

The Immersed Boundary Lattice Green Function method for External Aerodynamics

Gianmarco Mengaldo*, Sebastian Liska†, Ke Yu‡, Tim Colonius‡

California Institute of Technology, Pasadena, CA 91125, U.S.A.

Thierry Jardin§

ISAE-Supaero, Université de Toulouse, 31055 Toulouse Cedex 4, France.

In this paper, we document the capabilities of a novel numerical approach — the immersed boundary lattice Green’s function (IBLGF)¹ method — to simulate external incompressible flows over complex geometries. This new approach is built upon the immersed boundary method and lattice Green’s functions to solve the incompressible Navier-Stokes equations. We show that the combination of these two concepts allows the construction of an efficient and robust numerical framework for the direct numerical and large-eddy simulation of external aerodynamic problems at moderate to high-Reynolds numbers.

I. Introduction

High-fidelity computational fluid dynamics (CFD) tools, such as direct numerical simulation (DNS) and large-eddy simulation (LES), constitute very attractive alternatives to more established RANS-based CFD approaches, and they can substantially improve the prediction of separated and turbulent flows.^{2,3} From this perspective, applications involving micro air vehicles (MAV) are likely to become within the reach of DNS in the next few years. Here, typical Reynolds numbers are of the order of thousands and tens of thousands.⁴ On the other hand, DNS within the context of civil and military aircraft aerodynamics is still to be unattainable for many years as the Reynolds numbers involved can range from few to more than twenty million.⁴ For these applications LES is required for the foreseeable future.³

In order for these CFD tools to be adopted for engineering design, the underlying numerics must satisfy several requirements: accuracy, robustness, time-to-solution, geometrical flexibility, conservation, mimetic and topological properties of the numerical discretization are in fact crucial in order to achieve a ‘physically’ accurate solution, especially at coarse mesh resolution. These requirements are significantly exacerbated when dealing with problems involving complex moving boundaries.

Many of the discretization strategies developed for DNS/LES calculations struggle to satisfy at least one of these essential requirements. For instance, traditional finite-difference methods are limited in terms of geometrical flexibility, whereas unstructured-mesh methods, such as spectral element approaches, although particularly tailored for unsteady aerodynamic problems in complex geometries, suffer from a lack of robustness,⁵ that can be aggravated when simulating bodies with prescribed (rigid-) body motion. In addition, traditional CFD solvers adopted in industry usually require large computational domains to avoid altering the flow physics due to blockage issues. This substantially increases the computational resources needed, and requires the practitioner to tune the dimensions of the domain for each simulation and use appropriate boundary conditions to avoid spurious reflections from the boundaries. Furthermore, commonly used body-fitted methods require a significant pre-processing effort in terms of mesh generation. This is typically a very time-consuming (although automated) task in the design pipeline, since the mesh must conform to some minimum quality requirements in order for the solver to produce a numerically accurate solution.

*Postdoctoral Scholar, Department of Mechanical Engineering. AIAA Member.

†Graduate Student, Department of Mechanical Engineering. AIAA Member.

‡Professor, Department of Mechanical Engineering. AIAA Member.

§Institut Supérieur de l’Aéronautique et de l’Espace (ISAE-Supaero). AIAA member.

The immersed boundary (IB) method offers potential advantages for external flows over moving and deforming surfaces by obviating the need for labor- and computationally intensive gridding and re-meshing efforts.⁶ However, IB methods have, to date, been primarily used to compute low Reynolds number flows, thus limiting their adoption to a wider set of applications. Indeed, there is a mantra within the CFD community that the many advantages that IB methods enjoy will not hold true at high Reynolds numbers. The idea that IB methods will become inefficient at high Reynolds numbers stems from the fact that in many variants of the IB method, including ours, only first-order accuracy (in space) is achieved in a region near the surface. Worse, it is not possible to resolve the thin boundary layers that exist at high Reynolds numbers by clustering points in the surface normal direction, since IB methods do not use body-fitted meshes. Without remedy, these features would indeed result in very poor scale up of IB methods to high Reynolds. The IB community is already working to prove the mantra wrong. Cut-cell and sharp interface methods can extend the IB treatment to second order accuracy,⁷ and adaptive mesh refinement (AMR) can cluster grid points in the thin boundary layers with similar efficiency as in body-fitted grids.^{8–10}

In this paper, we describe a novel numerical approach introduced by Liska and Colonius,¹ the Immersed-Boundary Lattice-Green Function (IBLGF) method, and we show its capabilities for external incompressible aerodynamic problems at moderate Reynolds numbers. The IBLGF method is geometrically flexible, as it uses the IB formulation of the incompressible Navier-Stokes equations. The spatial discretization adopted is a second-order mimetic finite volume method on a staggered Cartesian grid, that permits the method to maintain crucial conservation properties and therefore to faithfully represent the underlying continuum physical problem. The novel concept adopted in the IBLGF method is however the use of lattice Green’s functions. These allow the computational domain to be restricted to regions that dictate the flow evolution, while naturally enforcing free-space boundary conditions. LGF-based approaches can be efficiently solved using fast multipole methods¹¹ in combination with a projection technique that computes the solutions to the viscous integrating factor half-explicit Runge-Kutta (HERK) time integration scheme used to solve the velocity and the pressure of the flow.¹² For a comprehensive overview of the IBLGF methodology and of the related concepts the interested reader can refer to Liska.¹³ This new framework can be complemented with LES models and AMR to simulate high-Reynolds number flow problems that are of primary interest in the aerospace industry, thus extending the range of capabilities of IB-based numerical technologies. In particular, we will highlight the use of the IBLGF code to perform DNS of low and moderate Reynolds number flows, for both non-moving and moving immersed bodies. The latter simulations are the first verification of the algorithm for immersed surfaces with non-rectilinear motion.

II. Governing equations

We consider the incompressible Navier-Stokes equations formulated in an IB framework. The IB treatment adopted here uses a distributed Lagrange multiplier method^{14–17} that is particularly tailored to computing flows around stationary and moving rigid bodies. The no-slip boundary conditions are introduced through an appropriate force vector $\mathbf{f}_{\partial\mathcal{B}}$ located along the surfaces $\partial\mathcal{B}$ of the immersed body \mathcal{B} , where the immersed surfaces are described in Lagrangian coordinates $\mathbf{X} = \mathbf{X}(\boldsymbol{\xi}, t)$, with $\boldsymbol{\xi}$ constituting the parametrization of the surfaces. The unknown forcing term $\mathbf{f}_{\partial\mathcal{B}}$ is then treated as a Lagrange multiplier which is added to the momentum equation. The system of equations to be solved assumes the following form

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{\text{Re}} \nabla^2 \mathbf{u} + \int_{\partial\mathcal{B}} \mathbf{f}_{\partial\mathcal{B}}(\boldsymbol{\xi}, t) \delta(\mathbf{X}(\boldsymbol{\xi}, t) - \mathbf{x}) d\boldsymbol{\xi}, \quad (1a)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (1b)$$

$$\int_{\mathcal{D}} \mathbf{u} \delta(\mathbf{x} - \mathbf{X}(\boldsymbol{\xi}, t)) d\mathbf{x} = \mathbf{u}_{\partial\mathcal{B}}(\boldsymbol{\xi}, t), \quad (1c)$$

where $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ and p denote the velocity vector and the pressure, respectively, Re is the Reynolds number, δ represents the Dirac delta function, $\mathbf{x} = \mathbf{x}(t)$ denotes the spatial coordinates in the Eulerian domain \mathcal{D} and $\mathbf{u}_{\partial\mathcal{B}}(\boldsymbol{\xi}, t) = \frac{\partial \mathbf{X}}{\partial t}$ denotes velocity of the non-deformable immersed body \mathcal{B} . We note that equation (1c) represents the no-slip condition on the immersed surface of the body $\partial\mathcal{B}$, and the convolutions with the Dirac delta function in equation (1a) and (1c) allow the transfer of information between $\partial\mathcal{B}$ and \mathcal{D} . The velocity vector \mathbf{u} and the pressure are defined for all $\mathbf{x} \in \mathcal{D}$ and are subject to boundary conditions $\mathbf{u}(\mathbf{x}, t) \rightarrow \mathbf{u}_{\infty}(t)$ as $|\mathbf{x}| \rightarrow \infty$, where \mathbf{u}_{∞} is the farfield velocity. Note also that the force vector $\mathbf{f}_{\partial\mathcal{B}}(\boldsymbol{\xi}, t)$ is calculated such that the velocity $\mathbf{u}(\mathbf{x}, t)$ satisfies (1c).

