Chapter 1
Immersed Boundary Projection Methods

Benedikt Dorschner and Tim Colonius

1.1 Overview of the Immersed Boundary Method

Most conventional numerical methods to simulate complex fluid–structure interaction problems utilize body-conforming discretizations, where the fluid–solid interface conditions are imposed as boundary conditions. In its most general formulation, the fluid–structure interface can both be moving and deforming as a result of the two-way coupling between fluid and structure. Common body-fitted approaches include arbitrary Lagrangian–Eulerian formulations (Hirt et al. 1974; Ahn and Kallinderis 2006) or space–time finite element methods (Tezduyar et al. 1992, 2006).

The generation of body-fitted meshes for complex, possibly moving and deforming geometries, is computationally expensive and requires sophisticated procedures to avoid severe mesh distortion and preserve accuracy (Thompson et al. 1998; Hermansson and Hansbo 2003; Tezduyar et al. 2006; Nakata and Liu 2012). An alternative is the use of non-conforming meshes, the most widely used example being the immersed boundary method (IB). The IB method was first proposed in Peskin (1972) to simulate blood flow inside a heart with flexible valves. In the IB method, the flow field is described on a non-conforming Eulerian grid. The immersed surface is represented in a Lagrangian framework, and the surface traction is determined by imposing the no-slip boundary condition on the Eulerian velocity field interpolated to the surface. In the continuous setting, the surface traction is a singular function (defined only on the surface) and is discretized by a smeared, discrete delta function that regularizes the forcing effect over the neighboring Eulerian grid cells.

B. Dorschner · T. Colonius (✉)
Department of Mechanical and Civil Engineering, California Institute of Technology,
Pasadena, CA 91125, USA
e-mail: colonius@caltech.edu

B. Dorschner
e-mail: bdorschn@caltech.edu

S. Roy et al. (eds.), *Immersed Boundary Method*, Computational Methods
in Engineering & the Sciences, https://doi.org/10.1007/978-981-15-3940-4_1

In the original IB method (Peskin 1972), the heart valves were modeled as flexible membranes, and Hooke's law was used as a constitutive relation to relate the forcing function to the motion of the Lagrangian points. Later, this scheme was extended to rigid bodies by taking large values for the spring constants (Beyer and LeVeque 1992; Lai and Peskin 2000). In addition, the concept of feedback control to compute the force on the rigid immersed surface was introduced by Goldstein et al. (1993), where the difference between the velocity solution and the boundary velocity is used in a proportional-integral controller. Note that for techniques using constitutive relations to model the flow over rigid bodies, the choice of gain (stiffness) is a tuning parameter whose value must be heuristically chosen to simultaneously avoid a restrictive time step size (large stiffness) and slip error (small stiffness).

Constitutive relations are eliminated in direct forcing methods and its variants (Mohd-Yusof 1997; Fadlun et al. 2000), where the momentum forcing is obtained by penalization of the slip at the surface. However, the no-slip condition is only enforced on an intermediate velocity field and hence requires iterations to approximate the no-slip condition on the final velocity fields. While the slip has been reported to be small (Fadlun et al. 2000), it cannot be estimated in a systematic fashion.

An alternative approach is to consider the boundary force as a Lagrange multiplier that is determined to enforce the no-slip condition (Glowinski et al. 1998; Taira and Colonius 2007; Colonius and Taira 2008; Kallemov et al. 2016). In this formulation, the discretized, incompressible Navier–Stokes equations (NSE) can be formulated in an analogous manner to the classical fractional step method by introducing appropriate regularization and interpolation operators. In addition, a modified Poisson equation, where the force and the pressure are lumped together, can be solved to determine the pressure and force unknowns. We refer to these methods as immersed boundary projection methods (IBPM). The advantage of IBPM is that continuity and no-slip conditions can be satisfied implicitly and with arbitrary accuracy at each time step. The Courant number is further only limited by the choice of the time-marching algorithm. With typical splitting methods (fractional step methods), one can achieve second-order accuracy uniformly in time and the matrices arising from the implicit treatment of the viscous terms as well as the modified Poisson equation can be made symmetric and positive definite. The resulting linear system can be solved with an efficient conjugate-gradient solver.

While the standard implementation with discrete delta functions is only first-order accurate in space, there have been efforts to improve the accuracy of IB methods in order to efficiently tackle higher Reynolds number flows. These include so-called sharp-interface and cut-cell approaches (Seo and Mittal 2011). In particular, standard IBs such as ghost-cell methods do not in general conserve mass or momentum at the interface, which manifests in spurious pressure oscillations (Mittal and Iaccarino 2005) and becomes particularly problematic for compressible flows or when coupled with large-eddy simulations and alike. A remedy was found in cut-cell methods, which strictly enforce conservation by reshaping finite volume boundary cells to locally conform with the geometry. A drawback of the cut-cell approach is that the fluid volume fractions of cut cells can become small, necessitating stabilization of the underlying time-stepping scheme using cell-merging (Ye et al. 1999), cell-linking

(Kirkpatrick et al. 2003) or flux redistribution techniques (Hu et al. 2006; Colella et al. 2006). Additionally, cut-cell methods inherit many of the complications of body-fitted meshes and must be adapted dynamically for fluid–structure interaction problems.

In discussing order of accuracy, we must tackle a misunderstanding that has permeated at least through part of the IB literature. Even for rigid bodies, the velocity gradients are not continuous at an immersed surface, and the discretization must account for the derivative singularity to achieve high-order accuracy. If the singular traction is regularized without respect to the discretization (for example by using a discrete delta function), the regularized solution will converge to the continuous one at first order irrespective of the order of accuracy of the schemes used to treat the derivative operators. The regularization error may in principle be made small independently of the discretization error, but then the region over which the surface is smeared must be made arbitrarily thin *compared to the grid spacing*. If the first-order regularization error does not satisfy this restriction, then these "high-order" methods simply converge to the incorrect, smeared solution faster. An interesting approach toward higher-order IB methods is presented in Stein et al. (2017).

In this review, we take the alternative approach of accepting the first-order error near the immersed surface, as this allows mimetic discretizations that achieve other desirable properties, such as stability, discrete conservation, and computational efficiency. It may be possible to achieve second-order accuracy while maintaining these other characteristics in the future.

In what follows, we will introduce the immersed boundary projection method as proposed in Taira and Colonius (2007), Colonius and Taira (2008). We will also present a strongly coupled fluid–structure interaction algorithm as in Goza and Colonius (2017), which is then applied to simulate the flow past an inverted flexible flag. Subsequently, an immersed boundary method based on Lattice Green's functions is introduced, and examples ranging from inclined rotating disks to turbulent flow past a sphere are shown. Finally, some perspectives to further increase the Reynolds number and inclusion of explicit turbulence models within the filtered NSE are provided.

1.2 Immersed Boundary Projection Method

The IB formulation for the incompressible Navier–Stokes equations with an additional singular boundary force $\boldsymbol{f}_\Gamma$ in the momentum equation reads as :

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \boldsymbol{u} + \int\limits_{\Gamma(t)} \boldsymbol{f}_\Gamma(\boldsymbol{X}(\boldsymbol{\xi}, t), t) \delta(\boldsymbol{X}(\boldsymbol{\xi}, t) - \boldsymbol{x}) \mathrm{d}\boldsymbol{\xi}, \tag{1.1a}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1.1b}$$

$$\int\limits_{\Omega} \boldsymbol{u}(\boldsymbol{x}) \delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\xi}, t)) \mathrm{d}\boldsymbol{x} = u_\Gamma(\boldsymbol{\xi}, t) = \frac{\partial \boldsymbol{X}(\boldsymbol{\xi}, t)}{\partial t}, \tag{1.1c}$$

where $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t)$, $p(\boldsymbol{x}, t)$, and Re denote the fluid velocity, pressure, and the Reynolds number, respectively. We assume that the equations have already been made non-dimensional with respect to characteristic length and velocity scales, and the (constant) fluid density.

The Dirac delta function is indicated by δ and Ω is the Eulerian domain. The immersed body surface Γ is described in Lagrangian coordinates $\boldsymbol{X} = \boldsymbol{X}(\boldsymbol{\xi}, t)$, where $\boldsymbol{\xi}$ is the surface parametrization, and $u_\Gamma(\boldsymbol{\xi}, t) = \frac{\partial X(\boldsymbol{\xi},t)}{\partial t}$ is the boundary velocity. The body force $\boldsymbol{f}_\Gamma$ is chosen such that the no-slip condition on the immersed surface, as prescribed by Eq. (1.1c), is satisfied. This step is agnostic to any model for the motion of the surface based on the fluid forces acting on it, which in general determines the *position* of the surface, which can be moving with respect to the underling Eulerian domain.

The convolution with the Dirac delta function Eq. (1.1a) and Eq. (1.1c) couples the immersed surface with the Eulerian grid Ω. The velocity field $\boldsymbol{u}$ as well as the pressure are defined for all $\boldsymbol{x} \in \Omega$ and satisfy the far field boundary conditions $\boldsymbol{u}(\boldsymbol{x}, t) \to \boldsymbol{u}_\infty(t)$ as $|\boldsymbol{x}| \to \infty$.

An issue with this formulation based on surfaces is that it is assumed that fluid resides on either side of the immersed surface. When this is not the case, i.e., the immersed surface comprises a substantial closed volume, there are wasted points in the "ghost fluid" inside the body. Moreover, it is important to remember that the fluid inside the body can exert a force on the body when the surface is accelerated. For example, the added mass on a fluid-filled hollow sphere is different from that of a solid sphere. To the extent that the fluid inside the surface is moving as a rigid body, the additional force can readily be tabulated and used to correct the IB results to non-hollow bodies. For the more general case involving deformation of the IB, this formulation is only directly applicable to thin structures.

1.2.1 Discretization

The spatial discretization may be obtained via a second-order mimetic finite volume method (Nicolaides and Wu 1997; Nicolaides 1992; Perot 2000; Zhang et al. 2002) on a staggered mesh $Q := \{\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{C}\}$, which consists of vertices $\mathcal{V}$, edges $\mathcal{E}$, faces $\mathcal{F}$, and cells $\mathcal{C}$ (see Fig. 1.1). Scalar quantities reside at cell centers and vertices, while faces and edges contain vector flow quantities. Grid functions with values on Q are denoted by $\mathbb{R}^Q$ and functions with vector values at the Lagrangian grid points are denoted by $\mathbb{R}^\Gamma$. The semi-discrete form of Eq. (1.1) reads as

$$M\frac{\mathrm{d}u}{\mathrm{d}t} + N(u,t) = -Gp + \frac{1}{\mathrm{Re}}L_{\mathcal{F}}u + R(t)f \tag{1.2a}$$

$$\bar{D}u = 0 \tag{1.2b}$$

$$R(t)u = u_\Gamma, \tag{1.2c}$$

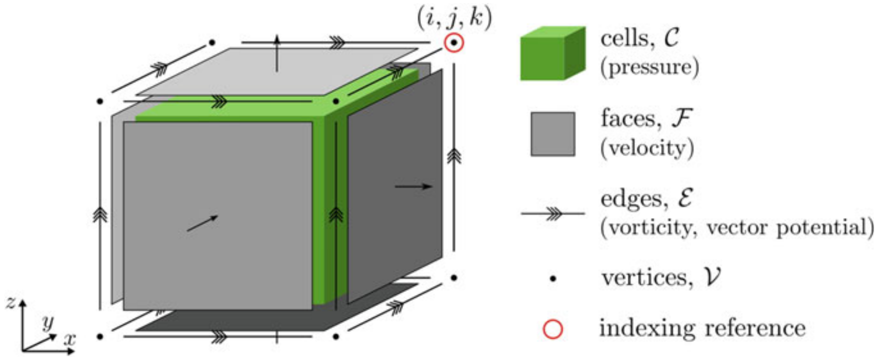


Fig. 1.1 Staggered cell object as used in the IBLGF method. Reprinted from Liska and Colonius (2016) with permission

where $u \in \mathbb{R}^{\mathcal{F}}$ and $p \in \mathbb{R}^{C}$ are the discrete velocity and pressure variables at time $t > 0$. The mass matrix is denoted by M. The discrete gradient, divergence, and Laplace operators are denoted by G, D, and L and, if ambiguous, a subscript is used to identify storage location. The set of discrete vector operators used in the following is given by:

$$\text{Gradient} \qquad G : \mathbb{R}^{C} \to \mathbb{R}^{\mathcal{F}}, \bar{G} : \mathbb{R}^{C} \to \mathbb{R}^{\mathcal{F}}, \tag{1.3}$$

$$\text{Curl} \qquad C : \mathbb{R}^{\mathcal{F}} \to \mathbb{R}^{\mathcal{E}}, \bar{C} : \mathbb{R}^{\mathcal{E}} \to \mathbb{R}^{\mathcal{F}}. \tag{1.4}$$

$$\text{Divergence} \qquad D : \mathbb{R}^{\mathcal{E}} \to \mathbb{R}^{\mathcal{V}}, \bar{D} : \mathbb{R}^{\mathcal{F}} \to \mathbb{R}^{C}, \tag{1.5}$$

$$\text{Laplace} \qquad L : \mathbb{R}^{Q} \to \mathbb{R}^{Q}. \tag{1.6}$$

The convection term $\boldsymbol{u} \cdot \nabla \boldsymbol{u}$ is approximated by the nonlinear operator $N(u, t)$. Different choices for this discretization lead to different (conservation) properties (Perot 2000). The discrete surface functions $f(i, t)$ and $u_\Gamma(i, t)$ denote the force and the velocity of the ith Lagrangian marker at $\boldsymbol{X}(\boldsymbol{\xi}_i, \boldsymbol{t})$ where $i \in [1, N_L]$. The interpolation and regularization operators $E(t)$ and $R(t)$ are time-dependent and constructed by regularizing the δ-function convolutions of Eq. (1.1a) and Eq. (1.1c), i.e., $R(\cdot)$ and $E(\cdot)$ are discretizations of $\int_\Gamma (\cdot)\delta_h(X(\boldsymbol{\xi}, t) - \boldsymbol{x})\mathrm{d}\boldsymbol{\xi}$ and $\int_\Omega (\cdot)\delta_h(X(\boldsymbol{\xi}, t) - \boldsymbol{x})\mathrm{d}\boldsymbol{x}$, respectively (see also Sect. 1.2.3). The interpolation and regularization operators are adjoints under the standard inner product such that $E = (\Delta x)^3 R^\dagger$.

The scheme is second-order accurate and by using a staggered Cartesian grid conserves momentum and either kinetic energy or circulation, depending on the discretization for $N(u, t)$, in the limit of vanishing viscosity and time-stepping errors (Lilly 1965; Morinishi et al. 1998; Perot 2000). Note that the operators G and D can be formulated such that $G = -D^\dagger$. Explicit expression for all operators can be found in Liska and Colonius (2017), Colonius and Taira (2008). For a uniform mesh, the mass matrix M is a constant multiple of the identity.

