Moving immersed boundary method

Shang-Gui Cai, Abdellatif Ouahsine, Julien Favier, Yannick Hoarau

Moving immersed boundary method

Shang-Gui Cai[1,*,†], Abdellatif Ouahsine[1], Julien Favier[2], Yannick Hoarau[3]

[1]*Sorbonne Universités, Université de Technologie de Compiègne, CNRS, UMR 7337 Roberval, Centre de Recherche Royallieu, CS 60319, 60203 Compiègne Cedex, France*
[2]*Aix Marseille Université, CNRS, Centrale Marseille, M2P2 UMR 7340, 13451 Marseille, France*
[3]*Université de Strasbourg, CNRS, ICUBE UMR 7357, 67000 Strasbourg, France*

SUMMARY

A novel implicit immersed boundary method of high accuracy and efficiency is presented for the simulation of incompressible viscous flow over complex stationary or moving solid boundaries. A boundary force is often introduced in many immersed boundary methods to mimic the presence of solid boundary, such that the overall simulation can be performed on a simple Cartesian grid. The current method inherits this idea and considers the boundary force as a Lagrange multiplier to enforce the no-slip constraint at the solid boundary, instead of applying constitutional relations for rigid bodies. Hence excessive constraint on the time step is circumvented and the time step only depends on the discretization of fluid Navier-Stokes equations, like the CFL condition in present work. To determine the boundary force an additional moving force equation is derived. The dimension of this derived system is proportional to the number of Lagrangian points describing the solid boundaries, which makes the method very suitable for moving boundary problems since the time for matrix update and system solving is not significant. The force coefficient matrix is made symmetric and positive definite so that the conjugate gradient method can solve the system quickly. The proposed immersed boundary method is incorporated into the fluid solver with a second order accurate projection method as a plug-in. The overall scheme is handled under an efficient fractional step framework, namely prediction, forcing and projection. Various simulations are performed to validate current method and the results compare well with previous experimental and numerical studies.

Received . . .

KEY WORDS: Immersed boundary method; Projection method; Fractional step method; Implicit scheme; Lagrange multiplier; Incompressible viscous flow

1. INTRODUCTION

The immersed boundary method (IBM) has emerged in recent years as an alternative to traditional body-conforming mesh method for simulating fluid flows over complex and moving objects. Through adopting an appropriate boundary force in fluid equations for the presence of immersed

solid boundaries, the simulations can be performed on a very simple Cartesian mesh. This significantly eases complicated mesh generations and eliminates moving boundary related issues, such as mesh distortions and mesh interpolation errors due to deforming-mesh and re-meshing.

Since first introduced by Peskin [1] for modeling blood flow through a beating heart, IBM has been extended to various applications in scientific and engineering fields. In the original method, the immersed elastic membrane is represented by a series of massless Lagrangian markers where the boundary force is evaluated by using constitutive laws. Discretized delta functions are employed as kernel functions for the data exchange between the two independent meshes of fluid and solid. The immersed finite element method (IFEM) [2, 3, 4] was later developed in finite element formulations for general structures that occupy finite volumes within the fluid domain.

Previous methods are well suited for deformable solids owing to their physical basis, but the constitutive laws are generally not well posed when solids reach the rigid limit. Beyer and LeVeque [5] provided a solution by using a spring to attach the solids to an equilibrium location with a restoring force. Goldstein *et al.* [6] and Saiki and Biringen [7] also proposed a feedback forcing strategy to control the velocity near the objects, which behaves as a system of springs and dampers. Nevertheless, artificial constants are introduced, which are *ad hoc* and should be chosen large enough in order to accurately impose the no-slip boundary condition. However large value makes the system very stiff and results in instabilities. The time step is severely limited, leading to a CFL number several magnitude smaller than the usual one [6, 8]. Mohd-Yosuf [9] and Fadlun *et al.* [8] proposed the direct forcing immersed boundary method to avoid the use of artificial constants via modifying the discrete momentum equation. No additional constraints are introduced to the time step. Instead of using the discrete delta function for velocity interpolation and force distribution, local velocity reconstruction approaches were employed to enforce the boundary condition. However Uhlmann [10] observed strong oscillations towards the boundary force. He attributed this problem to insufficient smoothing and re-used the discrete delta function in his direct forcing immersed boundary method. Although other strategies have also been proposed to enhance the local velocity reconstruction, special treatment should be taken for the phase change of cells near the moving boundaries [11, 12, 13].

In fact the boundary force is an unknown that is strongly coupled to the fluid velocity field. In the work of Uhlmann [10], the boundary force is calculated explicitly by a tentative fluid velocity. The no-slip boundary condition can never be satisfied and large errors occur near the immersed boundaries. Kempe and Fröhlich [14] reduced the error by adding a forcing loop within a few iterations. Further improvement of the boundary condition imposition, however, requires numerous iterations for convergence as the multidirect forcing immersed boundary method [15, 16]. Taira and Colonius [17] proposed the implicit immersed boundary projection method (IBPM) by formulating the boundary force and the pressure into a modified Poisson equation and solving them simultaneously in an enlarged system with sophisticated solvers. Despite the mathematical completeness and rigour, IBPM may have convergence problems when an immersed boundary point is very close to a fluid grid point, as the singular property of the interpolation and distribution functions deteriorates significantly the condition number of the coefficient matrix of the original well-defined pressure Poisson equation (PPE) [18].

In this paper we propose the moving immersed boundary method (MIBM) to optimally maintain the accuracy of the implicit IBPM and the efficiency of the explicit direct forcing IBM. The

projection method is served as the basic fluid solver where the proposed MIBM is integrated as a plug-in. Analogous to the role of the pressure in the projection method to satisfy the divergence-free condition, the boundary force is regarded as another Lagrange multiplier for the no-slip constraint in proposed MIBM. The global scheme follows the fractional step fashion and the fluid velocity, pressure and the boundary force are solved sequentially through the idea of operator splitting [19, 20, 21]. We follow the derivation of PPE in the projection method and derive an additional moving force equation for the boundary force. Therefore, the PPE is unchanged and immune from the convergence problem. Moreover, the force coefficient matrix is formulated to be symmetric and positive-definite so that generic linear system solvers can be applied directly.

The organization of this paper is as follows. First the fluid Navier-Stokes equations are discretized and a second order projection method is introduced as our fundamental fluid solver. In the following, the MIBM is presented in details and compared to other immersed boundary methods. In Section 5 a couple of numerical simulations are performed to validate the proposed MIBM. Finally, conclusions are drawn in Section 6.

2. FLUID EQUATIONS AND DISCRETIZATION

Consider the non-dimensionalized Navier-Stokes equations for an incompressible viscous fluid

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = -\nabla p + \dfrac{1}{Re}\nabla^2 \mathbf{u} \quad \text{in } \Omega \times [0,T], \\ \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega \times [0,T], \\ \mathbf{u}|_\Gamma = \mathbf{w} \quad \text{in } [0,T], \\ \mathbf{u}|_{t=0} = \mathbf{u}_0 \quad \text{in } \Omega, \end{cases} \tag{1}$$

where $\mathbf{u}$, p are the fluid velocity vector and the pressure. $Re = UL/\nu$ designates the Reynolds number, based on the reference velocity U, the reference length L and the kinematic viscosity ν.

Directly solving above equations is very difficult. First the equations are non-linear due to the convective terms; Secondly there is no equation to compute the pressure directly; Moreover the pressure and the velocity are coupled through the continuity (incompressibility or divergence-free) condition, and the pressure is often regarded as a Lagrange multiplier to satisfy this constraint; Besides the solution of the pressure is not unique and is determined up to an additive constant. Numerical solutions will be discussed in this paper for overcoming these difficulties.

The above equations are discretized in space on a staggered mesh in order to prevent the so-called even/odd decoupling or checkerboard effect, as shown in Figure 1. The spatial derivatives are approximated by second-order central differences.

To discretize the equations in time, fully implicit scheme is superior to explicit one in terms of stability, which in turn requires non-linear iterations. This could be expensive in computation and the convergence is not always ensured. Non-linear iterations can be avoided by linearizing the convective terms, resulting in a non-symmetric coefficient matrix for the velocity. The matrix needs to be re-computed at each time step, which becomes very costly when the grid number increases. Fully explicit formulation seems to be very efficient as no iterations are needed. But the time step

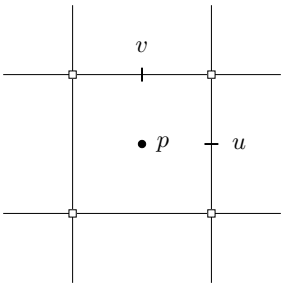


Figure 1. Staggered mesh arrangement for the pressure and the velocity.

should be kept small enough to maintain stability. In two dimensions the constraints on the time step are the diffusive stability condition

$$\Delta t \leqslant \frac{Re}{2}\left(\frac{1}{\Delta x_{\min}^2}+\frac{1}{\Delta y_{\min}^2}\right)^{-1}, \tag{2}$$

and the convective stability condition of the usual CFL (Courant–Friedrichs–Lewy) type

$$\Delta t \leqslant \min\left\{\frac{\Delta x_{\min}}{u_{\max}}, \frac{\Delta y_{\min}}{v_{\max}}\right\}. \tag{3}$$

It is easy to see that the diffusive constraint is more severe. Reducing the mesh size by half requires a four times smaller time step and it becomes more severe as the dimension increases. At low Reynolds number regime, the time constraint due to (2) dominates (3). It might be thought that for moderate to high Reynolds number flows, the diffusive stability condition is less restrictive. However in practice, the grid spacing is usually kept small under these circumstances for capturing small turbulence and the time step constraint of (2) is proportional to the square of the minimal mesh size.

In the present work, a semi-implicit time discretization scheme is employed, namely the convective terms are treated explicitly for avoiding the nonlinearity while the diffusive terms are treated implicitly for circumventing the severe diffusive time constraint. As a result, the entire system is linear and stable under the standard CFL condition. The velocity coefficient matrix remains symmetric and constant. To obtain a second order accurate system, we employ the second order Adams-Bashforth (AB2) scheme for the non-linear terms and the Crank-Nicolson (CN) scheme for the linear terms. The system now can be written as

$$\begin{cases} \dfrac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}+\left[\dfrac{3}{2}\mathcal{N}(\mathbf{u}^n)-\dfrac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right]=-\mathcal{G}p^{n+1}+\dfrac{1}{2Re}\mathcal{L}(\mathbf{u}^{n+1}+\mathbf{u}^n), \\ \mathcal{D}\mathbf{u}^{n+1}=0, \\ \mathbf{u}^{n+1}|_\Gamma=\mathbf{w}^{n+1}, \end{cases} \tag{4}$$

where $\mathcal{L}, \mathcal{N}, \mathcal{G}, \mathcal{D}$, are the discretized linear, non-linear, gradient, divergence operators, respectively. The superscript $n+1$ and n represent the current time level and the past time level. The initial condition is hereafter omitted for convenience.

3. PROJECTION METHOD

The projection method, also refereed to fractional step method or time-splitting method, emerged in late 1960s as an effective tool to solve the pressure-velocity coupling problem, by splitting the system into a serial decoupled elliptic equations. The projection method is rooted in the Helmholtz-Hodge decomposition, which states that any smooth vector field $\mathbf{v}$ could be decomposed into the sum of a divergence-free part and a gradient of a potential field

$$\mathbf{v} = \mathbf{v}_d + \mathcal{G}\phi, \tag{5}$$

where ϕ is often related to the pressure in the projection method. By taking the divergence of (5) and applying $\mathcal{D}\mathbf{v}_d = 0$, ϕ is the solution of the following Poisson equation

$$\mathcal{L}\phi = \mathcal{D}\mathbf{v}. \tag{6}$$

Once ϕ is calculated, the solenoidal velocity can be recovered by

$$\mathbf{v}_d = \mathbf{v} - \mathcal{G}\phi. \tag{7}$$

3.1. *Previous projection methods*

The original projection method proposed by Chorin [22] and Témam [23] decouples the dynamic momentum equation from the kinematic incompressibility constraint by first estimating a tentative velocity $\hat{\mathbf{u}}$ regardless of the pressure term, and then using the pressure to project the predicted velocity $\hat{\mathbf{u}}$ into its solenoidal part $\mathbf{u}^{n+1}$. The two sub-steps of prediction and projection are performed as

$$\begin{cases} \dfrac{\hat{\mathbf{u}} - \mathbf{u}^n}{\Delta t} + \left[\dfrac{3}{2}\mathcal{N}(\mathbf{u}^n) - \dfrac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right] = \dfrac{1}{2Re}\mathcal{L}(\hat{\mathbf{u}} + \mathbf{u}^n), \\ \hat{\mathbf{u}}|_\Gamma = \mathbf{w}^{n+1}, \end{cases} \tag{8}$$

$$\begin{cases} \dfrac{\mathbf{u}^{n+1} - \hat{\mathbf{u}}}{\Delta t} = -\mathcal{G}\phi^{n+1}, \\ \mathcal{D}\mathbf{u}^{n+1} = 0, \\ \mathbf{u}^{n+1} \cdot \mathbf{n}|_\Gamma = \mathbf{w}^{n+1} \cdot \mathbf{n}. \end{cases} \tag{9}$$

By taking divergence of the first equation of (9) along with the incompressibility constraint, the actual realization of the projection step follows

$$\begin{cases} \mathcal{L}\phi^{n+1} = \dfrac{1}{\Delta t}\mathcal{D}\hat{\mathbf{u}}, \\ \dfrac{\partial \phi^{n+1}}{\partial \mathbf{n}}|_\Gamma = 0, \\ \mathbf{u}^{n+1} = \hat{\mathbf{u}} - \Delta t \mathcal{G}\phi^{n+1}, \end{cases} \tag{10}$$

which is the same as (6) and (7) when Δt is absorbed to ϕ^{n+1}. In the projection method of Chorin [22] and Témam [23], the final pressure is set to $p^{n+1} = \phi^{n+1}$.

In spite of its efficiency, the original projection method suffers an irreducible splitting error of $\mathcal{O}(\Delta t)$, which deteriorates the original second order time discretization and prevents its extension to a higher-order method [24, 25, 26]. The error term can be found by adding the two sub-steps (8) and (9), and then comparing to (4)

$$\frac{1}{2Re}\mathcal{L}(\hat{\mathbf{u}}-\mathbf{u}^{n+1})=\frac{\Delta t}{2Re}\mathcal{L}\mathcal{G}p^{n+1}, \tag{11}$$

which is due to the time splitting scheme with the implicit treatment of the diffusive terms. Explicit treatment, however, would result in a severe limitation on the time step. It is rather natural to apply the physical boundary condition to the intermediate velocity $\hat{\mathbf{u}}$ in the prediction step (8). As a result, an artificial Neumann boundary condition $\partial p^{n+1}/\partial \mathbf{n}|_\Gamma = 0$ is enforced on the pressure. This artificial homogeneous Neumann boundary condition introduces a numerical boundary layer to the solution, which prevents the method to be fully first-order [25, 27, 26].

Improvements to the original projection method have been proposed in [28, 29, 30] to achieve a higher order time accuracy by using an incremental scheme, which can be summarized as

$$\begin{cases} \dfrac{\hat{\mathbf{u}}-\mathbf{u}^n}{\Delta t}+\left[\dfrac{3}{2}\mathcal{N}(\mathbf{u}^n)-\dfrac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right]=\dfrac{1}{2Re}\mathcal{L}(\hat{\mathbf{u}}+\mathbf{u}^n)-\mathcal{G}p^n, \\ \hat{\mathbf{u}}|_\Gamma=\mathbf{w}^{n+1}, \end{cases} \tag{12}$$

$$\begin{cases} \dfrac{\mathbf{u}^{n+1}-\hat{\mathbf{u}}}{\Delta t}=-\mathcal{G}\phi^{n+1}, \\ \mathcal{D}\mathbf{u}^{n+1}=0, \\ \mathbf{u}^{n+1}\cdot\mathbf{n}|_\Gamma=\mathbf{w}^{n+1}\cdot\mathbf{n}, \end{cases} \tag{13}$$

where an old value of pressure is retained in the prediction step, and ϕ^{n+1} here represents the pseudo pressure. The second sub-step is often performed as

$$\begin{cases} \mathcal{L}\phi^{n+1}=\dfrac{1}{\Delta t}\mathcal{D}\hat{\mathbf{u}}, \\ \dfrac{\partial\phi^{n+1}}{\partial\mathbf{n}}|_\Gamma=0, \\ \mathbf{u}^{n+1}=\hat{\mathbf{u}}-\Delta t\mathcal{G}\phi^{n+1}, \end{cases} \tag{14}$$

and the final pressure is updated by

$$p^{n+1}=p^n+\phi^{n+1}. \tag{15}$$

To study the spitting error, we sum up (12) and (13) and compare to (4). Considering that the pseudo pressure is the approximation of $\phi^{n+1}=p^{n+1}-p^n=\Delta t p_t$, the splitting error is found to be of second order [24, 31]

$$\frac{1}{2Re}\mathcal{L}(\hat{\mathbf{u}}-\mathbf{u}^{n+1})=\frac{\Delta t}{2Re}\mathcal{L}\mathcal{G}\phi^{n+1}=\frac{\Delta t^2}{2Re}\mathcal{L}\mathcal{G}p_t. \tag{16}$$

Note that the physical boundary condition is still assigned to the intermediate velocity in the prediction step (12). The resulting homogeneous Neumann boundary condition of ϕ^{n+1} implies

that

$$\frac{\partial p^{n+1}}{\partial \mathbf{n}}|_\Gamma = \frac{\partial p^n}{\partial \mathbf{n}}|_\Gamma = \cdots = \frac{\partial p^0}{\partial \mathbf{n}}|_\Gamma, \tag{17}$$

is enforced on the final pressure. This pressure boundary condition is not physical, thus it introduces a numerical boundary layer and prevents the scheme to be fully second order [26]. This error is irreducible, hence using a higher order time stepping scheme will not improve the overall accuracy.