When using LGF-based methods and non-deformable bodies moving in a quiescent fluid (e.g. a pitching airfoil or a rotating rotor), it is convenient to rewrite the IB formulation of the incompressible Navier-Stokes equations (1) in an accelerating frame of reference, but then using the velocity in the inertial frame of reference. The main reason behind this manipulation is due to the source terms arising from the accelerating frame of reference. Then the source terms can be incorporated into the pressure gradient and nonlinear terms and the boundary condition on the velocity at large distance from the body tends to zero, that is $\mathbf{u}(\mathbf{x}, t) \rightarrow 0$ for $|\mathbf{x}| \rightarrow \infty$. The system of equations (1) in the new frame of reference and with the change of variables for the velocity becomes

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}_a \cdot \nabla (\mathbf{u}_a + 2\boldsymbol{\Omega} \times \mathbf{x}_a) = -\nabla q + \frac{1}{\text{Re}} \nabla^2 \mathbf{u} + \int_{\partial \mathcal{B}} \mathbf{f}_{\partial \mathcal{B}}(\boldsymbol{\xi}, t) \delta(\mathbf{X}(\boldsymbol{\xi}, t) - \mathbf{x}) d\boldsymbol{\xi}, \quad (2a)$$

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

$$\int_{\mathcal{D}} \mathbf{u} \delta(\mathbf{x} - \mathbf{X}(\boldsymbol{\xi}, t)) d\mathbf{x} = \mathbf{u}_{\partial \mathcal{B}, a}(\boldsymbol{\xi}, t) + \mathbf{u}_r(\mathbf{X}, t). \quad (2c)$$

In equation (2), $\mathbf{x}_a = \mathbf{x} - \mathbf{R}(t)$ represents the position vector of a point relative to the origin of the accelerating-frame coordinates. The accelerating-frame coordinates are centered around $\mathbf{R}(t)$, translate with velocity $\mathbf{U}(t) = [\frac{d\mathbf{R}}{dt}](t)$ and rotate (with respect to $\mathbf{R}(t)$) with angular velocity $\boldsymbol{\Omega}(t)$. The vector $\mathbf{u}_a = \mathbf{u}(\mathbf{x}, t) - \mathbf{u}_r(\mathbf{x}_a, t)$ denotes the velocity in the accelerating reference frame, where $\mathbf{u}_r = \mathbf{U}(t) + \boldsymbol{\Omega}(t) \times \mathbf{x}_a$ is the velocity of a point in the accelerating reference frame relative to the inertial frame. The scalar $q(\mathbf{x}, t)$ is a pressure-like quantity that can be related to the inertial-frame pressure up to an arbitrary time-dependent constant, using $q(\mathbf{x}, t) = p(\mathbf{x}, t) - \frac{1}{2} |\mathbf{u}_r(\mathbf{x}_a, t)|^2$. The operators $\delta(\partial \mathcal{B}(t), \mathbf{f}_{\partial \mathcal{B}}, \mathbf{x})$ and $\delta(\partial \mathcal{B}(t), \mathbf{u}, \boldsymbol{\xi})$ denote the δ -convolutions in equation (1). Finally, the vectors $\mathbf{X}_a(\boldsymbol{\xi}, t) = \mathbf{X}(\boldsymbol{\xi}, t) - \mathbf{R}(t)$ and $\mathbf{u}_{\partial \mathcal{B}, a}(\boldsymbol{\xi}, t) = \mathbf{u}_{\partial \mathcal{B}}(\boldsymbol{\xi}, t) - \mathbf{u}_r(\mathbf{X}_a(\boldsymbol{\xi}, t), t)$ represent the position and velocity of a point on $\partial \mathcal{B}(t)$ in the accelerating frame of reference. Also, note that we used the same symbols for the differential operators as in equation (1) for simplicity of notation.

The IBLGF solver used in this paper implements both formulations. In the next section, without loss of generality, we will focus on equation (2).

III. The Immersed Boundary Lattice-Green Function method

The system of equations (2) is formally approximated in space by using a second-order (mimetic) finite-volume (FV) method on an unbounded staggered Cartesian grid. The spatially-discretized system is then recast into a discrete algebraic equation of index 2 (DAE-*i2*) by using an integrating factor technique. The DAE-*i2* system is discretized in time by means of an half explicit Runge-Kutta (HERK) method and solved through an exact fractional step technique.

In this section, we briefly describe the IBLGF method following closely Liska and Colonius.¹ In particular, in subsection [III.A] we focus on the spatial discretization, in subsection [III.B] we outline the integrating factor technique and the construction of the DAE-*i2* system and in subsection [III.C] we describe the key steps to solve the system of fully discrete equations. In addition, in subsection [III.D], we highlight the ability of the code of restricting the computational domain to regions that dictate the flow evolution.

III.A. Semi-discrete formulation

The spatial discretization is obtained via a mimetic FV method that employs a staggering of the variables, where the pressure-like term is defined at the cell center, the velocity is located at the center of the faces constituting a given cell and the vorticity along the edges. A schematic representation of a cell of the mesh is depicted in figure [1]. The notation adopted to define the mesh object \mathcal{Q} whose a given grid function \mathbf{g} belong to is $\mathbf{g} \in \mathcal{Q} := \{\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{C}\}$, where \mathcal{V} denotes vertices, \mathcal{E} denotes edges, \mathcal{F} denotes faces and \mathcal{C} denotes cells.

As a direct consequence of the staggering, the differential, regularization and interpolation operators can act on the various mesh objects $\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{C}$ and can eventually produce a resulting quantity that belongs to a different mesh object. The notation adopted in this case for a generic operator \mathbb{T} that operates on a grid function $\mathbf{g}_1 \in \mathcal{C}$ and produces a grid function $\mathbf{g}_2 \in \mathcal{F}$ is $\mathbb{T} : \mathbb{R}^{\mathcal{C}} \rightarrow \mathbb{R}^{\mathcal{F}}$. In table [1], we report the definition of the differential continuous and discrete counterparts of the operators necessary to define system (2). By

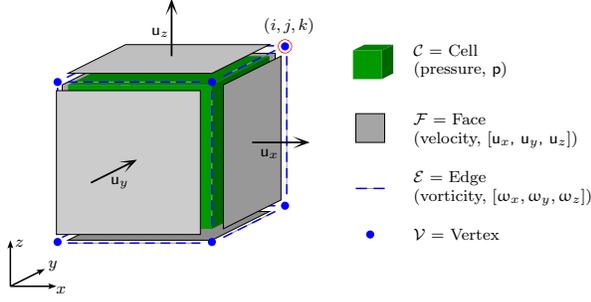


Figure 1. Staggered cell used by the IBLGF method.

Operator	Continuous	Discrete
Gradient	∇	\mathbf{G}
Divergence	$\nabla \cdot$	\mathbf{D}
Curl	$\nabla \times$	\mathbf{C}
Laplacian	∇^2	\mathbf{L}
Regularization	$\int_{\mathbb{R}^3} \mathbf{u} [\delta(\mathbf{x} - \mathbf{X})] d\mathbf{x}$	$[\mathbf{R}(t)]\mathbf{u}$
Interpolation	$\int_{\partial\mathcal{B}} \mathbf{f}_{\partial\mathcal{B}}(\boldsymbol{\xi}, t) [\delta(\mathbf{X} - \mathbf{x})] d\boldsymbol{\xi}$	$[\mathbf{E}(t)]\mathbf{u}$
Convection	$\mathbf{u}_a \cdot \nabla(\mathbf{u}_a + 2\boldsymbol{\Omega} \times \mathbf{x}_a)$	$\mathbf{N}(\mathbf{u})$

Table 1. List of operators adopted in both continuous and discrete form.

using the conventions just defined, the semi-discrete form of equation (2) becomes

$$\frac{d\mathbf{u}}{dt} + \mathbf{N}(\mathbf{u}, t) = -\mathbf{G}\mathbf{q} + \frac{1}{\text{Re}} \mathbf{L}_{\mathcal{F}}\mathbf{u} + [\mathbf{R}(t)]\mathbf{f}_{\partial\mathcal{B}}, \quad (3a)$$

$$\overline{\mathbf{D}}\mathbf{u} = 0, \quad (3b)$$

$$[\mathbf{E}(t)]\mathbf{u} = \mathbf{u}_{\partial\mathcal{B}}, \quad (3c)$$

where $\mathbf{u} \in \mathbb{R}^{\mathcal{F}}$ and $\mathbf{q} \in \mathbb{R}^{\mathcal{C}}$ are the discrete counterparts of the velocity and pressure-like variables, \mathbf{R} and \mathbf{E} are the regularization and the interpolation operators deriving from the discretization of the δ -convolutions in the system of equations (2) and $\mathbf{f}_{\partial\mathcal{B}}$, $\mathbf{u}_{\partial\mathcal{B}}$ are the discrete forcing term located along the surface of the immersed body and the velocity of the immersed body, respectively. The interpolation and regularization operators adopted in this work are adjoints under the standard inner product, so that $\mathbf{E} = \kappa (\mathbf{R}^\dagger)$, where κ is a scalar factor depending on the grid-cell size. Note that $\mathbf{G} : \mathbb{R}^{\mathcal{C}} \rightarrow \mathbb{R}^{\mathcal{F}}$, $\overline{\mathbf{D}} : \mathbb{R}^{\mathcal{F}} \rightarrow \mathbb{R}^{\mathcal{C}}$ and $\mathbf{L}_{\mathcal{F}} : \mathbb{R}^{\mathcal{F}} \rightarrow \mathbb{R}^{\mathcal{F}}$. Also, in the following, we will make use of $\mathbf{L}_{\mathcal{Q}} : \mathbb{R}^{\mathcal{Q}} \rightarrow \mathbb{R}^{\mathcal{Q}}$, with $\mathcal{Q} \in \{\mathcal{C}, \mathcal{E}, \mathcal{V}\}$, $\overline{\mathbf{G}} : \mathbb{R}^{\mathcal{V}} \rightarrow \mathbb{R}^{\mathcal{E}}$, $\mathbf{C} : \mathbb{R}^{\mathcal{F}} \rightarrow \mathbb{R}^{\mathcal{E}}$ and $\overline{\mathbf{C}} : \mathbb{R}^{\mathcal{E}} \rightarrow \mathbb{R}^{\mathcal{F}}$.

The discrete operators reported in table [1] and used in equation (3) satisfy specific topological and mimetic properties, given the staggered grid adopted. These properties are summarized below

$$\overline{\mathbf{D}} = -\mathbf{G}^\dagger, \quad \overline{\mathbf{G}} = -\mathbf{D}^\dagger, \quad \overline{\mathbf{C}} = \mathbf{C}^\dagger \quad (\text{symmetry}) \quad (4a)$$

$$\text{Null}(\mathbf{C}) = \text{Im}(\mathbf{G}), \quad \text{Null}(\mathbf{D}) = \text{Im}(\mathbf{C}) \quad (\text{orthogonality}) \quad (4b)$$

$$\mathbf{L}_{\mathcal{C}} = -\mathbf{G}^\dagger \mathbf{G}, \quad \mathbf{L}_{\mathcal{F}} = -\mathbf{G}\mathbf{G}^\dagger, \quad \mathbf{L}_{\mathcal{E}} = -\mathbf{D}^\dagger \mathbf{D} - \mathbf{C}\mathbf{C}^\dagger, \quad \mathbf{L}_{\mathcal{V}} = -\mathbf{D}\mathbf{D}^\dagger, \quad (\text{mimetic}), \quad (4c)$$

$$\mathbf{L}_{\mathcal{F}}\mathbf{G} = \mathbf{G}\mathbf{L}_{\mathcal{C}}, \quad (\text{commutativity}). \quad (4d)$$

and are extensively used throughout the formulation of the method. The choice of this spatial discretization provide a numerical scheme that is (i) second-order accurate in space for all the differential operators, (ii) conserves momentum, kinetic energy and circulation in absence of viscosity.

