Journal of Computational Physics 333 (2017) 269–296

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

# A staggered overset grid method for resolved simulation of incompressible flow around moving spheres

A.W. Vreman

*AkzoNobel, Research Development & Innovation, Process Technology, P.O. Box 10, 7400 AA Deventer, The Netherlands*

ARTICLE INFO

*Article history:*
Received 5 August 2016
Received in revised form 13 November 2016
Accepted 16 December 2016
Available online 23 December 2016

*Keywords:*
Overset grid method
Particle-resolved direct numerical simulation
Moving body problems

ABSTRACT

An overset grid method for resolved simulation of incompressible (turbulent) flows around moving spherical particles is presented. The Navier–Stokes equations in spherical coordinates are solved on body-fitted spherical polar grids attached to the moving spheres. These grids are overset on a fixed Cartesian background grid, where the Navier–Stokes equations in Cartesian coordinates are solved. The standard second-order staggered finite difference scheme is used on each grid. The velocities and pressures on different grids are coupled by third-order Lagrange interpolations. The method, implemented in the form of a Message Passing Interface parallel program, has been validated for a range of flows around spheres. In a first validation section, the results of simulations of four Stokes flows around a single moving sphere are compared with classical analytical results. The first three cases are the flows due to a translating, an oscillating sphere and a rotating sphere. The numerically produced velocity and pressure fields appear to converge to the corresponding (transient) analytical solutions in the maximum norm. The fourth Stokes case is the flow due to an instantaneously accelerated sphere. For this case, the results are compared with the corresponding numerical solution of the Basset–Boussinesq–Oseen equation. In a second validation section, results of three Navier–Stokes flows around one or more moving spheres are presented. These test configurations are a moving face-centered cubic array of spheres, laminar channel flow with a falling a sphere, and freely moving small spheres in a Taylor–Green flow. Results for the flow with the falling sphere are compared with the results from the literature on immersed boundary methods.



## 1. Introduction

For particle-resolved direct numerical simulation of turbulent flows with embedded rigid particles several numerical techniques exist. A powerful technique is the so-called immersed boundary method. In this method the particle boundary is approximated on a Cartesian grid and its effect on the flow is accounted for by a forcing term, which is either a so-called direct forcing term or a continuous forcing term [1]. The grid is usually chosen to be uniform and the forced Navier–Stokes equations are typically solved in the domain including the interiors of the particles [2–6], although there are also variants that exclude the interiors of the particles [7]. The papers cited are only a few of the large amount of publications on this method. A second method is the lattice Boltzmann method, which is also usually applied on a uniform Cartesian grid [8]. Another method, Physalis, uses series of analytical solutions of the steady Stokes equations to bridge the narrow gaps between the uniform Cartesian grid and the particle surfaces [9]. In Physalis, the boundaries of the spherical particles are

*E-mail addresses:* bert.vreman@akzonobel.com, bert@vremanresearch.nl.

http://dx.doi.org/10.1016/j.jcp.2016.12.027

implemented as sharp interfaces, such that, in contrast to immersed boundary and lattice Boltzmann methods, there is no explicit or implicit smoothing of the particle boundaries.

Traditionally, a body fitted grid is adopted to represent the boundary of an embedded object by a sharp interface, and then the flow around the object is solved by a finite difference, finite volume or finite element method or by a spectral or spectral element method (see [10] and [11] for applications of spectral methods to turbulent flows around a single fixed sphere). Also most commercial computational fluid dynamics packages are based on the body-fitted approach.

For flows around moving particles, the geometry of the flow domain changes in time. This is a challenge, in particular for methods using body-fitted grids. One way to deal with this problem is to use unstructured body-fitted grids, to let the grid points move and, if the grid has become too distorted, to project the solution on a new grid produced by an automatic generator for unstructured grids. In combination with a finite element method, this approach has been used by Johnson and Tezduyar [12] to simulate multiple spheres falling in a liquid-filled tube. Another way to handle the problem of the changing geometry is offered by the so-called overset grid method, also called composite mesh, overlapping grid or Chimera method. The main idea is to attach to each object a body-fitted structured orthogonal grid, which is overset on a background Cartesian grid, or on grids attached to other objects. Interpolation is used to connect the numerical solutions of two overlapping grids to each other.

The concept of overlapping grids was originally introduced for the solution of the Laplace equation and other elliptical partial differential equations on two dimensional domains with a complicated non-moving boundary [13,14]. For the simulation of compressible flows around multiple objects, the Chimera overset grid method was developed [15], which has been used for many flow problems in aerospace engineering [16]. Moreover, Chesshire and Henshaw [17] presented a general method for the construction of colocated overlapping grids for the solution of partial differential equations, general in the sense that it can be used for an arbitrary number of overlapping grids and also for higher-order schemes for the spatial discretization and interpolations. They applied their method to the compressible Navier–Stokes equations. For the incompressible Navier–Stokes equations, Henshaw [18] developed a fourth-order overset grid method and showed an application to steady flow around a sphere, in which the surface of the sphere was covered by two overlapping orthographic patches to avoid pole singularities. In this method the pressure is solved from a Poisson equation equipped with a damping term to keep the velocity divergence small. Others have used the artificial compressibility technique (instead of the pressure Poisson equation) in incompressible flow simulations on overset grids [19]. All overset grid methods mentioned in this paragraph were designed for colocated grids. It seems that, so far, none of them has been used for resolved simulation of particles in turbulent flows.

The first staggered overset grid method was developed by Burton and Eaton [20] and based on the standard second-order staggered approach by Harlow and Welch [21]. Staggered methods are known to be relatively robust, also in combination with standard central differencing. Furthermore, they naturally lead to a compact stencil of the discrete Laplacian of the pressure and a relatively straightforward treatment of the pressure near solid boundaries. The implementation of Burton and Eaton was developed for two fixed staggered grids: a spherical polar grid overset on a Cartesian background grid. The method was successfully applied to direct numerical simulation of turbulent incompressible flow around a single fixed sphere [22]. In each time step, the pressure Poisson equation was solved iteratively by a multigrid matrix factorization technique (a Schwarz alternating method), in which the Poisson equations on the two subdomains were each iteratively solved in turn, while the subdomain boundary conditions were obtained in the outer iteration loop by interpolation of the pressure gradient. Very recently, the staggered overset grid method for spherical particles was used in direct numerical simulation of turbulent flow modified by 64 fixed particles [23]. For that purpose the method was extended to multiple fixed particles. Instead of using an alternating method and interpolation of the pressure gradient, the BiCGStab method with interpolation of the pressure in each iteration was used to solve the pressure Poisson equation. The implementation was parallel, but limited to shared memory systems.

In the present paper, the staggered overset grid method is extended to (freely) moving spherical particles in incompressible flow. Thus a staggered spherical polar grid is attached to each particle and all these grids are overset on the Cartesian background grid. The data structure of the interpolation and grid routines of the implementation in [23] has been altered to realize a parallel implementation based on MPI (Message Passing Interface), which is the standard for high performance computing on large distributed memory systems. In order to make the step of increased complexity not too large, we avoid interpolations from one spherical to another spherical grid in this paper. Thus the particles cannot collide. Narrow gaps between particles can in principle be simulated, but at high computational cost. This apparent limitation of the overset grid method is compensated by a number of advantages. Due to its body-fitted character, the method facilitates an accurate computation of the turbulence dissipation rate and other turbulence statistics of spatial derivatives in the direct vicinity of the particle surfaces, as demonstrated in [23]. Another attractive feature of the overset grid method is that by radial stretching of the spherical grids the total number of grid cells can be largely reduced without sacrificing fine resolution near the particles. The latter feature is particularly useful for resolved simulations of flows around small particles in dilute flows.

After a presentation of the governing equations in section 2 and a description of the staggered overset grid method in section 3, seven validation studies for moving spherical particles will be presented in sections 4 and 5. In section 4, we will consider simulations of four Stokes flows around a single moving sphere and compare the results with non-trivial analytical results. The cases are the Stokes flows due to a translating sphere, an oscillating sphere, a rotating sphere, and an instantaneously accelerated sphere. In section 5, we will present validation results of three Navier–Stokes cases: a translating array of spheres, a falling sphere in a channel, and eight freely moving small spheres in Taylor–Green flow.

The material presented is novel in several respects. The first novelty is the extension of the staggered overset grid method to incompressible flows with moving spheres. Furthermore, pointwise convergence to analytical solutions is demonstrated in numerical tests, which is new in the context of moving spheres and 3D finite difference or finite volume methods. Moreover, the quantitative comparison of immersed boundary methods with a body-fitted method applied to a flow with moving spheres is novel. Also, resolved simulation of small moving spheres in incompressible three-dimensional Taylor–Green flow seems to have been performed for the first time.

Since the flow solver is a Navier–Stokes solver to simulate flows around an arbitrary number of spheres, it is called NSpheres. Source data of the validation simulations presented in this paper are available at www.vremanresearch.nl. Although the MPI implementation opens the way to use the method for large-scale computations on many processors with one or more particles per processor, it is outside the scope of the present work to perform such computations. The purpose of this paper is to describe the method and to validate it for cases that can be simulated in reasonable time with use of a few parallel MPI processes.

## 2. Governing equations

We denote the Cartesian position vector by $\mathbf{x} = [x_1\ x_2\ x_3]^T$. The Cartesian base vectors are denoted by $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$. We consider a spherical particle embedded in the flow, centered at position $\mathbf{x}^p$. The particle radius is denoted by $r_0$ and the particle diameter is denoted by $d_p = 2r_0$. The spherical coordinates around the particle are given by the nonlinear expressions

$$r = |\mathbf{x} - \mathbf{x}^p|, \quad \theta = \arccos(\frac{x_3 - x_3^p}{r}), \quad \phi = \mathrm{atan2}(x_2 - x_2^p, x_1 - x_1^p), \tag{1}$$

where $\mathrm{atan2}(y, x)$ is the function that provides the argument of the complex number with real part $x$ and complex part $y$, such that $-\pi < \phi \leq \pi$. Here $r$ denotes the radial, $\theta$ the polar and $\phi$ the azimuthal coordinate. The base vectors of the spherical coordinate system are denoted by $\tilde{\mathbf{e}}_1 = \tilde{\mathbf{e}}_r$, $\tilde{\mathbf{e}}_2 = \tilde{\mathbf{e}}_\theta$ and $\tilde{\mathbf{e}}_3 = \tilde{\mathbf{e}}_\phi$ and specified by the following orthogonal matrix

$$\mathbf{A} = [\mathbf{e}_r\ \mathbf{e}_\theta\ \mathbf{e}_\phi]^T = \begin{bmatrix} \sin\theta\cos\phi & \sin\theta\sin\phi & \cos\theta \\ \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\sin\phi & \cos\phi & 0 \end{bmatrix}. \tag{2}$$

Equation (1) describes the (nonlinear) transformation from Cartesian to spherical coordinates. The inverse transformation is given by

$$\mathbf{x} = \mathbf{x}^p + \begin{bmatrix} r\sin\theta\cos\phi \\ r\sin\theta\sin\phi \\ r\cos\theta \end{bmatrix}, \tag{3}$$

and the components can be written as $x_j = x_j^p + rA_{1j}$. If the last two columns of the matrix of partial derivatives (Jacobian matrix) of the inverse coordinate transformation are divided by $r$, then $\mathbf{A}^T$ is obtained.

The translational and rotational velocity vectors of the particle are denoted by $\mathbf{v}^p$ and $\boldsymbol{\omega}^p$, respectively. The fluid velocity vector is denoted by $\mathbf{u} = [u_1\ u_2\ u_3]^T$ in the Cartesian coordinate system and by $\tilde{\mathbf{u}} = [\tilde{u}_1\ \tilde{u}_2\ \tilde{u}_3]^T = [u_r\ u_\theta\ u_\phi]^T$ in the spherical coordinate system. The latter two vectors are related through $u_i\mathbf{e}_i = \tilde{\mathbf{u}}_i\tilde{\mathbf{e}}_i$, which implies the following (linear) relationships between the Cartesian and spherical velocity components

$$\tilde{\mathbf{u}} = \mathbf{A}(\mathbf{u} - \mathbf{v}^p), \quad \mathbf{u} = \mathbf{v}^p + \mathbf{A}^T\tilde{\mathbf{u}}. \tag{4}$$

In this paper, we will use the index notation and the summation convention (only for the indices $i$ and $j$) in some equations, for example in the Navier–Stokes equations in Cartesian and spherical coordinates and in equations (13)–(14) and (22)–(23), to indicate more clearly in which form these equations, term by term, are used in the computer code. In addition, we use the comma notation for derivatives, for example $u_{r,t} = \partial u_r/\partial t$ ($t$ denotes time), $u_{r,\theta} = \partial u_r/\partial\theta$ and $u_{j,i} = \partial u_j/\partial x_i$, for integer values $i$ and $j$, while $u_{j,ii}$ denotes $\nabla^2 u_j$ in Cartesian coordinates.

The form of the Navier–Stokes equations in the Cartesian frame of reference is given by

$$u_{i,i} = 0, \tag{5}$$

$$u_{j,t} + (u_i u_j)_{,i} = -q_{,j} + \nu u_{j,ii} + g_j + a_j, \tag{6}$$

for component $j = 1, 2, 3$, where $\nu$ denotes the kinematic viscosity and $\mathbf{g}$ the gravity vector, while $\mathbf{a}$ is a spatially constant acceleration term, useful for a driving force in combination with periodic boundary conditions for $q$. The latter is related to the physical pressure $p$ through the definition $q = p/\rho + \mathbf{a}\cdot\mathbf{x}$, where $\rho$ is the constant fluid density. If we mention the pressure in this paper, we usually refer to $q$. The second term on the left-hand side of (6) is called the convective term. The four terms on the right-hand side are called the pressure gradient term, the viscous term, the gravity term and the forcing term, respectively.

In the spherical frame of reference of a particle, the following form of the Navier–Stokes equations is used:

$$\frac{(r^2u_r)_{,r}}{r^2}+\frac{(u_\theta\sin\theta)_{,\theta}}{r\sin\theta}+\frac{u_{\phi,\phi}}{r\sin\theta}=0, \tag{7}$$

$$u_{r,t}+\frac{(r^2u_ru_r)_{,r}}{r^2}+\frac{(u_ru_\theta\sin\theta)_{,\theta}}{r\sin\theta}+\frac{(u_ru_\phi)_{,\phi}}{r\sin\theta}-\frac{u_\theta^2+u_\phi^2}{r}=-q_{,r}+\nu\Big(\tilde\nabla^2u_r+2\frac{u_r}{r^2}+2\frac{u_{r,r}}{r}\Big)-A_{1i}v^p_{i,t}+A_{1i}(g_i+a_i), \tag{8}$$

$$u_{\theta,t}+\frac{(r^2u_\theta u_r)_{,r}}{r^2}+\frac{(u_\theta u_\theta\sin\theta)_{,\theta}}{r\sin\theta}+\frac{(u_\theta u_\phi)_{,\phi}}{r\sin\theta}+\frac{u_\theta u_r-u_\phi^2\cot\theta}{r}=-\frac{q_{,\theta}}{r}+\nu\Big(\tilde\nabla^2u_\theta+2\frac{u_{r,\theta}}{r^2}-\frac{u_\theta+2u_{\phi,\phi}\cos\theta}{r^2\sin^2\theta}\Big)-A_{2i}v^p_{i,t}+A_{2i}(g_i+a_i), \tag{9}$$

$$u_{\phi,t}+\frac{(r^2u_\phi u_r)_{,r}}{r^2}+\frac{(u_\phi u_\theta\sin\theta)_{,\theta}}{r\sin\theta}+\frac{(u_\phi u_\phi)_{,\phi}}{r\sin\theta}+\frac{u_\phi u_r+u_\phi u_\theta\cot\theta}{r}=-\frac{q_{,\phi}}{r\sin\theta}+\nu\Big(\tilde\nabla^2u_\phi+\frac{2u_{r,\phi}\sin\theta+2u_{\theta,\phi}\cos\theta-u_\phi}{r^2\sin^2\theta}\Big)-A_{3i}v^p_{i,t}+A_{3i}(g_i+a_i). \tag{10}$$

Applied to a scalar $c$, the Laplace operator in spherical coordinates appearing in these equations is given by

$$\tilde\nabla^2c=\frac{(r^2c_{,r})_{,r}}{r^2}+\frac{(c_{,\theta}\sin\theta)_{,\theta}}{r^2\sin\theta}+\frac{c_{,\phi\phi}}{r^2\sin^2\theta}. \tag{11}$$

Impermeability and no-slip conditions are imposed at the particle surface ($r=r_0$), which means that the fluid velocity should be equal to the surface velocity of the solid particle: $\mathbf{u}=\mathbf{v}^p+r_0\boldsymbol{\omega}^p\times\mathbf{e}_r$ at $r_0$. The surface spherical velocity components are derived by substituting (2) into $\tilde{\mathbf{u}}=\mathbf{A}(\mathbf{u}-\mathbf{v}^p)=r_0\mathbf{A}(\boldsymbol{\omega}^p\times\mathbf{e}_r)$:

$$u_r=0, \tag{12}$$

$$u_\theta=r_0\omega^p_jA_{3j}, \tag{13}$$

$$u_\phi=-r_0\omega^p_jA_{2j}. \tag{14}$$

These expressions and the continuity equation (7) imply that $u_{r,r}=0$ at $r=r_0$.

The equations that govern the particle motion are given by:

$$\mathbf{x}^p_{,t}=\mathbf{v}^p_{,t}, \tag{15}$$

$$\rho_pV_p\mathbf{v}^p_{,t}=\mathbf{F}^p+\rho_pV_p\mathbf{g}+\rho V_p\mathbf{a}, \tag{16}$$

$$I_p\boldsymbol{\omega}^p_{,t}=\mathbf{T}^p, \tag{17}$$

where $\rho_p$ is the particle density, $V_p=\pi d_p^3/6$ the particle volume and $I_p=\rho_pd_p^2V_p/10$ the moment of the inertia of the particle. The expressions for the particle force $\mathbf{F}^p$ and particle torque $\mathbf{T}^p$ are based on the stress tensor in spherical coordinates,

$$\boldsymbol{\tau}=\rho(-q\delta_{ij}+\nu(G_{ij}+G_{ji}))\tilde{\mathbf{e}}_i\tilde{\mathbf{e}}_j, \tag{18}$$

where $\delta_{ij}$ is the Kronecker delta and the tensor $G_{ij}\tilde{\mathbf{e}}_i\tilde{\mathbf{e}}_j$ with

$$\mathbf{G}=\begin{bmatrix} u_{r,r} & u_{\theta,r} & u_{\phi,r} \\ \frac{u_{r,\theta}}{r}-\frac{u_\theta}{r} & \frac{u_{\theta,\theta}}{r}+\frac{u_r}{r} & \frac{u_{\phi,\theta}}{r} \\ \frac{u_{r,\phi}}{r\sin\theta}-\frac{u_\phi}{r} & \frac{u_{\theta,\phi}}{r\sin\theta}-\frac{u_\phi\cot\theta}{r} & \frac{u_{\phi,\phi}}{r\sin\theta}+\frac{u_r}{r}+\frac{u_\theta\cot\theta}{r}\end{bmatrix} \tag{19}$$

is the gradient of the velocity expressed in the spherical coordinate system [24].

