Immersed boundary methods and their applicability in wind energy

Krishnan, Navaneetha; Viré, Axelle; Schmehl, Roland; Van Bussel, Gerard

DOI
10.1088/1742-6596/1618/3/032013

Publication date
2020

Document Version
Final published version

Published in
Journal of Physics: Conference Series

Immersed boundary methods and their applicability in wind energy

Immersed boundary methods and their applicability in wind energy

Navaneetha Krishnan, Axelle Viré, Roland Schmehl, and Gerard van Bussel

Faculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, 2629 HS Delft, The Netherlands

E-mail: a.c.vire@tudelft.nl

Abstract. Airborne wind energy systems often use kites made of thin membranes to save material costs and increase mobility. However, this design choice increases the complexity of the aeroelastic behaviour of the system and demands high-fidelity tools. On the aerodynamic side of the multi-physics problem, it is quite challenging to create a high quality body conforming grid due to the complexity of the geometry and the degree of deformation it undergoes. Immersed boundary methods (IBMs) are quite popular in fluid-structure interaction (FSI) problems that involve arbitrarily deforming bodies with complex geometries and are more tolerant to deformations compared to mesh deforming methods like ALE. This paper will look at some of the popular IBMs, outline criteria to evaluate their applicability, and discuss the limitations they have in fulfilling those in problems involving thin membranes.

1. Introduction

Aeroelasticity is a significant area of study in the development of advanced wind energy systems. This fluid-structure interaction (FSI) phenomenon plays a key role in scaling up wind turbines in a cost-effective and safe way. With designs varying from rigid wing kites with onboard turbines to flexible membrane kites that convert power at the ground station, airborne systems are a lighter and less stiff subset in wind energy systems. Key arguments for airborne wind energy systems (AWES) are significant material savings, reduced environmental impact, mobility, and the ability to reach wind power at much higher altitudes [1]. However, as material savings increase from conventional wind turbines to membrane kites, so does the complexity of the aeroelastic interactions. Existing analytical and computationally less demanding solutions for understanding the aeroelastic behaviour of wind energy systems make several assumptions that do not hold for AWES. Thus, the study of AWES requires the use of high-fidelity tools. This paper looks only at the fluid-dynamics part of the FSI problem and discusses the following criteria that has to be fulfilled by IBMs to be relevant in wind energy: 1) ability to impose strong velocity boundary conditions, 2) ability to avoid the smearing of solution across thin bodies, 3) quality of the obtained pressure solution, and 4) computational cost.

The immersed boundary method (IBM) is a computational fluid dynamics (CFD) approach that is suitable for complex and deforming geometries such as flexible membrane wings under aerodynamic load. In an IBM, the Navier-Stokes equations are discretised on a computational grid extending over the space occupied by both the fluid and the structure [9, 12]. A representation of the structure is obtained on this grid via a projection or interpolation. Variants

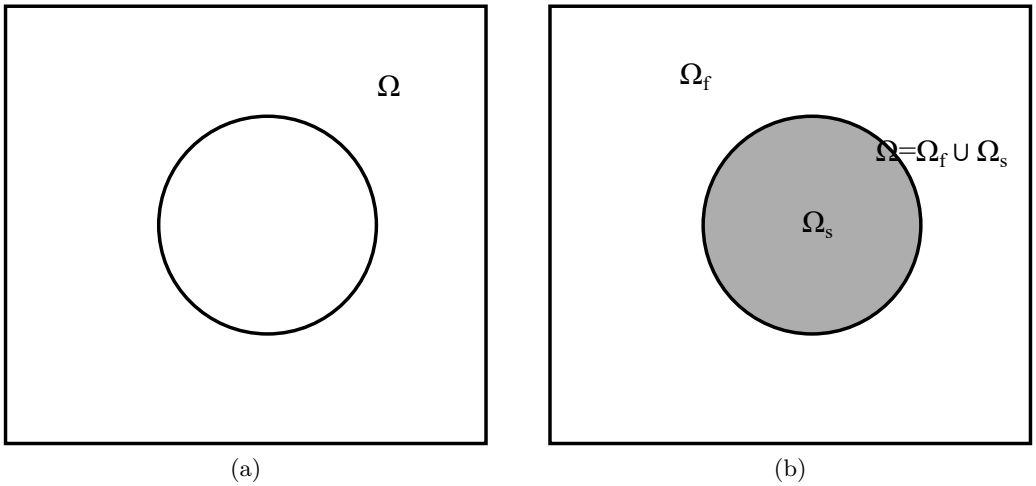


Figure 1: Schematic representation of the computational domain (Ω) in (a) body conforming CFD simulations and (b) immersed boundary simulations. The shaded region in (b) represents the area occupied by the immersed solid. Additionally, (b) shows solid (Ω_s) and fluid (Ω_f) subdomains in the case of the domain decomposed IBM.

of the IBM differ in how they use this information in the fluid equations. The most popular IBMs fall under a class termed, forcing IBMs. Forcing IBMs are popular mainly because they treat the information from the structure as corrections that have to be made in the fluid solution. In this work, we compare some IBMs and discuss their viability as an aeroelastic analysis tool for wind energy. We also demonstrate a novel IBM that can overcome the shortcomings of existing IBMs and can be applied to complex geometries that are encountered in wind energy related problems.