3.2. Rotational incremental pressure-correction projection method

To obtain a solution of second order accuracy with consistent boundary conditions, we propose to use the rotational incremental pressure-correction projection method of [32, 26]. The essential idea of this method is to absorb the splitting error into the pressure so that the sum of the sub-steps is consistent with the original discretized momentum equation (4). By considering the identity $\nabla^2\mathbf{u} = \nabla(\nabla\cdot\mathbf{u}) - \nabla\times\nabla\times\mathbf{u}$, the error term (16) can be rewritten as

$$\frac{1}{2Re}\mathcal{L}(\hat{\mathbf{u}} - \mathbf{u}^{n+1}) = \frac{1}{2Re}\mathcal{G}(\mathcal{D}\hat{\mathbf{u}}), \tag{18}$$

where $\nabla\times\nabla\times\hat{\mathbf{u}} = \nabla\times\nabla\times\mathbf{u}^{n+1}$ is used, which can be verified by the Helmholtz-Hodge decomposition. Now the error term in this form can be absorbed into the pressure

$$p^{n+1} = p^n + \phi^{n+1} - \frac{1}{2Re}\mathcal{D}\hat{\mathbf{u}}. \tag{19}$$

Most importantly, the pressure boundary condition is consistent with the original system. Therefore, no numerical boundary layer will be generated with this scheme. Higher than second order accuracy can be achieved if a higher-order time-stepping scheme is used. In the present work, the second order accuracy is found to be sufficient with the AB2 scheme and the CN scheme. The overall rotational incremental pressure-correction projection method can be summarized as follows

$$\begin{cases} \dfrac{\hat{\mathbf{u}} - \mathbf{u}^n}{\Delta t} + \left[\dfrac{3}{2}\mathcal{N}(\mathbf{u}^n) - \dfrac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right] = \dfrac{1}{2Re}\mathcal{L}(\hat{\mathbf{u}} + \mathbf{u}^n) - \mathcal{G}p^n, \\ \hat{\mathbf{u}}|_\Gamma = \mathbf{w}^{n+1}, \end{cases} \tag{20}$$

$$\begin{cases} \mathcal{L}\phi^{n+1} = \dfrac{1}{\Delta t}\mathcal{D}\hat{\mathbf{u}}, \\ \dfrac{\partial\phi^{n+1}}{\partial\mathbf{n}}|_\Gamma = 0, \\ \mathbf{u}^{n+1} = \hat{\mathbf{u}} - \Delta t\mathcal{G}\phi^{n+1}, \end{cases} \tag{21}$$

$$p^{n+1} = p^n + \phi^{n+1} - \frac{1}{2Re}\mathcal{D}\hat{\mathbf{u}}. \tag{22}$$

3.3. Pressure Poisson equation solver and Parallel computing

The aforementioned discretized equations lead to a set of linear systems to be solved. Among them the pressure Poisson equation is the most time-consuming part due to its high condition number, which is generally solved iteratively to save computational time and storage. The multi-grid (MG) method and the Krylov solvers like the conjugate gradient (CG), bi-conjugate gradient stabilized

(Bi-CGSTAB), generalized minimum residual (GMRES), etc., are very efficient for this problem. In addition, more efficiency can be achieved if pre-conditioning is applied, such as the incomplete Cholesky (IC) factorization, the incomplete lower-upper (ILU) decomposition and the approximate inverse (AINV) [33]. The MG method is found to be more efficient when used as a pre-conditioner in conjunction with Krylov solvers instead of a pure solver.

To further improve this work, the code is extended to allow parallel computing. First we integrate our method into the PETSc library [34], which employs the MPI for communications between the CPU cores. In the second mode, we parallelize the code through using the CUDA CUSP library on GPU [35]. In fact the CPU consists of a few cores optimized for sequential serial task, while the GPU may have massive smaller cores at the same price which is extremely efficient for handling multiple tasks simultaneously. Therefore, we send the parallelable and computationally intensive parts of the application to the GPU and run the remainders on the CPU. From the practical point of view, the second mode runs significantly fast.

Methods of parallelization	Cores	CG		CG+MG	
		Time (s)	Speed-up	Time (s)	Speed-up
CPU	1	63.00	1.0	25.25	1.0
	2	30.68	2.05	10.59	2.38
	4	15.26	4.13	4.56	5.23
	8	7.79	8.09	2.13	11.84
	16	4.20	15.00	1.10	22.92
	20	8.88	7.09	1.11	22.71
GPU	240	10.36	6.08	0.63	40.08

Table I. Time consummation and speed-up of the CPU and GPU parallelization for solving the pressure Poisson equation on a 400×400 grid. The tolerance is set to 1×10^{-10}.

Preconditioner	Construction time (s)	Application time (s)	Speed-up	Total time (s)	Speed-up
None	0	10.36	1.0	10.36	1.0
AINV	1.26	6.52	1.59	7.78	1.33
MG	0.51	0.12	86.33	0.63	16.44

Table II. Comparison of different preconditioners in GPU parallelization with the CUSP library, where the CG solver is used for solving the PPE on a 400×400 grid. The tolerance is set to 1×10^{-10}.

Table I illustrates the performances of the two parallelization modes. The test is performed by solving the PPE on a 400×400 grid with the Neumann boundary condition applied at all the boundaries. As a matter of fact, this system does not possess a unique solution (up to an additional constant). We pin a fixed value to one cell to remove the zero eigenvalue, as suggested in [17, 36]. The calculation is done on the platform PILCAM2 with the CPU Intel Xeon X7542 and the GPU Quadroplex 2200 S4. The process time decreases approximately by half when we double the cores of CPU from 1 to 16 in the test with CG solver. About 1.5-2.8 times' acceleration has been achieved when the MG method is applied as a preconditioner for the CPU parallelization. The parallelization of GPU greatly accelerates the calculation up to 40 times in the test with MG preconditioner. Different preconditioners like the AINV and the MG are compared in Table II. Actually since the

pressure coefficient matrix and its pre-conditioner are constructed only in the beginning and kept unchanged during the entire simulation in current method, around 86 times' acceleration is found.

4. MOVING IMMERSED BOUNDARY METHOD

4.1. Mathematical formulation of immersed boundary method

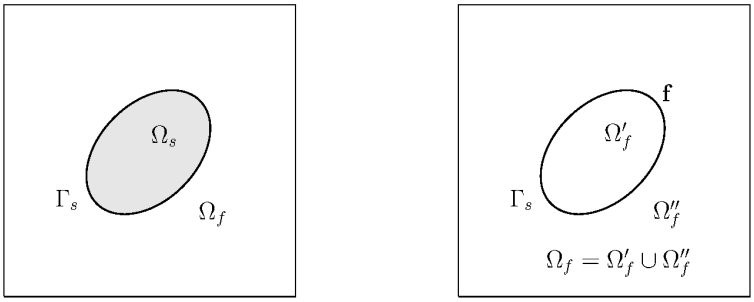
The fundamental idea of the immersed boundary method is to replace solid domain with surrounding fluid so that the whole calculation can be performed on a regular domain. The influence of solid on fluid flow is realized through a boundary force, as shown in Figure 2. The mathematical formulation of the immersed boundary method can be described as

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = -\nabla p + \dfrac{1}{Re}\nabla^2\mathbf{u} + \mathbf{f} \\ \nabla\cdot\mathbf{u} = 0 \\ \displaystyle\int_{\Omega_f} \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(s,t))d\mathbf{x} = \mathbf{U}_b \end{cases} \tag{23}$$

where

$$\mathbf{f}(\mathbf{x},t) = \int_{\Gamma_s} \mathbf{F}(s,t)\delta(\mathbf{x}-\mathbf{X}(s,t))ds \tag{24}$$

where $\mathbf{F}(s,t)$ represents the boundary force defined on the Lagrangian position $\mathbf{X}(s,t)$ and $\mathbf{f}(\mathbf{x},t)$ on the Eulerian frame respectively. δ designates the Dirac delta function. $\mathbf{U}_b$ is the solid boundary velocity.



(a) Original domain (b) Immersed domain

Figure 2. Illustration of the immersed boundary method.

By using an implicit scheme for the boundary force along with previous time discretization of Naiver-Stokes equations, the entire system becomes

$$\begin{cases} \dfrac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} + \left[\dfrac{3}{2}\mathcal{N}(\mathbf{u}^n) - \dfrac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right] = -\mathcal{G}p^{n+1} + \dfrac{1}{2Re}\mathcal{L}(\mathbf{u}^{n+1}+\mathbf{u}^n) + \mathcal{S}(\mathbf{F}^{n+1}) \\ \mathcal{D}\mathbf{u}^{n+1} = 0 \\ \mathcal{T}(\mathbf{u}^{n+1}) = \mathbf{U}_b^{n+1} \end{cases} \tag{25}$$

where $\mathcal{T}$ and $\mathcal{S}$ are the interpolation and spreading operators defined as

$$\mathcal{S}_{\mathbf{X}^{n+1}}(\mathbf{F}^{n+1}) = \int_{\Gamma_s} \mathbf{F}^{n+1}\delta(\mathbf{x}-\mathbf{X}^{n+1})ds \tag{26}$$

$$\mathcal{T}_{\mathbf{X}^{n+1}}(\mathbf{u}^{n+1}) = \int_{\Omega_f} \mathbf{u}^{n+1}\delta(\mathbf{x}-\mathbf{X}^{n+1})d\mathbf{x} \tag{27}$$

Figure 3 shows the discretized computational domain. The immersed boundary is represented by a set of Lagrangian points, whereas the fluid mesh does not necessarily conform the solid geometries.

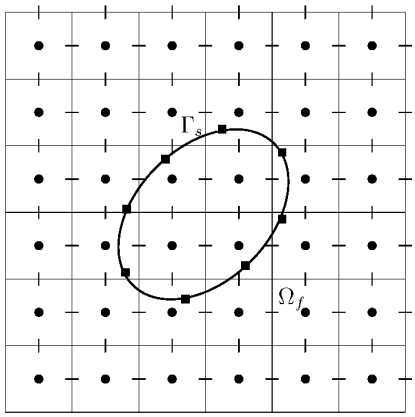


Figure 3. Computational domain of the immersed boundary method. The horizontal and vertical lines designate the components of velocity $\mathbf{u}$ and force $\mathbf{f}$. Pressure is centred on each cell and marked by •. The immersed boundary Γ_s is represented by Lagrangian marker ■ where the force $\mathbf{F}$ is defined.

4.2. Interpolation technique

As the fluid variables are defined separately in space in the staggered mesh, the immersed boundary points can never coincide with the underlying fluid mesh. Therefore it is necessary to interpolate the velocity and spread the force between Eulerian and Lagrangian locations. A great amount of techniques could be used to accomplish this task, ranging from simple linear and bilinear interpolations [37] to more complicated moving least square (MLS) method [38], reproducing kernel particle method (RKPM) [2, 3, 4, 39, 40], radial basis function (RBF) method [41], etc.

In the present study, we select the most frequently used discrete delta function as the kernel function. The two-dimensional discrete delta function is given by a product of one-dimensional functions scaled with a mesh width h, which has the form of

$$\delta_h(\mathbf{x}-\mathbf{X}) = \frac{1}{h^2}\phi(\frac{x-X}{h})\phi(\frac{y-Y}{h}), \tag{28}$$

For three-dimensional case an extra factor of $\frac{1}{h}\phi(\frac{z-Z}{h})$ is needed. For the construction of the one-dimensional function, the most simplest form can be the 2-point-width hat function [5]

$$\phi_2(r) = \begin{cases} 1-|r|, & |r|<1, \\ 0, & \text{otherwise}, \end{cases} \tag{29}$$

which however is not smooth in the support domain. Peskin [42] constructed a smoothed 4-point-width function as follows

$$\phi_4(r) = \begin{cases} \frac{1}{8}\left(3 - 2|r| + \sqrt{1 + 4|r| - 4r^2}\right), & |r| < 1, \\ \frac{1}{8}\left(5 - 2|r| - \sqrt{-7 + 12|r| - 4r^2}\right), & 1 \leqslant |r| < 2, \\ 0, & \text{otherwise}, \end{cases} \tag{30}$$

which is widely used in the literature. Roma *et al.* [43] also designed a 3-point-width function specially for the staggered mesh

$$\phi_3(r) = \begin{cases} \frac{1}{3}\left(1 + \sqrt{-3r^2 + 1}\right), & |r| < 0.5, \\ \frac{1}{6}\left(5 - 3|r| - \sqrt{-3(1 - |r|)^2 + 1}\right), & 0.5 \leqslant |r| < 1.5, \\ 0, & \text{otherwise}. \end{cases} \tag{31}$$

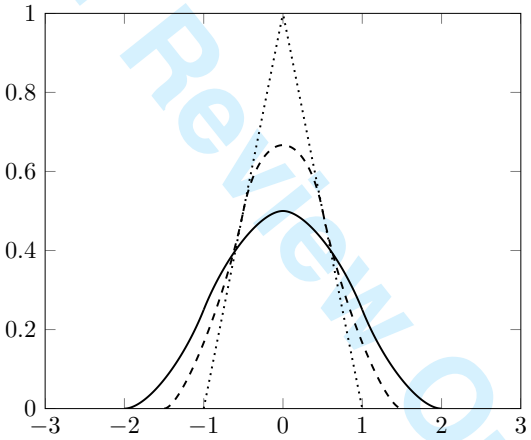


Figure 4. Comparison of the one-dimensional function $\phi(r)$. $\cdots$, the 2-point-width hat function [5]; - - - -, the 3-point-width function of Roma *et al.* [43]; ——, the 4-point-width function of Peskin [42].

The functions are plotted and compared in Figure 4. The one-dimensional function of Roma *et al.* [43] has a relative smaller support than the four-point version of Peskin [42], providing a sharper interface and a better numerical efficiency while maintaining good smoothing properties. The discrete delta functions used in the present work have the following properties:

- δ_h has a narrow support to reduce the computational cost and to obtain a better resolution of the immersed boundary.
- δ_h is second order accurate for smooth fields.
- δ_h satisfies certain moment conditions to meet the translation invariant interpolation rule, namely the total force and torque are equivalent between the Lagrangian and Eulerian locations.

$$\sum_{\mathbf{x} \in g_h} \delta_h(\mathbf{x} - \mathbf{X})h^2 = 1 \quad \text{(zeroth moment condition)}, \tag{32}$$

$$\sum_{\mathbf{x}\in g_h} (\mathbf{x}-\mathbf{X})\delta_h(\mathbf{x}-\mathbf{X})h^2 = 0 \quad \text{(first moment condition)}, \tag{33}$$

where g_h consists of uniformly distributed nodes $\mathbf{x} = (i,j)h$ over the domain Ω_f (i,j are the mesh indices).

4.3. Previous immersed boundary methods

The computation of the boundary force is crucial for the imposition of no-slip condition and distinguishes each kind of immersed boundary methods. Here we start with the direct forcing immersed boundary method proposed by Uhlmann [10], which can be recast in the rotational incremental pressure-correction projection method as follows:

(1) Velocity prediction of all the explicit terms in the discretized momentum equation

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t \left\{ -\left[\frac{3}{2}\mathcal{N}(\mathbf{u}^n) - \frac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right] - \mathcal{G}p^n + \frac{1}{2Re}\mathcal{L}\mathbf{u}^n \right\}. \tag{34}$$

(2) Interpolate the predicted fluid velocity into the immersed interface

$$\mathbf{U}^*(\mathbf{X}_l) = \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} \mathbf{u}^*\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_l)h^2. \tag{35}$$

(3) Evaluate the boundary force on the Lagrangian locations

$$\mathbf{F}^{n+1}(\mathbf{X}_l) = \frac{\mathbf{U}_b^{n+1}(\mathbf{X}_l) - \mathbf{U}^*(\mathbf{X}_l)}{\Delta t}. \tag{36}$$

(4) Spread the boundary force to the surrounding fluid cells

$$\mathbf{f}^{n+1}(\mathbf{x}_{i,j}) = \sum_{l=1}^{n_b} \mathbf{F}^{n+1}(\mathbf{X}_l)\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_l)\Delta V_l. \tag{37}$$

where $\Delta V_l \approx h^2$ is surface associated with the element $\mathbf{X}_l$ by setting the arc-length δs approximate the uniform mesh width h, as shown in Figure 5.

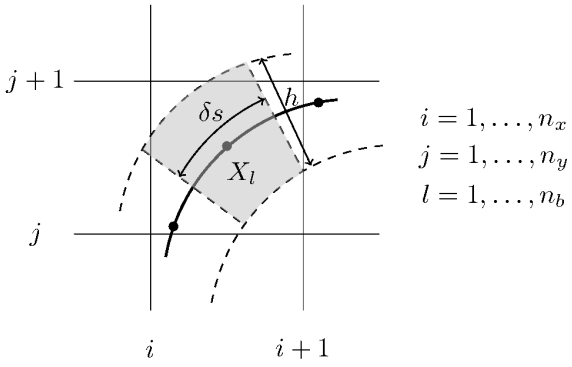


Figure 5. The lth element $\mathbf{X}_l$ and its associated surface $\Delta V_l \approx h^2$ marked by a shaded zone.

(5) Correct the fluid velocity with the boundary force to account for the immersed objects

$$\tilde{\mathbf{u}} = \mathbf{u}^* + \Delta t \mathbf{f}^{n+1}. \tag{38}$$

(6) Implicit treatment of the viscous term

$$\frac{1}{\Delta t}\hat{\mathbf{u}} - \frac{1}{2Re}\mathcal{L}\hat{\mathbf{u}} = \frac{1}{\Delta t}\tilde{\mathbf{u}}. \tag{39}$$

(7) Project the fluid velocity into the divergence-free field and update the pressure

$$\mathcal{L}\phi^{n+1} = \frac{1}{\Delta t}\mathcal{D}\hat{\mathbf{u}}, \tag{40}$$

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} - \Delta t \mathcal{G}\phi^{n+1}, \tag{41}$$

$$p^{n+1} = p^n + \phi^{n+1} - \frac{1}{2Re}\mathcal{D}\hat{\mathbf{u}}. \tag{42}$$

Here $\mathbf{u}^*$, $\tilde{\mathbf{u}}$, $\hat{\mathbf{u}}$, $\mathbf{u}^{n+1}$ represent the fluid velocity at each stage of the fractional step method, i.e., the prediction step of explicit terms, the immersed boundary forcing step, the viscous prediction step, and the projection step. $\mathbf{U}_b(\mathbf{X}_l)$ is the solid velocity of the lth element at the immersed boundary.

