Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 193 (2003) 136–158
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Advances in direct numerical simulations of 3D wall-bounded flows by Vortex-in-Cell methods

G.-H. Cottet [a,*], P. Poncet [b]

[a] LMC IMAG, Université Joseph Fourier, BP 53, 38041 Grenoble, Cedex 9, France
[b] Laboratoire MIP, INSA, 135 avenue de Rangueil, 31077 Toulouse, Cedex 4, France

Received 3 February 2003; received in revised form 5 August 2003; accepted 6 August 2003

Abstract

This paper is devoted to the design of Vortex-In-Cell (VIC) methods for the direct numerical simulations of wall-bounded flows. A first method using body-fitted grid is presented in the particular case of a cylinder wake. This method, which has been used in [Phys. Fluids 14(6) (2002) 2021] to investigate the effect on the wake topology of cylinder rotations, is an extension of the VIC method presented in [J. Comput. Phys. 175 (2002) 702] for periodic geometries. Features of the method that are specific to wall-bounded geometries – interpolation operators, field calculations and vorticity flux formulas to enforce no-slip boundary conditions – are described in details. The accuracy of the method in the calculation of the body forces is investigated by comparisons with experiments and benchmark calculations. A second class of methods is in the spirit of the immersed boundary methods. The paper in particular shows that the no-slip conditions are very naturally handled by the vorticity flux formulas, independently of the relative locations of the particles and the body. Numerical experiments on the test-case of a ring impinging on a cylinder suggest that the method is second-order accurate.

1. Introduction

The direct numerical simulation of three-dimensional (3D) bluff-body flows remains a challenging problem in CFD. Even for rather simple geometries, like a cylinder, the need for accuracy and robustness is very demanding for classical grid-based techniques. While accuracy often dictates the use of non-dissipative finite-difference or spectral element schemes, stability impose constraints, on the compatibility of the grid and flow topologies and on the time-step values, that can substantially slow down the methods.

By contrast, particle methods, when they use the appropriate tools, allow to some extent to by-pass the usual accuracy–stability dilemma. The advection part of the equations indeed relies on the advection of

* Corresponding author. Fax: +33-4-76-63-12-63.
E-mail address: Georges-Henri.Cottet@imag.fr (G.-H. Cottet).

0021-9991/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.jcp.2003.08.025

particle, and thus is linearly unconditionally stable. Nonlinear stability requires that particles do not collide, something which is guaranteed as long as the time-step does not exceed the time-scale on which the flow is strained. This condition reads $\Delta t \leqslant C|\nabla u|^{-1}$ and thus does not involve the mesh size. In practice, for well-resolved calculations where one wishes to use grid-sizes small enough to accurately capture high strain regions, this condition is often much less demanding than classical CFL type conditions. As for the diffusion part of the equation, deterministic particle methods are based on explicit solvers that are stable under finite-difference like conditions of the type $\nu\Delta t \leqslant Ch^{-2}$, where ν is the viscosity and h is the mesh size. For moderate to high Reynolds numbers and affordable resolutions, this condition is generally not a severe limitation in 3D calculations.

Systematic comparisons on a variety of 2D flows with non-dissipative finite-difference schemes have shown that, in many cases, time-step limitations are indeed far less restrictive for particle methods than for Eulerian methods, leading to substantial savings.

Concerning 3D computations, recent comparisons [4] with spectral methods for periodic laminar and turbulent flows have given some insight into the accuracy and the subgrid behavior of particle methods. For 3D wall-bounded flows, vortex methods have been successfully used to compute vortex–wall collision [21] and, more recently, the flow past a sphere at various Reynolds numbers [24]. In the first case, the method was a Vortex-In-Cell (VIC) method, combining Lagrangian transport of particles with Eulerian field calculation, while in the second case the authors used a totally grid-free vortex method, based on fast N body solvers for fields evaluation.

Our goal in this paper is twofold. First, we show that Vortex-In-Cell methods are, in terms of accuracy/cost balance, a viable alternative to Eulerian methods for DNS computations of cylinder wakes. Secondly, we propose and validate an immersed boundary Vortex-In-Cell method to handle more complex geometries.

Concerning the first point, one reason for focusing on cylinder wakes is that this is a well-documented flow (see [1,18,19] for example), in particular since the experimental work of Williamson [29]. For this flow however many open questions remain, due to the limitations of current CFD solvers. Some of these questions regarding the bi-dimensionalization of 3D wakes under cylinder rotations, are addressed elsewhere [8,26,27] and the present paper focuses on the numerical techniques underlying these numerical simulations.

For cylinder wakes, and more generally bluff-body flows, beside the time-step constraints already mentioned, Eulerian methods face additional difficulties in the treatment of outflow boundary conditions. Nonlinear stability requires special care there, sometimes at the expense of mesh refinement, although accuracy should not be a concern in this part of the flow. If grid-free particle methods are evidently free of these difficulties, this is at the expense of using time-consuming N body solvers. As for vortex in cell methods, since fields are computed on a grid, they have to introduce artificial boundary conditions on the outer parts of the computational box for the calculation of the fields. One goal of the present study is in particular to check whether accuracy, in the drag computation in particular, or stability impose drastic conditions on the size of the computational box for VIC methods. Another point of concern about vortex methods for wall-bounded flows is the consistent treatment of no-slip boundary conditions. Vorticity flux formulas have been demonstrated (see in particular [16]) to be the appropriate formulation for vortex methods to handle in two dimensions these boundary conditions. For 3D flows, an extension of this formulation is given in [5] for flat boundaries. In the present paper, we show that in the case of more general geometries the curvature of the boundary modifies the Neumann into a Robin type boundary condition for the azimuthal component of the vorticity.

Concerning the second point addressed in this paper, the method we propose is in the spirit of immersed boundary schemes originally proposed by Peskin [22] and recently revisited in the context of finite-difference techniques [9]. The general idea is to avoid technical difficulties in generating a body-fitted grid around a possibly moving complex 3D obstacle, by using for instance a finite-difference solver on a fixed Cartesian

grid overlapping with the boundary of the obstacle. Boundary conditions are enforced through a forcing term in the right-hand side of the Navier–Stokes equations.