2. Immersed Boundary Methods

The classical approach to numerically solve a fluid dynamics problem is to solve the discrete Navier-Stokes equations on a computational grid that excludes the region occupied by the solid. Figure 1a shows a 2D cylinder surrounded by a fluid. The region outside the cylinder occupied by the fluid is the computational domain, Ω. The computational domain is discretised to solve the governing equations and the quality of the generated mesh plays a significant role in the accuracy of the results obtained. As the geometries become complex it becomes increasingly difficult to create a high quality mesh. Furthermore, in FSI problems, every grid has a bearable range of movement after which the quality of the grid degrades, and in extreme cases, the cells start to collapse. Such problems need advanced tools like overset chimera grids, octree grids, adaptive mesh refinement, etc. IBM's approach to this problem is to simplify the computational grid and project the necessary fields onto it.

Unlike with the classical approach, in an IBM, the computational grid (Ω) extends over regions occupied by both the fluid Ω_f, and the solid Ω_s (Figure 1b). The first step in an IBM is to get an approximation of the immersed solid on the background mesh. It is an approximation because the background grid will almost never match exactly the immersed solid that has to be projected. After the projection, the presence of the solid is numerically mimicked such that the surrounding fluid behaves as if a boundary exists in the region covered by the solid. In the next sections, we will look at some of the most popular IBM variants and the way they achieve this.

2.1. Forcing IBMs
The essence of forcing methods is to add a forcing term (f) to the fluid's momentum equations to mimic the presence of the immersed body and make changes to the velocity field as imposed by the solid, and can be written as,

$$\int_{\Omega} \left(\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{uu}) - \nabla \cdot (\bar{\bar{\kappa}} \nabla \mathbf{u}) + \nabla p - \mathbf{f} \right) dV = 0, \tag{1}$$

where, $\mathbf{u}$ is the velocity vector, $\bar{\bar{\kappa}}$ is the viscosity tensor, p is the fluid pressure, and $\mathbf{f}$ is the forcing term. Continuity is left untouched as,

$$\int_{\Omega} (\nabla \cdot \mathbf{u}) dV = 0. \tag{2}$$

Notice that we assumed an incompressible fluid in this paper. Eqns. (1) and (2) are continuous and coupled. In order to be solved numerically, they are converted into discrete forms and usually, decoupled to reduce the computational effort. The projection method [3] is a commonly used method to decouple time dependent coupled equations. The first step in the projection method is to compute an intermediate velocity field ($\mathbf{u}^*$) using pressure from the previous time step. If it is the first time step, an initial guess can be optionally made by solving a pressure Poisson's system.

$$(\mathbf{M} + \Delta t(\mathbf{A} - \mathbf{D})) \frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -(\mathbf{A} - \mathbf{D})\mathbf{u}^n + \mathbf{C}p^n + \mathbf{f} \tag{3}$$

Then, using the intermediate velocity field a Poisson's system is solved to advance pressure ($p^{n+1} = p^n + \Delta p^n$).

$$\mathbf{L}\Delta p^n = \frac{\mathbf{C}^T \mathbf{u}^*}{\Delta t} \tag{4}$$

Finally, a correction is made to make velocity divergence free.

$$\mathbf{u}^{n+1} = \mathbf{u}^* + \Delta t \mathbf{M}^{-1} \mathbf{C} \Delta p^n \tag{5}$$

In Eqns. (3), (4), and (5), $\mathbf{M}$ is the mass matrix and $\mathbf{A}$, $\mathbf{D}$, $-\mathbf{C}$, $\mathbf{C}^T$, and $\mathbf{L}$ are the advection, diffusion, gradient, divergence, and Laplacian operators, respectively [11]. As shown below, different forcing methods vary based on the manner in which $\mathbf{f}$ is determined or treated in Eqn. (3).

2.1.1. Penalty forcing As the name suggests, penalty forcing method is a penalising error correction method [2]. The penalty force ($\mathbf{f}$) is defined as,

$$\mathbf{f} = \beta \alpha^s \frac{\mathbf{u}^s - \mathbf{u}^n}{\Delta t}. \tag{6}$$

In Eqn. (6), β is a relaxation factor and α^s is a scalar mask function used for the representation of the structure on the computational grid, or physically, a solid concentration field. $\mathbf{u}^s$ is the velocity vector of the immersed solid. While the scalar field α^s is key to properties such as sharpness of the interface and conservation of fields exchanged, the relaxation factor β is useful, because with proper tuning, stability of the simulations and accuracy of the results can be improved [12]. Variants of the penalty forcing method use different definitions for α^s and β [6, 15]. In this paper, we use a consistent interpolation function to define α^s and set the relaxation factor as one.

