Journal of Computational Physics 423 (2020) 109783

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Immersed boundary and overset grid methods assessed for Stokes flow due to an oscillating sphere

A.W. Vreman [a,b,*]

[a] Nouryon, Research Development & Innovation, Process Technology, P.O. Box 10, 7400 AA Deventer, the Netherlands
[b] Department of Mechanical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands

ARTICLE INFO

Article history:
Received 3 July 2019
Received in revised form 21 March 2020
Accepted 10 July 2020
Available online 14 August 2020

Keywords:
Immersed boundary methods
Overset grid methods
Moving body problems

ABSTRACT

A systematic comparison of three types of immersed boundary methods and an overset grid method is performed for incompressible Stokes flow due to an oscillating sphere. For each simulation the error between the numerical solution and the analytical solution is reported in terms of the maximum and L_2 norms of the errors in the velocity, the pressure and the velocity gradient fields. The three types of immersed boundary methods are: (A) a diffuse-interface immersed boundary method (IBM), (B) a sharp-interface IBM with standard treatment of the pressure, and (C) a sharp-interface IBM with pressure decoupling at interfaces. Upon grid refinement, the velocity gradient and pressure fields generated by methods A and B do not converge to the analytical solution in the maximum norm, while the total force on the sphere does converge. An inspection of the norms as function of the distance to the surface shows that the lack of convergence mainly occurs in the first layer of grid cells outside the sphere. If the pressure Poisson equation is not solved inside the sphere (method C), the velocity gradient and pressure seem to converge in the maximum norm. However, high resolution is required to achieve a smaller maximum error in the pressure in method C than in methods A and B. Overall, method C is somewhat more accurate than method B and the latter is somewhat more accurate than method A. In this assessment, the overset grid method clearly outperforms the IBMs and is able to produce errors down to one percent in the maximum norm at 32 grid points per sphere diameter. Within each of the four classes of methods, several variants are tested. For the overset grid method and in several sharp-interface IBMs, nonincremental projection is found to produce a more accurate pressure than incremental projection does. Finally, variants of method C and the overset grid method are proposed in which the projection method is replaced by an artificial compressibility method, while the staggered grid is maintained. This simplifies the methods, and it strongly reduces the maximum error in the pressure in case C, at the expense of an increased maximum error in the velocity divergence.

1. Introduction

The term immersed boundary method (IBM) is usually employed for numerical methods that solve the Navier-Stokes equations in complex geometries on Cartesian grids [1]. Many IBM variants can be found in literature, see for example [2–15]. IBMs are attractive methods since they simplify the grid generation and often permit the use of direct FFT (fast

* Correspondence to: Nouryon, Industrial Chemicals, Process Technology R&D, P.O. Box 10, 7400 AA Deventer, the Netherlands.
E-mail addresses: bert.vreman@nouryon.com, bert@vremanresearch.nl.

https://doi.org/10.1016/j.jcp.2020.109783

Fourier transform) solvers for the pressure. There is a crucial difference between IBMs and body-fitted methods. In the latter the local orientation of the grid conforms to the local orientation of the boundary of the fluid domain, which is not the case in IBMs.

The term IBM has been named after a method that Peskin introduced to simulate the motion of an elastic membrane immersed into a fluid [2]. In that method, the velocity boundary condition was applied through a regularized delta function forcing term in the momentum equations. The regularized delta function is a spreading operator that distributes the force from Lagrangian boundary cells to Eulerian cells. It acts as a filter that smoothes out the IBM force defined at the interface over a volume with a thickness of a few grid cells. Goldstein et al. [3] proposed a similar method and applied it to a flow around a solid object. The forcing term had no physical meaning but was just a smoothed delta function relaxation term that depended on a preset parameter and the velocity between the boundary and the fluid. These methods are examples of 'diffuse-interface' IBMs, because the regularization or smoothing of the delta function makes the interface diffuse.

Mohd-Yusof [4] defined the IBM force as the term that is needed to make the velocity at the forcing points equal to the desired velocity. This concept is called 'direct forcing' and needs no preset forcing parameter. Furthermore, the forcing points were not Lagrangian grid points at the boundary but Eulerian grid points adjacent to the surface and inside the body. The velocity at each forcing point was obtained by extrapolation based on the velocity at the physical location of the boundary and the velocity inside the fluid. Since the volume where the flow was forced was only one layer of grid cells, the method in that paper is an example of a 'sharp-interface' IBM. Another 'sharp-interface' IBM was proposed by Fadlun et al. [5]. They did not place the forcing points inside the solid but inside the fluid domain. The velocity at the forcing points was obtained by directional linear interpolation: for each forcing point a suitable one-dimensional interpolation stencil, directed along one of the three directions of the Cartesian grid, was chosen.

Uhlmann noticed that (a particular) sharp-interface direct forcing IBM produced large pressure oscillations [16], and therefore he integrated the direct forcing technique into the regularized delta function formulation of Peskin [8]. The resulting diffuse-interface IBM and variants of it have been applied to many computations of turbulent flows with many suspended particles (see for example [17,18]).

Most IBMs are (incremental) projection methods: in each time step an intermediate velocity is solved, the IBM forcing is applied, the pressure (increment) is solved from a Poisson equation, and its gradient is used to project the velocity on the divergence free space of functions. In the nonincremental projection method, originally proposed by Chorin [19] and Temam [20], the full pressure is used in the projection. In the more frequently used incremental projection method, the pressure gradient from the previous time level is used in the intermediate velocity, and the pressure increment is used in the projection step [22]. An implicit system for the pressure (increment) needs to be solved in both cases. In contrast to projection methods, the artificial compressibility method, also proposed by Chorin [21], often allows explicit treatment of the pressure, because an artificial term proportional to the time derivative of the pressure in the continuity equation is included.

In what we call standard IBMs, the pressure Poisson equation is solved in the entire computational domain, which includes the regions inside solid bodies. However, many IBMs in which the pressure equation is not solved in solid domains have also been proposed. An IBM in this class is called a decoupled immersed boundary method (DIBM) since the pressure in the solid domain is decoupled from the pressure in the fluid domain or not used at all. Thus a DIBM does not assume continuity of the pressure across the interface. An example of a DIBM is the ghost-cell IBM proposed by Tseng and Ferziger [7], who employed a colocated grid with a homogeneous Neumann pressure boundary condition and extrapolation of the velocity to forcing points inside the solid domain. A second example is the IBM approximated domain method [31], in which the method of Fadlun et al. [5] was modified by inclusion of a pressure boundary condition implied by a global mass conservation constraint imposed through a least-square approach on a colocated grid. In a DIBM, the discrete pressure Poisson equation is not defined in the entire computational domain and/or the structure of its left-hand side changes near the boundary, so that fast direct Poisson solvers cannot be used. A third example is the hybrid Cartesian/immersed boundary method proposed by Gilmanov & Sotiropoulos [23], who used a nonhomogeneous Neumann pressure boundary condition, a hybrid colocated staggered grid and an artificial compressibility scheme.

Like in a classical stair step-method, the equations of motion in a DIBM are solved in a set of Cartesian grid cells that forms a stair-step approximation of the physical domain. However, the distance between the location of the physical boundary and the location where the physical boundary velocity is imposed is typically only first order in the grid spacing in a classical stair-step method, while it is typically second order in DIBMs. Cut-cell methods form a another class of Cartesian methods that improve upon classical stair step methods. In cut-cell methods, Cartesian cells intersecting with the boundary are cut by the boundary, and the discrete mass and momentum conservation laws are also applied to the cut cells, see for example [24,25]. Cut-cell methods are likely to be more accurate than IBMs, but they are also much more complicated.

Unfortunately, the use of a regularized delta function or linear interpolation or extrapolation of the velocity combined with at least second-order accurate discretization schemes of the Navier-Stokes equations does not guarantee that the numerical solution is locally second-order accurate. For a sharp-interface standard IBM, Guy & Hartenstine [26] found the velocity field to be first-order accurate in the maximum norm in a test for 2D flow past a fixed cylinder. The local error was concentrated near the boundary, and the first-order accuracy of the velocity was attributed to discontinuity of the velocity gradient across the boundary. For a diffuse-interface standard IBM and a similar test case, Stein et al. [27] found the velocity to be first-order accurate in the maximum norm. The pressure and the velocity gradient showed no convergence in the maximum norm, but the force on the body did converge. The conclusions in these two papers were based on 2D flows and

fixed bodies. It is also known that sharp-interface IBMs can produce spurious pressure oscillations in moving-body problems [8,28–30].

In view of the problems mentioned in the previous paragraph, improvements of standard IBM have been proposed. One example is the immersed boundary approximated domain method (a DIBM in our terminology), which, in contrast to standard IBM, was found to predict accurate results for the wall pressure fluctuation in a turbulent channel flow with walls not aligned to the grid [31]. Kim et al. [6] proposed an IBM in which the Poisson equation, solved in the entire computational domain, was equipped with a cut-cell type of mass source term. The term was found to reduce the spurious pressure oscillations in a 2D moving-cylinder test [28]. Seo et al. [29] proposed a DIBM with a cut-cell discretization for mass conservation and found that it reduced the spurious pressure oscillations in a 2D moving-cylinder test by an order of magnitude, compared to another DIBM. Stein et al. [27] developed the immersed boundary smooth extension method, which has superior accuracy, second-order for the pressure and third-order for the velocity gradient. Due to its complexity, the implementation and validation of the method was restricted to 2D problems with fixed boundaries. In the immersed interface method [32,33], the order of accuracy of the standard immersed boundary method is improved by the inclusion of jump conditions for the discontinuities of the velocity derivatives and the pressure at the interface. Because of the jump conditions, this method is technically more complicated than most IBMs. In various formulations of the immersed interface method, the velocity has been found to be (nearly) second-order accurate and the pressure at least first order accurate in the infinity norm [33–35].

The question arises how accurate most IBMs actually are. Problems with the pressure and velocity gradient have been reported for several IBMs, but at the same time IBMs are widely used for particle-resolved direct numerical simulation (PR-DNS). The smallest turbulence length scale, the Kolmogorov scale, is set by the turbulence dissipation rate, which is directly based on the velocity gradient. For specific investigation of the local properties of the turbulence, for example the turbulence kinetic energy budget, the pressure is also a relevant variable. Thus the question whether the local velocity gradient and the pressure obtained in PR-DNS converge, and if not how large the local errors are, is a relevant one. At least three other questions arise from the above review: how accurate is artificial compressibility (compared to projection), how accurate is the nonincremental projection for moving-body problems (compared to incremental projection), and what is the effect of the order of the extrapolations or interpolations?

The present paper is a comparative study of various IBMs and overset grid methods for Stokes flow due to an oscillating sphere. This flow case is a suitable test case: the flow is 3D (in the Cartesian coordinate system), the body moves, and the analytical solution is known. This nontrivial fundamental flow has apparently not been used to assess IBMs before. It was used for the validation of the overset grid method [36], but the present study goes further and includes new results, also for this method.

An overset grid method is an example of a body-fitted method applicable to complex geometries with moving bodies. In an overset grid method, at least two mutually overlapping grids are used [32–44]. For example, a structured body-fitted grid attached to a moving sphere is overset on a Cartesian grid. The Cartesian grid contains a spherical hole around and moving with the sphere, and interpolation is used to interpolate the solution from one grid to another [36]. Comparisons between IBMs and body-fitted methods for moving problems have rarely been performed before. In a previous study, in which the author compared the overset grid method with a diffuse-interface method for a free particle falling in a laminar channel flow [36], good agreement for the translative and angular velocities of the particle was obtained, but no fluid variables were compared. For turbulent flow, a comparative study of different methods for particle-resolved simulations has been performed [45], but statistical uncertainty played a role, and no body-fitted method and no exact solution were available.

The main topic of this paper is the comparison of four base methods based on projection and two variants based on artificial compressibility. The four base methods are: (A) a diffuse-interface IBM, (B) a sharp-interface IBM, (C) a sharp-interface decoupled IBM, and (O) an overset grid method. The two variants based on artificial compressibility are derived from methods C and O. The results are compared in terms of maximum and L_2 norms of the errors in the velocity, velocity gradient and pressure. The maximum norms quantify the local errors, the L_2 norms the global errors. Furthermore, radial profiles of errors in the solution are compared, profiles that show the errors as functions of the distance to the center of the sphere.

Since it is impossible to compare or even cite all IBM variants in one paper, choices had to be made. To keep the IBM implementations relatively simple, IBMs that incorporate cut-cell technology have not been included into the comparison. Furthermore, no colocated or hybrid but only fully staggered grids [46] were used. Methods A and O are very similar to existing methods in literature [8,36], while other methods contain a few new elements. For example, the fully explicit partially quadratic extrapolation scheme used in methods B and C, appears to be a simple and effective scheme that does not appear in literature in this form, although it was inspired by various concepts in literature [5,6,14]. In addition, the definition of the forcing points in method C is probably new in an IBM, while the use of artificial compressibility in combination with fully staggered grids (in above-mentioned variants) is also unconventional. A number of other variants are included to show, for a moving-body problem, the effects of linear versus quadratic interpolation, incremental versus nonincremental projection, implicit versus explicit methods, interface retraction in diffuse interface methods and the effects of a few other options.

The set-up of this paper is as follows. The test case is formulated in section 2. The four base methods and the two artificially compressibility variants are described in section 3. The comparison of the results of the four base methods is presented in section 4. The comparison of the results of the two artificially compressibility variants and their projection

counterparts is performed in section 5. The other variants are briefly introduced and discussed in section 6. The conclusions are summarized in section 7.

2. Definition of test case

We consider a nonrotating spherical particle with position $\mathbf{x}^p$ and velocity $\mathbf{v}^p$, which both depend on time t. The particle radius is defined by $r_0 = 1/2$, so that the particle diameter $d_p = 2r_0$ equals 1. The sphere oscillates around location $\mathbf{x}^{cen}$ according to the following prescribed motion:

$$\mathbf{x}^p(t) = \mathbf{x}^{cen} - \Re\Big(\frac{e^{-i\omega t}}{\omega}\mathbf{V}\Big), \qquad \mathbf{v}^p(t) = \Re\Big(ie^{-i\omega t}\mathbf{V}\Big). \tag{1}$$

The function $\Re(.)$ takes the real part of its argument, and, only in this section, i represents the imaginary unit. We choose $\mathbf{V} = (1, 0, 0)$, which means that the sphere oscillates in the x_1 direction with velocity amplitude 1. The frequency of the oscillation (ω) is 2π. Thus one oscillation takes one time unit and the spatial amplitude of the oscillation is $1/2\pi$.