Since the outward normal of the particle surface $S_p$ is equal to $\tilde{\mathbf{e}}_1$, we write

$$\mathbf{F}^p=\int_{S_p}\boldsymbol{\tau}\cdot\tilde{\mathbf{e}}_1\mathrm{d}S=\rho\int_{S_p}\Big((-q+2\nu G_{11})\tilde{\mathbf{e}}_1+\nu(G_{12}+G_{21})\tilde{\mathbf{e}}_2+\nu(G_{13}+G_{31})\tilde{\mathbf{e}}_3\Big)\mathrm{d}S, \tag{20}$$

$$\mathbf{T}^p=\int_{S_p}r_0\mathbf{e}_1\times(\boldsymbol{\tau}\cdot\tilde{\mathbf{e}}_1)\mathrm{d}S=\rho\nu\int_{S_p}\Big(r_0(G_{12}+G_{21})\tilde{\mathbf{e}}_3-r_0(G_{13}+G_{31})\tilde{\mathbf{e}}_2\Big)\mathrm{d}S. \tag{21}$$

At the particle surface, $u_r=0$ and $u_{r,r}=G_{11}=0$, while the former also implies $u_{r,\theta}=0$ and $u_{r,\phi}=0$. Since the components of the spherical base vectors are given by (2) and $u_\theta$ and $u_\phi$ by (13) and (14), the integrals over $G_{21}\tilde{\mathbf{e}}_2$ and $G_{31}\tilde{\mathbf{e}}_3$ vanish. Thus the components of the particle force and particle torque can be written as

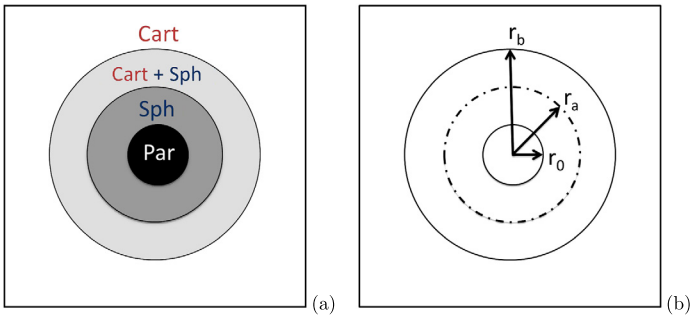


**Fig. 1.** (a) Two-dimensional illustration of an overlapping Cartesian (Cart) and spherical domain (Sph) around a particle (Par). The overlap region is denoted by Cart + Sph. (b) Two-dimensional illustration of the three concentrical spheres with radii $r_0$, $r_a$ and $r_b$.

$$F_j^p = \rho r_0^2 \int_0^{2\pi} \int_0^{\pi} \Big( -qA_{1j} + \nu(u_{\theta,r}A_{2j} + u_{\phi,r}A_{3j}) \Big) \sin\theta \, \mathrm{d}\theta \, \mathrm{d}\phi, \tag{22}$$

$$T_j^p = \rho r_0^2 \nu \int_0^{2\pi} \int_0^{\pi} \Big( (r_0 u_{\theta,r} - u_\theta)A_{3j} - (r_0 u_{\phi,r} - u_\phi)A_{2j} \Big) \sin\theta \, \mathrm{d}\theta \, \mathrm{d}\phi. \tag{23}$$

All fluid variables that appear in the latter two expressions are evaluated at $r = r_0$. The force $\mathbf{F}^p$ is naturally decomposed into a pressure part $\mathbf{F}^{p,q}$ and a viscous part $\mathbf{F}^{p,\nu}$. If $q$ is periodic in a direction and gravity acts in that direction, buoyancy enters through the term $\rho\mathbf{a}$ ($\mathbf{a} = -\mathbf{g}$ for a system at rest).

## 3. Computational method

In this section, the staggered overset grid method for moving particles is explained. Subsection 3.1 contains a basic description of the numerical method. The description is basic in the sense that the explanation of grid structures, interpolations, boundary conditions and parallelization are described in general terms or omitted. The technical explanation of the grid structure and interpolation procedure is provided in subsections 3.2, 3.3 and 3.4. In subsections 3.5 and 3.6, all activities during the initialization and time step are listed, including explanations of several details (on the implementation of boundary conditions, for example). Additional information and discussion of the parallelization can be found in Appendix A, and the interpolation formulas can be found in Appendix B.

### 3.1. Basic description of the numerical method

The present overset grid method is based on the standard second-order staggered discretization scheme for incompressible flow. Thus the velocity components are defined at the faces of a given grid cell. More specifically, velocity component $i$ is defined at the faces whose normal vector is aligned to the unit vector of direction $i$. The pressure and the discrete velocity divergence are defined at the center of the cell, such that the continuity equation reduces to the balance of the volumetric flows through the cell faces. The advantage of the staggered scheme is that the discrete velocity divergence and pressure gradient operators are computed on the most narrow stencils possible. In each direction, the stencils of these operators are only one grid spacing wide (the distance between two grid points). This automatically provides a tight coupling between the pressures at adjacent grid points, and staggered schemes are therefore known to be relatively robust.

We consider a rectangular computational domain $\Omega_c = [0, L_1] \times [0, L_2] \times [0, L_3]$ with $N_p$ spherical particles of diameter $d_p$. The domain $\Omega_c$ includes the regions inside the particles. The domain that excludes the volumes occupied by the particles is denoted by $\Omega$ and changes with time if the particles are moving. A two-dimensional illustration of a Cartesian and a spherical domain is shown in Fig. 1. The radii of the three concentrical spheres in Fig. 1(a) are denoted by $r_0$, $r_a$ and $r_b$, and these are visualized in Fig. 1(b). The particle is represented by the inner circle ($r \leq r_0$). The interior of the spherical domain is represented by the two rings around the particle ($r_0 < r < r_b$). The interior of the Cartesian domain is represented by the second ring and the white region outside ($r > r_a$). The union of the Cartesian and spherical domain represents $\Omega$ ($r > r_0$). The intersection of the Cartesian and spherical domain is the second ring. This is the region where the domains overlap. The Navier–Stokes equations in Cartesian coordinates are solved on the Cartesian domain, which is meshed by a staggered Cartesian grid. The Navier–Stokes equations in spherical coordinates are solved on each spherical domain, which is meshed by a staggered spherical grid, which can be stretched in the radial direction. The discretization on each domain is based on the standard second-order finite difference scheme for incompressible flow. The solutions on the different domains are connected through third-order Lagrange interpolations: the fluid variables at (or near) each spherical boundary (at $r = r_a$) of the Cartesian domain are obtained from the spherical solution and the fluid variables at (or near) each spherical boundary (at $r = r_b$) of the spherical domain are obtained from the Cartesian solution.

The rectangular domain $\Omega_c$ is partitioned into $M = M_1 \times M_2 \times M_3$ rectangular blocks. The total number of MPI processes is equal to $M$. Each one of the $M$ processors contains a Cartesian field, which we define as the numerical solution on the intersection of a given block and the interior of the entire Cartesian domain. Each block is equipped with a Cartesian grid which contains $N_k/M_k$ cells in the $x_k$ direction, such that all blocks together contain $N_1 \times N_2 \times N_3$ grid cells. A spherical polar grid of $N_r \times N_\theta \times N_\phi$ cells is attached to each particle. The numerical solution of the Navier–Stokes equations in spherical coordinates around a given particle is called a spherical field. Each processor contains up to $N_{p1}$ spherical fields, where $N_{p1}$ is the smallest integer number satisfying $N_{p1} \geq N_p/M$.

We proceed with an overview of the algorithms used in each time step. We distinguish between three parts of the time step, in which the variables are updated from time level $n$ to $n+1$ (the time level is included in the superscripts). At the end of each part the interpolation procedure is performed for the velocity, which means that the velocity interpolations from Cartesian to spherical grids and vice versa are performed and that the boundary conditions are applied. For conciseness, the terms of the Navier–Stokes equations are grouped in vectors. However, from the expressions in index notation in the previous section, the contents of the vectors can be deduced.

In the first part of the time step, the positions and the translational and angular velocities of the particles are updated:

$$\mathbf{x}^{p,n+1} = \mathbf{x}^{p,n} + \Delta t\, \mathbf{v}^{p,n}, \tag{24}$$

$$\mathbf{v}^{p,n+1} = \mathbf{v}^{p,n} + \Delta t\, (\mathbf{F}^{p,n} + \rho_p V_p \mathbf{g} + \rho V_p \mathbf{a}^n)/(\rho_p V_p), \tag{25}$$

$$\boldsymbol{\omega}^{p,n+1} = \boldsymbol{\omega}^{p,n} + \Delta t\, \mathbf{T}^{p,n}/I_p, \tag{26}$$

where $\Delta t$ is the length of the time step. Then the first intermediate fluid velocities, $\mathbf{u}^*$ and $\tilde{\mathbf{u}}^*$, are obtained:

$$\mathbf{u}^* = \mathbf{u}^n, \qquad \tilde{\mathbf{u}}^* = \tilde{\mathbf{u}}^n - \mathbf{A}(\mathbf{v}^{p,n+1} - \mathbf{v}^{p,n}). \tag{27}$$

The last expression accounts for the term $-\mathbf{A}\mathbf{v}^p_{,t}$ in (8) to (10). Afterwards, the new surface velocities are computed by substituting the angular velocity at level $n+1$ into equations (13)–(14). Then the interpolation procedure is applied to $\mathbf{u}^*$ and $\tilde{\mathbf{u}}^*$.

In the second part of the time step, the second intermediate velocities, $\mathbf{u}^{**}$ and $\tilde{\mathbf{u}}^{**}$, are computed. For each spherical domain, $\tilde{\mathbf{u}}^{**}$ is obtained by solving

$$(\mathbf{I} - \tfrac{1}{2}\Delta t(-\tilde{\mathbf{H}}^*{}_\phi + \tilde{\mathbf{J}}_\phi))\tilde{\mathbf{u}}^{**} = \tilde{\mathbf{u}}^* + \tfrac{1}{2}\Delta t(-\tilde{\mathbf{H}}^*{}_\phi + \tilde{\mathbf{J}}_\phi)\tilde{\mathbf{u}}^* + \Delta t(-\tilde{\mathbf{H}}^* + \tilde{\mathbf{J}}^*), \tag{28}$$

where $\mathbf{I}$ is the identity matrix, $\tilde{\mathbf{H}}^*{}_\phi$ the coefficient matrix of the convective derivatives with respect to $\phi$ (the coefficients depend on $u^*_\phi$) and $\tilde{\mathbf{J}}_\phi$ the coefficient matrix of the viscous second-order derivatives with respect to $\phi$. The other convective terms are denoted by $\tilde{\mathbf{H}}^*$, while the other viscous terms and the forcing terms are denoted by $\tilde{\mathbf{J}}^*$ (which does not include $-\mathbf{A}\mathbf{v}^p_{,t}$ since that term is treated in the first part of the time step). All terms in $-\tilde{\mathbf{H}}^* + \tilde{\mathbf{J}}^*$ are based on $\tilde{u}^*$. After evaluation of the right-hand side, Gauss elimination of the tri-diagonal systems is used to obtain $\tilde{\mathbf{u}}^{**}$ in each spherical domain. The second intermediate velocity $\mathbf{u}^{**}$ in each Cartesian block is computed by

$$\mathbf{u}^{**} = \mathbf{u}^* + \Delta t(-\mathbf{H}^* + \mathbf{J}^*), \tag{29}$$

where $\mathbf{H}^*$ represents the Cartesian convective terms and $\mathbf{J}^*$ the Cartesian viscous and forcing terms, all based on $\mathbf{u}^*$. For the spatial discretization, standard second-order central difference approximations are applied to the convective and viscous terms. The discretization of each term is based on the form specified in section 2. In the discretization of the convective terms, appropriate averages of velocity components over two points with weights $\frac{1}{2}$ and $\frac{1}{2}$ are used before a velocity component is squared or two velocity components are multiplied. The second part of the time step is completed by applying the interpolation procedure to $\mathbf{u}^{**}$ and $\tilde{\mathbf{u}}^{**}$.

In the third part of the time step is the pressure Poisson equation problem is solved for overset grids (see [23], but as the notation was slightly different there, the description is repeated below). The pressure Poisson equations of all domains are assembled and solved as a combined linear system. The augmented matrix approach of [18] is used, which means that an extra variable and an extra equation are introduced to overcome the singularity of the original linear system. The discrete system of equations for the pressure $q$ is given by

$$\begin{aligned}
&c_k(\nabla^2 q)_k + b = c_k(\nabla \cdot \mathbf{u}^{**})_k/\Delta t \quad \text{at all internal pressure points } k \text{ of all Cartesian grids,}\\
&c_k(\tilde{\nabla}^2 q)_k + b = c_k(\tilde{\nabla} \cdot \tilde{\mathbf{u}}^{**})_k/\Delta t \quad \text{at all internal pressure points } k \text{ of all spherical grids,}\\
&\sum_{k=1}^{N_q} q_k = 0,
\end{aligned} \tag{30}$$

where $\nabla^2$ and $\nabla\cdot$ represent the discrete Laplacian and divergence in Cartesian coordinates, while $\tilde{\nabla}^2$ and $\tilde{\nabla}\cdot$ represent the discrete Laplacian and divergence in spherical coordinates. The forms of the discrete Laplacians follow from the approximations of the continuity equation and the pressure gradient term by straightforward second-order central differences on staggered grids. The system contains $N_q + 1$ equations and $N_q + 1$ unknowns, where $N_q$ is the total number of internal points

**Table 1**
Array definitions for the pressure and the staggered velocity components in each Cartesian block and each spherical domain. For the original grid points, the ranges of the grid indices $i$, $j$, $k$ of the three coordinate directions are specified in the last three columns.

| Variable | Array definition | Original $i$ | Original $j$ | Original $k$ |
|---|---|---|---|---|
| Cartesian $q$ | $(0:N_1'+1, 0:N_2'+1, 0:N_3'+1)$ | $1:N_1'$ | $1:N_2'$ | $1:N_3'$ |
| $u_1$ | $(-1:N_1'+1, 0:N_2'+1, 0:N_3'+1)$ | $1:N_1'$ | $1:N_2'$ | $1:N_3'$ |
| $u_2$ | $(0:N_1'+1, -1:N_2'+1, 0:N_3'+1)$ | $1:N_1'$ | $1:N_2'$ | $1:N_3'$ |
| $u_3$ | $(0:N_1'+1, 0:N_2'+1, -1:N_3'+1)$ | $1:N_1'$ | $1:N_2'$ | $1:N_3'$ |
| Spherical $q$ | $(0:N_r+1, 0:N_\theta+1, 0:N_\phi+1, 1:N_{p1})$ | $1:N_r$ | $1:N_\theta$ | $1:N_\phi$ |
| $u_r$ | $(0:N_r, 0:N_\theta+1, 0:N_\phi+1, 1:N_{p1})$ | $1:N_r-1$ | $1:N_\theta$ | $1:N_\phi$ |
| $u_\theta$ | $(0:N_r+1, -1:N_\theta+1, 0:N_\phi+1, 1:N_{p1})$ | $1:N_r$ | $1:N_\theta-1$ | $1:N_\phi$ |
| $u_\phi$ | $(0:N_r+1, 0:N_\theta+1, -1:N_\phi+1, 1:N_{p1})$ | $1:N_r$ | $1:N_\theta$ | $1:N_\phi$ |

in all domains. The last equation is the extra equation and the extra variable is $b$ ($b$ converges to zero in the limit of zero grid size). The coefficients $c_k$ represent normalization coefficients, such that all diagonal elements of the first $N_q$ rows of the coefficient matrix are equal to 1. The total linear system to be solved is written as $\mathbf{Cy} = \mathbf{z}$, where $\mathbf{C}$ is the $(N_q+1)\times(N_q+1)$ coefficient matrix, $\mathbf{y}$ the vector of unknowns and $\mathbf{z}$ the vector of right-hand sides of (30). The system $\mathbf{Cy} = \mathbf{z}$ is iteratively solved by the BiCGstab(1) method [25]. The starting pressure is the pressure from the previous time step. In each iteration the pressure is interpolated from Cartesian to spherical grids and vice versa. The velocities at time level $n+1$ are obtained by projecting the intermediate velocities on the space of functions that is (approximately) divergence free:

$$\mathbf{u}^{n+1} = \mathbf{u}^{**} - \Delta t\, \nabla q, \quad \tilde{\mathbf{u}}^{n+1} = \tilde{\mathbf{u}}^{**} - \Delta t\, \tilde{\nabla} q, \tag{31}$$

where $\nabla$ and $\tilde{\nabla}$ represent the discrete gradient operators, in Cartesian and spherical coordinates respectively. Like the first and second part of the time step, the third part of the time step is completed by applying the interpolation procedure to the velocities, this time to $\mathbf{u}^{n+1}$ and $\tilde{\mathbf{u}}^{n+1}$.

### 3.2. Numbering of blocks, particles and grids

As mentioned above, the rectangular domain $\Omega_c$ is partitioned into $M = M_1 \times M_2 \times M_3$ rectangular blocks, the so-called physical blocks. The physical blocks and corresponding grids are numbered with the negative grid numbers $-1, -2, ..., -M$ (grid identification numbers). To account for periodic directions, a single layer of shadow blocks is defined around the rectangular domain. Each shadow block is labeled with a negative number less than $-M$ and larger than or equal to $-(M_1+2)(M_2+2)(M_3+2)$. For each block, a reference position vector is defined, which refers to the bottom left front corner of the block.

The spherical domains and the corresponding grid identification numbers are indicated with the positive grid identification numbers 1, 2, ..., $N_p$. The reference position vector of each spherical domain is given by $\mathbf{x}^p$ and corresponds to $r = 0$. Each $\mathbf{x}^p$ is updated such that it always remains inside $\Omega_c$. In addition to the $N_p$ physical particles, shadow particles are defined to account for periodic boundary conditions. These shadow particles are shifted copies of physical particles such that the centers of the shadow particles reside in the shadow blocks outside $\Omega_c$. For each shadow particle, the corresponding physical particle number is stored in a pointer array. The number of particles including shadow particles is called the virtual particle number. This number is in general larger than the number of spherical grids, which is equal to the true number of particles, $N_p$. For each block, a block list of particles is defined that contains the numbers of all particles whose cell centers lie inside the block. In addition a pointer array is defined which contains for each block the number of the physical block (and its grid), while it contains for each particle the number of the physical particle (and its grid).

The Cartesian grids of the blocks can be stretched, but in this paper we assume them to be uniform, and we also assume that each block has the same dimensions. Furthermore, all spherical grids are the same. Grids are essentially partitions of certain domains into grid cells. The corners of the grid cells are usually called vertices or nodes. The vertices are often not the locations where the discrete flow variables are defined. In the staggered method, the pressure is defined at the cell centers, and velocity component $i$ is defined at the centers of each cell face whose normal vector points in direction $i$. In this paper, each location where the discrete pressure or a discrete velocity component is defined is called a grid point.

The structure of the Cartesian and spherical grids makes it convenient to store, on each processor, the Cartesian fields as three dimensional arrays and the spherical fields as four dimensional arrays, as shown in Table 1. The *original* grid points, the grid points inside the Cartesian blocks and the grid points inside the spherical domains, are defined by the ranges of the grid indices listed in the last three columns of Table 1. Each Cartesian grid has $N' = N_k/M_k$ original pressure grid points in direction $x_k$. The corresponding $N_1' \times N_2' \times N_3'$ original cells cover the Cartesian block. Likewise, each spherical domain ($r_0 \le r \le r_b$, $0 \le \theta \le \pi$ and $-\pi < \phi \le \pi$) is covered by $N_r \times N_\theta \times N_\phi$ original cells. Each point that is not an original point is a *dummy* point by definition. The dummy grid points are located on the boundary or just outside the boundary of a domain and are used to account for boundary conditions or parallel communication or interpolation.

