Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 197 (2008) 2131–2146
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions

Tim Colonius *, Kunihiko Taira

Division of Engineering and Applied Science, California Institute of Technology, CA 91125, USA

Received 21 March 2007; received in revised form 3 August 2007; accepted 6 August 2007
Available online 12 September 2007

Abstract

We report on the continued development of a projection approach for implementing the immersed boundary method for incompressible flows in two and three dimensions. Boundary forces and pressure are regarded as Lagrange multipliers that enable the no-slip and divergence-free constraints to be implicitly determined to arbitrary precision with no associated time-step restrictions. In order to accelerate the method, we further implement a nullspace (discrete streamfunction) method that allows the divergence-free constraint to be automatically satisfied to machine roundoff. By employing a fast sine transform technique, the linear system to determine the forces can be solved efficiently with direct or iterative techniques. A multi-domain technique is developed in order to improve far-field boundary conditions that are compatible with the fast sine transform and account for the extensive potential flow induced by the body as well as vorticity that advects/diffuses to large distance from the body. The multi-domain and fast techniques are validated by comparing to the exact solutions for the potential flow induced by stationary and propagating Oseen vortices and by an impulsively-started circular cylinder. Speed-ups of more than an order-of-magnitude are achieved with the new method.
© 2007 Elsevier B.V. All rights reserved.

1991 MSC: 76D05; 76M12

PACS: 47.11.+j

Keywords: Immersed boundary method; Fractional step method; Projection method; Nullspace method; Vorticity/streamfunction formulation; Far-field boundary conditions; Multi-domain method; Fast Poisson solver; Finite volume method; Incompressible viscous flow

1. Introduction

In the immersed boundary method (IB method), immersed surfaces are generated by forces at a set of Lagrangian points [29,20,19]. The flow is solved on an Eulerian grid that does not conform to the body geometry – typically a uniform Cartesian grid is used. The boundary forces that exist as singular functions along the surface in the continuous equations are described by discrete delta functions that smear (regularize) the forcing effect over the neighboring Eulerian cells.

In the original IB method, surfaces were viewed as flexible elastic membranes with a constitutive relation (*e.g.* Hooke's law) relating the forces to the motion of the Lagrangian points [28]. This technique was later extended to surfaces with prescribed motion (and in particular rigid bodies) by taking the spring constant to be large [2,16]. Goldstein et al. [9] applied the concept of feedback control to compute the force on the rigid immersed surface. The difference between the velocity solution and the boundary velocity is used in a proportional-integral controller. Constitutive laws are eliminated in the *direct forcing method* [21,7]; forcing in the momentum equation is determined by penalizing the slip at the (interpolated) surface. For the aforementioned techniques that utilize constitutive relations, the choice of gain (stiffness) is *ad hoc*. Large gain

* Corresponding author.
E-mail addresses: colonius@caltech.edu (T. Colonius), kunihiko@caltech.edu (K. Taira).

0045-7825/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2007.08.014

results in restrictions on the time step, while small gain results in slip error.[1] Direct forcing methods similarly result in a slip error at the surface. While the slip error is reported to be small [7], the magnitude cannot be estimated in a deductive manner. Further information regarding the IB method and higher-order extensions are given in a recent review [20].

An alternative is to regard the boundary forces as Lagrange multipliers whose values are chosen to satisfy the no-slip constraint [8,36]. By introducing appropriate regularization and interpolation operators and grouping the pressure and force unknowns together, the discretized incompressible Navier–Stokes equations can be formulated with a structure algebraically identical to the traditional fractional step method [36]. The pressure and force unknowns are found by solving a (modified) Poisson equation. In what follows, we refer to this method as the immersed boundary projection method (IBPM).

The principle advantages of the IBPM technique are that the continuity and no-slip constraints can be satisfied (to arbitrary accuracy) implicitly at the next time level, and that the Courant number is only limited by the choice of time marching schemes for the viscous and advection terms in the momentum equation. Further, it is possible to arrange all operations so that the method is uniformly second-order accurate in time, and so that the matrix arising from implicit treatment of the viscous terms in the momentum equation as well as the modified Poisson matrix are both symmetric and positive definite. Consequently the conjugate-gradient method can be used to solve the linear systems. However, iterative solution of the linear systems results in a convergence error. This presents no difficulty in the momentum equation where the solution need only be converged to the extent that it is smaller than other discretization errors. But in the modified Poisson equation, convergence errors directly impact the accuracy to with which the divergence-free and no-slip constraints are satisfied. While the errors can be made arbitrarily small, large numbers of iterations may be required.

In the present paper, we revisit this method and propose some improvements to accelerate the IBPM. In Section 2, we review the original formulation and present some new results from a recent extension of the method to three-dimensional flows. In Section 3, we implement a nullspace (discrete streamfunction) method [11,4] that allows the divergence-free constraint to be automatically satisfied to machine roundoff. We show that if the grid is kept uniform throughout space (with equal spacing in all directions), the Poisson-like equation for the forces can be efficiently solved either directly for stationary bodies or iteratively for moving bodies through the use of a fast sine transform. While uniform grid spacing is in fact required in the vicinity of the body by the discrete delta function that is used to regularize the surface force, it is relatively inefficient for external flows where the domain needs to extend to large distance from the body. In the original IBPM, this difficulty is overcome by stretching the mesh away from the body, but this is incompatible with the nullspace/fast sine transform formulation introduced here. To overcome this restriction, we derive in Section 4 improved far-field boundary conditions that are compatible with the fast method and allow the domain to be more snug around the body. The new boundary conditions account for the extensive potential flow induced by the body as well as vorticity that advects/diffuses to large distance from the body. The boundary conditions rely on a multi-domain approach whereby the Poisson equation is solved (with the fast sine transform) on a series of increasingly larger, but coarser, computational domains. Validation examples presented in Sections 5 and 6 demonstrate the efficacy and improved efficiency, respectively, of the revised formulation.

2. Immersed boundary projection method

2.1. Projection approach

We consider the incompressible Navier–Stokes equations with a singular boundary force $\mathbf{f}$ added to the momentum equation as a continuous analog of the immersed boundary formulation:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = -\nabla p + \frac{1}{Re}\nabla^2\mathbf{u} + \int_s \mathbf{f}(\boldsymbol{\xi}(s,t))\delta(\boldsymbol{\xi}-\mathbf{x})\mathrm{d}s, \quad (1)$$

$$\nabla\cdot\mathbf{u} = 0, \quad (2)$$

$$\mathbf{u}(\boldsymbol{\xi}(s,t)) = \int_{\mathbf{x}} \mathbf{u}(\mathbf{x})\delta(\mathbf{x}-\boldsymbol{\xi})\mathrm{d}\mathbf{x} = \mathbf{u}_{\mathrm{B}}(\boldsymbol{\xi}(s,t)), \quad (3)$$

where $\mathbf{u}$ and p are the velocity and pressure variables, respectively. Note that we express the no-slip condition using a delta function convolution along the immersed surface. Here, non-dimensionalization is performed to yield a single parameter of Reynolds number, Re. Spatial variable $\mathbf{x}$ represents position in the flow field, $\mathscr{D}$, and $\boldsymbol{\xi}$ denotes coordinates along the immersed boundary, $\partial\mathscr{B}$ having a velocity of u_{B}. The geometry of the immersed object $\mathscr{B}$ is considered to be of arbitrary shape. In the present development, there are no forces interior to the body and any motion or deformation of the body is prescribed. Further generalizations[2] of the method are possible but await future work.

The above system is discretized with a standard staggered Cartesian grid finite volume method. The mesh and variable locations are depicted in Fig. 1. The computational domain, $\mathscr{D}$, is represented by a Cartesian grid, (x_i, y_i), and the immersed boundary, $\partial\mathscr{B}$ is described by a set of Lagrangian points, (ξ_k, η_k), which can be a function of

[1] Stiffness issues are also observed with elastic surfaces. Recently, stable semi- and fully-implicit temporal discretizations to couple the velocity field and the boundary force for elastic boundaries have been proposed by [24,22].

[2] For example fully coupled fluid–structure interaction via an immersed continuum method [38].

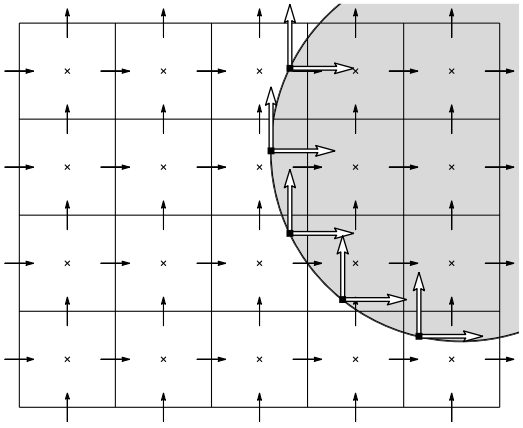

Fig. 1. Staggered grid discretization of a two-dimensional computational domain $\mathscr{D}$ and immersed boundary formulation for a body $\mathscr{B}$ depicted by a shaded object. The horizontal and vertical arrows ($\rightarrow$, $\uparrow$) represent the discrete u_i and v_i velocity locations, respectively. Pressure p_j is positioned at the center of each cell ($\times$). Lagrangian points, $\boldsymbol{\xi}_k = (\xi_k, \eta_k)$, along $\partial\mathscr{B}$ are shown with filled squares ($\blacksquare$) where boundary forces $\mathbf{f}_k = (f_{x,k}, f_{y,k})$ are applied ($\Rightarrow$, $\Uparrow$).

time. The body $\mathscr{B}$ is assumed to have a prescribed surface motion. Following the matrix–vector notation of [4], we can write Eqs. (1)–(3) semi-discretely as

$$M\frac{\mathrm{d}q}{\mathrm{d}t} + Gp - Hf = \mathscr{N}(q) + Lq + bc_1, \tag{4}$$

$$Dq = 0 + bc_2, \tag{5}$$

$$Eq = u_{\mathrm{B}}^{n+1}, \tag{6}$$

where q, p, and f are the discrete velocity flux vector, pressure, and boundary force. The discrete velocity, u, can be related to q by multiplying the cell face area normal to the vector, *i.e.*, $q = (q_{u,i},\ q_{v,i}) = (u_i\Delta y_i,\ v_i\Delta x_i)$. The above first, second, and third equations represent the discretized momentum equation, continuity equation, and no-slip condition along $\partial\mathscr{B}$. Discretized non-linear convective term $-\mathbf{u}\cdot\nabla\mathbf{u}$ is denoted by $\mathscr{N}(q)$ and operators M and L are the (diagonal) mass matrix and discrete Laplacian, respectively.