The interpolation on the grid of this forcing term, which by nature is singular and has support on the boundary, is in finite-difference methods a critical point that conditions the overall accuracy. For vortex methods, the situation is very different. No-slip boundary conditions are enforced by sources of vorticity which are located on the boundary. The accuracy of this procedure depends on the grid resolution underlying the distributions of source boundary points and flow particles, but not on the relative locations of these two sets of points. The vorticity flux of boundary sources onto the flow particles somehow plays the role of interpolation formulas needed in finite-difference methods. The method we propose is strongly based on this remark and thus only introduces minor changes – including for 3D geometries – over a method which would use a body-fitted grid. In that respect, our method differs significantly from the vortex method designed in [23,24] (see also [28]), where particles close to the boundaries are monitored and given special treatments.

The outline of the paper is as follows. In Section 2, we describe our method with body-fitted particles and grid for cylinder wake calculations. We detail the Poisson solver used to compute velocity and strain on the grid, and the vorticity boundary conditions needed to satisfy no-slip boundary conditions. We also give some indications on the cost of the method compared to a purely grid-free particle method. Our method is then validated by systematic inspection of drag curves for moderate Reynolds numbers. In Section 3 we turn to the immersed boundary vortex method. We describe the algorithms used to satisfy no-through flow and no-slip boundary conditions. The numerical validation is performed on the test-case of a ring-cylinder collision. Finally Section 4 is devoted to concluding remarks.

2. A VIC method for the computation of 3D cylinder wakes

The general idea behind VIC methods, and more generally Particle-In-Cell methods, is to use particles to transport conservative quantities and grid-based formulas to compute fields. For the incompressible Navier–Stokes equations written in the vorticity–velocity form

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + (\mathbf{u}\cdot\nabla)\boldsymbol{\omega} - (\boldsymbol{\omega}\cdot\nabla)\mathbf{u} - \nu\Delta\boldsymbol{\omega} = 0 \tag{1}$$

particles thus carry vorticity, while velocity and strain are computed on an Eulerian grid using Poisson solvers. The reason for using this strategy to compute the fields, instead of direct, Biot–Savart law inspired, integral formulas, is that even the fastest summation formulas are in many practical situations at least one-order of magnitude slower than FFT-based current Poisson solvers. This will appear clearly on the timings shown later in in this section.

The overall algorithm classically consists of alternating advection and diffusion equations. Convection is done by pushing particles with their local velocities and updating their strength to account for the local vorticity stretching (computed with centered fourth-order finite difference schemes).

Diffusion is done by a particle strength exchange (PSE) algorithm with appropriate Neumann boundary condition to cancel the slip resulting for the advection step (see for instance [5,16]).

To preserve the accuracy of Particle methods for long time simulations, it has long been observed that frequent regridding of particles on regular locations is necessary (see [14] for a convergence study of remeshing). In our algorithm, remeshing is done at every time-step just before diffusion. This allows to use the PSE scheme with formulas normalized on the basis of discrete moments, and thus avoids quadrature errors in the diffusion approximation (see [6,25] for instance). When there is a solid boundary, in the body-fitted method described in this section the grid fits with the solid boundary while particle are initialized and remeshed on a staggered grid.

Each time step of the algorithm can be summarized in this way.

Convection step:

- interpolation of vorticity from particles to grid,
- computation of velocity and strain on the grid,
- interpolation of velocity and strain on particles,
- update of particle locations and strengths.

Particle remeshing: Interpolation of particle strengths on regular locations.

Diffusion step: PSE scheme and vorticity flux formulas.

Note that, as already mentioned, the time step for diffusion and for convection/remeshing are constrained by different stability conditions and need not be the same. In practice several sub-steps of convection, using fourth-order Runge–Kutta time-stepping, may be done inside one diffusion step.

The overall structure of the algorithm has been described in a number of references (see in particular [4,6]) and we focus here on the particular aspects of the algorithm related to the cylindrical geometry: we first discuss the interpolation formulas needed to exchange information between particles and grid and to remesh particles; then we describe the Poisson solver used to compute velocity values on the grid; finally we derive vorticity flux formulas which translate the no-slip boundary conditions. The end of the section is devoted to the numerical validation of the algorithm on 2D and 3D wake simulations.

2.1. Interpolation formulas for particle–grid mapping and particle remeshing

Let us first give the notations corresponding to the geometry of a bluff-body flow. Ω denotes the computational domain, extending from the boundary of the obstacle, denoted by Γ_b, to the outer boundary, denoted by Γ_∞. In the case of a flow past a cylinder, we will denote by subscripts r, θ and z, respectively, radial, azimuthal and spanwise field components. We will assume L-periodic boundary conditions in the cylinder axis direction and the computational domain extends from $r = R_b$ to $r = R_\infty$.

Interpolation formulas are based on convolutions with a smooth kernel. The kernel used in the present work is based on the following 1D function which is third-order – in the sense that it preserves the three first moments of the distribution, twice continuously differentiable and symmetric (see [5]):

$$\zeta(x) = \begin{cases} (3x^3 - 5x^2 + 2)/2 & \text{if } 0 \leqslant x \leqslant 1, \\ (2-x)^2(1-x)/2 & \text{if } 1 \leqslant x \leqslant 2, \\ 0 & \text{if } x \geqslant 2. \end{cases} \tag{2}$$

Rescaling this function at a grid-size ε yields the following expression:

$$\zeta_\varepsilon(x) = \frac{1}{\varepsilon}\zeta\left(\frac{x}{\varepsilon}\right). \tag{3}$$

To account for cylindrical geometries, the interpolation is based on tensor products of this function in cylindrical coordinates. Assuming the same grid size in radial, angular and azimuthal directions, the redistribution of a given function f, extended by periodicity in the angular and azimuthal directions, into a function $\tilde{f}$ is given by

$$\tilde{f}(r,\theta,z) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\int_{R_0}^{R_\infty} f(s,\xi,u)\Lambda_r(r-s)\zeta_\varepsilon(\theta-\xi)\zeta_\varepsilon(z-u)s\,\mathrm{d}s\,\mathrm{d}\xi\,\mathrm{d}u, \tag{4}$$

where the subscript r in Λ_r means that the shape of the kernel depends on the location. The kernel $\Lambda_r = \zeta_\varepsilon$ is chosen unless particles and grid points are close to the boundary. For the second layer of grid points (corresponding to $r = R_b + \varepsilon$) the kernel ζ is used with ghost particles inside the body, carrying symmetric weights. This amounts to replacing ζ by

$$\Lambda_{R_b+\varepsilon}(r) = \begin{cases} \zeta(r) & \text{if } r \leqslant 0, \\ \zeta(r) + \zeta(2-r) & \text{if } 0 \leqslant r \leqslant 1, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

This kernel is second-order accurate.