The sphere is embedded in a fluid with density $\rho = 1$ and kinematic viscosity $\nu = 1$. The fluid variables are $\mathbf{u}$ (velocity) and q (pressure divided by density, also simply called pressure). The force on the sphere divided by ρ is defined by $\mathbf{F}^p$. The analytical solution for Stokes flow due to the oscillating sphere can be written as [47,36]:

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

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

$$\mathbf{F}^p_{ex} = -6\pi\nu r_0\,\Re\Big(ie^{-i\omega t}(1 + \lambda + \lambda^2/9)\mathbf{V}\Big), \tag{4}$$

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

$$B = 6\pi\nu r_0(1 + \lambda + \lambda^2/3), \qquad Q = -6\pi r_0^3(e^\lambda - 1), \qquad R = \frac{\lambda r}{r_0}, \tag{5}$$

$$\lambda = (1 - i)r_0\Big(\frac{|\omega|}{2\nu}\Big)^{1/2}. \tag{6}$$

The subscript 'ex' denotes that this velocity, pressure and force are exact solutions of

$$\nabla\cdot u = 0, \tag{7}$$

$$\frac{\partial\mathbf{u}}{\partial t} = -\nabla p + \nu\nabla^2\mathbf{u} - \mathbf{v}^p\cdot\nabla\mathbf{u}, \tag{8}$$

$$\mathbf{F}^p = \int_{S^p}(-p\mathbf{I} + \nu(\nabla u + (\nabla u)^T))\cdot\mathbf{n}\,\mathrm{d}S^p, \tag{9}$$

where S^p is the surface of the sphere and $\mathbf{n}$ the surface normal vector normal pointing towards the fluid. The last term in equation (8) is a convective term that appears after transformation of the unsteady Stokes equations (which are not Galilean invariant) from the reference frame attached to the sphere to the observer (nonmoving) reference frame.

We restrict the solution to the domain $\Omega = \Omega_c\backslash V$, where Ω_c is a cubical domain with length $L = 5r_0$. The center point of the oscillation, $\mathbf{x}^{cen}$ is also the center of Ω_c. The time dependent region V is the volume inside the sphere with center $\mathbf{x}^p(t)$ and radius r_0. The amplitude of the spatial oscillation is $d_p/(2\pi)$, so that the minimum distance of the sphere to the outer boundary of Ω is slightly larger than r_0.

The velocity boundary conditions are given by $\mathbf{u} = \mathbf{v}^p$ if $|\mathbf{x} - \mathbf{x}^p| = r_0$ and $\mathbf{u} = \mathbf{u}_{ex}$ at the outer boundary of Ω. The initial condition of the velocity is $\mathbf{u} = \mathbf{u}_{ex}$ at $t = 0$. Equations (7)-(8) are supplemented with the constraint

$$\int_\Omega p\,\mathrm{d}\Omega = \int_\Omega p_{ex}\,\mathrm{d}\Omega, \tag{10}$$

to ensure that p is unique.

3. Numerical methods

First we describe what is the same for all methods presented, and in the subsections afterwards, we present four base methods, which are all projection methods: (A) a diffuse-interface IBM, (B) a sharp-interface IBM, (C) a sharp-interface IBM with pressure decoupling at interfaces (DIBM), and (O) an overset grid method. In the last subsection, we formulate methods C′ and O′, the counterparts of methods C and O, in which the projection schemes are replaced by artificial compressibility schemes.

In all methods, the computational domain Ω_c (the entire cube) is discretized by a Cartesian grid with uniform grid size h (in the overset grid methods this is not the only grid). The grid size is $h/(16\kappa)$ and the time step is $\Delta t = 0.0004/\kappa^2$. We use the values $\kappa = 1$, 2 and 4. These values correspond to 40^3 and 80^3 and 160^3 grid cells, respectively. The number of grid cells across the particle diameter is 16κ.

Each method is fully staggered and uses the standard second-order central finite differencing scheme for the approximation of spatial derivatives. Thus the basic spatial accuracy of the methods is formally $O(h^2)$. The time step Δt is approximately $0.1h^2/\nu$, somewhat lower than the 3D viscous stability limit $h^2/6\nu$ of the forward Euler scheme. Using this definition, the formal truncation error of the forward Euler scheme is $O(\Delta t) = O(h^2)$.

The outer boundary of Ω, $\partial\Omega_c$, is treated as a Dirichlet boundary for $\mathbf{u}$, where the exact solution is prescribed: $\mathbf{u} = \mathbf{u}_{ex}$ at $\partial\Omega_c$. This boundary coincides with the faces of the (pressure) cells. The normal velocity is specified on $\partial\Omega_c$ and the tangential velocities in the layer of ghost-cells just outside Ω_c. The pressure constraint, equation (10), is satisfied in the post processing, by adding the integral of $p_{ex} - p$ to the pressure that was solved.

Physically, no pressure boundary condition is needed at a Dirichlet boundary for the velocity in an incompressible flow. Also the staggered scheme does not need a pressure boundary condition at $\partial\Omega_c$. However, it can be convenient to use a numerical boundary condition for the pressure. For example, if we do not want to change the stencil for the pressure Laplacian near the outer boundary, then we need the pressure in the ghost-cells just outside Ω_c. That pressure can be obtained from a numerical boundary condition for the normal component of the pressure gradient, $\partial p/\partial n$, whose value is related to the value prescribed to the normal component of the intermediate velocity just before projection. If the latter value is the same as the normal component of the velocity at the new time level, then the numerically implied pressure boundary condition is $\partial p/\partial n = 0$. It is stressed that this does not mean that the staggered method assumes that the physical pressure gradient is zero at the boundary. The numerical solution for the pressure remains the same if the physical value of $\partial p/\partial n$ is used, provided the boundary value of the normal component of the intermediate velocity before projection is adapted accordingly [48].

The iterative solver used is the BiCGstab method [49], combined with Jacobi preconditioning. The convergence criterion is set to 10^{-7} for the maximum norm of the residual based on a matrix with diagonal coefficients scaled to one.

3.1. Method A: diffuse-interface IBM

The diffuse-interface IBM described in this section and labeled as method A is based on the direct forcing IBM proposed by Uhlmann [8].

Each time step in method A consists of the following steps:

1. Compute the intermediate velocity in the entire domain (Ω_c):

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t \mathbf{H}^n. \tag{11}$$

2. Compute and apply the immersed boundary forcing $\mathbf{f}$,

$$\mathbf{u}^{**} = \mathbf{u}^* + \Delta t\, \mathbf{f}. \tag{12}$$

3. Solve the pressure Poisson equation in the entire domain (Ω_c),

$$\nabla^2 p^{n+1} = \frac{\nabla \cdot \mathbf{u}^{**}}{\Delta t}. \tag{13}$$

4. Compute the fluid velocity at time level $n+1$ in the entire domain Ω_c,

$$\mathbf{u}^{n+1} = \mathbf{u}^{**} - \Delta t\, \nabla p^{n+1}. \tag{14}$$

5. Compute the force on the particle,

$$\mathbf{F}^p = -\int_{\Omega_c} \mathbf{f}\, \mathrm{d}\Omega_c + \frac{\pi d_p^3}{6} \frac{d\mathbf{v}^p}{dt}. \tag{15}$$

6. Update the particle location and the particle velocity, according to the prescribed motion.

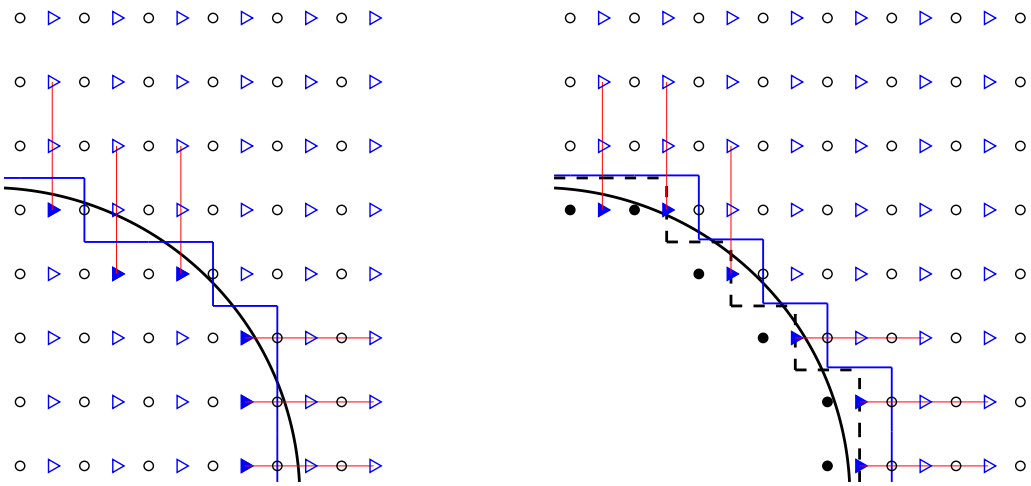

Fig. 1. 2D representation of the ghost cell definitions in method B (left (a)) and in method C (right (b)). The large circle represents the immersed boundary. Grid points of the pressure are indicated by circles and those of the horizontal component of the staggered velocity by triangles. The forcing (ghost) points of this velocity component, where the velocity is extrapolated, are denoted by filled triangles. The red lines connected to these points denote the extrapolation stencils. The thick stair-step line denotes the faces of the border of the domain formed by the grid points of the horizontal velocity component. This border is used to compute the horizontal component of the force on the object. The dashed stair-step line (right plot) is the border formed by the faces of the fluid pressure cells with one neighbor in the solid. These neighbors are the pressure ghost points (filled circles). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

The time level is denoted by superscript n. The term $\mathbf{H}^n$ denotes the combined viscous and convective terms, defined by $\nu\nabla^2\mathbf{u}^n - \mathbf{v}^p \cdot \nabla\mathbf{u}^n$ in this paper (see equation (8)).

The forcing term $\mathbf{f}$ in step 2 in method A is obtained as follows. The interface of the sphere is represented by approximately $\pi d_p^2/h^2$ Lagrangian points, equally distributed by a preprocessing program. In the actual immersed boundary simulation, the velocity $\mathbf{u}^{**}$ is interpolated from Eulerian to Lagrangian points. Then the direct forcing principle is applied by computing the forcing term $(\mathbf{v}^{p,n+1} - \mathbf{u}^*)/\Delta t$ in the Lagrangian points. Subsequent spreading of this forcing term from the Lagrangian to Eulerian points yields $\mathbf{f}$. For the interpolation and spreading operations, the regularized Dirac delta function of Roma et al. [50] is used. The volume integral of $\mathbf{f}$ in step 5 is obtained using the midpoint rule (the forcing term $\mathbf{f}$ is zero except in several layers of cells around the interface).

3.2. Method B: sharp-interface IBM

Method B differs from method A in two respects: the diffuse immersed boundary force is replaced by a sharp force implied by extrapolation of the velocity to so-called ghost-cells and the total force on the sphere is evaluated by integration over stair-step approximations of the immersed boundary. More specifically, steps 2 and 5 in method A are replaced by

2. Adapt the velocity at grid points close to and inside the surface of the sphere via a so-called ghost-cell technique and call the resulting velocity $\mathbf{u}^{**}$.
5. Compute the force on the particle, via a stair-step approximation of equation (9).

The immersed boundary forcing term can be derived and is equal to $\mathbf{f} = (\mathbf{u}^{**} - \mathbf{u}^*)/\Delta t$. In practice this term is not computed. The forcing points (or ghost points) are the grid points where the velocity is adapted in step 2. Each velocity grid point inside the sphere that has at least one neighbor in the fluid is a forcing point. Fig. 1a shows a 2D illustration of the forcing points of the horizontal velocity component and the corresponding extrapolation stencils.

The extrapolation stencils are one-dimensional, like the interpolation stencils in Fadlun et al. [5]. Each stencil consists of three points, corresponding to the three numbers X_0, X_1 and X_2. X_0 corresponds to the point on the surface, X_1 to the first grid point inside the fluid and X_2 to the second grid point inside the fluid. The latter is not always used. The corresponding values of the velocity component under consideration are denoted by U_0, U_1 and U_2. Furthermore, X_e corresponds to the extrapolation point, and the extrapolated value of the velocity component is denoted by U_e. Since the grid is uniform we can assume $X_e = 0$, $X_0 = \beta h$, $X_1 = h$ and $X_2 = 2h$, without loss of generality. Since the surface is spherical, it is straightforward to find an analytical expression for β. The extrapolation formula is given by

$$U_e = w_0 U_0 + w_1 U_1 + w_2 U_2. \tag{16}$$

The extrapolation weights w_0, w_1 and w_2 are given by

$$w_0 = 2/((1-\beta)(2-\beta)), \quad w_1 = -2\beta/(1-\beta), \quad w_2 = \beta/(2-\beta) \qquad \text{if } \beta < \beta_1; \tag{17}$$

$$w_0 = 2/((1-\beta_1)(2-\beta_1)), \quad w_1 = 2-(2-\beta)w_0, \quad w_2 = -1+(1-\beta)w_0 \qquad \text{if } \beta \geq \beta_1. \tag{18}$$

The first part of the extrapolation scheme ($\beta < \beta_1$) is third-order and the second part second-order accurate. The separation parameter β_1 is introduced to prevent large values of the weights for β close to 1 ($0 \leq \beta < 1$ in method B). Unless mentioned otherwise, $\beta_1 = 1/2$ in this paper. Equation (17) is equivalent to the quadratic extrapolation proposed by Deen et al. [14], who used $\beta_1 = 0.9999$ [51] after incorporating the weights into the coefficients of the matrix of the viscous terms, which were treated implicitly. Equation (18) is a new element, implied by the two constraints for second-order accuracy, $w_0 + w_1 + w_2 = 1$ and $\beta w_0 + w_1 + 2w_2 = 0$, and the additional constraint $w_0(\beta) = w_0(\beta_1)$. A linear variant of above extrapolation scheme is obtained after changing w_0 to $1/(1-\beta)$ if $\beta < \beta_1$ and changing w_1 to $-\beta/(1-\beta)$ and w_2 to zero if $\beta \geq \beta_1$. For $\beta = 1/2$, this linear variant reduces to the simplest linear extrapolation scheme in Kim et al. [6].

In the present paper, the direction of each one-dimensional extrapolation stencil is based on the local normal vector $\mathbf{n}$, which is the vector along the line normal to the surface and through the ghost point. The largest component of the normal vector determines the direction of the stencil, more precisely, the stencil is aligned with the x_k direction if $|n_k| \geq \max\{|n_1|, |n_2|, |n_3|\}$. If this leads to multiple values for k, then the lowest value k is chosen. It is remarked that also with respect to the extrapolation direction, method B differs from the method of Deen et al. [14], since they defined an extrapolation for each second-order partial derivative that used a ghost velocity and defined the direction of the extrapolation as the direction in which the partial derivative was taken.