We note that all of the matrices in the above (and all that follow) are sparse and are most efficiently coded as point-operators-subroutines return the matrix–vector multiply such that the matrices are never explicitly formed. For convenience, point-operator representations (for the case of a uniform grid) are given in Appendix A.

Operators G and D are the discrete gradient and divergence operators and can be formulated such that $G = -D^{\mathrm{T}}$ [27,4]. The remaining operators of E and H are the interpolation and regularization operators resulting from the regularization of the Dirac delta functions in Eqs. (1) and (3). The no-slip constraint is enforced by equating the boundary velocity, u_{B}, to the velocity value along $\partial\mathscr{B}$ interpolated by E from the neighboring cells. On the other hand, the regularization operator smears the effect of the singular boundary force along $\partial\mathscr{B}$ to the Cartesian grid. To preserve symmetry in the final algorithm, we construct these operators to satisfy $E = -H^{\mathrm{T}}$; see [36] for further discussion. We mention that matrices G, D, E, and H are not square. Consequently, Eqs. (4)–(6) can be written as a system of algebraic equations:

$$\begin{bmatrix} A & G & -H \\ D & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{pmatrix} q^{n+1} \\ p \\ f \end{pmatrix} = \begin{pmatrix} r^n \\ 0 \\ u_{\mathrm{B}}^{n+1} \end{pmatrix} + \begin{pmatrix} bc_1 \\ bc_2 \\ 0 \end{pmatrix}. \tag{7}$$

Submatrix $A = \frac{1}{\Delta t}M - \alpha_{\mathrm{L}}L$ results from the implicit treatment of the velocity term. Here we apply the implicit trapezoid rule on the viscous term with $\alpha_{\mathrm{L}} = \frac{1}{2}$. The convective term is discretized with the second-order Adam–Bashforth (AB2) method. In this case the right-hand side vector $r^n = \left[\frac{1}{\Delta t}M + \frac{1}{2}L\right]q^n + \frac{3}{2}\mathscr{N}(q^n) - \frac{1}{2}\mathscr{N}(q^{n-1})$. The AB2 method is not self-starting and we replace it with backward Euler for the first time step. The inhomogeneous terms bc_1 and bc_2 depend on the particular boundary conditions and are discussed in [36]. Boundary conditions are discussed in greater detail in Sections 3 and 4.

With the use of staggered Cartesian grid, we are able to globally conserve mass, momentum, kinetic energy, and circulation [17,23,26]. Detailed discussion on spatial discretizations of various forms of the non-linear convective term (rotational, divergence, skew-symmetric, and advective forms) are provided in [23,26]. The explicit right-hand side term in general also includes inhomogeneous terms, bc_1 and bc_2, generated by the boundary conditions from the discrete Laplacian L and the divergence D operators, respectively.

By applying the properties of the sub-matrices, Eq. (7) can be restated as

$$\begin{bmatrix} A & G & E^{\mathrm{T}} \\ G^{\mathrm{T}} & 0 & 0 \\ E & 0 & 0 \end{bmatrix} \begin{pmatrix} q^{n+1} \\ p \\ \tilde{f} \end{pmatrix} = \begin{pmatrix} r^n + bc_1 \\ -bc_2 \\ u_{\mathrm{B}}^{n+1} \end{pmatrix}, \tag{8}$$

where $\tilde{f}$ is the boundary force with an incorporated scaling factor. This form of the equation is known as the Karush–Kahn–Tucker (KKT) system where $(p,\tilde{f})^{\mathrm{T}}$ appear as a set of Lagrange multiplier to satisfy a set of kinematic constraints. In the discretized set of equations, the constraints are purely numerical and it is no longer necessary to distinguish the pressure and boundary force. Instead we can define a combined variable $\lambda = (p,\tilde{f})^{\mathrm{T}}$ for the Lagrange multipliers and group the submatrices as $Q = [G, E^{\mathrm{T}}]$. Note that by removing the boundary force and no-slip condition along $\partial\mathscr{B}$, the traditional discretization of the incompressible Navier–Stokes equations can be retrieved.

Since we now have formulated the immersed boundary formulation of the Navier–Stokes equations in an algebraically identical manner to the traditional discretization of the incompressible Navier–Stokes equations, standard solution techniques can be utilized. Here we apply the projection (fractional-step) algorithm to Eq. (8), which can be expressed as an approximate LU decomposition of the left-hand side matrix [27], to produce the immersed boundary projection method [36]:

$$Aq^* = r_1, \quad \text{(Solve for intermediate velocity)} \tag{9}$$

$$Q^{\mathrm{T}} A_j^{\dagger} Q\lambda = Q^{\mathrm{T}} q^* - r_2, \quad \text{(Solve a modified Poisson equation)} \tag{10}$$

$$q^{n+1} = q^* - A_j^{\dagger} Q\lambda, \quad \text{(Projection step)} \tag{11}$$

where $A_j^{\dagger}$ denotes the jth order Taylor series expansion of A^{-1} with respect to Δt. The explicit terms on the right-hand side have been grouped into r_1 and r_2. In [36], A and $Q^{\mathrm{T}} A_j^{\dagger} Q$ are constructed to be symmetric positive definite operators in order to use the conjugate-gradient method to efficiently solve for the intermediate velocity and the Lagrange multipliers. In contrast to the traditional immersed boundary methods, here the no-slip condition along $\partial\mathscr{B}$ 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 accurate in time and better than first order accurate in space in the L_2 measure. Since there is no need for any constitutive relations (*e.g.*, Hooke's law [2,16] and proportional-integral controller [9]) to compute the boundary force, stiffness issues are circumvented allowing the Courant number to be limited only by the choice of time marching schemes for the viscous and convective terms.

In the case of external flow, non-uniform grid stretching is utilized to position the computational boundary $\partial\mathscr{D}$ as far as possible from the immersed body to minimize the influence of the artificial boundary conditions on the inner flow field. All boundary conditions are set to uniform flow (U_∞, 0, 0) in the streamwise direction (x-direction) except for the outflow boundary where a convective boundary condition [33],

$$\frac{\partial \mathbf{u}}{\partial t} + U_\infty \frac{\partial \mathbf{u}}{\partial x} = 0, \tag{12}$$

has been applied. Boundary conditions along the computational boundary are discussed in greater detail in Section 4 in the context of the new formulation.

2.2. Three-dimensional IBPM

Two-dimensional validation examples and convergence studies for the IBPM are presented in [36]. To demonstrate that the IBPM can be implemented in three dimensions, we briefly describe results for three-dimensional flow over a low-aspect-ratio flat plate at angle of attack. As an example, a rectangular flat plate of aspect ratio, $AR = 2$, at an angle of attack of $\alpha = 30°$ is instantaneously generated in a uniform flow field at $t = 0$. The Reynolds number is set to $Re = 100$ and the computational domain is taken to be $[-4, 6.1] \times [-5, 5] \times [-5, 5]$ (normalized by the chord) with a grid size of $125 \times 55 \times 80$ (streamwise, vertical, and spanwise directions, respectively). Here, grid stretching is applied to regions away from the plate, while keeping uniform resolution in the close proximity of the immersed body. The time step and the minimum grid size are set to 0.01 and 0.04, respectively, to limit the maximum Courant number to 0.5 during the simulation.

In Fig. 2, the spanwise vorticity contours at the midspan are compared to digital particle image velocimetry (DPIV) measurements acquired from a companion experiment performed in an oil tow tank. Simulation results and the DPIV data are found to be in agreement along with force measurements on the plate validating the three-dimensional immersed boundary projection method. The corresponding three-dimensional wake structures are presented in Fig. 3 to illustrate the formation of leading-edge, trailing-edge,

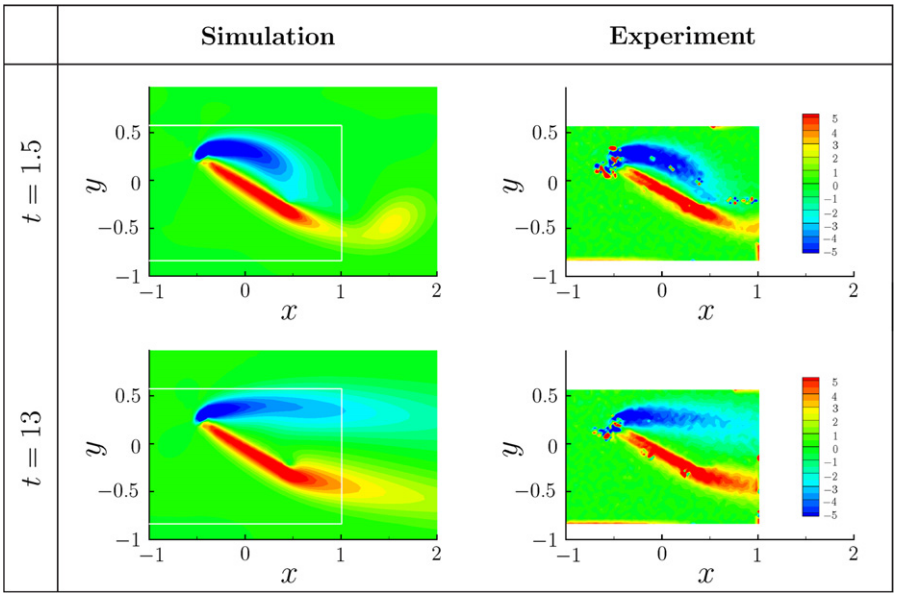

Fig. 2. Snapshots of spanwise vorticity (ω_z) profiles along the midspan ($z = 0$) at $Re = 100$ for a rectangular flat plate of $AR = 2$ and $\alpha = 30°$ based on simulations and the DPIV measurements.

