Computer Methods in Applied Mechanics and Engineering 453 (2026) 118833

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

A SIMPLE-based preconditioned solver for the direct-forcing immersed boundary method

Rachel Yovel [a], Eran Treister [a], Yuri Feldman [b,*]

[a] Department of Computer Sciences, Ben-Gurion University of the Negev, Beer-Sheva, Israel
[b] Department of Mechanical Engineering, Ben-Gurion University of the Negev, Beer-Sheva, Israel

ARTICLE INFO

Keywords:
Moving boundary simulations
Two way coupling FSI
Implicit immersed boundary method
Schur complement
Preconditioning

ABSTRACT

We present a robust and scalable solver for direct-forcing immersed boundary simulations, based on a preconditioned SIMPLE algorithm. The method applies block elimination to the pressure-force coupled system, and utilizes the discrete Laplacian operator as an efficient preconditioner for the resulting Schur complement. We rigorously demonstrate the spectral equivalence between the Schur complement and the discrete Laplacian, ensuring convergence behavior that is independent of grid resolution and physical parameters. This enables accurate, stable, and efficient two-way coupled fluid-structure interaction (FSI) simulations with moving boundaries and significant added-mass effects. These simulations are all executable on standard computing platforms. Extensive validation and verification — including simulations of oscillating, sedimenting, and buoyant spheres, as well as configurations involving multiple immersed bodies — confirm the solver's accuracy and efficiency across a broad range of FSI scenarios. The proposed approach introduces a novel and accessible framework for immersed boundary simulations requiring strong pressure-force coupling.

1. Introduction

Immersed boundary methods (IBMs) provide a versatile computational framework for fluid-structure interaction (FSI) problems involving complex or moving geometries, without requiring mesh regeneration or body-fitted grids. The coupled FSI problem is governed by fluid and structural equations, linked via interface constraints. These interface conditions enforce displacement and velocity continuity and stress balance, constituting a system of equations coupling velocity, pressure and Lagrangian forces. The numerical treatment of this system depends on the chosen solution strategy, broadly categorized as monolithic or partitioned. Monolithic methods solve all unknowns simultaneously, resulting in intrinsic coupling. In contrast, partitioned methods treat the fluid and structural problems within separate solver frameworks, exchanging information across the interface and enforcing coupling through explicit, semi-implicit, or fully implicit schemes. Explicit schemes apply fluid and structural updates sequentially within each time step. Semi-implicit methods introduce limited, typically one-way temporal coupling, updating one subsystem based on temporally lagged information from the other. Fully implicit schemes employ two-way coupling and iterate both subsystems to convergence at each time step. Explicit approaches can work well in steady or weakly coupled situations. However, they often fail when rapid transients occur, when the fluid and structure has comparable characteristic time scales, or when strong added-mass effects are present.

* Corresponding author.
E-mail addresses: yovelr@bgu.ac.il, racheli.yovel@gmail.com (R. Yovel), erant@cs.bgu.ac.il, erantreister@gmail.com (E. Treister), yurifeld@bgu.ac.il (Y. Feldman).

https://doi.org/10.1016/j.cma.2026.118833
Received 15 August 2025; Received in revised form 16 January 2026; Accepted 9 February 2026
Available online 20 February 2026

In these cases, quasi-static assumptions no longer hold. To maintain stability and accurately conserve linear and angular momentum, one must use either a monolithic formulation or a partitioned scheme with semi- or fully-implicit coupling. Examples of such configurations include nearly neutrally buoyant particles [1], stiff elastic membranes [2], deformable capsules or vesicles modeled as nonlinear membranes [3], fluid–particle interaction during binder jet 3D printing [4], and fluid-structure piezoelectric interaction [5]. For a detailed overview of existing coupling schemes and their applications in incompressible FSI, see the comprehensive review [6].

Monolithic formulations offer strong coupling and numerical robustness, ensuring consistent enforcement of all constraints. These include algebraic preconditioning of the Newton iteration applied to large displacement FSI configurations [7], direct inversion of an extended Helmholtz operator via LU decomposition [8], Schur complement reduction with FFT-based Laplacian inversion under periodic boundary conditions [9,10], Schur complement-based solution of the Stokes system accelerated by geometric multigrid preconditioning [11], and stream-function-vorticity formulations with efficiently precomputed fluid-structure coupling operators [12]. Despite their robustness, monolithic methods are often computationally expensive due to the need to solve large, indefinite, and ill-conditioned saddle-point systems. An additional drawback is that the numerical methodology must be developed essentially from scratch for each specific problem, which prevents reuse of legacy codes previously developed for fluid and structural solvers.

These limitations have contributed to the growing popularity of partitioned formulations over recent decades, which offer a practical alternative with greater modularity and scalability across diverse FSI applications. Partitioned formulations of IBMs, especially those based on the direct-forcing approach [13,14], have shown particular effectiveness in incompressible flow simulations when employing semi-implicit or fully implicit coupling. These methods avoid assumptions about structural dynamics and instead enforce interfacial constraints directly through the momentum equations. The constraints are imposed by introducing Lagrangian force terms into the momentum equations to match the desired boundary velocities. Pressure and interface force fields act as distributed Lagrange multipliers (DLM), intrinsically coupled to each other and to the velocity field through the governing equations. Several works are worth mentioning in this context: in [15], it is proposed to combine all unknown Lagrangian forces into a single, fully coupled system to better reflect the parabolic nature of the Navier-Stokes equations. The works [16,17] employ moving-least-squares reconstruction and isoparametric mapping to improve overall accuracy and robustness. The works [18,19] aim to achieve high-order accuracy in enforcing the no-slip constraint, and in [20], Dirichlet and Robin boundary conditions are implemented with high accuracy in sharp corner geometries.

The immersed boundary projection method (IBPM), originally introduced in [21], has emerged as a particularly influential development within the family of partitioned direct-forcing immersed boundary formulations. It couples pressure and Lagrangian forces — both serving as Lagrange multipliers — within a unified projection framework and supports both semi-implicit and fully implicit [22] implementations. Over the past decade, IBPM has been effectively applied to simulate FSI involving rigid bodies, including sedimentation and particulate flows in two and three dimensions [23–25], as well as biologically inspired problems such as insect flight [26], vesicle dynamics over a wide range of Reynolds numbers [27,28], and thermal convection involving heat transfer [29]. More recently, integration with SIMPLE [30] and PIMPLE [19,31,32] algorithms has considerably extended the method's applicability, enabling robust simulations of stationary and arbitrarily moving geometries within widely used CFD platforms such as OpenFOAM.

While algorithmic advancements have broadened the applicability of immersed boundary methods, rigorous theoretical analysis remains limited, particularly for direct-forcing approaches. Most formal proofs of well-posedness and numerical stability have been established within the framework of variational formulations and finite element methods, typically tailored to fictitious domain methods, as demonstrated in the recent studies [33,34] and references therein. Related efforts for continuous forcing formulation using finite-difference schemes have focused on proving the unconditional stability of backward Euler and Crank-Nicolson-like discretizations of the nonlinear immersed boundary terms [35], and on developing projection-based preconditioners to accelerate semi-implicit methods through Schur complement reduction [36]. Rigorous analysis of direct-forcing IBMs, particularly for methods based on finite volume or finite difference discretizations that use distributed Lagrange multipliers (DLMs) to enforce no-slip constraints is especially scarce. To the best of our knowledge, the only such effort is the preconditioned implicit direct-forcing (PIDF) method [37], which improves no-slip enforcement through iterative force computation. However, this method relies on empirically tuned parameters, which limits its generality.

The current work contributes to bridging this gap by introducing a new preconditioning strategy for a SIMPLE-based direct-forcing IBM. The method solves a regularized saddle-point system to simultaneously update pressure and Lagrangian force corrections, ensuring incompressibility and enforcement of the no-slip constraint. This saddle-point system is first reduced via a primal Schur complement which is in turn preconditioned by a Laplacian operator. Building on this formulation, we prove rigorously that the Laplacian is spectrally equivalent to the Schur complement, enabling an efficient and robust preconditioning strategy, and demonstrate numerically that the number of Krylov iterations remains low and independent of grid resolution and problem parameters. We demonstrate the applicability of the method to both moving boundary and two-way coupled FSI problems. First, the methodology is verified through simulations of flows induced by a transversely oscillating sphere. Second, the approach is challenged with more complex configurations involving multiple moving bodies, specifically porous spheres modeled as arrays of rigid sub-spheres. The simulations reveal distinct flow features associated with different array porosities, including variations in drag coefficients, phase shifts, and vortex evolution. Then we demonstrate the applicability of the approach to two-way FSI simulations, such as sedimenting and buoyant spherical particles. In all of the above mentioned configurations, the method shows good agreement with experimental and numerical reference data while preserving numerical stability and accuracy. Finally, we demonstrate that the method achieves close to linear and sub-linear scalability in computational time and memory usage, respectively, while maintaining notably low memory requirements, making it well-suited for realistic FSI simulations on standard workstations.

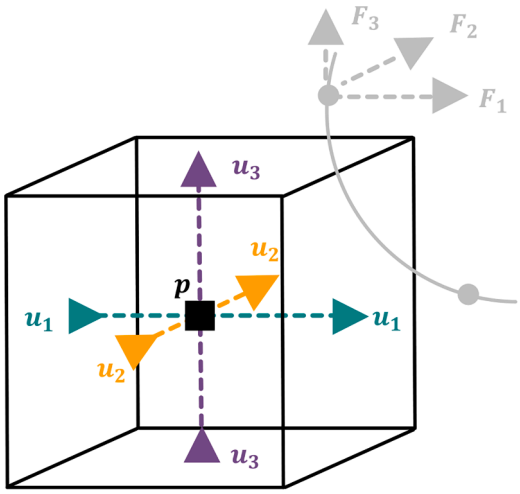


Fig. 1. An Eulerian discretization cell in 3D and two adjacent Lagrangian points on the surface of the immersed body.

2. Theoretical background

2.1. Governing equations for the dynamics of a spherical particle in a Newtonian flow.

In compliance with the direct-forcing IBM formulation, the incompressible flow around a solid body is governed by the non-dimensional Navier-Stokes (NS) and continuity equations, supplemented by the kinematic no-slip constraint on the body's surface:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla p + \frac{1}{Re}\nabla^2\mathbf{u} + \int_S \mathbf{F}(\boldsymbol{\mathcal{L}})\delta(\boldsymbol{\mathcal{L}}-\mathbf{x})dV_s \qquad \forall \mathbf{x}\in\Omega, \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0 \qquad \forall \mathbf{x}\in\Omega, \tag{2}$$

$$\mathbf{U}(\boldsymbol{\mathcal{L}}) = \int_\Omega \mathbf{u}(\mathbf{x})\delta(\mathbf{x}-\boldsymbol{\mathcal{L}})dV = \mathbf{u}_s + \boldsymbol{\omega}_s \times \widetilde{\boldsymbol{\mathcal{L}}} \tag{3}$$

where $\mathbf{u}$ and p are the velocity and pressure fields, $\boldsymbol{\mathcal{L}}$ is the series of Lagrangian points determining the surface S of the immersed body, and $\mathbf{x}$ stores the locations of the structured staggered Eulerian grid that hosts the discrete values of $\mathbf{u}$ and p (see Fig. 1). $\mathbf{F}$ is the Lagrangian force density defined within the shell unit volume dV_s surrounding each Lagrangian point, $\mathbf{U}$ is a velocity of Lagrangian point, δ is the Dirac delta function, and $\mathbf{u}_s$ and $\boldsymbol{\omega}_s$ are the translational and rotational velocities of the rigid body, respectively. $\widetilde{\boldsymbol{\mathcal{L}}}$ is the radius vector connecting the centroid of the rigid body with the surface point $\boldsymbol{\mathcal{L}}$, dV is the unit volume of an Eulerian cell, and Ω is the whole computational domain occupied by the fluid. The normalized Eqs. (1)–(3) are governed by a single non-dimensional parameter: the Reynolds number, $Re = U_0 D/\nu$ where ν is the kinematic viscosity and D and U_0 are the characteristic length and velocity, respectively. The time, pressure, and force density are normalized by utilizing the scales D/U_0, $\rho_f U_0^2$, and $\rho_f U_0^2/D$, respectively, where ρ_f is the fluid density.

Note that the last term of Eq. (1) constitutes the dynamic constraint, which defines the Lagrangian force density by which the immersed body acts on the surrounding flow, while Eq. (3) represents the kinematic no-slip constraint at the surface of the immersed body. If the kinematics of the immersed body is prescribed (e.g., the body moves with constant velocity or undergoes transverse oscillations), the configuration corresponds to a one-way coupled FSI problem. In this scenario, the system defined by Eqs. (1)–(3), complemented by appropriate boundary conditions, is mathematically closed and can be solved throughout the computational domain Ω.[1] In the case of two-way coupled FSI, the system of Eqs. (1)–(3) must be solved simultaneously with Newton's laws governing the dynamics of the solid body. These non-dimensional equations, obtained by normalizing the moment-of-inertia tensor, time, and force density using $\rho_f D^5$, D/U_0, and $\rho_f U_0^2/D$, scales respectively, and formulated within the immersed boundary framework as suggested in [38], are written for a spherical particle as:

$$\dot{\mathbf{u}}_s = \frac{6}{\pi\theta}\left[-\int_S \mathbf{F}(\boldsymbol{\mathcal{L}})\,dV_s + \frac{d}{dt}\int_{V_p}\mathbf{u}\,dV\right] - \frac{\theta-1}{\theta}Fr^{-2}\mathbf{e}_z, \tag{4}$$

$$\dot{\boldsymbol{\omega}}_s = \frac{60}{\pi\theta}\left[-\int_S \widetilde{\boldsymbol{\mathcal{L}}}\times\mathbf{F}(\boldsymbol{\mathcal{L}})\,dV_s + \frac{d}{dt}\int_{V_p}\mathbf{x}\times\mathbf{u}\,dV\right]. \tag{5}$$

[1] Note that according to the direct-forcing IBM formulation, the flow is also computed inside the immersed body. However, for a rigid body, this internal flow field does not have physical significance.

Here, the parameter $\theta = \rho_p/\rho_f$ is the ratio of particle to fluid density, and $Fr = gD/U_0^2$ is the Froude number. Importantly, for a spherical particle, the moment-of-inertia tensor reduces to a constant scalar, which allows Eqs. (4)–(5) to be conveniently formulated and solved in the inertial reference frame. However, for a general (non-spherical) particles, it is common to formulate the conservation of angular momentum in a body-attached reference frame. In this case, the coordinate system is typically aligned with the principal axes of the body, for which the moment-of-inertia tensor becomes a constant diagonal matrix. The angular velocity vector $\boldsymbol{\omega}_{cs}$ is then expressed in this body-fixed coordinate system. The transformation between the body-fixed and inertial frames is performed by utilizing rotation matrix which should be updated every time step as detailed in the next section.