For facilitating its practical implementation, it is convenient to rewrite equation (3) as follows

$$\frac{d\mathbf{u}}{dt} + \tilde{\mathbf{N}}(\mathbf{u}, t) = -\mathbf{G}\mathbf{d} + \frac{1}{\text{Re}} \mathbf{L}_{\mathcal{F}}\mathbf{u} - [\mathbf{E}(t)]^\dagger \tilde{\mathbf{f}}_{\partial\mathcal{B}}, \quad (5a)$$

$$\overline{\mathbf{D}}\mathbf{u} = 0, \quad (5b)$$

$$[\mathbf{E}(t)]\mathbf{u} = \mathbf{u}_{\partial\mathcal{B}}. \quad (5c)$$

In equation (5), we subtracted $\frac{1}{2}\mathbf{G}\mathbf{P}(\mathbf{u} - \mathbf{u}_r)$ from both sides of equation (3), such that $\tilde{\mathbf{N}} = \mathbf{N}(\mathbf{u}, t) - \frac{1}{2}\mathbf{G}\mathbf{P}(\mathbf{u} - \mathbf{u}_r)$ with $\mathbf{u}_r(\mathbf{n}, t) = \mathbf{u}_r(\mathbf{x}_{\mathcal{F}}(\mathbf{n}), t)$ and $\mathbf{d} = \mathbf{q} + \frac{1}{2}\mathbf{P}(\mathbf{u} - \mathbf{u}_r)$. Note that $\mathbf{P}(\mathbf{u})$ is a discrete approximation of $|\mathbf{v}|^2$. Therefore, the nonlinear term in equation (5), $\tilde{\mathbf{N}}$, is a discrete approximation of $\mathbf{u}_a \cdot \nabla(\mathbf{u}_a + 2\boldsymbol{\Omega} \times \mathbf{x}_a) - \frac{1}{2}\nabla|\mathbf{u}_a|^2$ and it decays faster than \mathbf{N} at large distances from the immersed body.

III.B. Integrating factor technique and time-integration

We now transform system (5) into a DAE-*i2*. This step is achieved by using an integrating factor technique that uses, as discrete integrating factor $\mathbf{H}_{\mathcal{Q}}$, the solution of the discrete heat equation $\frac{d\mathbf{h}}{dt} = \mathbf{c}\mathbf{L}_{\mathcal{Q}}\mathbf{h}$, that

is $\mathbf{H}_Q(t) = \mathbf{E}_Q\left(\frac{t-\tau}{\Delta x^2 \text{Re}}\right)$. Applying this solution technique and the two transformations $\mathbf{v} = [\mathbf{H}_{\mathcal{F}}(t)]\mathbf{u}$, $\mathbf{b} = \mathbf{H}_{\mathcal{C}}(t)\mathbf{d}$ to equation (5), we obtain:

$$\frac{d\mathbf{v}}{dt} + [\mathbf{H}_{\mathcal{F}}(t)]\tilde{\mathbf{N}}([\mathbf{H}_{\mathcal{F}}^{-1}(t)]\mathbf{v}, t) = -\mathbf{G}\mathbf{b} - [\mathbf{H}_{\mathcal{F}}(t)][\mathbf{E}(t)]^\dagger \tilde{\mathbf{f}}_{\partial\mathcal{B}}, \quad (6a)$$

$$\mathbf{G}^\dagger \mathbf{v} = 0, \quad (6b)$$

$$[\mathbf{E}(t)][\mathbf{H}_{\mathcal{F}}^{-1}(t)]\mathbf{v} = \mathbf{u}_{\partial\mathcal{B}}, \quad (6c)$$

where $\mathbf{f}_{\partial\mathcal{B}} \rightarrow -(\Delta x)^3 \tilde{\mathbf{f}}_{\partial\mathcal{B}}$. Note that we used the commutativity properties of the Laplacian operators and integrating factors when used in combination of the unbounded staggered Cartesian grid adopted here. The system in (6) constitutes a DAE- $i2$ system that can be efficiently approximated in time through an s -stage HERK scheme.

In particular, we define the s -stages of the HERK scheme using the superscript i and we use the subscript k to indicate time $t_k = k\Delta t$, where Δt is the time-step. If we now group the constraint variables (i.e. Lagrange multipliers), the right-hand side and the operators together as follows:

$$\lambda_k^i = \begin{bmatrix} \mathbf{d}_k^i \\ \tilde{\mathbf{f}}_{\partial\mathcal{B}_k}^i \end{bmatrix}, \quad \zeta_k^i = \begin{bmatrix} 0 \\ \mathbf{u}_{\partial\mathcal{B}_k}^i \end{bmatrix}, \quad \mathbf{Q}_k^i = \begin{bmatrix} \mathbf{G} & [\mathbf{E}(t_k^i)]^\dagger \end{bmatrix}, \quad (7)$$

and we introduce the following variables:

$$\mathbf{u}_k^i(\mathbf{n}) = \left[\mathbf{E}_{\mathcal{F}} \left(\frac{-\tilde{c}_i \Delta t}{(\Delta x^2) \text{Re}} \right) \right] \mathbf{v}_k^i(\mathbf{n}), \quad \mathbf{d}_k^i(\mathbf{n}) = \left[\mathbf{E}_{\mathcal{F}} \left(\frac{-\tilde{c}_i \Delta t}{(\Delta x^2) \text{Re}} \right) \right] \mathbf{b}_k^i(\mathbf{n}), \quad (8)$$

the k -th time-step of the IF-HERK algorithm to integrate equation(6), reads:

1. *initialize*: set $\mathbf{u}_k^0 = \mathbf{u}_k$ and $t_k^0 = t_k$.
2. *multi-stage*: for $i = 1, 2, \dots, s$, solve the linear system:

$$\begin{bmatrix} (\mathbf{H}_{\mathcal{F}}^i)^{-1} & \mathbf{Q}_k^{i-1} \\ \mathbf{Q}_k^{i-1} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}_k^i \\ \lambda_k^i \end{bmatrix} = \begin{bmatrix} \mathbf{r}_k^i \\ \zeta_k^i \end{bmatrix}, \quad (9)$$

where

$$\mathbf{H}_{\mathcal{F}}^i = \mathbf{E}_{\mathcal{F}} \left(\frac{(\tilde{c}_i - \tilde{c}_{i-1}) \Delta t}{\Delta x^2 \text{Re}} \right), \quad \mathbf{r}_k^i = \mathbf{h}_k^i + \Delta t \sum_{j=1}^{i-1} \tilde{a}_{ij} \mathbf{w}_k^{ij} + \mathbf{g}_k^i, \quad (10)$$

$$\mathbf{g}_k^i = -\tilde{a}_{ii} \Delta t \tilde{\mathbf{N}}(\mathbf{u}_k^{i-1}, t_k^{i-1}), \quad t_k^i = t_k + \tilde{c} \Delta t. \quad (11)$$

Variables \mathbf{h}_k^i and \mathbf{w}_k^{ij} are computed recursively for $i > 1$ and $j > 1$ using the following relations:

$$\mathbf{h}_k^i = \mathbf{H}_{\mathcal{F}}^{i-1} \mathbf{h}_k^{i-1}, \mathbf{h}_k^1 = \mathbf{u}_k^0, \quad (12)$$

$$\mathbf{w}_k^{ij} = \mathbf{H}_{\mathcal{F}}^{i-1} \mathbf{w}_k^{i-1,j}, \mathbf{w}_k^{ii} = (\tilde{a}_{ii} \Delta t)^{-1} (\mathbf{g}_k^i - \mathbf{Q}_k^{i-1} \tilde{\lambda}_k^i). \quad (13)$$

3. *finalize*: set $\mathbf{u}_{k+1} = \mathbf{u}_k^s$, $\lambda_{k+1} = (\tilde{a}_{s,s} \Delta t)^{-1} \lambda_k^s$, and $t_{k+1} = t_k^s$.

The solution of equation (9) is expected to be the most computationally expensive operation. A fast technique to solve it needs therefore to be adopted in order to reduce the computational costs of the algorithm. This is discussed in the next subsection.

III.C. Fast solution of the linear system

The solution of equation (9) is achieved through an exact (free of splitting errors) projection technique. This can be viewed as an operator-block decomposition, that, in our specific case is a matrix-block LU decomposition.