The indices in the direction of a staggered quantity refer to staggered locations. Staggered quantities have some grid points on the boundaries of the domain. For example, a grid point of the staggered $u_r$ velocity lies on the particle surface if the original $i$ index for the $u_r$ velocity is zero, which refers to a dummy location. For a staggered quantity near a boundary

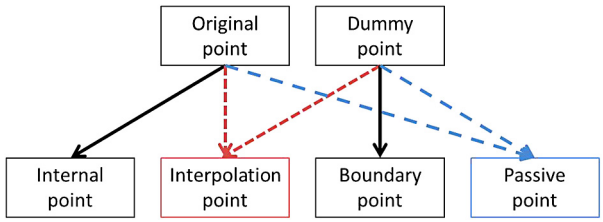


**Fig. 2.** Overview of the different categories of grid points.

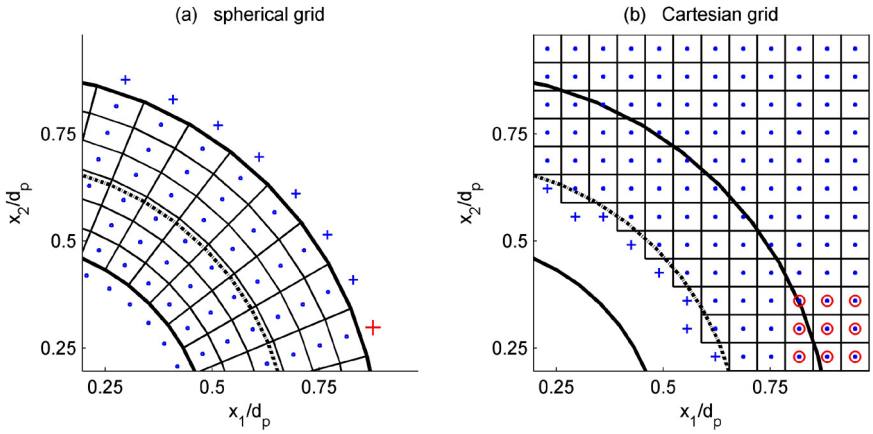


**Fig. 3.** Two-dimensional illustration of a spherical grid attached to a particle (a), overset on the Cartesian grid of a block (b). Both plots represent the same region of the overall domain $\Omega_c$. The center of the particle (at $x_1 = x_2 = 0$) lies outside the region shown. The grid cells around the internal pressure points are indicated by the meshes of thin solid lines. The three thick circular lines denote the three concentrical surfaces with radii $r_0$ (inner solid circle, the particle surface), $r_a$ (dashed circle) and $r_b$ (outer solid circle). The dots denote internal (and boundary) pressure points. The pluses denote interpolation points. The interpolation stencil corresponding to the enlarged plus in (a) is indicated by the encircled points in (b). Passive points are not shown.

that is not a wall, an extra dummy layer is required at index $i = -1$. This layer is not required for the discretization of derivatives, but is convenient for the interpolation stencil (this will be clarified later on). Since in this paper we assume that the interpolation stencils do not use points outside the physical domain, no extra dummy layer is required for the normal velocity at a wall. In Table 1, it is assumed that the block end faces are no walls. For example, if the $x_1$ end face were a wall, the original $i$ index for $u_1$ would run up to $N_1 - 1$, $i = N_1$ would coincide with the wall and thus be a dummy point, and $i = N_1 + 1$ would not play any role.

### 3.3. Interpolation points

In addition to the distinction between original and dummy points, we classify all grid points into four categories: *internal* points, where the Navier–Stokes equations are discretized; *interpolation* points (query points), where the pressure or a velocity component is obtained by interpolation; *boundary* points, which are dummy points used to implement the boundary conditions; and *passive* points, which (temporarily) play no role in the numerical scheme. Each original point is either an internal point, an interpolation point, or a passive point. Each dummy point is either an interpolation point, a boundary point, or a passive point. The relation between the two classifications is visualized in Fig. 2.

The characterization of the staggered grid points is derived from the characterization of the pressure grid points (the cell centers), and therefore we consider the pressure grid points first. For each spherical grid, the internal pressure grid points are precisely the original grid points, while the interpolation points are the dummy grid points with $r > r_b$ (see Fig. 3(a)). All other dummy pressure grid points are boundary points. The spherical grids have no passive points. For the Cartesian grids, the characterization of points is determined by $r_a$, the radius that is used to cut holes in $\Omega_c$. If a Cartesian pressure point is located inside a sphere with radius $r_a$ around a particle (see Fig. 3(b)), then the point is an interpolation or a passive point. Otherwise, it is an internal or a boundary point. If the point is an interpolation or a passive point, it is a passive point if none of the six direct neighbours is an internal or a boundary point.

In view of the characterization of points explained above, type arrays are defined, which are integer arrays with a similar structure as the floating point arrays in Table 1 (but each Cartesian type array has one extra dummy layer around the block). The types of the pressure points of a given Cartesian grid with grid identification number $b$ are set in two rounds. In the first round, the nearest particle is determined for each Cartesian pressure point (the distance to the nearest particle center is equal to the minimum of the distances to all the particles in the block lists of the block under consideration and the 26 surrounding blocks). If the distance to the nearest particle is less than $r_a$, then the type is set equal to the grid

identification number of the nearest particle (which can be a shadow particle), otherwise the type is set equal to the default type, which is $b$. The first round includes the outer dummy layers, which are the extra dummy layers just mentioned. In the second round, in another loop performed over all Cartesian pressure points except those in the outer dummy layers, the type is overwritten by 0 if the type of none of the six direct neighbours is equal to $b$. The result of these two rounds can be summarized as follows. If the type of a grid point is equal to $b$ the point is either an internal or a boundary point (to distinguish between internal and boundary points, the values of the grid indices $i$, $j$ and $k$ of the three coordinate directions are used). If the type is zero, the grid point is passive. However, if the type is unequal to zero and unequal to $b$, then the grid point is an interpolation point and the type value represents the grid identification number from which the fluid variable at the grid point under consideration is interpolated.

The types of the staggered points are derived from the types of the two nearest cell centers (each staggered point is located on a face that joins two neighbouring cells). If the type numbers of these center points are the same, the type is copied to the staggered type. If one of them is zero, the type of the staggered point is also set to zero (passive). If one of them is an internal point and the other one an interpolation point, the staggered point is an interpolation point. Although the staggered velocity points are not explicitly indicated in Fig. 3, above explanation may be clarified by mentioning that the spherical staggered points located on the two thick solid circular lines and the Cartesian staggered points on the step line twined around the dashed circular line are not internal grid points, but interpolation points ($r = r_b$ and inner step line) or boundary points ($r = r_0$).

For a given interpolation point, the grid on which the interpolation stencil is defined (say grid $a$) is called the donor grid, while the grid to which the interpolation point itself belongs is called the acceptor grid. Each interpolation stencil contains $3 \times 3 \times 3$ points. The type of each point of the interpolation stencil should be equal to $a$, which means that each point of the interpolation stencil should either be an internal point or a boundary point. Furthermore, a boundary point due to a wall is not allowed to be a member of an interpolation stencil, except if it is a boundary point for the wall normal velocity on the wall. The interpolation stencil is chosen such that the position of the interpolation point is not further away from the central point of the stencil than from any of the other 26 points of the stencil. As an example, we consider the pressure interpolation point indicated by the enlarged plus symbol in Fig. 3(a). For this point, the Cartesian grid is the donor grid and the spherical grid is the acceptor grid. The $3 \times 3$ encircled Cartesian pressure points near the bottom right corner of Fig. 3(b) are a two-dimensional representation of the interpolation stencil. If Figs. 3(a) and 3(b) had been overlaid, then the interpolation point would have shown up inside the cell around the central point of the interpolation stencil.

To allow direct access to the interpolation points and the interpolation stencils, appropriate lists are defined: acceptor, primary donor and secondary donor lists (separate lists for the pressure and for each velocity component). The acceptor lists of a given grid $b$ contain the grid indices $(i, j, k)$ of the interpolation points on grid $b$. For the Cartesian grids, only the interpolation points that are also original points are put in the acceptor lists. For each interpolation point of $b$, there is a donor grid $a$, which contains the interpolation stencil. The acceptor lists are sorted with respect to the donor grid numbers and then mapped upon primary donor lists. The primary donor lists of grid $a$ contain the acceptor grid numbers and the acceptor grid indices $(i, j, k)$ of all interpolation points for which grid $a$ is the donor grid or, in other words, all interpolation points that are interpolated from grid $a$. The distinction between acceptor and primary donor lists is useful for the MPI parallelization (see Appendix A). Further information for the interpolations is computed and stored in the secondary donor lists, such as the indices of the reference corner of the interpolation stencils and the weights $w_{ijk}$ of the interpolations. While for each pressure interpolation point 27 weight coefficients are stored, 81 weight coefficients are stored for each interpolation point of a velocity component, because due to the staggered grids, 3 interpolation stencils are used per interpolation point per velocity component. The definition of the interpolation weights can be inferred from the interpolation formulas specified in Appendix B.

### 3.4. The radial grid

The grid in the $r$ direction can be uniform or stretched. The cell center radial locations $r_i^c$ are defined by

$$r_i^c = r_0 f(i - 1/2), \quad (i = 0, 1, ..., N_r + 1), \tag{32}$$

where $f(\xi)$ represents the stretching function, $i$ denotes the grid index, while $i = 0$ and $i = N_r + 1$ represent dummy locations. The staggered radial locations of the $u_r$ velocity are defined by

$$r_i^s = r_0 f(i), \quad (i = 0, 1, ..., N_r), \tag{33}$$

where $i = 0$ and $i = N_r$ represent dummy locations. The radial grid size is defined by $\Delta r_i^c = r_i^s - r_{i-1}^s$ at location $r_i^c$ and by $\Delta r_i^s = r_{i+1}^c - r_i^c$ at location $r_i^s$. If we wish to simulate a case in which particles are relatively close to each other, we may want to minimize $N_r$ and $\Delta r$. Then a uniform radial grid seems to be a suitable option, which is obtained if $f(\xi) = 1 + \xi(\Delta r)/r_0$ and $\Delta r = (r_b - r_0)/N_r$. If stretching is used in the radial direction, we choose $f(\xi) = \exp(\gamma\xi)$ with stretching factor $\gamma = \Delta\theta = \pi/N_\theta$ [23], which implies that $\Delta r_1^c \approx r_0 \Delta\theta$. In all cases $r_0^s = r_0 = d_p/2$.

It is possible to derive sufficient conditions for $r_a$ (the radius of the holes in the Cartesian grid) and $r_b$ (the radius of the outer boundaries of the spherical grids), such that each point of an interpolation stencil is ensured to be an internal point or a staggered dummy point on a solid wall. For simplicity, we consider uniform Cartesian grids with the same grid spacing

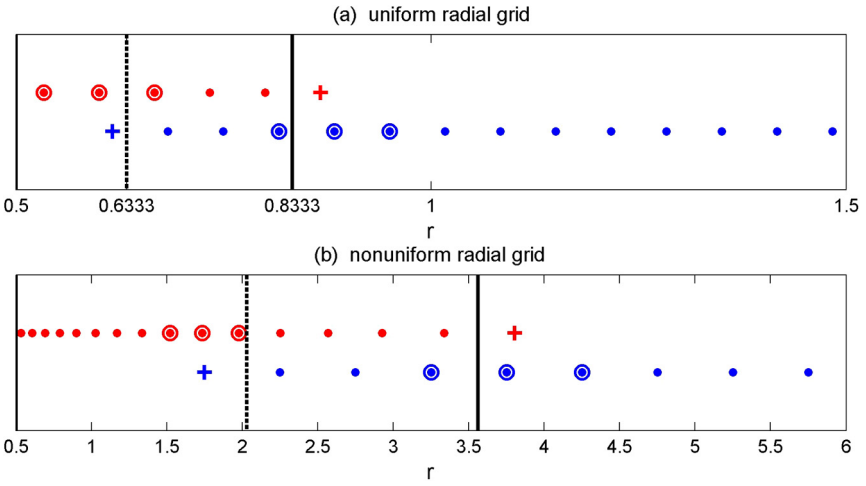


**Fig. 4.** One-dimensional illustration of the grid structure for the pressure. A uniform spherical grid with $N_r = 5$ (a) and a nonuniform spherical grid with $N_r = 15$ (b) are overset on uniform Cartesian grids. The thick vertical lines represent the three concentrical surfaces with radii $r_0$ (inner solid circle), $r_a$ (dashed circle) and $r_b$ (outer solid circle). Only internal (thick dots) and interpolation (pluses) pressure points are shown. A row of three encircled dots represents the interpolation stencil corresponding to the interpolation point (plus) drawn just above or below the stencil.

$h$ in each direction. Without loss of generality we assume that $\mathbf{x}^p = \mathbf{0}$ for the nearest particle. A one-dimensional illustration of two basic grid configurations used in this paper is shown in Fig. 4. The left and right grids in each subplot represent the spherical grid and Cartesian grid, respectively. The right grids have been shifted slightly downward, for clarity.

Consider a pressure or velocity interpolation point of the Cartesian grid located at $\mathbf{x}$. From the definition of the interpolation point, we infer that $|\mathbf{x}| \geq r_a - h$. The interpolation stencils of $\mathbf{x}$ do not include a dummy point inside a particle if $|\mathbf{x}| \geq r_1^s = r_0 + \Delta r_1^c$, which is ensured if

$$r_a - h \geq r_1^s. \tag{34}$$

Furthermore, $|\mathbf{x}| < r_a$ for a pressure location and $|\mathbf{x}| < r_a + \frac{1}{2}h$ for a velocity location. The radius of each outer dummy point of the spherical grid is at least $r_b$. Thus the velocity interpolation stencil does not include outer dummy points of the spherical grid if $|\mathbf{x}| < r_{N_r-1}^c$, which is ensured if

$$r_a + \tfrac{1}{2}h \leq r_{N_r-1}^c. \tag{35}$$

Moreover, the interpolation stencils of the spherical interpolation points should fit on the Cartesian grid. Let $\mathbf{x}$ be an interpolation point of the spherical grid. By definition, $|\mathbf{x}| \geq r_b$. The corresponding three interpolation stencils should consist of internal Cartesian velocity points. From the in total 81 stencil points, let $\mathbf{y}$ be the point farthest from $\mathbf{x}$ ($\mathbf{y}$ is not always unique). Because of the definition of the interpolation stencils,

$$|\mathbf{y} - \mathbf{x}| \leq \left( (\tfrac{3}{2}h)^2 + (\tfrac{3}{2}h)^2 + (\tfrac{3}{2}h)^2 \right)^{1/2}. \tag{36}$$

The interpolation stencil for the pressure fits if $|\mathbf{y}| \geq r_a$, but the interpolation stencils of the velocity are more critical. For a velocity interpolation stencil, we consider the two pressure points at distance $\frac{1}{2}h$ from the staggered $\mathbf{y}$, we choose the pressure point with the largest distance to $\mathbf{x}$, and call it $\mathbf{z}$. The latter distance satisfies

$$|\mathbf{z} - \mathbf{x}| \leq ((2h)^2 + (\tfrac{3}{2}h)^2 + (\tfrac{3}{2}h)^2)^{1/2} = (\tfrac{17}{2})^{1/2}h. \tag{37}$$

Since the point $\mathbf{y}$ is internal if the pressure point $\mathbf{z}$ is internal, the velocity interpolation stencils fit on the internal points of the Cartesian grid if $|\mathbf{z}| \geq r_a$, which is ensured if

$$r_b - (\tfrac{17}{2})^{1/2}h \geq r_a. \tag{38}$$

The argumentation also applies to multiple particles, provided the distance of each spherical interpolation point to the nearest particle is not smaller than $r_b$. This means that all interpolations can be performed if the distance between two particle centers is at least $r_b + r_{N_r+1}^c$ and the inequalities (34), (35) and (38) are satisfied. Mutual overlapping spherical grids are avoided as long as the minimum interparticle distance, $d_{min}$, satisfies

$$d_{min} \geq r_b + r_{N_r+1}^c. \tag{39}$$

In addition, as long as the minimum distance between a particle and a wall, $d_{min,wall}$, satisfies

$$d_{min,wall} \geq r^c_{N_r+1} + h, \tag{40}$$

the presence of a wall does not hinder the interpolations.

It is instructive to simplify the set of conditions (34), (35), (38) and expressions (39) and (40) for cases with uniform grid size $\Delta r$:

$$r_a \geq r_0 + \Delta r + h, \tag{41}$$

$$r_b \geq r_a + \tfrac{1}{2}h + \tfrac{3}{2}\Delta r, \tag{42}$$

$$r_b \geq r_a + (\tfrac{17}{2})^{1/2}h \approx r_a + 2.92h, \tag{43}$$

$$d_{min} \geq 2r_b + \tfrac{1}{2}\Delta r, \tag{44}$$

$$d_{min,wall} \geq r_b + \tfrac{1}{2}\Delta r + h. \tag{45}$$

In case $\Delta r = h$, the minimum suitable value for $r_b$ to guarantee the existence of the interpolation stencils for any configuration with a minimum interparticle distance of $2r_b + \frac{1}{2}\Delta r$ is $r_b = r_0 + 5h$. Then $r_a$ should be chosen between $r_0 + 2h$ and $r_0 + 2.08h$. Then $d_{min}$ should be at least $2r_0 + 10.5h$, which implies that the width of the gap between two particles should remain at least $10.5h$. The minimum width of the gap between a particle and a flat wall should remain at least $6.5h$ if $\Delta r = h$. In theory, arbitrarily small nonzero gap widths are allowed in the limit $h \to 0$ and $\Delta t \to 0$. In practice, a very large number of grid points will be required for a simulation that allows very small gap widths.

### 3.5. Details of the initialization

The initialization procedure consists of 7 substeps:

1. Define the spatial domains. Store the coordinates, the grid spacings, the elements of matrix **A**, and some metric quantities for cell-center and the staggered locations. Store also the matrix coefficients of the left-hand side of the pressure Poisson equation.
2. Initialize time $t$ and evaluation time $t_{eval}$, particle positions, and particle and fluid velocities and pressure.
3. Compute the velocity field on the particle surfaces.
4. Set up the interpolation lists.
5. Set the velocity and pressure at all dummy and interpolation points.
6. Make the initial velocity field divergence free (optional).
7. Call the evaluation routine if $t = t_{eval}$.

In substep 1, the initial reference position vectors and the sizes of the Cartesian and spherical domains are defined. Then the coordinates of all cell center and staggered points are defined. Each cell centered grid spacing is a distance between two staggered points, while each staggered grid spacing is the distance between two cell centers. It is important to mention that the elements of matrix **A** and metric quantities are directly computed by inserting $r$, $\theta$ and $\phi$, for the cell center and for the staggered locations, such that none of these quantities is obtained by interpolation.

The setup of the interpolation lists (substep 5) consists of several routines. First the block lists of particles (defined in subsection 3.2) are determined. Second, the block lists are used to find the distance to the nearest particle center. This distance and the corresponding particle number are stored for each Cartesian pressure point (cell center location). Third, the types of the grid points are set as described in subsection 3.3. Fourth, the acceptor and donor interpolations lists are set-up, and the interpolation weights are computed.