2.1.2. Semi-implicit forcing In Eqn. (3), and penalty forcing method, $\mathbf{f}$ is treated as an explicit term, i.e., $\mathbf{f}$ is computed only using the solution from the previous time step. However, considering the non-linearity of the forcing term, some methods add a certain degree of implicitness to the evaluation. In a semi-implicit method [13], the forcing term is removed from Eqn. (3) to get another intermediate velocity field ($\mathbf{u}^{**}$). $\mathbf{u}^{**}$ is then used to compute the forcing term,

$$\mathbf{f} = \alpha^s \frac{\mathbf{u}^s - \mathbf{u}^{**}}{\Delta t}, \tag{7}$$

and make a prior correction to velocity before going to the pressure correction step (Eqn. (5)).

$$\mathbf{u}^* = \mathbf{u}^{**} + \Delta t \mathbf{f} \tag{8}$$

2.1.3. Direct forcing If we look at Eqn. (3) as a balance of forces (inertia, convection, diffusion, and pressure gradient), the definitions of the forcing term we have seen so far only balances out the inertial term. In a direct forcing method [10, 4] the forcing term also includes all the previously unaccounted forces,

$$\mathbf{f} = \alpha^s \left(\frac{\mathbf{u}^s - \mathbf{u}^*}{\Delta t} + (\mathbf{A} - \mathbf{D})\mathbf{u}^* - \mathbf{C}p^n \right). \tag{9}$$

Furthermore, both the forcing methods we discussed before introduce an additional staggered step in the temporal integration scheme for the evaluation of the forcing term. However, since in the direct forcing method $\mathbf{f}$ is treated in an implicit manner, its evaluation does not introduce additional staggered steps, nor does it require the use of a relaxation term for stability or accuracy.

2.2. Lagrange multiplier

Unlike all the other methods we have discussed so far where velocity boundary conditions from the immersed solid are enforced in the prediction step (Eqn. (3)), a Lagrange multiplier method enforces velocity boundary conditions in the projection step (Eqn. (4)) [5, 14]. The system of Equations in (1) and (2) can be written in matrix form as,

$$\begin{bmatrix} \mathbf{H} & -\mathbf{C} & \mathbf{E}^T \\ -\mathbf{C}^T & 0 & 0 \\ \mathbf{E} & 0 & 0 \end{bmatrix} \begin{pmatrix} \mathbf{u} \\ p \\ \mathbf{f} \end{pmatrix} = \begin{pmatrix} \mathbf{R} \\ 0 \\ \mathbf{u}^s \end{pmatrix}, \tag{10}$$

where, $\mathbf{H}$ is the fluid operator that accounts for the inertial, convection, and diffusion terms in Eqn. (1), $\mathbf{R}$ is the right hand side of the equation, and $\mathbf{E}$ is a symmetric interpolation operator to project fields from the solid onto the computational domain. Here, a Lagrange multiplier method makes the observation that gradient and interpolation operators are Lagrange multipliers that act on pressure and boundary forces, respectively. Thus, the system can be simplified as,

$$\begin{bmatrix} \mathbf{H} & \mathbf{Q} \\ \mathbf{Q}^T & 0 \end{bmatrix} \begin{pmatrix} \mathbf{u} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{R} \\ \mathbf{R}_\lambda \end{pmatrix}, \tag{11}$$

where, $\mathbf{Q} = \begin{bmatrix} -\mathbf{C} & \mathbf{E}^T \end{bmatrix}$, $\lambda = \begin{pmatrix} p & \mathbf{f} \end{pmatrix}^T$, and $\mathbf{R}_\lambda = \begin{pmatrix} 0 & \mathbf{u}^s \end{pmatrix}^T$. This system can be solved using a projection algorithm as,

$$\mathbf{H}\mathbf{u}^* = \mathbf{R} - \mathbf{Q}\lambda^n \tag{12}$$

$$\mathbf{L}_\lambda \Delta\lambda^n = \mathbf{Q}^T\mathbf{u}^* - \mathbf{R}_\lambda \tag{13}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \mathbf{M}^{-1}\mathbf{Q}\Delta\lambda^n. \tag{14}$$

The above time integration scheme is identical to Eqns. (3), (4), and (5), except for the fact that boundary conditions are imposed in step 3 rather than in step 1. However, the size of the Poisson's system in step 2, although symmetric and comparatively easier to solve, has increased by a multiple of $(dim + 1)$. It is possible to avoid the simplification from Eqn. (10) to (11) and solve for the two Lagrange multipliers individually. But, breaking down the Poisson's system into two smaller pieces would increase the number of staggered steps in the time integration scheme and would require additional nonlinear steps to reach the same amount of accuracy.

2.3. Domain decomposed IBM

All the methods we have discussed so far evaluate the fluid equations throughout the computational domain, Ω. However, the authors argue that it is unnecessary to solve the fluid equations in elements that are fully covered by the immersed solid. Instead, it should be enough to enforce the governing equations of the fluid only in the region occupied by the fluid. Domain decomposed IBM [7] uses the solid concentration field, α^s, to decompose the computational domain, Ω, into solid and fluid subdomains, Ω_s and Ω_f, respectively (Figure 1b). In each domain, corresponding governing equations are defined as follows.

$$\int_{\Omega_f} \left(\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) - \nabla \cdot (\bar{\bar{\kappa}} \nabla \mathbf{u}) + \nabla p \right) dV = 0 \tag{15}$$

$$\int_{\Omega_s} \left(\frac{\partial \mathbf{u}}{\partial t} \right) dV = \int_{\Omega_s} \left(\frac{\partial \mathbf{u}^s}{\partial t} \right) dV \tag{16}$$

$$\int_{\Omega_f} (\nabla \cdot \mathbf{u}) dV = 0 \tag{17}$$

The main differences compared to Eqns. (1) and (2) are as follows. In Eqn. (15) there is no forcing term, Eqn. (16) is just velocity prescription from the solid, and Eqn. (17) imposes the divergence condition only in the fluid subdomain, Ω_f. Equations (15) and (16) impose a strong velocity boundary condition and Eqn. (17) imposes divergence free condition without introducing slip in the solid subdomain.