With these definitions, we can write Eq. (1.2) as a system of algebraic equations as

$$\begin{bmatrix} A & G & -R \\ D & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{bmatrix} u^{n+1} \\ p \\ f \end{bmatrix} = \begin{bmatrix} r^n \\ 0 \\ u_\Gamma^{n+1} \end{bmatrix} + \begin{bmatrix} \mathrm{bc}_1 \\ \mathrm{bc}_2 \\ 0 \end{bmatrix}, \tag{1.7}$$

where the submatrix A is the result of the implicit velocity treatment. Here, it is obtained by the implicit trapezoidal rule on the viscous term yielding $A = \frac{1}{\Delta t}M - \frac{1}{2}L$. The convection term is discretized by the second-order Adams–Bashforth method, leading to the right-hand side $r^n = [\frac{1}{\Delta t}M - \frac{1}{2}L]u^n + \frac{3}{2}N(u^n) - \frac{1}{2}N(u^{n-1})$. The inhomogeneous terms bc_1, bc_2 depend on the particular boundary conditions, which are discussed later. Using the above properties of the submatrices, Eq. (1.7) can be rewritten as

$$\begin{bmatrix} A & G & E^\dagger \\ G^\dagger & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{bmatrix} u^{n+1} \\ p \\ \tilde{f} \end{bmatrix} = \begin{bmatrix} r^n + \mathrm{bc}_1 \\ -\mathrm{bc}_2 \\ u_\Gamma^{n+1} \end{bmatrix}, \tag{1.8}$$

where $\tilde{f}$ is the scaled boundary force, which accounts for the scaling factor when expressing R with $E^\dagger$. The form of Eq. (1.8) is the Karush–Kahn–Tucker (KKT) system, where $(p, \tilde{f})$ is the set of Lagrange multipliers to satisfy a set of kinematic constraints. These constraints are purely algebraic, and there is no need for the pressure and boundary force to be distinguished anymore. Thus, we can group the Lagrange multipliers and the submatrices as $\lambda = [p, \tilde{f}]$ and $Q = [G, E^\dagger]$. The above system is algebraically identical to traditional discretizations of the NSE and allows the use of standard solvers. Here, the (projection) fractional step algorithm is applied to Eq. (1.8), which can be expressed as an approximate LU decomposition of the left side matrix (Perot 2000), which yields the immersed boundary projection method (IBPM):

$$Au^* = r_1, \quad \text{(Solve for intermediate velocity)} \tag{1.9}$$

$$Q^\dagger A_j^\ddagger Q\lambda = Q^\dagger u^* - r_2, \quad \text{(Solve modified Poisson equation)} \tag{1.10}$$

$$u^{n+1} = u^* - A_j^\ddagger Q\lambda \quad \text{(Projection step)}, \tag{1.11}$$

where $A_j^\ddagger$ is the jth order Taylor series expansion of A^{-1} with respect to Δt, and the explicit terms on the right-hand side are denoted by r_1 and r_2. Note that in Taira and Colonius (2007), A and $Q^\dagger A_j^\ddagger Q$ are constructed to be symmetric positive-definite operators such that the system can be solved efficiently with the conjugate-gradient method. Thus, the no-slip boundary condition is enforced on the solution by projecting the intermediate velocity field into the solution space that satisfies both divergence-free and no-slip constraints.

The IBPM is found to be second-order in time and first-order accurate in space.[1] There is no need for a constitutive relation to compute the boundary force. The IBPM therefore does not have any stability restrictions associated with the immersed surface, and the time step restrictions are imposed only by the choice of the marching scheme.

1.2.2 Nullspace Method for the Immersed Boundary Method

The nullspace or discrete streamfunction approach was originally proposed for solving Eq. (1.2) without the immersed boundary (Hall 1985; Chang et al. 2002), where only the incompressibility constraint needs to be satisfied. Using the discrete streamfunction s such that

$$u = Cs, \tag{1.12}$$

where the discrete curl operator C is constructed with column vectors corresponding to the basis of the nullspace of D. It follows that

$$DC = 0, \tag{1.13}$$

which automatically satisfies the incompressibility constraint for all times.

In addition, left-multiplication of the momentum equation with $C^\dagger$ removes the pressure term and thus reduces to a single equation to be solved per time step

$$C^\dagger A C s^{n+1} = C^\dagger (r_1^n + \mathrm{bc}_1). \tag{1.14}$$

Note that solution of the pressure Poisson equation is not required here and therefore the most expensive part of the fractional step method is eliminated, while exactly satisfying the continuity equation. In addition, the errors from the approximate LU decomposition are eliminated, which is why this scheme is also called the *exact fractional step method* (Chang et al. 2002).

Furthermore, a second-order approximation of the circulation is obtained by $\gamma = C^\dagger q$.

Especially in two-dimensional problems, where the streamfunction and vorticity have a single nonzero component, this can lead to a more efficient algorithm, even in the presence of an IB. For 2D problems, including the immersed boundary formalism into the nullspace approach leads to the KKT system (Colonius and Taira 2008)

$$\begin{bmatrix} C^\dagger A C & C^\dagger E^\dagger \\ EC & 0 \end{bmatrix} \begin{bmatrix} s^{n+1} \\ \tilde{f} \end{bmatrix} = \begin{bmatrix} C^\dagger r_1^n \\ u_\Gamma^{n+1} \end{bmatrix}. \tag{1.15}$$

[1] Typically, the first-order errors that are associated with the regularization of the delta functions are limited to a finite region near the surface, and this results in better than first-order accuracy in the L_2 norm.

The left-hand side matrix is symmetric but in general indefinite, which limits the efficiency for direct solutions. However, with the projection (fractional step) approach, we obtain

$$C^{\dagger}ACs^{*} = C^{\dagger}R_{1}^{n} \tag{1.16}$$

$$EC(C^{\dagger}AC)^{-1}(EC)^{\dagger}\tilde{f} = ECs^{*} - u_{\Gamma}^{n+1} \tag{1.17}$$

$$s^{n+1} = s^{*} - (C^{\dagger}AC)^{-1}(EC)^{\dagger}\tilde{f}. \tag{1.18}$$

A direct solution of the above system requires a nested iteration to solve the modified Poisson equation. For stationary bodies, however, one can compute a Cholesky factorization of $EC(C^{\dagger}AC)^{-1}(EC)^{\dagger}$ once, since the system size scales with the number of the immersed boundary points. Then, a system of equations of the form $C^{\dagger}ACx = b$ needs to be solved only once per Lagrangian force.

In Colonius and Taira (2008), it was shown that for a uniform grid with simple boundary conditions, a similar system to Eq. (1.9) can be solved efficiently using fast sine transforms. Assuming that the velocity outside the computational domain is known, simple Dirichlet boundary conditions can be applied to the velocity normal to the sides of the domain, while Neumann boundary conditions are imposed on the velocity tangent to the sides.

No-penetration boundary conditions for the normal component of the velocity and a zero vorticity (or no-stress) condition for the tangent components are natural boundary conditions for external flows, given a sufficiently large domain. With these simplifications, one can write the semi-discrete momentum equation as

$$\frac{\mathrm{d}\gamma}{\mathrm{d}t} = C^{\dagger}E^{\dagger}\tilde{f} = -\beta C^{\dagger}C\gamma + C^{\dagger}N(u) + \mathrm{bc} + \gamma, \tag{1.19}$$

where $Lq = -\beta CC^{\dagger}u = -\beta C\gamma$ has been used. Here, $\beta = 1/\Delta x^{2}\mathrm{Re}$ is constant. Under the aforementioned assumptions, the matrix $-\beta C^{\dagger}C$ corresponds to the standard discrete Laplace operator with zero Dirichlet boundary conditions for γ. This discrete Laplacian can be diagonalized by a sine transform, where the sine transform pair is denoted by

$$\hat{\gamma} = S\gamma \leftrightarrow \gamma = S\hat{\gamma}, \tag{1.20}$$

and $(\hat{\cdot})$ indicates Fourier coefficients. In addition, we use $\Lambda = SC^{\dagger}CS$, where Λ is a diagonal matrix with the eigenvalues of $C^{\dagger}C$, which are positive and known analytically.

Using the same time-marching scheme as above, the system becomes

$$S\left(I + \frac{\beta\Delta t}{2}\Lambda\right)S\gamma^{*} = \left(I - \frac{\beta\Delta t}{2}C^{\dagger}C\right)\gamma^{n} + \frac{\Delta t}{2}\left(3C^{\dagger}N(u^{n}) - C^{\dagger}N(u^{n-1})\right) + \Delta t\mathrm{bc}_{\gamma}, \tag{1.21}$$

$$EC\left(S\Lambda^{-1}\left(I+\frac{\beta\Delta t}{2}\Lambda\right)^{-1}S\right)(EC)^{\dagger}\tilde{f} = ECS\Lambda^{-1}S\gamma^{*} - u_{\Gamma}^{n+1}, \tag{1.22}$$

$$\gamma^{n+1} = \gamma^{*} - S\left(I+\frac{\beta\Delta t}{2}\Lambda\right)^{-1}S(EC)^{\dagger}\tilde{f}. \tag{1.23}$$

The velocity u^n can then be found by

$$u^n = Cs^n + \mathrm{bc}_u, \quad s^n = S\Lambda^{-1}S\gamma^n = \mathrm{bc}_s, \tag{1.24}$$

where each of the boundary conditions involve the assumed known values at the velocity edge.

Note that in the transformed system only one linear system associated with a symmetric positive-definite operator, Eq. (1.22), needs to be solved. In addition, the matrix dimensions are now $N_f \times N_f$, a drastic reduction compared to the original modified Poisson equation Eq. (1.10). A corresponding order of magnitude speedup was measured numerically in Colonius and Taira (2008). Further, if the body is stationary, the modified Poisson equation for the force can be solved efficiently using a triangular Cholesky decomposition.

To conclude, for a uniform grid and simple boundary conditions it is preferable to solve Eqs. (1.21)–(1.23). However, for simulations of external flows the simplified boundary conditions require large computational domains. Since the grid is also required to be uniform, this constraint quickly outweighs its benefits. However, the multi-domain approach as proposed in Colonius and Taira (2008) was found to be an effective solution to approximate the free-space boundary conditions. In Sect. 1.3, an IB method based on lattice Green's function is presented, which alleviates the need of far-field approximation and satisfies the free-space boundaries exactly.

1.2.3 Accurate Calculation of Surface Stresses and Forces

In this section, we will present a procedure to accurately calculate surface stresses and forces in the context of IB methods as proposed in Goza et al. (2016). In particular, we will focus on the set of IB methods that solve for surface stresses by imposing velocity boundary conditions. This is, for example, the case in the IBPM above, but in contrast to the original IB method of Peskin (1972), where the surface stresses are derived from specific constitutive laws. Velocity-based IB methods have been shown to suffer from inaccurate surface stresses and may also exhibit spurious oscillations in time traces, which originate from the ill-posedness of the first-kind integral equation for the surface stresses. It is important to note that the velocity is typically convergent in spite of the poor accuracy of the surface stresses. Thus, the following procedure is important for either post-processing the stress data or when the IB method is used in conjunction with a structural solver (see Sect. 1.2.4).

To understand the origin of the spurious oscillations, we consider the Poisson equation in two dimensions with an unknown singular source term f that takes nonzero values only on the immersed surface Γ as a model problem:

$$\begin{aligned}\nabla^2\varphi(\boldsymbol{x}) &= -\int\limits_{\Gamma} f(\boldsymbol{X}(\boldsymbol{\xi}))\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi}))\mathrm{d}\xi,\\ \varphi(\boldsymbol{x}) &= \varphi_{\partial\Omega}(\boldsymbol{x}),\ \ \boldsymbol{x}\in\partial\Omega,\\ \int\limits_{\Omega}\varphi(\boldsymbol{x})\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi}))\mathrm{d}\boldsymbol{x} &= \varphi_{\Gamma}(\boldsymbol{X}(\boldsymbol{\xi})).\end{aligned} \tag{1.25}$$

While numerical solutions of this equation and their errors have been analyzed for prescribed source terms f (Tornberg and Engquist 2004; Zahedi and Tornberg 2010), it is explicitly solved for in what follows by incorporating the third equation as a boundary constraint in order to mimic velocity-based IB method as closely as possible.

The immersed boundary Γ is taken to be a circle of radius $1/2$ in a unit square and is centered at $\boldsymbol{x}=0$,$\varphi_{\partial\Omega}(\boldsymbol{x}) = 1-\frac{1}{2}\log(2|\boldsymbol{x}|)$, and $\varphi_{\Gamma}(\boldsymbol{X})=1$. The exact solution to (1.25) is given by

$$\varphi_{\mathrm{ex}}(\boldsymbol{x}) = \begin{cases}1 & |\boldsymbol{x}|\le\frac{1}{2},\\ 1-\frac{1}{2}\log(2|\boldsymbol{x}|) & |\boldsymbol{x}|>\frac{1}{2},\end{cases} \tag{1.26}$$

$$f_{\mathrm{ex}}(\boldsymbol{X}) = 1. \tag{1.27}$$

Similar to the integrated surface force in the context of IB, we also define $F_{\mathrm{ex}} = \int_{\Gamma} f_{\mathrm{ex}}(\boldsymbol{X}(\boldsymbol{\xi}))\mathrm{d}\boldsymbol{\xi} = \pi$. As was done in previous sections, the delta function is replaced with a smeared delta function, $\delta_h(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi}))$, which is continuous with nonzero but compact support and is defined in terms of the grid spacing Δx. The numerical solution for a given grid spacing approximates

$$\varphi(\boldsymbol{x}) = -\int\limits_{\Omega}\int\limits_{\Gamma} f(\boldsymbol{X}(\boldsymbol{\xi}'))\delta_h(\boldsymbol{x}'-\boldsymbol{X}(\boldsymbol{\xi}'))G^L(\boldsymbol{x};\boldsymbol{x}')\mathrm{d}\boldsymbol{\xi}'\mathrm{d}\boldsymbol{x}', \tag{1.28}$$

where $G^L(\boldsymbol{x};\boldsymbol{x}')$ denotes the Green's function of the Poisson problem, and δ_h indicates the smeared delta function (see also Sect. 1.3 for further details). To obtain the unknown source term f, Eq. (1.28) is multiplied by δ_h and integrated over the domain Ω:

$$\int\limits_{\Omega}\int\limits_{\Omega}\int\limits_{\Gamma} f(\boldsymbol{X}(\boldsymbol{\xi}'))\delta_h(\boldsymbol{x}'-\boldsymbol{X}(\boldsymbol{\xi}'))G^L(\boldsymbol{x};\boldsymbol{x}')\delta_h(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi}))\mathrm{d}\boldsymbol{\xi}'\mathrm{d}\boldsymbol{x}'d\boldsymbol{x} = -\varphi_{\Gamma}(\boldsymbol{X}(\boldsymbol{\xi})), \tag{1.29}$$

and $\varphi(\boldsymbol{x})$ may be obtain upon substitution f into Eq. (1.28). Note that since δ_h is continuous, the kernel in the integral equation Eq. (1.29) is continuous and has finite support, which makes the integral operator compact and with a formally unbounded inverse (Kress 2014). A direct consequence is that discretizations of this equation lead to inaccurate surface source terms.

Examples of smeared delta functions include

- A 2-point hat function:

$$\delta_h^{\text{hat}}(r) = \begin{cases} \frac{1}{\Delta x} - \frac{|r|}{\Delta x^2}, & |r| \leq \Delta x \\ 0, & |r| > \Delta x \end{cases} \tag{1.30}$$

- A 3-point function:

$$\delta_h^3(r) = \begin{cases} \frac{1}{3\Delta x}\left(1 + \sqrt{1 - 3\left(\frac{r}{\Delta x}\right)^2}\right), & |r| \leq \frac{\Delta x}{2} \\ \frac{1}{6\Delta x}\left(5 - \frac{3|r|}{\Delta x} - \sqrt{1 - 3\left(1 - \frac{|r|}{\Delta x}\right)^2}\right), & \frac{\Delta x}{2} \leq |r| \leq \frac{3\Delta x}{2} \\ 0, & |r| > \frac{3\Delta x}{2} \end{cases} \tag{1.31}$$

- A 4-point cosine function:

$$\delta_h^{\cos}(r) = \begin{cases} \frac{1}{4\Delta x}\left(1 + \cos\left(\frac{\pi r}{2\Delta x}\right)\right), & |r| \leq 2\Delta x \\ 0, & |r| > 2\Delta x \end{cases} \tag{1.32}$$

- A Gaussian function:

$$\delta_h^G(r) = \begin{cases} \sqrt{\frac{\pi}{36\Delta x^2}}\, \mathrm{e}^{\frac{-\pi^2 r^2}{36\Delta x^2}}, & |r| \leq 14\Delta x \\ 0, & |r| > 14\Delta x \end{cases} \tag{1.33}$$

Discretization of Eq. (1.25) yields

$$L\varphi = -Rf + b_L, \tag{1.34}$$

$$E\varphi = \varphi_\Gamma, \tag{1.35}$$

and combining Eq. (1.34) and Eq. 1.35 results in

$$EL^{-1}Rf = -\varphi_\Gamma + EL^{-1}b_L, \tag{1.36}$$

which is a discretization of the integral equation (1.29).