In particular, we rewrite equation (6) in matrix form for a given k -th time-step and s -th stage of the IF-HERK algorithm

$$\underbrace{\begin{bmatrix} \mathbf{H}_{\mathcal{F}}^{-1} & \mathbf{G} & (\mathbf{E}_k^{i-1})^\dagger \\ \mathbf{G}^\dagger & 0 & 0 \\ \mathbf{E}_k^i & 0 & 0 \end{bmatrix}}_{\mathbf{M}_k^i} \begin{bmatrix} \mathbf{u}_k^i \\ \hat{\mathbf{d}}_k^i \\ \hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i \end{bmatrix} = \begin{bmatrix} \mathbf{r}_k^i \\ 0 \\ \mathbf{u}_{\partial\mathcal{B}_k}^i \end{bmatrix}, \quad (14)$$

where $\hat{\mathbf{d}}_k^i/\mathbf{d}_k^i = \hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i/\mathbf{f}_{\partial\mathcal{B}_k}^i = \tilde{a}_{ss}$ and \mathbf{M}_k^i is generally a non-symmetric matrix that cannot be symmetrized as the image of the regularization operator $(\mathbf{E}_k^{i-1})^\dagger$ and of the interpolation operator \mathbf{E}_k^{i-1} are different. Note that the non-symmetry of \mathbf{M}_k^i is typical of DAE- $i2$ system with time-dependent time-constraints (in our case the immersed surface can rotate/translate, thus the associated constraint is time-dependent). In the case of non-moving immersed rigid bodies, instead, matrix \mathbf{M}_k^i is symmetric. The nested projection technique for the solution of equation (14) reads as

$$(\mathbf{H}_{\mathcal{F}}^i)^{-1}\mathbf{u}^* = \mathbf{r}_k^i, \quad (\text{solve for intermediate velocity}), \quad (15a)$$

$$\mathbf{G}^\dagger\mathbf{H}_{\mathcal{F}^i}\mathbf{G}\mathbf{d}^* = \mathbf{G}^\dagger\mathbf{u}^*, \quad (\text{solve for intermediate pressure}), \quad (15b)$$

$$\mathbf{S}_k^i\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^* = \mathbf{E}_k^i[\mathbf{u}^* - \mathbf{H}_{\mathcal{F}}^i\mathbf{G}\mathbf{d}^*] - \mathbf{u}_{\partial\mathcal{B}_k}^i, \quad (\text{solve for intermediate IB forces}), \quad (15c)$$

$$\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i = \hat{\mathbf{f}}_{\partial\mathcal{B}_k}^*, \quad (\text{update forces}), \quad (15d)$$

$$\hat{\mathbf{d}}_k^i = \mathbf{d}^* - (\mathbf{G}^\dagger\mathbf{H}_{\mathcal{F}^i}\mathbf{G})^{-1}\mathbf{G}^\dagger\mathbf{H}_{\mathcal{F}}^i(\mathbf{E}_k^{i-1})^\dagger\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i, \quad (\text{correct pressure}), \quad (15e)$$

$$\mathbf{u}_k^i = \mathbf{u}^* - \mathbf{H}_{\mathcal{F}^i}[\mathbf{G}\hat{\mathbf{d}}_k^i + \mathbf{E}_k^i\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i], \quad (\text{correct velocity}), \quad (15f)$$

where

$$\mathbf{S}_k^i = \mathbf{E}_k^i\mathbf{H}_{\mathcal{F}}^i[\mathbf{I}_{\mathcal{F}} - \mathbf{G}(\mathbf{G}^\dagger\mathbf{H}_{\mathcal{F}^i}\mathbf{G})^{-1}\mathbf{G}^\dagger\mathbf{H}_{\mathcal{F}}^i](\mathbf{E}_k^i)^\dagger, \quad (16)$$

is the (force) Schur complement of the LU decomposition of equation (14), with $\mathbf{I}_{\mathcal{F}}$ being the identity operator for $\mathbb{R}^{\mathcal{F}}$. Equation (15) makes use of an operator-block LU decomposition of \mathbf{M}_k^i . In the above equations it is necessary to solve discrete Poisson-like problems - see for instance equation (15b) and (15c). By using the mimetic, orthogonality and commutativity properties of the discrete operators shown in equation (4), the nested projection technique in equation (15) reduces to the following form

$$\mathbf{L}_C\mathbf{d}^* = -\mathbf{G}^\dagger\mathbf{r}_k^i, \quad (17a)$$

$$\mathbf{S}_k^i\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i = \mathbf{E}_k^i\mathbf{H}_C^i[\mathbf{r}_k^i - \mathbf{G}^\dagger\mathbf{d}^*] - \mathbf{u}_{\partial\mathcal{B}_k}^i, \quad (17b)$$

$$\hat{\mathbf{d}}_k^i = \mathbf{d}^* + \mathbf{L}_C^{-1}\mathbf{G}^\dagger(\mathbf{E}_k^{i-1})^\dagger\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i, \quad (17c)$$

$$\mathbf{u}_k^i = \mathbf{H}_{\mathcal{F}}^i[\mathbf{r}_k^i - \mathbf{G}\hat{\mathbf{d}}_k^i - (\mathbf{E}_k^{i-1})^\dagger\hat{\mathbf{f}}_{\partial\mathcal{B}_k}^i], \quad (17d)$$

with the associated Schur complement being simplified to the following relation:

$$\mathbf{S}_k^i = \mathbf{E}_k^i[\mathbf{H}_{\mathcal{F}}^i + \mathbf{G}(\mathbf{H}_C^i)^{-1}\mathbf{L}_C\mathbf{G}^\dagger](\mathbf{E}_k^{i-1})^\dagger. \quad (18)$$

The efficient solution of equation (17) is obtained using an LGF fast multipole method (FMM) that uses flexible source and target regions. The LGF-FMM-based solution procedure permits in fact the confinement of active regions to a small neighbourhood around the support of the discrete delta functions defined along the immersed surfaces. This is particularly relevant for the calculation of the Schur complement \mathbf{S}_k^i and of the operators $\mathbf{E}_k^i\mathbf{H}_C^i$ and $\mathbf{L}_C^{-1}\mathbf{G}^\dagger(\mathbf{E}_k^{i-1})^\dagger$. The adaptive source/target region refinement plays a crucial role in reducing the operation count when compared to schemes that do not limit the source and target regions of elliptic problems.

The LGF-FMM procedure adopted here is based on the solution of the 7th-point discrete Laplacian, \mathbf{L} , for the following representative problem:

$$[\mathbf{L}\mathbf{z}](\mathbf{n}) = \mathbf{y}(\mathbf{n}), \quad \text{supp}(\mathbf{y}) \subseteq D \quad (19)$$

where \mathbf{z} and \mathbf{y} belong to $\mathbb{R}^{\mathcal{C}}$ and $\mathbb{R}^{\mathcal{V}}$, respectively, D is bounded region in \mathbb{Z}^3 . The procedure for solving the discrete Poisson problem in equation (19) using the discrete LGF of the linear operator \mathbf{L} is identical to

the procedure for solving free-space Poisson problems using the continuous LGF or fundamental solutions of ∇^2 , that is $\frac{-1}{4\pi|\mathbf{x}|}$. This is based on the convolution of the discrete right-hand side of equation (19) with the discrete Green's functions:

$$\mathbf{z}(\mathbf{n}) = [\mathbf{G}^L * \mathbf{y}](\mathbf{n}) = \sum_{\mathbf{m} \in D} \mathbf{g}_L(\mathbf{n} - \mathbf{m})\mathbf{y}(\mathbf{m}), \quad (20)$$

where \mathbf{G}^L is the LGF of the 7^{th} -point discrete Laplacian, \mathbf{L} .

The calculation of equation (17b) is probably the most expensive step of the nested projection technique. Its practical solution can be either obtained through dense linear algebra techniques or via iterative methods. In case of iterative methods, suitable options include: (i) the conjugate gradient method, tailored for symmetric \mathbf{S}_k^i — i.e. that is when the immersed body is rigid and non-moving — and (ii) the generalized minimal residual method (GMRES), useful when matrix \mathbf{S}_k^i is non-symmetric — i.e. that is in the more general case of a moving body. The dense linear algebra route can be convenient when the immersed body is rigid and non-moving. In this case, the computation of \mathbf{S}_k^i is just a pre-processing step that consist of Cholesky decomposition of $\mathbf{S}_k^i = \mathbf{C}$, that is $\mathbf{C} = \mathbf{L}\mathbf{L}^T$. This is then reused to evaluate $[\hat{\mathbf{f}}_{\partial B}] = \mathbf{f}_{\partial B} = \mathbf{L}^{-T}\mathbf{L}^{-1}\mathbf{r}$.

III.D. Domain adaptivity

As briefly mentioned above, IBLGF restricts the formally unbounded computational domain \mathcal{D} to regions that dictate the flow evolution. This feature is achieved by limiting all operations to a finite computational grid, also referred to as *active* computational domain or region, obtained by removing grid cells from the original unbounded computational domain. The cells removed are those that, up to a certain prescribed threshold, do not affect the evolution of the flow field. In particular, the ability of removing cells in this way is a direct consequence of the vorticity, $\boldsymbol{\omega} = \nabla \times \mathbf{u}$, decaying exponentially at large distances from the immersed body. If, for instance, we consider the solution of equation (17a), this is obtained as

$$\mathbf{d}^*(\mathbf{m}) = [\mathbf{G}_C^L * \mathbf{y}](\mathbf{m}), \quad \mathbf{y}(\mathbf{m}) = [-\mathbf{G}^\dagger r_k^i](\mathbf{m}). \quad (21)$$

In equation (21), the term $\mathbf{G}^\dagger r_k^i$ is a discrete approximation of the divergence of the Lamb vector $\nabla \cdot \boldsymbol{\ell}$ at $t = k\Delta t$, where $\boldsymbol{\ell} = \boldsymbol{\omega} \times \mathbf{u}$, while \mathbf{m} denotes the generic discrete location $\mathbf{m} = (p, q, r)$. Therefore, since $\boldsymbol{\omega} \rightarrow 0$ at large distances from the immersed body, it follows that also $\nabla \cdot \boldsymbol{\ell}$ and its discrete counterpart $\mathbf{G}^\dagger r_k^i$ become exponentially small. The domain on which the induced field of equation (21) is calculated is finite and consists of the flow region where $\mathbf{G}^\dagger r_k^i$ is greater than a given positive number (threshold) ϵ . The truncation of the domain using the assumption that the vorticity decays exponentially at large distances from the body induces a discretization error that can be controlled via the arbitrary threshold ϵ , that dictates the dimensions of the finite domain. While using the procedure just outlined, it is necessary to refresh the discrete velocity \mathbf{u} using the discrete vorticity $\mathbf{w} = \mathbf{C}\mathbf{u}$, where \mathbf{C} is the discrete curl operator defined in table 1, in order to obtain a consistent velocity field.