Other than the above mentioned methods, there are many more variations of the IBM. However, one of the main criterion we used while selecting methods for this study was the ability of a method to solve problems on unstructured grids. Resolving boundary layer for arbitrarily aligned bodies can be computationally heavy on Cartesian grids. Thus, for the matters of interest in this paper, it is sufficient to limit the evaluation to the aforementioned IBMs. All the tests are run in Fluidity [11] - a finite element based CFD code - and the mesh is discretised using piecewise continuous linear triangular elements.

3. Evaluation

3.1. Accuracy of the imposed velocity

The following qualitative test cases demonstrate the effects of Reynolds number and thickness of the immersed solid on the accuracy of the velocity solution. Figure 2(a) shows the velocity solution in the region occupied by a rigid stationary NACA0012 aerofoil. The aerofoil is placed at angle of attack 0° in a $25c \times 20c$ domain with open boundary conditions; c is the chord length. Chord-based Reynolds number of the test case is 1000. The aerofoil is stationary and therefore, the projected solid velocity is zero. Nevertheless, because of the strong advection forces, there is significant momentum seepage into the body. Furthermore, this permeable behaviour of the body reduces the reaction force obtained from the fluid and translates into reduced body forces. For the sake of completion, it is worth mentioning that β is set as one in our test cases. Figure

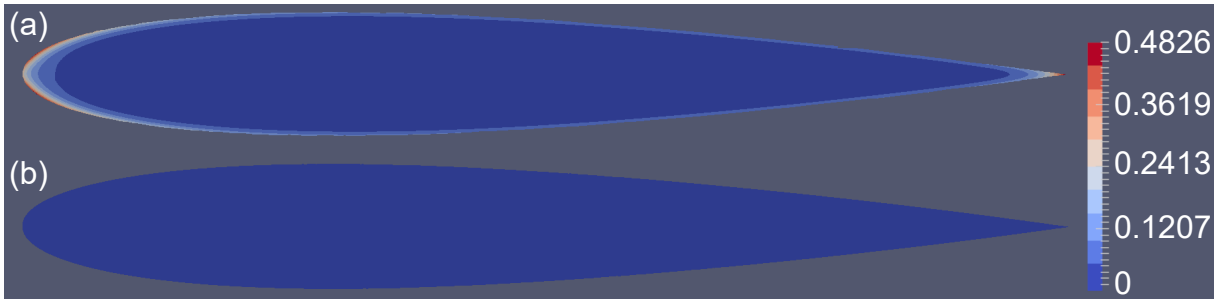
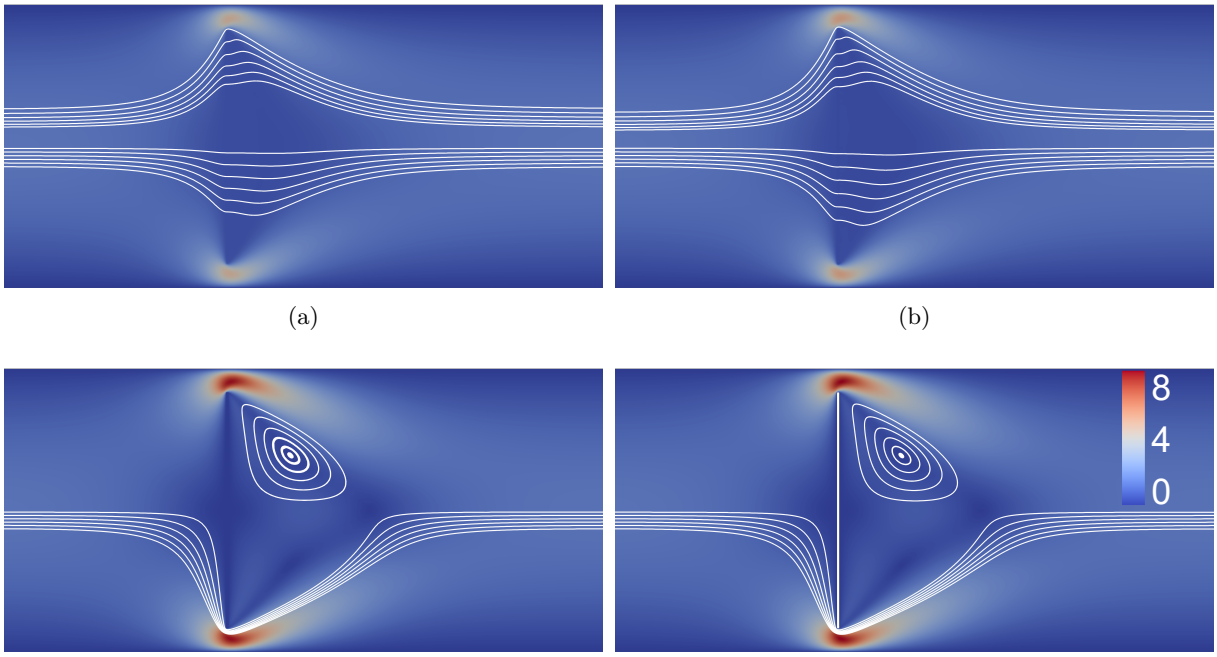


Figure 2: Flow past NACA0012 at Re=1000 modelled using (a) penalty forcing method and (b) domain decomposed method. The figure shows the magnitude of non-dimensionalised velocity inside the solid domain.