The method of Uhlmann [10] is favored in the literature as it is computational inexpensive due to its explicit treatment of the boundary force. However, numerical simulations have shown that it fails to impose the velocity boundary condition exactly on the immersed boundary [14]. A forcing error is introduced which is irreducible and depends on the time step and Reynolds number Re. This error comes from the fact that the tentative fluid velocity $\mathbf{u}^*$ is used for the boundary force evaluation. The ideal velocity should be the final fluid velocity $\mathbf{u}^{n+1}$ while it is unknown at the immersed boundary forcing step. Otherwise we need to iterate the whole system to achieve $\mathbf{u}^{n+1}$ implicitly, which could be too cumbersome to perform. But this implies one way of reducing the forcing error by choosing the closest value to the final velocity.

Kempe and Fröhlich [14] suggested to perform the viscous prediction step first and then use the intermediate velocity $\hat{\mathbf{u}}$ to compute the boundary force. To further improve the accuracy, a forcing loop is added in the immersed boundary forcing step. This additional loop is performed within few iterations without convergence. The method of Kempe and Fröhlich [14] can be expressed as

(1) Prediction of the explicit terms

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t \left\{ -\left[\frac{3}{2}\mathcal{N}(\mathbf{u}^n) - \frac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right] - \mathcal{G}p^n + \frac{1}{2Re}\mathcal{L}\mathbf{u}^n \right\}. \tag{43}$$

(2) Viscous prediction step

$$\frac{1}{\Delta t}\hat{\mathbf{u}} - \frac{1}{2Re}\mathcal{L}\hat{\mathbf{u}} = \frac{1}{\Delta t}\mathbf{u}^*. \tag{44}$$

(3) Immersed boundary forcing loop

Loop for k = 1 to 3 with $\hat{\mathbf{u}}^{(0)} = \hat{\mathbf{u}}$

$$\hat{\mathbf{U}}^{(k)}(\mathbf{X}_l) = \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} \hat{\mathbf{u}}^{(k-1)}\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_l)h^2, \tag{45}$$

$$\mathbf{F}^{(k)}(\mathbf{X}_l) = \frac{\mathbf{U}_b^{n+1}(\mathbf{X}_l) - \hat{\mathbf{U}}^{(k)}(\mathbf{X}_l)}{\Delta t}, \tag{46}$$

$$\mathbf{f}^{(k)}(\mathbf{x}_{i,j}) = \sum_{l=1}^{n_b} \mathbf{F}^{(k)}(\mathbf{X}_l)\delta_h(\mathbf{x}_{i,j} - \mathbf{X}_l)\Delta V_l, \tag{47}$$

$$\tilde{\mathbf{u}}^{(k)} = \hat{\mathbf{u}}^{(k)} + \Delta t \mathbf{f}^{(k)}, \tag{48}$$

$$\hat{\mathbf{u}}^{(k)} = \tilde{\mathbf{u}}^{(k)}. \tag{49}$$

End loop

(4) Projection step and update of the final fields

$$\mathcal{L}\phi^{n+1} = \frac{1}{\Delta t}\mathcal{D}\tilde{\mathbf{u}}, \tag{50}$$

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}} - \Delta t \mathcal{G}\phi^{n+1}, \tag{51}$$

$$p^{n+1} = p^n + \phi^{n+1} - \frac{1}{2Re}\mathcal{D}\hat{\mathbf{u}}. \tag{52}$$

If full convergence of the forcing loop is required more iterations are needed, such as the multidirect forcing scheme of [15, 16]. However, the convergence rate of this iteration becomes very slow after several iterations. The computational cost increases hugely when more Lagrangian points are involved in the additional forcing loop. Therefore the number of iteration is usually kept low for the computational efficiency. Even though the error is reduced, the method of Kempe and Fröhlich [14] is still explicit. The exact no-slip boundary condition can never be satisfied.

To impose the no-slip boundary condition exactly, Taira and Colonius [17] proposed the implicit immersed boundary projection method (IBPM) by combining the boundary force and the pressure into a modified Poisson equation and solving them simultaneously in one single projection step. However convergence problem may occur as one boundary point is very close to a fluid grid point [18], because the singular property of the interpolation and distribution functions undermines the coefficient matrix condition number of the PPE. Ji *et al.* [18] proposed to iterate each part to get rid of the convergence problem, which is inevitable computational expensive.

4.4. Novel implicit immersed boundary method

In this subsection we present a novel implicit but efficient IBM variant, termed as the moving immersed boundary method (MIBM) in this paper. The objective of MIBM to maintain the efficiency of the explicit direct forcing IBM but with an improved accuracy like the multidirect forcing IBM and the IBPM.

To this end, we first take the immersed boundary forcing part from the explicit IBM of Kempe and Fröhlich [14] for consideration, i.e., (45), (46), (47) and (48). By dropping the superscripts for convenience, the immersed boundary forcing part is written as

$$\hat{\mathbf{U}} = \mathcal{T}\hat{\mathbf{u}}, \tag{53}$$

$$\mathbf{F} = \frac{\mathbf{U}_b - \hat{\mathbf{U}}}{\Delta t}, \tag{54}$$

$$\mathbf{f} = \mathcal{S}\mathbf{F}, \tag{55}$$

$$\tilde{\mathbf{u}} = \hat{\mathbf{u}} + \Delta t\mathbf{f}. \tag{56}$$

We require that the interpolated velocity satisfies the no-slip wall boundary condition on the immersed interface after the immersed boundary forcing, namely $\mathcal{T}\tilde{\mathbf{u}} = \mathbf{U}_b$, then

$$\mathcal{T}(\hat{\mathbf{u}} + \Delta t\mathbf{f}) = \mathbf{U}_b. \tag{57}$$

Substituting (55) into (57) gives

$$\mathcal{T}(\hat{\mathbf{u}} + \Delta t\mathcal{S}\mathbf{F}) = \mathbf{U}_b, \tag{58}$$

which can be rearranged in order to separate the boundary force

$$(\mathcal{T}\mathcal{S})\mathbf{F} = \frac{\mathbf{U}_b - \mathcal{T}\hat{\mathbf{u}}}{\Delta t}. \tag{59}$$

We donate $\mathcal{M} = \mathcal{T}\mathcal{S}$ the moving force coefficient matrix. $\mathcal{M}$ is a function of the solid position, which changes its value as the boundary moves. Thus the force is redistributed just like the boundary force moves. The moving force equation can be rewritten in a more concise form

$$\mathcal{M}\mathbf{F} = \mathbf{F}^e, \tag{60}$$

where $\mathbf{F}^e = (\mathbf{U}_b - \mathcal{T}\hat{\mathbf{u}})/\Delta t$ is exactly the explicit forcing value used in [14].

Compared to the modified Poisson equation in the IBPM of [17], the moving force equation (60) is much smaller in size and easier to work with. At each dimension (x or y), the size of the force coefficient matrix is $n_b \times n_b$ since $\mathcal{T} \in \mathbb{R}^{n_b \times n_x n_y}$ and $\mathcal{S} \in \mathbb{R}^{n_x n_y \times n_b}$. Therefore, for moving boundaries, its update is computational less expensive than the modified Poisson equation.

Note that $\mathcal{S} = (\Delta V_l/h^2)\mathcal{T}^{\mathrm{T}}$ if the same function is used for interpolation and spreading, where $\Delta V_l/h^2 \approx 1$ is the volume ratio between the fluid and the solid cell. As a result, the moving force coefficient matrix $\mathcal{M} = (\Delta V_l/h^2)\mathcal{T}\mathcal{T}^{\mathrm{T}}$ is symmetric. It is also found that $\mathcal{M}$ is positive-definite irrespective of the time step and the approximation order as in the IBPM [17]. Moreover, the moving force equation is well conditioned, which converges quickly by using the conjugate gradient method.

Now we incorporate this moving force equation into the rotational incremental pressure-correction projection method. For the sake of simplicity, we rewrite the governing equations (25) as

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \mathcal{H} + \mathcal{P} + \mathcal{F}, \tag{61}$$

$$\mathcal{D}\mathbf{u}^{n+1} = 0, \tag{62}$$

$$\mathcal{T}\mathbf{u}^{n+1} = \mathbf{U}_b^{n+1}, \tag{63}$$

where $\mathcal{H}$, $\mathcal{P}$ and $\mathcal{F}$ are the operators defined as

$$\mathcal{H} := -\left[\frac{3}{2}\mathcal{N}(\mathbf{u}^n) - \frac{1}{2}\mathcal{N}(\mathbf{u}^{n-1})\right] + \frac{1}{2Re}\mathcal{L}(\mathbf{u}^{n+1} + \mathbf{u}^n) - \mathcal{G}p^n, \tag{64}$$

$$\mathcal{P} := -\mathcal{G}\phi^{n+1}, \tag{65}$$

$$\mathcal{F} := \mathcal{S}\mathbf{F}^{n+1}. \tag{66}$$

To decouple the momentum equation (61) from the divergence free condition (62) and the no-slip wall condition on the interface (63), we perform the following operator splitting algorithm:
(1) Prediction step by ignoring the immersed objects

$$\frac{\hat{\mathbf{u}} - \mathbf{u}^n}{\Delta t} = \mathcal{H}(\hat{\mathbf{u}}). \tag{67}$$

(2) Immersed boundary forcing step for satisfying the no-slip wall condition on the interface

$$\frac{\tilde{\mathbf{u}} - \hat{\mathbf{u}}}{\Delta t} = \mathcal{F}, \tag{68}$$

$$\mathcal{T}\tilde{\mathbf{u}} = \mathbf{U}_b^{n+1}. \tag{69}$$

Applying (69) to (68) gives the moving force equation that we have defined previously

$$\mathcal{M}\mathbf{F}^{n+1} = \frac{\mathbf{U}_b^{n+1} - \mathcal{T}\hat{\mathbf{u}}}{\Delta t}. \tag{70}$$

Once the boundary force is determined, we correct the fluid velocity with

$$\tilde{\mathbf{u}} = \hat{\mathbf{u}} + \Delta t \mathcal{S}\mathbf{F}^{n+1}. \tag{71}$$

(3) Projection step for obtaining the divergence free velocity $\mathbf{u}^{n+1}$ and the final pressure p^{n+1}

$$\frac{\mathbf{u}^{n+1} - \tilde{\mathbf{u}}}{\Delta t} = \mathcal{P}, \tag{72}$$

$$\mathcal{D}\mathbf{u}^{n+1} = 0. \tag{73}$$

Applying the divergence operator to (72) and using the divergence free condition (73) gives

$$\mathcal{L}\phi^{n+1} = \frac{1}{\Delta t}\mathcal{D}\tilde{\mathbf{u}}, \tag{74}$$

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}} - \Delta t \mathcal{G}\phi^{n+1}. \tag{75}$$

The final pressure is advanced by

$$p^{n+1} = p^n + \phi^{n+1} - \frac{1}{2Re}\mathcal{D}\hat{\mathbf{u}}. \tag{76}$$

Figure 6 shows the global structure of MIBM. The overall scheme follows the regular fractional step method so that the velocity, the pressure and the force are decoupled. Even though the interface velocity condition is enforced before the projection step, we have found that the velocity on the immersed boundary is essentially unchanged after the projection step. The same observation has also been made by Kempe and Fröhlich [14] and Fadlun *et al.* [8]. It is worth noting that the present MIBM recovers to the explicit method of Kempe and Fröhlich [14] with one iteration in the forcing loop, if $\mathcal{M}$ is set to the identity matrix. However it is not the case, hence our method is implicit.

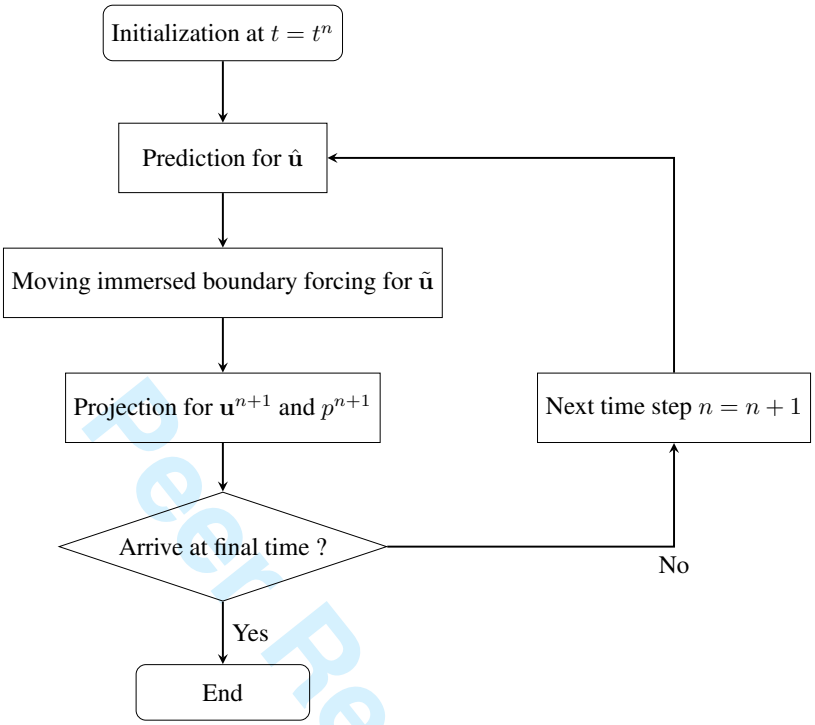


Figure 6. Global structure of the moving immersed boundary method.

4.5. Comparison of performance

To demonstrate the accuracy and efficiency of present moving immersed boundary method, we perform the following test

$$\begin{aligned} &\text{Given} \quad u_0(x,y) = e^x cos\, y - 2, \qquad 0 \leqslant x, y \leqslant 1, \\ &\text{Find } F \text{ such that } u(x,y) = u_0(x,y) + \Delta t \mathcal{S} F = U_b \quad \text{on } \Gamma_s, \end{aligned}$$

where Γ_s is described with a circle of a radius of 0.2 at $(0.52, 0.54)$ and $U_b = 0$. The domain is covered by 64×64 nodes with around 81 Lagrangian points on the circle surface. Δt is set to 1.

In this test, the fluid equations are not solved and only the immersed boundary forcing part is considered. The initial field $u_0(x,y)$ can be seen as a predicted fluid velocity component in one direction. This test is to examine different forcing strategies for imposing the desired velocity U_b at the interface Γ_s via a boundary force F. To facilitate the accuracy study, we define the velocity error norms of L_2 and L_∞ as follows

$$||e_u||_2 = \left[\frac{1}{n_x n_y} \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} (u_{i,j} - u_{i,j}^{\text{ref}})^2 \right]^{1/2}, \tag{77}$$

$$||e_u||_\infty = \max |u_{i,j} - u_{i,j}^{\text{ref}}|, \tag{78}$$

for $i = 1, \ldots, n_x, j = 1, \ldots, n_y$ where u^{ref} represents the reference value. It is worth noticing that the L_2-norm is a good measure of the global error while the L_∞-norm provides a good indicator for the local error.

Figure 7a displays the result of the explicit direct forcing IBM of Uhlmann [10], where u is far away from zero over the immersed boundary compared to Figure 7c. The accuracy is improved after 3 iterations with the method of Kempe and Fröhlich [14], as shown in Figure 7b. Figure 7d reveals that the results are nearly the same for present MIBM with the iterative multidirect forcing IBM of Luo *et al.* [15] and Breugem [16].

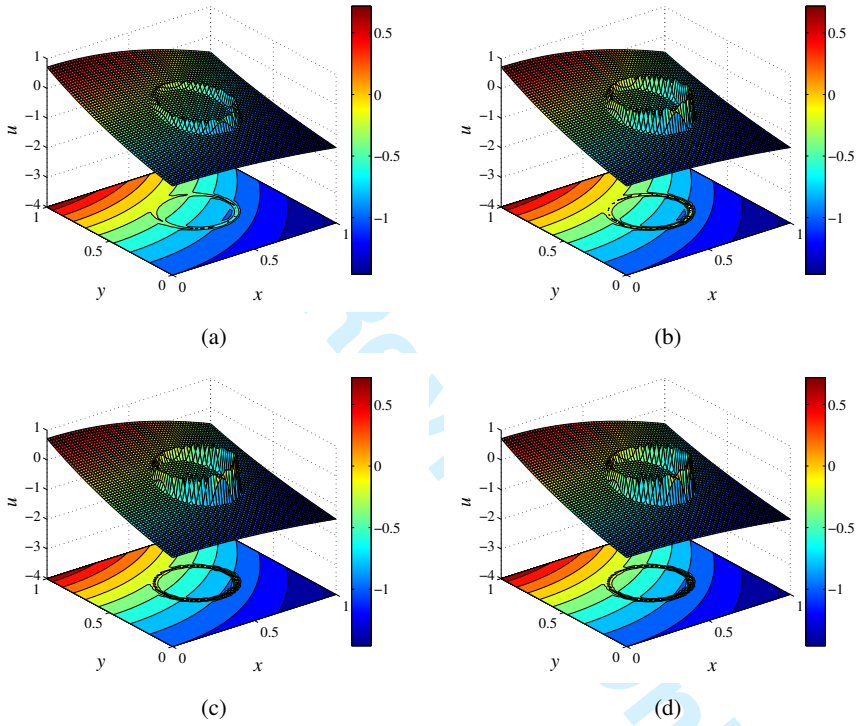


Figure 7. Contour of the scalar field after the boundary forcing: (a) The explicit direct forcing IBM of Uhlmann [10]; (b) The improved explicit direct forcing IBM of Kempe and Fröhlich [14]; (c) The multidirect forcing IBM of Luo *et al.* [15] and Breugem [16]; (d) Present MIBM.

Table III compares the computational time and velocity error on the interface of these immersed boundary methods. The error is measured in L_2-norm and the tolerance is 1×10^{-15}. The method of Uhlmann [10] is the quickest due to its explicit nature, but it suffers a large error of 3.01×10^{-1} on the immersed interface. The forcing loop of Kempe and Fröhlich [14] reduces the error by a factor of 4 with 3 iterations. However, the error of 7.41×10^{-2} is still considered large.