2.2. Updating the rotation matrix.

The rotation matrix $\mathbf{R}^{(n+1)}$ is updated using the Rodrigues rotation formula based on the angular velocity vector. At each time step n, the angular velocity vector expressed in the body-fixed (co-rotating) reference frame, $\boldsymbol{\omega}_{cs}^{(n)}$, is first normalized to obtain the direction of the rotation axis:

$$\hat{\boldsymbol{\omega}}_{cs}^{(n)} = \frac{\boldsymbol{\omega}_{cs}^{(n)}}{\|\boldsymbol{\omega}_{cs}^{(n)}\|}. \tag{6}$$

The rotation angle γ over the time interval Δt is computed using a second-order Adams–Bashforth scheme:

$$\gamma = \Delta t\left(\frac{3}{2}\|\boldsymbol{\omega}_{cs}^{(n)}\| - \frac{1}{2}\|\boldsymbol{\omega}_{cs}^{(n-1)}\|\right). \tag{7}$$

A skew-symmetric matrix $\mathbf{W}$ is then constructed from the unit rotation axis $\hat{\boldsymbol{\omega}}_{cs}^{(n)}$:

$$\mathbf{W} = \begin{bmatrix} 0 & -\hat{\omega}_{csz}^{(n)} & \hat{\omega}_{csy}^{(n)} \\ \hat{\omega}_{csz}^{(n)} & 0 & -\hat{\omega}_{csx}^{(n)} \\ -\hat{\omega}_{csy}^{(n)} & \hat{\omega}_{csx}^{(n)} & 0 \end{bmatrix}, \tag{8}$$

and the incremental rotation matrix is computed using the Rodrigues formula:

$$\mathbf{R}_\Delta = I_m + \sin\gamma\,\mathbf{W} + (1-\cos\gamma)\,\mathbf{W}^2, \tag{9}$$

where I_m is the identity matrix. Finally, the updated rotation matrix is obtained by composing the previous orientation $\mathbf{R}^{(n)}$ with the incremental rotation:

$$\mathbf{R}^{(n+1)} = \mathbf{R}^{(n)}\mathbf{R}_\Delta. \tag{10}$$

The obtained rotation matrix $\mathbf{R}^{(n+1)}$ is orthonormal. To express any vector given in the body-attached reference frame in the inertial frame, one multiplies this matrix by the vector. Conversely, multiplying the transpose of $\mathbf{R}^{(n+1)}$ by any vector expressed in the inertial reference frame transforms it into the coordinates of the body-attached frame. In particular, our current strategy — which is also applicable to non-spherical particles — is to compute all flow variables, Lagrangian force densities, and the translational velocity of the immersed body in the inertial frame, while computing the angular velocity of the immersed body in the body-attached reference frame. The latter is achieved by first transforming the right-hand side of Eq. (5) into the body-attached reference frame and solving it for $\boldsymbol{\omega}_{cs}^{(n)}$. Once $\boldsymbol{\omega}_{cs}^{(n)}$ is computed, it is first used to calculate the rotation matrix $\mathbf{R}^{(n+1)}$ via Eqs. (6)–(9), and then transformed back to the inertial frame to evaluate the angular velocity $\boldsymbol{\omega}_s$. This angular velocity is used to compute the velocity of each Lagrangian point $\mathbf{u}(\boldsymbol{\mathcal{L}})$ using the kinematic constraint defined in Eq. (3). The position of the body is updated by adding its linear translation to the angular displacement, which is obtained by multiplying $\mathbf{R}^{(n+1)}$ with the initial position vector $\mathbf{x}^{(0)}$. More details regarding the discretization procedure and algorithmic sequence are provided in the following sections.

2.3. Interpolation and regularization operators

Since the Eulerian and Lagrangian grids do not necessarily coincide, two adjoint operators are needed to be introduced to facilitate the data transfer between them: the regularization operator R, which smears the Lagrangian forces onto the underlying Eulerian grid, and the interpolation operator R^T, which interpolates the Eulerian velocities to Lagrangian points. These operators are defined by utilizing the following discrete Dirac delta function, proposed in [39]:

$$\delta(r) = \begin{cases} \frac{1}{6\Delta r}\left(5 - 3\frac{|r|}{\Delta r} - \sqrt{-3\left(1-\frac{|r|}{\Delta r}\right)^2 + 1}\right) & \text{for } 0.5\Delta r < |r| \le 1.5\Delta r, \\ \frac{1}{3\Delta r}\left(1 + \sqrt{-3\left(\frac{|r|}{\Delta r}\right)^2 + 1}\right) & \text{for } |r| \le 0.5\Delta r, \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where r represents the distance from the smeared point and Δr represents the cell width in the r direction. This discrete Dirac delta function gained popularity over the years [21,30,38,40], due to its compact support of at most three discretization cells in each direction.[2] The regularization and interpolation operators are defined using the discrete Dirac delta function from Eq. (11) as follows:

$$\int_{\Omega} \mathbf{u}(\mathbf{x})\delta(\mathbf{x}-\boldsymbol{\mathcal{L}})\,dV = (R^T\mathbf{u})_k \approx \Delta x^3 \sum_i \mathbf{u}_i\,\delta^3(\mathbf{X}_k-\mathbf{x}_i), \tag{12}$$

$$\int_{S} \mathbf{F}(\boldsymbol{\mathcal{L}})\delta(\boldsymbol{\mathcal{L}}-\mathbf{x})\,dV_s = (R\mathbf{F})_i \approx \Delta x^3 \sum_k \mathbf{F}_k\,\delta^3(\mathbf{x}_i-\mathbf{X}_k), \tag{13}$$

where $\mathbf{x}_i$ denotes the coordinates of the ith velocity location, and $\mathbf{X}_k$ denotes the kth Lagrangian point. These discrete summation formulas approximate the integrals under the assumption that the immersed surface has sufficiently smooth local curvature and that the Lagrangian points are uniformly distributed, with spacing comparable to that of the underlying uniform Eulerian grid. The notation δ^3 refers to the three-dimensional discrete Dirac delta function. As long as the same delta function is used for both interpolation and regularization, the two operators are transposes of each other in matrix representation. For notational convenience, we denote the regularization operator by R and the interpolation operator by R^T.

3. Method: time stepping and the pressure–force-density corrections system

3.1. One-way coupled moving boundary

We begin with a brief overview of the SIMPLE-IBM approach introduced in our recent work [30] by expressing the governing Eqs. (1)–(3) in a semi-discrete form:

$$\frac{3\mathbf{u}^* - 4\mathbf{u}^{(n)} + \mathbf{u}^{(n-1)}}{2\Delta t} = -\nabla p^{(n)} + \frac{1}{Re}\nabla^2\mathbf{u}^* + R\mathbf{F}^{(n)} - (\mathbf{u}\cdot\nabla\mathbf{u})^{(n)}, \tag{14}$$

$$\frac{3\mathbf{u}'}{2\Delta t} = -\nabla p' + R\mathbf{F}', \tag{15}$$

$$R^T\mathbf{u}^{(n+1)} = \mathbf{U}^{\Gamma}, \tag{16}$$

$$\nabla\cdot\mathbf{u}^{(n+1)} = 0, \tag{17}$$

where R is the regularization operator defined in Eq. (13), which smears the forces from the Lagrangian points to the adjacent Eulerian velocity locations; R^T is its adjoint interpolation operator, defined in Eq. (12), which interpolates the intermediate velocity field onto the corresponding Lagrangian points; and $\mathbf{U}^{\Gamma}$ is the prescribed velocity value on the surface of the immersed body. Here, $\mathbf{u}^*$ is an intermediate velocity field, which is used to compute the corrections to the pressure p', velocity $\mathbf{u}'$, and Lagrangian force-density field $\mathbf{F}'$. The final stage of the algorithm consists of a correction step, $p^{(n+1)} = p^{(n)} + p'$ and $\mathbf{F}^{(n+1)} = \mathbf{F}^{(n)} + \mathbf{F}'$, followed by a projection step, $\mathbf{u}^{(n+1)} = \mathbf{u}^* + \mathbf{u}'$, which yields a divergence-free velocity field that also satisfies the kinematic no-slip constraint on the surface of the immersed body. As already mentioned, in the case of one-way moving boundary coupling, the body kinematics is prescribed a priori (i.e., the body position $\mathbf{x}^{(n+1)}$ and velocity $\mathbf{U}^{\Gamma}$ are known from the kinematic law), and the above system is closed and can be solved subject to appropriate boundary conditions. A standard finite volume central-difference discretization was applied to all the spatial terms, while a second-order backward difference was utilized for discretization of all the temporal derivatives.

When proceeding with the solution, the intermediate velocity field $\mathbf{u}^*$ is first obtained from the solution of Eq. (14). We note that this velocity field does not necessarily satisfy the no-slip and divergence-free constraints. Second, a divergence is applied to both sides of Eq. (15), similarly to the original SIMPLE method [41], yielding the modified pressure Poisson equation:

$$-\nabla^2 p' + \nabla\cdot(R\mathbf{F}') = -\frac{3}{2\Delta t}\nabla\cdot\mathbf{u}^*, \tag{18}$$

where the right-hand side is obtained by utilizing the divergence-free constraint in Eq. (17), which implies $\nabla\cdot\mathbf{u}^* = -\nabla\cdot\mathbf{u}'$. However, Eq. (18) for the pressure correction is not closed, as it also includes the force correction field. Thus, using Eq. (15) again, we obtain:

$$\mathbf{u}^{(n+1)} = \mathbf{u}^* + \frac{2}{3}\Delta t(-\nabla p' + R\mathbf{F}'). \tag{19}$$

By applying the interpolation operator R^T to both sides of Eq. (19) and applying Eq. (16), we obtain the following system of equations governing the corrections for the pressure and force density fields:

$$\begin{bmatrix} L & B^T \\ B & -C \end{bmatrix}\begin{bmatrix} p' \\ \mathbf{F}' \end{bmatrix} = \begin{bmatrix} G^TG & G^TR \\ R^TG & -R^TR \end{bmatrix}\begin{bmatrix} p' \\ \mathbf{F}' \end{bmatrix} = \begin{bmatrix} RHS_{p'} \\ RHS_{\mathbf{F}'} \end{bmatrix} \tag{20}$$

[2] Since the discrete Dirac delta function in Eq. (11) provides only a first order of spatial accuracy, other discrete delta functions can be also considered to improve the accuracy (with the price of less favorable sparsity), see [10].

where G is the discretized gradient operator, $RHS_{p'} = -\frac{3}{2\Delta t}\nabla \cdot \mathbf{u}^*$, and $RHS_{\mathbf{F}'} = \frac{3}{2\Delta t}(R^T\mathbf{u}^* - \mathbf{U}^\Gamma)$. Once the system in Eq. (20) is solved, we update the velocity, pressure, and Lagrangian force density fields and proceed to the next time step.[3] The system of equations in Eq. (20) constitutes a large, regularized saddle-point problem, whose efficient solution is a major bottleneck of the method described for one-way moving boundary, and in the next subsection we derive a similar equation that constitutes the bottleneck for two-way FSI configurations as well. The efficient solution of this saddle-point problem is a core contribution of the present study and is addressed in detail in the following sections.

3.2. Two-way coupled FSI

The conceptual difference between one-way coupled moving boundary simulations and two-way coupled FSI lies in the fact that the latter requires coupling the NS equations, governing the fluid dynamics, with the structural equations governing the dynamics of the solid body – specifically, Eqs. (4) and (5), which represent the conservation of linear and angular momentum of the solid particle. Remarkably, both equations include contributions from fluid inertia, commonly referred to as the added mass effect, which accounts for the rate of change of linear and angular momentum of the fluid enclosed within the volume occupied by the immersed body [38]. These contributions appear as volume integrals over the domain occupied by the rigid body and must be evaluated accurately to ensure physical fidelity of the simulation. In the current study, we adopt the voxel-based strategy proposed by [40], wherein the particle surface is approximated using a Cartesian grid. In this approach, each Eulerian cell is assigned a partial volume fraction $\alpha_i^{(n)} \in [0,1]$ that represents the fraction of the cell lying within the particle at time $t^{(n)}$. The computation of $\alpha_i^{(n)}$ follows the linear reconstruction scheme introduced in [40], which uses the signed distance function to approximate the volume fraction of the particle within each cell. Accordingly, the volume integrals over the particle domain, appearing in the linear and angular momentum balance equations, are approximated as:

$$\int_{V_p^{(n)}} \mathbf{u}^{(n)}\, dV \approx \sum_{i \in V_p^{(n)}} \alpha_i \mathbf{u}_i^{(n)}\, \Delta V, \tag{21a}$$

$$\int_{V_p^{(n)}} \mathbf{x}^{(n)} \times \mathbf{u}^{(n)}\, dV \approx \sum_{i \in V_p^{(n)}} \alpha_i \mathbf{x}_i^{(n)} \times \mathbf{u}_i^{(n)}\, \Delta V. \tag{21b}$$

3.2.1. Iterative coupling algorithm

As already mentioned, our method belongs to the family of partitioned approaches; that is, the fluid and structural governing equations are solved separately and coupled through internal iterations at each time step to achieve strong fluid-structure interaction. At each time step, we employ an iterative predictor-corrector scheme. The algorithm, inspired by [23], is tailored specifically to our immersed boundary framework and proceeds as follows:

Step 1: Explicit prediction of particle position and orientation. The particle's new position and orientation are explicitly predicted with second-order accuracy in time using a backward differentiation formula:

$$\mathbf{x}^{(n+1)} = \frac{1}{3}(4\mathbf{x}^{(n)} - \mathbf{x}^{(n-1)}) + \frac{2}{3}\Delta t\, \mathbf{u}_s^{(n)} + \mathbf{R}^{(n+1)}\mathbf{x}^{(0)}. \tag{22}$$

These quantities remain fixed throughout the subsequent inner iterations to maintain computational efficiency.

Step 2: Eulerian velocity prediction. We solve the discretized Navier-Stokes equations once to obtain an initial prediction, $\mathbf{u}^{(k_0)}$, for the Eulerian velocity field:

$$\frac{3\mathbf{u}^{(k_0)} - 4\mathbf{u}^{(n)} + \mathbf{u}^{(n-1)}}{2\Delta t} = -\nabla p^{(n)} + \frac{1}{Re}\nabla^2\mathbf{u}^{(k_0)} + R\mathbf{F}^{(n)} - (\mathbf{u}\cdot\nabla\mathbf{u})^{(n)}. \tag{23}$$

At this stage, incompressibility and no-slip conditions are not yet enforced.

Step 3: Iterative correction procedure. An iterative correction loop indexed by $k \geq 0$ is initiated:

- **Translational velocity update:**

$$\begin{aligned}\mathbf{u}_s^{(k)} =& \frac{1}{3}(4\mathbf{u}_s^{(n)} - \mathbf{u}_s^{(n-1)}) - \frac{4\Delta t}{\pi\theta}\int_S \mathbf{F}^{(n,k)}(\boldsymbol{\mathcal{L}}^{(n+1)})\, dV_s \\ &+ \frac{2}{\pi\theta}\left(3\int_{V_p^{(n+1)}} \mathbf{u}^{(k)}\, dV - 4\int_{V_p^{(n)}} \mathbf{u}^{(n)}\, dV + \int_{V_p^{(n-1)}} \mathbf{u}^{(n-1)}\, dV\right) \\ &- \frac{2\Delta t}{3}\cdot\frac{\theta-1}{\theta} Fr^{-2}\mathbf{e}_z.\end{aligned} \tag{24}$$

Note that this discrete form of Eq. (4), which governs the conservation of linear momentum of the solid body, is obtained here for the spherical particle by discretizing all temporal derivatives, using a second-order backward finite difference.

[3] In principle, internal fixed-point iterations may be applied to improve accuracy. However, for one-way coupling configurations and a sufficiently small time step ($\Delta t = 10^{-3}$), the desired accuracy is achieved in a single step [30].