In Goza et al. (2016), Eq. (1.36) was solved numerically using a finite difference approximation. In the following, n_b and n_g are used to denote the number of points on the immersed body and the computational domain, respectively. In Fig. 1.2, it is apparent that the source term f does not converge with grid refinement, whereas the integrated source term F and the solution φ do converge at first order to their

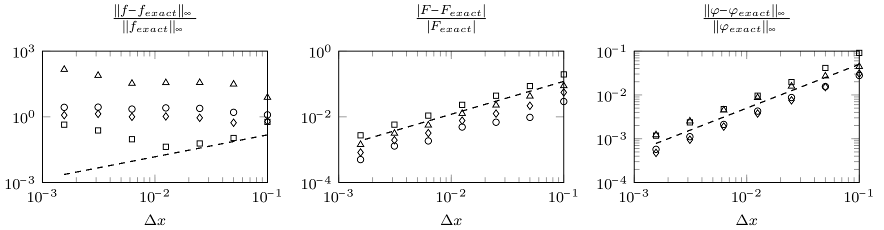


Fig. 1.2 Errors in $\tilde{f}$, F, and φ versus grid spacing (h) for the Poisson problem. ○: δ_h^{hat}, ◇: δ_h^3, △: $\delta_h^{\cos}$, □: δ_h^G, --: first-order convergence. Reprinted from Goza et al. (2016) with permission

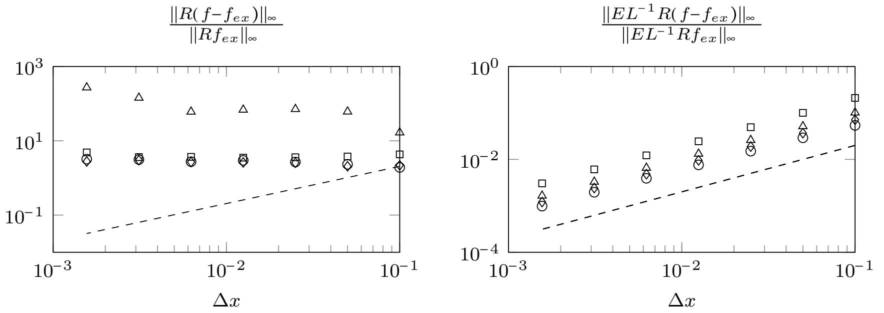


Fig. 1.3 Errors in Rf and $EL^{-1}Rf$ versus grid spacing (h) for the Poisson model problem. ○ :δ_h^{hat}, ◇: δ_h^3, △: $\delta_h^{\cos}$, □: δ_h^G, --: first-order convergence. Reprinted from Goza et al. (2016) with permission

exact solutions. Convergence of F is a consequence of solving Eq. (1.36). This is in contrast to other velocity-based IB methods, which only approximately enforce the boundary constraint. Such methods introduce inaccuracies in F (Uhlmann 2005; Huang and Sung 2009; Zhang and Zheng 2007), although Yang et al. (2009) proposed improvements.

From Fig. 1.3, it is also apparent that Rf does not converge to Rf_{ex} but $EL^{-1}Rf$ converges to $EL^{-1}Rf_{\text{ex}}$. Thus using the exact force f_{ex} to enforce the boundary condition, would not produce φ_Γ exactly, but rather converge to it at first order (see also Tornberg and Engquist 2004).

In Goza et al. (2016), the convergence behavior was studied further using a singular value decomposition (SVD) of $EL^{-1} = U\Sigma V^\dagger$. Using this decomposition, Rf_{ex} may be written as a projection onto the basis of vectors formed by V:

$$Rf_{\text{ex}} = \sum_{j=1}^{n_b} \alpha_j^{\text{ex}} v_j. \tag{1.37}$$

Similarly, $EL^{-1}Rf_{\text{ex}}$ may be expressed as

$$EL^{-1}Rf_{\mathrm{ex}} = \sum_{j=1}^{n_b} \alpha_j^{\mathrm{ex}} \sigma_j u_j \tag{1.38}$$

where $\sigma_1, \ldots, \sigma_{n_b}$ are the singular values, and the left (right) singular vectors are denoted by u_j (v_j) corresponding to σ_j. The coefficients are defined as $\alpha_j^{\mathrm{ex}} := (v_j^T Rf_{\mathrm{ex}})$.

Analogous expressions can be written for Rf by replacing f_{ex} with f in (1.37) and (1.38). It was shown in Goza et al. (2016) that the sum $\sum_{j=1}^{n_b} \alpha_j$ does not converge to $\sum_{j=1}^{n_b} \alpha_j^{\mathrm{ex}}$ under grid refinement, but converges when scaled by the σ_j. Since EL^{-1} is a discrete integral operator, the σ_j decay to small values (Hansen 1998) and the error is thus caused by high-index coefficients α_j corresponding to small σ_j.

Hence, smeared delta functions with rapidly decaying α_{ex} are favorable as spurious high-index coefficients can be filtered out effectively without loss of physical information. In contrast, delta functions with slow decay may lead to loss of physical information and thus inaccurate source terms due to inaccurate high-index coefficients.

An efficient filtering can be achieved by penalizing the spurious components of f. This can be done by pre-multiplying the source term with $\hat{E}R$ using a weighted interpolant $\hat{E} = EW$, which interpolates the smeared source term Rf onto the immersed body while preserving its integral value. The filtered source term is then $\hat{f} = \hat{E}Rf$.

In particular, W can be defined by a diagonal matrix with entries given by

$$W_{ii} = \begin{cases} 1/(R\mathbf{1})_i, & (R\mathbf{1})_i \neq 0 \\ 0, & \text{else,} \end{cases} \tag{1.39}$$

where $\mathbf{1} = [1, 1, \ldots, 1]^T \in \mathbb{R}^{n_g \times 1}$ and $(R\mathbf{1})_i$ is the ith entry in the vector $R\mathbf{1}$. Note that the weights are only nonzero within the support of the smeared delta function. The source term is redistributed by the filter $\hat{E}R$ by convolution with a kernel of smeared delta functions.

The rate of filtering of $\hat{E}R$ is proportional to the smoothness of the smeared delta function. This is a consequence of $\hat{E}R$ being an integral operator, for which the decay rate of its singular values is determined by the smoothness of its kernel (Hansen 1998). Applying this filtering technique yields more accurate source terms f as shown in Fig. 1.4. Indeed, the infinitely differentiable δ_h^G shows first-order convergence to f_{ex}, whereas the slow decay of the coefficients α_j hinders convergence for δ_h^{hat}, δ_h^3, and $\delta_h^{\cos}$. Finally, it is worth mentioning that filtering does not affect F by construction of $\hat{E}R$, and the solution φ is also unchanged since filtering is a post-processing step.

The extension of this filtering technique to the Navier–Stokes equation is straightforward. In particular, multiplication by the smeared delta and integration over the domain yields:

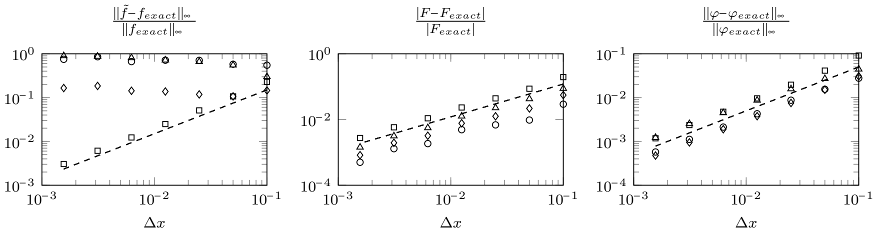


Fig. 1.4 Errors in $\hat{f}$, F, and φ versus grid spacing (h) for the Poisson problem. ○: δ_h^{hat}, ◇: δ_h^3, △: $\delta_h^{\cos}$, □: δ_h^G, --: first-order convergence. Reprinted from Goza et al. (2016) with permission

$$\begin{aligned}&\int_\Omega\int_\Gamma \mathbf{f}(\boldsymbol{X}(\boldsymbol{\xi}',t))\delta_h(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi},t))\delta_h(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi}',t))d\boldsymbol{\xi}'d\boldsymbol{x}\\&=\int_\Omega\left[\left(\frac{\partial}{\partial t}-\frac{1}{\text{Re}}\nabla^2\right)\mathbf{u}(\boldsymbol{x})+\mathbf{u}\cdot\nabla\mathbf{u}+\nabla p\right]\delta_h(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{\xi},t))d\boldsymbol{x}\end{aligned} \tag{1.40}$$

Analogous to the example above, the integral operator of Eq. 1.40 has an unbounded inverse because it contains a continuous kernel for any Δx. Hence, the same logic applies as above, and filtering is required to obtain accurate results. In the next section, this filtering approach is used for fully coupled fluid–structure interaction simulations.

1.2.4 Strongly Coupled Fluid–Structure Interaction

In this section, we present the extension of the IBPM to a strongly coupled fluid–structure interaction (FSI) solver for thin elastic structures (Goza and Colonius 2017).

In general, for FSI simulations, one can distinguish between monolithic and partitioned methods. In the monolithic approach, the fluid and structural equations are described with one system of equations using the same discretization scheme, which is solved by a single solver. By construction, consistent fluid–structure interface conditions are imposed in monolithic solvers. On the other hand, the partitioned approach uses individual solvers for the fluid and the structural equations, which are then coupled via appropriate boundary conditions to satisfy the solid–fluid interface conditions. This is a modular approach, which enjoys popularity for industrial as well as academic applications since separately optimized solvers for the fluid and the solid domain can be utilized. On the other hand, the main challenge of partitioned approaches is that the solid–fluid interface conditions are not implicitly satisfied. Hence, within the context of partitioned approaches, there is yet another distinction regarding the coupling methodology, namely between weakly and strongly coupled FSI schemes. While weakly coupled methods do not enforce the (nonlinear) fluid–

solid interface constraints at each time step, strongly coupled methods do converge to the monolithic equations using subiterative schemes.

On the one hand, this makes strongly coupled methods computationally more expensive. On the other hand, the staggered nature of weakly coupled schemes makes them susceptible to the so-called added-mass effect, where spurious energy is generated at the solid–fluid interface (Piperno and Farhat 2001). In particular for small solid–fluid density ratios, this can lead to fatal instabilities (Causin et al. 2005; Borazjani et al. 2008; Le Tallec 2001; Förster et al. 2007; Li and Favier 2017). For that reason we restrict ourselves to strongly coupled methods.

To impose the nonlinear interface constraint, most strongly coupled methods require the solution of a large nonlinear system. The block Gauss–Seidel method is a commonly used iterative procedure for solving the nonlinear system of equations, though it requires a relaxation parameter (often chosen heuristically) and typically converges slowly for small structural densities. Another common nonlinear solver is the Newton–Raphson method, which removes the relaxation parameter and typically converges rapidly even for small density ratios. However, this approach involves linear systems with large Jacobian matrices that cannot be solved directly, necessitating the use of large matrix-vector products in the context of some iterative solution process (Degroote et al. 2009; Mori and Peskin 2008; Hou et al. 2012). Among others, these strategies are reviewed in Sotiropoulos and Yang (2014).

In the following, we focus on thin elastic structures and solve the nonlinear algebraic system using the Newton–Raphson method. However, we avoid the need to solve a large linear system (without introducing any additional approximation into the solution process) by employing a block-LU factorization of the linearized system (Goza and Colonius 2017). The fluid part of the system is treated with the IBPM as outlined above.

For the coupled fluid–structure problem, the governing equations in Eq. (1.1) are extended by the structural equation as

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \boldsymbol{u} + \int\limits_{\Gamma(t)} \boldsymbol{f}_\Gamma(\boldsymbol{X}(\boldsymbol{\xi}, t), t) \delta(\boldsymbol{X}(\boldsymbol{\xi}, t) - \boldsymbol{x}) \mathrm{d}\boldsymbol{\xi}, \tag{1.41a}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1.41b}$$

$$\frac{\rho_s}{\rho_f} \frac{\partial^2 \boldsymbol{X}(\boldsymbol{\xi}, t)}{\partial t^2} = \frac{1}{\rho_f U_\infty^2} \nabla \cdot \boldsymbol{\sigma} + \mathbf{g}(\boldsymbol{X}) - \boldsymbol{f}_\Gamma(\boldsymbol{X}), \tag{1.41c}$$

$$\int\limits_\Omega \boldsymbol{u}(\boldsymbol{x}) \delta(\boldsymbol{x} - \boldsymbol{X}(\boldsymbol{\xi}, t)) \mathrm{d}\boldsymbol{x} = u_\Gamma(\boldsymbol{\xi}, t) = \frac{\partial \boldsymbol{X}(\boldsymbol{\xi}, t)}{\partial t}, \tag{1.41d}$$

where the solid and fluid density are denoted by ρ_s and ρ_f, respectively. The Cauchy stress is denoted by $\boldsymbol{\sigma}$, $\mathbf{g}$ is a body force, and the characteristic velocity is U_∞. The time derivative in the above is understood to be Lagrangian and the Cauchy stress is related to the second Piola–Kirchhoff stress $\boldsymbol{S}_s$ by

$$\boldsymbol{S}_s = J\boldsymbol{F}^{-1}\boldsymbol{\sigma}_s\boldsymbol{F}^{-T}, \tag{1.42}$$

where $J = \det(\boldsymbol{F})$ and $\boldsymbol{F}$ denotes the deformation gradient

$$\boldsymbol{F} = \boldsymbol{I} + \nabla\boldsymbol{u}_s, \tag{1.43}$$

and $\boldsymbol{u}_s$ is the displacement field of the solid. The second Piola–Kirchhoff stress is here defined as

$$\boldsymbol{S}_s = \mathbb{C} : \boldsymbol{E} \tag{1.44}$$

where

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{F}^T\boldsymbol{F} - \boldsymbol{I}) = \frac{1}{2}(\nabla\boldsymbol{u}_s + \nabla\boldsymbol{u}_s^T + \nabla\boldsymbol{u}_s^T\nabla\boldsymbol{u}_s) \tag{1.45}$$

is the Green–Lagrangian strain tensor and the stiffness tensor $\mathbb{C}$ is related to Young's modulus E_s, the bulk shear modulus, and Poisson's ratio ν_s.

The governing equations for the fluid region are discretized as above, and the solid equations can be discretized using a standard finite element procedure for thin elastic beams in a co-rotational formulation (see Goza and Colonius 2017; De Borst et al. 2012 for details), yielding the semi-discrete equation for the solid:

$$M_s\ddot{X} + K_s(X) = F_s(\mathbf{g} + W_s(X)f), \tag{1.46}$$

where the mass matrix M_s, the stiffness matrix K_s, and load F_s are given by

$$M_s = \frac{\rho_s}{\rho_f}\sum_j^{N_{\mathrm{el}}}\int_{\Gamma_j^0} B^\dagger B \mathrm{d}\boldsymbol{X}^0, \quad K_s(X) = \frac{1}{\rho_f U_\infty^2}\sum_j^{N_{\mathrm{el}}}\int_{\Gamma_j^0} B_E^\dagger \sigma_s \mathrm{d}\boldsymbol{X}^0 \tag{1.47}$$

$$F_s = \sum_j^{N_{\mathrm{el}}}\int_{\Gamma_j^0} B^\dagger B \mathrm{d}\boldsymbol{X}^0 = \frac{\rho_f}{\rho_s}M_s. \tag{1.48}$$

Here, the jth element of Γ in the undeformed domain is indicated by Γ_j^0 and the shape function matrix as well as their derivatives are given by B and B_E, respectively. This formulation is expressed in the co-rotational frame and therefore accounts for geometrical nonlinearity and assumes small strains (large strains can be incorporated into K_s without affecting the algorithm). For further details, we refer to standard finite element textbooks (De Borst et al. 2012; Bathe 1996).

Using the discretizations above, the fully coupled FSI equations can be written as a first-order system of differential-algebraic equation as

$$C^{\dagger}C\dot{s} = -N(u) + C^{\dagger}LCs - C^{\dagger}E^{\dagger}\tilde{f}, \tag{1.49}$$

$$M_s\dot{u}_{\Gamma} = -K_s(X) + F_s(\mathrm{g} + W_f(X)\tilde{f}), \tag{1.50}$$

$$\dot{X} = u_{\Gamma}, \tag{1.51}$$

$$ECs - u_{\gamma} = 0. \tag{1.52}$$

An Adams–Bashforth and Crank–Nicolson time-marching scheme is applied for the nonlinear term and the diffusive term, respectively, for Eq. (1.49) and an implicit Newmark scheme is used for Eqs. (1.50)–(1.51). Equation (1.52) is evaluated at the current time step. This yields the following nonlinear system of algebraic equations:

$$C^{\dagger}ACs^{n+1} + C^{\dagger}E^{\dagger}\tilde{f}^{n+1} = r_f^n, \tag{1.53}$$

$$\frac{4}{\Delta t^2}M_sX^{n+1} + K_s(X^{n+1}) - F_sW^{n+1}\tilde{f}^{n+1} = r_{u_{\Gamma}}^n, \tag{1.54}$$

$$\frac{2}{\Delta t}X^{n+1} - u_{\Gamma}^{n+1} = r_X^n, \tag{1.55}$$

$$ECs^{n+1} - u_{\Gamma}^{n+1} = 0, \tag{1.56}$$

where $A = \frac{1}{\Delta t}I - \frac{1}{2}L$, $r_n^f = (\frac{1}{\Delta t}C^TC + \frac{1}{2}C^TLC)s^n + \frac{3}{2}C^TN(Cs^n) - \frac{1}{2}C^T N(Cs^{n-1})$, $r_{u_{\Gamma}}^n = M(\frac{4}{\Delta t^2}X^n + \frac{4}{\Delta t}u_{\Gamma} + \dot{u}_{\Gamma}^n) + Qg$, and $r_n^X = u_{\Gamma} + \frac{2}{\Delta t}X^n$.