The practical implementation of the domain adaptivity is accomplished by partitioning the formally unbounded domain into a given number of equally sized blocks, $B(\mathbf{m}) = B_{pqr}$. Each block B_b is a Cartesian mesh composed by n_b^3 cells, where we assume blocks that have identical number of cells, n_b , in the three orthogonal directions and where $D_\infty = \cup_{p,q,r=1}^{N_M} B_{pqr}$ is the *active* computational domain. The latter, in turn, is composed by three nested subdomains, $D_{\text{supp}} \subseteq D_{\text{soln}} \subset D_{\text{xsoln}} \subset D_\infty$. The first, D_{supp} , is the union of blocks that define the support of the discrete Poisson problem — e.g. equation (21). The second, D_{soln} , corresponds to the union of blocks tracking the solution fields \mathbf{u} and \mathbf{d} . Finally, the third is a union of blocks surrounding D_{soln} . We additionally define the D_{buff} that is the union of blocks belonging to D_{xsoln} but not to D_{soln} , that is $D_{\text{buff}} = D_{\text{xsoln}} \setminus D_{\text{soln}}$. We also introduce a “mask operator” $\mathcal{M}_Q^\gamma : \mathbb{R}^Q \mapsto \mathbb{R}^Q$ defined as

$$[\mathcal{M}_Q^\gamma \mathbf{q}](\mathbf{m}) = \begin{cases} \mathbf{q}(\mathbf{m}), & \mathbf{q}(\mathbf{m}) \in \text{ind}[B] \text{ and } B(\mathbf{m}) \in D_\gamma, \\ 0, & \end{cases} \quad (22)$$

where, $\text{ind}[B]$ is the list of all indices of the unbounded Cartesian grid associated with block B . The mask operators are used to formally define a operations performed on mesh object Q and domain D_γ . For instance, the operation $\mathbf{G}\mathbf{d}$ over the domain D_{soln} is written as $\mathcal{M}_{\mathcal{F}}^{\text{soln}} \mathbf{G} \mathcal{M}_C^{\text{soln}} \mathbf{q}$.

The first step to construct the adaptive domain algorithm is to define how the support or source domain D_{supp} and the solution or target domain D_{soln} are identified. In this regard, we consider a function W ,

referred to as weight function, that maps an unbounded set of blocks — e.g. D_∞ — to an unbounded set of positive scalars defined on the grid, referred to as thresholds. By using the function W just introduced, the support and solution domains are defined as follows

$$D_{\text{supp}} = \{B(\mathbf{m}) : [W_{\text{supp}}(D_\infty)](\mathbf{m}) > \epsilon_{\text{supp}}, \quad \mathbf{m} \in \mathbb{Z}^3\}, \quad (23a)$$

$$D_{\text{soln}} = \{B(\mathbf{m}) : [W_{\text{soln}}(D_\infty)](\mathbf{m}) > \epsilon_{\text{soln}}, \quad \mathbf{m} \in \mathbb{Z}^3\}. \quad (23b)$$

The weight chosen for W_{supp} is such that it reflect the magnitude of the vorticity Cu and of the nonlinear term $\mathbf{G}^\dagger \tilde{\mathbf{N}}(\mathbf{u} - \mathbf{u}_\infty)$ over a given block $B(\mathbf{m})$. More specifically, the IBLGF solver uses the following weight functions:

$$W_{\text{supp}}(D_\infty)](\mathbf{m}) = \max(\mu(\mathbf{m})/\mu_{\text{global}}, \nu(\mathbf{m})/\nu_{\text{global}}), \quad (24)$$

where

$$\mu = \max_{\mathbf{s} \in \text{ind}[B(\mathbf{m})]} \left| [\text{Cu}](\mathbf{m}) \right|, \quad \mu_{\text{global}} = \max_{\mathbf{m} \in \mathbb{Z}^3} (\mu(\mathbf{m})), \quad (25a)$$

$$\nu = \max_{\mathbf{s} \in \text{ind}[B(\mathbf{m})]} \left| [\mathbf{G}^\dagger \tilde{\mathbf{N}}(\mathbf{u} - \mathbf{u}_\infty)](\mathbf{m}) \right|, \quad \nu_{\text{global}} = \max_{\mathbf{m} \in \mathbb{Z}^3} (\nu(\mathbf{m})). \quad (25b)$$

As the solution evolves over time, the domains D_{supp} and D_{soln} also change, providing an adaption of the domain that mirrors the magnitude of the vorticity as defined in equation (24) and (25). In particular, non-negligible source terms are prevented from being advected or diffused outside D_{supp} by recomputing and, when necessary, reinitializing the active domain at the beginning of a given time-step. Note that the tolerances chosen ϵ_{supp} and ϵ_{soln} affect the accuracy of the simulation, as they dictate the dimension of the computational domain

The successive steps consist of refreshing the discrete velocity from the discrete vorticity and proceed with the time-integration in order to compute the solution at the next time-level. Some of the algorithmic and implementation details were omitted here for the sake of brevity. The interested reader can refer to Liska.¹³

IV. Results

In this section, we document the capabilities of the code for both moving and non-moving bodies.

IV.A. Flow past a non-moving axisymmetric bluff body

We consider a flow past an axisymmetric bluff-body at relatively low Reynolds numbers. The geometry is relevant to the automotive industry, as it can be used to investigate the properties of the three-dimensional wake and its influence on the drag. Of particular interest in this case is the presence of a sequence of bifurcations that define different wake regimes. In the following, we perform a qualitative comparison with the results presented by Rigas et al.¹⁸ for three Reynolds numbers. The wake, for each Reynolds number, presents a reflectional symmetry (i.e symmetry with respect to a single plane) and produce a steady ($\text{Re} = 550$), a periodic ($\text{Re} = 600$) and an aperiodic ($\text{Re} = 800$) behavior, respectively. A summary of the parameters adopted for the test cases considered is reported in table 2.

Re	Wake	h	ϵ_{supp}	ds	dt	Re_h
550	Steady/reflectional symmetry	0.0255	0.0025	0.051	0.0085	14
600	Periodic/reflectional symmetry	0.0233	0.0025	0.047	0.0078	14
800	Aperiodic/reflectional symmetry	0.0175	0.0025	0.035	0.0058	14

Table 2. Parameters adopted and wake characteristics for the three flow regimes investigated. In the table, h represent the flow mesh spacing, ϵ is the tolerance adopted for the adaptivity algorithms, ds is the immersed body mesh spacing, dt is the time-step and Re_h is the grid Reynolds number computed as $\text{Re}_h = u_\infty h / \nu = h \text{Re}$, where ν is the kinematic viscosity and u_∞ is the free-stream velocity.

The geometry and surface body mesh of the axisymmetric bluff body is depicted in figure 2. It is possible to see the equally-spaced nature of point distribution of the mesh, aspect that is required by the IBLGF solver. The mesh was obtained using the freely available “distmesh” software, that solves for the equilibrium

of a truss structure, thereby producing an equally-spaced node distribution.¹⁹ We interfaced the original code (written in Matlab) with the IBLGF framework via Python in order to provide a robust and efficient mesh-generation workflow for the IBLGF solver.

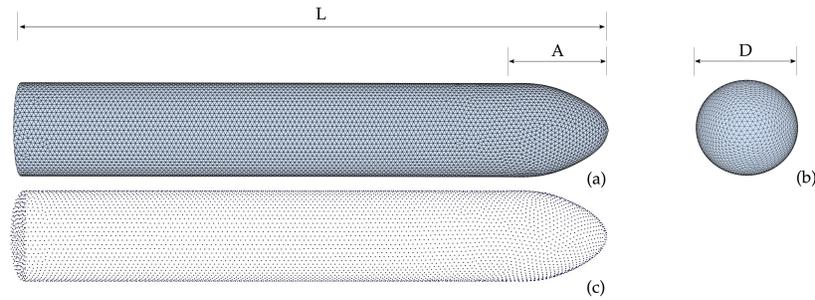


Figure 2. (a) and (b): Side and front views of the mesh for the axisymmetric bluff body. (c): Corresponding immersed boundary grid points.

The test case considered here corresponds to a length-to-diameter ration $L/D = 6.48$. The nose of the axisymmetric bluff body is described by a superellipse and its aspect ratio $AR = A/D$ is equal to 1.

In figure 3, we show the comparison of streamwise vorticity isocontours, $\omega_x = \pm 0.05$, between the original results by Rigas et al.¹⁸ (subfigure 3-(a)) and those obtained using IBLGF (subfigure 3-(b)). The

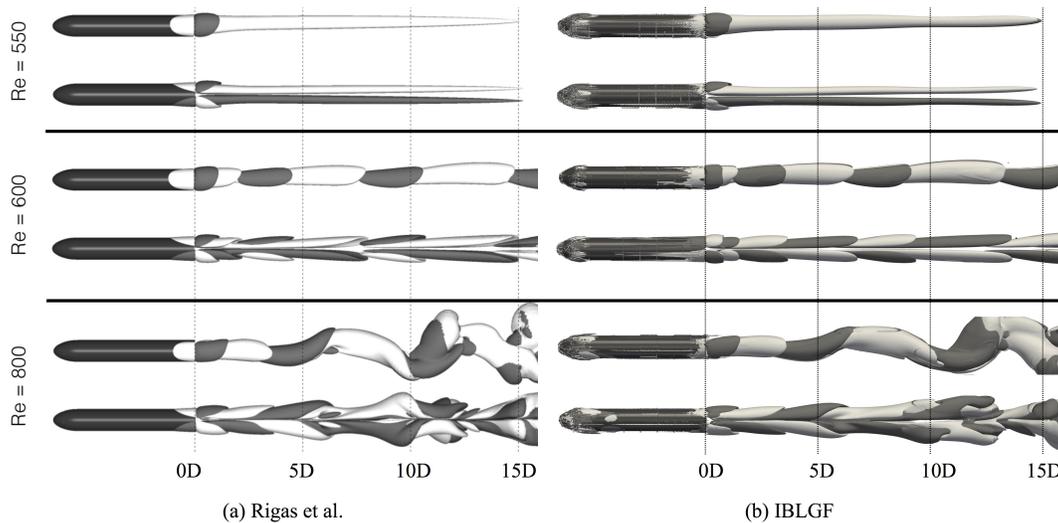


Figure 3. DNS simulation comparisons at Re: 550, 600, 800. Streamwise vorticity contours, $\omega_x = \pm 0.05$, in the wake of the bluff-body; side (top) and plane (bottom) views.

comparison indicates a good agreement for the first two cases (Re = 550 and 600), while some discrepancies can be observed for the third case (Re = 800). These are most likely due to the aperiodic regime that is affected by the initial transient. In our simulations, we used an impulsive start from a uniform flow, while Rigas et al. restarted the simulation from a solution at a lower Reynolds number. Note also that the results obtained using IBLGF are noisier on the body surface, as expected due to the IB treatment adopted.