- **Rotational velocity update:**

$$\boldsymbol{\omega}_s^{(k)} = \frac{1}{3}\left(4\boldsymbol{\omega}_s^{(n)} - \boldsymbol{\omega}_s^{(n-1)}\right) - \frac{40\Delta t}{\pi\theta}\int_S \widetilde{\boldsymbol{\mathcal{L}}}^{(n+1)} \times \mathbf{F}^{(n,k)}(\boldsymbol{\mathcal{L}})\, dV_s + \frac{20}{\pi\theta}\left(3\int_{V_p^{(n+1)}} \mathbf{x}^{(n+1)} \times \mathbf{u}^{(k)}\, dV - 4\int_{V_p^{(n)}} \mathbf{x}^{(n)} \times \mathbf{u}^{(n)}\, dV + \int_{V_p^{(n-1)}} \mathbf{x}^{(n-1)} \times \mathbf{u}^{(n-1)}\, dV\right). \tag{25}$$

  Similarly, this discrete form of Eq. (5), which governs the conservation of angular momentum of the solid body, is obtained here for the spherical particle by discretizing all temporal derivatives, using a second-order backward finite difference.
- **Lagrangian boundary velocity evaluation:** This step is required for updating the RHS of the system in Eq. (27).

$$\mathbf{U}^{(k)}(\boldsymbol{\mathcal{L}}^{(n+1)}) = \mathbf{u}_s^{(k)} + \boldsymbol{\omega}_s^{(k)} \times \widetilde{\boldsymbol{\mathcal{L}}}^{(n+1)}. \tag{26}$$

- **Pressure and force-density correction:** The coupled pressure and velocity corrections are computed at each iteration. As the iteration converges, the RHS of the system below approaches zero, and so do the correction values, since the operator is not singular.

$$\begin{bmatrix} G^T G & G^T R \\ R^T G & -R^T R \end{bmatrix} \begin{bmatrix} p' \\ \mathbf{F}' \end{bmatrix} = \begin{bmatrix} -\frac{3}{2\Delta t}\nabla \cdot \mathbf{u}^{(k)} \\ \frac{3}{2\Delta t}(R^T \mathbf{u}^{(k)} - \mathbf{U}^{(k)}) \end{bmatrix}. \tag{27}$$

- **Fields update and convergence check:** This is essentially a Gauss–Seidel step with ξ acting as an under- or over-relaxation parameter. Our numerical experiments revealed that ξ can range between 0.6 for the smallest and 1.1 for the largest values of ρ_p/ρ_f ratio.

$$\begin{aligned} p^{(n,k+1)} &= p^{(n,k)} + \xi p', \\ \mathbf{F}^{(n,k+1)} &= \mathbf{F}^{(n,k)} + \xi \mathbf{F}', \\ \mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + \xi \frac{2\Delta t}{3}\left(-\nabla p' + R\mathbf{F}'\right). \end{aligned} \tag{28}$$

  Iterations (i.e., *Step* 3) continue until the relative L_1-norm of the incremental force-density correction falls below a specified threshold (typically 10^{-3}), ensuring accurate enforcement of divergence-free and no-slip conditions:

$$\frac{\|\mathbf{F}'\|_1}{\|\mathbf{F}^{(n,k+1)}\|_1} < 10^{-3}. \tag{29}$$

  Upon convergence, we proceed to the subsequent time step.

This iterative scheme ensures consistent enforcement of the incompressibility and no-slip constraints while maintaining modularity between the fluid and solid solvers. Similarly to the one-way moving boundary case, the system (27) constitutes the bottleneck of the entire method and our main contribution is the preconditioning approach to address it, presented in the next section.

4. Method: preconditioning

To motivate the need of developing an efficient preconditioner, we first explain why the main computational bottleneck in our framework is the repeated solution of the regularized saddle-point linear system coupling the pressure and force-density corrections. First, we explain the computational framework of our solver. As described in the previous section, the fluid and structural dynamics equation are solved separately, and then we either advance directly to the next time step (for one-way coupled simulations) or perform fixed-point internal iterations (for two-way coupled simulations).

1. **Fluid dynamics solver:** The solution of the discretized Navier-Stokes equations, i.e., Eq. (14) or Eq. (23), does not present significant computational difficulties, as the governing discrete operators exhibit diagonal dominance for sufficiently small time steps (due to the presence of temporal derivative terms). In the present study, we have utilized the direct method proposed by [42], as implemented in [43], for factorizing the Helmholtz operator. However, previous experience suggests that standard Krylov subspace solvers, such as BiCGStab or GMRES, can also be employed effectively, typically converging within two or three iterations. Another feasible approach involves approximating the inverse Helmholtz operator via an Nth order Taylor series expansion, exploiting its symmetry and positive definiteness for an appropriate choice of the time step Δt and approximation order N [44].
2. **Structural dynamics solver:** in addition to the fluid equations, the system also includes equations governing structural dynamics — specifically Eqs. (22), (24), and (25) — relevant for two-way coupled fluid-structure interaction. These equations represent initial-value ordinary differential equations (ODEs), and their numerical solution is straightforward and computationally inexpensive. Currently, we discretize them using a second-order backward finite-difference scheme, although alternative methods, such as those from the Runge-Kutta family, or even analytical solutions could equally be considered.
3. **The pressure and force-density corrections coupled system:** Ultimately, the computational bottleneck and the cornerstone of our numerical framework is the efficient solution of the regularized saddle-point linear system given by either Eq. (20) for one-way coupling or Eq. (27) for two-way coupling scenarios. Since this system appears at each iteration or time step, its efficient solution has a significant impact on the overall computational cost and stability of the algorithm.

Let us examine more closely the challenges associated with solving the saddle-point system. A common approach is to solve the full coupled system directly, as originally proposed in the IBPM by [21]. In their formulation, the authors derived a modified Poisson equation by approximating the inverse Laplacian operator via a Taylor series expansion in time. Notably, when a first-order approximation is applied, it leads to a saddle-point system with the same left-hand-side as in the present work. The authors note that given a careful choice of time step and Lagrangian points, the modified Poisson operator is symmetric and positive definite, and hence the system can be solved using the conjugate gradient method. Nevertheless, the efficiency of the iterative solution is not directly addressed, and there is no guarantee for iteration count independent on grid resolution and physical parameters. A key challenge of this approach lies in the involvement of the Laplacian operator, which inherently governs the system. The Laplacian is known for its poor conditioning due to the elliptic nature of the diffusion process it represents. It spans multiple spatial scales, resulting in a wide eigenvalue spectrum and inherently poor conditioning. This limitation becomes more pronounced at higher resolutions, as finer grids capture smaller spatial scales, leading to unbounded growth in the condition number. Despite the existence of efficient methods for the solution of *standard* Poisson problems, such as multigrid method and specially tailored direct solvers, these methods are inefficient when applied directly to the *modified* Poisson equation arising in our saddle-point formulation. These considerations underscore the need for an efficient preconditioner to enable robust and rapid convergence of solutions to the saddle-point systems described by Eq. (20) or Eq. (27), as well as convergence theory. Particularly, if this preconditioner enables reduction of modified to standard Poisson problems, it will allow the efficient use of many existing methods. The development of such a preconditioner, along with both formal and numerical validation of its effectiveness, is the focus of the following sections.

4.1. The Laplacian as a preconditioner

The idea of constructing an efficient preconditioner originates from forming the *primal* Schur complement:

$$S_p := L + B^T C^{-1} B, \tag{30}$$

and subsequently solving the decomposed system of equations, first for the pressure correction p' and then for the force-density correction $\mathbf{F}'$:

$$S_p p' = RHS_{p'} + B^T C^{-1} RHS_{\mathbf{F}'}, \tag{31}$$

$$\mathbf{F}' = C^{-1}(Bp' - RHS_{\mathbf{F}'}). \tag{32}$$

To reduce computational cost, we adopt the scalar approximation $C \approx \frac{1}{2} I_m$ proposed in [30] [4] , thereby avoiding the explicit inversion of C by using $C^{-1} \approx 2I_m$. This leads to the following approximation of the primal Schur complement:

$$S_p \approx \tilde{S}_p := L + 2B^T B. \tag{33}$$

Substituting this approximation into Eqs. (31) and (32) yields the simplified system:

$$\tilde{S}_p p' = RHS_{p'} + 2B^T RHS_{\mathbf{F}'}, \tag{34}$$

$$\mathbf{F}' = 2(Bp' - RHS_{\mathbf{F}'}). \tag{35}$$

However, both S_p and $\tilde{S}_p$ remain large and ill-conditioned, primarily due to the system's dependence on the Laplacian operator. To address this, we explicitly employ the Laplacian L itself as a preconditioner. In the following sections, we demonstrate that Eq. (31) can be efficiently solved using a Krylov subspace method preconditioned by L, leveraging the spectral equivalence between S_p and L. Specifically, in Section 5, we show that the eigenvalues of the preconditioned operator $L^{-1}S_p$ are contained within a narrow, bounded interval that is independent of physical parameters and grid resolution. Provided that $\tilde{S}_p$ closely approximates S_p, similar spectral properties hold approximately for the operator $L^{-1}\tilde{S}_p$. This spectral equivalence leads to rapid convergence of Eq. (34) when solved with the Laplacian preconditioner, as verified numerically in Section 6. Since the pressure–force-density correction system constitutes the main computational bottleneck, this preconditioning strategy substantially improves overall computational efficiency and demonstrates favorable scalability.

Remark 4.1. There is abundance of literature on block-preconditioning for saddle-point matrices as appears in Eq. (20) or (27), mainly by Schur complement preconditioning. Since forming and inverting the exact Schur-complement is computationally expensive, broad research is dedicated to the search of cheap approximations for the inverse of the Schur-complement, see [45]. Our method offers an efficient approximation for the primal Schur complement for a certain class of saddle-point matrices, and can be implemented either by the steps described in Section 3 or by a corresponding block lower triangular Schur complement preconditioner.

Remark 4.2. It is worth noting that the condition number of the matrix $R^T R$ can grow significantly when the Lagrangian discretization is substantially over- or under-resolved relative to the Eulerian grid. Although the theoretical invertibility of $R^T R$ is ensured as long as the Lagrangian points are distinct (since, in this case, R has linearly independent columns), practical measures should be taken to avoid near-singularity. A common choice in the literature is to use comparable spacing between the two grids; see, e.g., [21]. Near-singularity may also occur in solid–solid or solid–boundary interactions. In such cases, a sufficiently fine grid is used to keep the points distinct, and the model includes a repulsion force that enforces point separation. An extended discussion of the grid-spacing effects on the $R^T R$ matrix–vector product approximation and the influence of boundary proximity on the preconditioner performance is provided in Appendices A and B, respectively.

[4] This scalar approximation was achieved in [30] by lumping the matrix $R^T R$.

4.2. Efficient matrix-free implementation

In this subsection, we describe the current matrix-free implementation. First, recall that the operator B^T consists of the (negative) divergence applied to the discrete Dirac delta function weights obtained from Eq. (11). A matrix-free implementation allows efficient shared-memory parallelization (implemented here with OpenMP and executed with 48 threads) and enables rapid recomputation of these operators at each time step. Second, at each step of the preconditioned Krylov iteration, only a matrix-vector product with the preconditioned operator is required. We employ left preconditioning, which is mathematically equivalent to solving Eq. (34) after multiplying both sides from the left by L^{-1}. However, we do not explicitly form the inverse operator L^{-1}, nor do we repeatedly apply a Laplacian solver to multiple columns. Instead, noting that the preconditioned system can be represented as:

$$L^{-1}\tilde{S}_p = I_m + 2L^{-1}B^T B,$$

where I_m is an identity matrix of appropriate dimension, we observe that the product of $L^{-1}\tilde{S}_p$ with a vector $\mathbf{v}$ can be efficiently obtained by applying the direct solver of [42] only once to the vector $B^T(B\mathbf{v})$. Employing Intel MKL routines[5] allows efficient computation of the sparse matrix-vector products. Furthermore, due to the scalar approximation for C^{-1}, computing the second term of Eq. (34) and the entire Eq. (35) is trivial, as both involve only a single matrix-vector product. Importantly, our matrix-free implementation requires only a single application of the operator L^{-1} to the vector $B^T(B\mathbf{v})$ per iteration or time step. Consequently, it incurs no additional computational cost compared to non-preconditioned solvers for Eqs. (20) or (27).

Finally, we discuss the factorization of the preconditioner L. Lynch [42] demonstrated that any matrix expressible as a sum of Kronecker products of smaller matrices with identity matrices can be factorized efficiently using eigen-decomposition of each smaller matrix. As shown therein, the resulting system can be solved with a computational complexity of $O(n^{4/3})$ for a regular 3D grid. The computational cost can be further reduced by applying the Thomas TDMA algorithm along one of the three coordinate directions. For transient problems, this approach is highly efficient, as the associated operators can be factorized once in advance and reused at every time step [43].

5. Theory: a spectral bound on the preconditioned system

In this section, we analyze the use of the Laplacian L as a preconditioner for the Schur complement S_p of the coupled pressure–force-density system (20) or (27). A good preconditioner should resemble the original matrix, in the sense of spectral equivalence, i.e., the spectrum of the preconditioned system should be bounded. Theorem 5.1 shows this property.

Theorem 5.1. *For every generalized saddle-point matrix of the form*

$$\begin{bmatrix} L & B^T \\ B & -C \end{bmatrix} = \begin{bmatrix} G^T G & G^T R \\ R^T G & -R^T R \end{bmatrix}, \tag{36}$$

where G is an $n \times m$ matrix and R an $n \times k$ matrix[6], the leading block L is spectrally equivalent to the primal Schur complement S_p from Eq. (30), in the sense

$$\mathrm{spec}(L^{-1}S_p) \subseteq [1, 2]. \tag{37}$$

Proof. It is readily seen that

$$L^{-1}S_p = I_m + L^{-1}B^T C^{-1}B$$

where I_m stands for the identity matrix of size $m \times m$. Hence, it is sufficient to show that

$$\mathrm{spec}(L^{-1}B^T C^{-1}B) \subseteq [0, 1],$$

or equivalently, that the generalized eigenvalue problem

$$B^T C^{-1}B\mathbf{v} = \lambda L\mathbf{v} \tag{38}$$

has only solutions that satisfy $\lambda \in [0, 1]$. Observe that

$$B^T C^{-1}B = G^T R(R^T R)^{-1}R^T G = G^T PG$$

where $P := R(R^T R)^{-1}R^T$ is an orthogonal projector (namely, $P^2 = P$ and $P^T = P$). To bound λ, we bound the generalized Rayleigh quotient

$$\frac{\langle B^T C^{-1}B\mathbf{v}, \mathbf{v}\rangle}{\langle L\mathbf{v}, \mathbf{v}\rangle} = \frac{\langle G^T PG\mathbf{v}, \mathbf{v}\rangle}{\langle G^T G\mathbf{v}, \mathbf{v}\rangle} = \frac{\langle PG\mathbf{v}, G\mathbf{v}\rangle}{\langle G\mathbf{v}, G\mathbf{v}\rangle}, \quad \mathbf{v} \in \mathbb{R}^m. \tag{39}$$

Since it is an orthogonal projector, the only eigenvalues of P are 0 and 1, and $\mathrm{Null}(P) = \mathrm{Range}(P)^{\perp}$, see, e.g., [46]. Particularly, every vector can be uniquely decomposed into a sum of orthogonal vectors in the nullspace and in the range of P, e.g., we can write $G\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2$, where $\mathbf{v}_1 \in \mathrm{Null}(P)$, $\mathbf{v}_2 \in \mathrm{Range}(P)$ and $\langle \mathbf{v}_1, \mathbf{v}_2\rangle = 0$. Consequently,

$$\langle PG\mathbf{v}, G\mathbf{v}\rangle = \langle P(\mathbf{v}_1 + \mathbf{v}_2), \mathbf{v}_1 + \mathbf{v}_2\rangle = \langle P\mathbf{v}_1, \mathbf{v}_1\rangle + 2\langle P\mathbf{v}_1, \mathbf{v}_2\rangle + \langle P\mathbf{v}_2, \mathbf{v}_2\rangle = \langle \mathbf{v}_2, \mathbf{v}_2\rangle = \|\mathbf{v}_2\|^2, \tag{40}$$

[5] We convert B and B^T into CSR format at each time step to facilitate efficient use of Intel MKL routines.