In substep 5 of the initialization routine, the interpolations are performed for the first time. This means that two interpolation procedures are called, one for the velocity, the other one for the pressure. Both the velocity and pressure interpolation procedure consist of three main actions: (1) call the boundary condition routine of the flow variable under consideration (set the values at dummy points), (2) perform the interpolations (set the variable at interpolation points), (3) call the boundary condition routine again.

In the boundary condition routines, the following actions are performed. The variables at the Cartesian dummy points are copied from the corresponding points in adjacent cell block or, in the case of periodic boundary conditions, from a block that becomes adjacent after a periodic shift. If the boundary is a wall, the normal velocity is prescribed at the wall location, while the tangential velocity components are extrapolated using the third-order extrapolation formula specified in an appendix of [26]. For each spherical grid, $u_r$ is set to zero at the particle surface. The tangential velocity components, $u_\theta$ and $u_\phi$, are set at the dummy points inside the particle, using the third-order extrapolation formula just mentioned. This extrapolation is based on the tangential velocity of the wall and the velocity at two internal grid points. The pressure at the dummy points is copied from the first layer of internal points ($q(r^c_0) = q(r^c_1)$). It is stressed that the pressure at the dummy points is only used by the procedure that solves the pressure Poisson equation. It does not mean at all that a homogeneous Neumann boundary condition of the pressure is prescribed to the continuous pressure Poisson equation, since the discrete pressure equation at $r^c_1$ should not be regarded as a discretization of the continuous pressure Poisson equation [26]. The values of the variables in the dummy points in the $\phi$ direction are obtained using the periodicity in that direction. For the poles ($\theta = 0$ and $\theta = \pi$),

the fourth-order interpolation formula specified in the appendix of [23] is used to calculate $u_\theta$ at the poles. It was shown in [23] that this pole treatment is adequate. In fact, this is the only pole treatment that is required for the discretization of the spherical Navier–Stokes equations, since many metrical quantities are zero if $\theta = 0$ or $\theta = \pi$. However, for the interpolation routines it is necessary to use meaningful extensions for the dummy grid cells in the $\theta$ direction (thus for $\theta < 0$ and for $\theta > \pi$). Near $\theta = 0$, we define for $\theta < 0$: $u_r(r,\theta,\phi,t) = u_r(r,-\theta,\phi+\pi,t)$, $u_\theta(r,\theta,\phi,t) = -u_\theta(r,-\theta,\phi+\pi,t)$, $u_\phi(r,\theta,\phi,t) = -u_\phi(r,-\theta,\phi+\pi,t)$ and $q(r,\theta,\phi,t) = q(r,-\theta,\phi+\pi,t)$. Analogous extensions are applied near $\theta = \pi$.

Substep 6, a correction of the initial velocity field, is optional and not applied by default (it has only been used in the simulations presented in subsections 4.1–4.3). To make the velocity field divergence free (down to the truncation error), $\nabla^2\psi = \nabla\cdot\mathbf{u}$ is solved iteratively, and $\nabla\psi$ is subtracted from $\mathbf{u}$. Although $\psi$ does not represent the variable $q$ at $t = 0$, the procedure that solves the Poisson equations for $\psi$ is the same as the procedure that solves the Poisson equations for $q$.

In the last substep of the initialization procedure the evaluation routine is called if $t$ is equal to $t_{eval}$, the evaluation time. This is a post processing routine, intermediate results are computed and written to file, for example forces and torques on particles, errors and some spatial integrals. In addition, $t_{eval}$ is set to the next evaluation time.

### 3.6. Details of the time step

The main parts of the time step and temporal and spatial discretization have been presented in section 3.1. In this section, a complete list of all actions performed during a time step is given, and additional explanation is provided. Each time step consists of 16 substeps:

1. Increase $t$ with the time step $\Delta t$.
2. Store a copy of the particle velocities in $\mathbf{v}^p_{old}$.
3. Compute the hydrodynamic forces and torques on the particles, $\mathbf{F}^p$ and $\mathbf{T}^p$.
4. Update the particle positions by incrementing the particle position vector $\mathbf{x}^p$ by $\Delta t\ \mathbf{v}^p$.
5. Increase $\mathbf{v}^p$ by $\Delta t\ (\mathbf{F}^p + \rho_p V_p \mathbf{g} + \rho V_p \mathbf{a})/(\rho_p V_p)$ and increase $\boldsymbol{\omega}^p$ by $\Delta t\ \mathbf{T}^p/I_p$.
6. Subtract $\mathbf{A}(\mathbf{v}^p - \mathbf{v}^p_{old})$ from $\tilde{\mathbf{u}}$ in each spherical domain. This accounts for the term $-\mathbf{A}\mathbf{v}^p_{,t}$ in (8) to (10).
7. Compute the velocity on the particle surfaces.
8. Set up the interpolation lists.
9. Set the velocity at dummy and interpolation points.
10. Compute the (second) intermediate fluid velocity (update the momentum equation without the pressure gradient term).
11. Set the velocity in dummy and interpolation points.
12. Compute the divergence of the intermediate fluid velocity.
13. Solve the pressure Poisson equation iteratively (set the pressure at dummy and interpolation points in each iteration).
14. Add the pressure gradient term to the fluid velocity.
15. Set the velocity in dummy and interpolation points.
16. Call the evaluation routine if $t = t_{eval}$.

In substep 3, the hydrodynamic forces and torques exerted on the particles are computed. The integrals over the particle surface in definitions (22) and (23) are performed using the midpoint rule. The tangential velocity derivatives at the wall are computed by central differences using the dummy points inside the particle and the first layer of internal points near the particle surface. The pressure at the particle surface is obtained by second-order accurate extrapolation using the first two layers of internal points near the particle surface,

$$q(r_0) = q(r_1^c) + (q(r_1^c) - q(r_2^c))\frac{r_1^c - r_0}{r_2^c - r_1^c}. \tag{46}$$

The pressure values in the dummy cells inside the particle are not used in this step. In substep 4, the particle locations are updated. For each periodic direction $i$, the domain length $L_i$ is added or subtracted from $x_i^p$ if that is required to keep $x_i^p$ inside the domain $\Omega_c$. In substep 5, the hydrodynamic forces and torques computed in substep 3 are used to further update the velocities and the angular velocities of the particles. Substeps 3 to 5 are only performed for freely moving particles. In the results section, we present several simulations of spheres with a prescribed motion. In those cases, substeps 3 to 5 are replaced by statements that compute $\mathbf{x}^p$, $\mathbf{v}^p$ and $\boldsymbol{\omega}^p$ by substitution of time $t + \Delta t$ in the functions that prescribe the motion of the particles.

In substep 6, the fluid velocity at the internal points of each spherical domain is modified to account for the change of the particle velocity. In substep 7, the surface velocity of each particle is modified to account for the change of the angular velocity of the particle. In substep 8, the new interpolation structure is set up, based on the new positions of the particles. Substep 8 includes an update of the types of all grid points. This update leads to so-called exposed points [17], internal points that were not internal before the types were updated. If $v_{max}\Delta t$ is sufficiently small ($v_{max}$ is the maximum of the magnitude of the particle velocity), each exposed point was an interpolation point in the previous time step, which implies that the variable at the exposed point has a meaningful value. Since for the present overset grid method only Cartesian points can be exposed, the condition $v_{max}\Delta t < h/\sqrt{3}$ should be satisfied. This is a mild restriction, such that no

special treatment of exposed points is needed, provided the spatial discretization is based on one previous time level ($n$). In substep 9, the first intermediate spherical and Cartesian velocities are completed by computing their values at interpolation and dummy points, using the new interpolation structure and the results of substeps 6 and 7.

The method reduces to the standard explicit Euler method for the intermediate velocity update if all particles are fixed. In the method restricted to fixed particles [23], the explicit part of the convective terms was based on two time levels ($n$ and $n-1$) and second-order in time. This has some advantages, higher accuracy of the convective motion and less severe convective restriction on the time step if the viscosity is low, but it also requires that the velocity at interpolation points is available at time levels $n$ and $n-1$. However, if particles are moving, an interpolation point at time level $n$ may have been a passive point at time level $n-1$. Thus for the convective term at exposed points the time level $n-1$ should not be used. To avoid a special treatment of exposed points, the temporal discretization of the convective terms has been simplified such that time level $n-1$ is not used at all anymore.

The time step $\Delta t$ remains fixed during a simulation. For the simulations in the present paper, the viscous stability criterion is much more restrictive than the common convective criterion that the Courant number should be less than one. The theoretical upperbound for numerical stability of the explicit viscous terms is given by

$$\Delta t_{visc} = \tfrac{1}{2}\nu/\min\Big((\Delta r_1^c)^{-2} + (r_0\Delta\theta)^{-2}, 3h^{-2}\Big). \tag{47}$$

Each $\Delta t$ used in this paper satisfies $\Delta t < 0.6\Delta t_{visc}$. This implies that the temporal truncation error, which is proportional to $\Delta t$, is second order in terms of the spatial grid spacing, like the truncation errors in the spatial derivatives. Another factor relevant for numerical stability is the density ratio $\rho_p/\rho$, which is at least one in all cases presented in this paper. However, it was found that the present method is unstable for $\rho_p/\rho < \frac{1}{2}$, while for $\rho_p/\rho \downarrow \frac{1}{2}$, the time step needs to be reduced for stability. The instability for $\rho_p/\rho < \frac{1}{2}$ is related to added mass effects in combination with the explicit advancement of the particle translational velocities, see for example [27].

In substep 13, the pressure Poisson equation problem is iteratively solved, as explained in subsection 3.1. The stopping criterion of the iterative method is that the maximum norm of the residual vector is less than a specified tolerance, $10^{-5}\min(u_{ref}^2/d_p, \nu u_{ref}/d_p^2)$, where $u_{ref}$ is a reference value for the fluid velocity relative to the particles. This tolerance can be regarded as an estimate of the absolute error in the pressure caused by the finite number of iterations. The pressure interpolation routine, executed in each iteration, includes the assignment $q(r_0^c) = q(r_1^c)$. As discussed in the previous subsection, this does not reflect a physical pressure boundary condition, but is just a convenient choice for the discrete system, a choice that implies that the boundary condition that should be imposed on the intermediate velocity is $u_r^{**} = 0$ at $r_0$. The latter is done in substep 11, before the divergence is taken in substep 12.

Even if the linear system were solved down to machine precision, the discrete divergence of the fluid velocity would not be zero down to machine precision. The first cause is the augmented matrix technique, which introduces a small error at each internal point. The second cause is the interpolation in substep 15, which leads to a nonzero velocity divergence at internal pressure points with at least one neighbouring velocity interpolation point. Both errors are truncation errors, and are therefore expected to converge to zero if the grid size is refined to zero. The second cause can be avoided if, instead of the pressure at pressure interpolation points, the pressure gradient is interpolated at velocity interpolation points in each iteration of the pressure Poisson equation [20]. Therefore, interpolation of the pressure gradient in combination with the BiCGstab iterative method was also implemented by the author of the present paper, but without success. The computational time per iteration was significantly increased and the iterative procedure failed to converge.

For completeness, it is mentioned that if two particles become too close, the program does not terminate, but modifies the velocities of the two particles according to an artificial hard-sphere repulsion (in between substeps 2 and 3). The repulsion occurs if the particles approach each other and the distance between the centers becomes smaller than $d_{min} + v_{max}\Delta t$. If a repulsion occurs, the user is warned. However, no repulsions occurred in the simulations presented in this paper. In particle resolved simulations performed with other methods, collisions have been modelled by soft-sphere collisions (or soft-sphere short-range repulsions), sometimes in conjunction with a lubrication model, see for example [2,5,9].

## 4. Results of Stokes simulations

In this section, we show that canonical analytical solutions of the (unsteady) Stokes equations for flows around a single sphere have been reproduced by the code. Results of four test cases are presented in four subsections. In each case $d_p = 1$, $\nu = 1$ and $\rho = 1$, while the acceleration terms $\mathbf{a}$ and $\mathbf{g}$ are zero unless mentioned otherwise. The Cartesian domain $\Omega_c$ is cubical ($L_1 = L_2 = L_3 = L$) and the Cartesian grid cells are cubes of uniform edge length $h$, which implies $N_1 = N_2 = N_3$. Each simulation was run in parallel on eight processors, using MPI, for which the Cartesian domain was divided into $2\times2\times2$ blocks. The computing times can be found in Appendix C.

In the simulations presented in this section, the convective terms in the code were multiplied by zero in the equations on the spherical grid, while they were replaced by the term $\mathbf{v}^p(t)\cdot\nabla\mathbf{u}$ in the equations on the Cartesian grid, where $\nabla\mathbf{u}$ was discretized by the standard second-order central difference scheme. The analytical Stokes solutions used are solutions of the unsteady Stokes equations in the frame of reference of the sphere. Since the unsteady Stokes equations are not Galilean invariant, the convective term $\mathbf{v}^p(t)\cdot\nabla\mathbf{u}$ appears in the unsteady Stokes equations after transformation to the Cartesian frame of reference.

**Table 2**
Definition of configurations X1, X2, X2a and X2b used in subsections 4.1–4.3, where X stands for Tran, Osc or Rot. The grid in radial direction is uniform with $\Delta r = h$. In each case $r_a = r_0 + 0.4(r_b - r_0) + 10^{-8}$.

| | $\mathbf{V}$ or $\boldsymbol{\omega}_0$ | $d_p/h$ | $N_r \times N_\theta \times N_\phi$ | $N_1^3$ | $(r_b - r_0)/d_p$ | $\Delta t$ |
|---|---|---|---|---|---|---|
| X1 | $\mathbf{e}_1$ | 15 | $5 \times 24 \times 48$ | $60^3$ | 1/3 | 0.0004 |
| X2 | $\mathbf{e}_1$ | 30 | $10 \times 48 \times 96$ | $120^3$ | 1/3 | 0.0001 |
| X2a | $\mathbf{e}_1$ | 30 | $5 \times 48 \times 96$ | $120^3$ | 1/6 | 0.0001 |
| X2b | $\mathbf{e}_3$ | 30 | $10 \times 48 \times 96$ | $120^3$ | 1/3 | 0.0001 |

### 4.1. Translating sphere

In this and the following two subsections, where we consider the Stokes flows due to a translating, oscillating and rotating sphere, the size of $\Omega_c$ is given by $L_1 = 4$. The center of $\Omega_c$ is denoted by $\mathbf{x}^{cen}$. At $t = 0$, the fluid velocity is set equal to the exact velocity. Each time the boundary condition routine is called, the exact fluid velocity vector at actual time $t$ is copied to the dummy points on and near the outer faces of $\Omega_c$. This is combined with zeroth order extrapolation of the pressure, which, as explained in section 3.4, is consistent since the exact velocity is also imposed on the intermediate velocity on the faces of $\Omega_c$, just before the divergence operator is applied. The resolutions used are summarized in Table 2. Each grid is uniform in the radial direction and $\Delta r = h$. Moreover, $r_a = r_0 + 0.4(r_b - r_0) + 10^{-8}$ (the grid structure is illustrated in Fig. 4(a)). In order to allow a meaningful comparison with the exact pressure, the simulated pressure is modified by subtracting the simulated pressure averaged over the particle surface. The subscript "exact" refers to the corresponding analytical solution.

In the first test case, we consider Stokes flow around a sphere translating with constant velocity $\mathbf{V}$. The translation is given by:

$$\mathbf{x}^p(t) = \mathbf{x}^{cen} + \mathbf{V}t, \qquad \mathbf{v}^p(t) = \mathbf{V}, \qquad \boldsymbol{\omega}^p = \mathbf{0}. \tag{48}$$

The exact solution can be written as [28] (p. 265):

$$\mathbf{u}(\mathbf{x}, t) = \frac{r_0}{4r}\left[3 + \left(\frac{r_0}{r}\right)^2\right]\mathbf{V} + \frac{3r_0}{4r^3}\left[1 - \left(\frac{r_0}{r}\right)^2\right](\mathbf{V} \cdot \mathbf{r})\mathbf{r}, \tag{49}$$

$$q(\mathbf{x}, t) = \frac{3\nu r_0}{2r^3}(\mathbf{V} \cdot \mathbf{r}), \tag{50}$$

$$\mathbf{F}^p = -6\pi\rho\nu r_0 \mathbf{V}, \qquad \mathbf{T}^p = \mathbf{0}, \tag{51}$$

where $\mathbf{r} = \mathbf{x} - \mathbf{x}^p(t)$ and $r = |\mathbf{r}|$. In the numerical tests, we use $\mathbf{V} = \mathbf{e}_1$ or $\mathbf{V} = \mathbf{e}_3$ ($u_{ref} = |\mathbf{V}| = 1$). These two choices are numerically not equivalent, because the polar axis in spherical coordinates is always aligned with $\mathbf{e}_3$, by definition.

The translating sphere simulations performed on the grids specified in Table 2 are indicated by Tran1, Tran2, Tran2a and Tran2b. The number in each name refers to the level of refinement. The deviations from the analytical solutions are shown in Fig. 5, as functions of time: the relative error in the particle force and the maximum norms of the velocity divergence, the error in the velocity and the error in the pressure. The discrete maximum norm is defined as the maximum of the absolute value of a quantity over the centers of all internal cells in the spherical domain and the centers of all internal cells in the Cartesian domain (thus the overlap region is used twice). This is straightforward if the quantity is the pressure or velocity divergence, since these are defined at the centers of the cells. For a velocity difference, we first square the velocity differences of each component at their staggered locations, then interpolate these squares to the centers of the cell using the weights 1/2 and 1/2, then we sum the three squares, and then the root of this sum is inserted as argument into the discrete maximum norm. The maximum norm of the error in the velocity is normalized by $V = |\mathbf{V}| = 1$. The force is normalized by $F_{exact} = 3\pi\rho\nu d_p V = 3\pi$. The maximum norm of the error in the pressure is normalized by the maximum of the analytical pressure field, $q_{\max} = 3\nu V/d_p = 3$.

All errors are around 1% or less for the coarsest grid and they decrease upon grid refinement (Fig. 5). Comparison between Tran2 and Tran2a shows that it matters if $r_a$ and $r_b$ are kept constant during grid refinement (Tran2) or if the radial extent of the grid shrinks during grid refinement (Tran2a). All errors are reduced in both cases, but the reduction is largest if $r_a$ and $r_b$ are kept constant. Thus fixed overlap during grid refinement reduces the error more than shrinking overlap. This was also found by others [17,20]. For fixed overlap, the particle force and the velocity field display second-order accuracy, but the pressure and the velocity divergence only first-order accuracy. The errors in case Tran2b, in which the velocity of the sphere is directed along the polar line, are comparable to those in Tran2, except the error of the force, which is much smaller in Tran2b than in Tran2.