For the first layer of points (on the cylinder) we use the following one-sided interpolation formula:

$$\Lambda_{R_b}(r) = \begin{cases} r^2 + 4r + 13/4 & \text{if } -(4+\sqrt{3})/2 \leqslant r \leqslant 0, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

This kernel has been chosen because it has the property to preserve circulation and linear impulse when particles and grid points lie on staggered grids. We recall that this is the procedure selected to initialize and remesh particles.

Finally, the interpolation of a quantity f_p carried by particles located at (r_p, θ_p, z_p) and whose volume is v_p is given by

$$\tilde{f}(r_p, \theta_p, z_p) = \sum_q f_q \Lambda_{r_p}(r_p - r_q)\zeta_\varepsilon(\theta_p - \theta_q)\zeta_\varepsilon(z_p - z_q)v_q. \tag{7}$$

Note that this summation involves image particles in the corresponding directions to take in account angular and spanwise periodicity. This formula easily extends to the case when different grid sizes are used in the three directions.

Formula (7) is used at three stages of the algorithm: when particle vorticity is interpolated on a fixed cylindrical grid where velocity are evaluated (see next section), when field values are interpolated back to particles, and finally to remesh an eventually distorted particle distribution into a fresh, regular distribution. As we already mentioned, to maintain accuracy of the particle discretization, in all our calculations we

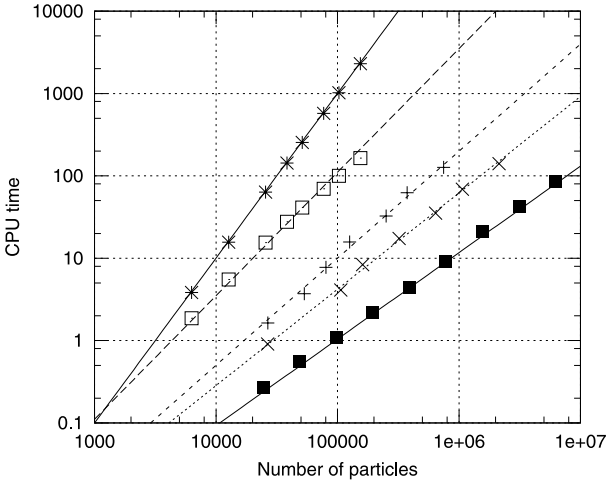

Fig. 1. Comparison of CPU time for the evaluation of particle velocities in a grid-free method based of on fast summation algorithm [17] and in the present VIC method based on a Cartesian or polar Poisson solver: (∗) direct summation [17], (□) fast multipole first-order calculation [17], (×) VIC method on a cylindrical grid filled with 65% particles, (+) VIC method on a cylindrical grid filled with 25% particles, and (■) VIC method on a Cartesian grid filled with 100% particles.

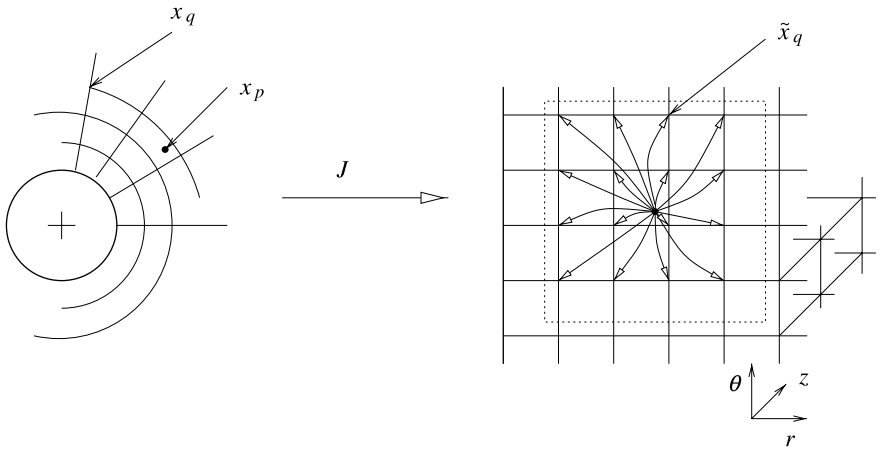

Fig. 2. Interpolation between particles (located at x_p) and grid points (located at x_q), in cylindrical coordinates (left picture) and mapped coordinates (right picture).

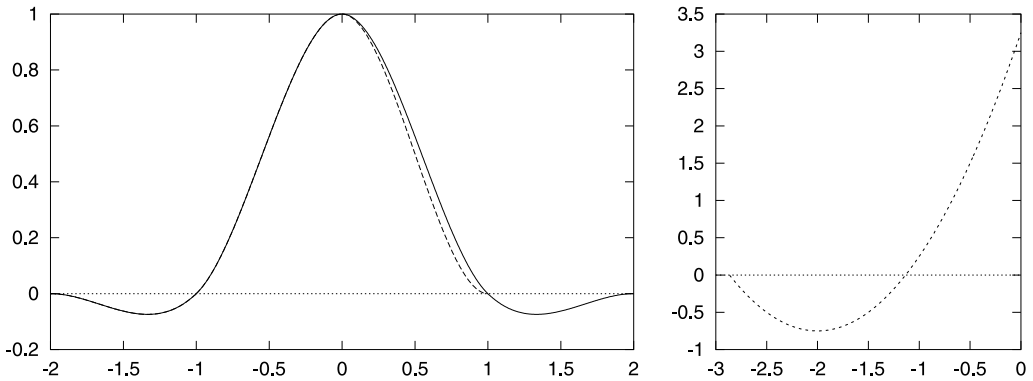

Fig. 3. Kernels used for remeshing and interpolation: ζ (——), $\Lambda_{R_b+\varepsilon}$ (— — —, left picture), and Λ_{R_b} (- - -, right picture).

remesh the particle at every time-step before the diffusion step. The remeshing procedure and formulas (5) and (6) are illustrated on Figs. 2 and 3.