To solve the nonlinear system, an iterative procedure is applied and the solution at time step n is used to initialize the iterative procedure at $k = 0$. During the kth iteration, the variables are updated as $X_{k+1}^{n+1} = X_{k+1}^{n+1} + \Delta X$ and $u_{\Gamma,k+1}^{n+1} = u_{\Gamma,k}^{n+1} + \Delta u_{\Gamma}$, which yields the following system:

$$\begin{bmatrix} C^TAC & 0 & 0 & C^TE^{\dagger} \\ 0 & 0 & \frac{4}{\Delta t^2}M + K_k & -F_sW_k^{n+1} \\ 0 & -I & \frac{2}{\Delta t}I & 0 \\ EC & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} s^{n+1} \\ \Delta u_{\Gamma} \\ \Delta X \\ \tilde{f}_{k+1}^{n+1} \end{bmatrix}$$

$$= \begin{bmatrix} r_n^f + O(\Delta t) \\ r_{u_{\Gamma}}^n - \frac{4}{\Delta t^2}MX_k^{n+1} - K_s(X_k^{n+1}) + O(\Delta t) \\ r_n^X - \frac{2}{\Delta t}X_k^{n+1} + u_{\Gamma,k}^{n+1} \\ u_{\Gamma,k}^{n+1} + O(\Delta t) \end{bmatrix} := \begin{bmatrix} r_n^f \\ r_{u_{\Gamma},k} \\ r_{X,k} \\ r_{c,k} \end{bmatrix}, \tag{1.57}$$

where $K_k = dK_s/dX|_{X=X_k^{n+1}}$. For flags, the stiffness matrix has well known analytical expressions (De Borst et al. 2012; Bathe 1996). The linear system Eq.(1.57) can be factored using a block-LU decomposition, which yields

$$s^* = (C^T AC)^{-1} r_n^f \tag{1.58}$$

$$\begin{bmatrix} B_k^{n+1} & I \\ -\frac{2}{\Delta t} F_s W_k^{n+1} & \hat{K}_k \end{bmatrix} \begin{bmatrix} \tilde{f}_{k+1}^{n+1} \\ \Delta u_\Gamma \end{bmatrix} = \begin{bmatrix} ECs^* - r_{c,k} \\ \frac{2}{\Delta t} r_{u_\Gamma,k} - r_{X,k} \end{bmatrix} \tag{1.59}$$

$$\Delta X = \frac{\Delta t}{2}(\Delta u_\Gamma + r_{X,k}) \tag{1.60}$$

$$s^{n+1} = s^* - (C^T AC)^{-1} C^T E^\dagger \tilde{f}^{n+1}, \tag{1.61}$$

where $\hat{K}_k := \frac{4}{\Delta t^2} M_s + K_k$ and $B_k^{n+1} := EC(C^\dagger AC)^{-1} C^T E^\dagger$. The LU-factorized equations (1.58)–(1.61) are analogous to the previous factorizations but now include the fully coupled FSI scheme. Note that Eq. (1.58) does not depend on variables at time $n + 1$ and thus must only be solved once per time step. Moreover, s^{n+1} is only updated after Eqs. (1.59)–(1.60) have converged in the iterative process. Hence, the iterations are restricted to Eqs. (1.59)–(1.60) which have dimensions of the order of number of body points rather then the entire flow domain.

The Poisson-like problems arising from Eq. (1.58), Eq. (1.61) and from each matrix-vector multiply with $B_{(k)}^{n+1}$ can be solved efficiently with fast Fourier transforms (FFT). In Goza and Colonius (2017), it was also argued that since the floating point operations of the FFT scale with the number points in the flow domain, it may be favorable to compute and store $(\hat{K}_k)^{-1}$. In that case, an analytical block Gaussian elimination of Eq. (1.59) may be performed to arrive at

$$\left(B_k^{n+1} + \frac{2}{\Delta t} (\hat{K}_k)^{-1} F_s W_k^{n+1} \right) \tilde{f}_{(k+1)}^{n+1} = ECs^* - r_{c,k} - \frac{2}{\Delta t} (\hat{K}_k)^{-1} r^{u_\Gamma,k} + r_{X,k} \tag{1.62}$$

$$\Delta u_\Gamma = \frac{2}{\Delta t} (\hat{K}_k)^{-1} (r_{u_\Gamma,k} + F_s W_k^{n+1} \tilde{f}_k^{n+1}) - r_{X,k} \tag{1.63}$$

Equation (1.62) can then be solved with a BICGSTAB scheme, which typically converges in a few iterations. In Goza and Colonius (2017), it was also shown that this iteration procedure typically converges in a few iterations and does not rely on heuristic relaxation parameters as for Gauss–Seidel-based approaches.

1.2.5 Example: The Inverted Flag Problem

Flow past a flag that is clamped at its leading edge is a canonical problem (Taneda 1968) and serves as an important benchmark problem for the development of numerical schemes (see Shelley and Zhang 2011 for a review). By contrast, when the flag is inverted, i.e., the trailing edge is clamped, only a few studies can be found in the literature (Kim et al. 2013; Gurugubelli and Jaiman 2015; Ryu et al. 2015). This configuration, however, is of interest due to its rich dynamical behavior, which includes small-deflection flapping, large-amplitude flapping, and chaotic flapping. From a numerical perspective, this setup is particular challenging as these regimes span a large range of solid–fluid density ratios and exhibit large deformations that require a strongly coupled FSI solver. In Goza et al. (2018), this setup was investigated using the approach described in the previous sections. In addition to validating the numerics, these studies show that the IB method can be readily adapted for use in stability and bifurcation analysis. A steady-state solver employing a Newton–Raphson iteration was used to determine (potentially unstable) equilibria of the full fluid–structure system, and linearizations of the discretized equations lead to large, sparse systems of algebraic equations whose stability properties were efficiently determined using Arnoldi methods.

For the inverted flag, there are three independent non-dimensional parameters that govern the system. These are the Reynolds number $\mathrm{Re} = \frac{U_\infty L}{\nu_s}$, the mass ratio $M_\rho = \frac{\rho_s h}{\rho_f L}$, and the bending stiffness $K_B = \frac{D}{\rho_f U_\infty^2 L^3}$, where ρ_f (ρ_s) is the fluid (structure) density, U_∞ is the free-stream velocity, L is the flag length, h is the flag thickness, and the flexural rigidity is given by $D = Eh^3/(12(1-\nu^2))$ with Young's modulus E. In what follows, we consider the case of $\mathrm{Re} = 200$ and $M_\rho = 0.05$ and present the effect of decreasing the flag's stiffness K_B. More configurations and thorough analysis can be found in Goza et al. (2018).

With decreasing K_B, the flag undergoes a transition from the undeformed equilibrium (I) regime to the deformed equilibrium (II) through a divergence instability. With decreasing stiffness, the deformed equilibrium is associated with an increasingly large tip deflection and transitions from stable to unstable regimes. In particular, the small-deflection flapping regime (III) is reached by a supercritical Hopf bifurcation of the deformed equilibrium state. Decreasing the flag's stiffness further leads to large-amplitude flapping (IV, see also Fig. 1.5), which can be associated to classical vortex-induced vibration for the small density ratios as presented here. This is in contrast to heavier flags, for which large-amplitude flapping is not a classical vortex-induced vibration. For a stiffness of $K_B = 0.32$, snapshots of one flapping period are shown in Fig. 1.5 by means of vorticity contours.

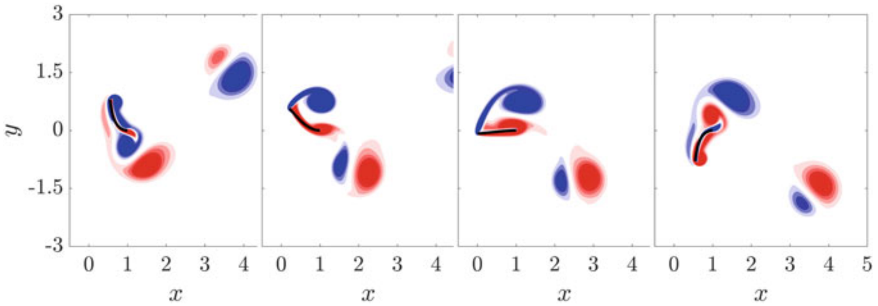


Fig. 1.5 Vorticity contours at four snapshots of a flapping period of a flag in large-amplitude flapping for $M_\rho = 0.05$. The Reynolds number and flag flexibility were chosen as $\mathrm{Re} = 200$, $K_B = 0.32$. Contours are in 18 increments from -5 to 5. Reprinted from Goza et al. (2018) with permission

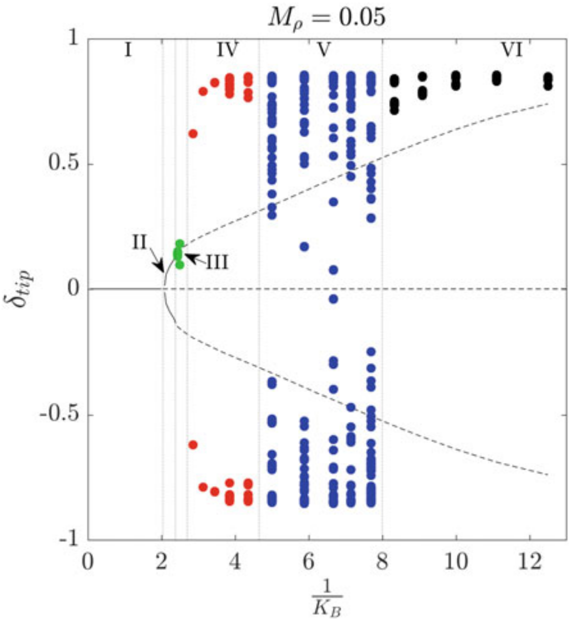


Fig. 1.6 Bifurcation diagram of inverted flag dynamics at Re = 200, showing tip deflection δ_{tip} as a function of inverse stiffness $(1/K_B)$. Regimes are denoted as I: undeformed equilibrium, II: deformed equilibrium, III: small-deflection deformed flapping, IV: large-amplitude flapping, V: chaotic flapping, VI: deflected mode. Reprinted from Goza et al. (2018) with permission

With decreasing stiffness, the system transitions to the chaotic flapping regime (V), and finally to the deflected mode regime (VI), where oscillations are primarily driven by vortex shedding. The chaotic regime is characterized by a strange attractor that alternatively samples regimes IV and VI.

These phenomena are summarized in the bifurcation diagram in Fig. 1.6 as obtained through nonlinear simulations. The simulations were started with an undeflected flag and an impulsively started flow at free-stream velocity U_∞. During the initial transient, a small body force was used to trigger any instabilities in the system. All simulations were run for at least 15 flapping cycles except for the chaotic state which requires 55 cycles. The first several cycles were neglected to avoid accounting for initial conditions. In Fig. 1.6, a set of markers at a given stiffness represents the tip deflection values δ_{tip} from a single nonlinear simulation when the flag changes direction. That is, the markers correspond to the zero-tip-velocity Poincaré sections of a velocity–displacement phase portrait of the leading edge. In addition, solid and dashed lines represent stable and unstable equilibria, respectively.

1.3 Fast Lattice Green's Function for External Flows

In this section, the immersed boundary lattice Green's function (IBLGF) method as proposed in Liska and Colonius (2017) is presented. The IBLGF is based on the unbounded domain lattice Green's function (LGF) flow solver (Liska and Colonius

2016) and the distributed Lagrange multiplier method to impose the no-slip boundary condition. The governing Navier–Stokes equations are spatially discretized on an unbounded staggered Cartesian grid, which retains crucial conservative, commutative, orthogonality, and symmetry properties of standard staggered Cartesian discretizations of infinite domains.

The advantage, however, is the use of the lattice Green's function technique, which implicitly satisfies the natural free-space boundary condition and allows block-adaptive grids to restrict the computation to a finite region (set of grid points) where the vorticity is nonzero (exceeds a small threshold). This is in contrast to common IB methods, which employ spatially truncated domains with approximate free-space boundaries. These approximations introduce blockage errors, which affect accuracy and may even change the dynamics of the flow (Tsynkov 1998; Colonius 2004; Pradeep and Hussain 2004). Thus, large computational domains in combination with stretched grids (Taira and Colonius 2007; Yun et al. 2006; Wang and Zhang 2011), local refinement (Roma et al. 1999; Griffith et al. 2007), and far-field approximations (Colonius and Taira 2008) are required to limit influence of the approximate free-space boundary condition (see also Sect. 1.2). In contrast, due to the natural free-space boundary inherent to IBLGF, an adaptive domain snugly conforming to regions of non-negligible vorticity and free of free-space boundary errors may be used for IBLGF.

By using a viscous integrating factor half-explicit Runge–Kutta scheme (IF-HERKS) in combination with an approximation-free nested projection technique and exploiting the aforementioned algebraic properties of the discrete operators, the projection steps reduce to simple discrete elliptic problems. These can in turn be solved efficiently using parallel lattice Green's function fast multipole methods (LGF-FMM) (Liska and Colonius 2014).

The operators satisfy the following topological and mimetic properties:

$$\text{Symmetry} \qquad \bar{D} = -G^\dagger, \bar{G} = -D^\dagger, \quad \bar{C} = C^\dagger \tag{1.64}$$

$$\text{Orthogonality} \quad \mathrm{Null}(C) = \mathrm{Im}(G), \mathrm{Null}(D) = \mathrm{Im}(G) \tag{1.65}$$

$$\text{Mimetic} \qquad L_C = -G^\dagger G, L_{\mathcal{F}} = -GG^\dagger, L_{\mathcal{E}} = -D^\dagger D, L_{\mathcal{V}} = -DD^\dagger \tag{1.66}$$

$$\text{Commutativity} \quad L_{\mathcal{F}} G = G L_C \tag{1.67}$$

1.3.1 Time Integration

In incompressible flow solvers, it is typical to use split time-stepping schemes where the viscous terms are advanced with an implicit method (alleviating any viscous time step constraint), whereas the advective terms are advanced with an explicit method which, despite introducing a CFL constraint on the time step, avoids iterative solution of nonlinear equations. A variety of specialized splitting methods have been employed, typically achieving second-order temporal accuracy. These schemes must also be cognizant of the kinematic constraints, which in general include enforcing a divergence-free velocity field but in the case of IB methods, also include enforcing the no-slip condition. In other words, the spatially discretized incompressible Navier–Stokes equations comprise a discrete algebraic equation of index 2 (DAE-$i2$). The CN-AB2 scheme used in the previous sections is a particularly common choice for tackling such systems.

In the LGF approach, there is an opportunity to improve the time-marching scheme because the viscous terms can be integrated exactly using an integrating factor, which in turn can be done efficiently using an LGF. Once this is done, the time stepper for the remaining DAE system need no longer be split, meaning that explicit Runge–Kutta methods for DAE systems suffice, which in turn enables a wide variety of tailored schemes (low dissipation, low memory, etc.) to be employed. In particular, a family of half-explicit Runge–Kutta (HERK) methods are derived in Liska and Colonius (2016). Note that the "half-explicit" terminology refers to the solution of ODEs and algebraic constraints—not to any viscous/inviscid splitting.