The force on the sphere is computed using a stair-step approximation of the surface (see Fig. 1a). Since the grid is staggered, three different stair-step approximations are used, one for each component of the force. We explain the computation of the x_1 component of the force in the x_1 direction. The corresponding stair-step boundary for the x_1 direction is formed by all boundary faces of the u_1 cells. Each face of a u_1 cell is located half way two neighboring u_1 grid points, and it is a boundary face if one grid point is inside the fluid domain and the other in the solid domain. The area of each face is h^2. The pressure on the boundary face is only needed if the normal of the boundary face is pointing in the x_1 direction, and then the pressure is located at the center of the face. For each boundary face, the velocity derivatives that are needed to evaluate the force are obtained via straightforward central differencing, using ghost and fluid points, in the same way as the derivatives in the viscous fluxes are computed.

3.3. Method C: sharp-interface decoupled IBM

Method C is a decoupled IBM that differs from method B in three main aspects: the pressure Poisson equation is not solved inside the sphere, the locations of ghost points are different, and the boundary motion and extrapolations appear at the beginning of the time step.

Although in DIBMs in literature colocated or hybrid colocated staggered grids are commonly used [7,23,31,9,29], the grid in method C is fully staggered, like in all other methods in this paper. To achieve low mass conservation errors in a staggered DIBM, it is useful to introduce another definition of the location of the staggered ghost points, as described in this subsection.

Each time step in method C consists of the following steps:

1. Update the particle location and particle velocity (to time level $n+1$), according to the prescribed motion.
2. Adapt the velocity at grid points close to and inside the surface of the sphere via a so-called ghost-cell technique, and based on the particle velocity at time level $n+1$. Update the velocity at the outer boundary $\partial\Omega_c$ to time level $n+1$. The velocity field including the new ghost velocities is called $\mathbf{u}^*$.
3. Extrapolate the pressure at fresh pressure points.
4. Compute an intermediate velocity (only in the fluid domain Ω and not in ghost cells):

$$\mathbf{u}^{**} = \mathbf{u}^n + \Delta t\, \mathbf{H}^*, \tag{19}$$

   where $\mathbf{H}^*$ denotes the convective and viscous terms based on $\mathbf{u}^*$.
5. Solve the following (augmented) Poisson problem for p^{n+1}, only inside the fluid domain Ω:

$$\nabla^2 p^{n+1} + b = \frac{\nabla \cdot \mathbf{u}^{**}}{\Delta t}, \tag{20}$$

$$\int_\Omega p^{n+1} d\Omega = 0. \tag{21}$$

6. Compute the fluid velocity at time level $n+1$ (only in the fluid domain Ω and not in ghost cells),

$$\mathbf{u}^{n+1} = \mathbf{u}^{**} - \Delta t\, \nabla p^{n+1}. \tag{22}$$

7. Compute the force on the particle, via a stair-step approximation of equation (9).

To decouple the pressure at the immersed boundary, pressure ghost points are defined, which are the pressure points inside the solid body with at least one neighbor pressure point in the fluid domain, see the illustration in Fig. 1b. The corresponding pressure boundary faces, faces between a ghost pressure and a fluid pressure point, form a stair-step approximation of the interface (dashed line in Fig. 1b). The velocity ghost points are determined by the pressure boundary faces: a velocity point is a ghost point if it is located on a pressure boundary face or if it is half-way in between two neighboring pressure ghost points. As a result, some velocity ghost points can also appear in the fluid domain.

Apart from the different locations of the ghost points, the extrapolation stencil of the velocity ghost points is defined in the same way as in method B. Also the same extrapolation formula is used, but in method C the range is extended to $-\frac{1}{2} \leq \beta < 1$. For negative β, the scheme performs an interpolation (the ghost point is then in the fluid domain).

Since the iterative solver of the Poisson equation starts with the pressure of the previous time step (p^n), the pressure is extrapolated to fresh pressure points (step 3). Fresh pressure points are pressure points in the fluid domain that were ghost pressure points in the previous time-step. The extrapolation is defined as the arithmetic average over the pressures at the neighboring points that were pressure points in the fluid domain in the previous time-step.

The variable b in equation (20) is a so-called augmented variable [39,41,43], which only depends on time. It is an effective way to remove the singularity of the pressure Poisson problem. Since the discrete $\nabla \cdot \mathbf{u}^{**}$ summed over all fluid grid points is not exactly zero, b is not zero, but usually b is small and converges to zero if the grid size and the tolerance of the iterative solver converge to zero.

The discretization of (20) is derived from the requirement $\nabla \mathbf{u}^{n+1} = b$ at all pressure points in the fluid domain. This requirement requires a specification of the normal component of the velocity $\mathbf{u}^{n+1}$ at the pressure boundary faces (see the dashed stair-step line in Fig. 1b). The assumption made is that $u_i^{n+1} = u_i^*$ at each pressure boundary face, where i refers to the direction of the face-normal vector of the boundary face. Because the grid is staggered, no prescription of the face-normal pressure derivative $\partial p/\partial x_i$ is needed at the boundary face. This can be explained as follows. Suppose we set $\partial p/\partial x_i = g_i$ and $u_i^{**} = u_i^* + g_i \Delta t$ at each pressure boundary face with face-normal vector pointing in the i direction. Then the terms with g_i in both sides of the discrete pressure Poisson equation cancel out. Thus, if we keep g_i at both sides of the discrete pressure Poisson equation, any value of g_i can be chosen in the numerical scheme. Whether we choose an accurate approximation for g_i or simply $g_i = 0$ does not matter, since g_i does not affect the computed pressure field (see also the explanation in [48]). We choose $g_i = 0$, an arbitrary but convenient choice, which implies that the pressure ghost points drop out of the stencil of the pressure Laplacian and $u_i^{**} = u_i^*$ is used at the pressure boundary faces to compute the discrete $\nabla \mathbf{u}^{**}$. In other words, the coefficient matrix of the pressure equation is modified, and there is no need to compute the pressure at the pressure ghost points.

3.4. Method O: overset grid method

Method O is the overset grid method previously developed by the author [36]. A spherical grid attached and moving with the sphere is overset on a stagnant Cartesian grid, which contains a moving hole cut around the sphere. The partial differential equations are solved in spherical coordinates on the spherical grid and in Cartesian coordinates on the Cartesian grid. It is not an immersed boundary method but a body-fitted method. The radius of the outer boundary of the spherical grid (r_b) is somewhat larger than the radius of the hole in the Cartesian grid (r_a). The grids overlap in the region in which distance r to the particle center satisfies $r_a < r < r_b$. The solutions on various grids are connected through third-order interpolations at $r = r_a = r_0 + 2\kappa h$ and $r = r_b = r_0 + 5\kappa h$. The Navier-Stokes equations are not solved inside the sphere.

Each time step in method O consists of the following steps:

1. Update the particle location and particle velocity (to time level $n+1$), according to the prescribed motion.
2. Update the interpolation stencils and interpolation weights.
3. Interpolate the velocity $\mathbf{u}^n$. Update the velocity at $\partial\Omega$ to time level $n+1$.
4. Compute the intermediate velocity:

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t\, \mathbf{H}, \tag{23}$$

   where $\mathbf{H}$ denotes the convective and viscous terms (implicit in the azimuthal direction on the spherical grid).
5. Interpolate the velocity $\mathbf{u}^*$.
6. Solve an augmented Poisson problem for p^{n+1}:

$$\nabla^2 p^{n+1} + b = \frac{\nabla \cdot \mathbf{u}^*}{\Delta t}, \tag{24}$$

$$\sum_k p_k^{n+1} = 0, \tag{25}$$

   where k is the grid point index. The pressure is interpolated in each iteration.
7. Compute the fluid velocity at time level $n+1$ (only in the fluid domain Ω and not in ghost cells),

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t\, \nabla p^{n+1}. \tag{26}$$

8. Interpolate the velocity $\mathbf{u}^{n+1}$.
9. Compute the force on the particle, via equation (9), using linear extrapolation to obtain the pressure at the surface and the mid-point rule for the integration over the surface.

3.5. Artificial compressibility variants

Methods C′ and O′ are the same as methods C and O except that the projection is replaced by artificial compressibility. Unconventional elements in the proposed artificial compressibility methods are the use of a staggered grid and a grid-dependent speed of sound. The artificial speed of sound, u_{sou}, is set to $h/(3\Delta t) \approx 52\kappa$. Thus the Courant number corresponding to the speed of sound is 1/3 and the speed of sound is proportional to the grid refinement factor κ. The corresponding artificial Mach number $|\mathbf{V}|/u_{sou}$ is approximately $0.02/\kappa$.

Method C′ is identical to method C, except that steps 4 to 6 in subsection 3.3 are replaced by:

4. Compute the fluid velocity at time level $n+1$ (only in the fluid domain Ω and not in ghost cells):

$$\mathbf{u}^* = \mathbf{u}^n - \Delta t\, \nabla p^n + \Delta t\, \mathbf{H}^n. \tag{27}$$

5. Repeat step 2 (ghost-cell velocity extrapolations for $\mathbf{u}^*$).
6. Solve the pressure at time level $n+1$ (only in the fluid domain Ω and not in ghost cells):

$$p^{n+1} = p^n - \Delta t\, u_{sou}^2 \nabla \cdot \mathbf{u}^*. \tag{28}$$

The overset grid method O′ is identical to method O, except that step 6 in subsection 3.4 is replaced by:

6. Update the pressure,

$$p^{n+1} = p^n - \Delta t\, u_{sou}^2 \Big(- \Delta t\, \nabla^2 p + \nabla \cdot \mathbf{u}^* \Big), \tag{29}$$

and interpolate the pressure afterwards.

The discretization of all second-order derivatives inside $\nabla^2 p$ is explicit, except those in the azimuthal direction on the spherical grid. For these implicit terms, a tri-diagonal solver is used.

4. Comparison of the four base methods

4.1. Definitions of discrete norms

For the definition of discrete norms, we need to define time-dependent sets of the discrete locations of variables. For a given time t, the set Ω^0 is the collection of pressure points inside the fluid domain, more specifically all Cartesian cell centers $\mathbf{x}$ with $|\mathbf{x} - \mathbf{x}^p| \geq r_0$. If two points are neighbors of each other, we call them a pair. A pair is always aligned to one of the coordinate directions. The locations of velocity component u_j are the midpoints of pairs of pressure points aligned to direction x_j. The set Ω_j^u is the collection of locations of u_j inside the physical domain (one of the locations of the corresponding pressure points can be outside the fluid domain). The locations of velocity gradient component $\partial u_i/\partial x_k$ are the midpoints of the pairs of u_i locations aligned to direction x_k. The set Ω_{jk}^g is the collection of locations of $\partial u_j/\partial x_k$ inside the physical domain. The latter nine sets fall apart in four different sets: $\Omega_{jk}^g = \Omega^0$ if $j = k$ and $\Omega_{jk}^g = \Omega_{kj}^g$ if $j \neq k$.

We define discrete L_∞ and L_2 norms over the physical domain Ω and a set of all discrete times between $T = [t_1, t_2]$. The L_∞ norm is also called the maximum norm. The L_2 norm used is also called the RMS (root mean square) norm since each sum is normalized by the number of points (or times) over which the sum is taken. The numbers of elements in a set are denoted by absolute values, for example $|T|$ denotes the number of elements in T and $|\Omega^0|$ the number of elements in Ω^0 at time t. The L_∞ norms are defined by

$$\|p\|_\infty = \max_{t\in T} \max_{\Omega^0} |p|, \tag{30}$$

$$\|\mathbf{u}\|_\infty = \max_j \max_{t\in T} \max_{\Omega_j^u} |u_j|, \tag{31}$$

$$\|\nabla \mathbf{u}\|_\infty = \max_j \max_k \max_{t\in T} \max_{\Omega_{jk}^g} \left|\frac{\partial u_j}{\partial u_k}\right|, \tag{32}$$

$$\|F_1^p\|_\infty = \max_{t\in T} |F_1^p|, \tag{33}$$

and the L_2 norms are defined by

Table 1
Discrete norms of the exact solution restricted to the Cartesian grids for the resolutions $\kappa = 1$, 2 and 4.

κ	$\|F^p_{ex,1}\|_\infty$	$\|\mathbf{u}_{ex}\|_\infty$	$\|q_{ex}\|_\infty$	$\|\nabla\mathbf{u}_{ex}\|_\infty$	$\|F^p_{ex,1}\|_2$	$\|\mathbf{u}_{ex}\|_2$	$\|q_{ex}\|_2$	$\|\nabla\mathbf{u}_{ex}\|_2$
1	20.40	1.000	7.03	6.25	14.42	0.208	0.759	0.744
2	20.40	1.000	7.01	6.25	14.42	0.208	0.759	0.742
4	20.40	1.000	7.05	6.25	14.42	0.208	0.759	0.741

$$\|p\|_2 = \Big(\frac{1}{|T|}\sum_{t\in T}\Big(\frac{1}{|\Omega^0|}\sum_{\Omega^0}|p|^2\Big)\Big)^{1/2}, \tag{34}$$

$$\|\mathbf{u}\|_2 = \Big(\sum_j\Big(\frac{1}{|T|}\sum_{t\in T}\Big(\frac{1}{|\Omega^u_j|}\sum_{\Omega^u_j}|u_j|^2\Big)\Big)\Big)^{1/2}, \tag{35}$$

$$\|\nabla\mathbf{u}\|_2 = \Big(\sum_j\sum_k\Big(\frac{1}{|T|}\sum_{t\in T}\Big(\frac{1}{|\Omega^g_{jk}|}\sum_{\Omega^g_{jk}}|\frac{\partial u_j}{\partial u_k}|^2\Big)\Big)\Big)^{1/2}, \tag{36}$$

$$\|F^p_1\|_2 = \Big(\frac{1}{|T|}\sum_{t\in T}|F^p_1|\Big)^{1/2}. \tag{37}$$

Since the grid is assumed to be uniform in these definitions, each spatial sum divided by the number of in the set represents an approximated spatial integral over the physical domain normalized by the volume of the physical domain. In the overset grid case, there are two spatial domains, the spherical domain and the Cartesian domain, which has a spherical hole larger than the body. Then each spatial maximum in the definition of the L_∞ norms is taken over the grid points in both the spherical and Cartesian domains. Likewise, each approximated spatial integral in the definitions of the L_2 norms is then replaced by the approximated integral over the spherical domain plus the approximated integral over the Cartesian domain. The sum of these integrals is then normalized by the sum of the volume of the spherical and the volume of the Cartesian domain.