2.2. Velocity evaluations

Once vorticity has been assigned to the grid, the velocity is computed according to the Helmholtz decomposition

$$\mathbf{u} = \bar{\mathbf{u}} + \nabla \times \boldsymbol{\psi} + \nabla\phi, \tag{8}$$

where $\bar{\mathbf{u}}$ is the potential flow around the cylinder with prescribed value at infinity. One then has $\nabla \times \mathbf{u} = \boldsymbol{\omega}$ and $\nabla \cdot \mathbf{u} = 0$ provided the stream function $\boldsymbol{\psi}$ and the potential ϕ satisfy the following Poisson equations:

$$-\Delta\boldsymbol{\psi} = \boldsymbol{\omega} \quad \text{in } \Omega, \tag{9}$$

$$\nabla \cdot \boldsymbol{\psi} = 0 \quad \text{in } \Omega, \tag{10}$$

$$\Delta\phi = 0 \quad \text{in } \Omega. \tag{11}$$

The boundary conditions to complement this system are adjusted to ensure no-through flow on the cylinder and the artificial boundary condition $\mathbf{u} = \overline{\mathbf{u}}$ on the outer limit of the computational domain.

More precisely our solution procedure is as follows. We first compute ψ_x and ψ_y solutions to $-\Delta\psi_x = \omega_x$, $-\Delta\psi_y = \omega_y$ with periodic boundary conditions in the z- and θ-direction, and homogeneous Dirichlet boundary conditions in the radial direction. Then we compute the remaining component ψ_z, satisfying $-\Delta\psi_z = \omega_z$ with periodic boundary conditions in the z- and θ-direction, and the following Dirichlet boundary condition in the radial direction:

$$\psi_z(r,\theta,z) = -\int_0^z \frac{\partial \psi_r}{\partial r}(r,\theta,s)\mathrm{d}s \quad \text{for } r = R_\mathrm{b} \text{ and } r = R_\infty. \tag{12}$$

With these boundary conditions, one has

$$\nabla \cdot \boldsymbol{\psi} = 0 \quad \text{on } \Gamma_\mathrm{b} \text{ and } \Gamma_\infty$$

and periodic boundary conditions in z. Since

$$\Delta(\nabla \cdot \boldsymbol{\psi}) = \nabla \cdot \boldsymbol{\omega} \quad \text{in } \Omega$$

and $\boldsymbol{\omega}$ is divergence-free, this implies that

$$\nabla \cdot \boldsymbol{\psi} = 0 \quad \text{in } \Omega.$$

It remains to compute the scalar potential ϕ. In order to impose the correct boundary condition on the cylinder, we require the following Neumann type boundary condition:

$$\frac{\partial \phi}{\partial \mathbf{n}} = -(\nabla \times \boldsymbol{\psi}) \cdot \mathbf{n} \quad \text{on } \Gamma_\mathrm{b} \text{ and } \Gamma_\infty, \tag{13}$$

to complement the Poisson equation (11) for ϕ. Fig. 4 summarizes the procedure to impose the no-through flow boundary condition.

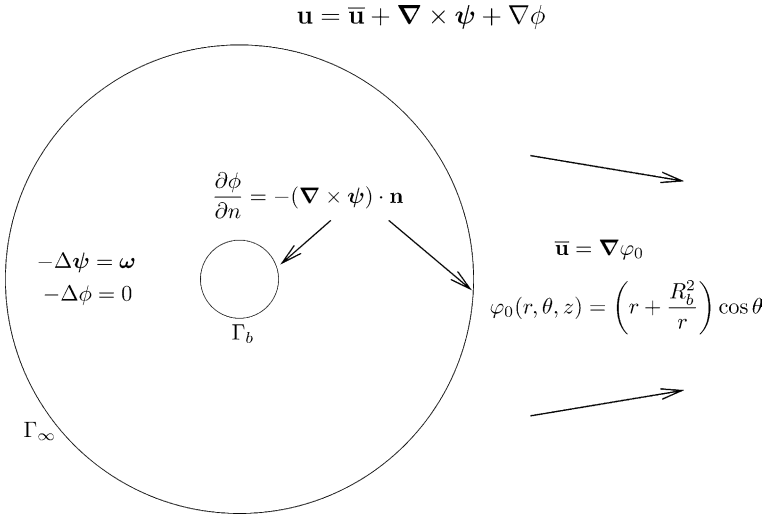

Fig. 4. Body and far-field boundary conditions for velocity field.

Note that, if

$$\int_0^L \frac{\partial \psi_r}{\partial r}(r,\theta,s)\mathrm{d}s \neq 0$$

the periodic boundary condition combined with (12) may create a singularity for ψ_z, and thus for ϕ, near the corners of Ω which can affect the accuracy in the finite-difference calculations of velocities. In [25], an algorithm to remove this singularity is derived, based on the resolution of additional 2D Poisson equations. However, in practical calculations, spurious effects of this singularity have never been observed and the original algorithm based on formula (12) without correction has been found satisfactory. It is also important to notice that the far-field boundary condition consists of imposing $\mathbf{u} = \overline{\mathbf{u}}$ at the outer boundary. This may seem a rather crude approximation, in particular compared to the exact far-field condition implicitly used in grid-free vortex methods. Nevertheless, the numerical results shown below (see Table 1) demonstrate that this boundary condition allows to obtain convergent results on the body with computational domains significantly smaller, in the streamwise direction, than those currently used in finite-difference methods.

One nice feature of using a scalar potential to compute velocities, is that boundary conditions do not couple the computations of the three components of the stream functions. This allows to use simple scalar Poisson solvers. In our simulations we used classical Fishpack package solvers.