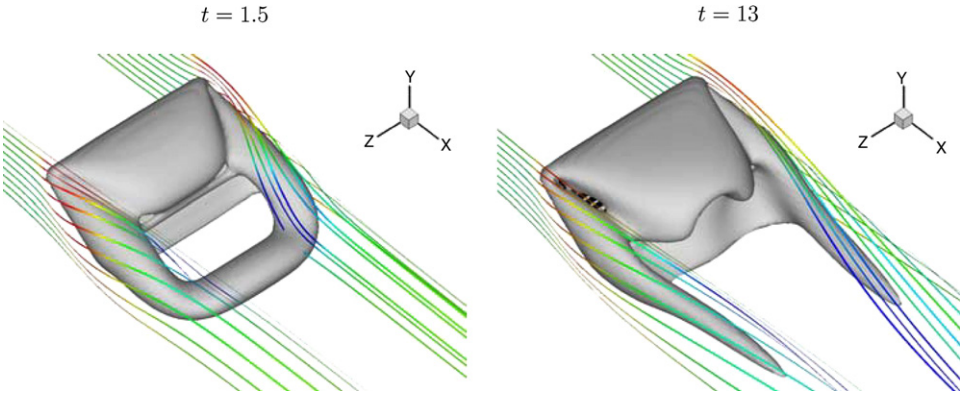

Fig. 3. Top view of vortical structure behind a rectangular plate of $AR = 2$ and $\alpha = 30°$ represented by an isosurface of $Q = 1$ for $Re = 100$ at different times. Streamlines are overlaid with color contour indicating the local velocity norm from blue to red in increasing magnitude. Flow direction from top left to bottom right. (For interpretation of the references to colour in this figure legend, the reader is referred to the conversion of this article.)

and tip vortices. The isosurface here are generated for unit Q-value (second invariant of the velocity gradient tensor) to show flow regions with significant rotation.[3] Streamlines are also depicted to illustrate the tip-effects. Initially a strong trailing-edge vortex is formed convecting downstream while the leading-edge and tip vortices stay stably attached to the plate ($t = 1.5$). Later at steady-state ($t = 13$), the diffused leading-edge vortical structure is still stably attached to the plate. In the case of three-dimensional flow, the viscous diffusion of vorticity in the spanwise direction and the tip-effect stabilizes the wake structure at this low Re. Results with various aspect ratio, angles of attack, and planform geometries are examined in further detail in [37].

3. Nullspace method for the immersed boundary method

3.1. Nullspace approach

The *nullspace* or *discrete streamfunction* approach [11,4] is a method for solving the system (7) without the immersed boundary formulation. In this case, the flow only needs to satisfy the incompressibility constraint, which leads us to the use of discrete streamfunction, s, such that

$$q = Cs, \tag{13}$$

where C represents the discrete curl operator. This operator is constructed with column vectors corresponding to the basis of the nullspace of D. Chang et al. [4] should be consulted for details. Hence, these operators enjoy the following relation:

$$DC \equiv 0, \tag{14}$$

which automatically enforces incompressibility at all time; $Dq^{n+1} = DCs^{n+1} = 0$. This discrete relation is consistent with the continuous version of the vector identity: $\nabla \cdot \nabla \times \equiv 0$.[4]

Pre-multiplying the momentum equation with C^{T}, the pressure gradient term can also be removed from the formulation since $C^{\mathrm{T}}Gp = -(DC)^{\mathrm{T}}p = 0$, resulting in only a single equation to be solved for each time step:

$$C^{\mathrm{T}}ACs^{n+1} = C^{\mathrm{T}}(r_1^n + bc_1). \tag{15}$$

In this method, the most computationally expensive component of the fractional step method, namely the pressure Poisson solver, is eliminated while the continuity equation is exactly satisfied. Moreover the fractional step error arising from using an approximate A^{-1} is not present since an approximate LU decomposition is not required. This feature led Chang et al. [4] to call this technique the *exact fractional step method*.

We note that the operator C^{T} is another discrete curl operation, and that:

$$\gamma = C^{\mathrm{T}}q, \tag{16}$$

is a second-order accurate approximation to the circulation in each dual cell (vorticity multiplied by the cell area normal to the vorticity component).

This method may in general be used on unstructured meshes in two and three dimensions [4], including, as a special case, the simple Cartesian mesh used in IB methods. In two dimensions, the discrete streamfunction and circulation have a single component (in the direction normal to the plane), which is naturally defined at the cell *vertices* (see Fig. 4) [4]. In three dimensions there are three components of the streamfunction and circulation that are defined at the centers of the edges of the Voronoi (dual) cell, analogously to the velocity components on the primal mesh.

[3] The Q-value (the second invariant of $\nabla \mathbf{u}$) is defined as $Q \equiv \frac{1}{2}(\|\mathbf{\Omega}\|^2 - \|\mathbf{S}\|^2)$, for incompressible flow where $\mathbf{\Omega}$ and $\mathbf{S}$ are the asymmetric and symmetric components of $\nabla \mathbf{u}$, respectively [13]. Compared to the vorticity norm, positive Q-values can highlight vortical structures by removing regions of high shear.

[4] Note that we have set $bc_2 = 0$ which is the case for the boundary conditions we consider here. More general situations that require $bc_2 \neq 0$ can be handled by finding a particular solution for the inhomogeneous vector and adding the solution to Eq. (13).

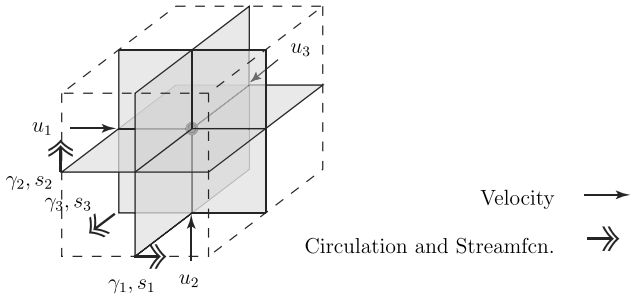

Fig. 4. Location of variables on staggered 3D mesh. Velocity components are defined at the center of each edge. Streamfunction and circulation are defined similarly for the Voronoi cell – in this case a cell that is offset by half a cell length in each direction.

3.2. Nullspace approach with an immersed boundary

In order to satisfy both the incompressibility and the no-slip conditions with the nullspace technique, it would be necessary to derive a basis for the nullspace of Q^{T}. Although, a singular value decomposition of Q^{T} can be performed to numerically determine the nullspace, the result is not in general a sparse representation which is desirable for computational feasibility. An analytical derivation of the nullspace operator does not seem to be an easy task either. Moreover, in the general case where the body is moving, the nullspace representation would need to be recomputed at least once per timestep.

To circumvent this difficulty, we once again rely on a projection approach. Consider the system that is obtained by incorporating C^{T} and $q^{n+1} = Cs^{n+1}$ to Eq. (8). The incompressibility constraint and the pressure variable are eliminated and we arrive at another KTT system:

$$\begin{bmatrix} C^{\mathrm{T}}AC & C^{\mathrm{T}}E^{\mathrm{T}} \\ EC & 0 \end{bmatrix} \begin{pmatrix} s^{n+1} \\ \tilde{f} \end{pmatrix} = \begin{pmatrix} C^{\mathrm{T}} r_1^n \\ u_{\mathrm{B}}^{n+1} \end{pmatrix}. \tag{17}$$

The left-hand side matrix is symmetric but in general indefinite, making a direct solution less efficient. The projection (fractional step) approach mimics Eqs. (9)–(11), and we obtain

$$C^{\mathrm{T}}ACs^* = C^{\mathrm{T}} r_1^n, \tag{18}$$

$$EC(C^{\mathrm{T}}AC)^{-1}(EC)^{\mathrm{T}}\tilde{f} = ECs^* - u_{\mathrm{B}}^{n+1}, \tag{19}$$

$$s^{n+1} = s^* - (C^{\mathrm{T}}AC)^{-1}(EC)^{\mathrm{T}}\tilde{f}, \tag{20}$$

where we have as not yet inserted an approximation for the inverse of $C^{\mathrm{T}}AC$. Direct solution of this system in the general case requires a nested iteration to solve the modified Poisson equation. This may be feasible in general (a rough operation count indicate that the work is similar to Eqs. (9)–(11)). In the case where the body is not moving, it is moreover possible to perform a Cholesky decomposition of $EC(C^{\mathrm{T}}AC)^{-1}(EC)^{\mathrm{T}}$ once and for all, since the dimension of the system scales with the number of forces for the immersed boundary. In this case a system of equations of the form $C^{\mathrm{T}}ACx = b$ need be solved once for each Lagrangian force at the beginning of the computation.

3.3. Fast method for uniform grid and simple boundary conditions

In this section we revert to the semi-discrete momentum equation,

$$M\frac{\mathrm{d}q}{\mathrm{d}t} + Gp + E^{\mathrm{T}}\tilde{f} = \mathcal{N}(q) + Lq + bc_1, \tag{21}$$

where symbols are as defined previously. The divergence free and no-slip constraints are unchanged.

We now show that with simplification, a similar system to Eqs. (9)–(11) may be solved using fast sine transforms, resulting in a significant reduction in computational work. When the grid is uniform (with equal grid spacing in all coordinate directions), the mass matrix M is the identity matrix. We assume for the moment that the values of the velocity are known in the region outside the computational domain. We apply simple Dirichlet boundary conditions to the velocity normal to the sides/edges of the computational domain, and a Neumann boundary condition to the velocities tangent to the sides. Lacking further information, one could specify, for example, a no-penetration BC for the normal component of velocity and a zero vorticity (or no-stress) condition for the remaining tangent components. These are natural boundary conditions for an external flow around the body, provided the domain is large. In the next section we will show how improved estimates for the velocities outside the computational domain can be obtained via a multi-domain approach.

With these simplifications we operate on Eq. (21) with C^{T} (which eliminates the pressure) and we obtain

$$\frac{\mathrm{d}\gamma}{\mathrm{d}t} + C^{\mathrm{T}}E^{\mathrm{T}}\tilde{f} = -\beta C^{\mathrm{T}}C\gamma + C^{\mathrm{T}}\mathcal{N}(q) + bc_\gamma. \tag{22}$$

In deriving this equation we have used that $Lq = -\beta CC^{\mathrm{T}}q = -\beta C\gamma$ provided that $Dq = 0$. Here β is a constant equal to $1/(Re\Delta^2)$, where Δ is the uniform grid spacing. This identity mimics the continuous identity $\nabla^2\mathbf{u} = \nabla(\nabla\cdot\mathbf{u}) - \nabla\times\nabla\times\mathbf{u} = -\nabla\times\nabla\times\mathbf{u}$.

With uniform grid and the aforementioned boundary conditions, the matrix $-\beta C^{\mathrm{T}}C$ is the standard discrete Laplacian operator on a 5- or 7-point stencil in two and three spatial dimensions, respectively. The boundary conditions discussed above result in zero Dirichlet boundary conditions for γ. This discrete Laplacian is diagonalized by a sine transform that can be computed in $\mathcal{O}(N\log_2 N)$ operations (where N is the dimension of γ) [30]. We denote here the sine transform pair:

$$\hat{\gamma} = S\gamma \leftrightarrow \gamma = S\hat{\gamma}, \tag{23}$$

where the circumflex denotes the Fourier coefficients. In writing the transform pair, we have used the fact that the sine transform can be normalized so that it is identical to its inverse. Further, we may write symbolically $\Lambda =$

$SC^{\mathrm{T}}CS$, where Λ is a diagonal matrix with the eigenvalues of $C^{\mathrm{T}}C$. These are positive and known analytically (*e.g.* [30]), and we note that there is no zero eigenvalue (since the boundary conditions are Dirichlet).