Due to time symmetries in the solution, all discrete grid configurations encountered in one oscillation period are the same as all grid configurations encountered in an arbitrary interval $[t_1, t_1+0.5]$. Thus the norms should formally not depend on t_1 if $t_1 = t_2 - 0.5$, apart from a short transient effect for small t_1, caused by the discrete representation of the initial condition. To give all methods sufficient time to adapt to the sudden imposition of the initial analytical condition, we choose $t_1 = 0.1$ and $t_2 = 0.6$ (because methods A and B solve the pressure inside the body, these methods need approximately 100 time steps for this adaptation, compared to approximately 5 time steps for methods C and O). Table 1 gives an overview of the discrete norms of the analytical solution.

4.2. Norms of errors

For the numerical solution, we express the norms of errors as percentages of the analytical norms. The relative (percent) L_∞ and L_2 errors of the velocity are defined by

$$\frac{\|\mathbf{u}-\mathbf{u}_{ex}\|_\infty}{\|\mathbf{u}_{ex}\|_\infty}\times 100\% \quad \text{and} \quad \frac{\|\mathbf{u}-\mathbf{u}_{ex}\|_2}{\|\mathbf{u}_{ex}\|_2}\times 100\%, \tag{38}$$

respectively. The percent errors of the pressure, velocity gradient and particle force are defined in the same way. Since the velocity divergence and the pressure are defined at the same set of points, the definition of the discrete pressure norm can also be applied to the velocity divergence. Since the exact velocity divergence is zero, the relative L_∞ and L_2 errors of $\nabla\cdot\mathbf{u}$ are given as percentages of the L_∞ and L_2 norms of $\nabla\mathbf{u}_{ex}$, respectively.

The results for the norms of the errors for the four base cases and for the two variants based on artificial compressibility are shown in Table 2, and the corresponding orders of convergence in Table 3. These two tables with results of six methods give an overview of the main results of this paper. The comparison of the results of the first four methods (A, B, C and O) is the subject of this section, and the comparison of the last two methods (C′ and O′) with their projection counterparts is the subject of the next section.

The norms of the velocity divergence in Table 2 are shown as zero for methods A and B, since in these cases the norms are of the order of the convergence criterion used for the iterative Poisson solver (10^{-7}). For methods C and O, the norms of the velocity divergence are not zero, but they are small ($< 0.5\%$ of the norms of the exact velocity gradient), and they converge to zero upon grid refinement. So far the discussion of the errors in the divergence in this section; we will return to this topic in the next section.

The error norms for F, $\mathbf{u}$, p and $\nabla\mathbf{u}$ are plotted in Figs. 2 and 3. The approximately linear slope of the curve of the quantities in these loglog plots represents the order of convergence of the quantities. The slopes have been estimated and are listed in Table 3.

Table 2
Percent L_∞ and L_2 errors for the four base methods A, B, C and O (and the variants C′ and O′ discussed in section 5), for the resolutions $\kappa = 1$, 2 and 4, corresponding to $d_p/h = 16$, 32 and 64, respectively.

	κ	Relative L_∞ errors [%]					Relative L_2 errors [%]				
		F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$	F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$
A	1	8.11	7.77	45.27	54.94	0	8.11	4.34	17.70	16.83	0
	2	3.62	4.27	54.88	52.05	0	3.63	2.12	10.60	11.10	0
	4	1.69	2.22	60.04	50.76	0	1.69	1.05	7.14	7.36	0
B	1	2.92	4.10	24.80	8.48	0	1.88	1.17	4.32	3.36	0
	2	1.96	1.92	21.46	7.10	0	0.99	0.67	2.46	1.81	0
	4	1.08	1.12	21.15	6.26	0	0.52	0.40	1.37	1.15	0
C	1	4.59	1.19	46.65	7.27	0.0015	2.64	0.25	3.74	1.30	0.0028
	2	1.89	0.52	22.86	2.87	0.0001	1.28	0.06	1.13	0.43	0.0003
	4	0.82	0.17	9.66	1.71	0.0000	0.62	0.02	0.33	0.15	0.0000
O	1	2.19	0.84	2.27	2.18	0.498	2.14	0.17	0.68	0.85	0.068
	2	0.88	0.12	0.58	0.94	0.139	0.88	0.04	0.15	0.23	0.014
	4	0.40	0.04	0.22	0.45	0.039	0.40	0.01	0.05	0.08	0.003
C′	1	3.02	1.14	7.89	8.97	7.69	3.01	0.62	2.15	1.46	0.30
	2	1.50	0.52	4.54	3.77	3.65	1.46	0.15	0.66	0.46	0.07
	4	0.71	0.17	2.74	1.70	1.63	0.71	0.04	0.21	0.16	0.02
O′	1	2.39	0.86	1.61	2.50	0.63	2.41	0.63	2.26	1.06	0.26
	2	0.96	0.13	0.57	1.03	0.19	0.96	0.16	0.63	0.29	0.06
	4	0.42	0.04	0.23	0.47	0.05	0.42	0.04	0.17	0.09	0.02

Table 3
Estimates of the orders of convergence in the L_∞ and L_2 norms for the four base methods A, B, C and O (and the variants C′ and O′ discussed in section 5).

	Order of convergence (L_∞)					Order of convergence (L_2)				
	F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$	F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$
A	1.1	0.9	-0.2	0.1	-	1.1	1.0	0.7	0.6	-
B	0.7	0.9	0.1	0.2	-	0.9	0.8	0.8	0.8	-
C	1.2	1.4	1.1	1.0	3.4	1.0	2.0	1.8	1.5	3.4
O	1.2	2.2	1.7	1.1	1.8	1.2	1.9	1.8	1.7	2.2
C′	1.0	1.4	0.8	1.2	1.1	1.0	2.0	1.7	1.6	2.0
O′	1.3	2.3	1.4	1.2	1.8	1.3	2.0	1.9	1.8	2.0

First, we discuss the errors in the force on the particle (Figs. 2a,b). The L_∞ and L_2 errors represent the local and global accuracies of the temporal signal of the particle force. The errors in the force are less than 10% for all methods on all grids. Overall, the diffuse interface IBM (A) offers the least accurate and the overset grid method (O) the most accurate force predictions (only on the coarsest grid the L_2 error for method B is slightly lower than for method O). For all methods, the errors in the force appear to converge to zero upon grid refinement. The orders of convergence of the force are roughly one: they vary from 0.7 for the L_∞ error for method B to 1.2 for the errors for method O (Table 3).

Second, the errors in the velocity field are also less than 10% for all methods on all grids (Figs. 2c,d). From method A to B to C and finally to O, we observe a clear increase in accuracy. In both norms, the velocity is only first-order accurate for the IBM methods that solve the pressure inside the particle (methods A and B). For the IBM method that does not solve the pressure inside the particle (method C), the velocity displays second order accuracy in the L_2 norm. For method O, the velocity is second-order accurate in both norms (Table 3). First-order convergence of the velocity in the L_∞ norm in IBMs was reported before, for another diffuse-interface standard IBM [27] and another sharp-interface standard IBM [26].

Third, we discuss the errors in the pressure field (Figs. 3a,b). We observe that the errors in the pressure produced by the overset grid method are an order of magnitude lower than those produced by the IBM methods. All IBM methods suffer from large local errors in the pressure: the L_∞ error (maximum error in space and time) is larger than 20%. This error is largest for method A, while it does not reduce upon grid refinement, both for methods A and B. For (another) diffuse interface IBM, Stein et al. [27] reported similar behavior of the pressure (and velocity gradient), and here we find that sharp interface IBM (method B) also suffers from this problem. These methods solve the pressure inside the solid body. If the pressure equation is not solved inside the body (method C), the L_∞ error in the pressure is also large, although it reduces upon grid-refinement in this case, at least until the error just drops below 10% on the finest grid. In fact the maximum errors of the pressure are that large in methods A, B and C that the conclusion must be drawn that for reasonable resolutions ($\kappa \leq 2$), none of these IBMs is able to accurately predict the local pressure at all locations in the fluid at all times. On the other hand, the L_2 errors in the pressure do reduce upon grid refinement for all methods, but the convergence is relatively slow for methods A and B (orders of accuracy less than 1).

Fourth, we discuss the errors in the velocity gradient (Figs. 4a,b). Again the errors produced by the overset grid method are much lower than those produced by the IBMs, apart from the L_2 errors produced by method C, which are also quite low. For method A, both the L_∞ and L_2 percent errors in the velocity gradient are comparable to those in the pressure. For

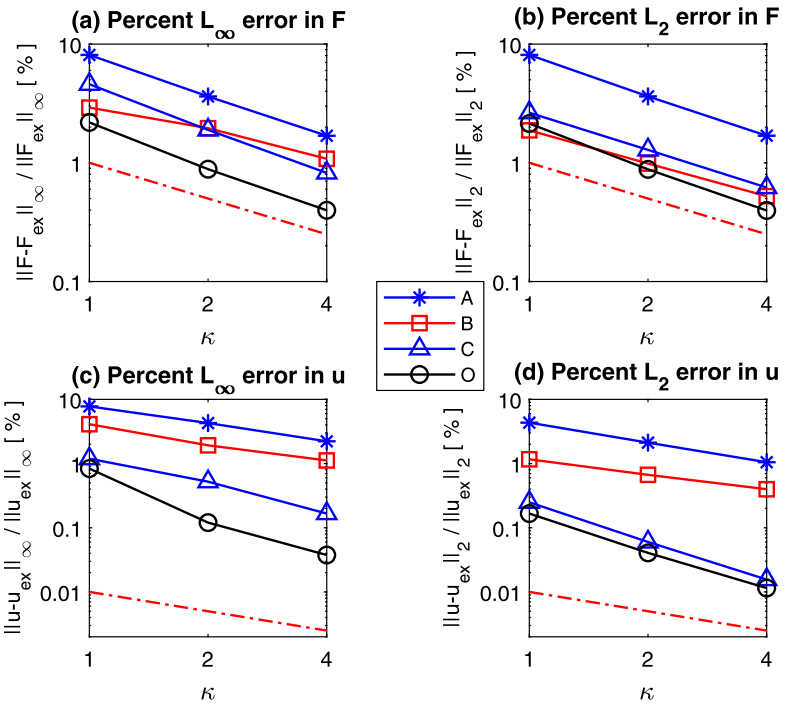

Fig. 2. Loglog plots of the L_∞ norms (left) and L_2 norms of the errors in the total force F on the sphere (top) and the velocity $\mathbf{u}$ (bottom) as functions of the resolution parameter κ for method A (stars), B (squares), C (large triangles) and O (circles). Slope of first-order convergence: red dash-dotted.

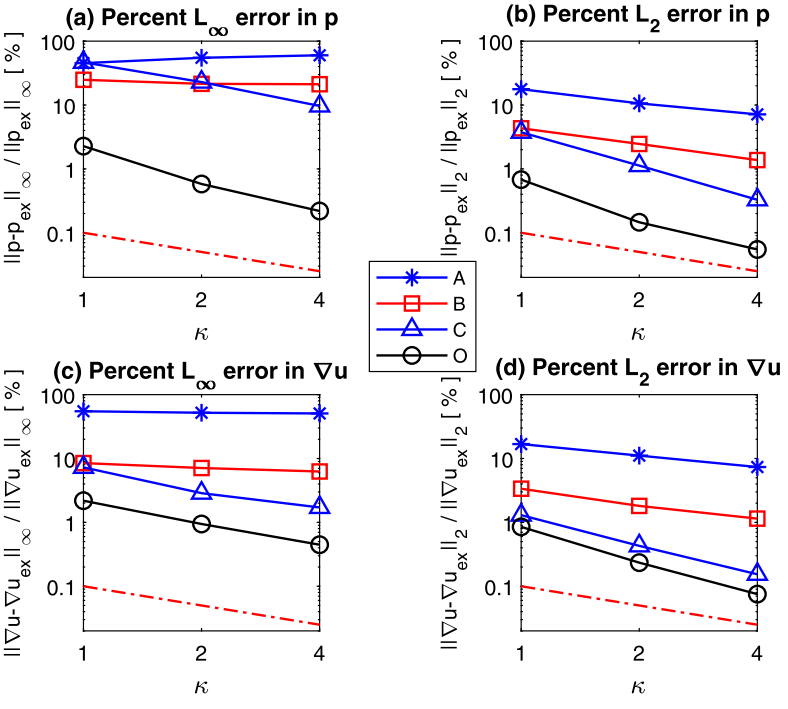

Fig. 3. Loglog plots of the L_∞ norms (left) and L_2 norms of the errors in the pressure (top) and the gradient of the velocity $\nabla\mathbf{u}$ (bottom) as functions of the resolution parameter κ for method A (stars), B (squares), C (large triangles) and O (circles). Slope of first-order convergence: red dash-dotted.