(a) (b)

(c) (d)

Figure 3: Non-dimensionalised velocity field and streamlines in a channel flow with an orthogonally placed membrane modelled using (a) penalty forcing method, (b) direct forcing method, (c) domain decomposed method, and (d) body conforming CFD at steady state.

2(b) shows the same using the domain decomposed method and shows accurate imposition of boundary conditions.

Figure 3 shows a thin rigid stationary plate placed orthogonally in a low Reynolds number channel flow. The test is setup as a channel flow so that the side spill from the plate is clearly visible, and at the inlet, a parabolic velocity profile ($u(0, y)$) is prescribed with peak velocity in the middle.

$$u(0, y) = 1.5U \frac{y(1.2h - y)}{(\frac{1.2h}{2})^2} \tag{18}$$

The Reynolds number based on the mean inlet velocity (U) and height of the plate (h) is 20.

The thickness of the plate is 1% of the height and the coefficient 1.2 besides h accounts for the 10% clearance between the plate and the channel walls on either sides. The plate is located at h units downstream of the inlet. Furthermore, the inlet velocity profile increases smoothly over an initial ramp up period as

$$u(0,y,t) = \begin{cases} u(0,y)\frac{1-cos(\pi t)}{2} & \text{if } t < 1 \\ u(0,y) & \text{if } t >= 1 \end{cases}. \tag{19}$$

Here, t is the non-dimensionalised time. Similar to the NACA0012 test case, the desired velocity conditions are zero in the region occupied by the solid. However, unlike the thick NACA0012 that showed some permeability, the thickness of the body is insufficient for the penalty forcing method (Figure 3a) and direct forcing method (Figure 3b) to enforce the boundary conditions, and allows the flow to penetrate through the thin plate. By plotting streamlines in the middle of the channel where convection forces are the strongest, we can see in Figure 3 how effective each of the methods are at diverting the flow. Figure 3d shows the solution from a body conforming simulation for reference. We expect the plate to fully divert the flow to the sides and a pair of vortices develop in the wake. Streamlines in the penalty forcing method (Figure 3a) show very small deflections and passes through the plate. In the case with direct forcing (Figure 3b), the deflections in the streamline are slightly larger but still lets part of the momentum through. In both the cases, the momentum seepage prevents the wake vortices from developing. Figure 3c shows the same using the domain decomposed method and behaves as desired.

In the projection algorithm that all three IBMs use, velocity is modified in the prediction and correction steps. In the decreasing order of significance, using an explicit forcing term will introduce a slip in the prediction step, and not enforcing velocity boundary conditions in the correction step will introduce a slip at that stage as well. Penalty forcing fails at both and produces the largest amount of slip, direct forcing neglects the second condition and allows a bearable amount of slip, and finally, the domain decomposed method avoids slip in both the steps.

3.2. Solution for the pressure field

All IBMs project a solid velocity which they try to enforce onto the fluid in the region occupied by the immersed solid. Unlike velocity, pressure has no defined solution inside this region. However, from the projection method, in Eqn. (4), we can see that pressure is the result of the Poisson's system that enforces continuity on the incompressible fluid. Almost all IBMs, except those like cut-cell methods, extend the continuity assumption to the region occupied by the immersed solid as well, and let the Poisson's system solve for a pressure field that is continuous across the immersed body. Figure 4 plots the variation of pressure along the centreline ($y = 1.1h$) in the thin plate test case that we discussed in the previous section. The shaded region in the figure shows the region occupied by the thin plate. The black markers denote nodes that are at the interface and the white markers denote nodes that are fully immersed in the solid. Pressure side is on the left and suction side is on the right. The red curve shows the pressure variation across the plate from a penalty forcing simulation, and the blue curve shows that from a direct forcing simulation. Both the pressure curves are continuous across the thin plate, however with a significant difference in their values. Because, penalty forcing method fails to enforce velocity conditions strongly, it under-predicts the pressure drop across the plate. As a result of stronger velocity boundary conditions, in the case of the pressure solution from the direct forcing simulation, pressure drop is larger compared to the same from penalty forcing. Still, because the Poisson's system is unnecessarily searching for a pressure solution that is continuous across the thin plate, discrete pressure equations of the elements on the pressure side and that on the suction side affect each other's outcome. This would be