To give an idea of the computational cost of the overall numerical procedure to compute the velocities, including interpolations and Poisson solvers, we show in Fig. 1 a comparison of CPU times for our method compared to the fast summation algorithm based on a tree-code given in [17]. For a sake of fairness, a scaling factor of 4, based on the CPU time needed for direct summation methods on the different platforms, has been applied to account for the difference in processor speeds between the SGI 75 MHz processor used by these authors and the Alpha 500 MHz processor we were using. Although the development and implementation of 3D fast solvers is a rapidly growing field, we believe that these comparisons give a good indication of the speed up offered by VIC methods, except when vorticity is strongly localized (which was the case in the vortex sheet calculations of [17]).

In practical implementations of vortex methods, an additional speed up factor can be obtained from the following remark: the convection of particles is most often done with a multi-step time-stepping. In a Biot–Savart type algorithm, velocities are in general recomputed at every sub-step since particles have moved. In all our VIC calculations, we have observed that it is possible to compute only once per time step the grid velocities without noticeably deteriorating the accuracy. Particle motions during the substeps have only to be taken into account when grid velocities are interpolated on particle locations. When a fourth-order Runge–Kutta time-stepping is used, this introduces another significant speed up (note that this remark also applies to Biot–Savart codes: in that case a fast summation, instead of a Poisson solver, would be used to compute velocity and strain on regular grid points).

To finish with these comparisons, let us again stress the fact that the speed up of VIC methods on Biot–Savart based methods is very much problem dependent. In case particles occupy only a very small portion

Table 1
2D drag coefficients and Strouhal numbers for $Re = 400$ and various domain sizes

$(R_\infty - R_b)/R_b$	N_{part} ($\times 10^{-3}$)	$\overline{C_D}$	S_t	$\widehat{C_L}$
2π	4.60	1.5270	0.2253	1.135
4π	8.29	1.4205	0.2247	1.130
8π	16.66	1.4080	0.2237	1.125
16π	33.12	1.4075	0.2232	1.123

N_{part} is the mean number of particles once oscillatory regime is established.

(say less that 10%) of a computational box that would be required in a VIC method, Biot–Savart inspired methods may become comparable or even more effective than VIC methods. Particular cases of localized vorticity are 1D or 2D vortex sheets (like in the calculations of [17]) or when one desires to follow a wake very far downstream (like in reference [24]). In our case, we were interested by body forces for a cylinder wake in a computational grid which was filled by approximately 25% of particles, and Biot–Savart methods were clearly outperformed by VIC methods (by a factor of about 20 according to Fig. 1) .

2.3. No-slip boundary condition and vorticity flux formulas

In vortex methods, the no-slip boundary condition is classically enforced by the creation of a vortex layer in the vicinity of the boundary [2]. In a fractional step algorithm, this vortex layer is designed to cancel the slip resulting from previous advection and diffusion steps. A clear-cut mathematical definition of this method is based on a vorticity flux formula – or Neumann type boundary conditions – in the vorticity diffusion equation. In two dimensions, if Δt is the diffusion time-step and $\mathbf{u}\cdot\boldsymbol{\tau}$ the residual slip resulting from the advection of particles and the PSE scheme, this formula reads

$$\frac{\partial\omega}{\partial t} - \nu\Delta\omega = 0 \quad \text{in } \Omega,$$

$$\nu\frac{\partial\omega}{\partial\mathbf{n}} = -\frac{\mathbf{u}\cdot\boldsymbol{\tau}}{\Delta t} \quad \text{on } \Gamma_b.$$

This equation has to be solved for a time-step Δt, with zero initial condition. The resulting field is then added to the vorticity obtained at the end of the previous advection–diffusion step. Fig. 5 is a sketch of this vorticity creation algorithm. The integral equation related to the above system, designed and implemented in the context of vortex methods in [16], has been since then the object of several works (see for instance [23]).

In 3D flows, boundary conditions are required for all three components of the vorticity. For plane boundaries, the 2D boundary conditions easily extend to give Neumann boundary conditions for the two tangential vorticity components. The flux of each tangential component must cancel the slip in the orthogonal direction [5]. For curved boundaries however, a closer look at the vorticity equation for the azimuthal component reveals that the vorticity flux for this component should correspond to a Robin type boundary condition. Indeed, from the Navier–Stokes equation, one gets in a viscous splitting algorithm the following diffusion equation for ω_θ:

$$\frac{\partial\omega_\theta}{\partial t} - \nu\left(\Delta\omega_\theta + \frac{2}{r^2}\frac{\partial\omega_r}{\partial\theta} - \frac{\omega_\theta}{r^2}\right) = 0 \quad \text{in } \Omega. \tag{14}$$

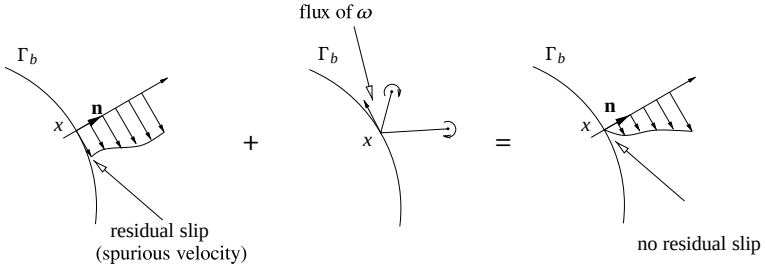

Fig. 5. Vorticity boundary conditions on the body Γ_b.

Upon observing that

$$\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}(r\omega_\theta)\right)=\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\omega_\theta}{\partial r}\right)-\frac{\omega_\theta}{r^2}$$

this above equation can be rewritten as

$$\frac{\partial\omega_\theta}{\partial t}-\nu\left[\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}(r\omega_\theta)\right)+\frac{1}{r^2}\frac{\partial^2\omega_\theta}{\partial\theta^2}+\frac{\partial^2\omega_\theta}{\partial z^2}+\frac{2}{r^2}\frac{\partial\omega_r}{\partial\theta}\right]=0.$$

It shows that the flux of azimuthal vorticity entering the flow through diffusion is given by (ν/r) $(\partial(r\omega_\theta)/\partial r)$. As a result, the natural boundary condition on ω_θ to cancel the slip in the spanwise direction reads

$$\frac{\nu}{r}\frac{\partial}{\partial r}(r\omega_\theta)=\frac{u_z}{\Delta t}\quad\text{on } \Gamma_{\mathrm{b}}. \tag{15}$$