The discrete integrating factor $E_Q(t)$ is the solution of the discrete heat equation $\frac{\mathrm{d}h}{\mathrm{d}t} = \kappa L_Q h$. Hence, for a given u at time τ and the integrating factor $H_Q = E_Q\left(\frac{t-\tau}{(\Delta x)^2\mathrm{Re}}\right)$, Eq.(1.2) for $t > \tau$ is given by

$$\frac{\mathrm{d}v}{\mathrm{d}t} + [H_{\mathcal{F}}(t)]\tilde{N}([H_{\mathcal{F}}^{-1}(t)]v, t) = -Gb - [H_{\mathcal{F}}(t)][E(t)]^{\dagger}\tilde{f}, \tag{1.68a}$$

$$G^{\dagger}v = 0, \tag{1.68b}$$

$$[E][H_{\mathcal{F}}^{-1}(t)]v = u, \tag{1.68c}$$

where $v = [H_{\mathcal{F}}(t)]u$ and $b = [H_C(t)]p$. The above system constitutes a DAE-$i2$ that can be solved efficiently using an s-stage HERK scheme. The s-stages of the HERK are defined using the superscript i and time $t_k = k\Delta t$ with the time step Δt in indicated by subscript k. We now group the Lagrange multipliers, the right-hand side and the operators together such that

$$\lambda_k^i = \begin{bmatrix} p_k^i \\ \tilde{f}_k^i \end{bmatrix}, \quad \zeta_k^i = \begin{bmatrix} 0 \\ u(t_k^i) \end{bmatrix}, \quad Q^i = \left[G\ [E(t_k^i)]^{\dagger}\right], \forall i \in [1, s] \tag{1.69}$$

and we introduce the following:

$$u_k^i(\boldsymbol{n}) = \left[E_{\mathcal{F}}\left(\frac{-\bar{c}\Delta t}{(\Delta x)^2\mathrm{Re}}\right)\right] v_k^i(\boldsymbol{n}), \quad p_k^i(\boldsymbol{n}) = \left[E_{\mathcal{F}}\left(\frac{-\bar{c}\Delta t}{(\Delta x)^2\mathrm{Re}}\right)\right] b_k^i(\boldsymbol{n}). \tag{1.70}$$

The kth time step of the IF-HERK(u_k, t_k) scheme with the shifted coefficients $\tilde{a}_{i,j}$ and the shifted nodes $\tilde{c}_i$ (Liska and Colonius 2016; Hairer et al. 2006) can be summarized as follows:

1. Initialize: Set $u_k^0 = u_k$, $t_k^0 = t_k$
2. Multi-stage: for $i = 1, 2, \cdots, s$ solve the linear system:

$$\begin{bmatrix} (H^i_{\mathcal{F}})^{-1} & Q^{i-1}_k \\ (Q^i_k)^\dagger & 0 \end{bmatrix} \begin{bmatrix} u^i_k \\ \lambda^i_k \end{bmatrix} = \begin{bmatrix} r^i_k \\ \zeta^i_k, \end{bmatrix} \tag{1.71}$$

where

$$H^i_{\mathcal{F}} = E_{\mathcal{F}}\left(\frac{(\tilde{c}_i - \tilde{c}_{i-1})\Delta t}{(\Delta x)^2 \mathrm{Re}}\right), \quad r^i_k = h^i_k + \Delta t \sum_{j=1}^{i-1} \tilde{a}_{i,j} w^{i,j}_k + g^i_k \tag{1.72}$$

$$g^i_k = -a_{i,i}\Delta t \tilde{N}(u^{i-1}_k, t^{i-1}_k) \quad t^i_k = t_k + +\tilde{c}_i \Delta t. \tag{1.73}$$

The variables h^i_k and $w^{ki,j}_k$ are recursively computed for $i > 1$ and $j > i$ using

$$h^i_k = H^{i-1}_{\mathcal{F}} h^{i-1}_k, \quad h^1_k = 0 \tag{1.74}$$

$$w^{i,j}_k = H^{i-1}_{\mathcal{F}} w^{i-1,j}_k, \quad w^{i,i}_k = (\tilde{a}_{i,i}\Delta t)^{-1}\left(Q^{i-1}_k \hat{\lambda}^i_k\right) \tag{1.75}$$

3. Finalize: Set $u_{k+1} = u^s_k$, $\lambda_{k+1} = (\tilde{a}_{s,s}\Delta t)^{-1}\hat{\lambda}^s_k$ and $t_{k+1} = t^s_k$.

1.3.2 Linear Solver

The solution of Eq. (1.71) dominates the computational costs and is efficiently solved using the exact projection technique. Note that in contrast to 2D discrete nullspace (discrete streamfunction) methods (see, Sect. 1.2), the following formulation does not express the discrete velocity–pressure equations as discrete streamfunction–vorticity equations. In three dimensions, the resulting discrete Poisson problems are scalar problems in the case of the velocity–pressure formulation but vector problems in the case of streamfunction–vorticity formulation. Equation (1.71) can be rewritten in terms of p^i_k and $\tilde{f}^i_k$

$$M^i_k \begin{bmatrix} u^i_k \\ \hat{p}^i_k \\ \tilde{f}^i_k \end{bmatrix} = \begin{bmatrix} (H^i_{\mathcal{F}})^{-1} & G & (E^{i-1}_k)^\dagger \\ G^\dagger & 0 & 0 \\ E^i_k & 0 & 0 \end{bmatrix} \begin{bmatrix} u^i_k \\ \hat{p}^i_k \\ \tilde{f}^i_k \end{bmatrix} = \begin{bmatrix} r^i_k \\ 0 \\ u^i_k \end{bmatrix}, \tag{1.76}$$

where $\hat{p}^i_k / p^i_k = \hat{f}^i_k / \tilde{f}^i_k = \tilde{a}_{s,s}$, $u^i_k = u(t^i_k)$, and $E^i_k = E(t^i_k)$. Note that M^i_k is in general not symmetric and cannot be symmetrized as the image of the regularization operator and the interpolation operator are different. The asymmetry is inherent to HERK integration of DAE systems of index 2 with time-dependent constraint operators (Hairer et al. 2006; Brasey and Hairer 1993). Note, however, M^i_k retains symmetry for flow past non-moving rigid bodies. The nested projection technique for Eq. (1.76), obtained from an operator–block-LU decomposition of M^i_k can be written in projection-like form

$$\text{Solve for intermediate velocity:} \quad (H^i_{\mathcal{F}})^{-1} u^* = r^i_k \tag{1.77a}$$

$$\text{Solve for intermediate pressure:} \quad G^\dagger H_{\mathcal{F}} G d^* = G^\dagger u^* \tag{1.77b}$$

$$\text{Solve for intermediate IB forces:} \quad S^i_k f^* = E^i_k [u^* - H^i_{\mathcal{F}} G d^*] - u^i_k \tag{1.77c}$$

$$\text{Update forces:} \quad \hat{f}^i_k = f^* \tag{1.77d}$$

$$\text{Correct pressure:} \quad \hat{p}^i_k = p^* - (G^\dagger H^i_{\mathcal{F}} G)^{-1} G^\dagger H^i_{\mathcal{F}} (E^{i-1}_k)^\dagger \hat{f}^i_k \tag{1.77e}$$

$$\text{Correct velocity:} \quad u^i_k = u^* - H^i_{\mathcal{F}} [G \hat{p}^i_k + (E^{i-1}_k)^\dagger \hat{f}^i_k] \tag{1.77f}$$

where the force Schur complement of the LU decomposition of Eq. (1.76) S^i_k is given by

$$S^i_k = E^i_k H^i_{\mathcal{F}} [\mathrm{I} - G(G^\dagger H^i_{\mathcal{F}} G)^{-1} G^\dagger H^i_{\mathcal{F}}] (E^{i-1}_k)^\dagger, \tag{1.78}$$

where the identity operator is indicated by I. By exploiting the mimetic, orthogonality and commutativity properties, the above can be rewritten in a computationally more convenient way such as

$$L_C p^* = -G^\dagger r^i_k \tag{1.79a}$$

$$S^i_k \hat{f}^i_k = E^i_k H^i_C [r^i_k - G^\dagger p^*] - u^i_k \tag{1.79b}$$

$$\hat{p}^i_k = p^* + L_C^{-1} G^\dagger (E^{i-1}_k)^\dagger \hat{f}^i_k \tag{1.79c}$$

$$u^i_k = H^i_{\mathcal{F}} [r^i_k - G \hat{p}^i_k - (E^{i-1}_k)^\dagger \hat{f}^i_k], \tag{1.79d}$$

where the Schur complement simplifies to

$$S^i_k = E^i_k [H^i_{\mathcal{F}} + G (H^i_C)^{-1} L_C G^\dagger] (E^{i-1}_k)^\dagger. \tag{1.80}$$

Note that with the exception of $\hat{f}^i_k$, every term can efficiently be computed with either the point-operator representation of discrete operators or the Lattice Green's function fast multipole method (LGF-FMM), which will be outlined in some detail in the next section.

With regards to the solving the force Schur complement (1.79b), there exits either the possibility to use dense linear algebra techniques or iterative methods. In case of iterative methods, the conjugate gradient is a suitable candidate for a symmetric Schur complement, implying rigid and stationary immersed boundaries or more generally, for moving or deforming geometries, Krylov solvers such as GMRES or BiCGSTAB. Note that for the rigid and non-moving case, the dense linear algebra route is more convenient since the construction of S^i_k needs to be done only once and a Cholesky decomposition can be stored and used to evaluate the forces.

1.3.3 Lattice Green's Function Method

In this section, we focus on how to efficiently solve elliptic difference equations with the lattice Green's function method (Liska and Colonius 2014) in unbounded domains as they occur in the IBLGF method. The solution may be obtained by convolution of the fundamental solution of the discrete operator (lattice Green's function) with the source terms of the equation. The LGF can be derived from Fourier integrals and approximated through its through its asymptotic expansion. A significant advantage of the FLGF method is that the formally unbounded meshes may be truncated such that only regions with non-negligible source are retained in the computational domain, yielding an adaptive block-structured mesh as shown exemplary in Fig. 1.9.

In what follows, the three-dimensional Poisson equation will serve as a model equation, which is defined as:

$$[\Delta\varphi](\boldsymbol{x}) = h(\boldsymbol{x}), \quad \text{supp}(h) \subseteq \Omega, \tag{1.81}$$

where $\boldsymbol{x} \in \mathbb{R}^3$ and Ω denotes a bounded domain in $\mathbb{R}^3$. The target field φ can be obtained by convolution of the fundamental solution of the Laplace operator $\mathrm{G}^L(\boldsymbol{x}) = -1/(4\pi|\boldsymbol{x}|)$ with the source field $h(\boldsymbol{x})$ such that

$$\varphi(\boldsymbol{x}) = [\mathrm{G}^L * h](\boldsymbol{x}) = \int\limits_{\Omega} \mathrm{G}^L(\boldsymbol{x} - \boldsymbol{\xi})h(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}. \tag{1.82}$$

In the discrete setting, Eq. (1.81) can be expressed as

$$[L_Q\varphi](\boldsymbol{x}_i) = h(\boldsymbol{x}_i), \quad \text{supp}(h) \subseteq \Omega_h, \tag{1.83}$$

where $\varphi, h \in \mathbb{R}^Q$, $\boldsymbol{x}_i \in \mathbb{Z}^3$, and Ω_h is a bounded domain in $\mathbb{Z}^3$. The target can then be obtained by discrete convolution such that

$$\varphi(\boldsymbol{x}_i) = [G^L * h](\boldsymbol{x}_i) = \sum_{\boldsymbol{x}_j \in \Omega_h} G^L(\boldsymbol{x}_i - \boldsymbol{x}_j)h(\boldsymbol{x}_i), \tag{1.84}$$

where G^L indicates the LGF of the discrete 7-pt Laplacian. By diagonalizing the Laplace operator L_Q in Fourier space, an expression for $G^L(\boldsymbol{x}_i)$ can be derived (see, e.g., Delves and Joyce 2001; Glasser and Zucker 1977), yielding

$$G^L(\boldsymbol{x}_i) = \frac{1}{8\pi^3} \int\limits_{[-\pi,\pi]^3} \frac{\exp(-i\boldsymbol{x}_i\boldsymbol{\xi})}{2\cos(\xi_1) + 2\cos(\xi_2) + 2\cos(\xi_3) - 6}\mathrm{d}\boldsymbol{\xi}. \tag{1.85}$$

In addition, Eq.(1.85) can equivalently be written as a one dimensional, semi-infinite integral as

$$G^L(\boldsymbol{x}_i) = -\int_0^\infty \exp(-6t) I_{x_1}(2t) I_{x_2}(2t) I_{x_3}(2t) \mathrm{d}t, \tag{1.86}$$

where $I_k(t)$ is the modified Bessel function of first kind and order k. Note that Eq.(1.86) can be evaluated using an adaptive Gauss–Kronrod quadrature or alike in a straightforward manner, but it is typically more efficient to evaluate the Green's function through its asymptotic expansion in the far-field, i.e., large $|\boldsymbol{x}_i|$. In particular, the target field φ can be written as

$$\varphi(\boldsymbol{x}_i) = \varphi^{\text{near}}(\boldsymbol{x}_i) + \varphi^{\text{far}}(\boldsymbol{x}_i) + \varepsilon(\boldsymbol{x}_i), \tag{1.87}$$

where

$$\varphi^{\text{near}}(\boldsymbol{x}_i) = \sum_{\boldsymbol{x}_j \in \Omega_h^{\text{near}}(\boldsymbol{x}_i)} G^L(\boldsymbol{x}_i - \boldsymbol{x}_j) h(\boldsymbol{x}_j) \tag{1.88}$$

$$\varphi^{\text{far}}(\boldsymbol{x}_i) = \sum_{\boldsymbol{x}_j \in \Omega_h \setminus \Omega_h^{\text{near}}(\boldsymbol{x}_i)} A_G^q(\boldsymbol{x}_i - \boldsymbol{x}_j) h(\boldsymbol{x}_j), \tag{1.89}$$

and Ω_h^{near}, $\varepsilon(\boldsymbol{x}_i)$ are the near field and the error due to approximating $G^L(\boldsymbol{x}_i)$ with $A_G^q(\boldsymbol{x}_i)$ in the far-field, respectively. The q-term asymptotic expansion of $G^L(\boldsymbol{x}_i)$ is defined such that $G^L(\boldsymbol{x}_i) = A_G^q(\boldsymbol{x}_i) + O(|\boldsymbol{x}_i|^{-2q-1})$ and for $q = 2$ it reads

$$A_G^2(\boldsymbol{x}) = -\frac{1}{4\pi|\boldsymbol{x}|} - \frac{x_1^4 + x_2^4 + x_3^4 - 3x_1^2x_2^2 - 3x_1^2x_3^2 - 3x_2^2x_3^2}{16\pi|\boldsymbol{x}|^7}. \tag{1.90}$$

In practice, the results from direct integration of Eq.(1.86) are tabulated for the near-field ($|\boldsymbol{x}_i| \leq 100$), and the asymptotic expansion with $q = 3$ and $q = 2$ can be used in the far-fields for $100 < |\boldsymbol{x}_i| \leq 600$ and $|\boldsymbol{x}_i| > 600$, respectively. This ensures an error bound of the asymptotic expansion compared to the direct integration of $|\varepsilon| < 10^{-12}$.

1.3.3.1 Fast Convolutions

The direct evaluation of Eq.(1.84) requires $O(N^2)$ amount of work for N degrees of freedom and is therefore prohibitive for large computational domains. A remedy is the fast multipole method (FMM), which reduces the computational complexity from $O(N^2)$ to $O(N)$. In particular, the FLGF method (Liska and Colonius 2014) uses a kernel-independent interpolation-based fast multipole method to compute the discrete convolutions in conjunction with block-wise FFT convolution. The FMM achieves linear complexity $O(N)$ by leveraging the fact that, the solution is much smoother in the far-field than in the near-field for an elliptic kernel. Thus, a low-rank representation of the kernel is sufficient to accurately compute the contribution of far-field, while only the near-field requires full-rank representation of the kernel.