A final important remark is the number of elements used for the simulation. This was in fact significantly larger than more traditional CFD tools that, because of the body-fitted grid, can exploit a finer mesh in proximity to the bluff body and a coarser mesh farther away. In the case of IBLGF, the flow mesh is uniform and therefore constrained by the resolution required by the boundary layer in proximity to the body. This resulted in an extremely fine mesh in regions of the wake very far downstream, where the number of elements required is significantly less. Specifically, the number of cells was equal to 16 million for Re = 550, to 18 million for Re = 600 and to 34 million for Re = 800. A comparison between the mesh adopted in Rigas et al.¹⁸ and the one adopted by IBLGF is illustrated in figure 4. This aspect emphasizes the need for AMR for the IBLGF solver that is currently under development.

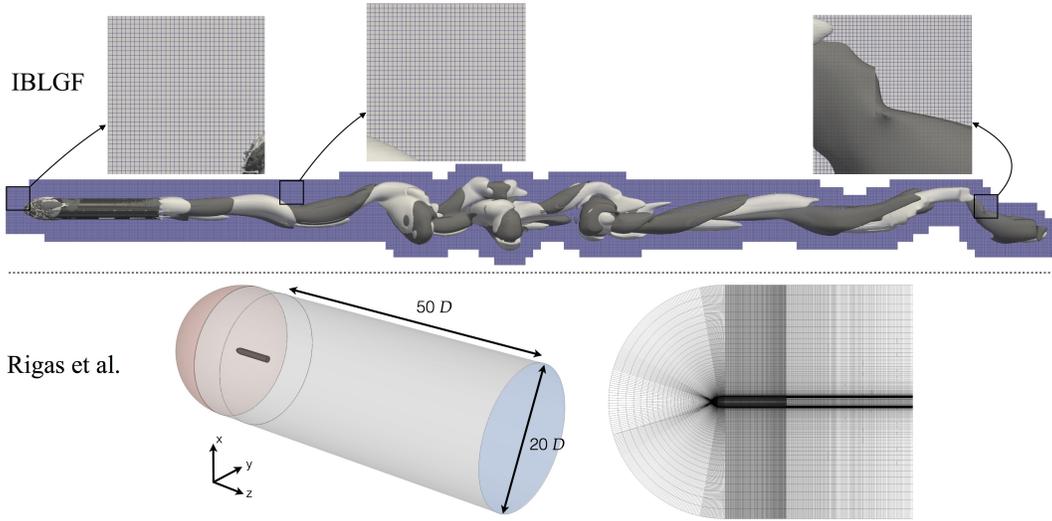


Figure 4. Comparison between the uniform mesh adopted by IBLGF and the body-fitted mesh used by Rigas et al.¹⁸

IV.B. Flow past moving bodies

IV.B.1. Rotating sphere

In this section, we present the first verification of the code for immersed bodies subject to non-rectilinear motion. The test case consists of a flow (aligned with the x -direction) past a sphere that is rotating at a velocity $\omega = 0.5$ with respect to the x -axis. The Reynolds number adopted is $\text{Re} = 250$. A non-rotating sphere is also shown to emphasize the difference in terms of drag coefficient between the moving and non-moving case. The results are compared against those obtained by Kim & Choi.²⁰

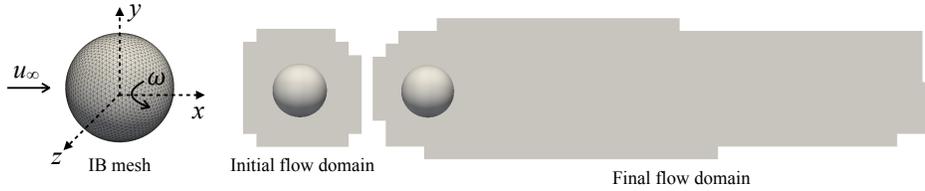


Figure 5. IB mesh, initial and final flow domains.

The details of the simulations are reported in table 3, while the IB mesh along with the initial and final flow domains are depicted in figure 5.

Flow configuration	h	ϵ_{supp}	ds	dt	Re_h	ω
non-moving	0.0189	5e-4	0.0378	0.0062	4.725	0
rotating	0.0189	5e-4	0.0378	0.0062	4.725	0.5

Table 3. Parameters adopted for the flow past a sphere for both the non-moving and rotating cases. In the table, h represent the flow mesh spacing, ϵ_{supp} is the tolerance adopted for the adaptivity algorithms, ds is the immersed body mesh spacing, dt is the time-step, Re_h is the grid Reynolds number computed as $\text{Re}_h = u_\infty h / \nu = h \text{Re}$, where ν is the kinematic viscosity and u_∞ is the free-stream velocity and ω is the velocity of rotation of the sphere.

In figure 6, we show a comparison of the vorticity iso-contour $\omega_z = \pm 0.5$ and the vorticity at the midplane between the rotating and non-rotating sphere test cases (note that ω without subscript indicates angular velocity, while ω_{dir} denotes vorticity with respect to direction dir).

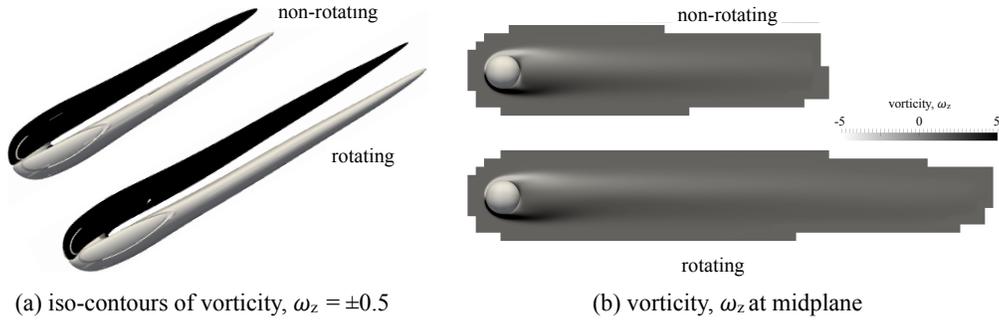


Figure 6. Flow field comparison between the non-moving and rotating sphere in terms of of vorticity ω_z iso-contour (a) and midplane slice (b).

In figure 7, we present the comparison in terms of drag coefficient C_D between the results calculated by means of IBLGF and those by Kim & Choi,²⁰ for both the non-rotating and rotating sphere. The initial transient that can be observed for the IBLGF curves is due to the impulsive initialization of the simulations. After approximately 22 time-units, the IBLGF results agree up to two decimal digits to the reference for both the cases. The relatively slow convergence to the steady state observed in the IBLGF simulations can be alleviated using a non-impulsive initialization of the flow field. The simulation for the non-rotating case was stopped earlier than the rotating one, as C_D converged to the reference value after approximately 20 time-units.

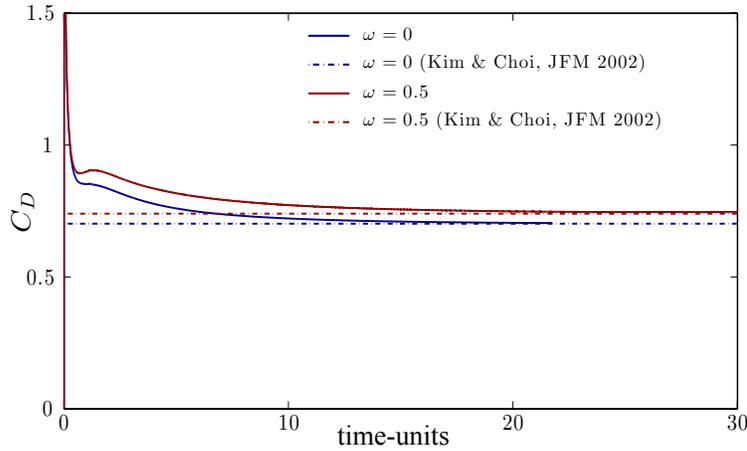


Figure 7. Force comparison between the non-moving and rotating sphere in terms of drag coefficient C_D .

The results presented in this section verify quantitatively the implementation of the IBLGF algorithm for rotating bodies.

IV.B.2. Revolving wing

As a further verification of the code for non-rectilinear body motion, we considered a revolving wing rotating at an angular velocity ω with respect to an axis set at a distance R_1 from the wing root.