As for the spanwise vorticity, since there is no curvature in the boundary on that direction, its boundary condition is a regular Neumann boundary condition, as in the case of a plane boundary. Finally the boundary condition for the normal vorticity component is clearly an homogeneous Dirichlet condition, since this component only involves tangential derivatives of the velocity at the wall. To summarize, the no-slip boundary condition is satisfied through the solution over the time-step Δt of the following diffusion equation:

$$\frac{\partial\boldsymbol{\omega}}{\partial t}-\nu\Delta\boldsymbol{\omega}=0\quad\text{in } \Omega, \tag{16}$$

$$\omega_r=0\quad\text{on } \Gamma_{\mathrm{b}}, \tag{17}$$

$$\nu\frac{\partial\omega_z}{\partial\mathbf{n}}=-\frac{u_\theta}{\Delta t}\quad\text{on } \Gamma_{\mathrm{b}}, \tag{18}$$

$$\nu\left(\frac{\omega_\theta}{r}+\frac{\partial\omega_\theta}{\partial\mathbf{n}}\right)=\frac{u_z}{\Delta t}\quad\text{on } \Gamma_{\mathrm{b}}, \tag{19}$$

where u_θ, u_z, respectively, denote the azimuthal and spanwise residual slip at the end of the previous convection step.

As a check of the consistency of these boundary conditions, it is worth noticing that they do not create any vorticity divergence. Along the same lines as in [5] and [27] we write, upon expanding the divergence and Laplace operators in the cylindrical basis $(\vec{e}_r,\vec{e}_\theta,\vec{e}_z)$:

$$\begin{aligned}\frac{\partial}{\partial r}(\nabla\cdot\boldsymbol{\omega})&=\frac{\partial^2\omega_r}{\partial r^2}+\frac{1}{r}\frac{\partial\omega_r}{\partial r}-\frac{\omega_r}{r^2}-\frac{1}{r^2}\frac{\partial\omega_\theta}{\partial\theta}+\frac{1}{r}\frac{\partial^2\omega_\theta}{\partial r\partial\theta}+\frac{\partial^2\omega_z}{\partial r\partial z}\\&=(\Delta\boldsymbol{\omega})\cdot\vec{e}_r+\frac{1}{r}\frac{\partial}{\partial\theta}\left(\frac{1}{r}\omega_\theta\right)+\frac{1}{r}\frac{\partial}{\partial\theta}\frac{\partial\omega_\theta}{\partial r}+\frac{\partial}{\partial z}\frac{\partial\omega_z}{\partial r}.\end{aligned}$$

In view of (14), (16) and (17), we obtain on the boundary Γ_{b},

$$\frac{\partial}{\partial r}(\nabla\cdot\boldsymbol{\omega})=\frac{1}{\nu}\frac{\partial\omega_r}{\partial t}+\frac{1}{r}\frac{\partial}{\partial\theta}\left(\frac{\omega_\theta}{r}+\frac{\partial\omega_\theta}{\partial r}\right)+\frac{\partial}{\partial z}\frac{\partial\omega_z}{\partial r}=\frac{1}{\nu}\frac{\partial\omega_r}{\partial t}+\frac{1}{\nu\Delta t}\frac{1}{r}\frac{\partial u_z}{\partial\theta}-\frac{1}{\nu\Delta t}\frac{\partial u_\theta}{\partial z}=\frac{1}{\nu}\frac{\partial\omega_r}{\partial t}+\frac{\widetilde{\omega}_r}{\nu\Delta t},$$

where we have denoted by $\widetilde{\boldsymbol{\omega}}$ the vorticity associated with the velocity field at the beginning of this diffusion step. We next observe that, by (17), the radial vorticity vanishes at the wall during the diffusion step. Since the normal velocity is zero at the wall, this remains true during the convection step, and the above right-hand side thus vanishes at the wall. The vorticity divergence finally satisfies

$$\begin{cases} \dfrac{\partial(\nabla\cdot\omega)}{\partial t} - \nu\Delta(\nabla\cdot\omega) = 0 & \text{in } \Omega, \\ \dfrac{\partial}{\partial r}(\nabla\cdot\omega) = 0 & \text{on } \Gamma_{\mathrm{b}}, \end{cases}$$

which proves our claim that, with the boundary conditions (17)–(19), no vorticity divergence is created during the diffusion step.

Once the correct boundary conditions have been identified, it remains to indicate how they translate in a vortex method. Following the idea developed in [15] we use a boundary integral formulation. The solution to (16)–(19) is written as

$$\boldsymbol{\omega}(x,t) = \int_0^t \int_{\Gamma_{\mathrm{b}}} G(\mathbf{x}-\boldsymbol{\xi}, 4\nu(t-s))\boldsymbol{\mu}(\boldsymbol{\xi},s)\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s, \tag{20}$$

where G, the kernel of the heat equation, is defined in 3D by

$$G(\zeta,\sigma) = \frac{\mathrm{e}^{-\zeta^2/\sigma^2}}{\pi^{3/2}\sigma^3}.$$

Applying Friedmann's theory [10] to Eqs. (16)–(19), one finds that components of $\boldsymbol{\mu}$ are solutions of the integral equations

$$-\frac{1}{2}\mu_r(\mathbf{x},t) + \nu\int_0^t\int_{\Gamma_{\mathrm{b}}} \frac{1}{r}G(\mathbf{x}-\boldsymbol{\xi},4\nu(t-s))(\boldsymbol{\mu}(\boldsymbol{\xi},s)\cdot\mathbf{n}_{\mathbf{x}})\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s = 0, \tag{21}$$

$$-\frac{1}{2}\mu_\theta(\mathbf{x},t) + \nu\int_0^t\int_{\Gamma_{\mathrm{b}}} \left[\frac{\partial G}{\partial \mathbf{n}_{\mathbf{x}}} + \frac{G}{r}\right](\mathbf{x}-\boldsymbol{\xi},4\nu(t-s))(\boldsymbol{\mu}(\boldsymbol{\xi},s)\cdot\boldsymbol{\tau}_{\mathbf{x}})\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s = -\frac{u_\theta(\mathbf{x},t)}{\Delta t}, \tag{22}$$

and

$$-\frac{1}{2}\mu_z(\mathbf{x},t) + \nu\int_0^t\int_{\Gamma_{\mathrm{b}}} \frac{\partial G}{\partial \mathbf{n}_{\mathbf{x}}}(\mathbf{x}-\boldsymbol{\xi},4\nu(t-s))(\mu_z(\boldsymbol{\xi},s))\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s = \frac{u_z(\mathbf{x},t)}{\Delta t}. \tag{23}$$