Using the generic interpolation function $\phi(\boldsymbol{x})$ and the coarse grained sampling points $\boldsymbol{x}_0, ..., \boldsymbol{x}_{n-1}$, a low-rank approximation of the kernel $K(\boldsymbol{x}, \boldsymbol{y})$ may be obtained by

$$\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \phi(\boldsymbol{x}) K(\boldsymbol{x}_i, \boldsymbol{y}_i) \phi(\boldsymbol{y}). \tag{1.91}$$

The discrete convolution can then be approximated by

$$\varphi(\boldsymbol{x}_i) \approx \sum_{j=0}^{M-1} \tilde{K}(\boldsymbol{x}_i, \boldsymbol{y}_j) h(\boldsymbol{y}_j) = \sum_{j=0}^{M-1} \sum_{p=0}^{n-1} \sum_{q=0}^{n-1} \phi(\boldsymbol{x}_i) K(\boldsymbol{x}_p, \boldsymbol{y}_q) \phi(\boldsymbol{y}_j) h(\boldsymbol{y}_j) \quad i = 0, ..., N-1, \tag{1.92}$$

where N is the number of target points and M the number of source points. The near- and far-field contributions are accounted for by constructing a hierarchical decomposition of the domain for which Eq.(1.92) is evaluated recursively. Typically, an octree structure $\mathcal{T}$ (quadtree in two dimensions) is used for this purpose. Let the tree have a depth L_B and the root is assumed to have level 0 and the base level $L_B - 1$ corresponds to physical domain. The tree nodes are also referred to as octants and octants without children are leaf nodes. The set of leafs on level l is indicated by B^l_{Leafs}. In the context of the FLGF, each tree node corresponds to a region, which is defined to be a Cartesian block of $N_b = n_b^3$ cells. Further, the ith octant or block at level l is denoted by $\mathcal{B}_i^l$ and the set of all octants at level l by $B^l = \bigcup_{i=0}^{N_B^l} \mathcal{B}_i^l$, where N_B^l is the number of octants on level l. The set of children and the parents are denoted by $\mathcal{C}(\mathcal{B}_i^l)$ and $\mathcal{P}(\mathcal{B}_i^l)$, respectively.

The target field $u_i^{L_B-1}$, defined on the octant $\mathcal{B}_i^{L_B-1}$, consist of both near- and far-field contributions. The near-field contribution is given by the interaction, i.e., convolution, with region $\mathcal{N}(\mathcal{B}_i^{L_B-1})$, containing the octant itself and the nearest neighbor octants on the finest tree level $L_B - 1$. The far-field contributions are then evaluated recursively for the levels $l = L_B - 1, \ldots, 0$ and are defined as the convolution with octants in the influence region $\mathcal{I}(\mathcal{B}_i^l) = \{\hat{\mathcal{B}}_i^l \in \mathcal{F}(\mathcal{B}_i^l) \setminus \mathcal{F}(\mathcal{B}_i^{l-1})\}$, which includes the well-separated octants, i.e., $\mathcal{F}(\mathcal{B}_i^l) = \bigcup_{l=0}^{L_B-1} B^l \setminus \mathcal{N}(\mathcal{B}_i^l)$, but excludes the regions well-separated from its parents $\mathcal{F}(\mathcal{P}(\mathcal{B}_i^l))$. Schematically, the domain decomposition in near and far-field regions is depicted in Fig. 1.7.

In the FLGF method, the octants are defined to be Cartesian blocks and the convolution between each block and its influence list can be computed by block-wise FFT-based convolutions, which reduces to a complex Hadamard product in Fourier space (see Liska and Colonius 2014, for details on FFT-based convolutions). Note that compared to a direct summation as in Eq.(1.84), the computational complexity is reduced from $\mathcal{O}(N_b^2)$ to $\mathcal{O}(N_b \log N_b)$ for each block convolution.

It should be clear from the above that given a union of source blocks $B_s = \bigcup_{i=0}^{N_s-1} \mathcal{B}_{s,i}$ and target blocks $B_t = \bigcup_{i=0}^{N_t-1} \mathcal{B}_{t,i}$ the convolution can be evaluated as the sum of the individual convolutions as

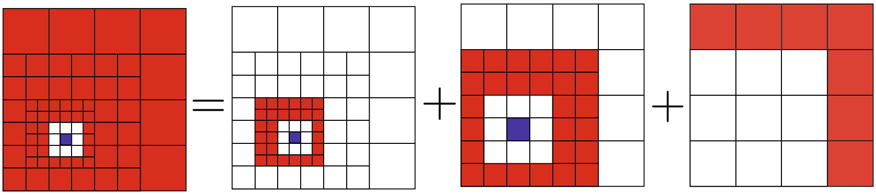


Fig. 1.7 Schematic of the hierarchical domain composition of the far-field (red, left) for an octant $\mathcal{B}_i^{L_B-1}$ (blue, left). While the near-field consist of the nearest neighbors only, the far-field is composed out of the set of influence lists for all levels. For level l the influence list contains the children of the nearest neighbors of $\mathcal{B}_i^l$'s parent, which are not contained in the near-field, i.e., are well separated. Reprinted from (Dorschner et al. 2020) with permission

$$\varphi_i = \sum_{j \in B_s} \mathrm{conv}(G^L_{i-j}, f_j), \quad \text{for } i = 0, ..., N_t - 1, \tag{1.93}$$

where the convolution operator is denoted as conv and G^L_{i-j} is the vector containing the unique values of $G^L(\boldsymbol{x}_i - \boldsymbol{x}_j)$ evaluated on the grid points $\boldsymbol{x}_j$ and $\boldsymbol{x}_i$ of the blocks $\mathcal{B}_{t,j}$ and $\mathcal{B}_{s,i}$, respectively.

With these definitions and the corresponding tree structure, the evaluation of Eq.(1.92) can be split into three consecutive steps. The first step is the *upward pass*, where the effective source terms are computed on each level by iterating bottom-up through the tree. Second, for each level, the convolution of each octant with its influence region is computed. This is called the *level interaction*. Finally, in the *downward pass* (iterating from the root to the leafs), all contributions are interpolated and accumulated on the next level. Schematically, the FLGF method is shown in Fig. 1.8 and can be summarized by the following algorithm:

1. *Upward pass*: Compute effective source terms at interpolation nodes
   For $l = L_B - 2, ..., 0$: For $i = 0, ...N_B^l$

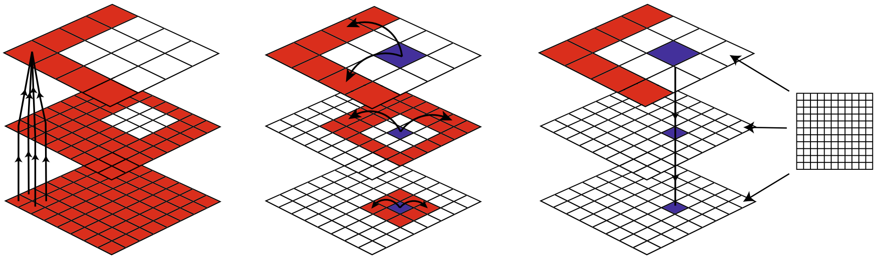


Fig. 1.8 Schematic of the fast multipole method: Left: *Upward pass*—Source regularization . Middle: *Level interaction*—Convolution of a block (blue) with its influence list (red). Right: *Downward pass*—Compute and accumulate the induced fields at the interpolation nodes. Note that each FMM cell corresponds to a Cartesian block in the FLGF. Reprinted from (Dorschner et al. 2020) with permission

$$\hat{h}_i^l = \sum_{j \in C(\mathcal{B}_i^l)} \hat{R}^{l+1} \hat{h}_j, \tag{1.94}$$

where the regularization operator $\hat{R}^{l+1}$ is the adjoint of the interpolation operator $\hat{E}^{l+1}$(see below).

2. *Level Interaction*: FFT convolution with the octant in the influence region
   For $l = 0, ..., L_B - 1$: For $i = 0, ...N_B^l$

$$\hat{v}_i^l = \sum_{j \in \mathcal{I}(\mathcal{B}_i^l)} \text{conv}(G^L_{i-j}, f_j), \tag{1.95}$$

   where conv$(\cdot)$ is the FFT convolution operator.

3. *Downward pass*: Compute and accumulate induced field at interpolation nodes
   For $l = 0, ..., L_B - 1$: For $i = 0, ...N_B^l$

$$\hat{\varphi}_i^l = \hat{v}_i^l + \hat{E}_i^{l-1} \hat{\varphi}_i^{l-1}, \tag{1.96}$$

   where the interpolation operator $\hat{E}^l$ interpolates from the parent onto the child block.

Owing to the regularity of the Cartesian block mesh, the interpolation operators are implemented using Lagrangian polynomials, and $n_I \leq 10$ interpolation nodes are used to yield a relative interpolation error of $\epsilon \approx 10^{-12}$ for an analytic function approximation. The regularization operator is given by the adjoint of the interpolation operator and often called anterpolation. In summary, the FLGF method combines the fastest methods for regular meshes, while retaining the geometrical flexibility and overall linear complexity inherent to FMM. Excellent computational rates and parallel performance have been reported in Liska and Colonius (2014) and (Dorschner et al. 2020).

1.3.3.2 Domain Adaptivity

As mentioned previously, the LGF method may truncate the formally unbounded computationally domain to regions with non-negligible source. When the LGF is employed in the context of the IBLGF this translates to solving the NSE only in regions that are dictated by the flow evolution and cells which, up to a prescribed threshold, do not affect the flow evolution may be removed in the course of a simulation. This is a direct consequence of the vorticity, $\omega = \nabla \times \boldsymbol{u}$, decaying exponentially at large distances form the immersed body. For instance, considering the solution of Eq. (1.79a):

$$p^*(\boldsymbol{x}_i) = [G^L_C * y](\boldsymbol{x}_i), \quad y(\boldsymbol{x}_i) = [-G^\dagger r^i_k](\boldsymbol{x}_i), \tag{1.97}$$

where $G^{\dagger}r_k^i$ is a discrete approximation of the divergence of the Lamb vector $\nabla \cdot \boldsymbol{l}$, where $\boldsymbol{l} = \boldsymbol{\omega} \times \boldsymbol{u}$, while $\boldsymbol{x}_i$ denotes a discrete location. Since $\omega \to 0$ at large distances from the body, it follows that also $\nabla \cdot \boldsymbol{l}$ and its discrete counterpart become exponentially small. Thus, the domain on which the field is induced is finite and is truncated when $G^{\dagger}r_k^i$ is smaller than a given threshold ϵ. When using this procedure, the discrete velocity needs to be refreshed using the discrete velocity $w = Cu$ in order to obtain a consistent velocity field. The domain adaptivity is implemented using the block-structured mesh as outlined above. For further implementation details on the domain adaptivity, the reader is referred to Liska and Colonius (2016).

1.3.4 Examples

1.3.4.1 Flow Past a Sphere

In this section, the benchmark simulation of the flow past an impulsively started sphere at Reynolds number Re $= 3700$ (Liska and Colonius 2017) is summarized. Additional benchmarks and validation studies can be found in Liska and Colonius (2017), Mengaldo et al. (2017). The turbulent flow past a sphere at Re $= 3700$ is a challenging, canonical benchmark problem and has thus been investigated by many researchers both experimentally (Kim and Durbin 1988) and numerically (Yun et al. 2006; Rodriguez et al. 2011; Dorschner et al. 2016). The nominal velocity of the sphere is $(U, 0, 0)$ and a small, initial perturbation is introduced in the flow field in order to break any symmetries. The flow is computed for $0 \leq t^*/U \leq 60$ using 81,920 Lagrangian markers, where t^* indicates initialization from a large-time solution of a sphere at Re $= 1000$. The sphere has unit radius and the grid spacing was chosen conservatively as $\Delta x \simeq 4.3 \times 10^{-3}$ to resolve the thin boundary layer on the surface of the sphere. The time step size is chosen such that the CFL number based on the maximum point-wise velocity remains below 0.9. The time-averaged results were obtained over the last five large-scale vortex shedding cycles (St $= 0.215$). A snapshot of the vortical structures using the Q-criterion is shown in Fig. 1.9 and is in line with what is reported in the literature. The adaptive nature of the grid is shown in Fig. 1.9, where the computational domain is comprised of only regions, where the vorticity is non-negligible. In this setup, the threshold for the grid adaptivity was chosen as $\epsilon = 5 \times 10^{-4}$ and verified to accurately capture the unbounded domain flow. The flow is further characterized by the mean surfaces stresses and the net body forces. First, we consider the mean skin friction coefficient $C_{\mathrm{f}} = \tau_w/(\frac{1}{2}\rho U^2)$, where τ_{w} is the local wall shear stress and the pressure coefficient $C_{\mathrm{p}} = (p - p_{\infty})/(\frac{1}{2}\rho U)$, where p and p_{∞} are the local and the free-stream pressure, respectively. Note that the raw point-wise values of the surface stress tensor are partially filtered using the boundary force post-processing technique (Goza et al. 2016), to construct a smoothed boundary forces. The time-averaged results of C_{p} and C_{f} along the polar angle θ are shown in Fig. 1.10. Finally, the mean values for drag coefficient $\overline{C}_{\mathrm{d}}$, base pressure

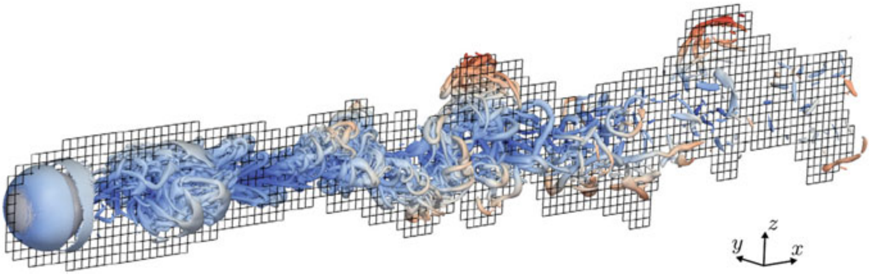


Fig. 1.9 Vortex cores in the wake of a sphere at Re $= 3700$ are illustrated by isosurfaces of constant Q-value. Depicted are isosurface of $QD^2/U^2 = 2$ colored by radial distance from the centerline of the sphere in the streamwise direction. The mesh is a cross-sectional cut of the block-wise adaptive computational domain that have been coarsened by a factor of two in each direction for visualization purposes. Reprinted from Liska and Colonius (2017) with permission

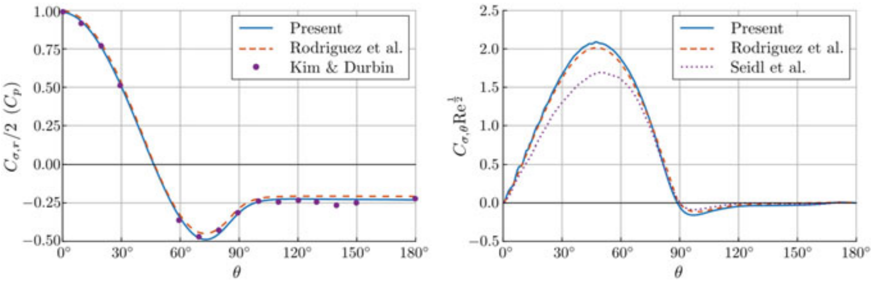


Fig. 1.10 Comparison of the time-averaged pressure (left) and skin friction (right) coefficients as functions of the polar angle, θ, for a sphere at Re $= 3700$. Results are compared to values reported by Rodriguez et al. (2011) (DNS at Re = 3700), Kim and Durbin (1988) (exp. at Re = 4200), and Seidl et al. (1997) (DNS at Re = 5000). Reprinted from Liska and Colonius (2017) with permission

coefficient $\overline{C}_{\mathrm{pb}}$, separation angle $\overline{\theta}_{\mathrm{s}}$, polar locations of the minimum surface pressure $\overline{\theta}_{\mathrm{p,min}}$ and of the maximum skin friction $\overline{\theta}_{\tau,\mathrm{max}}$ are reported in Table 1.1 and shows good agreement with the literature values.