The iterative multiforcing IBM of Luo *et al.* [15] and Breugem [16] is required to converge towards the machine precision, but it takes approximately 606 times more additional computational effort than the explicit method of Uhlmann [10]. Actually, the convergence rate in the multiforcing IBM decreases dramatically after about 10 iterations, as shown in Figure 8. In order to reduce the error to 1×10^{-6} around 1000 iterations are needed and 4443 iterations for the machine precision.

The present MIBM converges to the same machine precision only with 60 iterations by using the conjugate gradient solver. The iteration can be further reduced if preconditioning is taken, but we find that the conjugate gradient solver is sufficient for fast convergence. The computation is not increased considerably compared to the explicit method of Uhlmann [10], as we can see that the present method only takes twice the amount of computational time of the direct forcing IBM of Uhlmann [10]. It also worth noticing that present MIBM is almost as efficient as the method of Kempe and Fröhlich [14].

	Process time (s)					
	Interpolation	Forcing	Distribution	Total	Iter.	Error
Uhlmann [10]	2.77×10^{-3}	1.00×10^{-6}	3.23×10^{-3}	6.02×10^{-3}	1	3.01×10^{-1}
Kempe and Fröhlich [14]	8.15×10^{-3}	1.00×10^{-6}	8.92×10^{-3}	1.71×10^{-2}	3	7.41×10^{-2}
Luo *et al.* [15] and Breugem [16]	1.16×10^{1}	1.17×10^{-3}	1.31×10^{1}	3.65×10^{1}	4443	9.96×10^{-16}
Present	4.32×10^{-4}	1.19×10^{-4}	4.41×10^{-4}	1.33×10^{-2}	60	8.29×10^{-16}

Table III. Comparison of the computational time and the velocity error. The iteration number is fixed for the explicit methods of Uhlmann [10] and Kempe and Fröhlich [14], while others are solved until convergence under a tolerance of 1×10^{-15}.

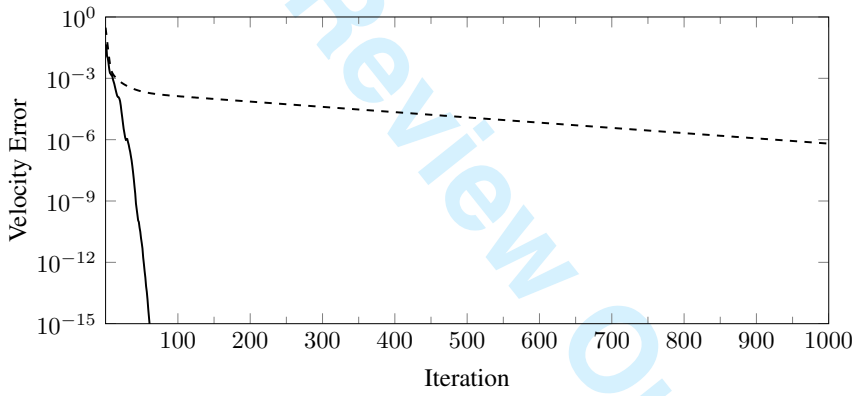


Figure 8. Comparison of convergence between present MIBM (——) and the multidirect forcing IBM of Luo *et al.* [15] and Breugem [16] (- - - -).

5. RESULTS

5.1. *Taylor-Green vortices*

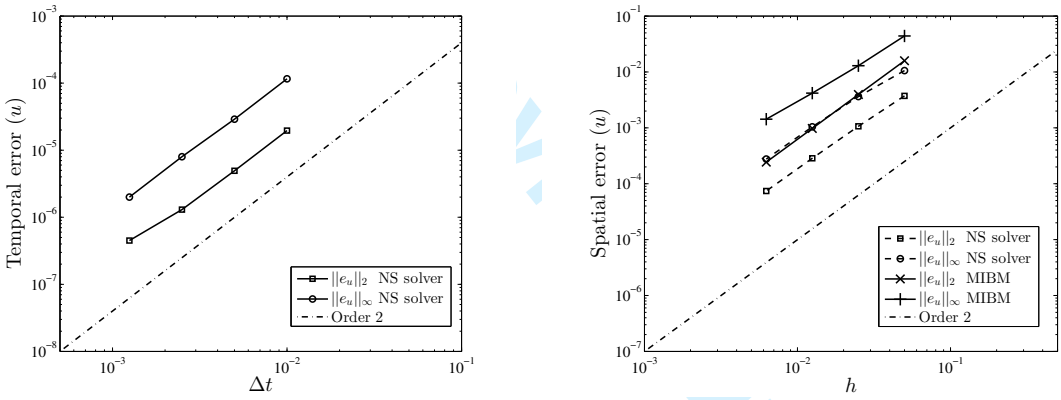
We first consider the two-dimensional unsteady case of an array of decaying vortices to assess the accuracy of the fluid solver. The analytical solution of the Taylor-Green vortices is given by

$$\begin{aligned} u(x,y,t) &= -cos(\pi x)sin(\pi y)e^{-2\pi^2 t/Re}, \\ v(x,y,t) &= sin(\pi x)cos(\pi y)e^{-2\pi^2 t/Re}, \\ p(x,y,t) &= -\frac{1}{4}(cos(2\pi x) + sin(2\pi y))e^{-4\pi^2 t/Re}. \end{aligned} \tag{79}$$

This simulation is performed on a square domain $\Omega = [-1.5, 1.5] \times [-1.5, 1.5]$ and the Reynolds number Re is prescribed to 10. The initial and boundary conditions are provided by the exact solution. We advance the equations for $0 \leqslant t \leqslant 0.2$.

To study the temporal accuracy, we compare the results at $t = 0.2$ to a reference solution obtained by a very fine time step $\Delta t = 1 \times 10^{-4}$ with the spatial resolution of $\Delta x = \Delta y = 9.375 \times 10^{-3}$. The errors on the velocity component u are computed by subtracting the reference solution from other numerical solutions ($\Delta t \in [0.00125, 0.01]$), to cancel out the error due to spatial discretization. The L_2, L_∞ error norms are then displayed in Figure 9a on a log-log plot. A second order temporal accuracy is observed, which confirms previous error estimation analysis for the rotational incremental pressure-correction projection method.

We also expect a second order spatial accuracy since the second order central differencing scheme is used for all the derivatives in this case. We use a small time step $\Delta t = 1 \times 10^{-4}$ to ensure that the temporal discretization error is negligible compared to the spatial one, and then vary the computational grids ($n_x \times n_y = 20 \times 20$, 40×40, 80×80, and 160×160). The error is obtained by comparing the results to the analytical solution. Figure 9b shows the spatial discretization error, indicating a second order spatial accuracy.



(a) (b)

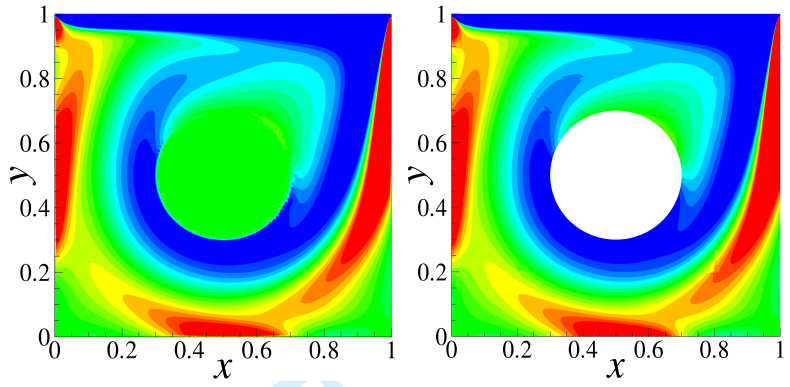
Figure 9. Temporal (a) and spatial (b) convergence analysis of current fluid solver and moving immersed boundary method for the decaying vortices problem.

It is well known that the discrete delta function undermines the space accuracy of the original fluid solver. Now we embed a circular cylinder of a unit radius in the center of the computational domain to study the accuracy of our MIBM. The time dependant no-slip boundary condition at the immersed cylinder surfaces is enforced by current MIBM. Figure 9b shows the variation of the velocity error as a function of the mesh size. It is evident that current MIBM introduces errors the original fluid solver but it still retains the second order accuracy, which corresponds to the interpolation properties of the discrete delta function for smooth fields.

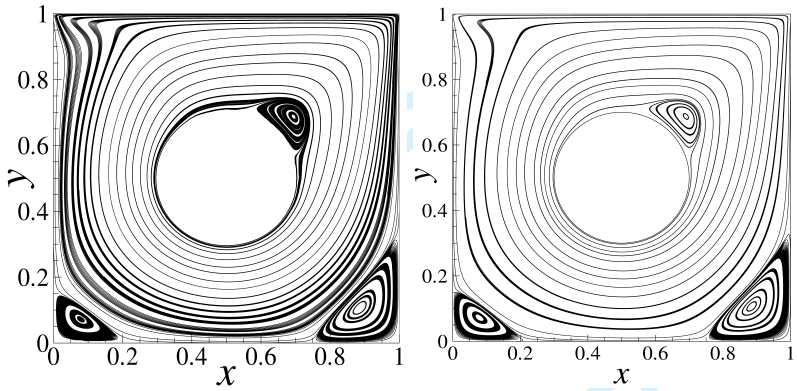
5.2. Lid-driven cavity flow with an embedded cylinder

In this test, we compare current immersed boundary method with the traditional body-conforming mesh method. The domain configuration and the boundary conditions are taken the same as in the

classical lid-driven cavity flow case, namely the top wall is moving with a constant velocity $u_\infty = 1$ while the others are stationary walls, except that we place a cylinder in the domain center. In order to compare with Vanella and Balaras [38], the diameter of the cylinder is set to $D = 0.4L$ with L being the cavity length. The Reynolds number is 1000 in this study based on the cavity length. A uniform mesh of 200×200 is employed in the immersed boundary method, and the same mesh size is used for the body-conforming mesh method for comparison.



(a) Vorticity



(b) Streamlines

Figure 10. Vorticity contours and streamlines of the lid-driven cavity flow with a cylinder at $Re = 1000$, where the vorticity contour value is varied from -3 (blue) to 3 (red) with an increment of 0.4. Results of present MIBM are listed on the left; Results of the body-conforming mesh method are on the right.

	(x_1, y_1)	(x_2, y_2)	(x_3, y_3)
Present	$(0.6942, 0.6881)$	$(0.0789, 0.0720)$	$(0.8852, 0.1063)$
Body-conforming mesh method	$(0.6906, 0.6872)$	$(0.0791, 0.0721)$	$(0.8849, 0.1063)$

Table IV. Comparison of vortices center positions for the proposed immersed boundary method and the body-conforming mesh method, where (x_1, y_1), (x_2, y_2), (x_3, y_3) are the vortices centers at the upper right to the cylinder, at the lower left corner and at the lower right corner respectively.

The flow reaches a final steady state as the time advances. Figure 10 shows the vorticity contours and streamlines for the flow at $Re = 1000$, which are similar to the results of [38]. As we can see,

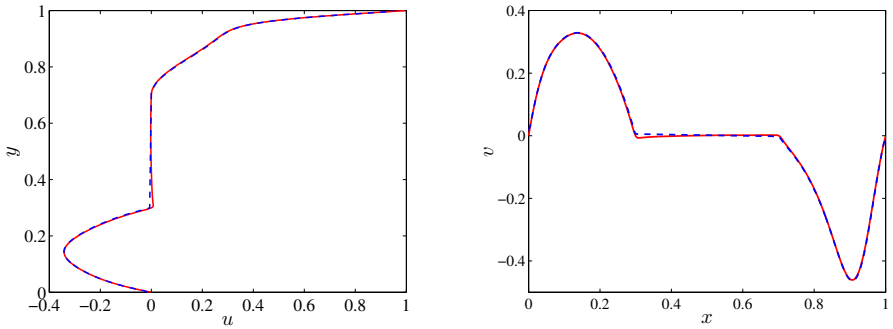


Figure 11. Comparison of velocity profiles of the lid-driven cavity flow with a cylinder at $Re = 1000$: (a) Distribution of velocity component u along $x = 0.5$; (b) Distribution of velocity component v along $y = 0.5$. Solid lines represent current method and dashed lines are the traditional body-conforming mesh method.

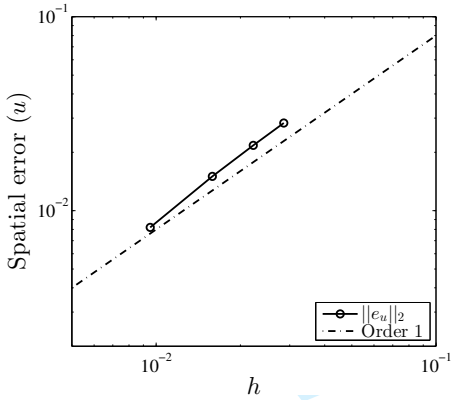


Figure 12. L_2 error norm of the horizontal velocity component (u) as a function of grid spacing for the lid-driven cavity flow with an embedded cylinder.

three vortices emerge in the flow. One at the upper right position of the cylinder and two near the bottom at each corners. It is noteworthy that the upper vortex is generated by the presence of the cylinder. The flow fields outside the cylinder are essentially the same for current MIBM and the body-conforming mesh method. The only difference is that there is a flow inside the cylinder in the immersed boundary method, which however is the key idea of the immersed boundary method to replace the solid domain with fluid. The velocity component u at the vertical midline $x = 0.5$ and the velocity component v at the horizontal midline $y = 0.5$ are plotted in Figure 11. The velocity profiles of both methods match pretty well. The location of the three vortices centers are also listed in Table IV. Very close results have been obtained.

Next we study the grid convergence for assessing the accuracy of present method for non-smoothed field. A series of computations are performed on a hierarchy of grids (70×70, 90×90, 126×126, 210×210 and 630×630). The variation of error of the velocity component u along with the grid spacing is displayed in Figure 12, showing a convergence rate of about 1.13. This is because the flow becomes not smooth near the immersed surface in this case, and the discrete

delta function used in present work can no longer maintain the second order accuracy. Beyer and LeVeque [5] analysed various discrete delta functions and pointed out that the second order accuracy can be recovered through using different functions for interpolation and spreading. This results in non-symmetric coefficient matrix of the boundary force in MIBM, which can be solved with the GMRES or Bi-CGSTAB methods.

5.3. Flow over a stationary circular cylinder

The flow past a stationary circular cylinder is considered as a canonical test case to validate current method, since a great amount of experimental and numerical studies at different Reynolds numbers are available for comparison. The flow characteristics depend on the Reynolds number $Re = u_\infty D/\nu$, based on the inflow velocity u_∞, the cylinder diameter $D = 1$ and the fluid kinematic viscosity ν. The simulation is performed in a rectangular domain, where the fluid flows from the left to the right (see Figure 13). At left boundary, a uniform velocity of $u_\infty = 1$ is imposed; The free slip boundary conditions are applied at lateral boundaries; At outlet, the convective boundary condition $\partial \mathbf{u}/\partial t + u_\infty \partial \mathbf{u}/\partial x = 0$ is employed for reducing the reflection effects because of the finite artificially truncated domain. The cylinder is placed at the center of the computational domain. The fluid domain is covered with a uniform mesh, and the cylinder surface is represented by a set of uniformly distributed Lagrangian points with $\delta s \approx h$.

For comparison the drag and lift coefficients are defined as

$$C_D = \frac{F_D}{\frac{1}{2}\rho u_\infty^2 D}, \quad C_L = \frac{F_L}{\frac{1}{2}\rho u_\infty^2 D}, \tag{80}$$

where F_D, F_L are the drag and lift forces on the cylinder exerted by the fluid, respectively. The fluid density ρ is set to 1 here. As a matter of fact, the spreading and interpolation operators constructed from the regularized delta function conserve the total force, hence F_D and F_L can be computed directly by summing up the forces over all the Lagrangian points

$$\begin{pmatrix} F_D \\ F_L \end{pmatrix} = -\sum_{l=1}^{n_b} \mathbf{F}(\mathbf{X}_l)\Delta V_l. \tag{81}$$

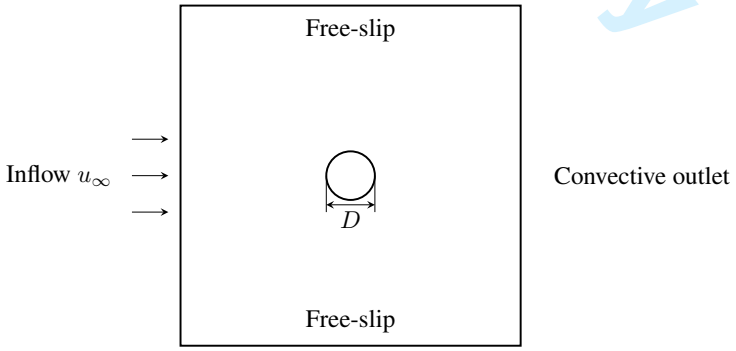


Figure 13. Sketch of the flow over a stationary circular cylinder.

5.3.1. $Re = 30, 40$

The flow presents a steady state at low Reynolds numbers $Re = 30, 40$ with a recirculating region in the wake of the cylinder. The wake dimensions are described by the length of the wake l, the stream-wise distance a from the vortex center to the nearest point at the cylinder surface, the cross-wise distance b between two vortices centers, and the angle θ of flow separation, as shown in Figure 14. The computations are performed under different mesh resolutions to check the grid convergence. Various domain sizes are also considered to ensure that the boundary confinement effect does not influence the solution. We select the time step such that the CFL condition is satisfied, and the current method yields a stable solution even with a Courant number close to one.

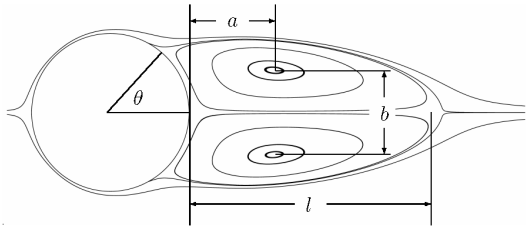


Figure 14. The definition of the characteristic wake dimensions for the steady flow over a stationary circular cylinder.