If the boundary was a flat plane, these three integral equations would be uncoupled. In cylindrical coordinates, there is a coupling between Eqs. (21) and (22). Note that in the case of a general 3D body Γ_{b}, there would be a coupling between all these three equations. Nevertheless, Eq. (21) can be rewritten

$$-\frac{1}{2}\mu_r(\mathbf{x},t) + \nu\kappa^3\int_0^t\int_{\Gamma_{\mathrm{b}}} G(\mathbf{x}-\boldsymbol{\xi},4\nu(t-s))\Big(\mu_r(\boldsymbol{\xi},s)\mathbf{x}\cdot\boldsymbol{\xi} + \mu_\theta(\boldsymbol{\xi},s)\det(\mathbf{x},\boldsymbol{\xi},\vec{e}_z)\Big)\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s = 0,$$

where $\kappa = 1/R$ is the curvature of the cylindrical physical boundary. Since, for symmetry reasons,

$$\int_0^t\int_{\Gamma_{\mathrm{b}}} G(\mathbf{x}-\boldsymbol{\xi},4\nu(t-s))\det(\mathbf{x},\boldsymbol{\xi},\vec{e}_z)\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s = 0,$$

to the leading order μ_r is solution of

$$-\frac{1}{2}\mu_r(\mathbf{x},t) + \nu\kappa^3 \int_0^t \int_{\Gamma_b} G(\mathbf{x}-\boldsymbol{\xi}, 4\nu(t-s))(\mu_r(\boldsymbol{\xi},s)\mathbf{x}\cdot\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}s = 0,$$

and thus $\mu_r \simeq 0$. Consequently, only the two independent Eqs. (22) and (23) have to be solved. By means of a Taylor development of the heat layer [15], one finally gets

$$\begin{bmatrix}\mu_\theta \\ \mu_z\end{bmatrix} \simeq \frac{-2}{\left(1+\kappa\sqrt{\nu\Delta t/\pi}\right)\Delta t} J_2 \begin{bmatrix}u_\theta \\ u_z\end{bmatrix}, \quad \text{where } J_2 = \begin{bmatrix}0 & 1 \\ -1 & 0\end{bmatrix}, \tag{24}$$

where we recall that u_θ, u_z are the spurious velocities obtained at the end of a convection step. The flux of vorticity defining the boundary layer is thus totally explicited. Numerical validations of these formulas can be found in [25].

2.4. Numerical results

In this section we present some numerical validation of the method just presented for wake calculations.

The wake of a cylinder remains a challenging case, in particular due to the computational effort devoted in grid-based methods to correctly approximate the outflow boundary conditions. Our goal here was in particular to investigate the effect of a rather short truncation of the computational domain on the accuracy in the computed drag coefficient. Figs. 10 and 6 summarize the dynamics of a typical wake, going from a 2D Kármán street (cf. Fig. 7) to a fully 3D flow (cf. Fig. 6), for a Reynolds number $Re = 300$. An indicator of the amount three-dimensionality of the flow is the enstrophy corresponding to the radial and azimuthal vorticity components, what we call transverse enstrophy, denoted as

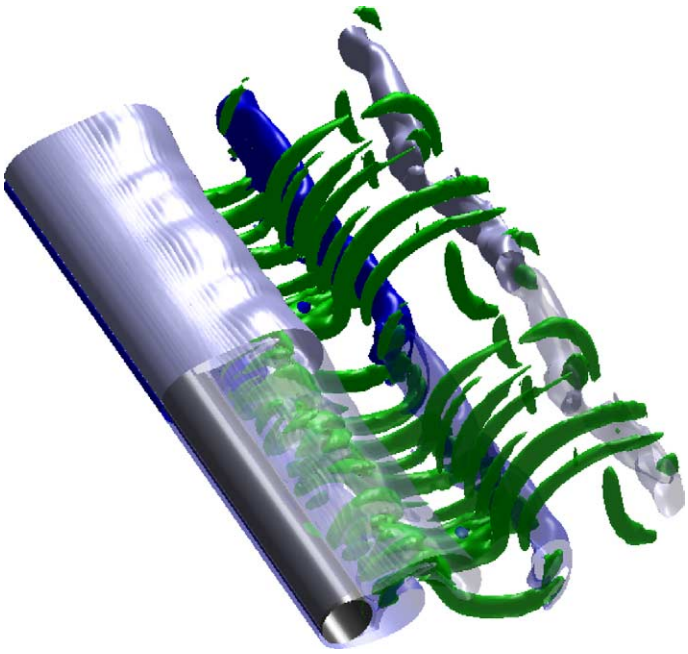

Fig. 6. Isovalues of transverse and spanwise vorticity at $Re = 300$, exhibiting a 3D saturated *mode* B instability.

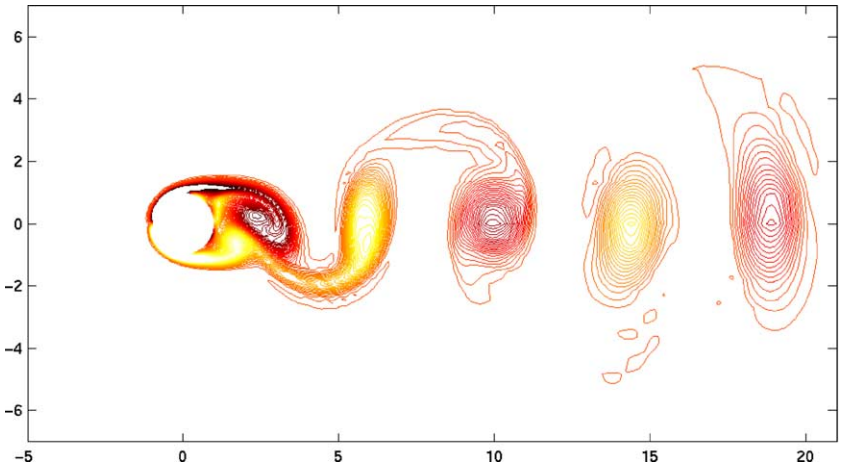

Fig. 7. 2D vorticity field at $Re = 300$.