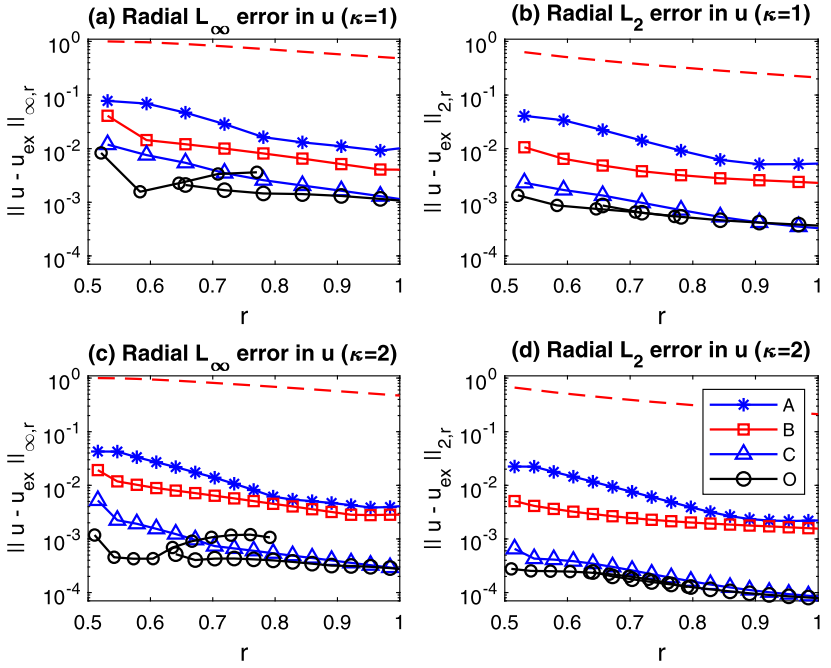

Fig. 4. Logarithmic plots of radial profiles of the error in the velocity: $\|\mathbf{u}-\mathbf{u}_{ex}\|_{\infty,r}$ (left) and $\|\mathbf{u}-\mathbf{u}_{ex}\|_{2,r}$ (right) for the resolutions $\kappa=1$ (top) and $\kappa=2$ (bottom) and for methods A (stars), B (squares), C (large triangles) and O (circles). The red-dashed lines represent $\|\mathbf{u}_{ex}\|_{\infty,r}$ (left) and $\|\mathbf{u}_{ex}\|_{2,r}$ (right).

methods B and C, we observe that the L_∞ errors in the velocity gradient are smaller than those in the pressure. However, the order of the convergence of the L_∞ error of the velocity gradient for method B is still close to zero. We conclude that, like the pressure, the gradient of the velocity is locally inconsistent both in diffuse and in sharp interface IBM projection methods that solve the pressure inside the solid domain without any particular treatment of the pressure at the boundary. The L_2 norms of the errors in the velocity gradient for methods B and C are much smaller than for method A. On the coarsest grid ($d_p=16/h$), the L_2 errors in $\nabla\mathbf{u}$ are 16.8%, 3.4%, 1.3% and 0.9% for methods A, B, C and O, respectively. In fact, the total dissipation rate (the volume and time averaged dissipation rate) is proportional to $\|\nabla\mathbf{u}\|_2^2$. Thus for $d_p/h=16$, a typical resolution for particle-resolved DNS of turbulent particle-laden flows, the relative errors in the total dissipation rate are approximately 34%, 7%, 3% and 2% for methods A, B, C and O, respectively.

It is remarked that most IBM studies of particle-resolved DNS in literature focus on predictions of the fluid velocity and the particle velocity and not on predictions of the pressure or fluid velocity gradient. Since the fluid velocity and particle force converge upon grid method for all four methods, it is expected that all four methods are able to provide reliable results for fluid velocity and particle velocity statistics in turbulent flows, provided the grid size is sufficiently small and ignoring the complications caused by light particles or inter-particle collisions. The results also show that for a given grid resolution, it matters which method is chosen, even if one is only interested in quantities that solely depend on the fluid velocity and the particle force.

4.3. Radial profiles of errors

It is expected that the local errors in the solution are relatively large in fluid cells close to the interface. The fact that most percent L_2 errors are considerably lower than the corresponding percent L_∞ errors, as observed in the previous section, supports this suggestion. To quantify the dependence of the local errors on the distance to the interface, we define spherical shells with radius r and a thickness of h centered around the particle center and $r=|\mathbf{x}-\mathbf{x}^p|$ as the distance to the particle center. Shell j is defined by the region $r_0+(j-1)h\le r<r_0+jh$. The shells have the same velocity as the particle. We respectively denote the radial L_∞ (maximum) norm by $\|\cdot\|_{\infty,r}$ and the radial L_2 (RMS) norm by $\|\cdot\|_{2,r}$. These norms are a function of r and for a given value of r, they are defined by the maximum and RMS taken over the time interval $[t_1,t_2]$ defined in subsection 3.1 and over all grid points inside a shell with radius r and thickness h. The midradius of the first shell is located at $r_0+h/2$. The precise definitions of the radial norms can be deduced from the definitions of the global norms given in section 4.2, by replacing of the entire volume Ω by the shell volume at radius r. Thus for the velocity vector or the velocity gradient tensor, the maximum and the sum are also taken over all elements of the vector or tensor.

The radial profiles of the errors in the velocity, pressure and velocity gradient are shown in Figs. 4, 5 and 6, respectively. For all radii shown ($r_0<r<2r_0=1$), the radial errors in the velocity and velocity gradient are much lower in case O than in cases A and B. Furthermore, the radial errors in the pressure are much lower in case O than in cases A, B and C. Each subplot

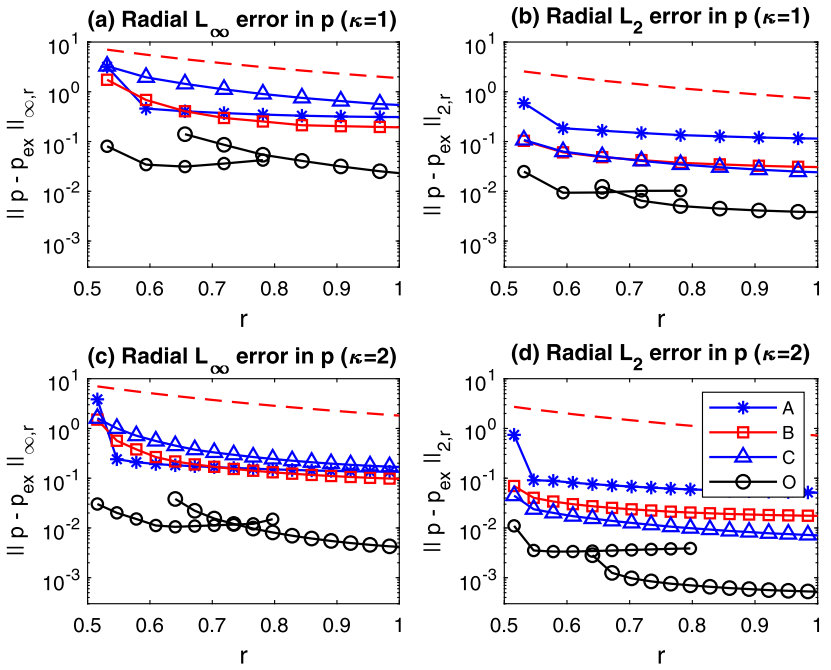

Fig. 5. Logarithmic plots of radial profiles of the error in the pressure: $\|p - p_{ex}\|_{\infty,r}$ (left) and $\|p - p_{ex}\|_{2,r}$ (right) for the resolutions $\kappa = 1$ (top) and $\kappa = 2$ (bottom) and for methods A (stars), B (squares), C (large triangles) and O (circles). The red-dashed lines represent $\|p_{ex}\|_{\infty,r}$ (left) and $\|p_{ex}\|_{2,r}$ (right).

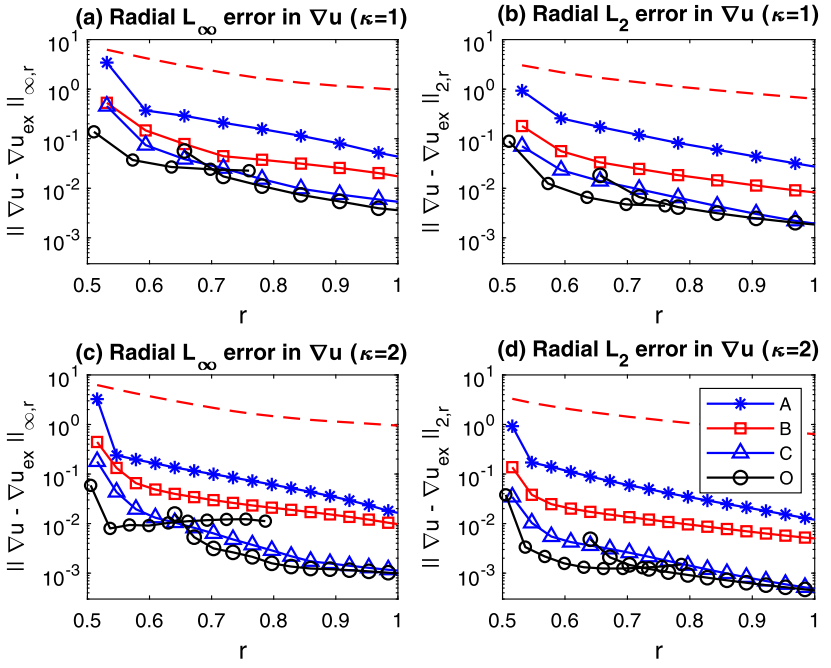

Fig. 6. Logarithmic plots of radial profiles of the error in the velocity gradient: $\|\nabla \mathbf{u} - \nabla \mathbf{u}_{ex}\|_{\infty,r}$ (left) and $\|\nabla \mathbf{u} - \nabla \mathbf{u}_{ex}\|_{2,r}$ (right) for the resolutions $\kappa = 1$ (top) and $\kappa = 2$ (bottom) and for methods A (stars), B (squares), C (large triangles) and O (circles). The red-dashed lines represent $\|\nabla \mathbf{u}_{ex}\|_{\infty,r}$ (left) and $\|\nabla \mathbf{u}_{ex}\|_{2,r}$ (right).

contains two profiles for the overset grid method (O), one based on grid points of the spherical grid ($r_0 < r < r_a \approx 0.81$) and another one based on the active grid points of the Cartesian background grid ($r > r_b \approx 0.63$). The symbols of the velocity and the velocity gradient shells on the spherical grid of method O have symbols left from the midpoints, because the average radius of the grid points where these quantities are defined in a given shell is smaller than the radius of the midpoint of

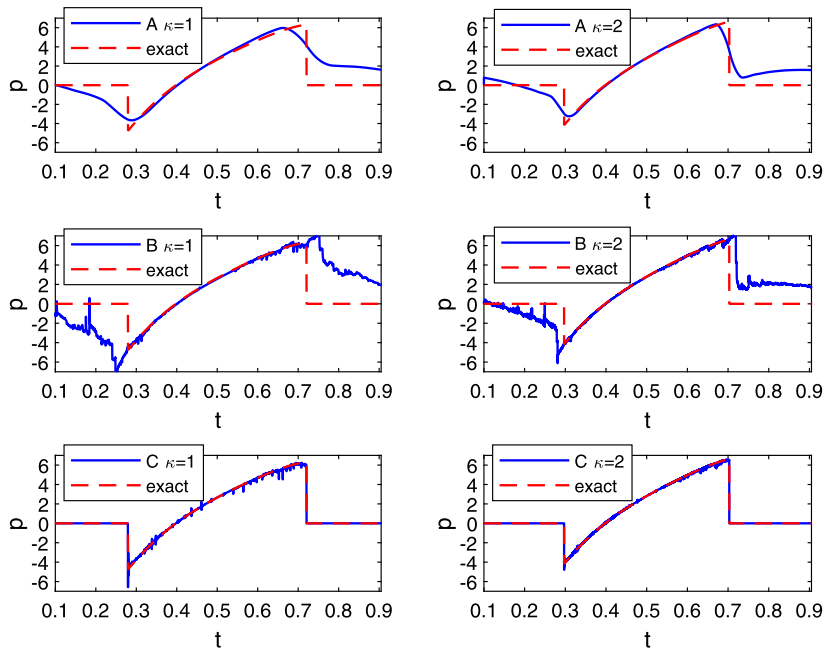

Fig. 7. Pressure as function of time at a fixed grid point ($\mathbf{x} \approx (0.47, 0, 0)$ on the axis of motion) for the resolutions $\kappa = 1$ (left) and $\kappa = 2$ (right) and for methods A (top), B (middle) and C (bottom). The red dashed curve is the exact pressure (zero inside the particle).

the shell. For example, the radial velocity component and the derivatives of the tangential velocity on the spherical grid of method O are available at $r = r_0$ and belong to the first shell.

For method A, the error in p and $\nabla \mathbf{u}$ in the first shell (the fluid region within a distance h to the particle surface, indicated by the first grid point of the radial profile) is much larger than in the grid points further away from the surface. In the first shell, both the radial L_∞ and radial L_2 errors in p and $\nabla \mathbf{u}$ do not reduce upon grid refinement in the first shell, but they do reduce upon grid refinement if $r > r_0 + h$. Thus the zeroth order convergence of p and $\nabla \mathbf{u}$ in method A and B seems to be limited to the first shell, which basically represents the fluid grid points that have at least one neighbor inside the solid.

It is quite remarkable that the radial L_∞ error in p in case C is larger than in cases A and B if $r > r_0 + h$, while the radial L_2 error in p in case C at least as small as in case B and much smaller than in case A. In the next subsection, we will show that the time signal of the pressure at fixed points displays spikes (sudden peaks rapidly decaying in time). These spikes occur at times when Cartesian grid points become active (move into the fluid) or passive (move into the solid). The slow radial decay of the L_∞ error in p in case C (Fig. 5a) indicates that this activation or passivation of grid points does not only affect the pressure in these grid points, but also the pressure further away from the interface. However, the duration of this influence is short on average, because the radial L_2 error in p in case C is much lower than in cases A and B (Fig. 5b).

4.4. Pressure signals in IBMs

We proceed to illustrate the behavior of the pressure in IBMs by plotting the pressure at a selected grid point on the axis of motion, a point where the interface passes during the simulation (Fig. 7). The grid point is inside the particle at $t = 0$ and the interface crosses the grid point at $t \approx 0.3$ and $t \approx 0.7$. Thus in between these times the grid point is inside the fluid. The exact time history of the pressure in this grid point is displayed by the red dashed line, which is discontinuous when the interface crosses the grid point. In the computation of the L_∞ and L_2 norms, the solution inside the particle was always excluded, but the pressure signals are also shown if the point is inside the particle. Only methods A and B compute a pressure inside the particle. For method C and for the exact solution, the value zero is plotted if the point is inside the particle. The figure shows that the pressure signals produced by methods A and B contain two jumps that look like discontinuities. Not surprisingly, the two jumps are smoother in case of method A than in case of method B, but they are sharpened by grid refinement in both cases. The jumps in case A are centered around the times when the grid point crosses the interface. In case B the jump occurs some time before the point enters the fluid or some time after the point leaves the fluid, probably because the IBM forcing in method B is applied inside the particle, on average at a distance $0.5h$ from the interface.

When the particle is inside the fluid, methods B and C produce noisy deviations from the exact pressure. In case C the noise is more spiky than in case B. The amplitude of the noisy error in case C seems to be somewhat larger in case B, but in case C the noise clearly reduces upon grid refinement. This is in line with the convergence behavior of the global maximum

Table 4
Average computation times (in seconds wall clock time) per time step and number of Poisson solver iterations per time step. For methods A, B, C and O, and resolution $\kappa = 2$. Percentages relative to the total computation time per time step are shown within brackets.