Applying the same time-marching schemes used previously we obtain the transformed system:

$$S\left(I+\frac{\beta\Delta t}{2}\Lambda\right)S\gamma^{*} = \left(I-\frac{\beta\Delta t}{2}C^{\mathrm{T}}C\right)\gamma^{n} + \frac{\Delta t}{2}\left(3C^{\mathrm{T}}\mathcal{N}(q^{n})-C^{\mathrm{T}}\mathcal{N}(q^{n-1})\right) + \Delta t b c_{\gamma}, \tag{24}$$

$$EC\left(S\Lambda^{-1}\left(I+\frac{\beta\Delta t}{2}\Lambda\right)^{-1}S\right)(EC)^{\mathrm{T}}\tilde{f} = ECS\Lambda^{-1}S\gamma^{*}-u_{\mathrm{B}}^{n+1}, \tag{25}$$

$$\gamma^{n+1} = \gamma^{*}-S\left(I+\frac{\beta\Delta t}{2}\Lambda\right)^{-1}S(EC)^{\mathrm{T}}\tilde{f}. \tag{26}$$

The velocity, needed for the next time step, may be found by introducing the discrete streamfunction:

$$q^{n} = Cs^{n}+bc_{q}, \quad s^{n} = S\Lambda^{-1}S\gamma^{n}+bc_{s}. \tag{27}$$

Each of the vectors $bc_{\gamma,s,q}$ involves the assumed known values of velocity at the edge of the computational domain. Their values are discussed in detail in the next section.

In the new system of equations, only one linear system need be solved, Eq. (25), with a positive definite left-hand side operator. That the matrix is positive definite can be seen by inspection. The dimensions of the matrix are now $N_f \times N_f$, and thus *many* fewer iterations are required than the original modified Poisson equation, Eq. (10). To be more precise, each iteration on Eq. (25) requires $\mathcal{O}(N(2\log_2 N + N_{bw} + 4d))$ operations, where N is the number of vorticity unknowns and N_{bw} is the bandwidth of the body-force regularization/interpolation operators,[5] and d is the dimensionality of the flow (2 or 3 for 2D or 3D, respectively). For the discrete Delta function with a support of 3Δ, we have $N_{bw} = 3d$. For the original Poisson equation, Eq. (10), the cost per iteration is $\mathcal{O}(N \times (N_{bw} + (2d+1)j) + 4d)$, where j is the order of the approximate Taylor-series inverse of A and the factor $2d+1$ is the stencil of the discrete Laplacian. Furthermore, using standard estimates for the number of iterations required for convergence of the conjugate-gradient method [35] along with the known eigenvalues of $C^{\mathrm{T}}C$, we can estimate that the operation count per time step for the Poisson solution has been reduced from[6]

$\mathcal{O}(N^{1/2}N(7d+(2d+1)j))$ operation count for Eq. (10)

to

$\mathcal{O}(N_f^{1/2}N(2\log_2 N + 7d))$ operation count for Eq. (25).

For example, in a three-dimensional case with $N = 128^3$, $N_f = 10^3$, $d = 3$, and $j = 3$ the estimated speedup is about 30. For a two-dimensional case with $N = 128^2$, $N_f = 200$, $d = 2$ and $j = 3$, the speedup is about 10. This is for the Poisson solve alone. Additional speedup occurs because it is no longer necessary to solve a system $A\ x = b$ for the momentum equation. Numerical experiments in Section 6 for the two-dimensional case confirm at least the order-of-magnitude of the speedup (the actual speedup is faster than predicted). Finally, we recall that the new system of equations results in no iterative error in satisfying the divergence-free constraint (it is automatically zero to round-off).

If the body is stationary, then the Poisson-like equation for the forces can be efficiently solved using a triangular Cholesky decomposition. This results in a vastly lower work per time-step, since the operation count for the Poisson solve is simply $\mathcal{O}(N_f^2)$. In this case the computational speed is limited only by the solution of Eq. (24).

To summarize, if the grid is uniform and simple boundary conditions are used, it is vastly preferable to solve Eqs. (24)–(26). We refer to this in what follows as the *fast* method. Unfortunately, for external flows, the simplified boundary conditions are not effective unless the computational domain is quite large. Since the grid is also required to be uniform, even far away from the body, the larger domain would quickly negate the benefit of fast method. However, in the next section we discuss an alternative strategy for implementing boundary conditions in the fast method that has a more modest cost penalty.

4. Far-field boundary conditions: a multi-domain approach

The fast method relies on simplified far-field boundary conditions, namely known velocity normal to the boundary and known vorticity. These can be set to zero if the computational domain is sufficiently large. For smaller domains, this will lead to significant errors and, in particular, the forces computed on the body will suffer a significant *blockage* error. The error arises from two sources. The first is the extensive, algebraically decaying potential flow induced by the body (or equivalently, the system of forces). The second is that vorticity may advect or diffuse through the boundary. In our original method discussed in Section 2, these errors are minimized by using a large domain with a highly stretched Cartesian mesh near the far-field boundaries (but retaining uniform grid spacing near the body), as well as by using an approximate convective outflow boundary condition. Unfortunately, stretched meshes are incompatible[7] with direct Fourier methods for solution of the Poisson equation. In this section, we show how to pose an accurate

[5] We have used the fact that $N_f \ll N$ in arriving at the estimate.

[6] Here the factors $N^{1/2}$ or $N_f^{1/2}$ are the estimated number of iterations of the conjugate gradient solver, the $2N\log_2 N$ factor comes from two (fast) sine transforms, the $(2d+1)j$ factor from the Laplacian, and $7d$ from the interpolation, regularization, C^{T}, and C operations together.

[7] In certain special circumstances stretched meshes can be combined with Fourier-transform methods for elliptic equations, *e.g.* [3].

far-field boundary condition that is also compatible with the fast method described in the last section.

We start by briefly reviewing relevant boundary conditions designed to reduce one or both of the aforementioned errors. For errors associated with the slowly decaying potential flow, a few techniques have been posed in the past to *patch in* the potential flow extending from the truncated computational boundary to infinity. Rennich and Lele [31] propose a technique for two unbounded directions and one periodic direction. Their method is based on matching the numerical solution to analytical representation of the solution to Laplace equation outside a cylindrical volume. They report a 50% increase per time step for a typical large-scale computation, but this cost is more than offset by the ability to use much more compact domains. Wang [39] presents a similar approach for two-dimensional flow in the form of a correction to a trial solution that satisfies an incorrect Dirichlet boundary condition. Vortex particle methods in principle automatically account for the extensive potential flow generated by the vorticity. However, in practice it is often necessary to remove particles that advect to large distance from the region of interest. An interesting technique to reduce errors associated with removal of particles is called *merging*, whereby the circulations of several vortex particles are combined into a single element when they are sufficiently far from the body [34,32].

The second type of error associated with vorticity advecting or diffusing through the boundary is typically handled by posing *outflow* boundary conditions. For incompressible flow these are usually called *convective boundary conditions*, whereas in compressible flow the term non-reflecting boundary condition is often used. Another technique is to selectively apply damping in a region near the computational boundary. Methods that employ this technique vary from *ad hoc* specification of layer width, damping strength, *etc.*, to techniques that theoretically specify the damping parameters according to a model. An example is the *perfectly matched layer* [1] for linear wave equations (including linearized compressible Euler equations [12]) that uses analytical solutions to the governing equations to derive damping terms that prevent reflection of waves from the interface. Another technique called *super-grid* [6] is based on an analogy with turbulence modeling – that the effect of the turbulence model is to model scales too fine to be resolved in the computational mesh, whereas the effect of the boundary condition is to model scales too large to be resolved in the computational domain. A full discussion of these techniques is beyond the scope of this paper; we refer the reader to some recent references for further details [33,15,25,5]. These techniques are designed to remove vorticity from the domain as smoothly as possible thereby preventing undesirable reflections or aliasing. Most do not account for the velocity induced by vorticity that has already exited the domain (a non-local effect).

We present here an alternative technique that shares some features with these previous methods, especially those of [31,34,6]. It is based on a multi-domain approach that also shares some operations with the multigrid method for solving elliptic equations. We first describe the method in words. The basic idea is to consider the domain as embedded in a larger domain but with a coarser mesh. The circulation on the inner (smaller, finer) mesh is then interpolated or *coarsified* onto the outer (larger, coarser) mesh. The Poisson equation is solved (with zero boundary conditions) on the outer domain. This solution is then interpolated along the boundary of the inner mesh and the Poisson equation is solved, with the "corrected" boundary value specified, on the inner mesh.

Similar to the vortex merging method discussed above, any existing circulation in the outer portion of the larger domain is retained from the previous time level. In this way, we approximately account for circulation that has advected or diffused out of the inner domain. Clearly, the solution on the coarser mesh contains a larger truncation error for the evolution of this vorticity. However, inversion of the Laplacian is a *smoothing* operation. High frequency components of the solution induced by circulation in the outer mesh decay more rapidly than low-frequency components. Being interested in the flow in the vicinity of the body (and its wake), we discard the solutions in the outer region only retaining the velocity it induces on the inner domain.

We apply this technique recursively a number of times, enlarging (and coarsening) the domain in each grid level. We choose to keep the total number of grid points in each direction fixed on each mesh; we magnify the domain and coarsen the grid by a factor of 2 at each grid level. The procedure is shown schematically in Fig. 5. The vorticity is repeatedly coarsified on each progressive grid. The Poisson equation is then solved on the largest domain, in turn providing a boundary condition for the next smaller domain. The process is then repeated until we return to the original domain.

The velocity field decays algebraically in the far-field and we thus expect errors associated with the boundary condition on the largest domain to decrease geometrically as the size of the largest domain is increased. In the worst case of a two-dimensional flow with non-zero total circulation, the velocity decays with the inverse of the distance to the vortical region. Analytical estimates given in Appendix B show that we obtain a factor of 4 reduction in the boundary error with each progressively larger grid. This, of course, is what would be obtained by simply extending the original grid to a distance equal to the extent of the largest grid, but due to the coarsening operation, the cost increases linearly with increasing extent, rather than quadratically (in two dimensions) or cubically (in three dimensions).