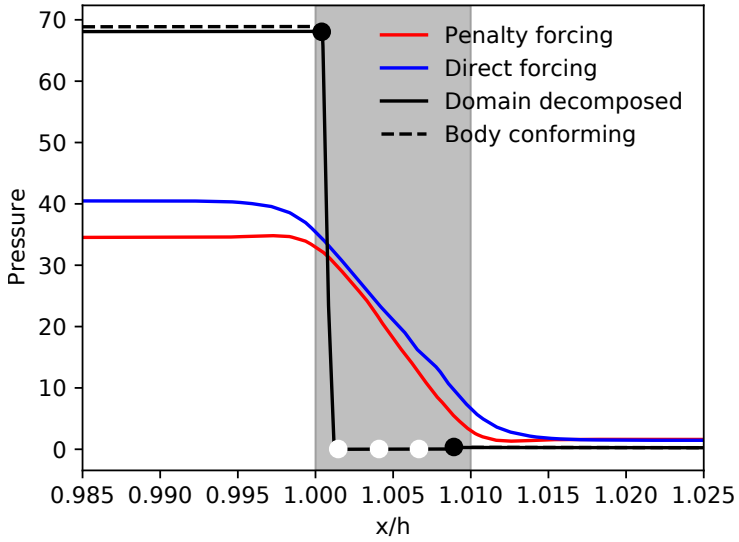


Figure 4: Variation of pressure across the immersed plate along the centreline ($y = 1.1h$). Shaded region denotes the region occupied by the plate. The black markers denote nodes that are at the interface and the white markers denote nodes that are fully immersed in the solid. Unlike the forcing methods, the domain decomposed IBM allows discontinuities in the pressure field.

an acceptable approximation for problems involving thick immersed solids when there are a sufficient number of elements inside the solid region to accurately resolve the pressure variation across the body. However, for problems with thin bodies or large pressure variations across the body, this assumption introduces errors into the pressure solution and thus also into the velocity solution. Going back to Figure 3b, we can see that despite the momentum solve step (Eqn. (3)) enforcing strong velocity boundary conditions, the direct forcing method fails to make the thin plate impermeable. Inaccuracies in the pressure projection (Eqn. (4)) leads to inaccuracies in the velocity correction (Eqn. (5)).

The alternate option, as employed in the domain decomposed method, is to solve the Poisson's system and correct velocity only in the region occupied by the fluid. This leaves the velocity prescribed in Eqn. (16) unaffected by the pressure solution, but leads to a discontinuous pressure field. In Figure 4, the black curve shows the pressure variation across the thin plate from a domain decomposed simulation. On both the sides of the plate there are discontinuities in the pressure field, and since there is no defined value for pressure inside the solid subdomain, pressure is set to zero in that region (white nodes in Figure 4). Unlike in the forcing methods which sought for a continuous pressure solution, this discontinuity keeps the discrete equations on either side from interacting with one another. Thus, in the absence of the unnecessary continuity constraint inside the immersed solid, the computed pressure drop across the plate is also higher.

Because of the discontinuities at the interface and the fictitious pressure values inside the solid region, it is inaccurate to evaluate pressure gradients at the interface. This is allowable as long as the discontinuity stays inside the solid region and does not corrupt the solution in the fluid region. But in cases with moving immersed solids, such an assumption cannot be made. Thus, although the choice of making the pressure field discontinuous at the solid/fluid interface leads to a better pressure solution, the pressure gradient term ($\mathbf{C}p^n$) in Eqn. (3) makes the projection algorithm as described in Eqns. (3), (4), and (5) unusable for problems with moving immersed

solids. To solve this, we modify the projection algorithm as follows. First, the momentum equations are solved without including the pressure gradient to get an intermediate velocity field.

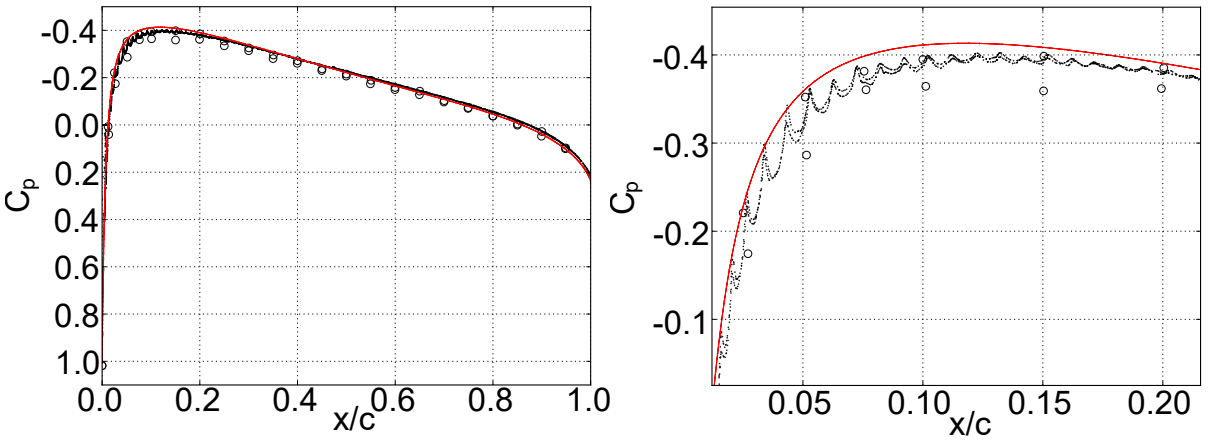
$$(\mathbf{M} + \Delta t(\mathbf{A} - \mathbf{D}))\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -(\mathbf{A} - \mathbf{D})\mathbf{u}^n + \mathbf{f} \tag{20}$$