	Total	Velocity updates	Moving boundary	Poisson equation	Iterations
A	0.238 (100%)	0.018 (7.6%)	0.015 (6.3%)	0.205 (86.1%)	218
B	0.294 (100%)	0.018 (6.1%)	0.008 (2.7%)	0.268 (91.2%)	285
C	0.436 (100%)	0.018 (4.1%)	0.006 (1.4%)	0.410 (94.5%)	318
O	0.426 (100%)	0.018 (4.3%)	0.147 (34.4%)	0.261 (61.3%)	258

error of the pressure, $\|p - p_{ex}\|_\infty$ (although the selected grid point in this subsection is not the grid point where the global maximum is attained).

4.5. Computation times

Although the main goal of this work is the assessment of the accuracy rather than the computational efficiency of different methods, we also show computation times for each base method (Table 4). In this subsection, computation time means always computation time per time step. All values in Table 4 are averages over all time steps between $t = 0.1$ and $t = 0.6$, measured for the intermediate resolution ($\kappa = 2$), in simulations run on nine cores of one and the same Intel Xeon 2670v2 node of a small Linux cluster. For the overset grid code, which was parallelized with MPI and originally developed for flows with multiple spherical grids, it was convenient to use nine cores to simulate the present flow: eight cores for the Cartesian grid partitioned in $2 \times 2 \times 2$ blocks and a ninth core for the single spherical grid. In order to compare the computation times of the four base methods, the three IBM codes, which were parallelized using OpenMP, were also run on nine cores. The use of nine cores instead of a single core was observed to speed up each of these four simulations by a factor of approximately six.

The total computation time was split into three contributions: the time for the velocity updates (evaluations of the convective, viscous and pressure gradient terms), the time for the moving boundary machinery (evaluations of regularized delta functions, extrapolations, interpolations) and the time for the iterative solution of the pressure Poisson equation. The time used to compute the force on the sphere was negligible (less than 0.1% of the total). The average number of iterations needed to obtain a converged solution of the Poisson equation within the prescribed tolerance (10^{-7}) is also shown.

According to Table 4, the total computation time was lowest for method A, followed by method B, while it was substantially higher for methods C and O. For each method, solving the Poisson equation required most of the computation time, but the average number of iterations varied per method. The discrete Poisson equation in methods A and B has the same simple structure in every grid cell of Ω_c, so that optimization of the Poisson computation time per iteration was more effective for these methods than for methods C and O. For flow cases that are (unlike the present case) defined in a periodic domain, the costs of methods A and B can be lowered further, by replacing the iterative Poisson solver by a direct FFT solver in methods A and B.

The computation time required for the moving boundary was relatively large for method O, which was mainly due to the interpolation of the pressure on overset boundaries in each iteration of the Poisson equation. The time required for these pressure interpolations (0.105 s per timestep) was not allocated to the Poisson equation but to the moving boundary. The loops in the moving boundary machinery routines can be subdivided into two groups: loops over a subset of grid points (group 1) and loops over all grid points (group 2). Group 1 represents the computation of the nonzero values of the immersed boundary force $\mathbf{f}$ in method A, the computations of the velocity at ghost points in methods B and C, and the computations of the interpolation weight coefficients and the interpolations of the velocity and pressure in method O. Group 2 represents the reinitialization of the immersed boundary forcing term $\mathbf{f}$ and the addition of $\Delta t\ \mathbf{f}$ to $\mathbf{u}^*$ in method A, the identification of ghost points in methods B and C, and the identification of the interpolation points and corresponding interpolation stencils in method O. The number of grid points used in group 1 scales with κ^2, while the number of grid points in group 2 (all grid points) scales with κ^3. Thus the computation time required for the moving boundary is expected to scale with κ^3 if κ is large. However, at finite κ, the work done in group 1 can still be larger than the work done in group 2. At $\kappa = 2$, this was found to be the case for methods A and O (but not for methods B and C). As a consequence, when the grid was refined by increasing κ from 2 to 4, the ratio of the computation time required for the moving boundary to the computation time for the velocity updates was reduced by a factor of 1.9 for method A and by a factor of 1.6 for method O.

5. Comparison of artificial compressibility with projection methods

The reason to include two artificial compressibility variants (methods C′ and method O′) is twofold. First, the L_∞ error in the pressure was found to be very large for all three base IBMs (A, B and C), while it did not converge to zero upon grid refinement if the pressure was solved inside the particle (methods A and B). The finite speed of sound in an artificial compressibility method gives the pressure more flexibility and might therefore reduce spikes on the pressure signals, which are apparently the cause of the large L_∞ error in the pressure in case of method C. Second, if we use an overset grid method,

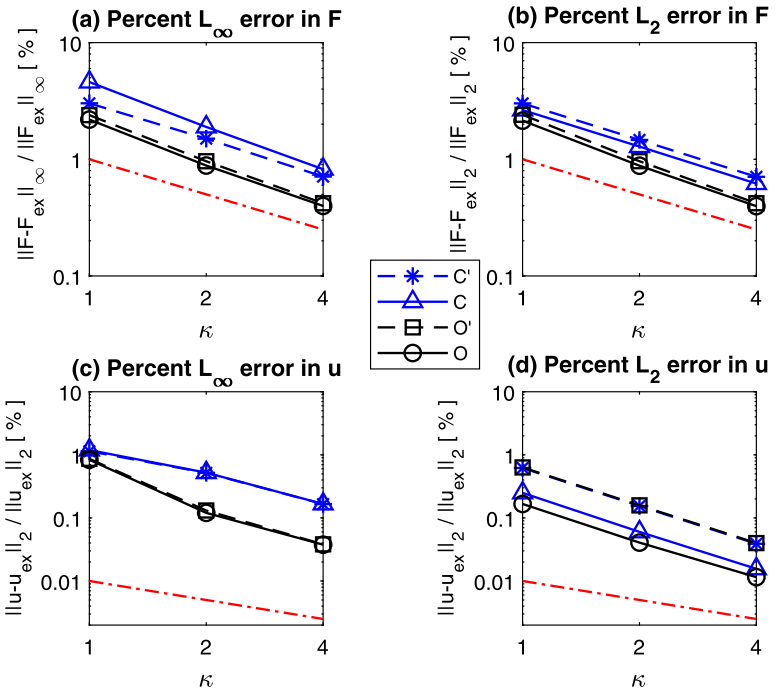

Fig. 8. Loglog plots of the L_∞ norms (left) and L_2 norms of the errors in the total force F on the sphere (top) and the velocity $\mathbf{u}$ (bottom) as functions of the resolution parameter κ for method C′ (squares), C (triangles), O′ (stars) and O (circles). Slope of first-order convergence: red dash-dotted.

or if we resort to method C to decouple the pressure at the interface, a direct FFT solver cannot be used for the pressure Poisson equation, even not if periodic boundary conditions at the outer boundary $\partial\Omega_c$ are used. For the same number of grid points, an iterative pressure solver is much slower than a FFT pressure solver. An artificial compressibility method can be a very efficient alternative to an iterative pressure method, especially if the time step is already small for other reasons and sufficiently small to allow a low artificial Mach number. In the present case, the average computation time per time step was found to be about 16 times smaller for method C′ compared to method C and about 6 times smaller for method O′ compared to method O.

Figs. 8–10 show the percent error norms for artificial compressibility methods C′ and O′. The results for the projection methods C and O are also included, so that the artificial compressibility methods can directly be compared with their projection counterparts. The numerical values of the errors and estimates of the order of the convergence from the slopes of the loglog plots can be found in Tables 2 and 3 in subsection 4.2.

First we discuss the IBM results (the comparison between methods C′ and C). The notable effects of the replacement of projection by artificial compressibility are (1) a smaller L_∞ error in the force (Fig. 8a, compare stars to triangles), (2) a larger (but still small) L_∞ error in the velocity (Fig. 8d), (3) a much smaller L_∞ and a smaller L_2 error in the pressure (Figs. 9a,b), (4) a somewhat larger L_∞ error in the velocity gradient (Fig. 9c), and (5) much larger errors in the divergence of the velocity (Fig. 10).

The main benefit of artificial compressibility appears to be a much smaller L_∞ error in the pressure. In both cases, C′ and C, the convergence of this error is roughly first order, but the smaller error on the finest grid ($\kappa = 4$) in case C′ (2.7% compared to 9.7% in case C) makes the convergence behavior more convincing in case C′. The price to be paid is a significant error in the velocity divergence due to the artificial compressibility (Fig. 10). The maximum (L_∞ norm) of the velocity divergence is 7.7% of the maximum velocity gradient, which is not small, but it does converge to zero upon grid refinement.

For the overset grid method the effects of the replacement of the projection scheme by an artificial compressibility scheme are less prominent than for the IBM. For the overset grid method we observe that artificial compressibility instead of projection leads to: (1) larger (but still small) L_2 errors in the velocity (Fig. 8d, compare squares to circles), (2) a smaller L_∞ error in the pressure on the coarsest grid (Fig. 9a, $\kappa = 1$), (3) larger (but still small) L_2 errors in the pressure (Fig. 9b), and (4) larger (but still small) L_2 norms of the velocity divergence. Furthermore, the L_2 error in the velocity in case C′ appears to be the same as in case O′, and the same observation applies to the error in the velocity divergence. We attribute this similarity to the finite speed of sound, which is also the same for both methods. The finite speed of sound introduces errors at all times and all locations. This is reflected by the global L_2 errors of the velocity and the velocity divergence and confirmed by the radial profiles of the errors discussed later on.

All four methods C′, C, O′ and O have a nonzero divergence. Thus at finite grid size, these methods permit a finite error in the discrete local and global mass conservation laws. This is formally not a problem, since in each case the both the L_∞ and

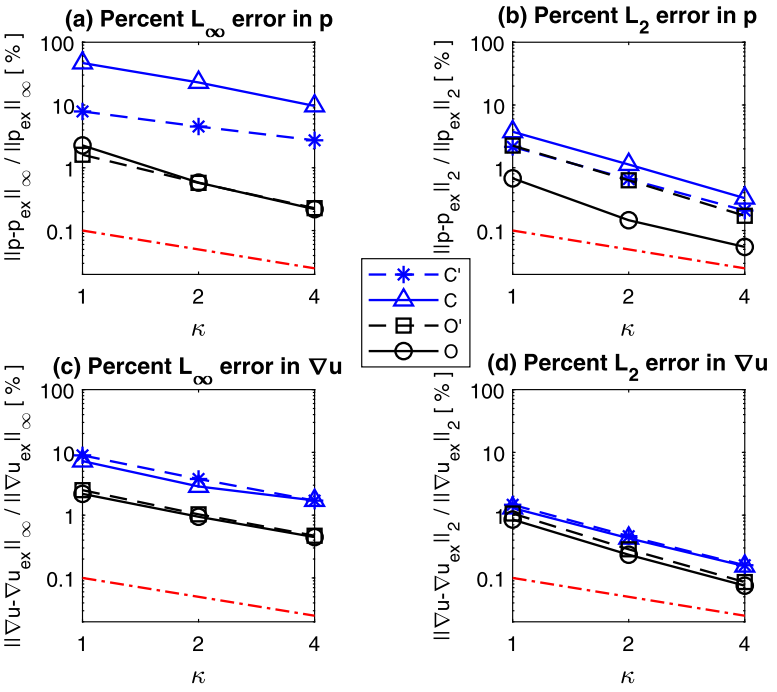

Fig. 9. Loglog plots of the L_∞ norms (left) and L_2 norms of the errors in the pressure (top) and the gradient of the velocity $\nabla\mathbf{u}$ (bottom) as functions of the resolution parameter κ for method C′ (squares), C (triangles), O′ (stars) and O (circles). Slope of first-order convergence: red dash-dotted.

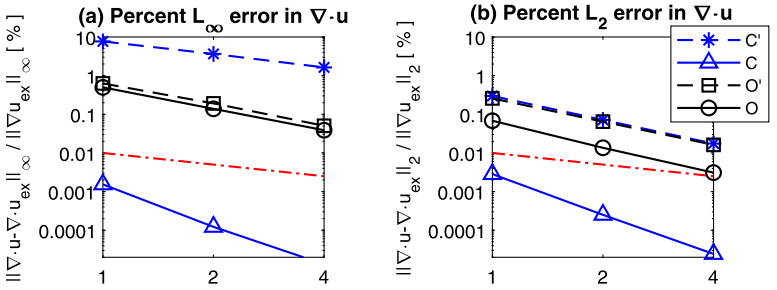

Fig. 10. Loglog plots of the L_∞ norms (left) and L_2 norms of the divergence of the velocity as functions of the resolution parameter κ for method C′ (squares), C (triangles), O′ (stars) and O (circles). Slope of first-order convergence: red dash-dotted.

L_2 norms converge to zero upon grid refinement. In other words, the mass conservation errors are just discretization errors that have hereby been shown to converge to zero upon grid refinement. Thus the divergence is still consistent. Whether the mass conservation errors are a problem in practice depends on the actual size of the errors and the application. The magnitude of the divergence errors depends on the method. They are rather large for method C′, small for method O′, even smaller for method O and negligible for method C. In methods A and B, the discrete velocity divergence is zero (less than the tolerance of the pressure solver), but the pressure and the velocity gradient in these methods are inconsistent since they do not converge to the analytical solution everywhere in the fluid domain. From a fundamental point of view, a method that generates a consistent velocity gradient and pressure and a nonzero but consistent discrete velocity divergence is more satisfying than a method that generates an inconsistent velocity gradient and pressure but a zero discrete velocity divergence.

The nonzero discrete divergence in the projection methods C and O is related to the nonzero value of the augmented variable b in the pressure Poisson equation. This variable was found to converge to zero upon grid refinement. In case C, the results for $\|b\|_\infty$ were 0.24, 0.076 and 0.03 if $\kappa = 1$, 2 and 4, respectively. In case O, these values were 0.043, 0.0087 and 0.002. In both cases, $\|b\|_2$ was 2 to 5 times smaller than $\|b\|_\infty$.