[6] Typically, $n > m > k$, but this assumption is not necessary for the Theorem's statement.

where the third equality holds because $P\mathbf{v}_1 = 0$ and $P\mathbf{v}_2 = \mathbf{v}_2$. Finally, we substitute (40) into (39), which yields

$$0 \leq \frac{\langle PG\mathbf{v}, G\mathbf{v}\rangle}{\langle G\mathbf{v}, G\mathbf{v}\rangle} = \frac{\|\mathbf{v}_2\|^2}{\|\mathbf{v}_1 + \mathbf{v}_2\|^2} = \left(\frac{\|P(\mathbf{v}_1 + \mathbf{v}_2)\|}{\|\mathbf{v}_1 + \mathbf{v}_2\|}\right)^2 \leq \|P\|_2^2 = (\rho(P))^2 = 1 \tag{41}$$

where $\rho(P)$ is the spectral radius of P, and since P is symmetric, it is equal to $\|P\|_2$. □

The importance of Theorem 5.1 stems from its generality, i.e., for the spectral equivalence to hold, any discretisation of the gradient and any regularization operator can be considered. The expected scalability and robustness of the method is therefore proved for a large variety of cases, with symmetry being the only limitation, i.e., the divergence must be the transpose of the gradient, and the interpolation must be the transpose of the regularization.

The assumption of symmetry is not particularly restrictive, as it is natural to take the same discrete Dirac delta functions for the regularization and interpolation, and the same discretization for the gradient and divergence. In practice, however, some configurations might lead to a non-symmetric Laplacian. These include, for example, computational domains with all no-slip boundary conditions, which require adding a Dirichlet point within the computational domain to prevent singularity of the pressure-correction Laplacian. Nevertheless, even when the Laplacian slightly differs from G^TG, the spectral result of Theorem 5.1 seems to hold in practice, as demonstrated in the next section.

Remark 5.1. As an immediate corollary of Theorem 5.1, the error iteration matrix $T = I_m - L^{-1}S_p$ of the preconditioned system satisfies $\rho(T) \leq 1$. When this inequality is strong, convergence is guaranteed. A Krylov iterative method typically further accelerates the convergence and might converge in just a few iterations even when the spectral radius of the error iteration matrix is even greater than 1. However, the convergence rate of a Krylov iterative method is determined by the scattering of the eigenvalues, which can depend on the specific configuration. In typical cases, the number of Lagrangian points is very small compared to the number of Eulerian cells. In such a case,

$$\tilde{S}_p = L + 2B^T B \approx L \tag{42}$$

and the error iteration matrix has a large null-space, which implies that the eigenvalues are clustered and the Krylov iteration count is low. In the next section, we demonstrate that even when the number of Lagrangian points grows — by taking multiple bodies — the observed iteration count remains low.

Remark 5.2. A similar result can be proved for a weighted version of the saddle-point matrix, such as

$$\begin{bmatrix} G^TWG & G^TWR \\ R^TWG & -R^TWR \end{bmatrix}, \tag{43}$$

where W is SPD, by taking $\tilde{G} = W^{\frac{1}{2}}G$ and $\tilde{R} = W^{\frac{1}{2}}R$. Since such a weighted system appears in [21], it gives rise to a utilization of a similar preconditioning approach in the context of projection methods.

6. Results

In this section we demonstrate the efficiency of our method in solving various problems. In Section 6.1 we present several model problems in 3D, and verify our method by comparing the simulation results to existing references. In Section 6.2 we demonstrate the computational efficiency of our preconditioning approach in 2D and in 3D.

6.1. Verification study

6.1.1. Model problem 1: a transversely oscillating solid sphere (one-way coupling)

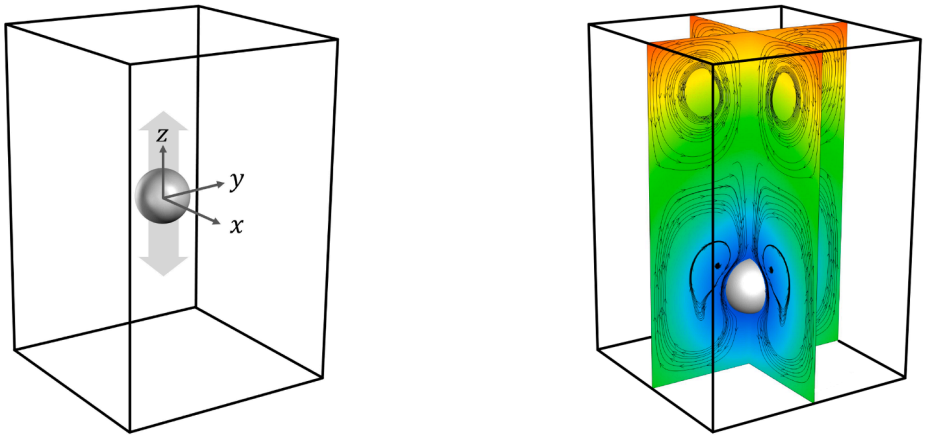
A transversely oscillating solid sphere of diameter D placed within a $4D \times 4D \times 6D$ box filled with a quiescent fluid (see Fig. 2(a)) is considered. The sphere oscillates vertically with velocity and position given by:

$$U_z = U_{max}\sin(\omega T), \qquad Z = -\frac{U_{max}}{\omega}\cos(\omega T) \tag{44}$$

where ω is the angular frequency and T is the time. No-slip boundary conditions are imposed on all solid boundaries, including the surface of the sphere and the walls of the box. To non-dimensionalize the system, D, $U_{\max}$, $U_{\max}/D$, and $\rho U_{\max}^2$ are utilized as characteristic scales of length, velocity, time, and pressure, respectively, yielding the following relationships governing the sphere's kinematics:

$$u_z = \sin\left(\frac{D}{A}t\right), \qquad z = -\frac{A}{D}\cos\left(\frac{D}{A}t\right) \tag{45}$$

where $A = U_{\max}/\omega$ represents the amplitude of the oscillations. The system's behavior is governed by two non-dimensional parameters: the Reynolds number, $Re = U_{\max}D/\nu$, and the amplitude ratio, A/D (where D/A is also known as the reduced frequency). A snapshot of representative flow including pathlines and contours of the vertical velocity component u_z around the sphere, typical of the given set-up, is presented in Fig. 2(b).



(a) Physical model (b) Snapshot of a representative flow

Fig. 2. An oscillating sphere of diameter D and a box of dimensions $4D \times 4D \times 6D$.

For completeness, we recall the force balance equation of an accelerating sphere in an otherwise quiescent fluid, as described in [47]. In accordance with the IBM formalism where the sphere is filled with the same fluid as that outside, this equation can be expressed as [48]:

$$\frac{d}{dt}\int_{V_{sph}} \mathbf{u} dV = \mathbf{f}_D + \int_{V_{sph}} \mathbf{f} dV \tag{46}$$

where $\mathbf{f}_D$ is the instantaneous drag force exerted by the fluid on the sphere and the integral of $\mathbf{f}$ is the instantaneous external force required to maintain the prescribed kinematics of the body. This external force can be directly computed by summing all the IBM forces:

$$\int_{V_{sph}} \mathbf{f} dV \approx \sum_{i,j,k} \mathbf{f}_{ijk} \Delta x \Delta y \Delta z. \tag{47}$$

The drag force, $\mathbf{f}_D$, can be determined by using Eqs. (46) and (47). Following [49], we assume rigid-body motion of the fluid within the sphere, allowing us to approximate the left-hand side term of Eq. (46) as

$$\frac{d}{dt}\int_{V_{sph}} \mathbf{u} dV \approx \frac{d\mathbf{u}_c}{dt} V_{sph} \tag{48}$$

where $d\mathbf{u}_c/dt$ is the acceleration of the sphere's center of mass. For comparison purposes, we express the drag force $\mathbf{f}_D$ in terms of the drag coefficient for a spherical geometry:

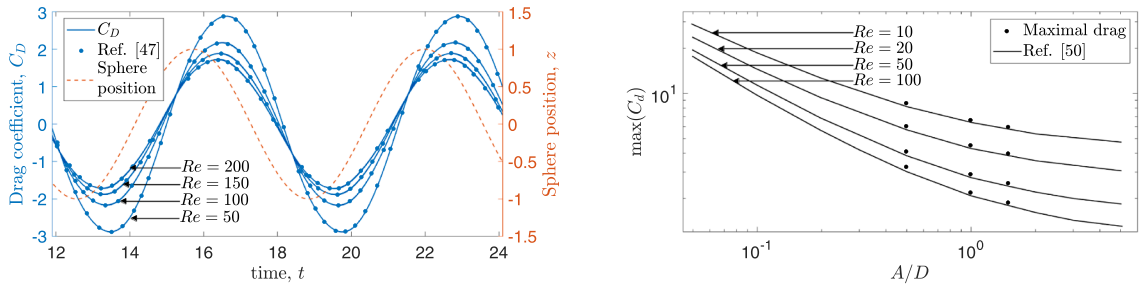
$$\mathbf{C}_D = \frac{8\mathbf{f}_D}{\rho U_{max}^2 \pi D^2}. \tag{49}$$

After scaling the drag force by $\rho U_{\max}^2 D^2$ the z component of the drag coefficient $C_D \equiv (C_D)_z$ becomes $C_D = 8 f_{D_z}/\pi$.

To verify our method, we compare the time evolution and maximum values of the drag coefficient C_D obtained on a $400 \times 400 \times 600$ grid for four Reynolds numbers in the range $50 \leq Re \leq 200$ at $A/D = 1$ with the corresponding data reported in [47] and [50]. Following the principles outlined in Section 2.1, the immersed boundary is represented by Lagrangian points uniformly distributed with spacing approximately equal to the Eulerian grid size. This uniform distribution was achieved using Leopardi's noniterative method [51], which ensures each Lagrangian point is associated with a virtual surface region of equal area. The comparison of the drag coefficient time evolution is presented in Fig. 3(a), where the results of our current simulations are shown by solid lines, while the reference data reported in [47] are denoted by filled circles. The sphere's position is indicated by a dotted line. Excellent agreement is obtained for the entire range of governing parameters, with the maximum relative deviation not exceeding 2.1%, thus successfully verifying our results. In Fig. 3(b), the maximum drag coefficient values obtained in the present study are compared with the results for the entire range of Re and A/D values reported in [50], under the assumption of axisymmetric non-confined flow. The maximum drag coefficients for $A/D = 0.5, 1$ and 1.5 are shown by filled circles, while the results from [50] are denoted by a solid line. The results demonstrate good agreement, with deviations not exceeding 6.7%. The slightly higher values obtained in the present study can be attributed to the presence of walls with no-slip conditions in our setup.

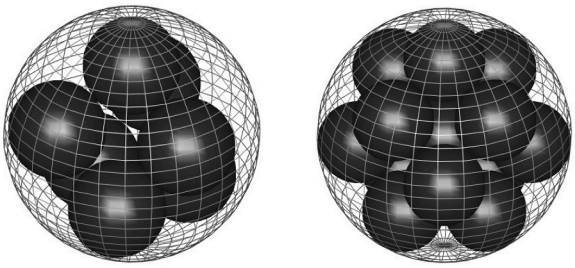
6.1.2. Model problem 2: multiple packed spheres (one-way coupling)

In the previous subsection we verified our method for the case of a single body. In this subsection we preform additional experiments to further demonstrate the capabilities of the developed method for simulation of multiple moving bodies.



(a) Time evolution of the drag coefficient, C_D (b) Maximal values of the drag coefficent, $\max(C_D)$

Fig. 3. Time evolution of the drag coefficient $C_D = 8f_z/\pi$ as a function of time for $A/D = 1$, and maximal values of the drag coefficient, $\max(C_D)$, as a function of A/D. The dotted line shows the position of the sphere's center (right axis).



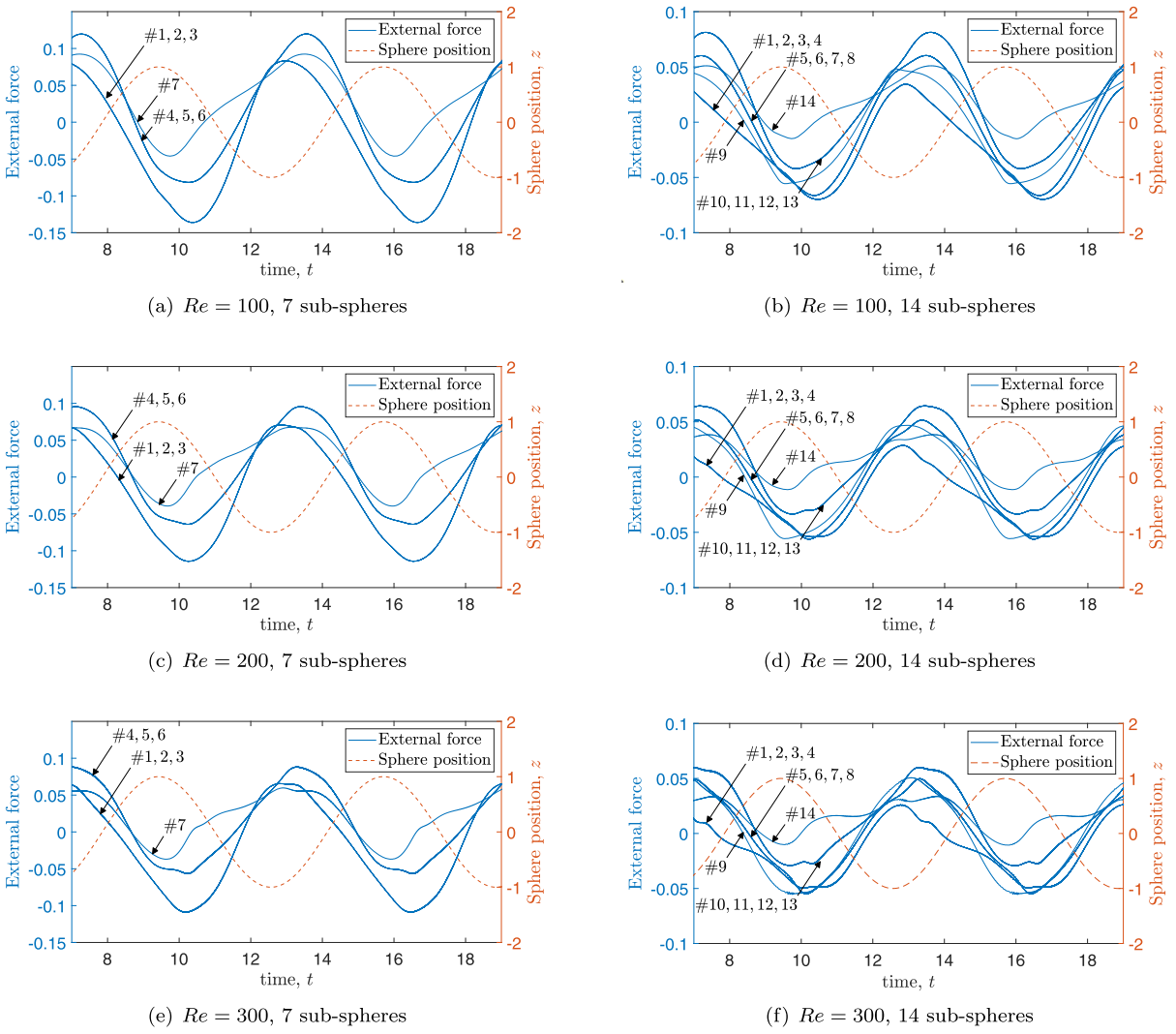
(a) 7 packed sub-spheres (b) 14 packed sub-spheres

Fig. 4. Schematic representation of porous spheres modeled by arrays of (a) 7 and (b) 14 solid sub-spheres packed without overlap within a bounding sphere of diameter D.