Table V. Comparison of the steady-state wake dimensions and the drag coefficient for the flow over a stationary cylinder at $Re = 30, 40$. The experimental results are marked with ($\star$).

		l/D	a/D	b/D	θ°	C_D
$Re = 30$	Coutanceau and Bouard [44]*	1.55	0.54	0.54	50.0	-
	Tritton [45]*	-	-	-	-	1.74
	Pinelli *et al.* [39]	1.70	0.56	0.52	48.1	1.80
	Toja-Silva *et al.* [41]	1.71	0.56	0.53	47.9	1.78
	Present ($\Omega = 30D \times 30D$, $h = 0.04D$)	1.66	0.58	0.52	45.0	1.78
	Present ($\Omega = 30D \times 30D$, $h = 0.029D$)	1.64	0.58	0.53	49.9	1.78
	Present ($\Omega = 30D \times 30D$, $h = 0.02D$)	1.64	0.58	0.52	46.5	1.78
	Present ($\Omega = 40D \times 40D$, $h = 0.029D$)	1.65	0.57	0.53	47.4	1.75
	Present ($\Omega = 60D \times 60D$, $h = 0.029D$)	1.64	0.57	0.53	49.8	1.73
$Re = 40$	Coutanceau and Bouard [44]*	2.13	0.76	0.59	53.8	-
	Tritton [45]*	-	-	-	-	1.59
	Wang and Zhang [46]	2.36	0.72	0.6	53.8	1.54
	Taira and Colonius [17]	2.30	0.73	0.60	53.7	1.54
	Present ($\Omega = 30D \times 30D$, $h = 0.04D$)	2.38	0.77	0.59	52.0	1.58
	Present ($\Omega = 30D \times 30D$, $h = 0.029D$)	2.34	0.76	0.62	54.5	1.58
	Present ($\Omega = 30D \times 30D$, $h = 0.02D$)	2.36	0.77	0.60	53.1	1.59
	Present ($\Omega = 40D \times 40D$, $h = 0.029D$)	2.36	0.75	0.62	52.1	1.56
	Present ($\Omega = 60D \times 60D$, $h = 0.029D$)	2.34	0.76	0.62	54.5	1.54

The streamlines, vorticity and pressure contours are shown in Figure 15, which are in close agreement with those reported in the literature. Table V compares the wakes dimensions and the drag coefficient against other numerical and experimental results. Good agreements have been obtained.

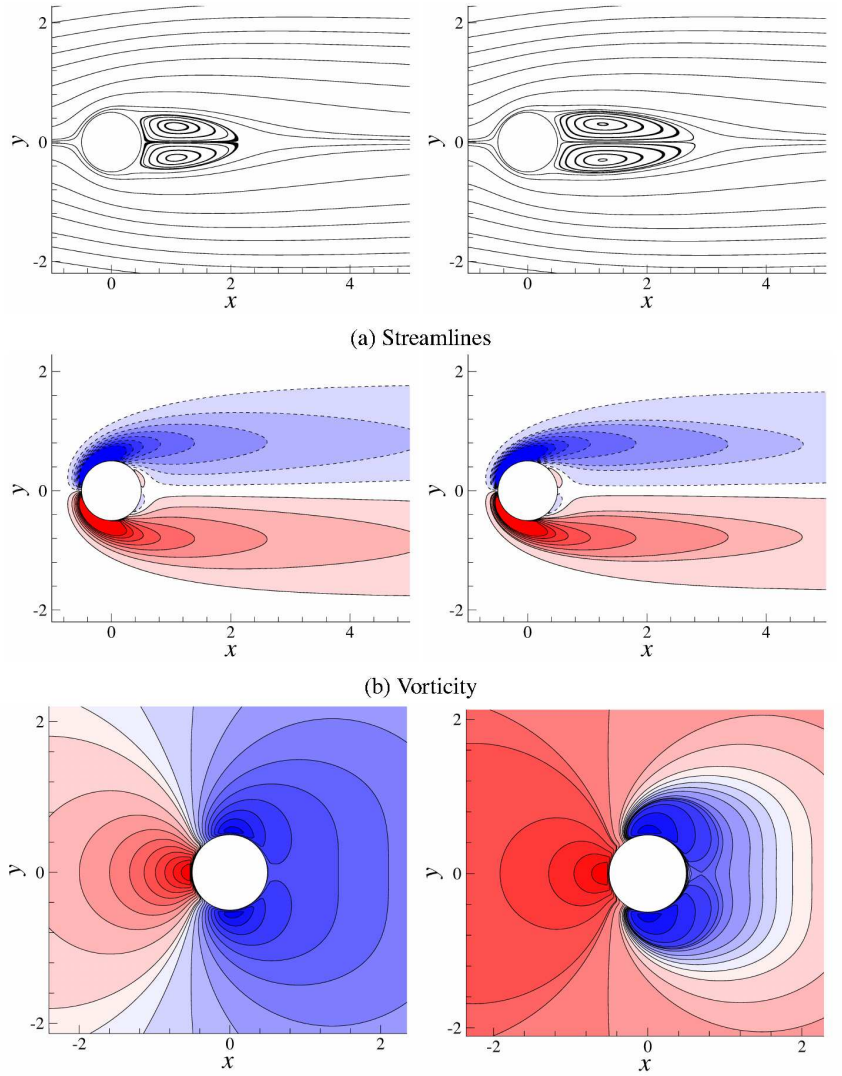
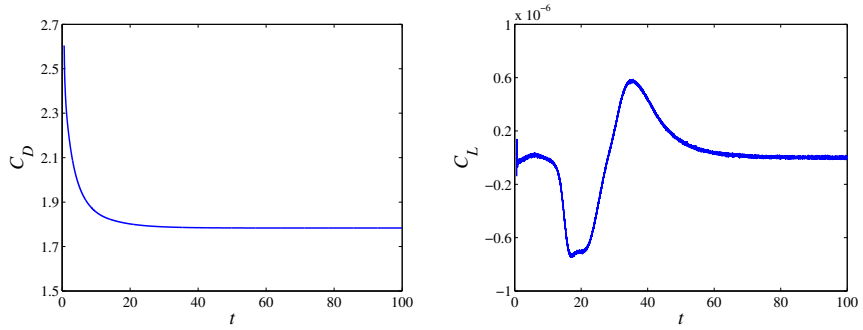


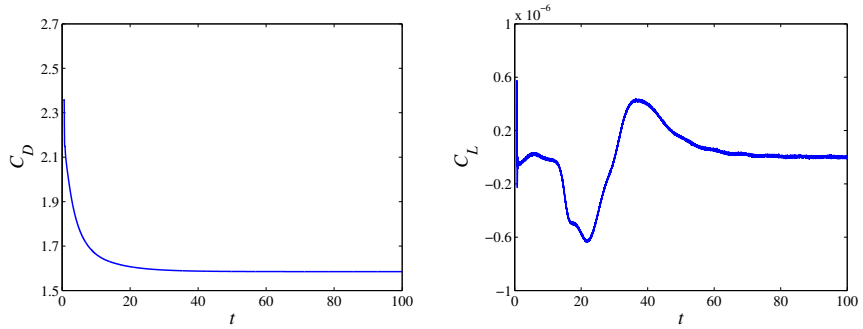
Figure 15. Streamlines, vorticity and pressure contours for the steady-state flow around a circular cylinder at $Re = 30$ (left) and $Re = 40$ (right).

It can be concluded from Table V that narrow domain size leads to a larger value of the drag coefficient, which was also observed in [10, 47, 17]. For example at $Re = 30$ the drag coefficient for the domain $\Omega = 30D \times 30D$ is 3% higher than the value with the largest domain. By enlarging the domain size to $\Omega = 40D \times 40D$, the drag coefficient is reduced by 2%. This confinement effect is due to the finite distance of the lateral boundaries treated as slip walls. The time history of the drag and lift coefficients is shown in Figure 16.

The vorticity ω_z and pressure coefficient C_P along the cylinder surface at $Re = 40$ are displayed in Figure 17, where $\theta = 0°$ and $\theta = 180°$ correspond to the stagnation point and the base point, respectively. The wall pressure coefficient is defined as $C_P = (p - p_\infty)/(\frac{1}{2}\rho u_\infty^2)$, where p_∞ is the

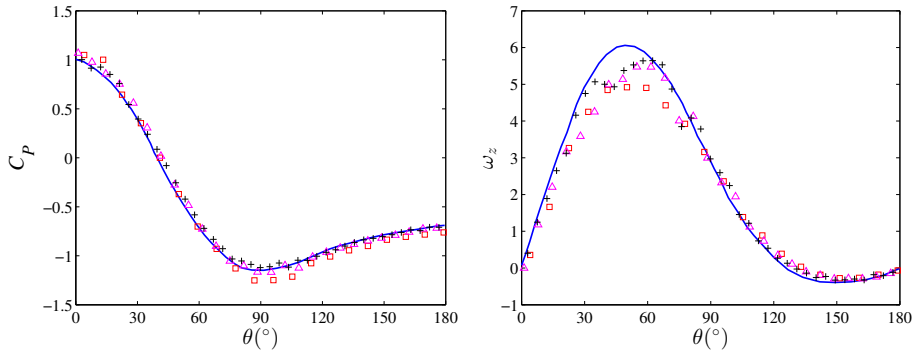


(a) $Re = 30$



(b) $Re = 40$

Figure 16. Drag and lift coefficients versus time for flow over a stationary cylinder.



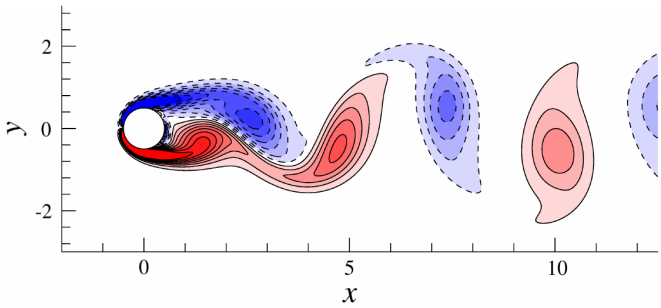
(a) (b)

Figure 17. The wall pressure coefficient C_p and the wall vorticity W_z for flow over a stationary cylinder at $Re = 40$. ——, results of boundary-fitted grid of Braza *et al.* [29]; □, present $h = 0.04D$; △, present $h = 0.029D$; +, present $h = 0.02D$.

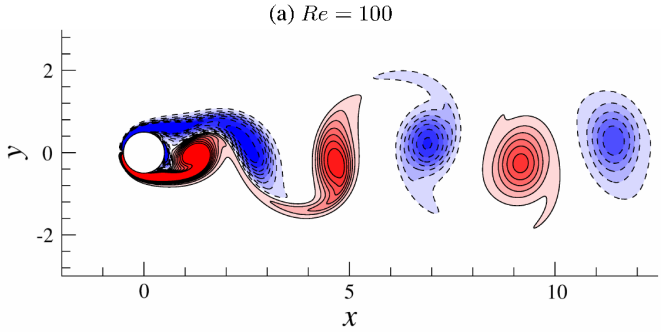
free-stream pressure. Numerical results obtained with body-fitted grid of Braza *et al.* [29] are also included for comparison. The results with present MIBM are very close to those with body-fitted grid. The wall vorticity and pressure coefficient with current MIBM converge to the body-fitted results as the mesh resolution is increased.

5.3.2. $Re = 100, 200$

Increasing the Reynolds number to $Re = 100$ and 200, the flow becomes unsteady and periodic shedding of vortices is found. The well-known von Kármán vortex street is shown in Figure 18. Figure 19 shows the corresponding pressure field. The time evolution of the drag, lift coefficients at $Re = 100$ and $Re = 200$ are plotted in Figure 20. It should be pointed out that the oscillating frequency of the drag is twice that of the lift, which is in fact the vortex shedding frequency f_s [47]. The Strouhal number $St = Df_s/u_\infty$ as well as the coefficients of drag and lift are summarized in Table VI. For comparison, we list the well established experimental results of Williamson [48] and the numerical results with the body-fitted mesh methods of Braza *et al.* [29] and Liu *et al.* [49]. Results with other IBM variants are also included, e.g. the explicit direct forcing IBM of Uhlmann [10], the vortex penalization method of Mimeau *et al.* [50], the immersed interface method of Xu and Wang [51], the iterative direct forcing IBM of Ji *et al.* [18] and the IBPM of Taira and Colonius [17]. Good agreement has been obtained towards the flow quantities.



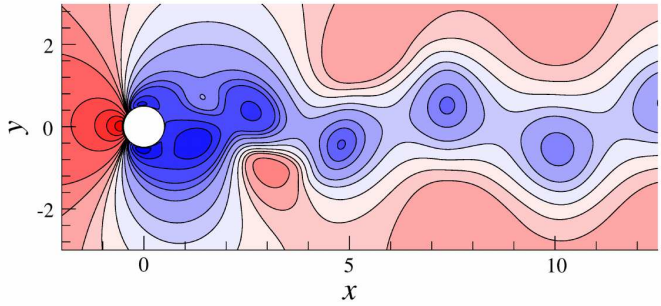
(a) $Re = 100$



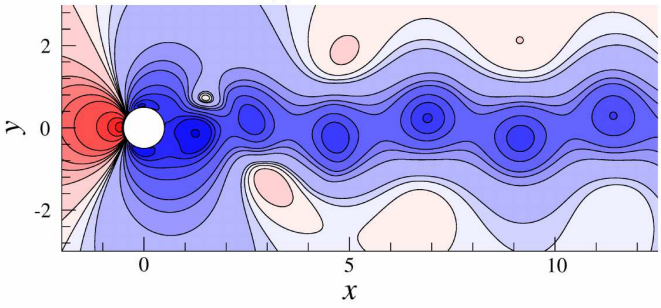
(b) $Re = 200$

Figure 18. Instantaneous vorticity contours of flow over a circular cylinder at (a) $Re = 100$ and (b) $Re = 200$, where the contour level is set from -3 to 3 with an increment of 0.4.

From Table VI we can see that the current method yields an over-prediction of the mean drag coefficient only with 2% error, while Uhlmann [10] over-predicted the mean drag coefficient value by approximately 11%, for the computational domain $\Omega = 30D \times 30D$ with the mesh resolution $h = 0.029D$. The reference value is taken from Liu *et al.* [49]. This improved accuracy can be attributed to the exact imposition of the no-slip boundary condition at the interface in current

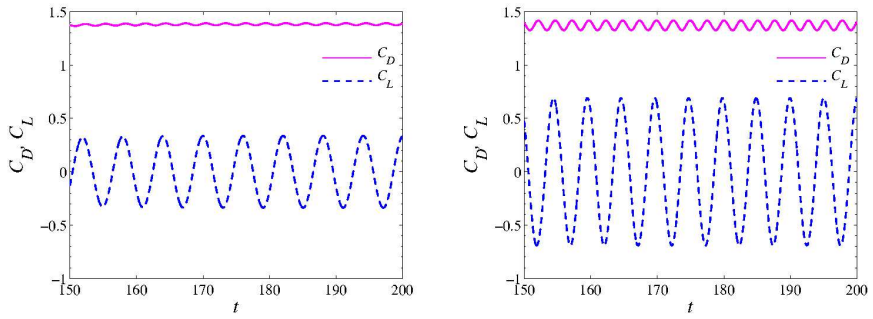


(a) $Re = 100$



(b) $Re = 200$

Figure 19. Instantaneous pressure contours of flow over a circular cylinder at (a) $Re = 100$ and (b) $Re = 200$.



(a) (b)

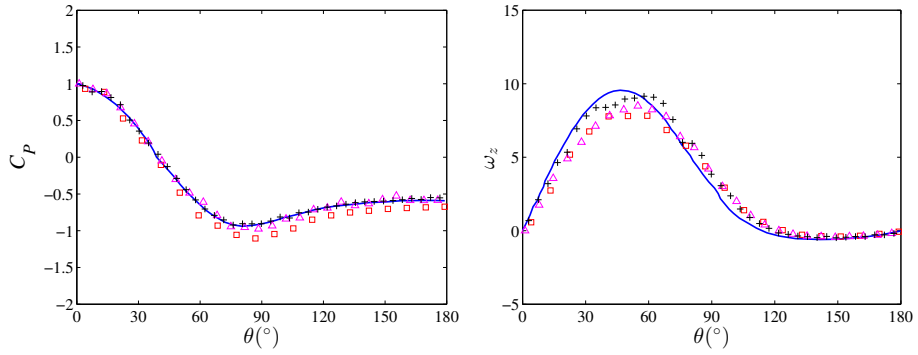
Figure 20. Time evolution of drag and lift coefficients at (a) $Re = 100$ and (b) $Re = 200$.

method. This error is further reduced to 1% when we use an enlarged domain of $\Omega = 40D \times 40D$ while a relative large error of 8% is still found in [10].

Compared to the vortex penalization method of Mimeau *et al.* [50] and the immersed interface method of Xu and Wang [51], our implementation does not introduce artificial constants and thus is much suitable for flow with rigid bodies. Our results are very close to those of the iterative direct forcing IBM of Ji *et al.* [18] and the IBPM of Taira and Colonius [17]. However, our method is non-iterative compared to Ji *et al.* [18]. The original system is unchanged and only a small system is solved additionally at each time step. Therefore the current method can be much more efficient than the IBPM of Taira and Colonius [17].

Table VI. Comparison of the drag, lift coefficients and the Strouhal number for the flow around a stationary cylinder at $Re = 100, 200$. The experimental results are marked with ($\star$).