Then, instead of solving for a pressure correction, a Poisson's system is solved for the actual pressure.

$$\mathbf{L}p^{n+1} = \frac{\mathbf{C}^T\mathbf{u}^*}{\Delta t} \tag{21}$$

This new pressure field matches with the current position of the immersed solid and the discontinuities are inside the solid subdomain. Finally, the pressure gradient is used to throw the velocity into a divergence free space.

$$\mathbf{u}^{n+1} = \mathbf{u}^* + \Delta t\mathbf{M}^{-1}\mathbf{C}p^{n+1} \tag{22}$$



(a) (b)

Figure 5: Pressure coefficient along the surface of a NACA0012 at an angle of attack of 0° and a Reynolds number of 6 million. The circles are experimental results [8], the red line is from a body conforming CFD simulation, and the black line shows results using the IBM. In (a) variation along the entire chord is shown and (b) shows a closer look at the leading edge of the aerofoil where the oscillations are the strongest.

However, a disadvantage in choosing to solve for the pressure field only in the fluid subdomain is that the irregularities in the projection becomes visible in the pressure solution. At low Reynolds numbers or with bluff bodies, viscous forces smoothen this out. But for streamlined bodies at high Reynolds numbers, mismatch between the background grid and the immersed boundary causes perturbations in the pressure field near the surface of the solid. Figure 5 plots the pressure coefficient on the surface of a NACA0012 aerofoil at an angle of attack of 0° in a flow with a chord-based Reynolds number of 6 million. The circular markers are experimental results [8] and the red line is the result of a body conforming CFD simulation. The black line is the pressure curve from the domain decomposed immersed boundary simulation. Both the CFD simulations use k-ω SST turbulence modelling. Although the curve is smooth for the major part of the chord length and agrees with the reference results, there are visible oscillations in the curve near the leading tip of the aerofoil (Figure 5b). Other than the incongruity between the

immersed and computational grids, the strength of the oscillations seem to have a correlation with the magnitude of the pressure gradient.

This dilemma of whether to keep the flow variable continuous or discontinuous at the interface between the solid and fluid subdomains is not exclusive to pressure, but to all flow variables that do not have a physical/defined value inside the solid region to prescribe. For example, when dealing with high Reynolds number flows and using RANS models, turbulent kinetic energy (k) and turbulent dissipation (ϵ) have defined values at the solid boundary, and can be extended to continuously prescribe the solution inside the solid subdomain as well. However, variables like turbulent frequency (ω) have no defined value inside the solid region. The choice depends entirely on the problem at hand.

3.3. Computational performance

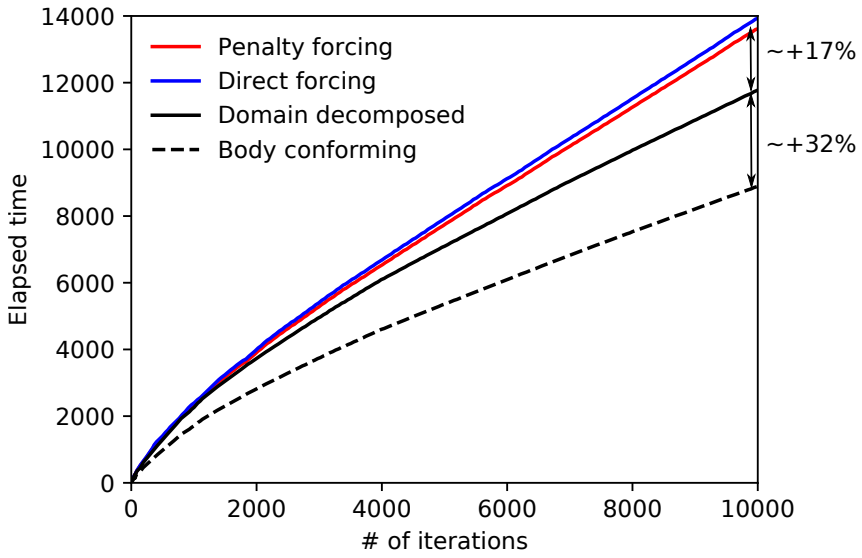


Figure 6: Performance comparison of penalty forcing (red line), direct forcing (blue line), and domain decomposed (black line) IBMs with respect to a body conforming simulation (black dashed line). Caveat: relative performance is highly dependent on the size of the immersed solid and the number of nodes that are in the solid region.

The flexibility that IBMs offer in the grid generation process takes a toll in the computational cost. Figure 6 shows the time elapsed for a penalty forcing (red line), direct forcing (blue line), and domain decomposed (black line) IB simulation with respect to a body conforming simulation (black dashed line). The plots are from the NACA0012 at Re=1000 test case that we discussed in section 3.1. The domain decomposed method took $\sim$ 32% more computational time with respect to the body conforming simulation, while, penalty forcing and direct forcing IBMs took $\sim$ 17% on top of that. The main reason IBMs are computationally costlier than body conforming simulations is that they execute over an extended domain spanning both solid and fluid regions. Depending on the size of the immersed solid, for the same level of grid resolution at the interface, the degrees of freedom associated with an IB simulation can be considerably higher. In the case at hand, by extending the computational domain over the region occupied by the aerofoil, the degrees of freedom involved went up by $\sim$ 26% for all IB cases. It is worth mentioning that using adaptive mesh refinement this mark up can be brought down significantly. But, we keep a fixed grid so that the IBMs can be compared on a level field.