The method can thus be written as follows. We define the domain of each grid as $\mathscr{D}^{(k)}$, $k = 1, 2, \ldots, N_g$, where $k = 1$ refers to the original (smallest) grid and $k = N_g$ refers to the largest one. We then define the multi-domain inverse Laplacian

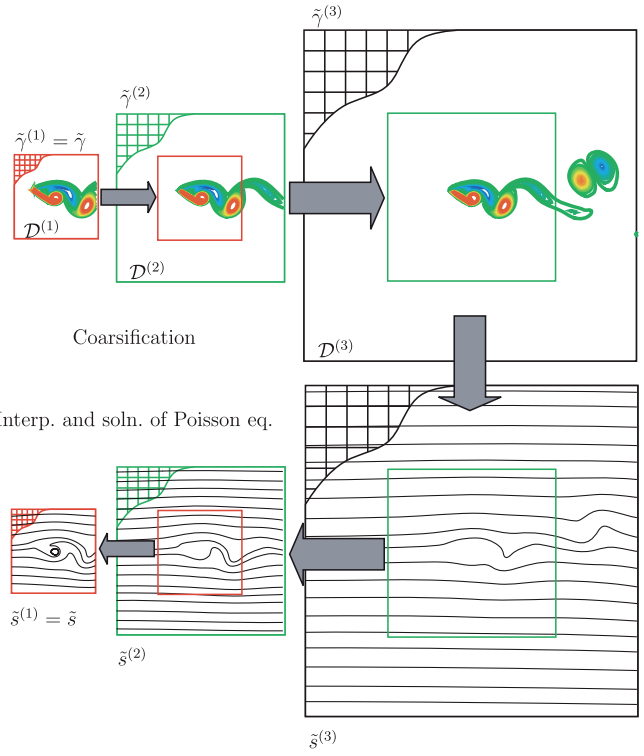

Fig. 5. Schematic of 3-level multi-domain solution of the Poisson equation.

$$\tilde{s} = S\overline{\Lambda^{-1}}S\tilde{\gamma}, \tag{28}$$

where $\tilde{\gamma}$ is an arbitrary input vector (with length equal to the number of discrete circulation values on the grid), $\tilde{s}$ is the solution (with length equal to the number of discrete streamfunction values), and the operator $S\overline{\Lambda^{-1}}S$ implies the following operations:

$$\tilde{\gamma}^{(1)} = \tilde{\gamma}, \tag{29}$$

$$\tilde{\gamma}^{(k)} = \begin{cases} \tilde{\gamma}^{(k)} & \text{where } x \in \mathscr{D}^{(k)} \setminus \mathscr{D}^{(k-1)}, \\ P^{(k-1)\to(k)}(\tilde{\gamma}^{(k-1)}) & \text{where } x \in \mathscr{D}^{(k-1)}, \\ & k = 2,3,\ldots,N_g, \end{cases} \tag{30}$$

$$\tilde{s}^{(N_g+1)} = 0, \tag{31}$$

$$\tilde{s}^{(k)} = S\Lambda^{-1}S\tilde{\gamma}^{(k)} + bc_s[P^{(k+1)\to(k)}(\tilde{s}^{(k+1)})], \quad k = N_g, N_g-1,\ldots,1, \tag{32}$$

$$\tilde{s} = S\overline{\Lambda^{-1}}S\tilde{\gamma} = \tilde{s}^{(1)}. \tag{33}$$

Here $P^{(k-1)\to(k)}$ is a fine-to-coarse interpolation operator and $P^{(k)\to(k-1)}$ is its coarse-to-fine counterpart restricted to $\partial\mathscr{D}^{(k-1)}$ by bc_s.

In constructing P, it would be desirable to preserve (to machine roundoff) certain moments of the circulation distribution so that the velocity decay rate far from the body is correct. In the present implementation, we attempt to preserve only the total circulation. Switching from matrix/vector to point-operator notation, we write, for the two-dimensional case,

$$\begin{aligned} P^{(k-1)\to(k)}(\tilde{\gamma}^{(k-1)})_{2i,2j} = {} & \tilde{\gamma}^{(k-1)}_{i,j} + \frac{1}{2}\tilde{\gamma}^{(k-1)}_{i-1,j} + \frac{1}{2}\tilde{\gamma}^{(k-1)}_{i+1,j} \\ & + \frac{1}{2}\tilde{\gamma}^{(k-1)}_{i,j-1} + \frac{1}{2}\tilde{\gamma}^{(k-1)}_{i,j+1} + \frac{1}{4}\tilde{\gamma}^{(k-1)}_{i-1,j-1} \\ & + \frac{1}{4}\tilde{\gamma}^{(k-1)}_{i+1,j-1} + \frac{1}{4}\tilde{\gamma}^{(k-1)}_{i-1,j+1} \\ & + \frac{1}{4}\tilde{\gamma}^{(k-1)}_{i+1,j+1}. \end{aligned} \tag{34}$$

The 9-point stencil leads to a conservation of the total circulation and is second-order accurate based on a Taylor-series expansion. We note that the coefficients in Eq. (34) sum to 4 since the circulation in the (dual) cell is the vorticity multiplied by the area, and coarsifying the grid by a factor of 2 results in a factor of 4 increase in cell area. The three-dimensional version of Eq. (34) consists of averaging Eq. (34) over two adjacent (i, j) planes of data normal to the vorticity component, for each of the three components.

For the coarse-to-fine interpolation at the boundary of the next-finer mesh, we use the value from the coarser mesh for those grid points that coincide, and a mid-point linear interpolation (again second-order accurate) for those points in between.

We note that circulation is only strictly preserved if there is no vorticity advecting or diffusing out of the original domain. During vorticity transfer from fine to coarse mesh, circulation is only preserved to the level of discretization error, since the discretization error is different on each mesh and advection and diffusion rates are therefore slightly different. Tests below confirm that changes in circulation as structures pass between the different domains are appropriately small.

Utilizing the multi-domain description of the circulation and solution of the Poisson equation, we now write the overall system of equations to be solved at each time-step.

$$\begin{aligned} & S\left(I + \frac{\beta\Delta t}{2}\Lambda\right)S\gamma^{(k)^*} \\ & \quad = \left(I - \frac{\beta\Delta t}{2}C^{\mathrm{T}}C\right)\gamma^{(k)^n} + \frac{\Delta t}{2}(3C^{\mathrm{T}}\mathscr{N}(q^{(k)^n}) - C^{\mathrm{T}}\mathscr{N}(q^{(k)^{n-1}})) \\ & \qquad + \frac{\Delta t}{2}bc_\gamma([P^{(k+1)\to(k)}(\gamma^{(k+1)^*})] + [P^{(k+1)\to(k)}(\gamma^{(k+1)^n})]), \\ & \quad k = N_g, N_{g-1},\ldots,1, \end{aligned} \tag{35}$$

$$EC\left(S\overline{\Lambda^{-1}}\left(I + \frac{\beta\Delta t}{2}\Lambda\right)^{-1}S\right)(EC)^{\mathrm{T}}\tilde{f} = ECS\overline{\Lambda^{-1}}S\gamma^{(1)^*} - u_{\mathrm{B}}^{n+1}, \tag{36}$$

$$\gamma^{n+1} = \gamma^{(1)^*} - S\left(I + \frac{\beta\Delta t}{2}\Lambda\right)^{-1}S(EC)^{\mathrm{T}}\tilde{f}, \tag{37}$$

$$s^{n+1} = S\overline{\Lambda^{-1}}S\gamma^{n+1}. \tag{38}$$

Note that in solving for the streamfunction at the next time step, Eq. (38), we save the coarsified circulation fields and streamfunctions to use on the right-hand side of Eq. (35) at the next time step.

When vorticity crosses the boundary of a given grid level, the $\gamma^{(j)}$ fields are not necessarily smooth across the interfaces, especially at the coarsest levels. The propagation of a vortex through mesh levels is examined in the next section and it is possible to see some slight internal reflections of the local circulation near the boundary. However, the errors remain confined to a small region near the boundary and diffused over time by the physical viscosity.

The multi-domain technique comes with a significant increase in computational expense. Since we now solve the intermediate vorticity equation each Poisson equation N_g times, the operation count goes up by a factor of N_g. Nevertheless, it enables us to utilize the fast algorithm described in the previous section. Moreover, we find that the multi-domain is sufficiently accurate that computational domain can be made snug around the body. Run times for particular examples are discussed below.

We note that in many situations, it is desirable to specify a uniform flow about a body. This is simple to accomplish in the nullspace formulation, as there is no circulation associated with it. One need only add the uniform flow to q^n resulting from Eq. (27) and to u_B^{n+1} in Eq. (36). In principle one could add *any* potential flow in this way, at least provided it satisfies the *discrete* Poisson equation with zero right-hand side.

5. Validation examples

5.1. Velocity field for an Oseen vortex

The two-dimensional velocity field associated with a Gaussian distribution of vorticity (Oseen vortex) is computed with the multi-domain boundary conditions. This test is used to validate the methodology since it is possible to derive analytically the expected improvement in multi-domain solution with increasing N_g for this case. As discussed above, the largest domain uses no penetration/no stress boundary conditions. An analytical solution for the velocity field with these boundary conditions may be constructed by the method of images such that the expected error for the multi-domain boundary conditions can be evaluated. The procedure is straightforward and is described in Appendix B. The results show that the error should decrease as 4^{-N_g} in general, and for the special case of a square domain, the rate improves to 16^{-N_g}.

The vorticity field is initialized with

$$\omega(x,y) = \frac{\Gamma}{4\pi\nu t} e^{-\frac{r^2}{4\nu t}}, \tag{39}$$

where $r = \sqrt{x^2+y^2}$ is the distance from the origin. The analytical solution for the azimuthal velocity is

$$u_\theta(x,y) = \frac{\Gamma}{2\pi r}\left(1 - e^{-\frac{r^2}{4\nu t}}\right). \tag{40}$$

We start the computation at time $t = t_0$ and choose Γ and t_0 such that the maximum speed is U at $r = R$. In what follows, all lengths and velocities are normalized by R and U, respectively. The vorticity is evaluated at the vertices of a rectangular domain with uniform (and equal) grid spacing in both directions and the Poisson equation is solved using the multi-domain method discussed above. In Fig. 6, contours of the velocity in the x direction are plotted for a case with $N_g = 5$; the velocity computed on each of the five domains are overlaid to show that the velocity field remains smooth through the domain transitions. In Fig. 7, the L_1 error of u (the entire discrete velocity field) is plotted as N_g is varied from 1 to 5, for two different computational domains. For the rectangular domain extending to ± 4 and ± 8 in the x and y directions, respectively, the decay follows the 4^{-N_g} theoretical estimate through $N_g = 5$. For the square domain extending to ± 5 in each direction, we observe the 16^{-N_g} decay down to errors around 10^{-3} which can be shown to be roughly the level of the truncation error for the second-order finite volume method at this grid density. For the non-square domain, we require about $N_g = 5$ to reduce the boundary condition error to a similar level.

