *Microhydrodynamics: Principles and Selected Applications*. Butterworth-Heinemann, Boston, MA; reprinted by Dover, New York, NY, 1991.

[67] Kynch, G. A theory of sedimentation. *Trans. Faraday Soc.*, 48:166, 1952.

[68] Ladd, A. Dynamical simulations of sedimenting spheres. *Phys. Fluids A-Fluid*, 5(2): 299–310, 1993.

[69] Lamb, H. *Hydrodynamics*. Cambridge University Press, Cambridge, UK; reprinted by Dover, New York, NY, 1932.

[70] Lashgari, I., Ardekani, M. N., Banerjee, I., Russom, A., and Brandt, L. Inertial migration of spherical and oblate particles in straight ducts. *J. Fluid Mech.*, 819: 540–561, 2017.

[71] Lashgari, I., Picano, F., Costa, P., Breugem, W. P., and Brandt, L. Turbulent channel flow of a dense binary mixture of rigid particles. *J. Fluid Mech.*, 818:623–645, 2017.

[72] Lebedev, V. I. Quadratures on a sphere. *USSR Comput. Math. Math. Phys.*, 16(2): 10–24, 1976.

[73] Lewis, J. G. and van de Geijn, R. A. Distributed memory matrix-vector multiplication and conjugate gradient algorithms. In *Supercomputing '93. Proceedings*, pages 484–492, 1993.

[74] Mavrantzas, V. G. and Theodorou, D. N. Atomistic simulation of polymer melt elasticity: Calculation of the free energy of an oriented polymer melt. *Macromolecules*, 31: 6310–6332, 1998.

[75] Maxey, M. R. Simulation method for particulate flows and concentrated suspensions. *Annu. Rev. Fluid Mech.*, 49:171–193, 2017.

[76] Mittal, R. and Iaccarino, G. Immersed boundary methods. *Ann. Rev. Fluid Mech.*, 37:23926, 2005.

[77] Müller, E., Guo, X., Scheichl, R., and Shi, S. Matrix-free GPU implementation of a preconditioned conjugate gradient solver for anisotropic elliptic PDEs. *Comput. Visual Sci.*, 16:41–58, 2013.

[78] Müller, E., Scheichl, R., and Vainikko, E. Petascale solvers for anisotropic pdes in atmospheric modelling on GPU clusters. *Parallel Comput.*, 50:53–69, 2015.

[79] Nicolai, H., Herzhaft, B., Hinch, E. J., Oger, L., and Guazzelli, É. Particle velocity fluctuations and hydrodynamic self-diffusion of sedimenting non-Brownian spheres. *Phys. Fluids*, 7(1):12–23, 1995.

[80] Nvidia. Nvidia HGX-2, June 2018. [Online]. Available: https://https://www.nvidia.com/en-us/data-center/hgx/.

[81] Nvidia Developer Zone. Nvidia GPUDirect, December 2017. [Online]. Available: https://developer.nvidia.com/gpudirect.

[82] Peskin, C. S. Numerical analysis of blood flow in the heart. *J. Comput. Phys.*, 25: 220252, 1977.

[83] Pope, S. *Turbulent Flows*. Cambridge University Press, 2000.

[84] Prosperetti, A. Life and death by boundary conditions. *J. Fluid Mech.*, 768:1–4, 2015.

[85] Prosperetti, A. and Satrape, J. V. Stability of two-phase flow models. In Joseph, D. D. and Schaeffer, D. G., editors, *Two Phase Flows and Waves*, pages 98–117, 1990.

[86] Prosperetti, A. and Tryggvason, G. *Computational Methods for Multiphase Flows.* Cambridge University Press, 2009.

[87] Pumir, A., Bodenschatz, E., and Xu, H. Tetrahedron deformation and alignment of perceived vorticity and strain in a turbulent flow. *Phys. Fluids*, 25:035101, 2013.

[88] Richardson, J. and Zaki, W. Sedimentation and fluidisation: Part I. *Trans. Inst. Chem. Eng.*, 32:35, 1954.

[89] Rudnick, J. and Gaspari, G. The asphericity of random walks. *J. Phys. A.*, 19:191–193, 1986.

[90] Rudnick, J. and Gaspari, G. The shapes of random walks. *Science*, 237:384–389, 1987.

[91] Saad, Y. *Iterative Methods for Sparse Linear Systems.* PWS Pub. Co, 1996.

[92] Sangani, A. S. and Mo, G. Inclusion of lubrication forces in dynamic simulations. *Phys. Fluids*, 6(5):1653, 1994.

[93] Segrè, P. N., Helbolzheimer, E., and Chaikin, P. M. Long-range correlations in sedimentation. *Phys. Rev. Lett.*, 79:2574–2577, 1997.

[94] Segrè, P. N., Liu, F., Umbanhowar, P., and Weitz, D. A. An effective gravitational temperature for sedimentation. *Nature*, 409:594–597, 2001.

[95] Sierakowski, A. J. GPU-centric resolved-particle disperse two-phase flow simulation using the Physalis method. *Comput. Phys. Commun.*, 207:24, 2016.

[96] Sierakowski, A. J. and Prosperetti, A. Resolved-particle simulation by the Physalis method: enhancements and new capabilities. *J. Comput. Phys.*, 309:164–184, 2016.

[97] Slis, P., Willemse, T., and Kramers, H. The response of the level of a liquid fluidized bed to a sudden change in the fluidization velocity. *Appl. Sci. Res., Ser. A*, 8:209, 1959.