It is surprising that the accuracy of the pressure in an overset grid method based on second-order finite differences and third-order interpolations reduces to first-order (at some times at some locations). It is mentioned that the requirement of vanishing maximum norms of truncation errors in simulations of flows around moving objects can be regarded as severe tests, because pointwise convergence of the numerical solution to nontrivial exact solutions is verified. The first-order accuracy of the pressure can probably not be attributed to unsteady behaviour, since it also occurs for uniform Stokes flow past

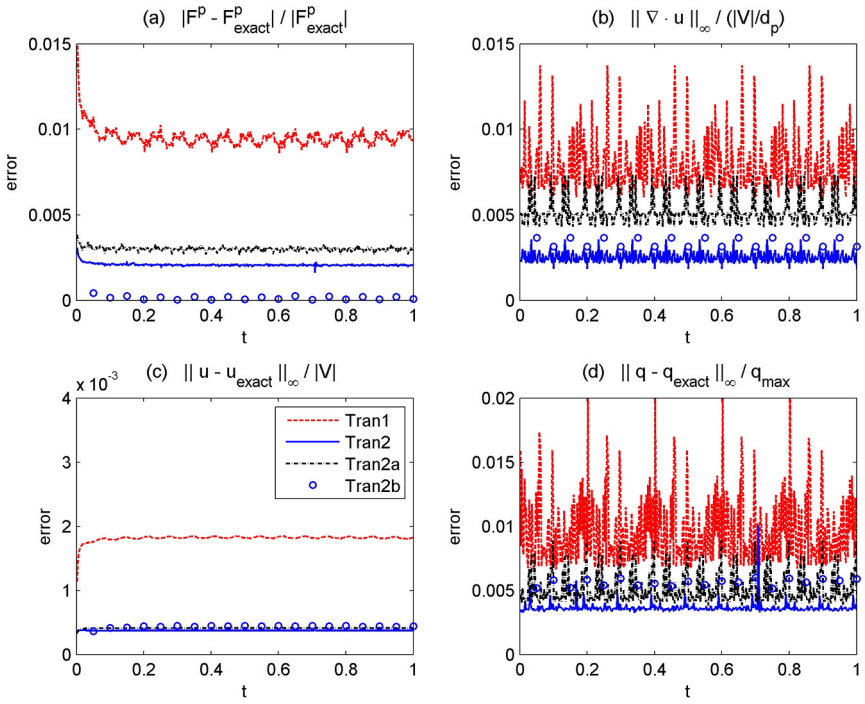


**Fig. 5.** Translating sphere. Simulations Tran1 (red dashed), Tran2 (blue solid), Tran2a (black dashed) and Tran2b (circles): (a) Relative error in the particle force, (b) maximum norm of the velocity divergence, (c) maximum norm of the error in the velocity, and (d) maximum error of the error in the pressure. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

a fixed sphere, which is a steady state flow (see Appendix D, where also a reduced order of accuracy is observed for the pressure, in the maximum norm and to smaller extent also in the $L_2$ norm).

The reduction of the order of accuracy of the divergence and the pressure can be caused by the abrupt change of the structure of the discretization error in the overlapping region. Consider an internal velocity point adjacent to an interpolation point. The leading terms of the truncation errors in both points are both second order (since the velocity is shown to be second-order accurate), but their prefactors may be discontinuous and then the accuracy of the derivative based on these two points can reduce to first order. The same argument shows that the accuracy of a second-order velocity derivative based on three points with one of them being an interpolation point can locally reduce to zeroth order. Because of the projection of the second intermediate velocity, part of the viscous term induces part of the pressure gradient. Thus if the viscous term is zeroth order at some points, the pressure gradient required to make the velocity divergence free can be a zeroth order quantity at some points. If this is the case, the pressure itself can be first-order accurate at some points.

To validate a three-dimensional particle-resolved simulation method, analytical solutions of flows around an isolated sphere have been used before, but apparently only for steady flow around a fixed non-rotating and a fixed rotating sphere. For example, Tang et al. [29] and Baltussen [30] used the Stokes expressions for the force and torque on an isolated fixed sphere to test their immersed boundary methods. Kempe et al. [31] used the exact solutions for both no-slip and free-slip surface conditions to test an immersed boundary method for steady flow past a fixed sphere. For the free slip case, the maximum norm of the error in the velocity was shown, which remained (slightly) above 1% in all tests. According to Ref. [23], the maximum norm of the velocity error dropped below 0.2% in overset grid simulations of steady flow past a fixed sphere with a no-slip surface. Moreover, Hasimoto's analytical expressions for the drag force exerted on spheres in ordered particle arrays have been used in verification of numerical methods, see for example [8,3,5,29]. Not for the purpose of validation, but to investigate the effect of Reynolds number ranging from 0.1 to 100, Mei [32,33] has simulated unsteady flow past a fixed sphere, using an axisymmetric flow solver.

It is remarked that in Ref. [23] the Cartesian and spherical domains were much larger than in this section since the spherical grid was stretched. The present test is more challenging, not only because the sphere moves but also because the interpolations are performed much closer to the surface of the sphere. Near the surface the variation of the pressure is relatively strong (the pressure decays with the square of the distance to the center).

### *4.2. Oscillating sphere*

The second test case is an oscillating sphere. The oscillation is given by

$$\mathbf{x}^p(t) = \mathbf{x}^{cen} - \Re\Big(\frac{e^{-i\omega t}}{\omega}\mathbf{V}\Big), \qquad \mathbf{v}^p(t) = \Re\Big(ie^{-i\omega t}\mathbf{V}\Big), \qquad \boldsymbol{\omega}^p = \mathbf{0}, \tag{52}$$

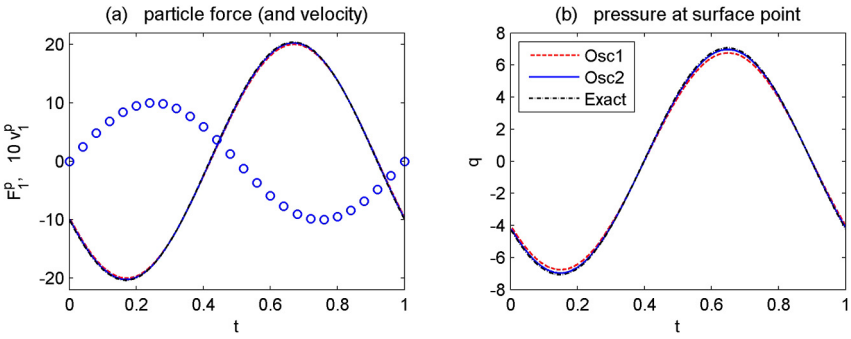


**Fig. 6.** Oscillating sphere. Simulations Osc1 (red dashed), Osc2 (blue solid) and the analytical solution (black dash-dotted): (a) force exerted on the oscillating sphere, while the prescribed particle velocity oscillation is shown after multiplication by ten (circles); (b) the pressure at a stagnation point on the surface of the sphere ($\theta = \pi/2$ and $\phi = 0$). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $i^2 = -1$ (only in this subsection $i$ represents the imaginary unit), the function $\Re(.)$ takes the real part of its argument, while the constants $\mathbf{V}$ and $\omega$ denote the velocity amplitude and the oscillation frequency, respectively. The exact solution, presented in terms of Stokeslets in [28] (p. 310), can be rewritten as

$$\mathbf{u}(\mathbf{x},t) = \Re\Big(\frac{ie^{-i\omega t}B}{8\pi\nu}\Big[\Big(2e^{R}(1+\frac{1}{R}+\frac{1}{R^2})-\frac{2}{R^2}\Big)\mathbf{V}+\Big(\frac{6}{R^2}-2e^{R}(1+\frac{3}{R}+\frac{3}{R^2})\Big)\frac{\mathbf{V}\cdot\mathbf{r}}{r^3}\mathbf{r}\Big]\Big) + \Re\Big(\frac{ie^{-i\omega t}Q}{4\pi}\Big[-e^{R}(1+R+R^2)\mathbf{V}+3e^{R}(1+R+\frac{R^2}{3})\frac{\mathbf{V}\cdot\mathbf{r}}{r^5}\mathbf{r}\Big]\Big), \tag{53}$$

$$q(\mathbf{x},t) = \Re\Big(\frac{ie^{-i\omega t}B}{4\pi r^3}(\mathbf{V}\cdot\mathbf{r})\Big), \tag{54}$$

$$\mathbf{F}^p = -6\pi\nu r_0\rho\,\Re\Big(ie^{-i\omega t}(1+\lambda+\lambda^2/9)\mathbf{V}\Big), \qquad \mathbf{T}^p = \mathbf{0}, \tag{55}$$

where $\mathbf{r} = \mathbf{x} - \mathbf{x}^p(t)$ and $r = |\mathbf{r}|$, while

$$B = 6\pi\nu r_0(1+\lambda+\lambda^2/3), \qquad Q = -6\pi r_0^3(e^{\lambda}-1), \qquad R = \frac{\lambda r}{r_0}, \tag{56}$$

$$\lambda = (1-i)r_0\Big(\frac{|\omega|}{2\nu}\Big)^{1/2}. \tag{57}$$

In the numerical tests, (52) is prescribed, using $\omega = 2\pi$, and $\mathbf{V} = \mathbf{e}_1$ or $\mathbf{V} = \mathbf{e}_3$.

The oscillating sphere simulations performed on the grids specified in Table 2 are indicated by Osc1, Osc2, Osc2a and Osc2b. The force exerted on the sphere and the pressure at stagnation point $(\theta,\phi) = (\pi/2, 0)$ on the particle surface are shown in Fig. 6 for Osc1 (coarse grid) and Osc2 (fine grid), together with the analytical solutions. The amplitude of the force, $F_{max} \approx 20.4$, is more than two times higher than $3\pi$, the magnitude of the force exerted on the sphere translating with velocity $V = 1$. This indicates that the oscillating flow is essentially different from the translating case and that it can certainly not be considered as a quasi-steady flow in the frame of reference of the sphere. In Fig. 6(a) the curves for the force are on top of each other, while in Fig. 6(b) slight underprediction of the pressure amplitude by the coarse grid simulation can be observed. The amplitude of the pressure at this location equals $q_{\max}$, the maximum of the absolute pressure over space and time ($q_{\max} \approx 7.1$).

Results for the relative error of the force and the maximum norms of the velocity divergence and the errors in the velocity vector and pressure are shown in Fig. 7, for simulations Osc1, Osc2, Osc2a and Osc2b. Comparing Osc2 to Osc2a, we again notice that the error reduction by grid refinement is larger for grid 2 (fixed overlap) than for grid 2a (shrinking overlap).

In cases Osc2 and Osc2b, the same grid is used, but the direction of the velocity vector is different. The results of these two cases are equally good. The figures indicate pointwise convergence of the numerical solution to the analytical solution upon grid refinement. To check the approximate order of convergence it may be more appropriate to look to the temporal maximum or average of the spatial norm than to individual times. For the velocity, we see that the maximum over time reduces by a factor 4 upon grid refinement, indicating that the velocity is second-order accurate. Whereas for the translating sphere, second-order accurate convergence of the particle force was observed, the error reduction of the force upon grid refinement is reduced by factor of approximately 2.8, which suggests that the order of convergence is 1.5 in this case. The pressure and velocity divergence display roughly first-order convergence in the maximum norm (see the discussion on the order of accuracy in the previous subsection). It is not known whether other methods would provide better than first-order pointwise convergence of the pressure if they were applied to this problem. Despite the first-order spatial accuracy of the

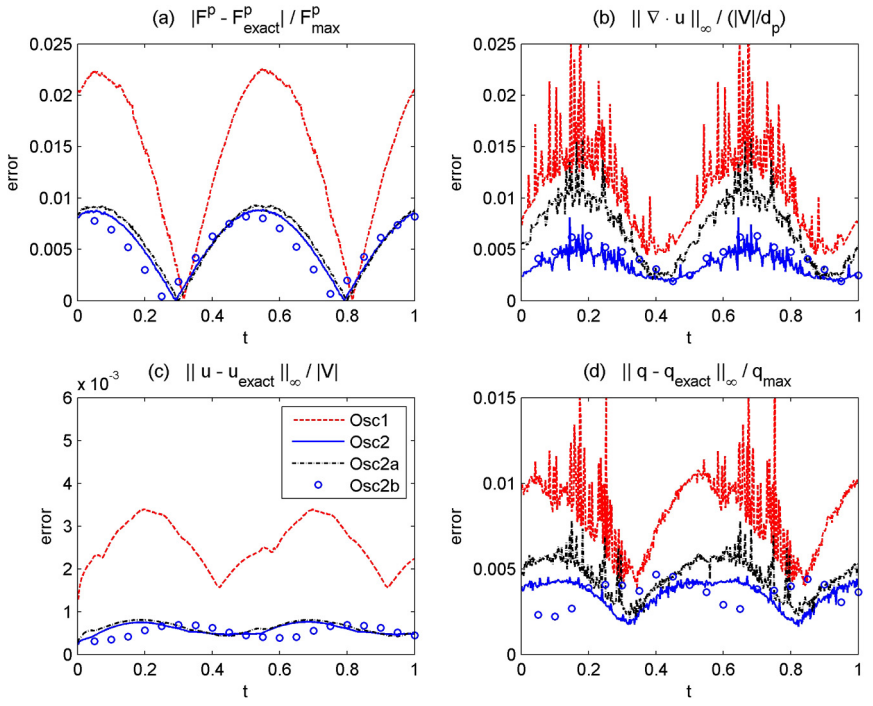


**Fig. 7.** Oscillating sphere. Simulations Osc1 (red dashed), Osc2 (blue solid), Osc2a (black dashed) and Osc2b (circles): (a) Relative error in the particle force, (b) maximum norm of the velocity divergence, (c) maximum norm of the error in the velocity, and (d) maximum error of the error in the pressure. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

**Table 3**
Results for the rotating sphere at $t = 0.3$. The maximum norms of the velocity difference, pressure difference and velocity divergence are normalized by $|\boldsymbol{\omega}_0| d_p$, $|\boldsymbol{\omega}_0| d_p \nu$ and $|\boldsymbol{\omega}_0|$, respectively.

| Case | $d_p/h$ | $\lvert\mathbf{T}^p - \mathbf{T}^p_{exact}\rvert / \lvert\mathbf{T}^p_{exact}\rvert$ | $\lVert\mathbf{u} - \mathbf{u}_{exact}\rVert_\infty$ | $\lVert q - q_{exact}\rVert_\infty$ | $\lVert\nabla \cdot \mathbf{u}\rVert_\infty$ |
|---|---|---|---|---|---|
| Rot1 | 15 | 0.00026 | 0.00150 | 0.0189 | 0.0059 |
| Rot2 | 30 | 0.00032 | 0.00041 | 0.0065 | 0.0024 |
| Rot1a | 30 | 0.00047 | 0.00048 | 0.0103 | 0.0037 |
| Rot2a | 30 | 0.00030 | 0.00017 | 0.0048 | 0.0024 |

pressure, all errors, including those in the pressure, are less than 1% of the reference values in all internal grid cells for all times for 30 grid points per particle diameter. In conclusion, this test case is a further confirmation of adequate performance of the present overset grid method.

### 4.3. Rotating sphere

The third test case is a rotating sphere. The rotation, which is the only movement of the sphere, is given by

$$\mathbf{x}^p(t) = \mathbf{x}^{cen}, \qquad \mathbf{v}^p(t) = \mathbf{0}, \qquad \boldsymbol{\omega}^p = \boldsymbol{\omega}_0. \tag{58}$$

The exact solution can be written as [28] (p. 268):

$$\mathbf{u}(\mathbf{x}, t) = (\boldsymbol{\omega}_0 \times \mathbf{r})\left(\frac{r_0}{r}\right)^3, \qquad q(\mathbf{x}, t) = \mathbf{0}, \qquad \mathbf{F}^p = \mathbf{0}, \tag{59}$$

$$\mathbf{T}^p = -8\pi\rho\nu r_0^3 \boldsymbol{\omega}_0, \tag{60}$$

where $\mathbf{r} = \mathbf{x} - \mathbf{x}^p(t)$ and $r = |\mathbf{r}|$. Since the translational velocity of the sphere is zero, this is a solution of the (steady) Stokes equations in both the Cartesian and spherical frames of reference. Baltussen [30] used this flow to test a second-order immersed boundary method and concluded that $d_p/h > 80$ was required to predict the torque within 1% accuracy.

Table 3 shows the steady state results of the overset grid method simulations, which were performed on the spatial grids specified in Table 2 ($u_{ref} = \pi |\boldsymbol{\omega}_0| d_p = \pi$) for $0 \le t \le 0.3$. The discrepancies with the analytical solution are small. The torque errors are very small, less than 0.05% in all cases. In fact, the coarse grid leads to the smallest torque error. This is probably due to a fortunate cancellation of errors for this quantity since the maximum norms do show that the simulation

on the coarse grid (Rot1) is the least accurate one. The reduction of the maximum norms in case Rot2 compared to those in case Rot2a (grid refinement with fixed overlap) clearly indicates pointwise convergence of the numerical to the analytical solution. Overall, the smallest errors are obtained if the torque directed into the $x_3$ direction (Rot2b), which is not surprising since in this case there is only one component of the spherical velocity nonzero and that one is constant ($u_\phi$).

### 4.4. Instantaneously accelerated sphere

In the fourth test case, a step change is applied to the velocity of a sphere. Initially the fluid and the sphere are both at rest. At $t = 0$, the sphere is given an instantaneous impulse $\rho_p V_p v_0 \mathbf{e}_1$ and afterwards the system evolves freely. There is no gravity.

An exact closed system for the velocity $\mathbf{v}^p = v(t)\mathbf{e}_1$ of the sphere is given by

$$\rho_p V_p \frac{dv(t)}{dt} = F(t), \quad v(t) = 0 \text{ if } t < 0, \tag{61}$$

$$F(t) = -6\pi\rho\nu r_0 v(t) - 6\rho r_0^2\sqrt{\pi\nu}\int_{-\infty}^{t}\left(\frac{dv}{dt}\right)_{t=s}\frac{1}{\sqrt{t-s}}ds - \tfrac{1}{2}\rho V_p\frac{dv(t)}{dt} + \rho_p V_p v_0\delta(t). \tag{62}$$

The first three terms on the right-hand side of (62) are the Stokes drag force, the Boussinesq-Basset viscous memory force and the reaction acceleration force (added mass force), respectively. These well-known terms can be derived from the Laplace transform of the unsteady Stokes equations, see [28]. The fourth term, proportional to the Dirac delta function $\delta(t)$, represents the momentum pulse injected into the system at $t = 0$. This system is a special case of the so-called Basset–Boussinesq–Oseen equation.

It is convenient to define

$$\hat{v}_0 = v_0\rho_p/(\rho_p + \tfrac{1}{2}\rho), \tag{63}$$

$$v(t) = \hat{v}_0 H(t) + v'(t), \tag{64}$$

$$F(t) = \rho_p V_p \hat{v}_0\delta(t) + F'(t), \tag{65}$$

where $H(t)$ is the Heaviside function. We derive the following system for $v'(t)$:

$$\rho_p V_p \frac{dv'(t)}{dt} = F'(t), \quad v'(t) = 0 \text{ if } t \le 0, \tag{66}$$

$$F'(t) = -6\pi\rho\nu r_0(\hat{v}_0 + v'(t)) - 6\rho r_0^2\sqrt{\pi\nu}\left(\frac{\hat{v}_0}{\sqrt{t}} + \int_{-\infty}^{t}\left(\frac{dv'}{dt}\right)_{t=s}\frac{1}{\sqrt{t-s}}ds\right) - \tfrac{1}{2}\rho V_p\frac{dv'(t)}{dt}. \tag{67}$$

The function $v'(t)$ is continuous at $t = 0$, such that $v'(t)$ and $F'(t)$ for $t > 0$ can be accurately computed by applying a standard numerical method to the latter system. The implied solutions $v(t)$ and $F(t)$, which follow from (64) and (65), are called $v_{exact}$ and $F_{exact}$.