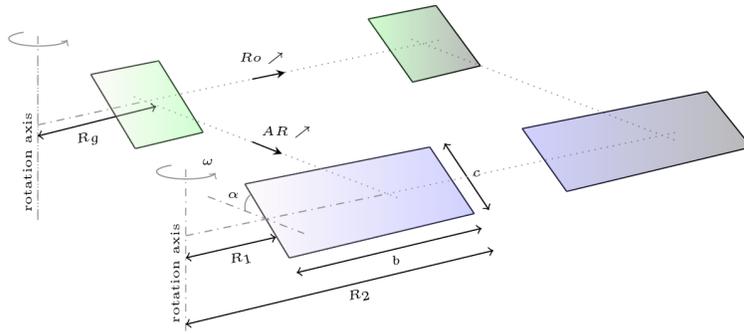


Figure 8. Geometric parameters of the wing and relation with aspect ratio AR and Rossby number Ro .

The distance of the wing tip from the axis is $R_2 = R_1 + b$, where b is the span of the wing that has chord c . A schematic representation of the simulation setup is depicted in figure 8. In the figure, we also show the effect of varying the aspect ratio $AR = b/c$ of the wing and of varying the Rossby number $Ro = V/(\omega c) = R_g/c$, where $V = \omega R_g$, with R_g being the radius of gyration, that, for a generic wing is $R_g = \sqrt{\frac{1}{A} \int_{R_1}^{R_2} r^2 c(r) dr}$. Note that increasing Ro means moving the wing farther away from the rotation axis, while increasing AR means enlarging the span of the wing b , or, equivalently, reducing its chord c . The Reynolds number is defined as $Re = Vc/\nu$, where ν is the kinematic viscosity of the fluid. We carried out three verification tests

1. The first consists of a wing with $R_1 = 0$ and $AR = 4$, hence $Ro = 2.3$. The angle-of-attack α is set to 45° and the Reynolds number is $Re = 577$. The comparison was against data obtained with established tools by Jardin & David.²¹
2. The second is a wing with $R_1 = 0.52c$ and $AR = 1$ where $Re = 520$. In this case we computed the mean lift and drag coefficients over the range $\phi \in [45^\circ - 315^\circ]$ for different angles of attack α and compared against the results obtained by Garmann et al.²²
3. The third corresponds to a wing with $R_1 = 0.5c$ and $AR = 2$. The angle of attack α is set to 45° and the Reynolds number $Re = 2010$. This configuration is similar to that reported in Medina & Jones,²³

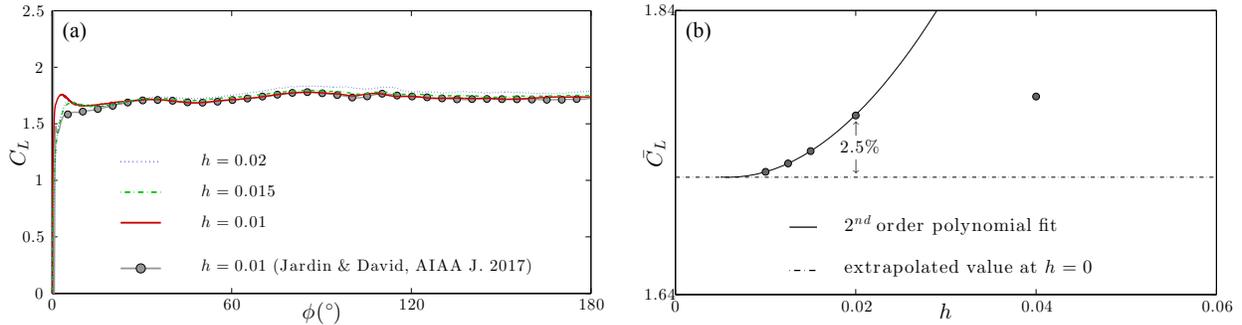


Figure 9. Left, subfigure (a), lift coefficient C_L as a function of the rotation angle ϕ for three grid spacing h . Results are compared with those reported in Jardin & David.²¹ Right, subfigure (b), mean value of C_L computed over the range $\phi \in [5^\circ - 180^\circ]$ plotted as a function of h .

Hereafter, the lift coefficient is defined as $C_L = L/\frac{1}{2}\rho V^2 A$, with L being the lift of the wing (force parallel to the axis of rotation), ρ being the density of the fluid and A being the area of the wing. Analogously, the drag coefficient is defined as $C_D = D/\frac{1}{2}\rho V^2 A$, with D being the drag of the wing.

We first performed grid convergence tests on a reference case where $R_1 = 0$ and $AR = 4$, hence $Ro = 2.3$. The angle-of-attack α is set to 45° and the Reynolds number is $Re = 577$. Figure 9-(a) shows the lift coefficient C_L as a function of the revolution angle ϕ for three grid spacing $h = 0.02, 0.015$ and 0.01 . In addition, results are compared with those obtained by Jardin & David²¹ for a 4% thickness wing operating under similar conditions. It can be seen that C_L is weakly dependent on h for $h \leq 0.02$ and that it converges with h towards values obtained using other well-established methods.²¹

The dependency of C_L on h is further highlighted in figure 9-(b) where the mean value of C_L computed over the range $\phi \in [5^\circ - 180^\circ]$ is plotted as a function of h . It is shown that errors with respect to the extrapolated solution at $h = 0$ is below 2.5% for $h \leq 0.02$. In light of these convergence tests, we set

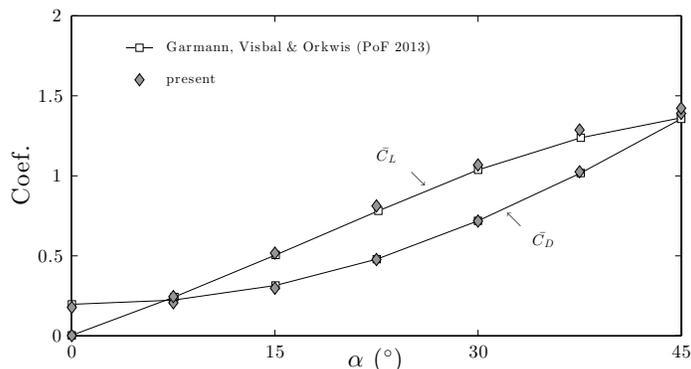


Figure 10. Comparison of lift and drag coefficients, C_L and C_D , against results obtained by Garmann et al.²² for different angles of attack α .

$h = 0.015$ and apply the IBLGF method to the case of a revolving wing with $R_1 = 0.52c$ and $AR = 1$. We set $Re = 520$ and compute the mean lift and drag coefficients over the range $\phi \in [45^\circ - 315^\circ]$ for different angles of attack α . This configuration is similar to that reported in Garmann et al.²² except that they considered a 4% thickness wing and initiated the rotation using Eldredge's function (whereas we apply an impulsive start, i.e. step function). Figure 10 compares results obtained using the IBLGF method with those obtained by Garmann et al.²² Good agreement is observed between both approaches with some discrepancies that may partly arise from wing thickness and acceleration profile. Note that values in Garmann et al.²² are non-dimensionalized using the wing velocity at midspan and have thus been reproduced using the wing velocity at the radius of gyration as a reference scale.

As a final verification result, we applied the IBLGF method to the case of a revolving wing with $R_1 = 0.5c$ and $AR = 2$. The angle of attack α is set to 45° and the Reynolds number $Re = 2010$. This configuration is similar to that reported in Medina & Jones,²³ again with the exception they considered a 5% thickness wing and initiated the rotation using Eldredge's function. Figure 11 compares non-dimensional spanwise vorticity $\omega_z c / V_{tip}$ and velocity $V_z c / V_{tip}$ flow fields obtained using Stereoscopic Particle Image Velocimetry (S-PIV) with those obtained using the present approach. Results are shown for three rotation angles ϕ in a spanwise cross-section located at the wing midspan. It is shown that the present approach accurately captures the flow structure which here consists in a strong leading edge vortex that develops on the upper surface of the wing, together with a strong outboard flow.

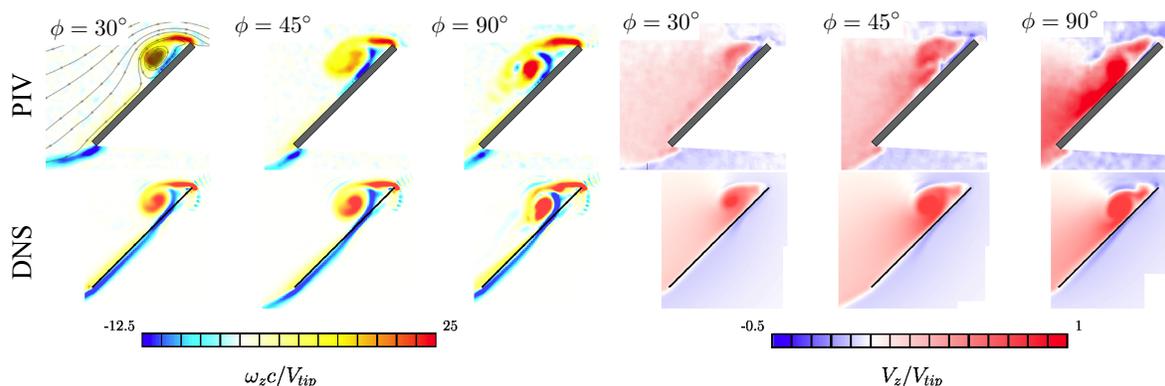


Figure 11. Lift coefficient C_L as a function of the rotation angle ϕ for three grid spacing h . Results are compared with those reported in [1].

For the second verification test case, we also performed a parametric study to investigate the influence of the AR on the lift coefficient C_L .