[98] Snabre, P., Pouligny, B., Metayer, C., and Nadal, F. Size segregation and particle velocity fluctuations in settling concentrated suspensions. *Rheol. Acta*, 48:855–870, 2009.

[99] Sundaresan, S. Instabilites in fluidized beds. *Ann. Rev. Fluid Mech.*, 35:63, 2003.

[100] The Open MPI Project. OpenMPI v2.1.2, 2017. [Online]. Available: https://www.open-mpi.org/software/ompi/v2.1.

[101] Theodorou, D. N. and Suter, U. W. Shape of unperturbed linear polymers: Polypropylene. *Macromolecules*, 18:1206–1214, 1985.

[102] Tsuji, Y., Tanaka, T., and Ishida, T. Lagrangian numerical simulation of plug flow of cohesionless particles in a horizontal pipe. *Powder Technol.*, 71(3):239–250, 1992.

[103] Uhlmann, M. An immersed boundary method with direct forcing for the simulation of particulate flows. *J. Comput. Phys.*, 209:448476, 2005.

[104] Uhlmann, M. and Chouippe, A. Clustering and preferential concentration of finite-size particles in forced homogeneous-isotropic turbulence. *J. Fluid Mech.*, 812:991–1023, 2017.

[105] Uhlmann, M. and Doychev, T. Sedimentation of a dilute suspension of rigid spheres

at intermediate Galileo numbers: the effect of clustering upon the particle motion. *J. Fluid Mech.*, 752:310–348, 2014.

[106] van Noije, T. P. C. and Ernst, M. H. Velocity distributions in homogeneous granular fluids: the free and the heated case. *Granular Matter*, 1:57–64, 1998.

[107] van Wijngaarden, L. and Kepteyn, C. Concentration waves in dilute bubble/liquid mixtures. *J. Fluid Mech.*, 212:111, 1990.

[108] Wallis, G. *One Dimensional Two-Phase Flow*. McGraw-Hill, New York, NY, 1969.

[109] Wang, H., Potluri, S., Bureddy, D., Rosales, C., and Panda, D. K. GPU-aware MPI on RDMA-enabled clusters: Design, implementation and evaluation. *IEEE T. Parall. Distr.*, 25(10):2595–2605, 2014.

[110] Wang, Y., Sierakowski, A. J., and Prosperetti, A. Fully-resovled simulation of particle flows with particles-fluid heat transfer. *J. Comput. Phys*, 350:638–656, 2017.

[111] Whitham, G. *Linear and Nonlinear Waves*. Wiley, New York, NY, 1974.

[112] Willen, D. P. and Prosperetti, A. Resolved simulations of sedimenting suspensions of spheres. *Phys. Rev. Fluids*, Submitted, 2018.

[113] Willen, D. P. and Sierakowski, A. J. Bluebottle: Many-GPU-centric particulate multiphase flow simulations using the Physalis method, 2018. v3.0.0 [Online]. Available: physaliscfd.org.

[114] Willen, D. P. and Sierakowski, A. J. Resolved particle simulations using the Physalis method on many GPUs. *Comput. Phys. Commun*, Submitted, 2018.

[115] Willen, D. P., Sierakowski, A. J., Zhou, G., and Prosperetti, A. Continuity waves in resolved-particle simulations of fluidized beds. *Phys. Rev. Fluids*, 2:114305, Nov 2017.

[116] Xu, H., Pumir, A., and Bodenschatz, E. The pirouette effect in turbulent flows. *Nat. Phys.*, 7:709–712, 2011.

[117] Yin, X. and Koch, D. Hindered settling velocity and microstructure in suspensions of solid spheres with moderate Reynolds numbers. *Phys. Fluids*, 19:0093302, 2007.

[118] Yin, X. and Koch, D. Velocity fluctuations and hydrodynamic diffusion in finite-Reynolds number sedimenting suspensions. *Phys. Fluids*, 20(4):043305, 2008.

[119] Zaidi, A. A., Tsuji, T., and Tanaka, T. Hindered settling velocity and structure formation during particle settling by direct numerical simulation. *Procedia Eng.*, 102: 1656–1666, 2015.

[120] Zenit, R., Koch, D., and Sangani, A. Measurements of the average properties of a suspension of bubbles rising in a vertical channel. *J. Fluid Mech.*, 429:307, 2001.

[121] Zhang, Y. Personal communication, 2017.

[122] Zhu, X., Phillips, E., Spandan, V., Donners, J., Ruetsch, G., Romero, J., Ostilla-Mónico, R., Yang, Y., Lohse, D., Verzicco, R., Fatica, M., and Stevens, R. J. A. M. AFiD-GPU: a versatile Navier-Stokes solver for wall-bounded turbulent flows on GPU clusters. *Comput. Phys. Commun*, 229:199–210, 2018.

Curriculum Vitae

Daniel P. Willen was born in Cincinnati, OH on December 19, 1991. He attended Johns Hopkins University from 2010 to 2018, where he obtained his Bachelor of Science (2014), Master of Science (2015), and Doctor of Philosophy (2018). His thesis work was performed under the guidance of Professor Andrea Prosperetti.

Daniel's research experience includes computational develepment work as well as analysis of simulation results. He extended a legacy computational fluid dynamics simulation code to run on many graphics processing units. Additionally, he investigated continuity waves and microscopic properties of particle-laden flows.