Since the velocity of the sphere undergoes a step change, the overset grid simulations of this problem are denoted by Step1 and Step2. A velocity step change of a sphere is an interesting and also relevant phenomenon, because in reality collisions create sudden changes of particle velocities, relatively large changes if $\rho_p/\rho$ is large, like in a gas–solid flow. For this reason $\rho_p/\rho = 1000$ is selected in Step1 and Step2. The investigation of a step change is also relevant from a more theoretical point of view; if a particle of finite size is injected in a nontrivial smooth flow at location $\mathbf{x}$, discontinuities are generated at the particle surface, even if the particle is injected with translational and angular velocity of the particle respectively equal to the velocity and half vorticity of the fluid at $\mathbf{x}$ just before injection.

The impulsive acceleration is applied through the initial condition: at $t = 0$ the fluid velocity and pressure field are zero, but the velocity of the sphere is set to $v_0\mathbf{e}_1$ ($v_0 = u_{ref} = 1, \boldsymbol{\omega}^p = \mathbf{0}$). It is essential that the option to make the velocity divergence free after the initialization is not used in this case. The velocity of the sphere is not prescribed for $t > 0$. Since for this case no analytical expression for the full velocity field was found in literature, the domain $\Omega_c$ is large ($L_1 = 20$) and equipped with periodic boundary conditions. The sphere is initially positioned at the center of $\Omega_c$. The extent of the spherical domain size is determined by $r_b \approx 3.562 d_p$, while $r_a = (r_0 + r_b)/2$. The spatial resolution of two simulations is given by $15\kappa \times 24\kappa \times 48\kappa$ and $N_1^3 = (40\kappa)^3$, where $\kappa = 1$ (Step1) or $\kappa = 2$ (Step2). The grid configuration for $\kappa = 1$ is illustrated in Fig. 4(b). The minimum radial grid spacing is approximately $d_p/(15\kappa)$. The radial grid is stretched using the exponential stretching function defined in section 3. The time step equals $\Delta t = 0.0004/\kappa^2$.

Fig. 8 shows the velocity change and the force on the particle due to the impulse input at $t = 0$ as functions of time. The axis scalings are logarithmic to capture the large variation of quantities and to show the large changes in the first time steps clearly. Equation (62) contains three terms that are singular at $t = 0$. The impulse injected into the system at $t = 0$, $\rho_p V_p \delta(t)$, is accounted for by the initial condition of the simulations. The other two singular terms are the history force and the added mass force, which are both negative at $t = 0$. These negative singularities lead to the very large $-F$ in the first

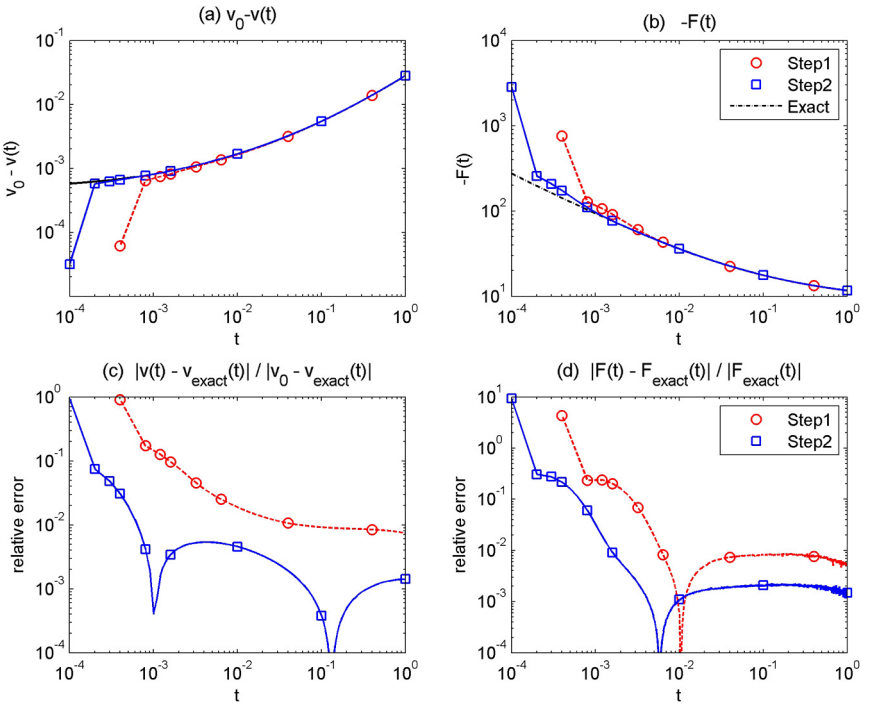


**Fig. 8.** Instantaneously accelerated sphere. (a) The change of the velocity $v_0 - v(t)$ and (b) minus $F(t)$ from simulations Step1 (circles, red dashed) and Step2 (squares, blue solid). In (a) and (b), the so-called exact solution obtained by numerical integration of the Basset–Boussinesq–Oseen equation is also included (dash-dotted). In (c) and (d) the relative error of the deviation from the so-called exact solution is shown for the $v_0 - v(t)$ (c) and $F(t)$ (d). The symbols denote the values after time step 1, 2, 3, 4, 8, 16, 100, 1000 and 10000. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

time step of the simulations, roughly proportional to $1/\Delta t$. The largest contribution of $-F$ in the first time step is due to the added mass singularity since the delta function singularity in the added mass force is stronger than the root singularity in the history force. In both cases, the simulations takes two time steps to adapt $v_0$ to approximately $\hat{v}_0$, which indicates that in the numerical solutions $v(2\Delta t)$ converges to $\hat{v}_0$ if $\Delta t \to 0$. This reflects the theoretical limit $v(t) \to \hat{v}_0$ if $t \downarrow 0$. In the fine grid simulation (Step2), the relative error drops below 1% within 8 time steps for $v_0 - v(t)$ and within 16 time steps ($t = 0.0016$) for $F(t)$. In a test for density ratio $\rho_p/\rho = 1$, a larger error was found (less than 4% in $F(t)$ for $t \geq 0.0016$, for the same grid and same time step as in Step 2), but in that test the error could be lowered by a factor 2 by reducing the time step. It is concluded that the response to a step change of the particle velocity can be captured by the present method.

## 5. Results of Navier–Stokes simulations

In this section, we present results of simulations of three Navier–Stokes flow cases: (1) a moving array of spheres, (2) a sphere falling in channel flow, and (3) eight small spheres freely moving in a Taylor–Green flow. In the last two cases, the stretching option of the spherical grid is used to reduce the number of grid points of the Cartesian background grid with orders of magnitude (compared to uniform Cartesian grid methods). In each case $d_p = 1$, $\nu = 1$ and $\rho = 1$. The acceleration terms **a** and **g** are zero, except in the second case. The Cartesian grid cells are cubes of uniform edge length $h$. Each simulation was run in parallel on eight processors, using MPI, for which the Cartesian domain was divided into $2 \times 2 \times 2$ blocks. The computing times can be found in Appendix C.

Although in the multi-particle cases, cases (1) and (3), the particles do not collide, these cases do add value to the present validation study. The results of the first case demonstrate that the Galilean invariance of the Navier–Stokes equations is numerically respected, which is a non-trivial validation of the implementation of the convective terms in the overset grid approach. The third case is a transient flow with a number of symmetries in the initial condition. The motion of multiple particles, initially symmetrically embedded in the flow, appears to obey the initial symmetry up to high accuracy during the entire simulation. The verification that these expected symmetries are persistent in the simulation confirms that the implementation of geometrically complicated parts of the algorithm, such as the time-dependent motion of interpolation points and interpolation stencils is correct. Furthermore, the present code is the first MPI implementation of a staggered overset grid method that can be used for particle-resolved direct numerical simulations of turbulent flows around fixed or oscillating particles, which is a valid reason to demonstrate that the method is able to simulate flows with multiple non-colliding particles. Of course such flows are theoretical, but they do give insight into particle–fluid interaction [37,23].

**Table 4**
Results of simulations of an FCC array of particles moving with velocity $V_1$. The particle volume fraction equals 0.2 and the Reynolds number is 40.

| $V_1$ | $\kappa$ | $d_p/h$ | $F_1^{p,q}/F_0$ | $F_1^{p,\nu}/F_0$ | $\rho a_1 V_p/F_0$ | $F_1^{p,tot}$ | $F_1^p/F_1^{p,tot}$ |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 19.2 | 2.703 | 4.645 | 1.907 | 9.255 | 0.794 |
| 0 | 2 | 38.4 | 2.800 | 4.671 | 1.880 | 9.351 | 0.799 |
| 0 | 4 | 76.8 | 2.820 | 4.673 | 1.875 | 9.369 | 0.800 |
| $-U_1/(1-\alpha)$ | 1 | 19.2 | 2.702 | 4.641 | 1.910 | 9.252 | 0.794 |
| $-U_1/(1-\alpha)$ | 2 | 38.4 | 2.806 | 4.670 | 1.878 | 9.354 | 0.799 |
| $-U_1/(1-\alpha)$ | 4 | 76.8 | 2.827 | 4.673 | 1.871 | 9.371 | 0.800 |

### 5.1. Moving FCC array of spheres

We consider a face-centered cubic (FCC) particle array with particle volume fraction $\alpha = 0.2$ at finite Reynolds number, like in Refs. [3,23,29,8]. Periodic boundary conditions are applied in each direction and 4 spherical particles are put at locations $(L_1, L_1, L_1)/4$, $(L_1, 3L_1, 3L_1)/4$, $(3L_1, L_1, 3L_1)/4$ and $(3L_1, 3L_1, L_1)/4$. The size of the cubical domain is given by $L_1 = (10\pi/3)^{1/3} \approx 2.19$. The velocities of the particles are constant and equal to $\mathbf{V} = V_1\mathbf{e}_1$ and the forcing term is set to $\mathbf{a} = a_1\mathbf{e}_1$ with

$$a_1(t) = \max(-2000, \min(2000, [U_1 - (1-\alpha)(\langle u_1\rangle_\Omega - V_1)]/\Delta t)), \tag{68}$$

where $U_1 = 40$ is the preset superficial relative velocity. For stability reasons the explicit forcing term depends on $\Delta t$. The difference between the preset and the computed superficial relative velocity decreases with decreasing $\Delta t$. The Reynolds number $\rho d_p U_1/\nu$ is equal to 40.

In the definition of $a_1$, the operator $\langle\cdot\rangle_\Omega$ appears. This represents a volume average over $\Omega$, the domain that excludes the particles. For the approximation of the volume integral over $\Omega$, we define the radius $r_e$ for the evaluation of integrals, which should coincide with a staggered radial position ($r_e = 0.8r_b$ in this subsection). The integration over $\Omega$ is then the sum of: (1) a midpoint integration over all spherical cells within distance $r_e$ of a particle location, (2) a midpoint integration over all Cartesian cells entire outside the spheres with radius $r_e$, and (3) a midpoint integration over tiny Cartesian subcells that cover the parts of $\Omega$ not covered by (1) and (2). The size of the tiny Cartesian subcells is $h/5$. This integration procedure has been described in more detail in Ref. [23].

Simulations have been performed for both fixed particles, $V_1 = 0$, and moving particles, $V_1 = -U_1/(1-\alpha)$, for three different resolutions and for $0 \le t \le 0.3$. The number of Cartesian grid cells is $N_1^3 = (42\kappa)^3$, where the grid refinement factor $\kappa$ equals 1, 2 or 4. The corresponding resolutions of the spherical grids are $5\kappa \times 24\kappa \times 48\kappa$. In each case, the radial grid spacing $\Delta r$ is uniform and equal to $h$, while $u_{ref} = U_1$ and $\Delta t = 0.000025$. The grid structure for $\kappa = 1$ is the same as in Fig. 4(a), except that $h$ is approximately 1.3 times smaller than in Fig. 4(a). The time step is given the same small value in each case to ensure that the difference between preset and computed superficial relative velocity is negligible in each case. Results of the simulations, the three contributions to the particle force, $F_1^{p,q}$, $F_1^{p,\nu}$ and $\rho a_1 V_p$ at $t = 0.3$, are also shown in Table 4 (after normalization by $F_0 = 3\pi d_p \nu U_1$). The third contribution is due to the spatially constant pressure gradient $-\rho a_1$. Moreover, the total particle force $F_1^{p,tot} = F_1^{p,q} + F_1^{p,\nu} + \rho a_1 V_p$ is included. The forces shown in the table are mean values, averaged over the particles and over time between $t = 0.25$ and $t = 0.3$. The corresponding standard deviations were verified to be small: for the fixed particles they were zero, while for the moving particles the relative standard deviations of $F_1^{p,q}$ were 1.2% for the coarse, 0.2% for the finer and 0.03% for the finest grid (those of $F_1^{p,\nu}$ were an order of magnitude smaller).

Table 4 demonstrates clear convergence upon grid refinement for both the fixed and moving configurations. The differences between the results of the fixed and moving configurations are negligible, which confirms that the implementation mimics the Galilean invariance of the Navier–Stokes equations. A perfect momentum balance implies that the mean of $(1-\alpha)a_1$ is equal to the mean of $N_p F_1^p$. Thus the last number in the table, the mean of $F_1^p$ divided by the mean of $F_1^{tot}$, should converge to $1-\alpha = 0.8$, and it does. For fixed particles, the same test case was considered in Refs. [3] and [23]. It is remarked that the force contributions $F^{p,q}$ and $F^{p,\nu}$ mentioned in Ref. [23] were larger because those were normalized by $(1-\alpha)F_0$. As a check, the values from Table 4 were divided by $(1-\alpha)$ and compared to those in [23]. Small discrepancies (<1.5%) were observed. Their source was traced, and a small error in the treatment of interpolation points outside $\Omega_c$ was found in the code used for the simulations in [23]. This error affected only the FCC test case and no other simulation presented in [23].

### 5.2. Falling sphere in channel flow

In this subsection we consider a falling sphere in a laminar channel flow. The test case was introduced by Uhlmann [2], while the initial condition was slightly modified by Breugem [5]. The domain $\Omega_c$ is given by $L_3 = H = 20d_p$ and $L_1 = L_2 = H/2$. There is a flat solid no-slip wall at $x_3 = 0$ and a free slip wall at $x_3 = H$. Periodic boundary conditions are used in the $x_1$ and $x_2$ directions. The initial velocity of the fluid is given by the parabolic profile $\mathbf{u} = U_b(x_3/H)(2 - x_3/H)\mathbf{e}_2$, where $U_b$ is the specified bulk velocity. A single sphere of diameter $d_p = 1$ is initially located at $\mathbf{x}^p = (L_1, L_2, L_3)/2$. The translational

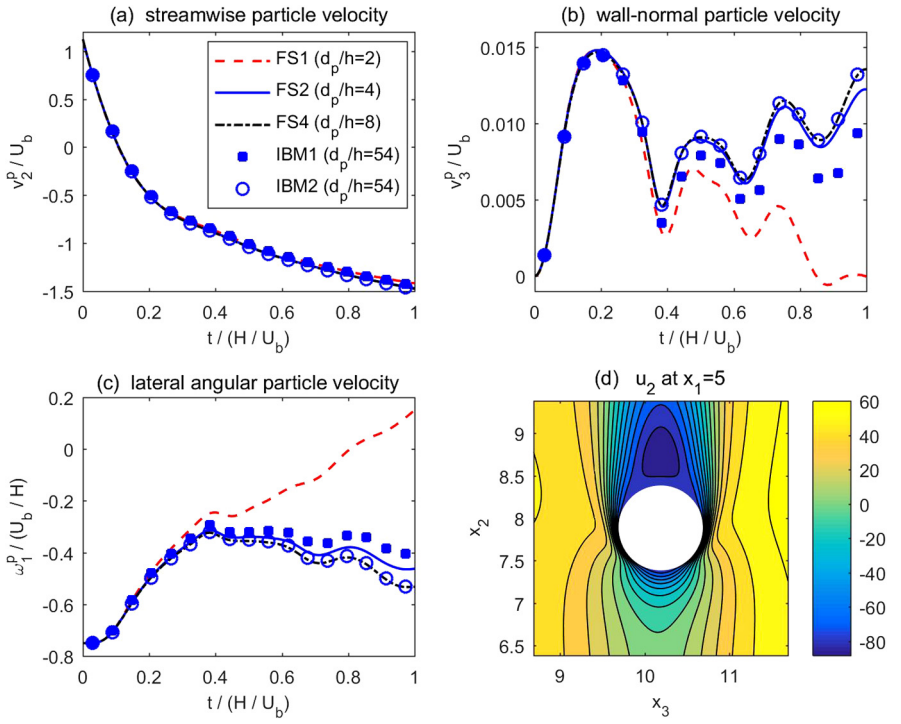


**Fig. 9.** Falling sphere in laminar channel flow. Streamwise particle velocity $v_2^p$ (a), wall-normal particle velocity $v_3^p$ (b) and lateral angular velocity $\omega_1^p$ (c), for three consecutively refined grids. Simulations FS1 ($d_p/h = 2$, red dashed), FS2 ($d_p/h = 4$, blue solid) and FS4 ($d_p/h = 8$, black dash-dotted). For comparison results of two high-resolution IBM simulations performed and presented by Breugem [5] are included: IBM1 (filled squares) and IBM2 (open circles). Contours of streamwise (vertical) velocity from FS4 in the plane $x_1 = 5$ at $t = 1$ are shown in (d). The contour increment is 10. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and rotational initial velocities are given by $\mathbf{v}^p = 1.125U_b\mathbf{e}_2$ and $\boldsymbol{\omega}_1^p = -0.75(U_b/H)\mathbf{e}_1$. In addition, $\mathbf{g} = -1.1036(U_b^2/d_p)\mathbf{e}_2$ and $\rho_p/\rho = 4.17$, while $\mathbf{a} = a_2\mathbf{e}_2$ is determined such that $\langle u_2\rangle_\Omega$ remains constant in time. The bulk velocity $U_b$ is set to 50, such that $U_bH/\nu$ is equal to 1000.

The overset-grid simulations that have been performed are called FS1, FS2 and FS4. The number in each name denotes the grid refinement factor $\kappa$, again equal to 1, 2 or 4. The spherical grid contains $15\kappa \times 24\kappa \times 48\kappa$ cells and the Cartesian background grid $20\kappa \times 20\kappa \times 40\kappa$ cells, while the time step is equal to $0.001(H/U_b)/\kappa^2$ and $u_{ref} = U_b$. The Cartesian grid contains $d_p/h = 2\kappa$ grid points per particle diameter. The minimum radial grid spacing is approximately $d_p/(15\kappa)$. The grid structure is the same as in subsection 4.4 and is illustrated in Fig. 4(b). The radius $r_e$ for the evaluation of integrals (introduced in the previous subsection) is approximately $3.125d_p$ in this case.

The evolution of the streamwise and wall normal particle velocity and the lateral rotational particle velocity are shown in Fig. 9. A clear dependency on resolution is observed, in particular for the streamwise and wall normal velocity. The fact that the differences between FS1 and FS2 are much smaller than the differences between FS2 and FS4 suggests that the grid refinement sequence is convergent. The curves of the streamwise velocity from FS2 and FS4 fall one upon the other. At the highest resolution ($\kappa = 4$), the total number of grid cells is approximately 1.2 million and the total number of time steps 16000.

Furthermore, Fig. 9 shows results of two immersed boundary simulations IBM1 and IBM2 performed by Breugem, both for $d_p/h = 54$ (see Figs. 13–15 in [5]). IBM1 is the immersed boundary method as originally developed by Uhlmann [2] and has been used in impressive resolved simulations with many particles, see for example [34]. Breugem [5] showed that the first-order accurate IBM1 can be made second-order accurate if the fluid-solid interface is retracted. Thus, he formulated IBM2, in which the rectraction distance is $0.3h$. Other improvements of IBM2 over IBM1 are that IBM2 uses the technique of multidirect forcing and, like [4], takes the fluid inertia inside the particle into account.