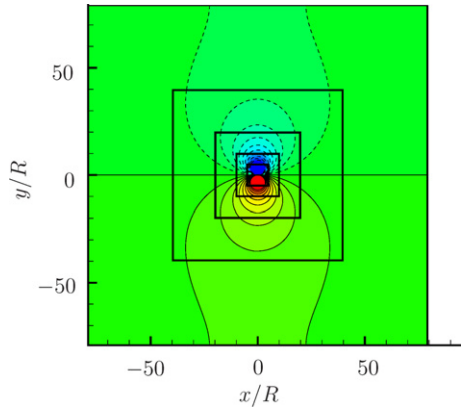

Fig. 6. Multidomain solution of the Poisson equation with $N_g = 5$ for an Oseen vortex. Contours of the velocity component in the x direction are plotted for each of the 5 grids. The smallest grid extends to $\pm 5R$, with grid spacing $\Delta = 0.05R$. Contour levels: min $= -0.2$, max $= 0.2$, increment $= 0.02$.

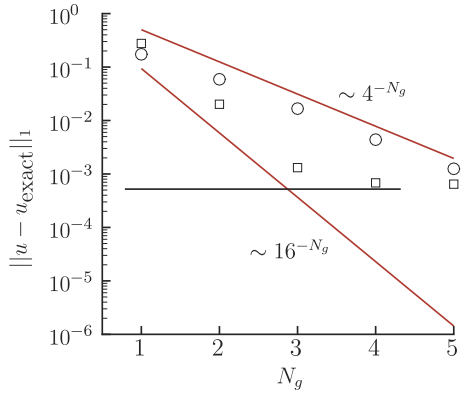

Fig. 7. L_1 error in u with multidomain solution of Poisson equation with increasing N_g for the Oseen vortex. The horizontal black line shows the approximate level of the truncation error.

5.2. Propagation of an Oseen vortex

In order to evaluate errors associated with vorticity advecting/diffusing through the computational boundary, we again use the analytical solution associated with an Oseen vortex. The vortex is initialized at $(x,\ y) = (0,\ 0)$ and advected by an otherwise uniform flow with speed equal to the maximum velocity of the vortex. The vorticity and azimuthal velocity are still given by Eqs. (39) and (40), respectively, but with the radius, r redefined with $r = \sqrt{(x - Ut)^2 + y^2}$. Again, Γ and the initial time, t_0 are set so that at $t = t_0$, the maximum speed associated with the vortex alone is U and occurs at $r = R$. Again we set $Re = 300$.

Fig. 8 shows the error in the velocity at the origin for a domain that nominally extends to $\pm 5R$ with $\frac{\Delta}{R} = 0.05$. Since the velocity decays like $1/r$, it has a long-range effect. To achieve less than 1% error without corrected boundary conditions, the domain would need to extend to $\pm 100R$. The error is initially zero (even with the uncorrected boundary conditions) due to symmetry. As time progresses, the error increases and reaches 25% for $N_g = 1$. This occurs as the vortex propagates through the right boundary of the domain. With $N_g > 1$, the vortex is progressively transferred to the next largest mesh at intervals of time $5 \times 2^{n-1}$, $n = 1, \ldots, N_g$. With $N_g = 5$, the error stays below 1% up to non-dimensional time 80, when it leaves the coarsest, largest mesh. There are small oscillations in the error evident during grid-to-grid transfer times. The associated total circulation changes by at most 5% during these transfers. With $N_g = 10$, error from the boundary condition is undetectable up to time 100 and the error is controlled by the second-order discretization error and stays below about 0.2%. The solution at time 100 is shown in Fig. 9 on the largest mesh. The magnified region is shown as in an inset and shows contours of the vorticity and normal velocity. By time 100, the vortex would have physically diffused to a core size of about $1.6R$, whereas the grid spacing on the largest domain is $12.8R$! The velocity field near the core is completely wrong, but the circulation is nearly conserved and this induces the correct potential flow far from the core. The physical (smallest) domain is also depicted on the plot and, as is shown in Fig. 8, the error at the origin is still less than about one percent of the correct value at that time.

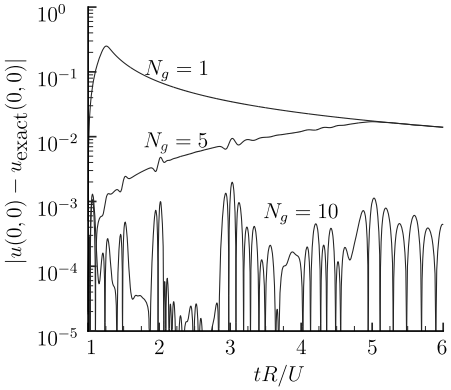

Fig. 8. Error in normal velocity at the origin for propagating Oseen vortex with $N_g = 1$, 5, and 10.

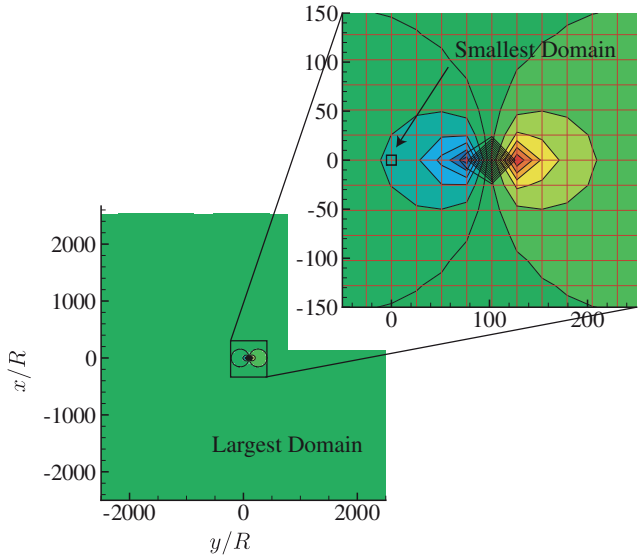

Fig. 9. Propagating Oseen vortex at $t = 100R/U$ on the largest domain with $N_g = 10$. Color contours represent the normal velocity. The inset shows a zoomed region near the vortex. The black lines are contours of the circulation which is represented on only a few grid points of the largest mesh at this time. (For interpretation of the references to colour in this figure legend, the reader is referred to the conversion of this article.)

5.3. Potential flow over a cylinder

As a final example, we consider the potential flow induced at $t = 0^+$ by an impulsively started cylinder of diameter D. The immersed boundary uses 571 equally spaced Lagrangian points and the domain is defined snugly around the body, extending to $\pm 0.55D$ in each direction with grid spacing $\Delta = 0.0055D$. We initiate a uniform flow

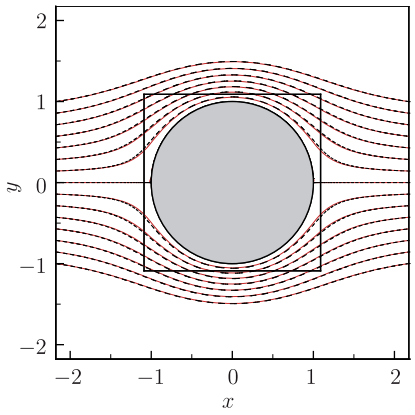

Fig. 10. Streamlines around a circular cylinder for potential flow for $N_g = 4$ with solutions from the first two inner multi domains shown. Present result (——) and the exact solution (– – – –) are presented.

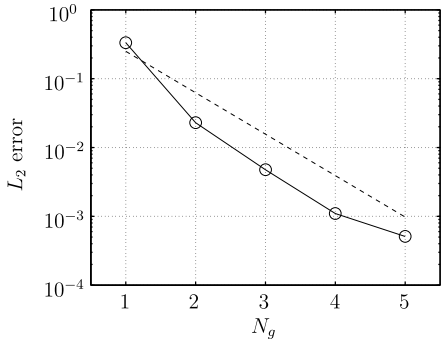

Fig. 11. Velocity error along the top boundary of the smallest domain for different N_g (○). A guide of 4^{-N_g} is also shown (– – – –).

with speed U and let the body "materialize" at $t = 0$. The solution is obtained by performing 1 time-step of the Navier–Stokes solution using the fast method with multi-domain boundary conditions. A flow field obtained with $N_g = 4$ is presented with the exact potential flow solution in Fig. 10. The streamlines are found to be in agreement with a slight difference near the immersed boundary due to the regularized nature of the discrete delta function. In Fig. 11, we compare the exact potential flow solution to the numerical solution along the top boundary of the innermost domain for different N_g. We observe the estimated $\mathcal{O}(4^{-N_g})$ convergence (see Appendix B) down to a level of about 10^{-3} after which the leading-order error is dominated by the truncation error arising from the discrete delta functions at the immersed boundary and the discretization of the Poisson equation.

6. Performance of the fast method

We conclude by measuring the performance of the fast nullspace/multi-domain immersed boundary method compared to the original performance by the IBPM. First, we simulate flows over a stationary circular cylinder of diameter D and compare to previously published results [18,36]. Computations are performed on the domain $\mathscr{D}^{(1)} = [-1, 3] \times [-2, 2]$ with $\Delta = 0.02D$ where N_g is varied between 1 and 5. The cylinder is centered at the origin. The flow is impulsively started at $t = 0$, and the body is stationary. Thus the Cholesky decomposition is used to solve Eq. (36).

After transient effects associated with the impulsively-started flow have died away, we examine wake structures and forces on the cylinders from for different values of N_g. These are compared with previous results for $Re = 40$ and *200* in Tables 1 and 2, respectively. For the steady flow at $Re = 40$ we report characteristic dimensions of the recirculation bubble in the wake, and for the unsteady flow at $Re = 200$, we report shedding frequency and fluctuating lift and drag coefficients. Characteristic dimensions of the wake are illustrated in Fig. 12. It is evident that as N_g is increased, the fast method gives nearly identical results to the previously published data. It appears that $N_g = 4$ is sufficient to recover the previous results. Note that for the original IBPM, computations are performed over a domain of $[-30, 30] \times [-30, 30]$ by 300×300 stretched grid points with the finest resolution of $\Delta x = \Delta y = 0.02$. The time step for all cases are chosen to be $\Delta t = 0.01$ to limit the maximum Courant number to 1.