1.3.4.2 Flow Past an Inclined, Rotating, Circular Disk

As an outlook, this setup can easily be extended to rotating disks by prescribing appropriate boundary velocities for the Lagrangian IB points. Such studies are of interest for developing novel designs of unmanned air vehicles or alike. While spheres and disks have both been studied extensively in literature, significant discrepancies between numerical and experimental measurements for disks and flat cylinders of large aspect ratio have fueled recent interest in further numerical investigations and thus makes it a good benchmark for the IBLGF (Gao et al. 2018; Tian et al. 2017;

Table 1.1 Turbulent flow past sphere at $\mathrm{Re} = 3700$ and the comparison of the mean values for drag coefficient $\overline{C}_\mathrm{d}$, base pressure coefficient $\overline{C}_\mathrm{pb}$, separation angle $\overline{\theta}_\mathrm{s}$, polar locations of the minimum surface pressure $\overline{\theta}_{p,\mathrm{min}}$ and of the maximum skin friction $\overline{\theta}_{\tau,\mathrm{max}}$ with literature values

Contribution		Re	$\overline{C}_d$	$\overline{C}_\mathrm{pb}$	$\overline{\theta}_\mathrm{s}$	$\overline{\theta}_\mathrm{p,min}$	$\overline{\theta}_{\tau,\mathrm{max}}$
IBLGF	DNS	3700	0.389	−0.230	88.9	73	47
Yun et al. (2006)	LES	3700	0.355	−0.194	90	–	–
Rodriguez et al. (2011)	DNS	3700	0.394	−0.207	89.4	72	48
Dorschner et al. (2016)	DNS	3700	0.383	−0.220	89.993	–	–
Kim and Durbin (1988)	exp.	4200	–	−0.224	–	–	–
Seidl et al. (1997)	DNS	5000	0.38	–	89.5	71	50

Chrust et al. 2015). In addition, an extension to rotating disks (e.g., Frisbees) will be presented. The rotating disk has been suggested as a configuration for a micro-air vehicle based on the stabilizing influence of rotation on the flight dynamics (Potts and Crowther 2001, 2002; Lorenz 2007).

The Reynolds number is defined as $\mathrm{Re} = \frac{U_\infty D \sin(\alpha)}{\nu}$, where U_∞ is the free-stream velocity, α is the angle of attack, and D is the disk diameter, which is set to unity. The Strouhal number is defined by $\mathrm{St} = \frac{f D \sin(\alpha)}{U_\infty}$, where f is the frequency of the primary vortex shedding.

Mean drag and lift coefficients for various angles of attack and a fixed Reynolds number of $\mathrm{Re} = 500$ are compared to the numerical simulations of Tian et al. (2017) in Fig. 1.11. The grid spacing was chosen to be $\Delta x = 0.012$ and the time step is set to $\Delta t = 0.004$. The adaptivity threshold for the grid was set to $\epsilon = 5 \times 10^{-4}$ unless stated otherwise. For simulations at lower Reynolds numbers the grid spacing is chosen based on the estimated scaling of the boundary layer. With an expected $\mathrm{Re}^{-\frac{1}{2}}$ scaling of the laminar boundary layer thickness, the value of $(\Delta x)\mathrm{Re}^{\frac{1}{2}}$ is kept approximately constant as in Liska and Colonius (2017). The CFL number based on the maximum point-wise velocity is kept below 0.5 by setting $\Delta t/\Delta x = 1/3$. While in Tian et al. (2017) the disk was modeled to have a thickness of $0.002D$, the

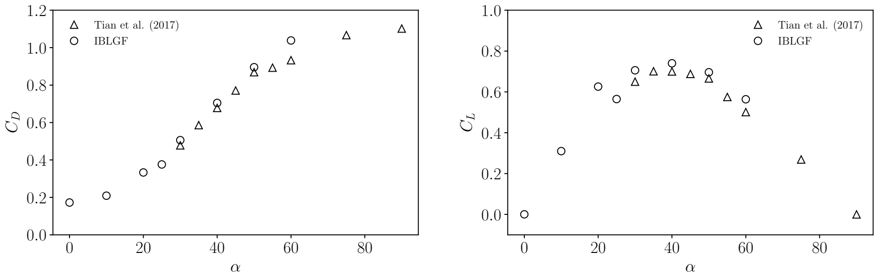


Fig. 1.11 Drag and lift coefficient for various angles of attack α at $\mathrm{Re} = 500$

IBLGF simulations assume an infinitely thin disk, which is in turn regularized on the grid scale via the discrete delta function. Mean quantities are averaged over at least 170 convective time units for unsteady solutions ($\alpha \geq 25°$) and 130 units for steady solutions ($\alpha \leq 20°$). Good agreement between both numerical simulations validates the IBLGF. Exemplary, in Fig. 1.13 the vorticity isosurfaces for ω_x along with a slice of the computational mesh are shown for angles of attack of $\alpha = 20°$ and $\alpha = 70°$, respectively. Note the disjoint computational mesh for $\alpha = 70°$, which reduces the computational cost significantly compared to a uniform solver (Fig. 1.12).

In addition, the simulations are validated by comparison of the critical Reynolds number and Strouhal number as the flow transitions from a steady state regime to a periodic state through a supercritical Hopf bifurcation with increasing Reynolds number and fixed angle of attack. In Fig. 1.12, the threshold of the Hopf instability as a function of angle of attack is plotted in terms of critical Reynolds and Strouhal number. The critical values for both Reynolds and Strouhal number are obtained by extrapolating from unsteady cases run near the critical point as done in Ghaddar et al. (1986), Pereira and Sousa (1993). As apparent from the plot, the critical values

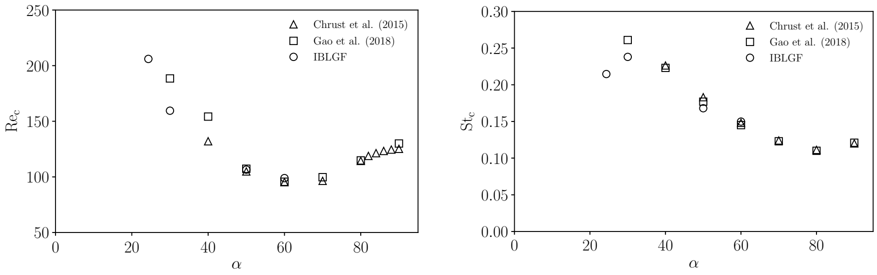
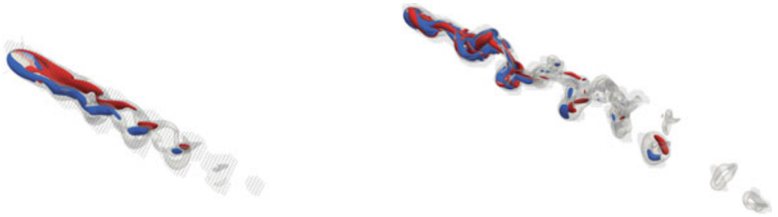


Fig. 1.12 Critical Reynolds number Re$_c$ and critical Strouhal number St$_c$ for various angles of attack α



(a) $\alpha = 25°$. Red, blue and grey isosurfaces are shown for $\omega_x = +2$, $\omega_x = -2$ and $|\omega| = 1.5$, respectively.

(b) $\alpha = 70°$. Red, blue and grey isosurfaces are shown for $\omega_x = +0.5$,$\omega_x = -0.5$ and $|\omega| = 0.25$, respectively. The adaptive threshold was set to $\epsilon = 5 \cdot 10^{-5}$ for this case.

Fig. 1.13 Isosurfaces of streamwise vorticity ω_x for flow over an infinitely thin disk at Re $= 500$, $\lambda = 0$. The grid of blocks is shown on the spanwise xy-plane through the centerline of the disk. Each block consists of $10 \times 10 \times 10$ cells. Adaptive threshold is set to $\epsilon = 5 \times 10^{-5}$

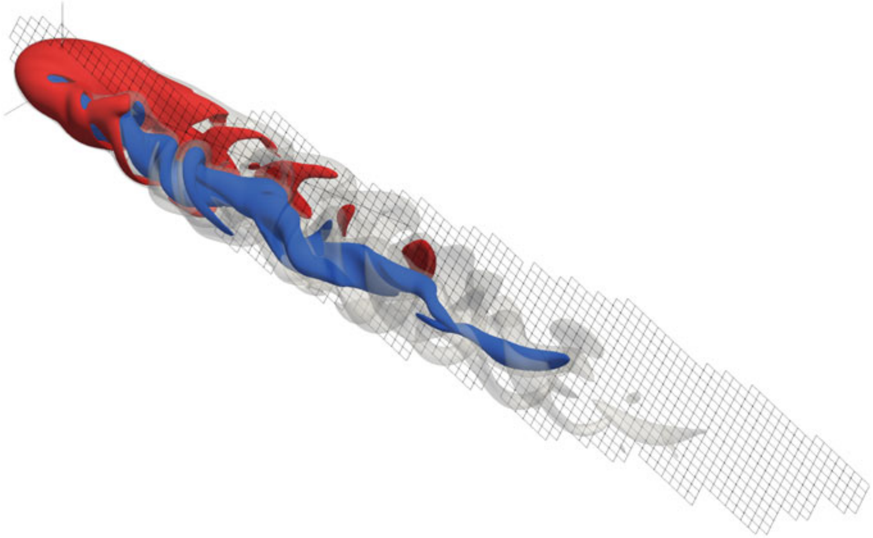


Fig. 1.14 Isosurfaces of vorticity for flow over an infinitely thin disk at Re = 500, an angle of attack $\alpha = 0$ and rotating at a tip-speed ratio of $\lambda = 3$. See also caption of Fig. 1.13

agree well with the numerical simulations of Gao et al. (2018), Chrust et al. (2015). Note that while (Chrust et al. 2015) modeled an infinitely thin disk, the simulations reported in Gao et al. (2018) modeled the disk to have a thickness of $0.02D$. A snapshot of a rotating disk with a tip-speed ratio of $\lambda = 3$ is shown in Fig. 1.14 as an outlook of future studies.

1.3.5 Toward AMR

Note that while in the IBLGF the grid is adaptive, the resolution is uniform, which can become limiting in many applications. In fact, while uniform Cartesian meshes can significantly decrease the cost per degree of freedom (DoF), the total number of DoF can be prohibitive for strongly anisotropic or inhomogeneous problems with localized source regions. This issue is particularly prominent for, e.g., high Reynolds number flows or problems where the range of scales is large. A particularly acute challenge is for bluff bodies where one must resolve thin attached laminar or turbulent boundary layers at the same time as a broad wake. Here, adaptive mesh refinement methods (AMR), pioneered by Berger and Colella (1989), Berger and Oliger (1984) can be crucial. These methods adapt the local density of the computational elements to the local spatial resolution requirements. In the context of the IB method, AMR techniques have been embedded (Roma et al. 1999; Griffith et al. 2007; Vanella et al. 2014) and have shown to greatly reduce the computational costs of simulations. In addition, block-wise AMR has become popular compared to cell-wise refinement (see Dubey et al. 2014 for a survey of popular block-structured mesh refinement packages), since the overhead of the underlying data structure and load balancing has shown to be significantly more efficient (Nissen et al. 2013; Dreher and Grauer 2005).

While the FLGF method discussed above is adaptive and utilizes a hierarchical block-structured grid for the FMM solution, it is not an AMR technique. A natural extension of the FLGF is thus to a block-structured AMR solver without additional overhead of the underlying data structure. Note, however, that when refining the physical domain by embedding locally refined grid patches within the computational domain, the free-space boundary conditions implied by the lattice Green's functions become problematic since the refinement patches itself do have a well defined boundary condition, which is imposed by the surrounding domain and is not the free space. A remedy was found in Dorschner et al. (2020), by projecting the source field onto each level within its support by appropriate coarsening and interpolation operators and subsequent application of the FLGF method on each level independently. The resulting multi-resolution scheme retains linear computational complexity and second-order accuracy and additional allows for arbitrary block-wise refinement by factors of two. An exemplary mesh topology of the multi-resolution scheme is shown in Fig. 1.15 for a thin vortex ring.

While in Dorschner et al. (2020) the groundwork toward a multi-resolution IBLGF solver was laid, its incorporation into the full IBLGF solver is ongoing work.

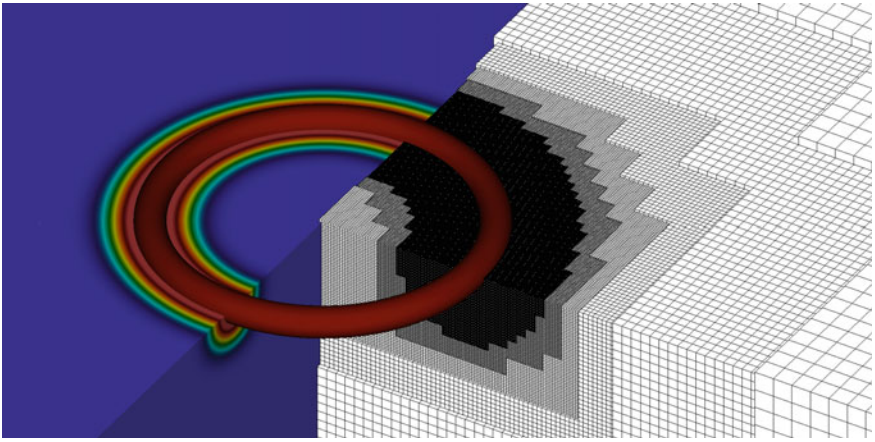


Fig. 1.15 Mesh topology and vorticity field for a thin vortex ring and the numerical solution of the streamfunction for five levels of refinement

1.4 Summary and Future Directions

In this chapter, we have summarized a projection approach to the IB method that exploits features of the underlying, uniform Cartesian mesh to achieve robust, efficient solution of incompressible external flow problems. Extensions for fully coupled FSI for thin elastic structures, and a novel fast lattice Green's function for 3D problems were highlighted. Examples problems such as inverted flag flutter, and flow past a sphere and rotating disks were presented to demonstrate the capabilities of the numerics for complex flow problems.

The IBPM takes advantages of formulating the continuity and no-slip boundary conditions as Lagrange multipliers to enforce these constraints to prescribed precision without the need of an artificial constitute relation. This formulation removes any additional constraints on the CFL number for rigid bodies, which is then only limited by the choice of the time-stepping algorithm. Through carefully chosen discrete operators, a modified Poisson equation arises, which is symmetric and positive definite and thus amenable to efficient numerical solvers such as conjugate-gradient method. In addition, we reviewed appropriate filtering techniques to accurately compute surfaces stresses and forces, which allowed the extension to strongly coupled fluid interaction for thin elastic structures. The simplicity of the IBPM formulation proves useful in finding (potentially unstable) steady equilibrium solutions of coupled FSI problems, as well as solving associated global linear stability problems. This capability was demonstrated on an inverted elastic flag setup.

When combined with a lattice Green's function approach, the IB method can be used for three-dimensional external flows while, at the same time, minimizing the computational expense by restricting the computational domain to a snug region near the immersed surface in which the vorticity is nonzero. The method naturally incorporates exact free-space boundary conditions. By using a viscous integrating factor half-explicit Runge–Kutta scheme in combination with the approximation-free nested projection technique and exploiting mimetic properties of the discrete operators, the projection steps reduce to simple discrete elliptic problems, which can be solved efficiently with the FLGF method. The FLGF method is based on a kernel-independent interpolation-based fast multipole method and therefore exhibits linear computational complexity, while taking advantage of a block-structured Cartesian mesh by using standard FFT routines for fast convolutions.

While the combined IBLGF approach has been demonstrated in benchmarks such as the turbulent flow past a sphere or flow past an inclined rotating disk, its extension to higher Reynolds numbers requires further developments. While it is true that uniform grids in general allow us to use efficient numerical solvers, the number of degrees of freedom becomes prohibitive for the large range of scales in high Reynolds number flows. Thus, future efforts are directed toward a multi-resolution IBLGF scheme. A block-structured multi-resolution algorithm based on LGFs for elliptic difference equations, which retains favorable computational complexity and necessary regularity has been detailed in Dorschner et al. (2020). Incorporation in the IBLGF will be reported in future contributions. Such multi-resolution scheme will not only open the door for high Reynolds number direct numerical simulations but also inclusion of explicit turbulence models such as wall-resolved or wall-modeled large-eddy simulations (LES). In the context of IB, this has been successfully implemented in Piomelli and Balaras (2002), Iaccarino and Verzicco (2003), Yang and Balaras (2006), You et al. (2007), Catchirayer et al. (2018) (and references therein) and therefore provides a promising path toward high Reynolds numbers and applications with engineering relevant complexity. In addition, by using Krylov solvers such as GMRES, BIGSTAB, and their flexible extensions (Saad 1993; Chen et al. 2016) to solve the force Schur complement, the IBLGF scheme can be extended to a fully coupled three-dimensional fluid–structure solver. This would retain the benefits of IBLGF in terms of accuracy, scalability, and efficiency due to truncated computational domain, while increasing the range of applications significantly.

Acknowledgements BD gratefully acknowledges support from the Swiss National Science Foundation (SNF Grant No. P2EZP2_178436), TC was supported by ONR grant N00014-16-1-2734. We are likewise grateful to numerous students and collaborators, particularly Sam Taira, Sebastian Liska, and Andres Goza, who contributed to the development of IB projection techniques described herein. We thank Marcus Lee and Ke Yu for recent contributions to the AMR code development; Marcus Lee also contributed his results for flow past a spinning disk to this article.