Figs. 11 and 12 show the radial profiles of the errors (for brevity only for $\kappa = 1$). The radial profile of the maximum error in the pressure (Fig. 11c) shows that due to the finite speed of sound the maximum error in case C′ is not only strongly reduced compared to case C, but its decay over the first few radial grid points is also relatively strong. Furthermore, the increase of the error in the velocity due to artificial compressibility is not limited to the direct vicinity of the particle but continues until at least $r = 2r_0 = 1$ (Figs. 11a,b). The increase of this error is apparently connected to the increased the

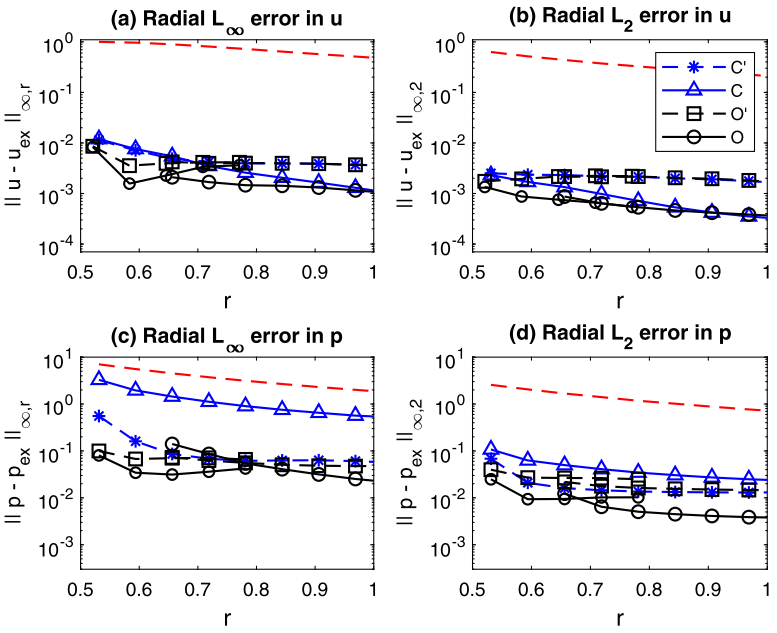

Fig. 11. Logarithmic plots of the radial L_∞ error (left) and radial L_2 error (right) in the velocity (top) and the pressure (bottom) for the resolution $\kappa = 1$ and for methods C′ (squares), C (triangles), O′ (stars) and O (circles). The red-dashed lines represent the radial L_∞ norm (left) and the radial L_2 norm (right) of the exact velocity (top) and the exact pressure (bottom).

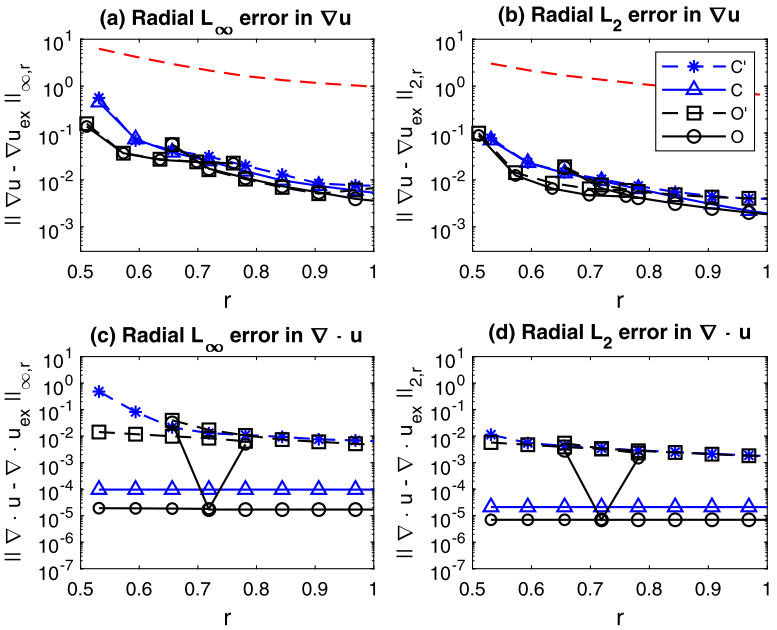

Fig. 12. Logarithmic plots of the radial L_∞ error (left) and radial L_2 error (right) in the velocity gradient (top) and the velocity divergence (bottom) for the resolution $\kappa = 1$ and for methods C′ (squares), C (triangles), O′ (stars) and O (circles). The red-dashed lines represent the radial L_∞ norm (left) and the radial L_2 norm (right) of the exact velocity gradient (top) and the exact velocity divergence (bottom).

velocity divergence, which also continues until at least $r = 2r_0 = 1$ (Figs. 12c,d). With respect to the projection methods, Figs. 12c,d show that the nonzero velocity divergence for overset grid method O is concentrated in the first shell of the Cartesian grid and the last shell of the spherical grid. These shells contain the grid points where the discrete divergence operator uses one or more interpolated velocity values.

In Fig. 13a, the time signal of the pressure at a fixed grid point is shown for method C′ ($\kappa = 1$). The fixed grid point is the same as in section 4.4. We observe that due to artificial compressibility the spikes in the predicted pressure signal

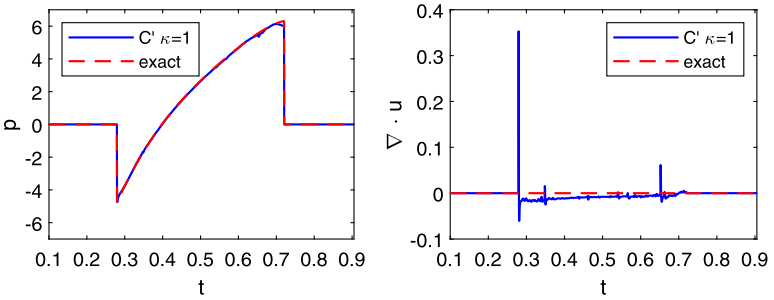

Fig. 13. Pressure (left) and velocity divergence (right) as function of time at a fixed grid point ($\mathbf{x} \approx (0.47, 0, 0)$ on the axis of motion) for the resolution $\kappa = 1$ for method C′. The red dashed curves denote the exact pressure (left) and exact velocity divergence, which is zero (right).

disappear (compare the signal for method C and $\kappa = 1$ in Fig. 7). Unfortunately, spikes in the velocity divergence emerge instead (Fig. 13b). Spikes like these lead to the rather large global L_∞ norm of the divergence in case C′ (7.7% for $\kappa = 1$).

6. Discussion of other variants of the base methods

Not only the artificial compressibility variants of methods C and O, but many more variants were tested. It is useful to discuss these other variants too, since they show how the results change if a retracted interface, an incremental projection method, an implicit method, or a linear extrapolation or interpolation method is used. For one resolution ($\kappa = 1$), brief descriptions and norms of errors of variants are presented in Table 5 (classes A and B) and Table 6 (classes C and O). The description of the variant briefly indicates the one or two features that distinguish the variant from the base method. The effect of these elements can be deduced by direct comparison of the results of the variant with the results of the base method. The base method is shown as first method in each class.

6.1. Variants in classes A and B

The variants of the diffuse-interface IBMs (class A) illustrate the effects of incremental projection versus nonincremental projection, explicit Runge-Kutta versus explicit Euler time-stepping and interface retraction. All base methods use the so-called nonincremental projection scheme, in which no pressure gradient term in the equation for the intermediate velocity appears. In the so-called incremental projection scheme, $-\nabla p^n$ is included into the equation for the intermediate velocity. Then a Poisson equation for the pressure increment $p' = p^{n+1} - p^n$ is solved, and $-\nabla p'$ is used to obtain $\mathbf{u}^{n+1}$, while p^{n+1} is obtained by adding p' to p^n. The results indicate that the improvements caused by the change to incremental projection are small (case A1). The variant with the second-order three stage Runge-Kutta method with coefficients proposed by Jameson and Baker [52,48], with the same time step and with immersed boundary forcing and incremental projection in each stage, also produces similar results (case A2).

The diffuse-interface IBM can be improved if the interface of the particle is retracted [13,53,54]. In variant A3, the interface is retracted by locating the Lagrangian points at distance $r_0 - 0.3h$ from the center of the sphere. The retraction distance $0.3h$ is an empirical value, recommended by Breugem [13]. We observe that interface retraction reduces the error in the force by almost a factor 3, while also the errors in the velocity, pressure and gradient of the velocity are reduced, another indication that these errors are caused by the grid points near the interface (see subsection 4.3). It is remarked that all grid points with a distance less to than $0.3h$ from the retracted surface inside the body were excluded from the norm computations. A grid refinement study of case A3 (not included in the table) indicated that interface retraction does not improve the order convergence of the pressure and the gradient of the velocity; the observed order remained approximately zero in the L_∞ norm and less than one in the L_2 norm. In addition to interface retraction, Breugem [13] adopted an alternative procedure to compute the particle force, as developed by Kempe & Fröhlich [12] for freely moving particles with particle fluid density ratio smaller than 1.2, for which the original method is unstable [8]. However, for the present flow, in which the particle motion is prescribed, the alternative procedure was found to strongly increase the L_∞ and L_2 errors in the particle force (to 42.2% and 13.7%, respectively, due to spikes on the force signal).

In class B, the incremental projection method leads to a strong increase of most errors (case B1). A sharp-interface method applied to a moving body problem leads to sharp pressure discontinuities in space and in time (see section 4.4), i.e. a nondifferentiable pressure, implying that $\nabla p'$ is not necessarily smaller than ∇p^{n+1}, even not upon grid refinement. Therefore, the use of p' deteriorates the results.

The extrapolation scheme in base methods B and C is quadratic for forcing points not too far inside the particle ($\beta < \beta_1 \leq 1/2$), see section 3.2. In the same section the weights of a linear version of this extrapolation scheme were specified. The linear scheme produces somewhat less accurate results (case B2) than the default quadratic scheme (case B) for all variables, except the pressure, which has become slightly more accurate.

In the next variant, B3, a linear extrapolation not aligned with a Cartesian coordinate direction, but aligned with the normal vector of the surface is used. It is based on two velocities: the velocity at the intersection point of extrapolation

Table 5
Results of all methods in classes A and B for $\kappa = 1$.

Method		Relative L_∞ errors [%]					Relative L_2 errors [%]				
		F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$	F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$
A	base	8.11	7.77	45.3	54.9	0	8.11	4.34	17.7	16.8	0
A1	incr. proj.	8.84	7.93	46.7	54.0	0	8.86	4.50	19.3	16.6	0
A2	incr. proj. & RK	8.86	7.93	44.4	55.4	0	8.87	4.59	18.8	17.4	0
A3	retracted interface	2.95	4.79	29.8	38.1	0	2.95	2.19	8.68	10.2	0
B	base	2.92	4.10	24.8	8.48	0	1.88	1.17	4.32	3.36	0
B1	incr. proj.	5.83	14.4	72.8	38.0	0	1.98	0.79	7.45	3.99	0
B2	lin. extr.	3.60	4.41	23.3	8.45	0	2.49	1.55	5.23	3.81	0
B3	distant image	4.35	4.83	17.2	8.54	0	3.28	1.95	6.69	4.26	0

Table 6
Results of all methods in classes C and O for $\kappa = 1$.

Method		Relative L_∞ errors [%]					Relative L_2 errors [%]				
		F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$	F_1^p	$\mathbf{u}$	p	$\nabla\mathbf{u}$	$\nabla\cdot\mathbf{u}$
C	base	4.59	1.19	46.7	7.27	0.00	2.64	0.25	3.74	1.30	0.00
C1	lin. extr.	5.41	1.95	72.2	9.56	0.00	2.90	0.43	6.10	2.80	0.01
C2	const. extr.	7.10	8.29	90.5	38.3	0	4.37	2.13	13.0	14.1	0
C3	$\mathbf{u}^*$ correction at Γ_{ss}	4.54	1.19	44.5	7.28	0	2.64	0.25	3.35	1.30	0
C5	impl. $\mathbf{u}$	4.51	1.47	64.1	8.58	0.00	2.13	0.38	4.87	1.85	0.01
C6	impl. $\mathbf{u}$ & incr. proj.	10.7	2.10	168	10.8	0.00	2.83	0.25	11.3	1.30	0.00
C7	impl. $\mathbf{u}$ & $\beta = 0.99$	4.21	1.37	65.9	7.91	0.00	2.01	0.31	4.60	1.52	0.01
C′	art. compr.	3.02	1.14	7.89	8.97	7.69	3.01	0.62	2.15	1.46	0.30
O	base	2.19	0.84	2.27	2.18	0.50	2.14	0.17	0.68	0.85	0.07
O1	incr. proj.	2.12	0.95	5.38	2.16	0.91	2.07	0.27	3.04	0.97	0.03
O2	lin. interp.	2.11	0.87	3.70	2.24	0.83	2.11	0.42	1.87	1.30	0.25
O′	art. compr.	2.39	0.86	1.61	2.50	0.63	2.41	0.63	2.26	1.06	0.26

line and surface and the velocity at a so-called image point in the fluid at distance h from the surface. We call this method the distant image method. It is very similar to mirror image methods proposed in literature [9,10], with the difference that in method B3 all image points have the same distance to the surface, while in the mirror image method the distance from image point to surface is the same as the distance from extrapolation point to surface. The velocity at the image point is obtained by linear interpolation using the 2^3 grid points surrounding the image points. Since some stencil points can also be forcing points, an iterative forcing procedure is required. No more than 5 (Picard) iterations were typically required to let the extrapolated velocities to a residual less than 10^{-10}. The results produced by the distant image method (B3) are not very different from those produced by the linear one-dimensional extrapolation scheme (B2), slightly worse, apart from the L_∞ error in the pressure, which is the lowest in this class. Furthermore, the mirror image method, mentioned above, was also tested and found to produce much larger pressure errors than the distant image method B3 and base method B. We conclude that, compared to extrapolation along one of the Cartesian coordinate directions, extrapolation along the local normal vector of the boundary does not significantly increase the overall accuracy of the method.

6.2. Other variants in classes C and O

An important difference between classes A and B on the one hand (Table 5) and classes C and O on the other hand (Table 6) is that the methods in the latter classes solve no velocity and pressure fields inside the body. First we discuss the sharp-interface DIBM variants in class C.

The base method in class C clearly benefits from quadratic extrapolation since all errors are smaller than the corresponding errors produced by method with the linear version of the extrapolation scheme (case C1). On its turn the linear version performs much better results than constant extrapolation (case C2), which means that the velocity at all ghost points is set equal to the velocity of the body at time level $n+1$. Method C2 is a classical stair-step method. A formal advantage of this variant is local mass conservation satisfied by the velocity at the end of the time step. In contrast, the discrete velocity divergence in method C is not zero by construction, although it appears to be very small and to converge to zero upon grid refinement.

In variant C3, the divergence in the velocity is constructed to be zero in the following way. The method is the same as method C, except that just before $\nabla\cdot\mathbf{u}^*$ is computed, a correction is applied to $\mathbf{u}^*$ at the faces of the stair-step surface Γ_{ss}, represented by the dashed line in Fig. 1b. Only the ghost velocities at these faces are corrected since the other ghost velocities are not needed to compute $\nabla\cdot\mathbf{u}^*$. The absolute value of the correction is the same for each cell and follows from the constraint that after correction the total mass flux through Γ_{ss} is zero. The results show that method C3 is slightly but not significantly more accurate than method C, confirming that the main errors in case C are not caused by the absence of

strict global mass conservation. It is remarked that Mark & van Wachem [10] ensured strict local mass conservation in a DIBM by setting the ghost values of the intermediate velocity equal to the velocity of the object, just before the divergence was taken and not at all ghost points but only at those needed to compute the discrete divergence in fluid cells.