		$\overline{C}_D$	C'_D	C'_L	St
$Re = 100$	Williamson [48]*	-	-	-	0.164
	Uhlmann [10]	1.453	$\pm$0.011	$\pm$0.339	0.169
	Ji *et al.* [18]	1.376	$\pm$0.010	$\pm$0.339	0.169
	Braza *et al.* [29]	1.359	$\pm$0.019	$\pm$0.293	0.16
	Liu *et al.* [49]	1.350	$\pm$0.012	$\pm$0.339	0.165
	Mimeau *et al.* [50]	1.40	$\pm$0.010	$\pm$0.32	0.165
	Xu and Wang [51]	1.423	$\pm$0.013	$\pm$0.34	0.171
	Present ($\Omega = 30D \times 30D$, $h = 0.04D$)	1.380	$\pm$0.010	$\pm$0.343	0.160
	Present ($\Omega = 30D \times 30D$, $h = 0.029D$)	1.377	$\pm$0.010	$\pm$0.337	0.160
	Present ($\Omega = 30D \times 30D$, $h = 0.02D$)	1.379	$\pm$0.010	$\pm$0.346	0.160
	Present ($\Omega = 40D \times 40D$, $h = 0.029D$)	1.366	$\pm$0.010	$\pm$0.342	0.160
	Present ($\Omega = 60D \times 60D$, $h = 0.029D$)	1.353	$\pm$0.010	$\pm$0.335	0.160
$Re = 200$	Williamson [48]*	-	-	-	0.197
	Taira and Colonius [17]	1.35	$\pm$0.048	$\pm$0.68	0.196
	Ji *et al.* [18]	1.354	$\pm$0.044	$\pm$0.682	0.20
	Braza *et al.* [29]	1.386	$\pm$0.040	$\pm$0.766	0.20
	Liu *et al.* [49]	1.31	$\pm$0.049	$\pm$0.69	0.192
	Mimeau *et al.* [50]	1.44	$\pm$0.05	$\pm$0.75	0.200
	Xu and Wang [51]	1.42	$\pm$0.04	$\pm$0.66	0.202
	Present ($\Omega = 30D \times 30D$, $h = 0.04D$)	1.355	$\pm$0.042	$\pm$0.677	0.200
	Present ($\Omega = 30D \times 30D$, $h = 0.029D$)	1.365	$\pm$0.044	$\pm$0.696	0.200
	Present ($\Omega = 30D \times 30D$, $h = 0.02D$)	1.374	$\pm$0.046	$\pm$0.705	0.200
	Present ($\Omega = 40D \times 40D$, $h = 0.029D$)	1.358	$\pm$0.044	$\pm$0.682	0.200
	Present ($\Omega = 60D \times 60D$, $h = 0.029D$)	1.345	$\pm$0.043	$\pm$0.682	0.200



(a) (b)

Figure 21. The wall pressure coefficient C_p and the wall vorticity W_z for flow over a stationary cylinder at $Re = 100$. Time-averaged values are used. ——, results of boundary-fitted grid of Braza *et al.* [29]; □, present $h = 0.04D$; △, present $h = 0.029D$; +, present $h = 0.02D$.

The time-averaged values of the wall vorticity ω_z and the wall pressure coefficient C_P are shown in Figure 21 for $Re = 100$. Good agreements have been found compared to the results of Braza *et al.* [29]. The effects of different discrete delta functions on the results are also tested in Table VII

Table VII. Effects of different discrete delta functions on the drag, lift coefficients and the Strouhal number for the flow around a stationary cylinder at $Re = 100$ and 200.

		$\overline{C}_D$	C'_D	C'_L	St
$Re = 100$	ϕ_2	1.388	± 0.010	± 0.346	0.166
	ϕ_3	1.377	± 0.010	± 0.339	0.166
	ϕ_4	1.379	± 0.011	± 0.343	0.166
$Re = 200$	ϕ_2	1.391	± 0.047	± 0.709	0.198
	ϕ_3	1.365	± 0.044	± 0.696	0.200
	ϕ_4	1.358	± 0.045	± 0.688	0.195

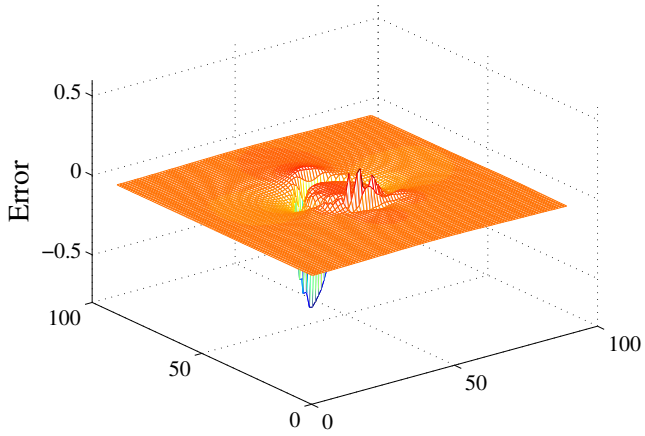


Figure 22. Distribution of the horizontal velocity error on the 90×90 grid for the flow over a stationary circular cylinder.

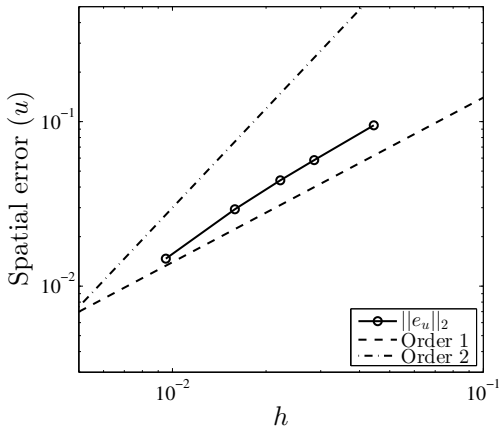


Figure 23. L_2 error norm of horizontal velocity u versus the computational grid size for the flow over a stationary circular cylinder.

for $Re = 100$, 200, where a domain of $\Omega = 30D \times 30D$ is used and the mesh resolution is set to $h = 0.029D$.

A careful grid convergence study is also performed to examine the order of accuracy in this case. Since the exact solution does not exist, we use the solution calculated on a highly resolved grid of 630×630 as our reference for computing the error. The computation domain is taken as $[-2D, 2D] \times [-2D, 2D]$ with the Reynolds number $Re = 100$. The equations are advanced until 0.2 and a relative small time step of 5×10^{-4} is chosen such that the time discretization error will not influence the results. Same computations but on different grids are performed and compared the reference solution, namely 45×45, 70×70, 90×90, 126×126 and 210×210. The distribution of velocity error in the x-direction for the 90×90 grid is shown in Figure 22. Large magnitudes of error in velocity are located near the cylinder. Figure 23 displays the L_2 norm of this error on a log-log plot. A convergence rate of around 1.21 is observed.

5.3.3. $Re = 1000$

We further extend our method to a higher Reynolds number flow $Re = 1000$. At this regime, the convection effects become predominant and the boundary layer thickness decreases, which can be estimated by $\delta \approx D/\sqrt{Re} = 0.032$. To capture the thin boundary layer, a fine grid resolution of $h = 0.01D$ is taken, as recommended in [52, 53]. Note that the grid resolution is only marginal for resolving the boundary layer at this Reynolds number. Nevertheless, the results are satisfactory and the essential features of the flow are well captured. The computational domain is chosen to be $[-20D, 20D] \times [-20D, 20D]$. The two-point-width hat function ϕ_2 is employed in this case as it provides a sharp interface. Figure 25 shows the instantaneous vorticity field. The coefficients of drag and lift are plotted in Figure 24. Note that the flow is inherently three-dimensional at this Reynolds number. We compare our simulations with other two-dimensional results available in the literature. The properties of the drag and lift coefficients are summarized in Table VIII. Good agreements have been found.

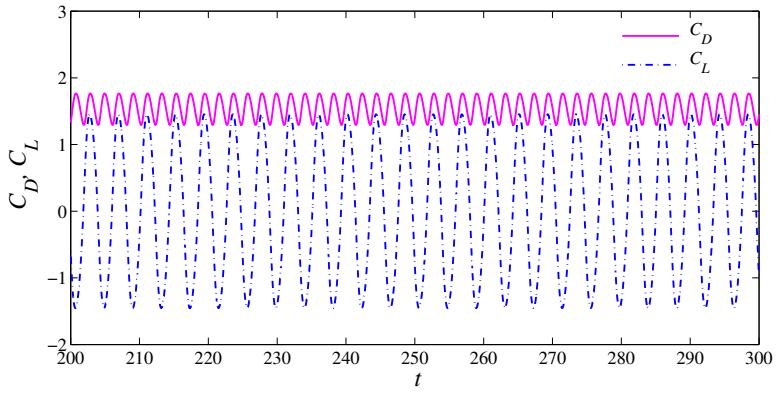


Figure 24. Time evolution of drag and lift coefficients for the flow over a stationary cylinder at $Re = 1000$.

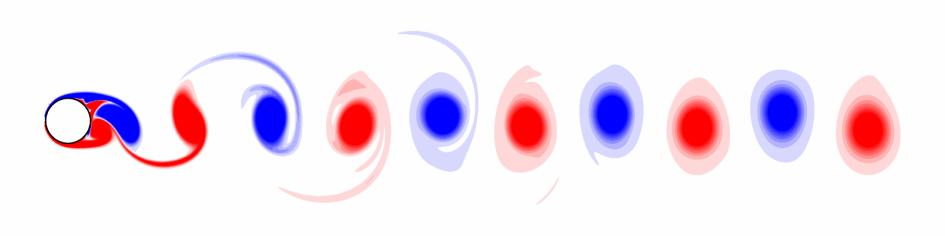


Figure 25. Instantaneous vorticity field for flow over a stationary cylinder at $Re = 1000$.

Table VIII. Comparison of the drag, lift coefficients and the Strouhal number for the flow around a stationary cylinder at $Re = 1000$.

		$\overline{C}_D$	C'_D	C'_L	St
$Re = 1000$	Mittal and Kumar [54]	1.48	± 0.21	± 1.65	0.250
	Apte *et al.* [53]	1.50	-	-	0.238
	Mittal *et al.* [52]	1.48	-	-	-
	Mimeau *et al.* [50]	1.51	± 0.23	± 1.54	0.245
	Present	1.55	± 0.22	± 1.46	0.240

5.4. In-line oscillating circular cylinder in a fluid at rest

We consider the flow induced by an oscillating circular cylinder as another test, in order to demonstrate the ability of our method for handling moving boundaries. The motion of the cylinder is described by a simple harmonic oscillation as follows

$$x(t) = -A\,sin(2\pi ft), \tag{82}$$

where $x(t)$ is the streamwise location of the cylinder center. A and f are the amplitude and the frequency of oscillation, respectively.

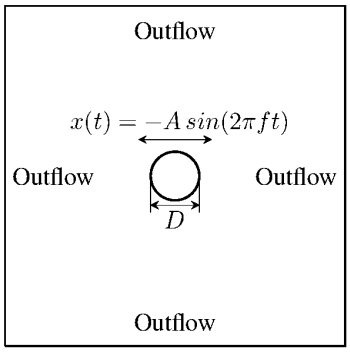
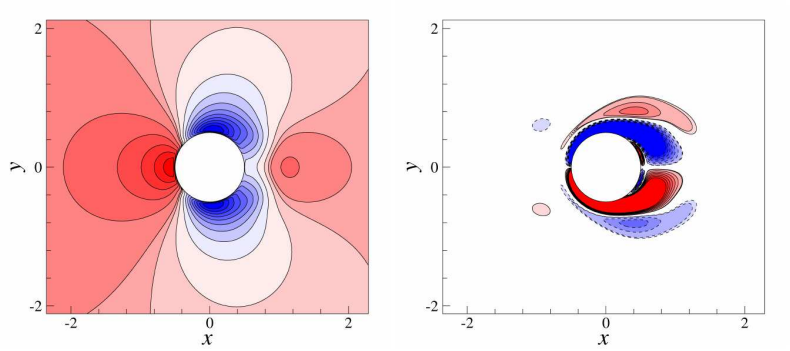
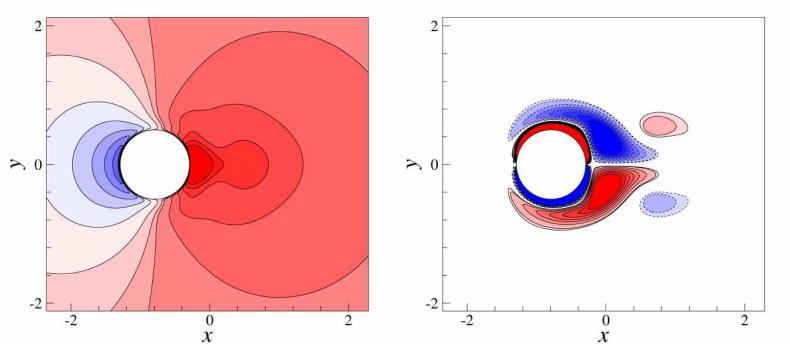


Figure 26. Sketch of the oscillating circular cylinder in a fluid at rest.

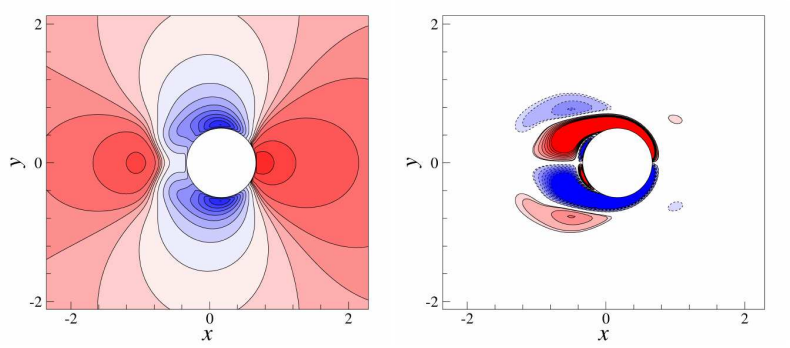
Two key parameters determine the flow characteristics: the Reynolds number $Re = U_{max}D/\nu$ and the Keulegan–Carpenter number $KC = U_{max}/fD$, where U_{max} is the maximum velocity of the oscillating cylinder, ν is the kinematic viscosity and D is the cylinder diameter. Here Re is set to 100 and KC is 5, corresponding to the LDA experiments and the numerical simulations of [55].



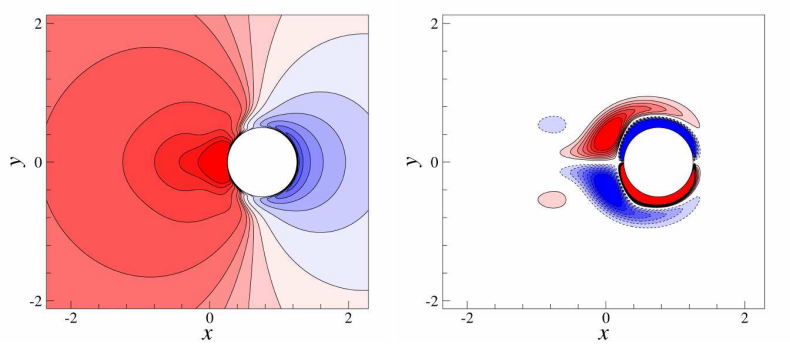
(a) $\phi = 0°$



(b) $\phi = 96°$



(c) $\phi = 192°$



(d) $\phi = 288°$

Figure 27. Pressure and vorticity contours at four different phases.

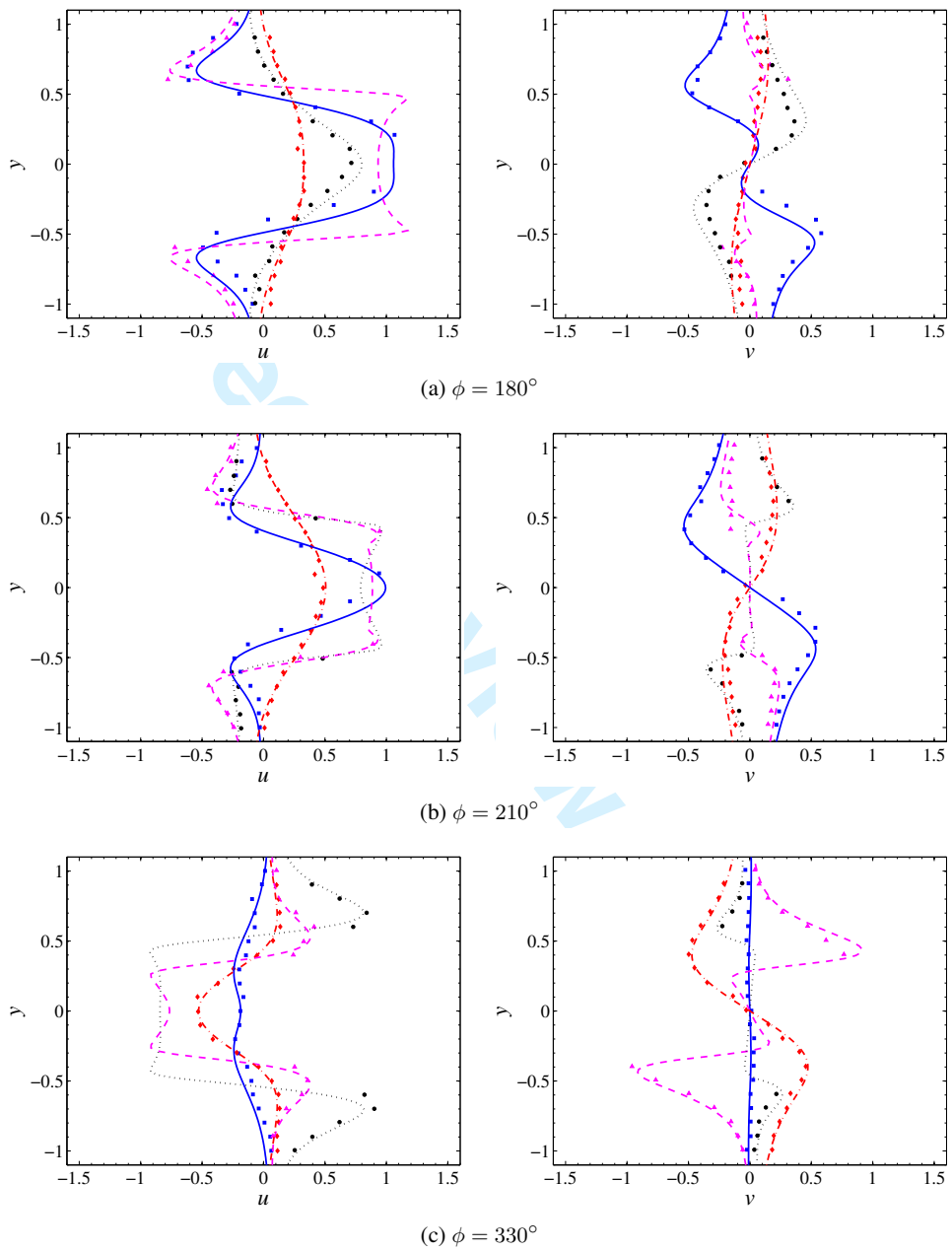


Figure 28. Comparison of the velocity profiles u (left) and v (right) at four different cross-sections and three phase positions: (a) $\phi = 180^\circ$, (b) $\phi = 210^\circ$, (c) $\phi = 330^\circ$. The experimental results of Dütsch *et al.* [55] are marked with ■ at $x = -0.6D$, ▲ at $x = 0D$, ● at $x = 0.6D$, ♦ at $x = 1.2D$. The present results are represented by —— at $x = -0.6D$, - - - - at $x = 0D$, · · · · at $x = 0.6D$, — · — · at $x = 1.2D$.