Table 1
Comparison of results from the fast-method with previously reported values for steady-state flow around a cylinder at $Re = 40$

	l/d	a/d	b/d	θ	C_D	Speed-up
Re = 40						
Present ($N_g = 2$)	1.69	0.60	0.55	53.4°	1.92	25.8
Present ($N_g = 3$)	2.01	0.67	0.58	54.0°	1.68	18.5
Present ($N_g = 4$)	2.17	0.70	0.59	53.8°	1.58	14.2
Present ($N_g = 5$)	2.20	0.70	0.59	53.5°	1.55	11.3
Linnick and Fasel [18]	2.28	0.72	0.60	53.6°	1.54	–
Taira and Colonius [36]	2.30	0.73	0.60	53.7°	1.54	1

Table 2
Comparison of results from the fast-method with previously reported values for unsteady flow around a cylinder at $Re = 200$

	St	C_D	C_L	Speed-up
Re = 200				
Present ($N_g = 2$)	0.206	1.47 ± 0.049	± 0.66	121.1
Present ($N_g = 3$)	0.200	1.40 ± 0.052	± 0.70	84.7
Present ($N_g = 4$)	0.197	1.36 ± 0.046	± 0.70	65.4
Present ($N_g = 5$)	0.195	1.34 ± 0.045	± 0.68	53.0
Linnick and Fasel [18]	0.197	1.34 ± 0.044	± 0.69	–
Taira and Colonius [36]	0.196	1.35 ± 0.048	± 0.68	1

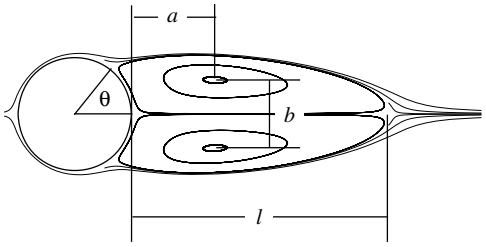

Fig. 12. Characteristic wake dimensions behind a cylinder.

In the tables, speed-up is defined as the time required to compute the last 50 time steps in the simulations normalized by the time elapsed for the original IBPM. By measuring the last 50 time steps, we give a conservative estimate for speed-up since the original method is iterative and typically requires many more iterations for earlier times. Thus with $N_g = 4$ the fast method reduces the computational time by a factor of about 15 for the steady flow and 65 for the unsteady flow. We have found similar speed-ups in a variety of problems on which we have tested the code. We note that we have thus far only implemented the fast method in two dimensions (the original algorithm has been

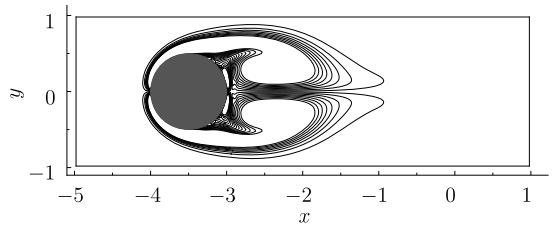

Fig. 13. Vorticity field around an impulsively started cylinder at $t = 3.5$ for $Re = 200$ with contour levels from -3 to 3 in increments of 0.4. The box shows the innermost domain $\mathscr{D}^{(1)}$.

validated in both two and three dimensions). Speed-ups for three-dimensional problems are likely to be more dramatic as discussed in Section 3.3.

Next, we compare the speed-up from for a translating circular cylinder simulated by moving the Lagrangian boundary points. Now Eq. (36) is solved iteratively with the conjugate-gradient method. A cylinder originally at the origin at $t = 0$ is impulsively translated to the left with unit velocity with $Re = 200$. The innermost domain is selected as $\mathscr{D}^{(1)} = [-5, 1] \times [-1, 1]$ with $\Delta = 0.02D$ and we use $N_g = 4$ multi-domains. Inside this highly confined $\mathscr{D}^{(1)}$, the translating cylinder generates two counter rotating vortices in the wake as shown in Fig. 13 for $t = 3.5$. The vorticity profile is in accord with previous results reported in [36]. Compared to a computation performed with the original approach, the present computation is found to be 43.4 times faster. Recall that a speed-up of 53.0 is observed for a stationary cylinder (Table 2), which suggests that the overall algorithm is still solved efficiently even with a moving immersed boundary.

7. Summary

We have reported on improvements to the immersed boundary projection method for flow over two- and three-dimensional bodies with prescribed motion. In previous work [36], we showed that the IB method can be formulated in an algebraically identical way to the incompressible Navier–Stokes equations without an immersed boundary. This formulation enables the classical fractional step method to be applied to the IB equations, eliminating the need for any constitutive relation for the motion of the body (and hence associated stiffness), and ensuring that the no-slip and divergence-free constraints are satisfied to arbitrarily high precision. In this paper we showed that the solution can be substantially accelerated by employing a nullspace (discrete streamfunction) method to satisfying the divergence free-constraint, and by restricting the computation to equally-spaced meshes.

In this fast method, the viscous terms, divergence-free, and no-slip constraints are still treated implicitly, but the linear systems associated with the Poisson equation and implicit viscous terms can be solve directly with fast sine transforms. In the solution, the divergence-free constraint is automatically satisfied to machine precision. For stationary bodies, the no-slip constraint can also be enforced to machine precision by direct solution of the equation for the body forces by using a Cholesky decomposition. For moving bodies, iterative solution of the linear system for the body forces is still required, but the size of the system is proportional to the number of Lagrangian surface points; the matrix is positive definite and the conjugate-gradient technique is efficient for its solution.

Near the IB, the restriction to uniform mesh is a standard requirement of the discrete delta function; however, far from the body, this would in general be overly restrictive as it is useful to stretch the mesh so that the domain can be made large to approach the solution on an unbounded domain. We pursued an alternative strategy of improving the far-field boundary conditions to the point where the domain can enclose the body (and the portion of the wake one wishes to resolve) snugly. We derived a multi-domain technique that solves the Poisson equation on progressively larger, but coarser, meshes. Vorticity is allowed to advect and diffuse from finer to coarser mesh. The resulting flow on the original domain then accounts for both (i) the slowly decaying potential flow induced by the body motion, and (ii) the slowly decaying induced velocity associated with vorticity that has advected to large distance from the body. While there is cost penalty associated with the multi-domain solution, the overall scheme employing the fast nullspace method and multi-domain boundary conditions is still more than an order-of-magnitude faster than our original method in two dimensions. The speed-up results from benefits associated with the fast nullspace method as well as being able to use more compact domains.

The fast nullspace method and multi-domain boundary conditions are equally valid for both two- and three-dimensional flows. Two-dimensional test cases including stationary and advecting Oseen vortices and flow over impulsively-started cylinders demonstrate the accuracy of the multi-domain boundary conditions. We note that the techniques are generally applicable to the incompressible Navier–Stokes equations on unbounded domains with or without immersed boundaries. The multi-domain technique, in particular, should prove useful in simulating flows that involve weak interactions of finite-circulation vortices that have plagued methods employing periodic or other simplified boundary conditions in the past (*e.g.* [10,14]).

Acknowledgements

The authors thank Prof. Blair Perot for the enlightening discussions on the fractional step method. The experimental data presented in Section 2.2 were generously shared by Dr. William Dickson. This work was partially supported by the United States Air Force Office of Scientific Research (AFOSR/MURI FA9550-05-1-0369) and the National Science Foundation (DMS-0514414).

Appendix A

Examples of the point-operator notation implied by some of the matrix–vector multiplies in the paper are given here, for the (relevant) case of a uniform, two-dimensional, staggered grid with equal grid spacing, Δ, in both directions (for which $M = I$). These operators can all be simply derived as special cases of those used for unstructured meshes [4], and three-dimensional versions are straightforward. Not reported here are the regularization/interpolation operators (E, H) which are given by [36]. In what follows the subscripts i and j refer to the ith and jth cells in the x and y directions, respectively, and the superscript (k), if present, refers to the kth component of a vector quantity such as velocity or gradient of pressure.

$$(Dq)_{i,j} = q^{(1)}_{i+1,j} - q^{(1)}_{i,j} + q^{(2)}_{i,j+1} - q^{(2)}_{i,j}, \tag{41}$$

$$(Gp)^{(1)}_{i,j} = p_{i,j} - p_{i-1,j}, \tag{42}$$

$$(Gp)^{(2)}_{i,j} = p_{i,j} - p_{i-1,j}, \tag{43}$$

$$(Cs)^{(1)}_{i,j} = s_{i,j+1} - s_{i,j}, \tag{44}$$

$$(Cs)^{(2)}_{i,j} = -(s_{i+1,j} - s_{i,j}), \tag{45}$$

$$(C^{\mathrm{T}}q)_{i,j} = q^{(2)}_{i,j} - q^{(2)}_{i-1,j} - (q^{(1)}_{i,j} - q^{(1)}_{i,j-1}), \tag{46}$$

$$Re\Delta^2 (Lq)^{(k)}_{i,j} = q^{(k)}_{i+1,j} + q^{(k)}_{i-1,j} + q^{(k)}_{i,j+1} + q^{(k)}_{i,j-1} - 4q^{(k)}_{i,j}, \quad k = 1, 2. \tag{47}$$

Appendix B

In this appendix, we derive the theoretical estimates for the velocity error expected from the use of multi-domain boundary conditions for the Poisson equation. We consider the velocity errors for potential flow around a cylinder and stationary Oseen vortex discussed earlier in the Section 5.

B.1. Error estimate for potential flow over a cylinder

Let us consider a circular cylinder of diameter D situated at the origin inside a rectangular domain $[-L/2,\ L/2]\times[-rL/2,\ rL/2]$. Side length L here denotes the size of the smallest $\mathscr{D}^{(1)}$ and we represent the size of the largest domain $\mathscr{D}^{(N_g)}$ by αL, where $\alpha = 2^{N_g}/2$. Aspect ratio of the domains is denoted by r. To assess the velocity error, we compute the velocity error at top center of $\mathscr{D}^{(1)}(x = 0, y = \frac{r}{2}L)$. Other points in the domain scale similarly. The exact potential flow solution at this point for an unbounded domain is

$$u_{\mathrm{exact}} = U\left[1 + \left(\frac{D}{L}\right)^2\right], \tag{48}$$

where U is the freestream velocity value. The corresponding vertical velocity v is zero at this point.