Furthermore, variants were implemented in which the explicit Euler scheme for $\mathbf{u}^*$ was replaced by the implicit Euler scheme for the viscous terms. In these cases, the extrapolation procedure for the velocity at the forcing points was included by adding the extrapolations equations to the matrix of the implicit system for $\mathbf{u}^*$. The reason to implement this was to obtain a stable simulation for a value of $\beta_1 \approx 1$, since then nearly all extrapolations are performed with the quadratic leg of the scheme.

The tests with implicit variants indicate that the implicit scheme by itself does not improve the results, except for the L_2 error in the particle force (case C4), that combination of the implicit scheme with incremental projection deteriorates the results, the pressure in particular (case C5), while combination with $\beta_1 = 0.99$ (case C6) does not lead to a method with an overall better accuracy than the base method. It is remarked that these conclusions apply to a moving body problem. For flows with nonmoving boundaries, incremental projection is known to reduce the splitting errors induced by the implicit terms [22]. Especially for steady state flow past a nonmoving body, an implicit method with incremental projection is expected to perform better than a nonincremental one, because in the steady state the pressure increment becomes zero, reducing the splitting errors to zero too. It was verified that the results did not change when the nonincremental projection in base method C is replaced by incremental projection without any other change. This was expected, because $\mathbf{u}^*$ in method C is fully explicit and the forcing is applied not after but before the update of $\mathbf{u}^*$.

For the class of overset grid methods (Table 6), we find that incremental projection (case O1) produces a less accurate pressure than the nonincremental projection (case O), while other errors are comparable. Furthermore, the replacement of the triquadratic interpolations on stencils of 27^3 points (method O) by trilinear interpolation schemes on stencils of 2^3 points (method O2) makes the method somewhat less accurate, but all errors still remain below 4% on the coarsest grid ($\kappa = 1$), which is not achieved by any of the IBMs tested. Furthermore, a variant was tested in which the Poisson matrix was not augmented before but after scaling of the diagonal, as in [36]. Up to all decimals reported in Table 6, the results of that variant were the same as those of method O.

7. Conclusions

A systematic comparison of immersed boundary methods (IBMs) and overset grid methods was performed for Stokes flow due to an oscillating sphere. The three base IBMs were: (A) a diffuse-interface immersed boundary method, (B) a sharp-interface IBM with standard treatment of the pressure, and (C) a sharp-interface IBM with pressure decoupling at interfaces. In contrast to methods A and B, no pressure field was solved inside the sphere in method C. In methods B and C, an explicit hybrid extrapolation scheme was introduced, third-order accurate for extrapolation points no further than $h/2$ inside the object and second-order accurate otherwise. The fourth base method was an overset grid method (O), which unlike the IBMs used a body-fitted mesh around the sphere. All four methods were based on the (nonincremental) projection method.

For each method, the error between the numerical solution and the analytical solution was computed and reported in terms of L_∞ (maximum) and L_2 (RMS) norms of the errors in the velocity, the pressure and the velocity gradient fields. Results were shown for three levels of grid refinement, $\kappa = 1$, 2 and 4 ($h = d_p/16$, $d_p/32$ and $d_p/64$). In this assessment, the overset grid method clearly outperformed the IBMs. It was the only method that was able to produce errors down to one percent in the maximum norm for 32 points per particle diameter ($\kappa = 2$). The results suggested the following ranking of the methods with respect to their overall performance in terms of accuracy: O (most accurate), C, B, A. This order was reflected by the errors in the L_2 norms in $\mathbf{u}$, p and $\nabla\mathbf{u}$.

Upon grid refinement, the velocity gradient and pressure fields produced by method A and B did not converge to the analytical solution in the maximum norm, but the total force on the sphere did converge in the maximum norm. Furthermore, an investigation of radial norms of errors, norms as function of the distance to the particle center, showed that the lack of convergence of the velocity gradient and the pressure in the maximum norm in case of methods A and B was caused by the first layer of grid cells around the object. When the pressure Poisson equation was not solved inside the solid domain (method C), the velocity gradient and pressure seemed to converge in the maximum norm, but high resolution was required to achieve a maximum error in the pressure that was smaller than in method A and B. Thus, in fact no IBM based on the projection method was found to be able to predict the pressure with a small maximum error at acceptable grid sizes.

For flows with periodic boundary conditions, the pressure in methods A and B can be solved using FFT since the discrete Laplacian operator is then the same at every point. FFT is unfortunately no option for methods C and O, which is a practical disadvantage of these methods. However, no iterative pressure solver in methods C and O is needed if the projection method is replaced by an artificial compressibility method. This introduces other errors in the solution that depend on a additional parameter, the speed of sound, but these errors are expected to vanish in the limit of zero Mach number. Two staggered artificial compressibility variants were proposed, IBM method C′ and overset grid method O′. The speed of sound was made proportional to $1/h$ and the Mach number was only 0.02 for the largest value of h used.

The main effects of artificial compressibility were a lower error in the pressure and a larger error in the velocity divergence. The effect was very strong in method C′ and to much smaller extent also present in method O′. While the maximum errors in O′ were clearly smaller than in C′, C′ was the only IBM method that produced less than 10% maximum error in

the pressure for the coarsest grid and less than 3% for the finest grid. The price to pay was that the maximum error in the velocity divergence error was 8% of the maximum velocity gradient on the coarsest grid and 2% on the finest grid. It was found that artificial compressibility successfully removed the spikes in the pressure in the IBM case but introduced them in the velocity divergence. Furthermore, the maximum error in the total particle force in method C′ was noticeably smaller than in method C. Overall, we cautiously conclude that method C′ was actually the best performing IBM method in this assessment.

In tests with other variants, it was found that incremental instead of nonincremental projection can deteriorate the results (shown for methods B and O, and for an implicit variant of method C). Furthermore, the errors of the diffuse-interface IBM were significantly reduced when the interface was retracted using the empirically determined retraction distance from literature. However, it did not make the performance of the diffuse-interface method better than the other base methods, with respect to overall accuracy, and it did not result in convergence of the velocity gradient and pressure in the maximum norm. It is remarked that (some) results of the other types of methods could probably be also improved if the interface is shifted or another empirical factor proportional to the grid size is introduced. Finally, the sharp-interface IBMs methods (B and C) with their explicit hybrid extrapolation scheme (partially quadratic) were compared to variants with fully linear extrapolation schemes. Also overset grid method O using triquadratic interpolations was compared to a variant using trilinear interpolations. For methods C and O, a clear positive effect of the (partially) quadratic extrapolations and interpolations was noticed.

CRediT authorship contribution statement

A.W. Vreman wrote the computer programs, performed the simulations, analyzed the results and wrote the article.

Declaration of competing interest

The author declares that he has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 239–261.
[2] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271.
[3] D. Goldstein, R. Handler, L. Sirovich, Modeling a no-slip flow boundary with an external force field, J. Comput. Phys. 105 (1993) 354–366.
[4] J. Mohd-Yusof, Combined immersed-boundary/B-spline methods for simulations of flow in complex geometries, in: CTR Annual Research Briefs, NASA Ames/Stanford Univ., 1997, pp. 317–327.
[5] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35–60.
[6] J. Kim, D. Kim, H. Choi, An immersed-boundary finite volume method for simulations of flow in complex geometries, J. Comput. Phys. 171 (2001) 132–150.
[7] Y. Tseng, J.H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2003) 593–623.
[8] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys. 209 (2005) 448–476.
[9] R. Mittal, H. Dong, M. Bozkurttas, F.M. Najjar, A. Vargas, A. von Loebbecke, A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries, J. Comput. Phys. 227 (2008) 4825–4852.
[10] 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.
[11] 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.
[12] 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.
[13] 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.
[14] 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.
[15] V. Spandan, V. Meschini, R. Ostiall-Mónico, D. Lohse, G. Querzoli, M.D. de Tullio, R. Verzicco, A parallel interaction potential approach coupled with the immersed boundary method for fully resolved simulations of deformable interfaces and membranes, J. Comput. Phys. 348 (2017) 567–590.
[16] M. Uhlmann, First experiments with the simulation of particulate flows, Technical Report No. 1020, CIEMAT, Madrid, 2003.
[17] M. Uhlmann, Interface-resolved direct numerical simulation of vertical particulate channel flow in the turbulent regime, Phys. Fluids 20 (2008) 053305.
[18] P. Costa, F. Picano, L. Brandt, W.P. Breugem, Universal scaling laws for dense particle suspensions in turbulent wall-bounded flows, Phys. Rev. Lett. 117 (2016) 134501.
[19] A.J. Chorin, Numerical solution of the Navier-Stokes equations, Math. Comput. 22 (1968) 745–762.
[20] R. Temam, Sur 1′ approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires I, Arch. Ration. Mech. Anal. 32 (1969) 135–153.
[21] A.J. Chorin, A numerical method for solving incompressible flow problems, J. Comput. Phys. 2 (1967) 12–26.
[22] J.L. Guermond, P. Minev, J. Shen, An overview of projection methods for incompressible flows, Comput. Methods Appl. Mech. Eng. 195 (2006) 6011–6045.
[23] A. Gilmanov, F. Sotiropoulos, A hybrid Cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies, J. Comput. Phys. 207 (2005) 457–492.
[24] M.P. Kirkpatrick, S.W. Armfield, J.H. Kent, A representation of curved boundaries for the solution of the Navier-Stokes equations on a staggered three-dimensional Cartesian grid, J. Comput. Phys. 184 (2003) 1–36.

[25] L. Schneiders, C. Günther, M. Meinke, W. Schröder, An efficient conservative cut-cell method for rigid bodies interacting with viscous compressible flows, J. Comput. Phys. 311 (2016) 62–86.
[26] R.D. Guy, D.A. Hartenstine, On the accuracy of direct forcing immersed boundary methods with projection methods, J. Comput. Phys. 229 (2010) 2479–2496.
[27] D.B. Stein, R.D. Guy, B. Thomases, Immersed boundary smooth extension (IBSE): a high-order method for solving incompressible flows in arbitrary smooth domains, J. Comput. Phys. 335 (2017) 155–178.
[28] J. Lee, J. Kim, H. Choi, K.-S. Yang, Sources of spurious force oscillations from an immersed boundary method for moving-body problems, J. Comput. Phys. 230 (2011) 2677–2695.
[29] J.H. Seo, R. Mittal, A sharp-interface immersed boundary method with improved mass conservation and reduced spurious oscillations, J. Comput. Phys. 230 (2011) 7347–7363.
[30] C. Liu, C. Hu, An efficient immersed boundary treatment for complex moving object, J. Comput. Phys. 274 (2014) 654–680.
[31] S. Kang, G. Iaccarino, F. Ham, P. Moin, Prediction of wall-pressure fluctuation in turbulent flows with an immersed boundary method, J. Comput. Phys. 228 (2009) 6753–6772.
[32] R.J. Leveque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.
[33] Z. Li, M.-C. Lai, The immersed interface method for the Navier-Stokes equations with singular forces, J. Comput. Phys. 171 (2001) 822–856.
[34] S. Xu, Z.J. Wang, A 3D immersed interface method for fluid-solid interaction, Comput. Methods Appl. Mech. Eng. 197 (2008) 2068–2086.
[35] E.M. Kolahdouz, A. Pal Singh Bhalla, B.A. Craven, B.E. Griffith, An immersed interface method for discrete surfaces, J. Comput. Phys. 400 (2020) 108854.
[36] A.W. Vreman, A staggered overset grid method for resolved simulation of incompressible flow around moving spheres, J. Comput. Phys. 333 (2017) 269–296.
[37] J.A. Benek, P.G. Buning, J.L. Steger, A 3-D Chimera grid embedding technique, AIAA Paper No. 85-1523, 1985.
[38] G. Chesshire, W.D. Henshaw, Composite overlapping meshes for the solution of partial differential equations, J. Comput. Phys. 90 (1990) 1–64.
[39] W.D. Henshaw, A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids, J. Comput. Phys. 113 (1994) 13–25.
[40] T.M. Burton, J.K. Eaton, Analysis of a fractional-step method on overset grids, J. Comput. Phys. 177 (2002) 336–364.
[41] A.W. Vreman, Particle-resolved direct numerical simulation of homogeneous isotropic turbulence modified by small fixed spheres, J. Fluid Mech. 796 (2016) 40–85.
[42] A.R. Koblitz, S. Lovett, N. Nikiforakis, W.D. Henshaw, Direct numerical simulation of particulate flows with an overset grid method, J. Comput. Phys. 343 (2017) 414–431.
[43] A.W. Vreman, J.G.M. Kuerten, Turbulent channel flow past a moving array of spheres, J. Fluid Mech. 856 (2018) 580–632.
[44] W.J. Horne, K. Mahesh, A massively-parallel, unstructured overset method for mesh connectivity, J. Comput. Phys. 376 (2019) 585–596.
[45] J.C. Brändle de Motta, P. Costa, J.J. Derksen, C. Peng, L.-P. Wang, W.-P. Breugem, J.L. Estivalezes, S. Vincent, E. Climent, P. Fede, P. Barbaresco, N. Renon, Assessment of numerical methods for fully resolved simulations of particle-laden turbulent flows, Comput. Fluids 179 (2019) 1–30.
[46] F.E. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phys. Fluids 8 (1965) 2182–2189.
[47] C. Pozrikidis, Introduction to Theoretical and Computational Fluid Dynamics, Oxford University Press, Oxford, 1997.
[48] 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.
[49] 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.
[50] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509–534.
[51] M.W. Baltussen, J.A.M. Kuipers, N.G. Deen, Direct numerical simulation of effective drag in dense gas-liquid-solid three-phase flows, Chem. Eng. Sci. 158 (2017) 561–568.
[52] A. Jameson, T. Baker, Solution of the Euler equations for complex configurations, AIAA Paper No. 83-1929, 1983.
[53] Y. Tang, S. Kriebitzsch, E. Peters, M. Van der Hoef, J. 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.
[54] K. Luo, Z. Wang, J. Tan, J. Fan, An improved direct-forcing immersed boundary method with inward retraction of Lagrangian points for simulation of particle-laden flows, J. Comput. Phys. 376 (2019) 210–227.