References

Ahn HT, Kallinderis Y (2006) Strongly coupled flow/structure interactions with a geometrically conservative ALE scheme on general hybrid meshes. J Comput Phys 219(2):671–696

Bathe KJ (1996) Finite element procedures. Prentice-Hall, Englewood Cliffs, NJ

Berger MJ, Colella P (1989) Local adaptive mesh refinement for shock hydrodynamics. J Comput Phys 82(1):64–84

Berger MJ, Oliger J (1984) Adaptive mesh refinement for hyperbolic partial differential equations. J Comput Phys 53(3):484–512

Beyer RP, LeVeque RJ (1992) Analysis of a one-dimensional model for the immersed boundary method. SIAM J Numer Anal 29(2):332–364

Borazjani I, Ge L, Sotiropoulos F (2008) Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies. J Comput Phys 227(16):7587–7620. https://doi.org/10.1016/j.jcp.2008.04.028

Brasey V, Hairer E (1993) Half-explicit Runge-Kutta methods for differential-algebraic systems of index 2. SIAM J Numer Anal 30(2):538–552

Catchirayer M, Boussuge JF, Sagaut P, Montagnac M, Papadogiannis D, Garnaud X (2018) Extended integral wall-model for large-eddy simulations of compressible wall-bounded turbulent flows. Phys Fluids 30(6):065106

Causin P, Gerbeau JF, Nobile F (2005) Added-mass effect in the design of partitioned algorithms for fluid-structure problems. Comput Methods Appl Mech Eng 194(42–44):4506–4527. https://doi.org/10.1016/j.cma.2004.12.005

Chang W, Giraldo F, Perot B (2002) Analysis of an exact fractional step method. J Comput Phys 180(1):183–199

Chen J, McInnes LC, Zhang H (2016) Analysis and practical use of flexible BiCGStab. J Sci Comput 68(2):803–825

Chrust M, Dauteuille C, Bobinski T, Rokicki J, Goujon-Durand S, Wesfreid J, Bouchet G, Dušek J (2015) Effect of inclination on the transition scenario in the wake of fixed disks and flat cylinders. J Fluid Mech 770:189–209

Colella P, Graves DT, Keen BJ, Modiano D (2006) A Cartesian grid embedded boundary method for hyperbolic conservation laws. J Comput Phys 211(1):347–366

Colonius T (2004) Modeling artificial boundary conditions for compressible Flow. Ann Rev Fluid Mech 36(1):315–345

Colonius T, Taira K (2008) A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions. Comput Methods Appl Mech Eng 197(25–28):2131–2146. https://doi.org/10.1016/j.cma.2007.08.014

De Borst R, Crisfield MA, Remmers JJ, Verhoosel CV (2012) Nonlinear finite element analysis of solids and structures. Wiley

Degroote J, Bathe KJ, Vierendeels J (2009) Performance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction. Comput Struct 87(11–12):793–801

Delves RT, Joyce GS (2001) On the Green function for the anisotropic simple cubic lattice. Ann Phys 291(1):71–133

Dorschner B, Frapolli N, Chikatamarla SS, Karlin IV (2016) Grid refinement for entropic lattice Boltzmann models. Phys Rev E 94(5):053311

Dorschner B, Yu K, Mengaldo G, Colonius T (2020) A fast multi-resolution lattice green's function method for elliptic difference equations. J Comput Phys 109270

Dreher J, Grauer R (2005) Racoon: A parallel mesh-adaptive framework for hyperbolic conservation laws. Parallel Comput 31(8–9):913–932

Dubey A, Almgren A, Bell J, Berzins M, Brandt S, Bryan G, Colella P, Graves D, Lijewski M, Löffler F et al (2014) A survey of high level frameworks in block-structured adaptive mesh refinement packages. J Parallel Distrib Comput 74(12):3217–3227

Fadlun E, Verzicco R, Orlandi P, Mohd-Yusof J (2000) Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. J Comput Phys 161(1):35–60

Förster C, Wall WA, Ramm E (2007) Artificial added mass instabilities in sequential staggered coupling of nonlinear structures and incompressible viscous flows. Computer Methods Appl Mech Eng 196(7):1278–1293. https://doi.org/10.1016/j.cma.2006.09.002

Gao S, Tao L, Tian X, Yang J (2018) Flow around an inclined circular disk. J Fluid Mech 851:687–714

Ghaddar N, Korczak K, Mikic B, Patera A (1986) Numerical investigation of incompressible flow in grooved channels. Part 1. Stability and self-sustained oscillations. J Fluid Mech 163:99–127

Glasser ML, Zucker IJ (1977) Extended Watson integrals for the cubic lattices. Proc Natl Acad Sci 74(5):1800–1801

Glowinski R, Pan TW, Periaux J (1998) Distributed Lagrange multiplier methods for incompressible viscous flow around moving rigid bodies. Comput Methods Appl Mech Eng 151(1–2):181–194

Goldstein D, Handler R, Sirovich L (1993) Modeling a no-slip flow boundary with an external force field. J Comput Phys 105(2):354–366

Goza A, Colonius T (2017) A strongly-coupled immersed-boundary formulation for thin elastic structures. J Comput Phys 336:401–411

Goza A, Liska S, Morley B, Colonius T (2016) Accurate computation of surface stresses and forces with immersed boundary methods. J Comput Phys 321:860–873

Goza A, Colonius T, Sader JE (2018) Global modes and nonlinear analysis of inverted-flag flapping. J Fluid Mech 857:312–344

Griffith BE, Hornung RD, McQueen DM, Peskin CS (2007) An adaptive, formally second order accurate version of the ibm-Peskin.pdf. J Comput Phys 223(1):10–49

Gurugubelli P, Jaiman R (2015) Self-induced flapping dynamics of a flexible inverted foil in a uniform flow. J Fluid Mech 781:657–694

Hairer E, Lubich C, Roche M (2006) The numerical solution of differential-algebraic systems by Runge-Kutta methods, vol 1409. Springer, Berlin

Hall CA (1985) Numerical solution of Navier-Stokes problems by the dual variable method. SIAM J Algebraic Discrete Methods 6(2):220–236

Hansen PC (1998) Rank-deficient and discrete illposed problems: numerical aspects of linear inversion, vol 4. SIAM

Hermansson J, Hansbo P (2003) A variable diffusion method for mesh smoothing. Commun Numer Methods Eng 19(11):897–908. https://doi.org/10.1002/cnm.639

Hirt CW, Amsden AA, Cook JL (1974) An arbitrary Lagrangian-Eulerian computing method for all flow speeds. J Comput Phys 14(3):227–253

Hou G, Wang J, Layton A (2012) Numerical methods for fluidstructure interaction—a review. Commun Comput Phys 12(2):337–377

Hu XY, Khoo B, Adams NA, Huang F (2006) A conservative interface method for compressible flows. J Comput Phys 219(2):553–578

Huang WX, Sung HJ (2009) An immersed boundary method for fluid-flexible structure interaction. Comput Methods Appl Mech Eng 198(33):2650–2661

Iaccarino G, Verzicco R (2003) Immersed boundary technique for turbulent flow simulations. Appl Mech Rev 56(3):331–347

Kallemov B, Bhalla A, Griffith B, Donev A (2016) An immersed boundary method for rigid bodies. Commun Appl Math Comput Sci 11(1):79–141

Kim H, Durbin P (1988) Observations of the frequencies in a sphere wake and of drag increase by acoustic excitation. Phys Fluids 31(11):3260–3265

Kim D, Cossé J, Cerdeira CH, Gharib M (2013) Flapping dynamics of an inverted flag. J Fluid Mech 736

Kirkpatrick M, Armfield S, Kent J (2003) A representation of curved boundaries for the solution of the Navier-Stokes equations on a staggered three-dimensional Cartesian grid. J Comput Phys 184(1):1–36

Kress R (2014) Linear integral equations, 3rd edn, vol 82. Springer, Berlin

Lai MC, Peskin CS (2000) An immersed boundary method with formal second-order accuracy and reduced numerical viscosity. J Comput Phys 160(2):705–719

Le Tallec P (2001) Fluid structure interaction with large structural displacements. Computer Methods Appl Mech Eng 190(24–25):3039–3067. https://doi.org/10.1016/S0045-7825(00)00381-9

Li Z, Favier J (2017) A non-staggered coupling of finite element and lattice Boltzmann methods via an immersed boundary scheme for fluid structure interaction. Comput Fluids 143:90–102. https://doi.org/10.1016/j.compfluid.2016.11.008

Lilly DK (1965) On the computational stability of numerical solutions of time-dependent non-linear geophysical fluid dynamics problems. Mon Weather Rev 93(1):11–26

Liska S, Colonius T (2014) A parallel fast multipole method for elliptic difference equations. J Comput Phys 278:76–91

Liska S, Colonius T (2016) A fast lattice Green's function method for solving viscous incompressible flows on unbounded domains. J Comput Phys 316:360–384

Liska S, Colonius T (2017) A fast immersed boundary method for external incompressible viscous flows using lattice Green's functions. J Comput Phys 331:257–279

Lorenz RD (2007) Spinning flight: dynamics of frisbees, boomerangs, samaras, and skipping stones. Springer Science & Business Media

Mengaldo G, Liska S, Colonius KYT, Jardin T (2017) Immersed boundary lattice Green function methods for external aerodynamics. In: 23rd AIAA computational fluid dynamics conference, p 3621

Mittal R, Iaccarino G (2005) Immersed boundary methods. Ann Rev Fluid Mech 37:239–261

Mohd-Yusof J (1997) For simulations of flow in complex geometries. Ann Res Briefs 317–327

Mori Y, Peskin CS (2008) Implicit second-order immersed boundary methods with boundary mass. Comput Methods Appl Mech Eng 197(25–28):2049–2067

Morinishi Y, Lund TS, Vasilyev OV, Moin P (1998) Fully conservative higher order finite difference schemes for incompressible flow. J Comput Phys 143(1):90–124

Nakata T, Liu H (2012) A fluid-structure interaction model of insect flight with flexible wings. J Comput Phys 231(4):1822–1847

Nicolaides RA (1992) Direct discretization of planar div-curl problems. SIAM J Numer Anal 29(1):32–56

Nicolaides RA, Wu X (1997) Covolume solutions of three-dimensional div-curl equations. SIAM J Numer Anal 34(6):2195–2203

Nissen A, Kreiss G, Gerritsen M (2013) High order stable finite difference methods for the Schrodinger equation. J Sci Comput 55(1):173–199

Pereira J, Sousa J (1993) Finite volume calculations of self-sustained oscillations in a grooved channel. J Comput Phys 106(1):19–29

Perot B (2000) Conservation properties of unstructured staggered mesh schemes. J Comput Phys 159(1):58–89

Peskin CS (1972) Flow patterns around heart valves: A numerical method. J Comput Phys 10(2):252–271. https://doi.org/10.1016/0021-9991(72)90065-4

Piomelli U, Balaras E (2002) Wall-layer models for large-eddy simulations. Ann Rev Fluid Mech 34(1):349–374

Piperno S, Farhat C (2001) Partitioned procedures for the transient solution of coupled aeroelastic problems - part II: Energy transfer analysis and three-dimensional applications. Comput Methods Appl Mech Eng 190(24–25):3147–3170

Potts J, Crowther W (2001) Flight control of a spin stabilised axisymmetric disc-wing. In 39th aerospace sciences meeting and exhibit, p 253

Potts J, Crowther W (2002) Frisbee (TM) aerodynamics. In 20th AIAA applied aerodynamics conference, p 3150

Pradeep DS, Hussain F (2004) Effects of boundary condition in numerical simulations of vortex dynamics. J Fluid Mech 516:115–124

Rodriguez I, Borell R, Lehmkuhl O, Segarra CDP, Oliva A (2011) Direct numerical simulation of the flow over a sphere at Re= 3700. J Fluid Mech 679:263–287

Roma AM, Peskin CS, Berger MJ (1999) An Adaptive Version of the Immersed Boundary Method. J Comput Phys 153(2):509–534. https://doi.org/10.1006/jcph.1999.6293

Ryu J, Park SG, Kim B, Sung HJ (2015) Flapping dynamics of an inverted flag in a uniform flow. J Fluids Struct 57

Saad Y (1993) A flexible inner-outer preconditioned GMRES algorithm. SIAM J Sci Comput 14(2):461–469

Seidl V, Muzaferija S, Perić M (1997) Parallel DNS with local grid refinement. Appl Sci Res 59(4):379–394

Seo JH, Mittal R (2011) A sharp-interface immersed boundary method with improved mass conservation and reduced spurious pressure oscillations. J Comput Phys 230(19):7347–7363

Shelley MJ, Zhang J (2011) Flapping and bending bodies interacting with fluid flows. Ann Rev Fluid Mech 43:449–465

Sotiropoulos F, Yang X (2014) Immersed boundary methods for simulating fluid-structure interaction. Prog Aerosp Sci 65:1–21. https://doi.org/10.1016/j.paerosci.2013.09.003

Stein DB, Guy RD, Thomases B (2017) Immersed Boundary Smooth Extension (IBSE): A high-order method for solving incompressible flows in arbitrary smooth domains. J Comput Phys 335:155–178

Taira K, Colonius T (2007) The immersed boundary method: A projection approach. J Comput Phys 225(2):2118–2137. https://doi.org/10.1016/j.jcp.2007.03.005

Taneda S (1968) Waving motions of flags. J Phys Soc Jpn 24(2):392–401

Tezduyar TE, Behr M, Mittal S, Liou J (1992) New strategy for finite element computations involving moving boundaries and interfaces. The deforming-spatial domain/space-time procedure. II. Computation of free surface flows, two-liquid flows, and flows with drifting cylinders. Comput Methods Appl Mech Eng 94(3):353–371

Tezduyar TE, Sathe S, Keedy R, Stein K (2006) Space-time finite element techniques for computation of fluid-structure interactions. Comput Methods Appl Mech Eng 195(17–18):2002–2027

Thompson JF, Soni BK, Weatherill NP (1998) Handbook of grid generation. CRC Press. p 1136 https://doi.org/10.1201/9781420050349

Tian X, Xiao L, Zhang X, Yang J, Tao L, Yang D (2017) Flow around an oscillating circular disk at low to moderate Reynolds numbers. J Fluid Mech 812:1119–1145

Tornberg AK, Engquist B (2004) Numerical approximations of singular source terms in differential equations. J Comput Phys 200(2):462–488

Tsynkov SV (1998) Numerical solution of problems on unbounded domains. A review. Appl Numer Mathematics 27(4):465–532. https://doi.org/10.1016/S0168-9274(98)00025-7

Uhlmann M (2005) An immersed boundary method with direct forcing for the simulation of particulate flows. J Comput Phys 209(2):448–476

Vanella M, Posa A, Balaras E (2014) Adaptive mesh refinement for immersed boundary methods. J Fluids Eng 136(4):040909

Wang S, Zhang X (2011) An immersed boundary method based on discrete stream function formulation for two- and three-dimensional incompressible flows. J Comput Phys 230(9):3479–3499

Yang J, Balaras E (2006) An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries. J Comput Phys 215(1):12–40

Yang X, Zhang X, Li Z, He GW (2009) A smoothing technique for discrete delta functions with application to immersed boundary method in moving boundary simulations. J Comput Phys 228(20):7821–7836

Ye T, Mittal R, Udaykumar H, Shyy W (1999) An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries. J Comput Phys 156(2):209–240

You D, Wang M, Moin P, Mittal R (2007) Large-eddy simulation analysis of mechanisms for viscous losses in a turbomachinery tip-clearance flow. J Fluid Mech 586:177–204

Yun G, Kim D, Choi H (2006) Vortical structures behind a sphere at subcritical Reynolds numbers. Phys Fluids 18(1):015102. https://doi.org/10.1063/1.2166454

Zahedi S, Tornberg AK (2010) Delta function approximations in level set methods by distance function extension. J Comput Phys 229:2199–2219

Zhang N, Zheng ZC (2007) An improved direct-forcing immersed-boundary method for finite difference applications. J Comput Phys 221(1):250–268

Zhang X, Schmidt D, Perot B (2002) Accuracy and conservation properties of a three-dimensional unstructured staggered mesh scheme for fluid dynamics. J Comput Phys 175(2):764–791