To estimate the error induced by the multi-domain approach, the effect of Dirichlet boundary conditions on the largest domain can be assessed using the method of images. Assuming the cylinder is placed at the origin, we obtain:

$$u_{\mathrm{images}} = U + U\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}(-1)^{i+j}\frac{(D/2)^2}{(x_i^2 + y_j^2)^2}(y_j^2 - x_i^2), \tag{49}$$

where x_i and y_j corresponds to the distance from the center of the (i, j)th cylinder to the point of error assessment $\left(x = 0, y = \frac{r}{2}L\right)$. Substituting $x_i = \alpha L i$ and $y_j = \alpha r L j + \frac{rL}{2}$, and subtracting the exact (free-space) solution, we obtain the error

$$\begin{aligned}\epsilon &= u_{\mathrm{images}} - u_{\mathrm{exact}} \\ &= -\frac{UD^2}{r^2L^2} + \frac{UD^2}{4\alpha^2L^2}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}(-1)^{i+j}\frac{r^2\left(j+\frac{1}{2\alpha}\right)^2 - i^2}{\left[r^2\left(j+\frac{1}{2\alpha}\right)^2 + i^2\right]^2} \\ &= -\frac{UD^2}{r^2L^2} + \frac{UD^2}{4\alpha^2L^2}\frac{\pi^2}{4}\sum_{j=-\infty}^{\infty}(-1)^j\{\mathrm{csch}^2[\beta_j(\alpha, r)] \\ &\quad + \mathrm{sech}^2[\beta_j(\alpha, r)]\},\end{aligned} \tag{50}$$

where $\beta_j(\alpha, r) = \frac{\pi r}{4\alpha}(1 + 2\alpha j)$. The sum can be broken up as follows. For the $j = 0$ term, a Taylor series expansion for large α is used. For $j \neq 0$, $1 + 2\alpha j$ can be replaced by $2\alpha j$ for large α in the exponentials. We then obtain:

$$\epsilon = -\frac{UD^2}{L^2}\frac{\pi^2}{8\alpha^2}\left[\frac{1}{3} + C(r)\right] + \mathscr{O}(\alpha^{-4}), \tag{51}$$

where the sum

$$C(r) = \sum_{j=1}^{\infty}(-1)^j\left[\mathrm{csch}^2\left(\frac{\pi r j}{2}\right) + \mathrm{sech}^2\left(\frac{\pi r j}{2}\right)\right] \tag{52}$$

is independent of α and can be evaluated numerically for a given aspect ratio, r. Thus we obtain the estimate

$$\epsilon \sim -\frac{UD^2}{L^2}\left[\frac{1}{3} + C(r)\right]4^{-N_g}, \tag{53}$$

which is the desired result that shows that the error decreases geometrically with increasing grid level. Note that we can be more precise by noting that the sum $C(r)$ goes rapidly to zero for $r > 1$, and increases like $1/r^2$ for small r. Thus we can also write

$$\epsilon \sim \frac{UD^2}{[\min(L, rL)]^2}4^{-N_g}. \tag{54}$$

B.2. Error estimate for stationary Oseen vortex

For the Oseen vortex, we follow a similar setup as the previous section, but place an Oseen vortex at the origin of the domain. We now consider the normal velocity error at the point $\left(x = \frac{L}{2}, y = 0\right)$. Assuming that the boundary of the innermost domain is outside the vortex core, we have

$$v_{\text{exact}} = \frac{\Gamma}{2\pi(L/2)}, \tag{55}$$

whereas the solution with Dirichlet boundary conditions is

$$u_{\text{images}} = \frac{\Gamma}{2\pi\alpha}\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}(-1)^{i+j}\frac{\alpha Li+\frac{L}{2}}{\left(\alpha Li+\frac{L}{2}\right)^2+(\alpha rj)^2}, \tag{56}$$

so that the error can be written (after similar simplifications to analysis in the previous section)

$$\begin{aligned}\epsilon &= u_{\text{images}} - u_{\text{exact}}\\ &= \frac{\Gamma}{2\pi}\left\{-2+\frac{\pi}{\alpha r}\operatorname{csch}\left(\frac{\pi}{2\alpha r}\right)+\frac{\pi}{\alpha r}\sum_{i=1}^{\infty}(-1)^i[\operatorname{csch}(\gamma_i^+(\alpha,r))\right.\\ &\quad\left.+\operatorname{csch}(\gamma_i^-(\alpha,r))]\right\}\end{aligned} \tag{57}$$

where $\gamma_i^{\pm}(\alpha,r)=\frac{\pi}{2\alpha r}(1\pm 2\alpha i)$. Again, we can expand the terms for large α. In this case, however, care must be taken in evaluating the sum for large α since the sum is zero to first order in α. We obtain a similar result to the previous section, namely

$$\epsilon \sim \frac{\Gamma}{\min(L,rL)}4^{-N_g}. \tag{58}$$

A further cancellation between the second and third terms on the right-hand side of Eq. (57) occurs when $r=1$. We can then show that

$$\epsilon \sim \frac{\Gamma}{L}16^{-N_g} \quad \text{when } r=1. \tag{59}$$

References

[1] J.P. Berenger, A perfectly matched layer for the absorption of electromagnetic waves, J. Comput. Phys. 114 (1994) 185–200.

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

[3] A.B. Cain, J.H. Ferziger, W.C. Reynolds, Discrete orthogonal function expansions for non-uniform grids using the fast Fourier transform, J. Comput. Phys. 56 (1984) 272–286.

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

[5] T. Colonius, Modeling artificial boundary conditions for compressible flow, Annu. Rev. Fluid Mech. 36 (2004) 315–345.

[6] T. Colonius, H. Ran, A super-grid-scale model for simulating compressible flow on unbounded domains, J. Comput. Phys. 182 (2002) 191–212.

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

[8] R. Glowinski, T.W. Pan, J. Périaux, Distributed Lagrange multiplier methods for incompressible viscous flow around moving rigid bodies, Comput. Method Appl. Mech. Engrg. 151 (1998) 181–194.

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

[10] P.M. Gresho, Incompressible fluid dynamics: fundamental formulation issues, Annu. Rev. Fluid Mech. 23 (1991) 413–453.

[11] C.A. Hall, Numerical solution of Navier–Stokes problems by the dual variable method, SIAM J. Alg. Disc. Methods 6 (2) (1985) 220–236.

[12] F.Q. Hu, On absorbing boundary conditions for linearized Euler equations by a perfectly matched layer, J. Comput. Phys. 129 (1) (1996) 201–219.

[13] J.C.R. Hunt, A.A. Wray, P. Moin, Eddies, stream, and convergence zones in turbulent flows, Center for Turbulence Research, Rep. CTR-S88, 1988.

[14] D.S. Pradeep, F. Hussain, Effects of boundary condition in numerical simulations of vortex dynamics, J. Fluid Mech. 516 (2004) 115–124.

[15] G. Jin, M. Braza, A nonreflecting outlet boundary condition for incompressible unsteady Navier–Stokes calculations, J. Comput. Phys. 107 (1993) 239–253.

[16] M. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.

[17] D.K. Lilly, On the computational stability of numerical solutions of time-dependent non-linear geophysical fluid dynamics problems, Mon. Wea. Rev. 93 (1) (1965) 11–26.

[18] M.N. Linnick, H.F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, J. Comput. Phys. 204 (2005) 157–192.

[19] W.K. Liu, Y. Liu, D. Farrell, L. Zhang, X.S. Wang, Y. Fukui, N. Patankar, Y. Zhang, C. Bajaj, J. Lee, J. Hong, X. Chen, H. Hsu, Immersed finite element method and its applications to biological systems, Comput. Method Appl. Mech. Engrg. 195 (2006) 1722–1749.

[20] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 239–261.

[21] J. Mohd-Yusof, Combined immersed-boundary/B-spline methods for simulations of flow in complex geometries, Center for Turbulence Research, Annual Research Briefs, 1997, pp. 317–327.

[22] Y. Mori, C.S. Peskin, Implicit second order immersed boundary methods with boundary mass, Comput. Methods Appl. Mech. Engrg. 197 (25–28) (2008) 2049–2067.

[23] Y. Morinishi, T.S. Lund, O.V. Vasilyev, P. Moin, Fully conservative higher order finite difference schemes for incompressible flow, J. Comput. Phys. 143 (1998) 90–124.

[24] E.P. Newren, A.L. Fogelson, R.D. Guy, R.M. Kirby, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2007) 702–719.

[25] M.A. Ol'shanskii, V.M. Staroverov, On simulation of outflow boundary conditions in finite difference calculations for incompressible fluid, Int. J. Numer. Methods Fluid 33 (2000) 499–534.

[26] B. Perot, Conservation properties of unstructured staggered mesh schemes, J. Comput. Phys. 159 (2000) 58–89.

[27] J.B. Perot, An analysis of the fractional step method, J. Comput. Phys. 108 (1993) 51–58.

[28] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271.

[29] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.

[30] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in FORTRAN: The Art of Scientific Computing, second ed., Cambridge University Press, 1992.

[31] S.C. Rennich, S.K. Lele, Numerical method for incompressible vortical flows with two unbounded directions, J. Comput. Phys. 137 (1997) 101–129.

[32] L.F. Rossi, Merging computational elements in vortex simulations, SIAM J. Sci. Comput. 18 (4) (1997) 1014–1027.

[33] R.L. Sani, P.M. Gresho, Résumé and remarks on the open boundary condition minisymposium, Int. J. Numer. Methods Fluid 18 (1994) 983–1008.

[34] D. Shiels, Simulation of controlled bluff body flow with a viscous vortex method, Ph.D. Thesis, California Institute of Technology, 1998.

[35] J.R. Shewchuk, An introduction to the conjugate gradient method without the agonizing pain. Online at http://www.cs.cmu.edu/~jrs/jrspapers.html (1994).

[36] K. Taira, T. Colonius, The immersed boundary method: a projection approach, J. Comput. Phys. 225 (2007) 2118–2137.
[37] K. Taira, W.B. Dickson, T. Colonius, M.H. Dickinson, C.W. Rowley, Unsteadiness in flow over a flat plate at angle-of-attack at low Reynolds numbers, AIAA 2007-710, 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 2007.
[38] X. Wang, From immersed boundary method to immersed continuum method, Int. J. Multiscale Comput. Engrg. 4 (2006) 127–145.
[39] Z.J. Wang, Efficient implementation of the exact numerical far field boundary condition for Poisson equation on an infinite domain, J. Comput. Phys. 153 (1999) 666–670.