The computational domain is chosen to be $14D \times 14D$, as shown in Figure 26. The cylinder is initially located at the center of the computational domain. The outflow boundary condition $\partial \mathbf{u}/\partial \mathbf{n} = 0$ is applied at the domain contours. A uniform mesh of 560×560 is adopted for the fluid domain and the cylinder is represented by 126 points due to $\delta_s \approx h$. The transient no-slip velocity boundary condition at the cylinder surface is enforced by present MIBM at each time level

$$u(t) = -2\pi f A cos(2\pi f t). \tag{83}$$

The pressure and vorticity contours at four different phases ($\phi = 2\pi f t = 0°, 96°, 192°, 288°$) are shown in Figure 27, where two counter-rotating vortices are formulated during the oscillation. The vortices contours are drawn from -3 to 3 with an increment of 0.4, which display the same structure as in [55].

Figure 28 shows the profiles of the velocity components u and v at four different stream-wise locations ($x = -0.6D$, $0D$, $0.6D$, $1.2D$) for three phase ($\phi = 2\pi f t = 180°, 210°, 330°$). The experimental results of [55] by LDA measurements are also plotted for comparison. The velocity profiles outside the cylinder agree well those of [55]. The only discrepancy is the velocity inside the cylinder. Since the present IBM treats the solid domain as fluid, the velocity is non-zero inside the cylinder. From Figure 28 we can see that this treatment, however, does not influence the flow field outside the solid. Various internal treatments of the body have been discussed in the work of [56], such as applying the force inside the body and thus changing the velocity distribution. Iaccarino and Verzicco [56] also concluded that for direct forcing IBM, there is essentially no difference. Therefore, for simple implementation we just leave the interior of the solid free to develop a flow without imposing anything.

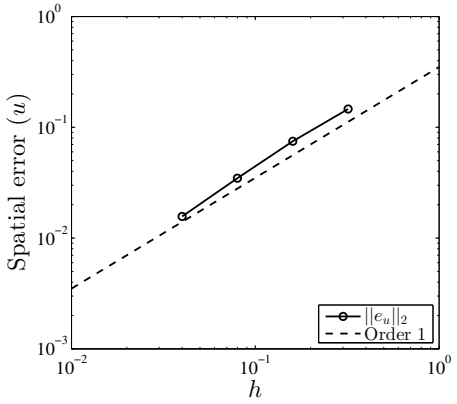


Figure 29. L_2 norm of the horizontal velocity component (u) versus grid spacing for the oscillating cylinder problem.

Figure 29 shows the results of convergence study on a domain of $[-2D, 2D] \times [-2D, 2D]$. A time step of 10^{-4} is selected and the calculation is performed for 2000 time steps. A slightly better than first order accuracy is found in this case.

5.5. Flow around a flapping wing

In this example, we investigate the flow induced by a flapping wing, in order to demonstrate the ability of current method for handling non-circular object in both translational and rotational motions. The configuration of this problem is shown in Figure 30. The hovering wing is a geometrical 2D ellipse with major axis c (chord length) and minor axis b. The aspect ratio is defined as $e = c/b$. The wing is initially located at the origin with an angle of attack of θ_0, then shifts along a stroke plane inclined at an angle β. The translational and rotational motions of the hovering wing are described as follows

$$A(t) = \frac{A_0}{2}\left[cos(\frac{2\pi t}{T}) + 1\right], \tag{84}$$

$$\theta(t) = \theta_0\left[1 - sin(\frac{2\pi t}{T} + \phi_0)\right], \tag{85}$$

where A_0 is the translational amplitude, $2\theta_0$ the rotational amplitude, T the flapping period and ϕ_0 the phase difference. The chord length c and the maximum velocity $U_{\max} = \pi A_0/T$ along the flapping path are used as the length and the velocity scales, respectively. The Reynolds number is defined as $Re = U_{\max}c/\nu$. We employ the same parameters as used in [57, 51, 58]: $c = 1, e = 4$, $A_0 = 2.5c$, $\theta_0 = \pi/4$, $T = \pi A_0/c$, $\beta = \pi/3$, $\phi_0 = 0$, $Re = 157$.

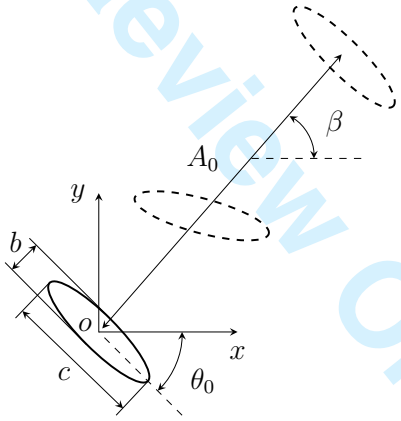


Figure 30. Configuration for flow over a flapping wing.

As suggested by Yang and Stern [58], this simulation is performed on a large square domain of $[-24c, 24c] \times [-24c, 24c]$ to obtain a better periodicity for the results. A uniform mesh of 2400×2400 is employed to cover the computational domain and the mesh spacing around the wing is $0.02c$, which is slightly finer than the grid resolution used in [51, 58]. A larger time step is selected in the present study ($\Delta t = 0.01$) based on the CFL number ($\text{CFL}_{\max} = 0.72$), while a much smaller time step $\Delta t = 0.001$ is used in the immersed interface method (IIM) of Xu and Wang [51] to reduce the body shape distortion.

Figure 31 shows the vorticity fields near the flapping wing in one flapping period at four different positions, which are very similar to those given in [57, 51, 58]. A pair of leading and trailing edge vortices of opposite rotation is formed into a dipole. The dipole moves downward, generating the

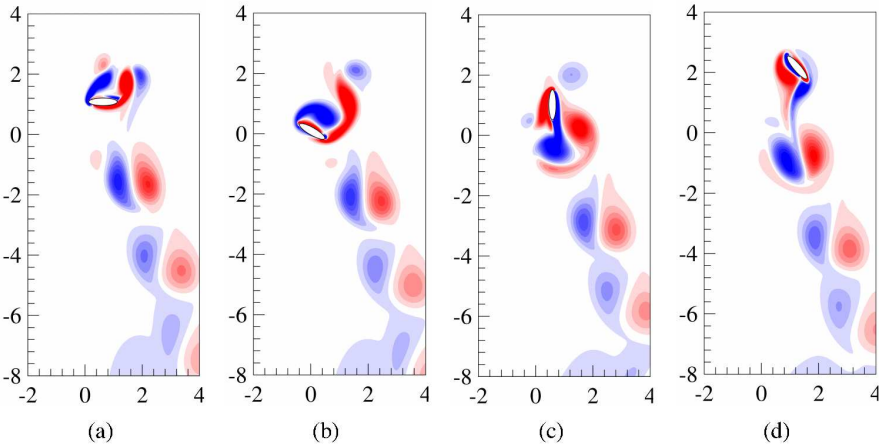


Figure 31. Snapshots of the vorticity fields around a flapping wing at $Re = 157$ for four different positions: (a) $t = 0.25T$; (b) $t = 0.44T$; (c) $t = 0.74T$; (d) $t = 0.99T$.

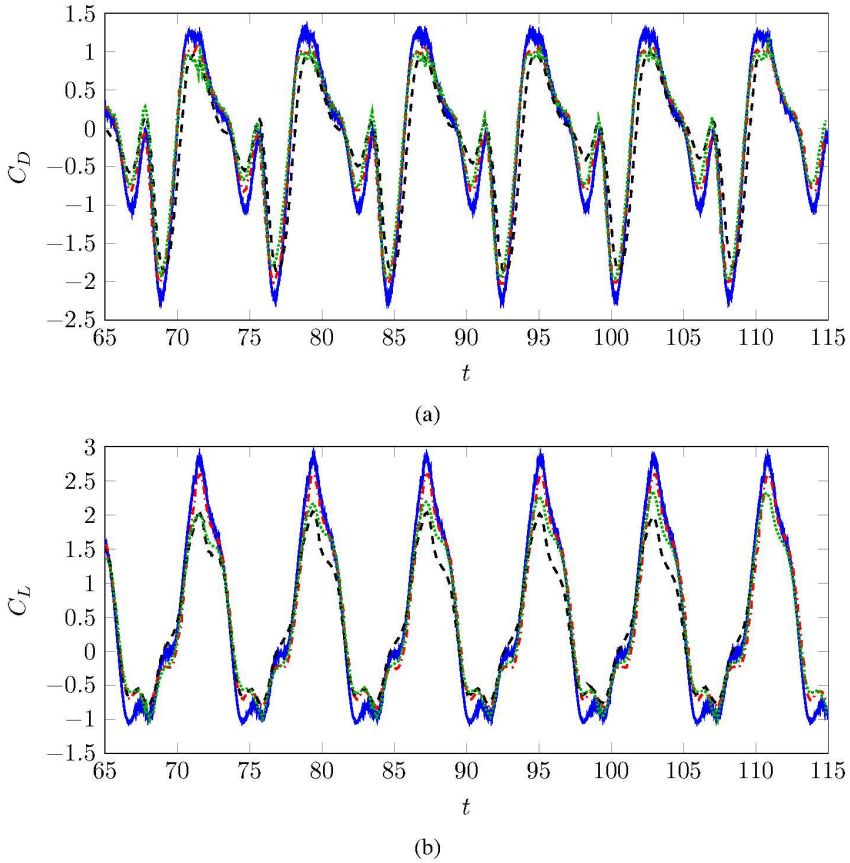


Figure 32. Time history of drag and lift coefficients for the problem of flow around a flapping wing at $Re = 157$. ——, present MIBM; — · —·, the IBM of Yang and Stern [58]; - - - -, the body-conforming mesh method of Wang [57]; · · · ·, the IIM of Xu and Wang [51].

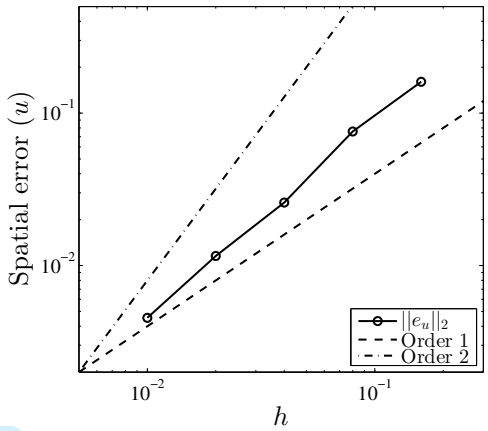


Figure 33. L_2 norm error of the horizontal velocity component (u) as a function of grid spacing for the flapping wing problem.

lift of the wing. The vortices shed from the wing by the self-induced flow, without interfering the new vortices in the next cycle.

The time history of the drag and lift coefficients are plotted in Figure 32 and compared to the results of [57, 51, 58]. Good agreements have been found. Note that in order to maintain the shape of the rigid body in the immersed interface method of [51], a feedback control technique is employed and the time step is kept small to reduce the shape distortion. The present immersed boundary method is found to be much more satisfactory, since no additional springs for feedback control are needed and the no-slip boundary condition is exactly imposed at the interface.

A grid convergence study is also conducted to assess the accuracy of current MIBM in this case. A domain size of $[-4D, 4D] \times [-4D, 4D]$ is chosen and the grid spacing varies sequentially. The numerical solution after one flapping period is used for the analysis. A fine time step of 10^{-4} is selected in order to ensure the analysis is not influenced by the temporal discretization error. Figure 33 shows the error of the horizontal velocity in L_2 norm as a function of the grid spacing. A convergence rate of around 1.29 is observed.

5.6. Flow past an impulsively started cylinder

As our last example we present results of a suddenly accelerated circular cylinder in a quiescent fluid at different Reynolds numbers $Re = U_0 D/\nu$ ranging from 40 to 3000, with U_0 being the cylinder moving velocity. Initially we place the cylinder with unit diameter ($D = 1$) at the origin and suddenly set it into motion to the left at a constant velocity $U_0 = -1$, as illustrated in Figure 34.

We first consider the Reynolds number $Re = 40$ and compare our results to the IBPM of Taira and Colonius [17]. A uniform grid is used to cover the computational domain with no-slip boundary condition applied at all outer boundaries. The grid resolution is $h = 0.01D$ and the time step is set to $\Delta t = 0.001$. Two computational domains are employed to examine the effect of finite domain size on the results, namely a large domain of $[-16.5D, 13.5D] \times [-15D, 15D]$ as used by Taira and Colonius [17] and a relative smaller domain $[-8D, 4D] \times [-5D, 5D]$ as used by Mimeau *et al.* [50]. The time history of the drag coefficient is plotted in Figure 36a from $t = 0$ to 3.5. Our

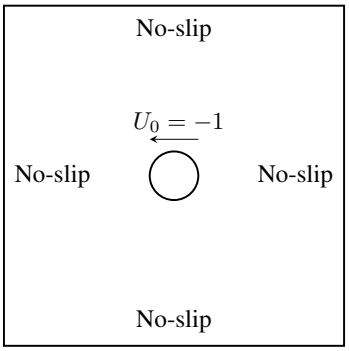


Figure 34. Sketch of the flow past an impulsively started cylinder.

results are in excellent agreement with the immersed boundary projection method [17] on the large computational domain. When the computational domain is reduced the resulting drag coefficient is increased, which has also been observed in previous test cases. The snapshots of the vorticity field are shown in Figure 35a. Good agreements have been found compared to IBPM of Taira and Colonius [17].

At this regime, a grid convergence study has been performed on a domain $[-2D, 2D] \times [-2D, 2D]$. The time step is set to $\Delta t = 0.0001$ and the grid spacing changes sequentially. The numerical errors are computed at $t = 0.5$ based on a very fine grid. Figure 37 shows the variation of the L_2 norm error for the horizontal velocity as a function of the grid spacing. A little better than first order spatial accuracy is observed.

Next we increase the Reynolds number to $Re = 550$ and compare our results to the vortex methods of Koumoutsakos and Leonard [59] and Mimeau *et al.* [50]. In this case, the computational domain $[-8D, 4D] \times [-5D, 5D]$ is used and the mesh resolution is set to $h = 0.005D$ as suggested by Mimeau *et al.* [50]. The time step $\Delta t = 0.001$ is used. The time evolution of drag coefficient is displayed in Figure 36b. The current method has difficulties in drag prediction at early times of impulsive motion, which is also encountered by the immersed boundary projection method of Taira and Colonius [17] and the vortex penalization method of Mimeau *et al.* [50]. At later stage, our results are comparable to those using vortex method. The corresponding vorticity fields are shown in Figure 35b, which compare well with the simulation results of [50, 52, 59, 60].

At $Re = 1000$, the grid is further refined to $h = 0.0025D$ in order to solve the very thin boundary layer, while the computational domain $[-8D, 4D] \times [-5D, 5D]$ is kept unchanged. The time step is reduced to $\Delta t = 0.0005$. As mentioned by Mimeau *et al.* [50], the two-dimensional simulation performed here is valid since only the impulsive start of the flow is considered before the onset of three-dimensional instabilities. Figure 36c and Figure 35c show the drag time evolution and the snapshots of vortex structures at different stages, respectively. We notice that the predicted drag coefficient with present method is slightly higher than that with vortex methods [50, 59]. This can be attributed to the finite domain size used in the present study.