Fig. 9 shows a remarkably good agreement between IBM2 and the overset grid simulation FS4. The results of these two simulations are apparently more accurate than those of IBM1. On coarse grids (for example $d_p/h \approx 16$), IBM2 produced small but clearly discernible rapid oscillations on the curve of $\omega_1$, which were attributed to grid locking. Such oscillations are not observed in the results of the overset grid simulations, also not in those of FS1 (see Fig. 9(c)).

The total number of grid cells in FS4 is approximately 260 times smaller than in IBM1 and IBM2, which used a grid of 314.9 million cells. Also timewise FS4 is more efficient; the simulation required approximately 20 times less CPU time than IBM2 on the same hardware (see Appendix C). It is remarked that the immersed boundary method is more suitable for dense flows, whereas the present overset grid method is more suitable for dilute flows (with non-colliding particles).

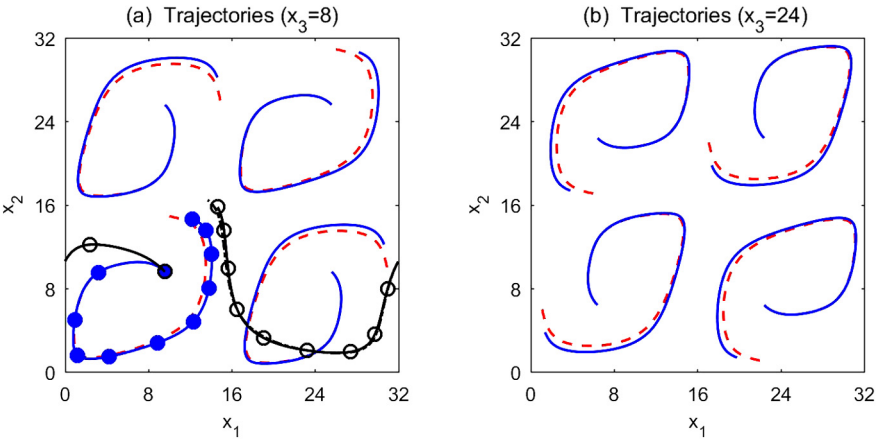


**Fig. 10.** Trajectories of the particles injected into Taylor Green flow in the planes $x_3 = 8$ (b) and $x_3 = 24$ (b) for $0 \le t \le 5$. The solid line with filled circles and the solid lines without markers denote the trajectories from the fine grid simulation TG2 ($\rho_p/\rho = 1$). A trajectory from simulation TG2a ($\rho_p/\rho = 8$) is also included (open circles). In addition trajectories from the coarse grid simulations TG1a (red dashed) and TG2a (black dashed) are shown. The trajectories start at the ends where solid and dashed lines coincide. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

### 5.3. Moving spheres in Taylor–Green flow

The third example of a Navier–Stokes flow is the three-dimensional Taylor–Green flow that evolves from the following initial condition for the fluid velocity in Cartesian coordinates:

$$\hat{u}_1 = \ U \sin(2\pi x_1/L_1)\cos(2\pi x_2/L_1)\sin(2\pi x_3/L_1), \tag{69}$$

$$\hat{u}_2 = -U\cos(2\pi x_1/L_1)\sin(2\pi x_2/L_1)\sin(2\pi x_3/L_1), \tag{70}$$

$$\hat{u}_3 = 0. \tag{71}$$

The flow is interesting because this very simple initial condition evolves into a complicated flow in which small scales are created by vortex stretching, like in turbulent flows [35]. The flow is periodic in each direction. The initial condition contains many symmetries and, provided the Reynolds number is moderate, the symmetries are probably stable under small perturbations. The domain $\Omega_c$ is cubical with $L_1 = 32$, while $U = u_{ref} = 40$ and $\nu = 1$, such that the initial Reynolds number is 1280. Schneiders et al. [36] performed a resolved simulation of small particles in a three-dimensional compressible Taylor Green flow (at low Mach number, 0.1). They showed an illustrative snapshot of the vorticity, but no quantitatively verifiable result of a flow variable.

Eight small spheres of diameter $d_p = 1$ are injected at $t = 0$, four in the plane $x_3 = 8$ and four in the plane $x_3 = 24$. The initial particle velocity is obtained by substituting the initial particle positions into (69) to (71). The initial angular velocities of the particles are set to $\boldsymbol{\zeta}/2$ at the particle positions ($\boldsymbol{\zeta} = \boldsymbol{\nabla} \times \hat{\mathbf{u}}$). We consider two cases: density ratio $\rho_p/\rho = 1$ (TG1 and TG2) and $\rho_p/\rho = 8$ (TG1a and TG2a, no gravity). Simulations have been performed using spherical grids of $30\kappa \times 24\kappa \times 48\kappa$ cells overset on a Cartesian grid of $(64\kappa)^2$ cells, for $\kappa = 1$ (TG1 and TG1a) and $\kappa = 2$ (TG2 and TG2a). The radial stretching function, $r_a$, $r_b$ and $r_e$ are the same as in the previous subsection, and again $d_p/h = 2\kappa$ (see the illustration in Fig. 4(b)).

The initial vorticity field in the planes $x_3 = 8$ and $x_3 = 24$ is given by $\zeta_1 = \zeta_2 = 0$ and

$$\zeta_3 = \frac{2\pi U}{L_1}\sin(2\pi x_1/L_1)\sin(2\pi x_2/L_1)\sin(2\pi x_3/L_1). \tag{72}$$

Thus in the plane $x_3 = 8$ there are four planar vortices, viewed from above, anti-clockwise rotating vortices centered at $(8, 8)$ and $(24, 24)$ and clockwise rotating vortices at $(24, 8)$ and $(8, 24)$. Initially, a particle is placed in each of these vortices, slightly off-center, namely at the locations $(9.6, 9.6)$, $(25.6, 9.6)$, $(9.6, 25.6)$ and $(25.6, 25.6)$. Respecting the point symmetry around $(16, 16, 16)$, the initial particle locations in the plane $x_3 = 24$ are given by $(22.4, 22.4)$, $(6.4, 22.4)$, $(22.4, 6.4)$, $(6.4, 6.4)$.

The locations of the particles and the structure of the velocity field is clarified by the simulated particle trajectories, shown in Fig. 10. For example, the trajectory of the first particle in TG2 is indicated by open circles. The diameter of the circles indicates the particle size on scale. When time evolves each particle moves to the outer region of the corresponding vortex. Each pair of neighbouring circles on the same curve corresponds to a time interval of 0.5. The simulations were terminated at $t = 5$. Due to the structure of the velocity ($u_3$ remains zero in these two planes), the $x_3$ coordinate of the particles does not change in time. Comparing the fine grid to the coarse grid simulations, we observe that the trajectories are similar, but not the same. Apparently, the effect of discretization errors gradually increases and becomes significant at later times in case TG1. For clarity, only the trajectory of the first particle is shown for $\rho_p/\rho = 8$, one for TG1a and one for

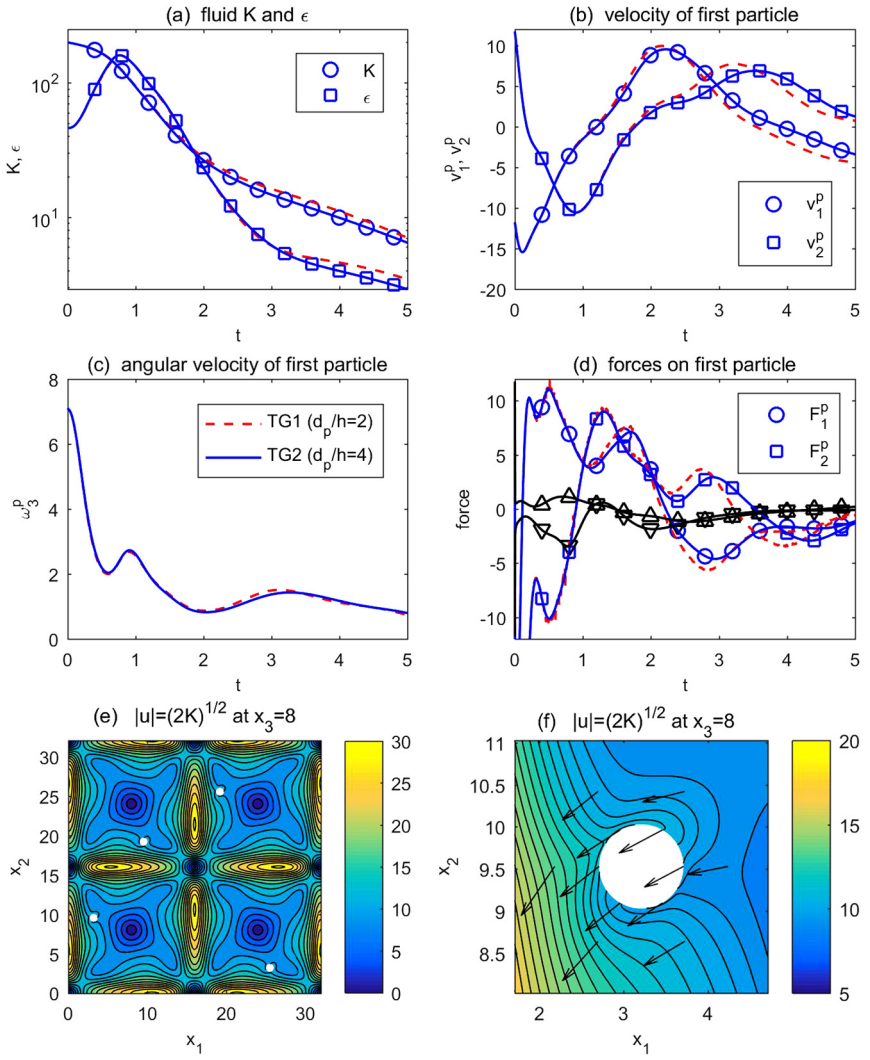


**Fig. 11.** Eight particles in Taylor–Green flow with $\rho_p/\rho = 1$, simulations TG1 (dashed) and TG2 (solid). (a) Turbulence kinetic energy $K$ (circles) and dissipation rate $\epsilon$ (squares). For the first particle (injected at $(9.6, 9.6, 8)$) the figure shows: (b) the velocity components $v_1^p$ (circles) and $v_2^p$ (squares); (c) the angular velocity component $\omega_3^p$; (d) the particle forces $F_1^p$ (circles) and $F_2^p$ (squares) and their viscous parts $F_1^{p,\nu}$ (upward point triangles) and $F_2^{p,\nu}$ (downward pointing triangles). (e, f) Contours of the magnitude of the velocity from TG2 at $t = 0.5$ in the plane $x_3 = 8$. The contour increment equals 2 in (e) and 0.5 in (f), which also shows 12 velocity vectors (6 at $r = 0.5$ and 6 at $r = 1$).

TG2a. The initial particle positions are the same as for $\rho_p/\rho = 1$ and also in these cases the symmetries are preserved. For $\rho_p/\rho = 1$ particles remain in the vortex where they were injected. However, for $\rho_p/\rho = 8$ the particles respond to the fluid motion more slowly: when the particle shown exits the periodic domain at $x_1 = 0$ and reenters at $x_1 = 32$, it leaves the original vortex and moves over to a vortex rotating in opposite direction.

Further details of the simulations for $\rho_p/\rho = 1$ are shown in Fig. 11. In Fig. 11(a), the total kinetic energy $K = \frac{1}{2}\langle|\mathbf{u}|^2\rangle_\Omega$ and the total energy dissipation rate $\epsilon = \nu\langle\nabla\mathbf{u} : \nabla\mathbf{u}\rangle_\Omega$ are shown as functions of time. The kinetic energy decays, but the dissipation rate first increases, due to the generation of smaller scales, until it attains its maximum around $t \approx 1$, and then it decreases. The differences between the coarse and fine grid simulations remain small until $t \approx 1.5$. At later times most differences gradually increase and can be clearly observed. Fig. 11(c) shows that the angular velocity is surprisingly accurate, compared to the velocity. The particle force components of the first particle are shown in Fig. 11(d) and the corresponding velocity components in Fig. 11(b). While the particle velocity looks very smooth for both resolutions, its time derivative (the particle force) displays small rapid oscillations in the coarse grid case TG1, which is another indication that TG1 is somewhat underresolved at later times. Furthermore, the viscous contribution to the particle force is shown in Fig. 11(d). It appears to be much smaller than the total force, which implies that the particle force is dominated by the pressure contribution if $\rho_p/\rho = 1$.

The motion of the eight particles follows the theoretically expected symmetries very accurately. The particles at the left bottom quarter of Fig. 11(a) and 11(b) are particle 1 and 5, respectively. Furthermore, in both figures, the numbering is

anti-clockwise. The theoretically expected symmetries are $v_1^1 = v_2^2$, $v_2^1 = v_1^2$, $v_3^1 = v_3^2 = 0$, $\omega_1^1 = \omega_2^2 = 0$, $\omega_3^1 = -\omega_3^2$, $\mathbf{v}^1 = \mathbf{v}^3 = -\mathbf{v}^6 = -\mathbf{v}^8$, $\mathbf{v}^2 = \mathbf{v}^4 = -\mathbf{v}^5 = -\mathbf{v}^7$, $\boldsymbol{\omega}^1 = \boldsymbol{\omega}^3 = \boldsymbol{\omega}^6 = \boldsymbol{\omega}^8$, and $\boldsymbol{\omega}^2 = \boldsymbol{\omega}^4 = \boldsymbol{\omega}^5 = \boldsymbol{\omega}^7$ (the superscripts denote the particle numbers). All these equations have been explicitly checked for case TG2; during the entire simulation, all equations were satisfied with errors less than $10^{-4}$.

## 6. Conclusions

A staggered overset grid method for resolved simulation of incompressible flows around moving spheres has been presented. In this method, body-fitted spherical polar grids attached to the moving spheres are used to resolve the flow in the vicinity of the spheres. An accurate representation of the sharp particle–fluid interface is thus provided. The spherical grids are overset on a fixed Cartesian background grid. The Navier–Stokes equations in spherical coordinates and in Cartesian coordinates are respectively approximated on the spherical and Cartesian grids, using the standard staggered second-order finite difference scheme. The velocities and pressures on different grids are coupled by third-order Lagrange interpolations. A comprehensive description of the method has been provided, including the description of the data structure and communication algorithms required for an implementation in the form of an MPI parallel program.

The solver has been validated for seven different cases, four Stokes flows and three Navier–Stokes flows. The first three Stokes flows were the flows around a translating sphere, oscillating sphere and a rotating sphere. The analytical solutions for these flows are known, and the results obtained indicated pointwise convergence of the numerical to the analytical solutions. More specifically, it was found that the spatial maximum norm of the error in both velocity and pressure dropped below 1% of the reference values if the grid contains 30 points per particle diameter. The fourth test case was an instantaneously accelerated sphere. The effects of the step change of the velocity of the sphere were compared with those predicted by the Basset–Boussinesq–Oseen equation of particle motion. Good agreement was obtained within several time steps.

The first Navier–Stokes case was the flow due to a translating FCC array of spherical particles at particle volume fraction 0.2. The grid refinement study displayed a clear convergence of both the pressure and the viscous part of the drag force. Second, a sphere falling in a laminar channel flow was simulated using grid refinement. The results were compared to results obtained by two immersed boundary methods, IBM1 (first-order) and IBM2 (second-order, due to retraction of the particle surface), which were published in literature [5]. The results of the overset grid simulation at the highest resolution and those of the IBM2 simulation at the highest resolution displayed a very good agreement. In this case, the radial stretching facility of the overset grid method was exploited, such that the overset simulation at the highest resolution used approximately 260 times less grid points and 20 times less computing time than the IBM2 simulation at the highest resolution. The radial stretching facility was also used in the third Navier–Stokes case, incompressible three-dimensional Taylor–Green flow with 8 freely moving particles. The effect of grid refinement was shown for this flow, as for the other flows in this paper.

In the present method, the spherical grids do not overlap each other. Thus particle–particle contact cannot be simulated. Simulations in which particles closely approach each other at random locations require a fine Cartesian background grid (the minimum number of grid points across the gap between particles needs to be at least ten). Therefore, the present method is intended for relatively theoretical studies, for parallel simulations of incompressible flows with a low volume fraction of non-colliding solid spherical particles, flows that are described by an initial boundary value problem of the Navier–Stokes equations that is not too complicated for a direct numerical simulation in which the Navier–Stokes equations are accurately solved without simplification of the equations in any part of the fluid domain. In the opinion of the author, particle-resolved direct numerical simulation of cases with multiple non-colliding particles are useful for insight into particle–fluid interaction and for validation of point-particle models in turbulent flows, even if collisions are prevented by fixing the particles or limiting their motion in another way, see for example [37] and [23]. Furthermore, the overset grid method can be used to validate and benchmark other methods for particle-resolved simulation (for example, in the case of the falling sphere, not only the overset grid method, but also the immersed boundary methods IBM1 and IBM2 were tested).

To increase the flexibility and applicability of the present method, there are several possibilities. The first option is to allow mutual overlap of spherical grids. (In fact, the routines for interpolating from one spherical grid to another and for cutting out parts of spherical grids have been implemented in the solver, but apart from the observation that these routines work robustly, they have not been validated yet.) A second option is to construct an (implicit) method in which the radial extent of the spherical grids is kept as small as possible and the Cartesian background grid is nonuniform and adaptive. One could also think of a fine spherical grid overset on a local fine Cartesian grid, which is on its turn overset on the background Cartesian grid. For the narrow gaps between particles one could attempt to devise a deformable (perhaps non-orthogonal) grid overset on mutually overlapping spherical grids.

## Acknowledgements

The author is grateful to J.G.M. Kuerten for many discussions and for repeating simulation FS4 on another computer. With respect to the results of the IBM simulations shown in Fig. 11, W.P. Breugem kindly provided the source data and repeated (part of) a simulation to measure the computing time.

## Appendix A. Parallel implementation

A common parallel algorithm for particle-resolved simulation is the so-called master and slave technique developed by Uhlmann [38]. In that technique the block structure is two-dimensional, while each time when a particle center moves to another block, the fluid variables associated to a particle are transferred to the processor of the new block (the particle gets another master). The present method differs from the master and slave technique in two important respects: (1) in order to minimize the total area of the contact surfaces of the blocks and the related MPI communication, the present method employs a three-dimensional instead of a two-dimensional block structure, and (2) in the present method, the center of the corresponding moving particle does not necessarily reside inside the Cartesian block on that processor, because a given spherical field stored on a certain processor remains on this processor during the entire simulation. The latter is done to avoid MPI communication of entire three-dimensional spherical fields.

As explained in section 3.1, each of the $M$ processors contains the Cartesian fluid field of one Cartesian block and the spherical fluid field of maximum $N_{p1}$ particles ($N_p/M \leq N_{p1} < N_p/M+1$). Thus, if $N_p = MN_{p1}$, each processor is associated with $N_{p1}+1$ grids, namely of the Cartesian block grid and the grids of $N_{p1}$ particles. The acceptor lists and the primary and secondary donor lists, which include all interpolation weight coefficients, are accordingly distributed over the processors. Furthermore, the hydrodynamic forces exerted on the particles are distributed over the processors. However, all reference position vectors, all particle translational and angular velocities, the pointer array relating shadow to physical particles and the block lists containing particle numbers are stored on each processor. This is not strictly necessary for the algorithm, but it is convenient and not expected to produce significant overhead or affect the scalability of the solver, provided the number of particles is not too large (say less than $10^4$). Under this condition, the resources required are almost entirely determined by the arrays for the fluid motion on the Cartesian and spherical grids, and these arrays are, for a given particle or a given block, stored on one processor only.