Table 1
Center coordinates of 7 and 14 sub-spheres constituting the model of a bounding sphere of diameter D. Here r is the radius of sub-spheres normalized by D, and ϵ is the corresponding porosity. The center of the bounding sphere is located at the origin of the coordinate system.

7 sub-sphere configuration, $r = 0.193$, $\epsilon = 0.598$				14 sub-sphere configuration, $r = 0.161$, $\epsilon = 0.526$			
# Sub-sphere	Coordinates of sub-sphere center			# Sub-sphere	Coordinates of sub-sphere center		
	$x \times 10$	$y \times 10$	$z \times 10$		$x \times 10$	$y \times 10$	$z \times 10$
1	1.1140	−1.9296	−2.1126	1	0	−2.287	−2.4917
2	−2.2281	0	−2.1126	2	−2.2875	0	−2.4917
3	1.1140	1.9296	−2.1126	3	2.2875	0	−2.4917
4	−1.5009	−2.5997	0.64522	4	0	2.2875	−2.4917
5	3.0019	0	0.64522	5	−2.3820	−2.3820	−0.30478
6	−1.5009	2.5997	0.64522	6	2.3820	−2.3820	−0.30478
7	0	0	3.0704	7	−2.3820	2.3820	−0.30478
				8	2.3820	2.3820	−0.30478
				9	0	0	0
				10	0	−2.8412	1.8355
				11	−2.8412	0	1.8355
				12	2.8412	0	1.8355
				13	0	2.8412	1.8355
				14	0	0	3.3825

To this end, we focus on investigating flow properties in general and drag coefficients in particular that are typical of oscillating porous spheres. This knowledge is critical for understanding and controlling many practical applications, including the behavior of particle clusters in fluidized bed reactors, the movement of particulate matter in power generation systems, the dynamics of microcarrier transport in bioprocess applications, and the settling behavior of sediments in coastal waters [52,53], to name a few. Following the methodology of Ma et al. [54], a porous sphere is modelled as an array of small solid sub-spheres packed without overlap within a larger bounding sphere of diameter D. The packing configuration of these small spheres is not unique and can be achieved through various approaches. For the present study, we employed the serial symmetrical relocation algorithm developed by Huang and Liang [55] to generate configurations with 7 and 14 sub-spheres.



(a) $Re = 100$, 7 sub-spheres

(b) $Re = 100$, 14 sub-spheres

(c) $Re = 200$, 7 sub-spheres

(d) $Re = 200$, 14 sub-spheres

(e) $Re = 300$, 7 sub-spheres

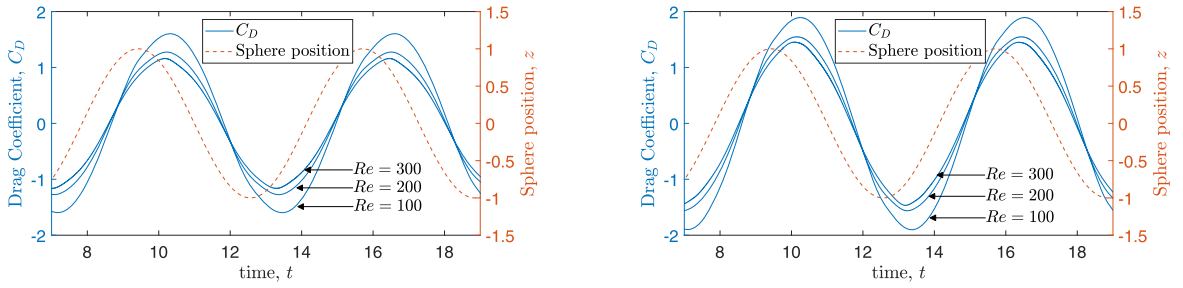
(f) $Re = 300$, 14 sub-spheres

Fig. 5. Time evolution of external forces applied to individual sub-spheres in the 7-sphere array and in the 14-sphere array during oscillatory motion. Each solid curve corresponds to the force applied to a sub-sphere identified by its serial number from Table 1.

The details regarding the radii and packing arrangements for each configuration of sub-spheres are given in Table 1. The table also includes the corresponding porosity values ϵ, defined as the ratio of void volume to total volume ($\epsilon = V_{void}/V_{total}$). For numerical stability in the IBM implementation, each sub-sphere's radius was reduced by half a grid spacing to ensure proper surface contact resolution while avoiding ill-conditioning. The physical configurations of the bounding spheres, containing 7 and 14 sub-spheres, are illustrated in Fig. 4a and b, respectively. A spherical wire-frame is shown in these figures to indicate the boundary of the theoretical bounding sphere.

To understand the dynamics of the system, we analyze the external forces (calculated using Eq. (47)) required to drive the oscillatory motion of each sub-sphere. Fig. 5 presents the temporal evolution of these external forces (left axis) for the 7 sub-sphere and 14-sphere arrays under three different Reynolds numbers: $Re = 100$, $Re = 200$, and $Re = 300$. The dotted line represents the temporal evolution of the z-coordinate of the bounding sphere's center (right axis).

The analysis of Fig. 5(a)–(c) demonstrates that sub-spheres experience different temporal forces, with their magnitudes and patterns determined by vertical position within the bounding sphere. Sub-spheres at equal vertical positions experience identical temporal forces regardless of their horizontal coordinates, confirming the axial nature of the force distribution. The temporal evolution of forces on individual sub-spheres exhibits clear asymmetry, quantified by differences between positive and negative force peaks. Sub-spheres #1, #2, and #3, positioned below the bounding sphere's center, experience negative forces approximately 50% larger than their positive counterparts during downward motion, with reduced forces during upward motion due to shielding from upper sub-spheres. Sub-sphere #7, positioned at the top edge of the bounding sphere, displays an inverse asymmetry pattern, with positive force peaks



(a) 7 sub-spheres (b) 14 sub-spheres

Fig. 6. Time evolution of the total drag coefficient, C_D obtained for the values of $Re = 100$, 200 and 300.

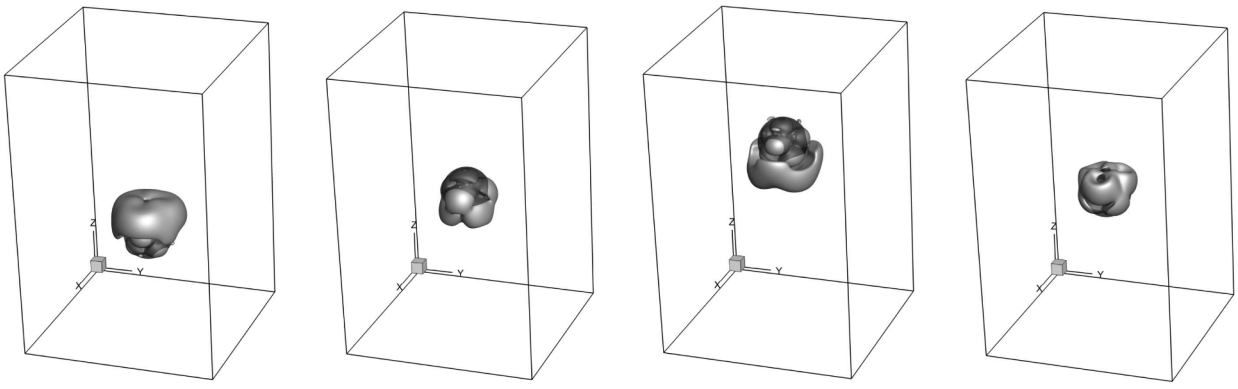
exceeding negative values by approximately 40%. Sub-spheres #4, #5, and #6, located slightly above the center and attached to the outer boundary of the bounding sphere, remain unshielded by other sub-spheres throughout the oscillation cycle. This unshielded position results in nearly symmetric forces about zero, with peak-to-peak variations not exceeding 10%. The magnitude of these forces systematically decreases with increasing Reynolds number across all cases, while maintaining consistent phase relationships relative to the bounding sphere's position. This reduction in force amplitude demonstrates the diminishing effect of viscous forces at higher Reynolds numbers, though the fundamental force distribution patterns persist.

The behavior of external forces in the case of 14 sub-spheres exhibits similar patterns while revealing additional features due to the more complex internal structure. The analysis of Fig. 5(b)–(f) confirms that sub-spheres at equal vertical positions experience identical temporal forces regardless of their horizontal coordinates. Sub-spheres #1 through #4, positioned at $z \approx -0.249$ below the bounding sphere's center, experience negative forces approximately 45% larger than their positive counterparts during downward motion, with reduced forces during upward motion due to shielding from upper sub-spheres. Sub-spheres #5 through #8, located near the equatorial plane at $z \approx -0.030$, show intermediate behavior with force asymmetry less pronounced than for the bottom spheres. Their position, slightly below the geometric center, results in force patterns that reflect partial shielding effects from both upper and lower neighboring spheres. The force distribution of sphere #9, uniquely positioned at the exact geometric center ($z = 0$) of the confining sphere, demonstrates nearly symmetric behavior about zero, with peak-to-peak variations not exceeding 5%, due to balanced shielding from surrounding sub-spheres. Sub-spheres #10 through #13, located at $z \approx 0.184$, display moderate force asymmetry with positive peaks approximately 30% larger than negative ones. Sub-sphere #14, positioned at $z \approx 0.338$ at the top edge of the bounding sphere, exhibits the strongest asymmetry in the upper region, with positive force peaks exceeding negative values by approximately 35%. As in the previous case, these asymmetric force patterns persist across all examined Re values, with diminishing magnitudes at higher Reynolds numbers.

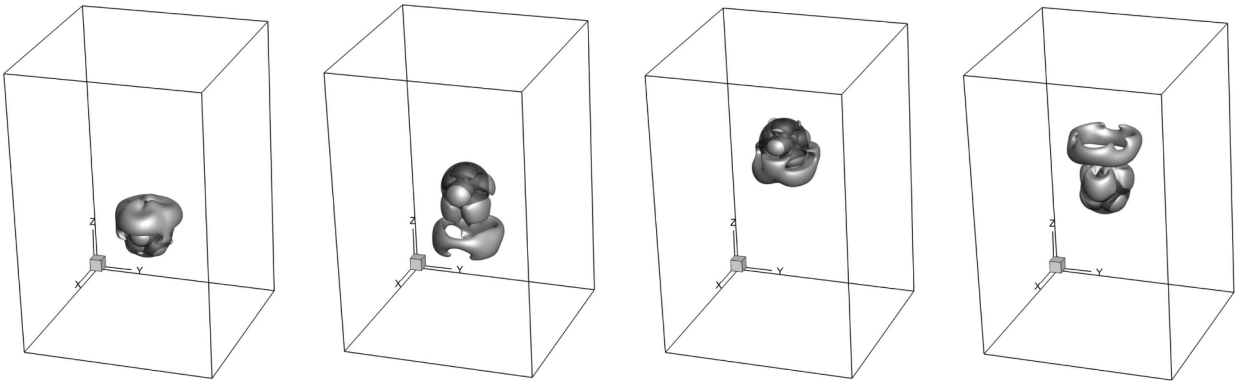
We next analyze the total drag coefficients, C_D, for arrays consisting of 7 and 14 sub-spheres. These coefficients are calculated by summing the drag forces in the z direction acting on each sub-sphere within the corresponding array and multiplying by a factor of $8/\pi$ (see Eq. (49)). The time evolution of C_D, calculated for arrays consisting of 7 and 14 sub-spheres, is shown in Fig. 6.

Looking at Fig. 6, we observe three key trends in the drag coefficient evolution. First, the magnitude of C_D systematically decreases with increasing Reynolds numbers, for both 7 and 14 sub-sphere arrays. This trend directly reflects diminishing viscous effects at higher Re values, resulting in reduced drag forces. Second, the 14 sub-sphere configuration exhibits consistently higher C_D values compared to the 7 sub-sphere array across all Reynolds numbers, approaching the drag coefficient values that characterize a non-porous sphere, as reported in our previous work (see [47], Fig. 7). This behavior is expected, as increasing the number of sub-spheres reduces the array's porosity, making it more similar to a solid sphere. Third, a persistent phase lag exists between the sphere position and drag coefficient evolution. This phase-lag arises from the fundamental relationship between force and motion in oscillatory flows: the drag force is primarily dependent on the velocity rather than the displacement. At low Re values, the force reaches its maximum when the velocity is highest (at zero displacement) and its minimum when the velocity is zero (at maximum displacement), creating an inherent quarter-period phase shift. This kinematic relationship persists across all Re values, with additional inertial effects becoming more pronounced at higher Reynolds numbers but not altering the basic phase relationship. Note that a similar phase-lag is present in Fig. 3 from the previous subsection, corresponding to a single oscillating sphere.

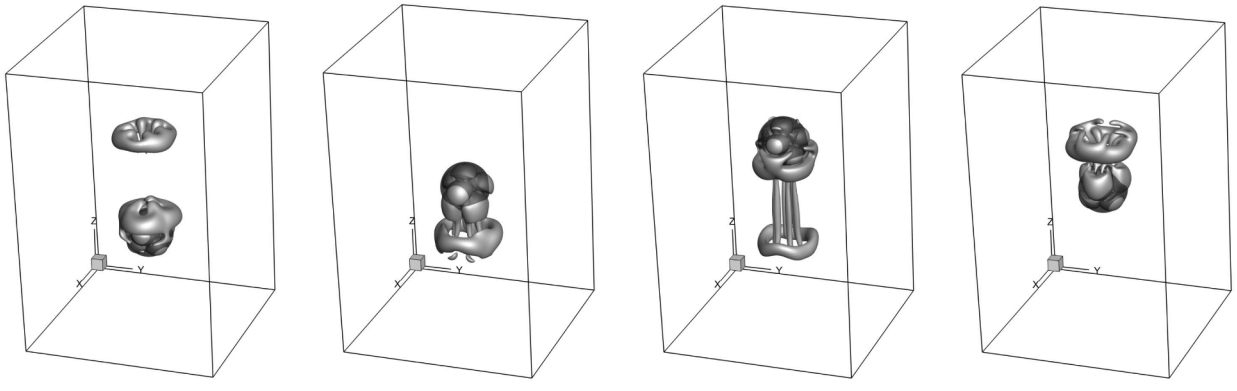
To gain deeper insight into the fluid dynamics associated with the transverse motion of porous spheres, we visualize the vortical structures generated by the flow. The visualization utilizes isosurfaces of the Q-criterion ($Q = 0.05$), where Q represents the second invariant of the velocity gradient tensor. This value of Q was chosen to effectively capture coherent vortical structures while filtering out weaker vorticity regions, thereby highlighting the dominant flow features. The evolution of vortical structures over one oscillation period is shown in Figs. 7 and 8 for configurations containing 7 and 14 sub-spheres, respectively. In both configurations, a clear Reynolds number dependence is observed in the persistence of vortical structures. At $Re = 100$, vortices dissipate rapidly due to dominant viscous diffusion. As $Re = 300$, vortical structures maintain their coherence over longer periods and travel further from the sphere surface before dissipating, reflecting the diminishing role of viscous effects relative to inertial forces.