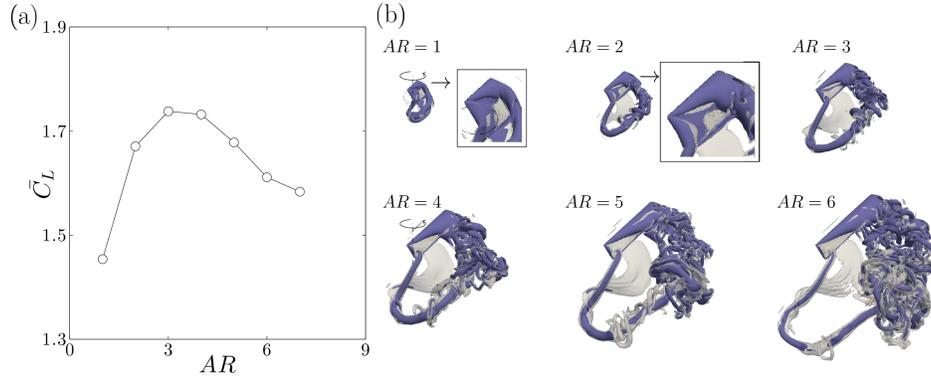


Figure 12. (a) Mean lift coefficient obtained for wings with aspect ratio $AR \in [1 - 7]$ and constant root location $R_1 = 0$. (b) Q -criterion isosurfaces obtained at $\phi = 180^\circ$ for $AR \in [1 - 6]$.

In particular, in figure 12a, we show the mean lift coefficient $\overline{C_L}$ (averaged over the first 180° of revolution) obtained for the first configuration (with aspect ratios ranging from 1 to 7), Figure 12b shows Q -criterion iso-surfaces obtained at the end of the revolving motion ($\phi = 180^\circ$) for each AR . For sufficiently high aspect ratios ($AR > 4$), it is possible to observe different flow structures behind the inboard and the outboard regions. In particular, on the inboard upper surface, there is a conical leading-edge vortex developing that extends three chords away from the wing root. The outer region flow is instead characterized by smaller scale unsteady vortices that indicate local flow instability. This distinction between inboard quasi-steady flow and outboard unsteady flow as well as the extent of the quasi-steady inboard region is consistent with data obtained by Kruyt²⁴ and demonstrate the ability of the code to capture correctly the physics of the problem.

V. Conclusions and future work

In this paper, we presented the capabilities of the immersed boundary lattice Green’s function (IBLGF) algorithm for solving incompressible external aerodynamic flows. We first applied it to a flow past a axisymmetric bluff body at low Reynolds numbers and we showed that the wake was correctly captured by IBLGF.

We successively presented the first verification of the code for moving bodies. The two test cases taken into account were a flow past a rotating sphere and a flow past a revolving wing. In both cases the comparisons with results present in the literature were in good agreement and provided a solid verification of the algorithm implementation.

We also highlighted that the current implementation of the code does not provide AMR, thus the mesh is over-resolved in various regions of the flow field, especially those farther downstream, where at least an order of magnitude less elements is required. AMR capabilities are currently under development. In particular, we aim to keep the same algorithmic building blocks outlined in section III, with the exception that we will operate on M Cartesian meshes, each having a different mesh resolution. We denote these different meshes as \mathcal{M}_ℓ , with $\ell = 1, \dots, M$, where the first mesh, $\ell = 1$, is the coarsest while the last, $\ell = M$, is the finest. The cell dimensions are constrained to have a factor two between consecutive coarser and finer meshes, that is: $(\Delta x)_{\mathcal{M}_\ell} / (\Delta x)_{\mathcal{M}_{\ell-1}} = 2$. This implies that for each face of a coarser cell that is adjacent to a finer one, there are four contributions in terms of velocity vector and pressure. The various meshes will need to interact with each other. More specifically, we will need to redefine the 2^{nd} -order finite volume operators introduced in section III.A in order to handle non-conformal meshes. The implementation of AMR capabilities in the code will enable more mesh-resolution-efficient simulations and reduce significantly the computational costs of the simulations presented in this paper.

In parallel with the development of AMR, we are also implementing the stretched-vortex LES model^{25,26} to target high-Reynolds number flows. The stretched-vortex model assumes the subgrid vortices to be aligned in the direction of the eigenvector associated with the largest positive eigenvalue of the strain-rate tensor. An assumed Kolmogorov energy spectrum for the subgrid vortices and the local dissipation balance together give the closure to the filtered Navier-Stokes equations.

The development of AMR and LES capabilities, along with the favorable properties of the IB method — i.e. no need for re-gridding in presence of moving bodies — and of the LGFs — i.e. ability of using a very compact computational domain without the need for far-field boundary conditions — make the IBLGF algorithm potentially promising for CFD simulations involving complex external aerodynamics problems at high Reynolds number with moving surfaces.

Acknowledgments: This research was supported in part by a grant from ONR (N00014-16-1-2734) under the direction of Mr. Ken Iwanski.

References

- ¹Liska, S. and Colonius, T., “A fast immersed boundary method for external incompressible viscous flows using lattice Green’s functions,” *arXiv preprint arXiv:1604.01814*, 2016.
- ²Moin, P. and Mahesh, K., “Direct numerical simulation: a tool in turbulence research,” *Annual Review of Fluid Mechanics*, Vol. 30, No. 1, 1998, pp. 539–578.
- ³Sagaut, P., *Large eddy simulation for incompressible flows: an introduction*, Springer Science & Business Media, 2006.
- ⁴Cadieux, F., Domaradzki, J. A., Sayadi, T., and Bose, S., “Direct numerical simulation and large eddy simulation of laminar separation bubbles at moderate Reynolds numbers,” *Journal of Fluids Engineering*, Vol. 136, No. 6, 2014, pp. 060902.
- ⁵Mengaldo, G., De Grazia, D., Moxey, D., Vincent, P., and Sherwin, S., “Dealiasing techniques for high-order spectral element methods on regular and irregular grids,” *Journal of Computational Physics*, Vol. 299, 2015, pp. 56–81.
- ⁶Mittal, R. and Iaccarino, G., “Immersed boundary methods,” *Annual Review of Fluid Mechanics*, Vol. 37, 2005, pp. 239–261.
- ⁷Seo, J. and Mittal, R., “A sharp-interface immersed boundary method with improved mass conservation and reduced spurious pressure oscillations,” *Journal of Computational Physics*, Vol. 230, No. 19, 2011, pp. 7347–7363.
- ⁸Roma, A., Peskin, C., and Berger, M., “An adaptive version of the immersed boundary method,” *Journal of Computational Physics*, Vol. 153, No. 2, 1999, pp. 509–534.
- ⁹Griffith, B., Hornung, R., McQueen, D., and Peskin, C., “An adaptive, formally second order accurate version of the immersed boundary method,” *Journal of Computational Physics*, Vol. 223, No. 1, 2007, pp. 10–49.
- ¹⁰Vanella, M., Posa, A., and Balaras, E., “Adaptive mesh refinement for immersed boundary methods,” *Journal of Fluids Engineering*, Vol. 136, No. 4, 2014, pp. 040909.
- ¹¹Liska, S. and Colonius, T., “A parallel fast multipole method for elliptic difference equations,” *Journal of Computational Physics*, Vol. 278, 2014, pp. 76–91.
- ¹²Liska, S. and Colonius, T., “A fast lattice Green’s function method for solving viscous incompressible flows on unbounded domains,” *Journal of Computational Physics*, Vol. 316, 2016, pp. 360–384.
- ¹³Liska, S., *Fast lattice Green’s function methods for viscous incompressible flows on unbounded domains*, Ph.D. thesis, California Institute of Technology, 2016.
- ¹⁴Taira, K. and Colonius, T., “The immersed boundary method: a projection approach,” *Journal of Computational Physics*, Vol. 225, No. 2, 2007, pp. 2118–2137.
- ¹⁵Colonius, T. and Taira, K., “A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions,” *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, No. 25, 2008, pp. 2131–2146.
- ¹⁶Bhalla, A. S., Bale, R., Griffith, B., and Patankar, N., “A unified mathematical framework and an adaptive numerical method for fluid–structure interaction with rigid, deforming, and elastic bodies,” *Journal of Computational Physics*, Vol. 250, 2013, pp. 446–476.
- ¹⁷Kallemov, B., Bhalla, A., Griffith, B., and Donev, A., “An immersed boundary method for rigid bodies,” *Communications in Applied Mathematics and Computational Science*, Vol. 11, No. 1, 2016, pp. 79–141.
- ¹⁸Rigas, G., Esclapez, L., and Magri, L., “Symmetry breaking in a 3D bluff-body wake,” *arXiv preprint arXiv:1703.07405*, 2017.
- ¹⁹Persson, P.-O. and Strang, G., “A simple mesh generator in MATLAB,” *SIAM review*, Vol. 46, No. 2, 2004, pp. 329–345.
- ²⁰Kim, D. and Choi, H., “Laminar flow past a sphere rotating in the streamwise direction,” *Journal of Fluid Mechanics*, Vol. 461, 2002, pp. 365–386.
- ²¹Jardin, T. and David, L., “Root cut-out effects on the aerodynamics of a low aspect ratio revolving wing,” *AIAA Journal*, Vol. in press, 2017.
- ²²Garmann, D., Visbal, M., and Orkwis, P., “Three-dimensional flow structure and aerodynamic loading on a revolving wing,” *Physics of Fluids*, Vol. 25, No. 3, 2013, pp. 034101.
- ²³Medina, A. and Jones, A., “Leading-edge vortex burst on a low-aspect-ratio rotating flat plate,” *Physical Review Fluids*, Vol. 1, No. 4, 2016, pp. 044501.
- ²⁴Kruyt, J. W., van Heijst, G. F., Altshuler, D. L., and Lentink, D., “Power reduction and the radial limit of stall delay in revolving wings of different aspect ratio.” *J. R. Soc. Interface*, Vol. 12, 2015.
- ²⁵Voelkl, T., Pullin, D., and Chan, D., “A physical-space version of the stretched-vortex subgrid-stress model for large-eddy simulation,” *Physics of Fluids*, Vol. 12, No. 7, 2000, pp. 1810–1825.
- ²⁶Chung, D. and Pullin, D., “Large-eddy simulation and wall modelling of turbulent channel flow,” *Journal of Fluid Mechanics*, Vol. 631, 2009, pp. 281–309.