With respect to the actions in each time step in subsection 3.6, MPI communication is required in substeps 5, 8, 9, 11, 13, 15 and 16. In substep 5, after the particle translational and angular velocities have been updated on a given processor for the $N_{p1}$ particles associated to the processor, these quantities are broadcasted to all other processors. In substep 16, the post processing substep, MPI communication is required for the computation of extrema and integrals over the entire flow domain. The MPI communication in the other substeps is related to either setting up the interpolation structure, performing the interpolations, or determining the fluid variables at boundary points.

Let us consider the determination of the interpolation structure on a given processor $m$. Since the Cartesian type arrays are equipped with multiple dummy layers, no MPI communication is required to fill the type arrays. Also, the construction of the acceptor lists does not require MPI communication, but the construction of the donor lists does. Processor $m$, being a future acceptor, prepares an a priori unknown number of fixed-length packages for distribution to other processors. A given package contains a package identification number, the number of acceptor interpolation points represented by the package, the processor number $m_b = m$, the acceptor grid number $b$, the donor grid number $a$, and, for all interpolation points represented by the package, the grid acceptor grid indices $(i, j, k)$. All interpolations points in a given package have the same donor grid $a$. Thus the package is prepared to be sent to processor $m_a$ (which could be equal to $m$). Processor $m$ may prepare multiple packages for given processor $m_a$. Because of the fixed size of the packages, the last package in a row of packages prepared for $m_a$ may be partially empty. After the preparation of a row of packages, processor $m$ uses MPI communication to inform each processor $m_a$ how many packages it can expect from processor $m$. At the same time processor $m$ is informed by other processors how many packages it can expect from them. Afterwards processor $m$, being a future acceptor, uses MPI to send all packages in the row sequentially to each processor $m_a$. At the same time, processor $m$, being also a future donor, may receive packages from other processors. All received packages are copied into the primary donor lists.

After processor $m$ has filled its primary donor lists, it collects additional information for the interpolation points in these lists. Going through its primary donor lists, it performs the following actions for each interpolation point in the lists. First, being a future donor, it recomputes the acceptor interpolation coordinates of a for each acceptor interpolation point $x_b$, using the acceptor grid number and the grid indices $(i, j, k)$ of the acceptor interpolation point. Subsequently, the indices of the reference corner of the corresponding interpolation stencil (which resides on processor $m$) and the interpolation weights are determined and stored in the secondary donor lists.

The interpolations themselves are performed in the following way. We remind that the donor lists contain the package identification numbers that were sent by the acceptor processors when the interpolation lists were set up. According to these identification numbers floating point packages of fixed size are prepared. Such a package contains the package identification number (converted to floating point format) and the corresponding interpolated values. MPI is used to send the package to the acceptor processor. After the MPI communication, the contents of the received packages is copied to the corresponding interpolation points, whose location is received from the acceptor lists. Before and after the interpolations are performed, the boundary conditions routines are called. The latter routines require MPI communication to copy the fluid variables at Cartesian dummy points from other blocks.

## Appendix B. Interpolation formulas

In this appendix the formulas of the third-order interpolations are specified, which are based on quadratic Lagrange polynomials. For this purpose, the fluid variables are written as functions of grid indices. For example $q(i,j,k)$ denotes $q$ at the Cartesian pressure grid point with indices $i$, $j$ and $k$, while $\tilde{q}(i,j,k)$ denotes $q$ at the spherical pressure grid point with indices $i$, $j$ and $k$. The summation convention is not used in this subsection.

Let us consider the pressure at $\mathbf{x}$, where $\mathbf{x}$ is an interpolation point $(i'', j'', k'')$ on a Cartesian grid and is connected to an interpolation stencil with reference corner point $(i', j', k')$ on a spherical grid. The coordinate transformation specified in (1) provides $(r,\theta,\phi)$ that correspond to $\mathbf{x}$. We define $W_i^r$ as the three weight coefficients of one-dimensional third-order Lagrange interpolation at query point $r$ using sample points $i'$, $i'+1$, and $i'+2$. Analogously, $W_j^\theta$ and $W_k^\phi$ are defined. Consecutive application of the three one-dimensional interpolations leads to the 27 weight coefficients

$$w_{ijk} = W_i^r W_j^\theta W_k^\phi \tag{73}$$

and the following interpolation formula

$$q(i'', j'', k'') = \sum_{i=0}^{2}\sum_{j=0}^{2}\sum_{k=0}^{2} w_{ijk}\tilde{q}(i'+i, j'+j, k'+k). \tag{74}$$

The formula for interpolation of the pressure from Cartesian to spherical grids is analogous. For each pressure interpolation point on the spherical grid, the weight coefficients are implied by three consecutively applied one-dimensional third-order Lagrange interpolations on a Cartesian interpolation stencil.

The interpolations of the velocity components are more complicated, but they are all based on equation (4). Suppose that $(i'', j'', k'')$ are the indices of a Cartesian interpolation point of $u_m$. The interpolation of $A_{1m}\tilde{u}_1$ is performed using three consecutively applied one dimensional Lagrange interpolations using an interpolation stencil consisting of $3\times3\times3$ staggered $\tilde{u}_1$ points. The corresponding weights are denoted by $w_{m1ijk}$. The corresponding reference corner point on the spherical $\tilde{u}_1$ grid is denoted by $(i'_{m1}, j'_{m1}, k'_{m1})$. The interpolations of $A_{2m}\tilde{u}_2$ and $A_{3m}\tilde{u}_3$ are performed on stencils of staggered $\tilde{u}_2$ points and $\tilde{u}_3$ points, respectively, and the corresponding weights are $w_{m2ijk}$ and $w_{m3ijk}$, while the corresponding reference corner points are $(i'_{m2}, j'_{m2}, k'_{m2})$ and $(i'_{m3}, j'_{m3}, k'_{m3})$. The interpolation formula for Cartesian velocity component $u_m$ is then given by

$$u_m(i'', j'', k'') = v_m^p + \sum_{l=1}^{3}\sum_{i=0}^{2}\sum_{j=0}^{2}\sum_{k=0}^{2} w'_{mlijk}\tilde{u}_l(i'_{ml}+i, j'_{ml}+j, k'_{ml}+k), \tag{75}$$

where

$$w'_{mlijk} = w_{mlijk}A_{lm}(i'_{ml}+i, j'_{ml}+j, k'_{ml}+k). \tag{76}$$

For the interpolation of a spherical velocity component $\tilde{u}_m$ at spherical interpolation point $(i'', j'', k'')$ and associated interpolation stencils on staggered Cartesian grids with reference corner points $(i'_{ml}, j'_{ml}k'_{ml})$ and corresponding interpolation weights $w_{mlijk}$, the following formula is used

$$\tilde{u}_m(i'', j'', k'') = -\sum_{l=1}^{3} A_{ml}(i'', j'', k'')v_l^p + \sum_{l=1}^{3}\sum_{i=0}^{2}\sum_{j=0}^{2}\sum_{k=0}^{2} w'_{mlijk}u_l(i'_{ml}+i, j'_{ml}+j, k'_{ml}+k), \tag{77}$$

where

$$w'_{mlijk} = A_{ml}(i'', j'', k'')w_{mlijk}. \tag{78}$$

Thus in the interpolation from Cartesian to spherical grids (77)–(78), the transformation matrix is effectively applied to interpolated spherical velocities, whereas in the interpolation from spherical to Cartesian grids (75)–(76), the interpolation is effectively applied to transformed velocities. In this manner, the transformation matrix $\mathbf{A}$ is required on the spherical grids only, such that the available versions of this matrix (initially computed versions for the different staggered locations on the spherical grid) can be used. Thus for each interpolation of a velocity component, 81 weight coefficients are computed and stored in the secondary interpolation lists. Also, the first term on the right-hand side of (77) is stored in the secondary interpolation lists.

## Appendix C. Computing times

The computing times required for the overset grid simulations presented in sections 4 and 5 are summarized in Table 5. The CPU times are the wall clock times multiplied by the number of processors (cores, threads), which was 8 in each case. An Intel Xeon 2680v2 machine, equipped with OpenMPI and the gfortran compiler, was used.

**Table 5**
CPU times of Stokes test cases (column 2) and Navier–Stokes test cases (column 4).

| case | CPU time [hours] | case | CPU time [hours] |
|---|---|---|---|
| Tran1 | 0.5 | FCC1 | 2.7 |
| Tran2 | 16.9 | FCC2 | 19.3 |
| Osc1 | 0.7 | FCC4 | 191 |
| Osc2 | 14.1 | FS1 | 0.3 |
| Rot1 | 0.1 | FS2 | 5.5 |
| Rot2 | 0.3 | FS4 | 150 |
| Step1 | 0.5 | TG1 | 3.8 |
| Step2 | 8.5 | TG2 | 110 |

**Table 6**
Uniform flow past a fixed sphere. Maximum and $L_2$ norms of errors at $t = 1$. The numbers in brackets are estimates of the order of accuracy based on $\kappa - 1$ and $\kappa$. The pressure $q$ was normalized by $q_{max} = 3$.

| $\kappa$ | $d_p/h$ | $\|\mathbf{u} - \mathbf{u}_{exact}\|_\infty$ | $\|q - q_{exact}\|_\infty$ | $\|\mathbf{u} - \mathbf{u}_{exact}\|_2$ | $\|q - q_{exact}\|_2$ |
|---|---|---|---|---|---|
| 1 | 15 | 0.001862 | 0.00648 | 0.0001732 | 0.000647 |
| 2 | 30 | 0.000387 (2.27) | 0.00325 (1.00) | 0.0000356 (2.28) | 0.000318 (1.03) |
| 4 | 60 | 0.000122 (1.66) | 0.00171 (0.93) | 0.0000088 (2.02) | 0.000122 (1.38) |

A small test of scalability of the parallel implementation up to 8 processors was performed by repeating the first 100 time steps of case TG2, on $M = 1$, 2, 4 and 8 processors. The scaling efficiency for $M$ processors is defined by the wall clock time for $M = 1$, divided by the product of $M$ and the wall clock time for $M$ processors. The scaling efficiencies in this test were excellent since they appeared to be larger than one: 1.17, 1.30 and 1.11 for $M = 2$, 4 and 8.

Furthermore, the computing times of the overset grid simulation FS4 and the immersed boundary simulation using method IBM2 (see section 5.2) were compared. The results of these simulations displayed the same accuracy (see section 5.2). In order to perform this comparison for the same computer architecture, (parts of) these simulations were repeated on Intel Xeon 2690v3 nodes equipped with an Intel compiler (ifort). Since due to the 260 times larger grid, IBM2 required much more memory than FS4, more than 8 (namely 30) processors were used for IBM2. It was found that the CPU time was $8 * 13.0 = 104$ hours for FS4 (J.G.M. Kuerten, private communication) and $30 * 74.7 = 2241$ hours for IBM2 (W.P. Breugem, private communication). It is fair to scale the ratio of the computing times with the ratio $\Delta t/\Delta t_{visc}$. This ratio is 0.36 for FS4 and 0.33 for IBM2 (For IBM2, $\Delta t = (H/U_b)/17000$, while $\Delta t_{visc}$ is the theoretical maximum viscous time step for a 3-stage Runge–Kutta method). Thus, it is concluded that, for this dilute flow, the overset grid method is approximately $(2240/104) * (0.33/0.36) \approx 20$ times faster than the immersed boundary method.

## Appendix D. Order of accuracy (steady Stokes flow)

To verify that the first-order behaviour of the pressure was not necessarily due to unsteady effects, a convergence analysis was performed for the steady state variant of the translating sphere case: uniform Stokes flow past a fixed sphere, for $d_p = 1$, $\nu = 1$ and free stream velocity equal to $-\mathbf{e}_1$. The analytical solution was prescribed at $t = 0$ and on the outer boundaries of the Cartesian domain for all times ($0 \le t \le 1$). The domain sizes were the same as in case Tran1. The spherical and Cartesian grids contain $5\kappa \times 24\kappa \times 48\kappa$ and $(60\kappa)^3$ cells, and three simulations were performed ($\kappa = 1$, 2 and 4). In Table 6, the maximum and $L_2$ norms are shown for both the velocity and the pressure. The estimated order of accuracy based on $\kappa - 1$ and $\kappa$, denoted by the number in brackets, is defined by $^2\log(E_{\kappa-1}/E_\kappa)$, where $E_\kappa$ is the error in the quantity under consideration. Both the maximum and $L_2$ norm were based on all internal points. The $L_2$ norms were normalized by the total volume of all internal cells.

## References


[1] G. Tryggvason, R. Scardovelli, Direct Numerical Simulations of Gas–Liquid Multiphase Flows, Cambridge University Press, Cambridge, 2011.
[2] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys. 209 (2005) 448–476.
[3] S. Tenneti, R. Garg, S. Subramaniam, Drag law for monodisperse gas–solid systems using particle-resolved direct numerical simulation of flow past fixed assemblies of spheres, Int. J. Multiph. Flow 37 (2011) 1072–1092.
[4] T. Kempe, J. Fröhlich, An improved immersed boundary method with direct forcing for the simulation of particle laden flows, J. Comput. Phys. 231 (2012) 3663–3684.
[5] W.P. Breugem, A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows, J. Comput. Phys. 231 (2012) 4469–4498.
[6] N.G. Deen, S.H.L. Kriebitzsch, M.A. van der Hoef, J.A.M. Kuipers, Direct numerical simulation of flow and heat transfer in dense fluid–particle systems, Chem. Eng. Sci. 81 (2012) 329–344.
[7] A. Mark, B.G.M. van Wachem, Derivation and validation of a novel implicit second-order accurate immersed boundary method, J. Comput. Phys. 227 (2008) 6660–6680.
[8] R. Hill, D.L. Koch, A.J.C. Ladd, Moderate-Reynolds-number flows in ordered and random arrays of spheres, J. Fluid Mech. 448 (2001) 243–278.

[9] A.J. Sierakowski, A. Prosperetti, Resolved-particle simulation by the Physalis method: enhancements and new capabilities, J. Comput. Phys. 330 (2016) 164–184.
[10] P. Bagchi, S. Balachandar, Effect of turbulence on the drag and lift of a particle, Phys. Fluids 15 (2003) 3496–3513.
[11] L. Zeng, S. Balachandar, P. Fischer, F. Najjar, Interactions of a stationary finite-sized particle with wall turbulence, J. Fluid Mech. 594 (2008) 271–305.
[12] A.A. Johnson, T.E. Tezduyar, Simulation of multiple spheres falling in a liquid-filled tube, Comput. Methods Appl. Mech. Eng. 134 (1996) 351–373.
[13] E.A. Volkov, Method of composite meshes for finite and infinite domain with a piecewise smooth boundary, Proc. Steklov Inst. Math. 96 (1968) 145–185.
[14] G. Starius, Composite mesh difference methods for elliptic boundary value problems, Numer. Math. 28 (1977) 243–258.
[15] J.A. Benek, P.G. Buning, J.L. Steger, A 3-D Chimera Grid Embedding Technique, AIAA Paper No. 85-1523, 1985.
[16] W.M. Chan, Overset grid technology development at NASA Ames Research Center, Comput. Fluids 38 (2009) 496–503.
[17] G. Chesshire, W.D. Henshaw, Composite overlapping meshes for the solution of partial differential equations, J. Comput. Phys. 90 (1990) 1–64.
[18] W.D. Henshaw, A fourth-order accurate method for the incompressible Navier–Stokes equations on overlapping grids, J. Comput. Phys. 113 (1994) 13–25.
[19] H.W. Tang, S. Casey Jones, F. Sotiropoulos, An overset-grid method for 3D unsteady incompressible flows, J. Comput. Phys. 191 (2003) 567–600.
[20] T.M. Burton, J.K. Eaton, Analysis of a fractional-step method on overset grids, J. Comput. Phys. 177 (2002) 336–364.
[21] F.E. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phys. Fluids 8 (1965) 2182.
[22] T.M. Burton, J.K. Eaton, Fully resolved simulations of particle–turbulence interaction, J. Fluid Mech. 545 (2005) 67–111.
[23] A.W. Vreman, Particle-resolved direct numerical simulation of homogeneous isotropic turbulence modified by small fixed spheres, J. Fluid Mech. 796 (2016) 40–85.
[24] N. Phan-Thien, Understanding Viscoelasticity, Springer-Verlag, Berlin, 2013.
[25] H.A. van der Vorst, Bi-CGSTAB – a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear-systems, SIAM J. Sci. 13 (2) (1992) 631–644.
[26] A.W. Vreman, The projection method for the incompressible Navier–Stokes equations: the pressure near a no-slip wall, J. Comput. Phys. 263 (2014) 353–374.
[27] H.H. Hu, N.A. Patankar, M.Y. Zhu, Direct numerical simulations of fluid–solid systems using the arbitrary Lagrangian–Eulerian technique, J. Comput. Phys. 169 (2001) 427–462.
[28] C. Pozrikidis, Introduction to Theoretical and Computational Fluid Dynamics, Oxford University Press, Oxford, 1997.
[29] Y. Tang, S.H.L. Kriebitzsch, E.A.J.F. Peters, M.A. van der Hoef, J.A.M. Kuipers, A methodology for highly accurate results of direct numerical simulations: drag force in dense gas–solid flows at intermediate Reynolds number, Int. J. Multiph. Flow 62 (2014) 73–86.
[30] M.W. Baltussen, Bubbles on the Cutting Edge – Direct Numerical Simulations of Gas–Liquid–Solid Three-Phase Flows, PhD thesis, Eindhoven University of Technology, 2015.
[31] T. Kempe, M. Lennartz, S. Schwarz, J. Fröhlich, Imposing the free-slip condition with a continuous forcing immersed boundary method, J. Comput. Phys. 282 (2015) 183–209.
[32] R. Mei, History force on a sphere due to a step change in the free-stream velocity, Int. J. Multiph. Flow 19 (1993) 509–525.
[33] R. Mei, Flow due to an oscillating sphere and an expression for unsteady drag on the sphere at finite Reynolds number, J. Fluid Mech. 270 (1994) 133–174.
[34] A.G. Kidanemariam, M. Uhlmann, Direct numerical simulation of pattern formation in subaqueous sediment, J. Fluid Mech. 750 (2014) R2.
[35] M.E. Brachet, D.I. Meiron, S.A. Orszag, B.G. Nickel, R.H. Morf, U. Frisch, Small-scale structure of the Taylor–Green vortex, J. Fluid Mech. 130 (1983) 411–452.
[36] L. Schneiders, J.H. Grimmen, M. Meinke, W. Schröder, An efficient numerical method for fully-resolved particle simulations on high-performance computers, Proc. Appl. Math. Mech. 15 (2015) 495–496.
[37] L. Botto, A. Prosperetti, A fully resolved numerical simulation of turbulent flow past one or several spherical particles, Phys. Fluids 24 (2012) 013303.
[38] M. Uhlmann, Simulation of Particulate Flows on Multi-processor Machines with Distributed Memory, Technical Report No. 1039, CIEMAT, Madrid, Spain, 2003.