Finally we increase the Reynolds number to $Re = 3000$. At this Reynolds number, the simulation is quite challenging as it requires a very fine grid to capture the boundary layer. We reduce the grid size to $h = 0.00125D$ and adjust the time step respectively to $\Delta t = 0.0002$. Due to memory limits, we select a much smaller computational domain $[-4D, 2D] \times [-3D, 3D]$. The temporal

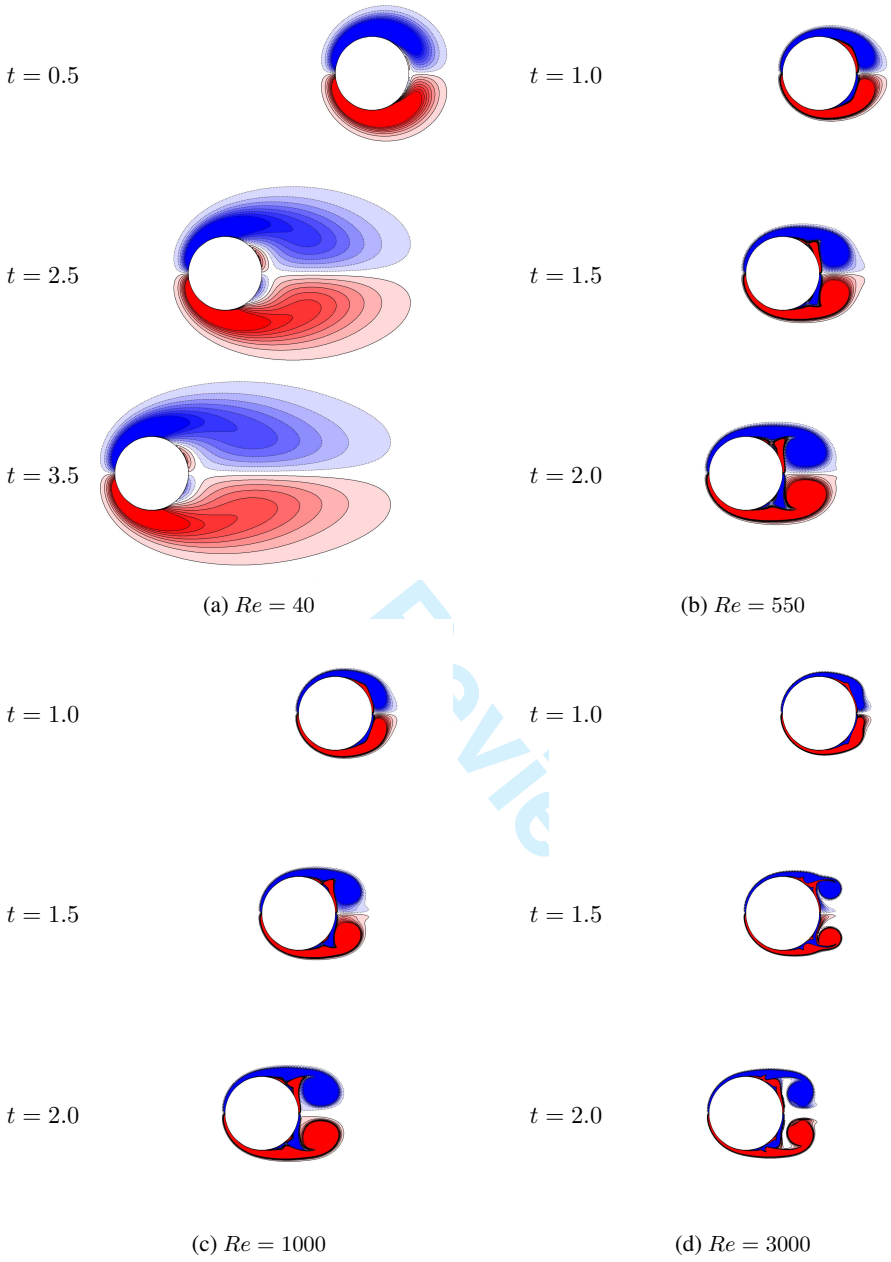
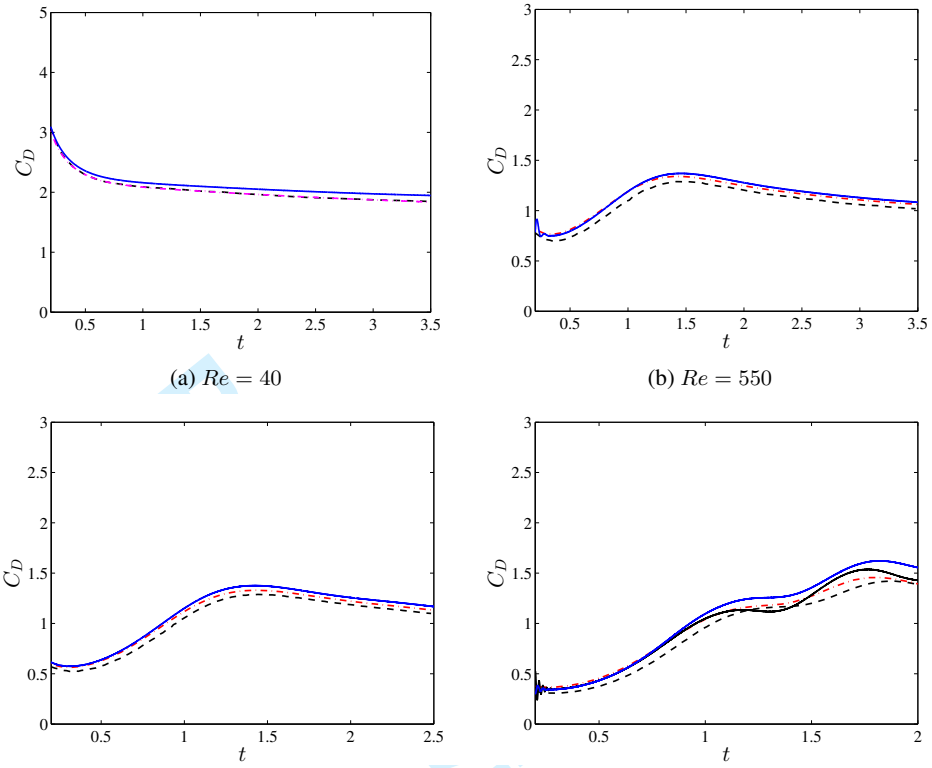


Figure 35. Computed vorticity contours for a suddenly started cylinder at different stages in the start-up process. Contour levels are set from -3 to 3 in increments of 0.4.

history of the drag coefficient is shown in Figure 36d along with the results computed with past settings $\Omega = [-8D, 4D] \times [-5D, 5D], h = 0.0025D$. Even though the magnitude of the predicted drag coefficient with current MIBM is higher than that with vortex methods because of the small domain size, the variation follows well the benchmark results. The corresponding vorticity fields



(a) $Re = 40$ (b) $Re = 550$

(c) $Re = 1000$ (d) $Re = 3000$

Figure 36. Time history of the drag coefficient for flow past an impulsively started cylinder at different Reynolds numbers. (a): – – –, results of Taira and Colonius [17]; – · –·, present $\Omega = [-16.5D, 13.5D] \times [-15D, 15D]$; ——, present $\Omega = [-8D, 4D] \times [-5D, 5D]$. (b) and (c): – – –, results of Koumoutsakos and Leonard [59]; – · –·, results of Mimeau *et al.* [50]; ——, present $\Omega = [-8D, 4D] \times [-5D, 5D]$. (d): —— (black), present $\Omega = [-8D, 4D] \times [-5D, 5D], h = 0.0025D$; —— (blue), present $\Omega = [-4D, 2D] \times [-3D, 3D], h = 0.00125D$; – – –, results of Koumoutsakos and Leonard [59]; – · –·, results of Mimeau *et al.* [50].

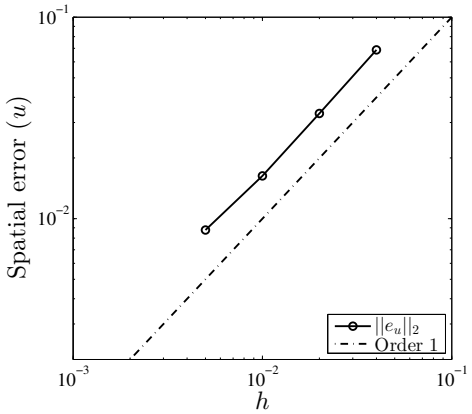


Figure 37. L_2 norm error of velocity (u) for the impulsively started cylinder problem.

are shown in Figure 35d at different time levels, which are in close agreement with those reported by the vortex methods [50, 59, 60].

6. CONCLUSIONS

We presented a new implicit but very efficient formulation of immersed boundary method for simulating incompressible viscous flow past complex stationary or moving boundaries. The current method treats the boundary force and the pressure as Lagrange multipliers for satisfying the no-slip and the divergence-free constraints. The fractional step method is applied to decouple the pressure as well as the boundary force from the fluid velocity field, and the two Lagrange multipliers are solved separately within their own systems. The main advantages of current approach are the accurate imposition of the no-slip condition and the efficiency in computation. The system matrices are well conditioned and generic solvers can be used directly. Especially for moving boundaries, only the boundary force coefficient matrix is updated while the coefficient matrices of velocity and pressure remain unchanged. Even though we have only dealt with rigid boundary in this article, deformable body with its motion known *a priori* can also be handled. A variety of distinct two dimensional flows are simulated and the results are in excellent agreement with available data sets in the literature, demonstrating the fidelity of the proposed method.

ACKNOWLEDGEMENT

The first author greatly acknowledges the financial support of the China Scholarship Council. The calculations were performed on the platform PILCAM2 at the Université de Technologie de Compiègne and the HPC at the Université de Strasbourg.

REFERENCES

1. Peskin CS. Flow patterns around heart valves: A numerical method. *J. Comput. Phys.* 1972; **10**(2):252–271.
2. Wang X, Liu WK. Extended immersed boundary method using FEM and RKPM. *Comput. Methods Appl. Mech. Engrg.* 2004; **193**:1305–1321.
3. Zhang L, Gerstenberger A, Wang X, Liu WK. Immersed finite element method. *Comput. Methods Appl. Mech. Engrg.* 2004; **193**:2051–2067.
4. Liu WK, Liu Y, Farrell D, Zhang L, Wang XS, Fukui Y, Patankar N, Zhang Y, Bajaj C, Lee J, *et al.*. Immersed finite element method and its applications to biological systems. *Comput. Methods Appl. Mech. Engrg.* 2006; **195**:1722–1749.
5. Beyer RP, Leveque RJ. Analysis of a one-dimensional model for the immersed boundary method. *SIAM J. Numer. Anal.* 1992; **29**(2):332–364.
6. Goldstein D, Handler R, L S. Modeling a no-slip flow boundary with an external force field. *J. Comput. Phys.* 1993; **105**(2):354–366.
7. Saiki E, Biringen S. Numerical simulation of a cylinder in uniform flow: Application of a virtual boundary method. *J. Comput. Phys.* 1996; **123**(2):450–465.
8. Fadlun E, Verzicco R, Orlandi P, Mohd-Yusof J. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *J. Comput. Phys.* 2000; **161**(1):35–60.
9. Mohd-Yosuf J. Combined immersed Boundary/B-spline methods for simulation of flow in complex geometries. *Annu. res. briefs*, Center for Turbulence Research 1997.
10. Uhlmann M. An immersed boundary method with direct forcing for the simulation of particulate flows. *J. Comput. Phys.* 2005; **209**(2):448–476.
11. Tseng YH, Ferziger J. A ghost-cell immersed boundary method for flow in complex geometry. *J. Comput. Phys.* 2003; **192**:593–623.

12. Wang K, Rallu A, Gerbeau JF, Farhat C. Algorithms for interface treatment and load computation in embedded boundary methods for fluid and fluid-structure interaction problems. *Int. J. Numer. Meth. Fluids* 2011; **67**:1175–1206.
13. Lakshminarayan V, Farhat C, Main A. An embedded boundary framework for compressible turbulent flow and fluid-structure computations on structured and unstructured grids. *Int. J. Numer. Meth. Fluids* 2014; **76**:366–395.
14. Kempe T, Fröhlich J. An improved immersed boundary mehod with direct forcing for the simulation of particle laden flows. *J. Comput. Phys.* 2012; **231**(9):3663–3684.
15. Luo K, Wang Z, Fan J, Cen K. Full-scale solutions to particle-laden flows: Multidirect Forcing and immersed boundary method. *Phys. Rev. E* 2007; **76**:066 709.
16. Breugem WP. A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows. *J. Comput. Phys.* 2012; **231**:4469–4498.
17. Taira K, Colonius T. The immersed boundary method: A projection approach. *J. Comput. Phys.* 2007; **225**(2):2118–2137.
18. Ji C, Munjiza A, Williams JJR. A novel iterative direct-forcing immersed boundary method and its finite volume applications. *J. Comput. Phys.* 2012; **231**(4):1797–1821.
19. Cai SG, Ouahsine A, Favier J, Hoarau Y. Improved implicit immersed boundary method via operator splitting. *Computational Methods for Solids and Fluids*, Ibrahimbegovic A (ed.). chap. 3, Springer Verlag, 2016; 49–66.
20. Cai SG, Ouahsine A, Favier J, Hoarau Y. Implicit immersed boundary method for fluid-structure interaction. *La Houille Blanche* 2017; **1**:33–36.
21. Cai SG. Computational fluid-structure interaction with the moving immersed boundary method. PhD Thesis, Université de Technologie de Compiègne 2016.
22. Chorin AJ. Numerical solution of the Navier-Stokes equations. *Math. Comp.* 1968; **22**(104):745–762.
23. Témam R. Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires (II). *Arch. Rational Mech. Anal.* 1969; **33**(5):377–385.
24. Perot JB. An Analysis of the Fractional Step Method. *J. Comput. Phys.* 1993; **108**(1):51–58.
25. E W, Liu JG. Projection method I: Convergence and numerical boundary layers. *SIAM J. Numer. Anal.* 1995; **32**(4):1017–1057.
26. Guermond J, Minev P, Shen J. An overview of projection methods for incompressible flows. *Comput. Methods Appl. Mech. Engrg.* 2006; **195**(44-47):6011–6045.
27. Guermond JL, Quartapelle L. On stability and convergence of projection methods based on pressure Poisson equation. *Int. J. Numer. Meth. Fluids* 1998; **26**(9):1039–1053.
28. Goda K. A multistep technique with implicit difference schemes for calculating two- or three-dimensional cavity flows. *J. Comput. Phys.* 1979; **30**(1):76–95.
29. Braza M, Chassaing P, Minh HH. Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder. *J. Fluid Mech.* 1986; **165**:79–130.
30. Van Kan J. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM J. Sci. Stat. Comp.* 1986; **7**(3):870–891.
31. Armfield S, Street R. An analysis and comparison of the time accuracy of fractional-step methods for the navier-stokes equations on staggered grids. *Int. J. Numer. Meth. Fluids* 2002; **38**(3):255–282.
32. Timmermans LJP, Minev PD, Vosse FNVD. An approximate projection scheme for incompressible flow using spectral elements. *Int. J. Numer. Meth. Fluids* 1996; **22**(7):673–688.
33. Chow E, Saad Y. Approximate inverse preconditioners via sparse-sparse iterations. *SIAM J. Sci. Comput.* 1998; **19**(3):995–1023.
34. Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Eijkhout V, Gropp WD, Kaushik D, *et al.*. PETSc users manual. *Technical Report ANL-95/11 - Revision 3.6*, Argonne National Laboratory 2015.
35. Dalton S, Bell N, Olson L, Garland M. Cusp: Generic parallel algorithms for sparse matrix and graph computations 2014.
36. Kassiotis C, Ibrahimbegovic A, Niekamp R, Matthies HG. Nonlinear fluid–structure interaction problem. Part I: implicit partitioned algorithm, nonlinear stability proof and validation examples. *Comput. Mech.* 2011; **47**(3):305–323.
37. Kim J, Kim D, Choi H. An immersed-boundary finite-volume method for simulations of flow in complex geometries. *J. Comput. Phys.* 2001; **171**(1):132–150.
38. Vanella M, Balaras E. A moving-least-squares reconstruction for embedded-boundary formlations. *J. Comput. Phys.* 2009; **228**(18):6617–6628.
39. Pinelli A, Naqavi IZ, Piomelli U, Favier J. Immersed-boundary methods for general finite-difference and finite-volume Navier-Stokes solvers. *J. Comput. Phys.* 2010; **229**(24):9073–9091.

40. Favier J, Revell A, Pinelli A. A Lattice Boltzmann–Immersed Boundary method to simulate the fluid interaction with moving and slender flexible objects. *J. Comput. Phys.* 2014; **261**(0):145–161.
41. Toja-Silva F, Favier J, Pinelli A. Radial basis function (RBF)-based interpolation and spreading for the immersed boundary method. *Comput. Fluids* 2014; **105**:66–75.
42. Peskin CS. The immersed boundary method. *Acta Numer.* 2002; **11**:479–517.
43. Roma AM, Peskin CS, Berger MJ. An adaptive version of the immersed boundary method. *J. Comput. Phys.* 1999; **153**(2):509–534.
44. Coutanceau M, Bouard R. Experimental determination of the main features of the viscous flow in the wake of a circular cylinder in uniform translation. Part 1. Steady flow. *J. Fluid Mech.* 1977; **79**(2):231–256.
45. Tritton DJ. Experiments on the flow past a circular cylinder at low Reynolds numbers. *J. Fluid Mech.* 1959; **6**(4):547–567.
46. Wang S, Zhang W. An immersed boundary method based on discrete stream function formulation for two- and three-dimensional incompressible flows. *J. Comput. Phys.* 2011; **230**(9):3479–3499.
47. Lai MC, Peskin C. An immersed boundary method with formal second-order accuracy and reduced numerical viscosity. *J. Comput. Phys.* 2000; **160**(2):705–719.
48. Williamson C. Oblique and parallel modes of vortex shedding in the wake of a circular cylinder at low Reynolds numbers. *J. Fluid Mech.* 1989; **206**:579–627.
49. Liu C, Zheng X, Sung CH. Preconditioned Multigrid Methods for Unsteady Incompressible Flows. *J. Comput. Phys.* 1998; **139**(1):35–57.
50. Mimeau C, Gallizio F, Cottet GH, Mortazavi I. Vortex penalization method for bluff body flows. *Int. J. Numer. Meth. Fluids* 2015; **79**:55–83.
51. Xu S, Wang ZJ. An immersed interface method for simulating the interaction of a fluid with moving boundaries. *J. Comput. Phys.* 2006; **216**(2):454–493.
52. Mittal R, Dong H, Bozkurttas M, Najjar F, Vargas A, von Loebbecke A. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. *J. Comput. Phys.* 2008; **227**(10):4825–4852.
53. Apte SV, Martin M, Patankar NA. A numerical method for fully resolved simulation (FRS) of rigid particle–flow interactions in complex flows. *J. Comput. Phys.* 2009; **228**(8):2712 – 2738.
54. Mittal S, Kumar V. Flow-induced vibrations of a light circular cylinder at reynolds numbers 10^3 to 10^4. *J. Sound Vibration* 2001; (923-946).
55. Dütsch H, Durst F, Becker S, Lienhart H. Low-Reynolds-number flow around an oscillating circular cylinder at low Keulegan-Carpenter numbers. *J. Fluid Mech.* 1998; **360**:249–271.
56. Iaccarino G, Verzicco R. Immersed boundary technique for turbulent flow simulations. *Appl. Mech. Rev.* 2003; **56**(3):331–347.
57. Wang ZJ. Two dimensional mechanism for insect hovering. *Phys. Rev. Lett.* 2000; **85**(10):2216–2219.
58. Yang J, Stern F. A simple and efficient direct forcing immersed boundary framework for fluid–structure interactions. *J. Comput. Phys.* 2012; **231**(15):5029–5061.
59. Koumoutsakos P, Leonard A. High-resolution simulations of the flow around an impulsively started cylinder using vortex methods. *J. Fluid Mech.* 1995; **296**:1–38.
60. Ploumhans P, Winckelmans G. Vortex methods for high-resolution simulations of viscous flow past bluff bodes of general geometry. *J. Comput. Phys.* 2000; **165**:354–406.