(a) Isosurfaces of the $Q = 0.05$ criterion for $Re = 100$, calculated at different points of the sphere's trajectory: the lowest point, mid-point on the way up, highest point and mid-point on the way down (from left to right)



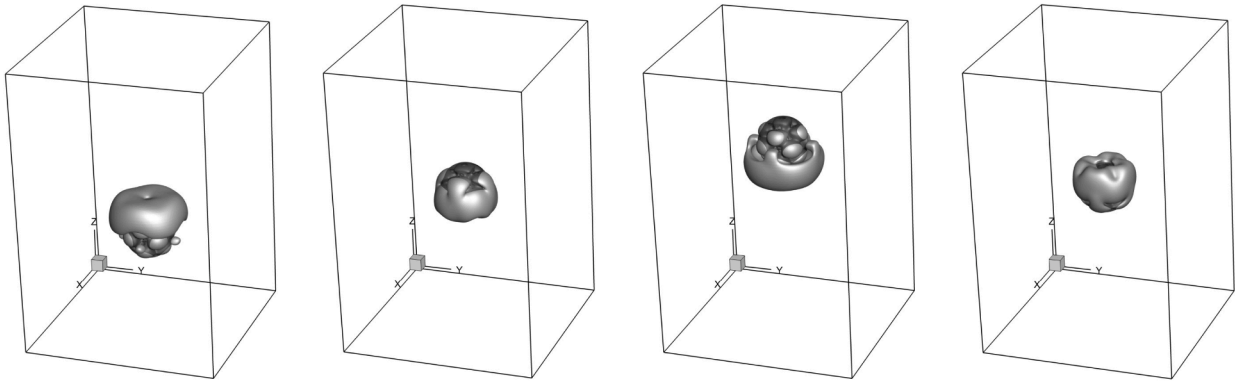
(b) Isosurfaces of the $Q = 0.05$ criterion for $Re = 200$, calculated at different points of the sphere's trajectory: the lowest point, mid-point on the way up, highest point and mid-point on the way down (from left to right)



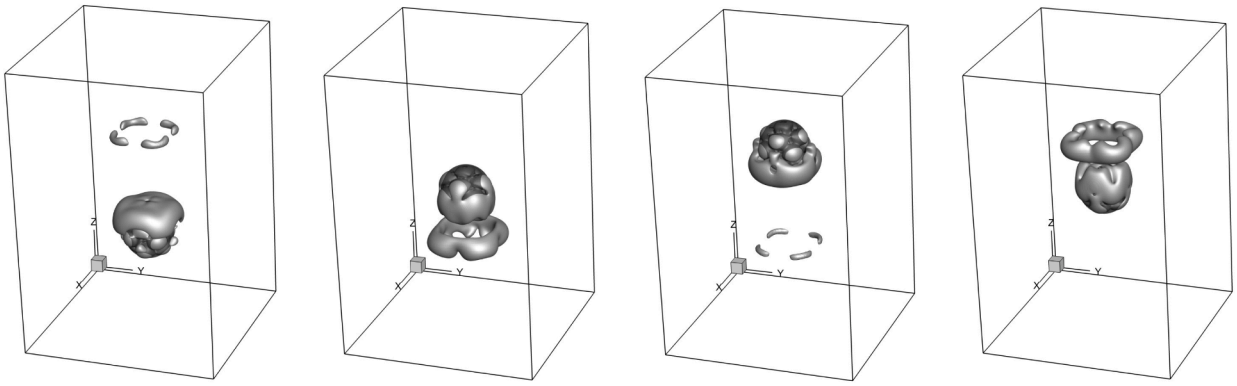
(c) Isosurfaces of the $Q = 0.05$ criterion for $Re = 300$, calculated at different points of the sphere's trajectory: the lowest point, mid-point on the way up, highest point and mid-point on the way down (from left to right)

Fig. 7. Visualization of vortical structures (isosurfaces of $Q = 0.05$) generated by a transversely oscillating sphere over a single oscillation period, calculated for $Re = 100$, 200, and 300 at $A/D = 1$. The bounding sphere containing 7 sub-spheres is indicated by a semitransparent surface.

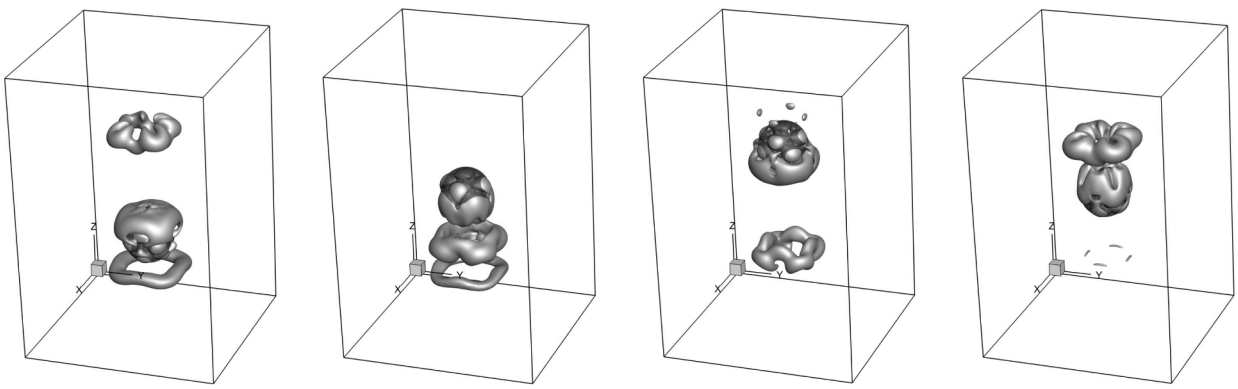
A notable distinction between the two configurations lies in their flow penetration characteristics. The 7 sub-sphere array, with its higher porosity, exhibits more pronounced flow penetration between sub-spheres, manifested in elongated vortical structures that extend through the array (Fig. 7). In contrast, the 14 sub-sphere configuration shows more compact vortical structures that primarily develop around the array's outer boundary (Fig. 8), consistent with its lower porosity. This difference is particularly evident at $Re = 300$, where the vortical structures in the 7 sub-sphere case show distinct columnar formations that are absent in the denser 14 sub-sphere arrangement. Additionally, both configurations demonstrate asymmetric vortex-shedding patterns between upward and



(a) Isosurfaces of the $Q = 0.05$ criterion for $Re = 100$, calculated at different points of the sphere's trajectory: the lowest point, mid-point on the way up, highest point and mid-point on the way down (from left to right)



(b) Isosurfaces of the $Q = 0.05$ criterion for $Re = 200$, calculated at different points of the sphere's trajectory: the lowest point, mid-point on the way up, highest point and mid-point on the way down (from left to right)



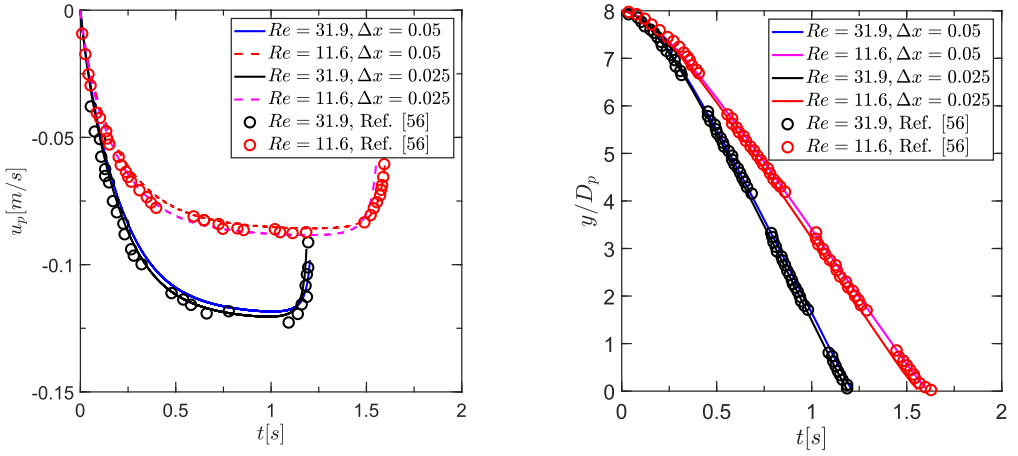
(c) Isosurfaces of the $Q = 0.05$ criterion for $Re = 300$, calculated at different points of the sphere's trajectory: the lowest point, mid-point on the way up, highest point and mid-point on the way down (from left to right)

Fig. 8. Visualization of vortical structures (isosurfaces of $Q = 0.05$) generated by a transversely oscillating sphere over a single oscillation period, calculated for $Re = 100$, 200, and 300 at $A/D = 1$. The bounding sphere containing 14 sub-spheres is indicated by a semitransparent surface.

downward motions, with more intense vortex formation occurring during direction reversal at the trajectory extrema. This asymmetry becomes more pronounced with increasing Reynolds number, particularly visible in the higher resolution structures at $Re = 300$.

6.1.3. Model problem 3: sedimentation and buoyant rise of a spherical particle (two-way coupling)

To further validate the accuracy and robustness of our numerical solver, we now consider fully two-way coupled FSI problems involving the sedimentation and buoyancy-driven motion of spherical particles in a quiescent fluid. These scenarios serve as a natural extension to the one-way coupled validation cases previously presented and pose additional numerical challenges due to the strong



(a) Time evolution of the vertical velocity

(b) Time evolution of the vertical coordinate

Fig. 9. Sedimentation of a spherical particle within a quiescent fluid: present results for $\rho_p/\rho_f = 1.164$, $Re_p = 11.6$, $Fr = 0.237$ and $\rho_p/\rho_f = 1.167$, $Re_p = 31.9$, $Fr = 0.334$, compared against the experimental data of [56].

mutual interaction between the fluid and the solid dynamics. In our first numerical experiment, we simulate the free sedimentation of a spherical particle in a rectangular domain of size $10D \times 10D \times 15D$, where D denotes the particle diameter. The particle is initially positioned along the vertical centerline at a height of $8.5D$ and released from rest. Free-slip boundary conditions were imposed on all faces of the computational domain to minimize wall effects and allow the particle to evolve as in an unbounded domain. Two grid resolutions, $200 \times 200 \times 300$ and $400 \times 400 \times 600$, were tested to assess sensitivity to spatial discretization. As shown in Fig. 9, the discrepancy between the solutions obtained on the coarse and fine grids does not exceed 7%, indicating satisfactory grid convergence. The simulated particle trajectories and velocities compare favorably with the experimental results reported by ten Cate et al. [56]. Since the fluid and solid dynamics are fully coupled, the simulations were carried out using the internal predictor-corrector iteration scheme described in Section 3.2.1, which is required to enforce the no-slip and divergence-free constraints at the fluid-solid interface. The iteration continues until the relative ℓ_1-norm of the incremental force-density correction falls below a prescribed threshold, typically $\varepsilon = 10^{-3}$. For all cases reported in this section, convergence was achieved within 2 to 3 iterations per time step. Importantly, the tested cases correspond to near-neutral buoyancy conditions, with density ratios $\rho_p/\rho_f \approx 1$, which are challenging due to the breakdown of the assumption that the fluid within the immersed body moves as a rigid body. In such regimes, the added-mass contributions must be explicitly captured for accurate prediction, further validating the correctness of our formulation.

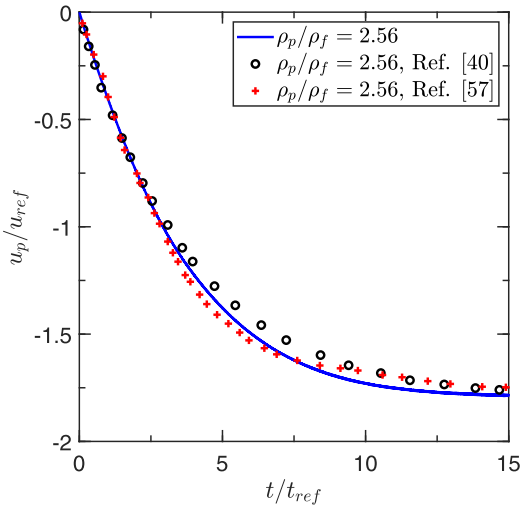
We further extend our validation by simulating additional two-way coupled cases involving both sedimenting and buoyant spherical particles. The computed vertical velocities are normalized using $u_{\text{ref}} = \sqrt{g D_p}$ and $t_{\text{ref}} = \sqrt{D_p/g}$ for velocity and time, respectively. The results are compared against both experimental data [57] and previous numerical simulations [40], as shown in Fig. 10. Excellent agreement is observed across entire range of density ratios and Reynolds numbers, further successfully verifying our two-way coupled immersed boundary solver.

6.2. Preconditioner efficiency

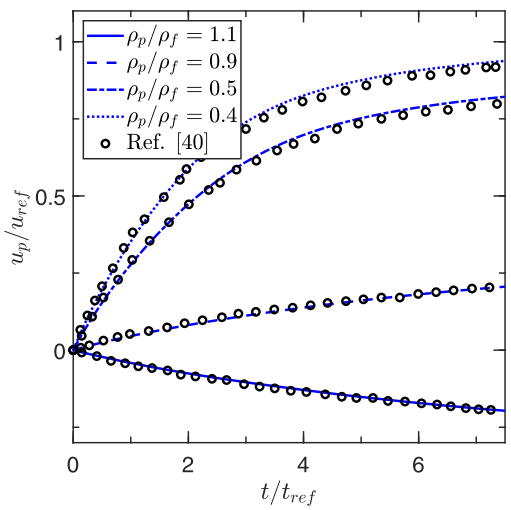
The aim of this subsection is to validate the theoretical result of Theorem 5.1 numerically, and to demonstrate that the low iteration count needed for the solution of the preconditioned system is nearly constant, for different scenarios.

In the first experiment, we verify the spectral properties of the preconditioned system on a 2D framework consisting of a stationary cylinder placed at the center of a lid-driven cavity. In Table 2, we measure the spectrum's support $[\lambda_{\min}, \lambda_{\max}]$ and show that $\text{spec}(L^{-1}S_p) \subseteq [1, 2]$, as predicted theoretically. Since the scalar approximation of C is crucial for the efficiency of our method, we also measure the spectrum's support for $L^{-1}\tilde{S}_p$, with $\tilde{S}_p$ from Eq. (33). In this case, most of the eigenvalues satisfy $\lambda \in [1, 2]$, and although some eigenvalues slightly exceed 2, the maximal eigenvalue tends to 2 as the grid resolution increases. It can be explained by that fact that the quality of the scalar approximation $C \approx \frac{1}{2}I$ improves with the grid resolution, as shown in [30].

In the second experiment we compare the GMRES iteration count required for the solution of the original system $\tilde{S}_p p' = \mathbf{b}$ and the preconditioned system $L^{-1}\tilde{S}_p p' = L^{-1}\mathbf{b}$. Note that during a full solution of the problem, the right hand side $\mathbf{b}$ changes at any time step. Here, we solved the systems for a random right hand side to demonstrate the improvement of the conditioning in terms of iteration count. The results shown in Fig. 11 demonstrate that the preconditioned system can be solved in just a few iterations



(a) Present results (solid lines); experimental data from [57] (pluses); simulation results from [40] (circles). Case: $\rho_p/\rho_f = 2.56$, $Re_p = 367$, $Fr = 1.8$.