Compared to the domain decomposed IBM, the other two takes $\sim 17\%$ more time. The main reason the domain decomposed IBM outperforms the forcing methods is because in the discrete system of equations the entries associated with fully immersed nodes are diagonal; both in the momentum equations and the pressure equations. Thus, the system of equations involved in the domain decomposed IBM is sparser and leads to a faster convergence.

4. Conclusions

The IBM is a powerful numerical approach for solving FSI problems involving complex, arbitrarily deforming structures. Their defining feature is the capability to do that without the need to generate body conforming grids that deform with the body. IBMs have already been applied to a wide range of fluid flow problems in engineering and biology. As we discussed in this paper, several variants exist with varying degrees of complexity, advantages and disadvantages, and applications. Where the conventional CFD approach falls short, IBMs provide a viable option. A major limitation of IBMs is in their inability to maintain grid resolution near the boundary and thus, they fail at high Reynolds number problems involving streamlined objects. For such cases, the conventional body-fitting CFD approach is still superior. To overcome this drawback, future research should focus on coupling IBMs with adaptive mesh refinement algorithms. Enabling IBMs to handle high Reynolds number problems will make it feasible to simulate and analyse advanced wind energy devices.

Acknowledgments

Part of this research was supported by the European Union Seventh Framework Programme (FP7/2007-2013) under a Marie Curie Career Integration Grant (Grant Agreement PCIG13-GA-2013-618159). A. Viré also acknowledges support from the Rijksdienst voor Ondernemend Nederland (RVO) through the TSE Hernieuwbare Energie funding scheme (ABIBA project). R. Schmehl is financially supported by the AWESCO (H2020-ITN-642682) and REACH (H2020-FTIPilot-691173) projects. This work was partly carried out on the Dutch national e-infrastructure with the support of SURF Cooperative (project number 17215).

References

[1] U. Ahrens, M. Diehl, and R. Schmehl, editors. *Airborne wind energy*. Green Energy and Technology. Springer, 2013.
[2] E. Arquis and J. P. Caltagirone. Sur les conditions hydrodynamiques au voisinage d'une interface milieu fluide-milieux poreux: application à la convection naturelle. In *Comptes Rendus de l'Academie des Sciences*, volume 299, pages 1–4, 1984.
[3] A.J. Chorin. Numerical solution of the Navier-Stokes equations. *Mathematics of Computation*, 22:745–762, 1968.
[4] E.A. Fadlun, R. Verzicco, P. Orlandi, and J. Mohd-Yusof. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics*, 161(1):35–60, jun 2000.
[5] R. Glowinski, T.W. Pan, and J. Périaux. Distributed Lagrange multiplier methods for incompressible viscous flow around moving rigid bodies. *Computer Methods in Applied Mechanics and Engineering*, 151(1):181–194, jan 1998.
[6] D. Goldstein, R. Handler, and L. Sirovich. Modeling a no-slip flow boundary with an external force field. *Journal of Computational Physics*, 105(2):354–366, 1993.
[7] N. Krishnan, A. Viré, R. Schmehl, and G. van Bussel. An immersed boundary method based on domain decomposition. *Computers & Fluids*, 202:104500, 2020.
[8] C.L. Ladson. Effects of independent variation of Mach and Reynolds numbers on the low-speed aerodynamic characteristics of the NACA 0012 airfoil section. Technical Report NASA-TM-4074, NASA, 1988.
[9] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37(1):239–261, 2005.
[10] J. Mohd-Yusof. Combined immersed-boundary/b-spline methods for simulations of flow in complex

geometries. In *Center for Turbulence Research Annual Research Briefs*, pages 317–327. Stanford University, 1997.

[11] M.D. Piggott, G.J. Gorman, C.C. Pain, P.A. Allison, A.S. Candy, B.T. Martin, and M.R. Wells. A new computational framework for multi-scale ocean modelling based on adapting unstructured meshes. *International Journal for Numerical Methods in Fluids*, 56:1003–1015, 2008.

[12] F. Sotiropoulos and X. Yang. Immersed boundary methods for simulating fluid-structure interaction. *Progress in Aerospace Sciences*, 65:1–21, 2014.

[13] S.-W. Su, M.-C. Lai, and C.-A. Lin. An immersed boundary technique for simulating complex flows with rigid boundary. *Computers & Fluids*, 36(2):313–324, feb 2007.

[14] K. Taira and T. Colonius. The immersed boundary method: a projection approach. *Journal of Computational Physics*, 225:2118–2137, 2007.

[15] A. Viré, J. Xiang, F. Milthaler, P.E. Farrell, M.D. Piggott, J.-P. Latham, D. Pavlidis, and C.C. Pain. Modelling of fluid-solid interactions using an adaptive mesh fluid model coupled with a combined finite discrete element model. *Ocean Dynamics*, 62(10-12):1487–1501, 2012.