(b) Present results (solid and dashed lines) compared to numerical results from [40]. Case: $Re_p = 367$, $Fr = 1.8$.

Fig. 10. Time evolution of the vertical velocity of sedimenting and buoyant spherical particles in quiescent fluid: comparison with prior experimental and numerical data.

Table 2
The spectrum's support of the preconditioned system $L^{-1}S_p$ and the corresponding system with an approximate Schur complement $L^{-1}\tilde{S}_p$, obtained for a stationary circular cylinder placed at the center of a 2D lid-driven cavity.

Spectrum's support for the preconditioned system				
Grid size (cells)	24×24	48×48	96×96	192×192
spec$(L^{-1}S_p) \subseteq$	[1,1.9997]	[1,1.9997]	[1,1.9996]	[1,1.9996]
spec$(L^{-1}\tilde{S}_p) \subseteq$	[1,1.9944]	[1,1.9986]	[1,2.0099]	[1,2.0067]

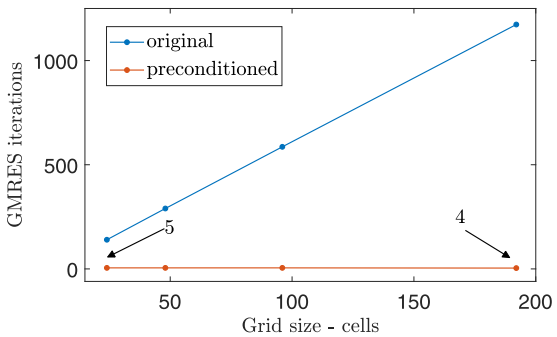


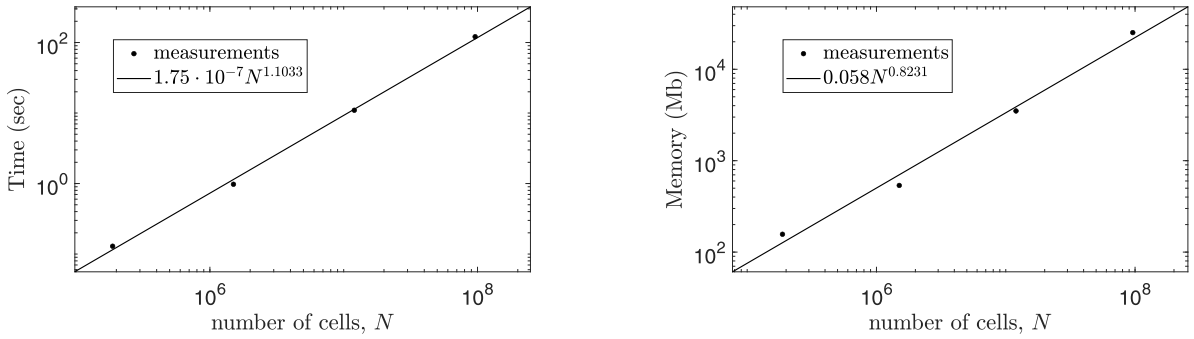
Fig. 11. GMRES iteration count for the solution of a system with a random right-hand-side for the corresponding matrices.

regardless of the grid size, while the original system requires hundreds of iterations for very small grids and thousands for larger grids. This scalability property enables performing the simulations on dense 3D grids, as shown in the next experiment.

In the third experiment we verify that the scalability shown above extends to 3D experiments with moving bodies, by demonstrating it on model problem 1 from Section 6.1.1: an oscillating sphere. We evaluate the efficiency of the proposed preconditioner by measuring the average number of BiCGStab iterations required to converge the preconditioned system to a tolerance of 10^{-12} for various Reynolds numbers and grid sizes. Table 3 summarizes these results, averaged over three amplitude-to-diameter ratios ($A/D = 0.5,\ 1.0,\ 1.5$) and four representative locations of the sphere along its periodic trajectory: bottom dead center, top dead center, and the midpoints during upward and downward motion. The results demonstrate that the average iteration count remains nearly constant–

Table 3
Average BiCGStab iteration count required for convergence of the preconditioned system to a 10^{-12} tolerance. Results are averaged over different A/D ratios and over four representative locations of the sphere along its oscillation cycle.

Iteration count for a 3D oscillating sphere						
Grid size	$Re = 10$	$Re = 20$	$Re = 50$	$Re = 100$	$Re = 200$	$Re = 300$
$50 \times 50 \times 75$	4.0	4.0	4.0	4.0	4.0	4.0
$100 \times 100 \times 150$	4.25	4.0	4.0	4.17	4.17	4.0
$200 \times 200 \times 300$	4.33	4.25	4.17	4.33	4.33	4.67
$400 \times 400 \times 600$	4.17	4.0	4.08	4.08	4.33	4.25



(a) Time scalability (b) Memory scalability

Fig. 12. Wall-clock time per time step and memory consumption as functions of grid size. Results are averaged over three A/D ratios (0.5, 1.0, 1.5) and six Reynolds numbers (10 to 300). The solid line represents a linear regression trend.

Table 4
Period-averaged BiCGStab iteration count for solving the preconditioned system at different Reynolds numbers and grid resolutions, with convergence tolerance of 10^{-12}. Results shown for configurations with 7 and 14 sub-spheres, for $A/D = 1$.

Iteration count for a 3D porous sphere								
	7 sub-spheres configuration				14 sub-spheres configuration			
Grid size	$Re = 50$	$Re = 100$	$Re = 200$	$Re = 300$	$Re = 50$	$Re = 100$	$Re = 200$	$Re = 300$
$100 \times 100 \times 150$	4.59	4.48	4.66	4.64	4.74	4.74	4.79	4.76
$200 \times 200 \times 300$	4.94	4.96	4.95	4.95	4.98	4.98	4.98	4.93
$400 \times 400 \times 600$	4.81	4.82	4.82	4.87	4.97	4.98	4.98	5.00

typically between 4 and 5 –across all tested Reynolds numbers, grid sizes, and sphere locations. This robustness highlights the effectiveness and scalability of the preconditioner.

To further assess the computational performance, we measured the wall-clock time per time step and memory consumption as functions of grid size. These experiments were conducted on a standard Linux server equipped with 64 GB DDR3 shared memory and two Intel Xeon 12C processors (48 threads total). The results, shown in Fig. 12, were averaged over the same set of A/D ratios and six Reynolds numbers ($Re = 10,\ 20,\ 50,\ 100,\ 200,\ 300$). The measured wall-clock time scales approximately as $O(N^{1.1})$, where N is the number of unknowns. This scaling aligns well with the expected complexity of the direct solver used in our implementation estimated as $O(N^{1.33})$ in [42,43], which is further improved through the use of the Thomas algorithm in one coordinate direction. Memory consumption is sub-linear and remains modest: even for the largest grid ($400 \times 400 \times 600$, hosting approximately 384 million unknowns), the total memory requirement was only 25 GB. This is a significant improvement over previous work [47], where memory consumption exceeded 128 GB for comparable problem sizes at low Reynolds numbers. Importantly, the computational efficiency and memory consumption are nearly independent of the Reynolds number and time step, provided that the CFL number remains below 0.1.

In the last experiment, we demonstrate the same scalability property on model problem 2 from Section 6.1.2, which contains multiple spheres. In the view of Remark 5.1, the number of immersed bodies might affect the sparsity of B, and thus the scattering of the eigenvalues of the preconditioned system. The aim of this experiment is to demonstrate that the same scalable behaviour holds, regardless of the number of bodies. Table 4 shows the BiCGStab iteration count needed to solve Eq. (31) with preconditioning, averaged over a single oscillation period, for different Reynolds numbers (Re = 50, 100, 200, and 300) and grid resolutions. The results show that this period-averaged iteration count remains nearly constant, requiring 4–5 iterations for both 7 and 14 sub-sphere configurations. Furthermore, these period-averaged iteration counts match the results in Table 3 for a single oscillating sphere, demon-

strating consistent performance across different configurations. These numerical results, combined with Theorem 5.1, demonstrate our method's applicability and scalability across a wide range of applications.

7. Conclusion and future work

We present a novel, preconditioned direct-forcing IBM for efficient moving boundary and two-way coupled FSI simulations, building upon the SIMPLE-IBM framework [30]. To address the coupling between pressure and force-density corrections, we employ a block reduction technique that solves for the pressure correction first, using the Laplacian operator as a preconditioner for the resulting primal Schur complement. We rigorously prove and numerically verify that the Laplacian is spectrally equivalent to the primal Schur complement. This key observation enables an efficient and robust solution strategy: since the pressure Laplacian is independent of the immersed body configuration, it can be factorized once using a fast direct solver [42] and reused throughout the simulation.

We demonstrate both theoretically and numerically that the performance of our solver is independent of grid resolution and the number of moving bodies, ensuring excellent scalability in terms of computational time and memory consumption. As a result, the method maintains low memory requirements, enabling high-fidelity moving boundary simulations that would typically require high-performance computing to run efficiently on standard workstations or even laptops. This improves the accessibility of advanced immersed boundary simulations for the broader CFD community. In this work, we demonstrated the applicability of the method to both moving boundary and two-way coupled FSI problems. The methodology was successfully verified by simulating flows induced by a transversely oscillating sphere, as well as sedimenting and buoyant spherical particles, showing good agreement with experimental and numerical reference data across a wide range of operating parameters. The approach was then extended to more complex configurations involving multiple moving bodies, specifically porous spheres modeled as arrays of rigid sub-spheres. The simulations revealed distinct flow features associated with different array porosities, including variations in drag coefficients, phase shifts, and vortex evolution, while preserving numerical stability and accuracy.

Future research should focus on extending the method to more complex two-way coupled FSI problems, particularly for simulating undulatory locomotion and deformable mesoscale porous media. The efficiency of our preconditioner does not depend on the specific choice of discrete delta function, which opens opportunities to improve accuracy by using wider-support kernels [10]. However, since our method relies on a scalar approximation of the regularization block, a key challenge will be to identify discrete delta functions with sufficiently wide support for which this scalar approximation remains valid.

Finally, the current implementation employs a direct Poisson solver [42] designed for homogeneous boundary conditions. While this is not restrictive in our context, where the pressure correction equation naturally adopts such conditions, future extensions may benefit from replacing the direct solver with a geometric multigrid V-cycle [58]. Such a strategy, if properly optimized, may achieve linear-time scaling and further enhance the versatility and accuracy of our approach for a wider class of FSI problems.

CRediT authorship contribution statement

Rachel Yovel: Conceptualization, Formal analysis, Validation, Visualization, Writing – original draft, Writing – review & editing; **Eran Treister:** Funding acquisition, Supervision, Methodology, Writing – review & editing; **Yuri Feldman:** Conceptualization, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Data availability

No data was used for the research described in the article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the Israel Science Foundation, GrantsNo. 656/23, 2746/25, and by the Lynn and William Frankel Center for Computer Science at BGU. The second author is also supported by the Ariane de Rothschild scholarship and by the Kreitman High-tech scholarship. The third author is supported by the Israel Ministry of Energy and Infrastructure (grant No. 222-11-049).

Appendix A. Grid-spacing effects on the $R^T R$ matrix–vector product approximation

We investigate the effect of the relative spacing between the Eulerian and Lagrangian grids on the validity of the $R^T R$ matrix–vector product approximation by a scaled diagonal matrix. This approximation, based on the assumption of a gradually varying distribution of Lagrangian forces for sufficiently smooth geometries, as established in our previous study [30], is critical for achieving high computational efficiency of the developed solver. The purpose is to investigate to what extent the validity of this assumption

Table A.1
Influence of the Eulerian–Lagrangian grid spacing on convergence and solution accuracy. The values of $[R^T R]$ (min, max, avg) are identical for both Reynolds numbers. For all converged cases, 4–5 BiCGStab iterations were required per time step. When divergence occurred, it typically led to fatal failures in satisfying the continuity equation and enforcing the no-slip kinematic constraint on the surface of the transversely oscillating sphere.

Re = 11.6			**Re = 31.9**			$[R^T R]$(min, max, avg)
$\Delta h_{Eu}/\Delta h_L$	Converged	v_z	$\Delta h_{Eu}/\Delta h_L$	Converged	v_z	
2.00	×	−0.05478	2.00	×	−0.06166	1.984, 2.026, 2.081
1.58	×	−0.05520	1.58	×	−0.06240	1.230, 1.300, 1.264
1.33	✓	−0.10978	1.33	✓	−0.06215	0.857, 0.936, 0.900
1.143	✓	−0.11103	1.143	✓	−0.06220	0.634, 0.697, 0.662
≈1.00	✓	**−0.11156**	**≈1.00**	✓	**−0.06339**	**0.486, 0.534, 0.506**
0.89	✓	−0.11169	0.89	✓	−0.06344	0.385, 0.424, 0.400
0.80	✓	−0.11171	0.80	✓	−0.06345	0.313, 0.342, 0.324

depends on the overlap of the influence domains of the discrete Dirac delta function [39] currently utilized in both the regularization and interpolation operators.

A.1. Numerical setup

The study was conducted using the same physical and numerical configuration as in Section 6.1.3, namely sedimentation of a spherical particle in a viscous fluid. The Eulerian grid resolution was fixed to $200 \times 200 \times 400$, while the spacing of the Lagrangian markers was varied systematically. The simulations were initialized with the particle at rest and advanced for the first 100 time steps. For each configuration, the following quantities were evaluated:

- the ratio $\Delta h_{Eu}/\Delta h_L$ between the Eulerian and Lagrangian grid spacings,
- the minimum, maximum, and average row (or column) sums of the explicitly constructed matrix $R^T R$,
- convergence or divergence of the time integration,
- the value of sedimentation velocity v_z,
- the number of BiCGStab iterations required for convergence of the system governing the pressure and force-density corrections (see Eq. (20)).

A.2. Stability, accuracy, and convergence

The results of the convergence and stability study are summarized in Table A.1. For dense Lagrangian discretizations ($\Delta h_{Eu}/\Delta h_L \geq 1.58$), the support of the discrete delta functions overlaps strongly, leading to near-singular behavior of the operator $R^T R$. In this regime, the approximation $(R^T R)^{-1} \approx 0.5I$ is no longer valid, and the simulations diverge after a small number of time steps.

For coarser Lagrangian grids ($\Delta h_{Eu}/\Delta h_L \leq 1.33$), the simulations remain stable. In this regime, the diagonal approximation provides an accurate representation of the matrix–vector product involving $(R^T R)^{-1}$. The accuracy of the obtained results was assessed by computing the relative deviation of the sedimentation velocity v_z after 100 time steps with respect to a reference configuration characterized by approximately equal Eulerian and Lagrangian grid resolutions (highlighted in bold in Table A.1). For all converged cases, the deviation of v_z remained below approximately 2%, indicating that the diagonal approximation does not compromise accuracy provided that the Lagrangian spacing is neither excessively fine nor excessively coarse.

For the two densest configurations, stability could be recovered by replacing the constant diagonal approximation with a rescaled form $(R^T R)^{-1} \approx (R^T R)^{-1}_{\text{avg}} I$. However, this correction was found to be effective only at higher Reynolds numbers, whereas at $Re = 11.6$ the relative error between the reference solution and the results obtained using the rescaled approximation reached approximately 50%. An important observation is that the performance of the preconditioner is essentially independent of the Lagrangian discretization. In all tested cases, including those that eventually diverged, the number of BiCGStab iterations required for convergence of the pressure-force correction system (Eq. (20)) remained between 4 and 5. These presented results define the accuracy limits for applying the diagonal approximation of the matrix-vector product $(R^T R)^{-1}$, and identify the practical range of applicability of the proposed method.

Appendix B. Influence of boundary proximity on the preconditioner performance

To further assess the limits of applicability of the developed preconditioner, we examine its performance in configurations where the immersed body is located in close proximity to the computational boundary. Several points should be clarified in this context. First, in order to ensure physical consistency of the interpolation and regularization operators, the currently utilized discrete Dirac delta function must satisfy a number of fundamental properties, including consistent interpolation of uniform fields and conservation of linear and angular momentum of moving bodies (see, e.g., Roma [39]). These properties require that the delta function be applied symmetrically with respect to the computational grid. In particular, the region to which a physical quantity is interpolated must be

Table B.1
Period-averaged BiCGStab iteration count for solving the preconditioned system at different Reynolds numbers and grid resolutions and A/D ratios, with convergence tolerance of 10^{-12}. Results shown for the configuration of a single sphere transversely oscillating in a proximity of vertical wall.

Amplitude ratio	Grid size	Re = 10	Re = 20	Re = 50	Re = 100	Re = 200	Re = 300
$A/D = 0.5$	50×50×50	4.02	4.00	4.00	4.01	4.00	4.00
	100×100×150	4.08	4.11	4.07	4.04	4.05	4.09
	200×200×300	4.63	4.46	4.24	4.24	4.27	4.37
	400×400×600	4.56	4.58	4.55	4.41	4.22	4.17
$A/D = 1.0$	50×50×50	4.01	4.01	4.00	4.00	4.00	4.00
	100×100×150	4.22	4.15	4.13	4.14	4.24	4.26
	200×200×300	4.81	4.70	4.51	4.44	4.56	4.70
	400×400×600	4.20	4.48	4.51	4.54	4.38	4.73
$A/D = 1.5$	50×50×50	4.01	4.01	4.00	4.00	4.00	4.00
	100×100×150	4.18	4.17	4.11	4.17	4.29	4.23
	200×200×300	4.61	4.61	4.61	4.62	4.61	4.61
	400×400×600	4.29	4.65	4.63	4.64	4.69	4.78

surrounded by Eulerian grid points extending at least over the full support of the discrete delta function. Conversely, when regularizing a Lagrangian quantity onto the Eulerian grid, the computational domain must extend by at least one delta-function support width beyond the location of the Lagrangian point.

For this reason, the application of direct-forcing immersed boundary methods incorporating symmetric delta functions requires special treatment in situations where an immersed body approaches a physical boundary or another solid object. A commonly adopted approach is the introduction of short-range conservative repulsion forces (see, e.g., [59–61]). These forces ensure that Lagrangian points do not approach each other more closely than approximately the support width of the corresponding delta function, thereby preserving the validity of the interpolation and regularization operators. An alternative approach, apparently providing a more physically consistent scenario, is to employ discrete delta functions with a one-sided kernel such as those proposed in [16,62,63]. In the present study, we restrict ourselves to the use of symmetric delta functions only.

When utilizing symmetric delta functions together with repulsion forces, there is no fundamental distinction between interactions of an immersed body with the boundary of the computational domain and interactions between two independently moving immersed bodies. In both cases, the repelling mechanism enforces a minimal separation distance. For this reason, explicit simulations of multiple independently moving bodies do not introduce additional conceptual challenges beyond those already present in the body–boundary interaction case. Taking these considerations into account, we have chosen to focus on configurations in which an immersed body remains in close proximity to a boundary for an extended period of time, or, in the limiting case, throughout the entire simulation. To investigate this scenario, we therefore consider an additional test involving a transversely oscillating sphere whose surface is located at a distance of only two grid cells from the horizontal boundary. This distance corresponds to the minimal separation at which repulsion forces are not yet activated and at the same time guarantees the correct application of the discrete operators $G^T R$ and $R^T G$. All remaining governing parameters, including the Reynolds number and the ratio between the oscillation amplitude, A, and the particle diameter, D, are chosen to be identical to those used in Section 6.1.1 of the main text. All the velocity boundary conditions were set as no-slip.

The results of the period-averaged BiCGStab iteration count for different Re and A/D values are summarized in Table B.1. The table shows that the number of iterations of the preconditioned solver remains nearly constant, within the range of approximately 4 to 5, for the entire set of operating parameters. This behavior is consistent with that observed for configurations simulated far from boundaries in the main text of the paper. These results verify the efficiency of the developed preconditioned solver also for configurations involving body–boundary interactions.

References

[1] S. Tschisgale, T. Kempe, J. Fröhlich, A general implicit direct forcing immersed boundary method for rigid particles, Comput. Fluids 170 (2018) 285–298.
[2] Y. Mori, C.S. Peskin, Implicit second-order immersed boundary methods with boundary mass, Comput. Methods Appl. Mech. Eng. 197 (25-28) (2008) 2049–2067.
[3] D.V. Le, J. White, J. Peraire, K.M. Lim, B.C. Khoo, An implicit immersed boundary method for three-dimensional fluid–membrane interactions, J. Comput. Phys. 228 (2009) 8427–8445.
[4] J.J. Wagner, C.F. Higgs, III, Coupled CFD-DEM simulation of interfacial fluid–particle interaction during binder jet 3D printing, Comput. Methods Appl. Mech. Eng. 421 (2024) 116747.
[5] Z. Li, X.-R. Huang, L. Fang, Numerical modeling of fluid–structure–piezoelectric interaction for energy harvesting, Comput. Methods Appl. Mech. Eng. 414 (2023) 116164.
[6] M.A. Fernández, Coupling schemes for incompressible fluid-structure interaction: implicit, semi-implicit and explicit, SeMA J. 55 (2011) 59–108.
[7] R.L. Muddle, M. Mihajlovic, M. Heil, An efficient preconditioner for monolithically-coupled large-displacement fluid-structure interaction problems with pseudo-solid mesh updates, J. Comput. Phys. 231 (2) (2012) 731–764.
[8] Y. Feldman, Y. Guldberg, An extension of the immersed boundary method based on the distributed Lagrange multiplier approach, J. Comput. Phys. 322 (2016) 248–266.
[9] D.B. Stein, R.D. Guy, B. Thomases, Immersed boundary smooth extension: a high-order method for solving PDE on arbitrary smooth domains using Fourier spectral methods, J. Comput. Phys. 304 (2016) 252–274.

[10] 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.
[11] B. Kallemov, A. Bhalla, B. Griffith, A. Donev, An immersed boundary method for rigid bodies, Comm. App. Math. Comp. Sci. 11 (1) (2016) 79–141.
[12] N.J. Nair, A. Goza, A strongly coupled immersed boundary method for fluid-structure interaction that mimics the efficiency of stationary body methods, J. Comput. Phys. 454 (2022) 110897.
[13] J. Mohd-Yusof, For simulations of flow in complex geometries, Ann. Res. Briefs 317 (1997) 35.
[14] 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 (1) (2000) 35–60.
[15] J. Wu, C. Shu, Implicit velocity correction-based immersed boundary-lattice Boltzmann method and its applications, J. Comput. Phys. 228 (2009) 1963–1979.
[16] M. Vanella, E. Balaras, A moving-least-squares reconstruction for embedded-boundary formulations, J. Comput. Phys. 228 (2009) 6617–6628.
[17] A. Posa, M. Vanella, E. Balaras, An adaptive reconstruction for Lagrangian, direct-forcing, immersed-boundary methods, J. Comput. Phys. 351 (2017) 422–436.
[18] I.N. Yildiran, N. Beratlis, F. Capuano, Y.-H. Loke, K. Squires, E. Balaras, Pressure boundary conditions for immersed-boundary methods, J. Comput. Phys. 510 (2024) 113057.
[19] E. Farah, A. Ouahsine, P.G. Verdin, An improved implicit direct-forcing immersed boundary method (DF-IBM) around arbitrarily moving rigid structures, Phys. Fluids 36 (10) (2024) 103618.
[20] K. Kimmo, S.B. Amneet, Pal, S. Brennan, W. Ning, T. Nils, A direct forcing, immersed boundary method for conjugate heat transport, J. Comput. Phys. 538 (2025) 114135.
[21] K. Taira, T. Colonius, The immersed boundary method: a projection approach, J. Comput. Phys. 225 (2) (2007) 2118–2137.
[22] D.V. Le, B.C. Khoo, K.M. Lim, An implicit-forcing immersed boundary method for simulating viscous flows in irregular domains, Comput. Methods Appl. Mech. Eng. 197 (25-28) (2008) 2119–2130.
[23] C. Wang, J.D. Eldredge, Strongly coupled dynamics of fluids and rigid-body systems with the immersed boundary projection method, J. Comput. Phys. 295 (2015) 87–113.
[24] U. Lācis, K. Taira, S. Bagheri, A stable fluid-structure-interaction solver for low-density rigid bodies using the immersed boundary projection method, J. Comput. Phys. 305 (2016) 300–318.
[25] K.C. Ong, Y. Seol, M.-C. Lai, An immersed boundary projection method for solving the fluid-rigid body interaction problems, J. Comput. Phys. 466 (2022) 111367.
[26] R.-Y. Li, C.-M. Xie, W.-X. Huang, C.-X. Xu, An efficient immersed boundary projection method for flow over complex/moving boundaries, Comput. Fluids 140 (2016) 122–135.
[27] K.C. Ong, M.-C. Lai, An immersed boundary projection method for simulating the inextensible vesicle dynamics, J. Comput. Phys. 408 (2020) 109277.
[28] K.C. Ong, M.-C. Lai, Y. Seol, An immersed boundary projection method for incompressible interface simulations in 3D flows, J. Comput. Phys. 430 (2021) 110090.
[29] T. Xu, J.-I. Choi, Efficient monolithic immersed boundary projection method for incompressible flows with heat transfer, J. Comput. Phys. 477 (2023) 111929.
[30] K. Goncharuk, O. Oshri, Y. Feldman, The immersed boundary method: a SIMPLE approach, J. Comput. Phys. 487 (2023) 112148.
[31] E. Constant, J. Favier, M. Meldi, P. Meliga, E. Serre, An immersed boundary method in openFOAM: verification and validation, Comput. Fluids 157 (2017) 55–72.
[32] M. Askarishahi, Immersed-boundary/soft-sphere method for particle-particle-fluid interaction in a viscous flow: an openFOAM solver, Adv. Powder Technol. 34 (2023) 104204.
[33] D. Boffi, L. Gastaldi, Existence, uniqueness, and approximation of a fictitious domain formulation for fluid-structure interactions, Rend. Lincei 33 (1) (2022) 109–137.
[34] A. de Castro, H. Lee, M.M. Wiecek, A Lagrange multiplier method for fluid-structure interaction: well-posedness and domain decomposition, Comput. Math. Appl. 181 (2025) 193–215.
[35] E.P. Newren, A.L. Fogelson, R.D. Guy, R.M. Kirby, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2) (2007) 702–719.
[36] Q. Zhang, R.D. Guy, B. Philip, A projection preconditioner for solving the implicit immersed boundary equations, Numer. Math. Theory Methods Appl. 7 (4) (2014) 473–498.
[37] H. Park, X. Pan, C. Lee, J.-I. Choi, A pre-conditioned implicit direct forcing based immersed boundary method for incompressible viscous flows, J. Comput. Phys. 314 (2016) 774–799.
[38] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys. 209 (2) (2005) 448–476.
[39] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (2) (1999) 509–534.
[40] T. Kempe, J. Fröhlich, An improved immersed boundary method with direct forcing for the simulation of particle laden flows, J. Comput. Phys. 231 (9) (2012) 3663–3684.
[41] S.V. Patankar, D.B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, in: Numerical Prediction of Flow, Heat Transfer, Turbulence and Combustion, Elsevier, 1983, pp. 54–73.
[42] R.E. Lynch, J.R. Rice, D.H. Thomas, Direct solution of partial difference equations by tensor product methods, Numer. Math. 6 (1964) 185–199.
[43] H. Vitoshkin, A.Y. Gelfgat, On direct and semi-direct inverse of Stokes, Helmholtz and Laplacian operators in view of time-stepper-based Newton and Arnoldi solvers in incompressible CFD, Commun. Comput. Phys. 14 (4) (2013) 1103–1119.
[44] J.B. Perot, An analysis of the fractional step method, J. Comput. Phys. 108 (1993) 51–58.
[45] H.C. Elman, D.J. Silvester, A.J. Wathen, Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics, Oxford university press, 2014.
[46] Y. Saad, Numerical Methods for Large Eigenvalue Problems: Revised Edition, SIAM, 2011.
[47] R. Sela, E. Zemach, Y. Feldman, A semi-implicit direct forcing immersed boundary method for periodically moving immersed bodies: a Schur complement approach, Comput. Methods Appl. Mech. Eng. 373 (2021) 113498.
[48] W.-P. Breugem, A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows, J. Comput. Phys. 231 (13) (2012) 4469–4498.
[49] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys. 209 (2005) 448–476.
[50] H.M. Blackburn, Mass and momentum transport from a sphere in steady and oscillatory flows, Phys. Fluids 14 (11) (2002) 3997–4011.
[51] P. Leopardi, A partition of the unit sphere into regions of equal area and small diameter, Electron. Trans. Numer. Anal. 25 (2006) 309–327.
[52] C. Li, M. Ye, Z. Liu, On the rotation of a circular porous particle in 2D simple shear flow with fluid inertia, J. Fluid Mech. 808 (2016) R3.
[53] P. Yu, Y. Zeng, T.S. Lee, X.B. Chen, H.T. Low, Numerical simulation on steady flow around and through a porous sphere, Int. J. Heat Fluid Flow 36 (2012) 142–152.
[54] L. Ma, S. Xu, X. Li, Q. Guo, D. Gao, Y. Ding, M. Ye, Z. Liu, Particle tracking velocimetry of porous sphere settling under gravity: preparation of the model porous particle and measurement of drag coefficients, Powder Technol. 360 (2020) 241–252.
[55] W. Huang, Y. Liang, Serial symmetrical relocation algorithm for the equal sphere packing problem, arXiv preprint arXiv:1202.4149 (2012).
[56] A.t. Cate, C.H. Nieuwstad, J.J. Derksen, H.E.A. V.d. Akker, Particle imaging velocimetry experiments and lattice-Boltzmann simulations on a single sphere settling under gravity, Phys. Fluids 14 (11) (2002) 4012–4025.
[57] N. Mordant, J.F. Pinton, Velocity measurement of a settling sphere, Eur. Phys. J. B 18 (2) (2000) 343–352.
[58] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comput. 31 (138) (1977) 333–390.
[59] B. Maury, A many-body lubrication model, C.R. Acad. Sci., Ser. I 325 (1997) 1053–1058.

[60] R. Glowinski, T.-W. Pan, T. Hesla, D.D. Joseph, J. Périaux, A distributed Lagrange multiplier/fictitious domain method for flows around moving rigid bodies: application to particulate flows, Int. J. Multiphase Flow 25 (1999) 755–794.
[61] R. Glowinski, T.-W. Pan, T. Hesla, D.D. Joseph, A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies, Int. J. Numer. Methods Fluids 31 (1999) 109–129.
[62] M. Haji Mohammadi, F. Sotiropoulos, J. Brinkerhoff, Moving least squares reconstruction for sharp interface immersed boundary methods, Int. J. Numer. Methods Fluids 90 (11) (2019) 6117–6138.
[63] R. Bale, A.P.S. Bhalla, B.E. Griffith, M. Tsubokura, A one-sided direct forcing immersed boundary method using moving least squares, J. Comput. Phys. 440 (2021) 110359.