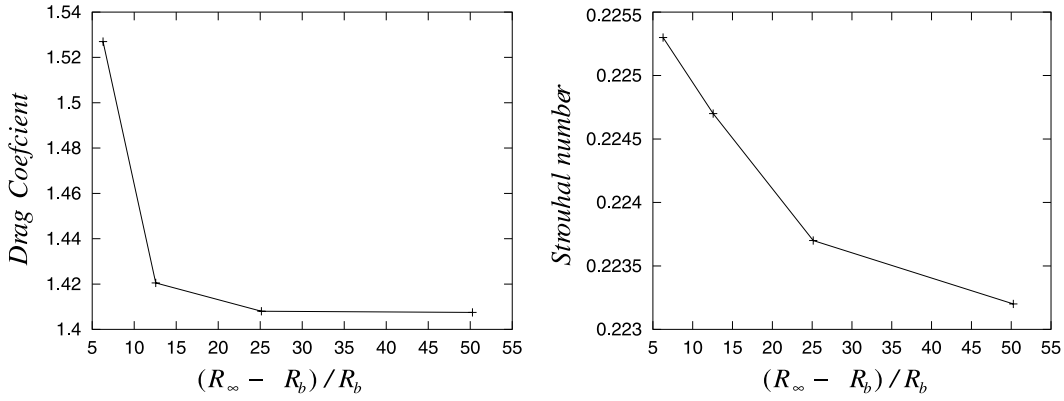

Fig. 8. Convergence of drag coefficient and Strouhal number for 2D simulations at $Re = 400$.

$$Z^{\perp} = \int_{\Omega} \omega_r^2(x) + \omega_\theta^2(x)\mathrm{d}x.$$

Fig. 12 shows that this transverse enstrophy remains in a first stage basically at the round-off level, then increases exponentially while streamwise vorticity develops along so-called "mode B" waves [29] (see Fig. 6), whose wavelength is close to the diameter of the cylinder. An interesting tool to track these waves is the spectral profile, defined as the norm of the spanwise Fourier transform of the velocity field for a given wave number. The spectral profile associated to the main-growth wavelength, in the present case $\lambda/D = 0.79$, is shown on Fig. 11. This result compares well with the spectral profile provided in [1], whose main-growth wavelength is predicted $\lambda/D = 0.82$.

The dynamics evolution from 2D to 3D is accompanied by a decrease in the drag value, as shown in Fig. 10. A thorough discussion of these results is given in [27].

Beyond this stage, streamwise structures of vorticity interact together and with the von Kármán alleys. The flow enters a saturated regime, represented by the saturation of the transverse enstrophy, plotted on Fig. 12.

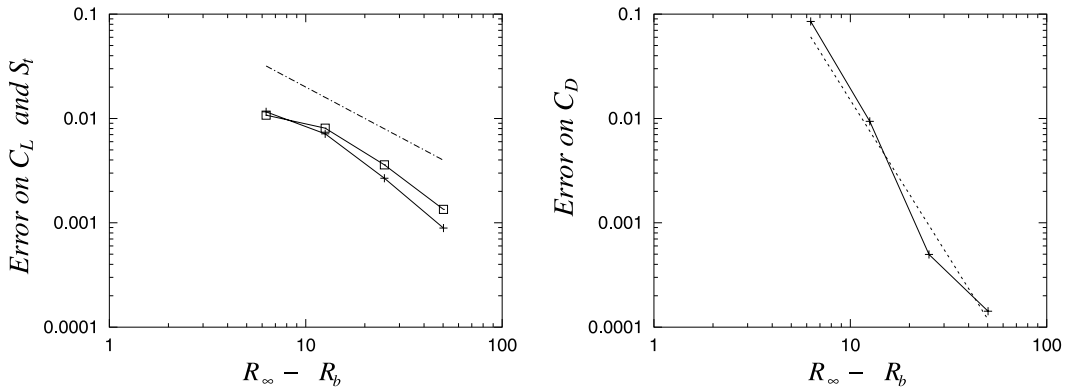

Fig. 9. Relative error in drag coefficient C_D (+), lift coefficient C_L (×) and Strouhal number S_t (□) for 2D simulations at $Re = 400$. Dotted lines correspond to first-order (left picture) and third-order (right picture) convergence.

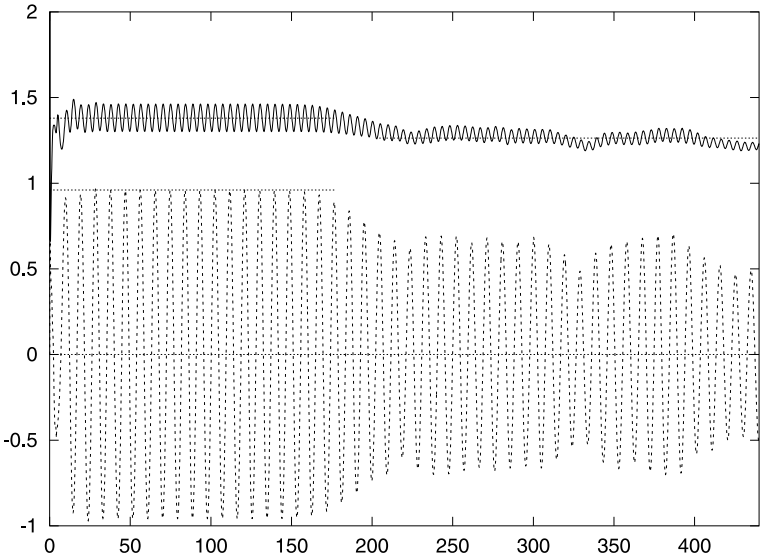

Fig. 10. Direct numerical simulation of an unstable 3D wake: typical drag (——) and lift (- - -) response to three-dimensionality for $Re = 300$. Dotted lines correspond to 0, 0.96, 1.262 and 1.38.

In these calculations the computational domain was

$$R \leqslant r \leqslant (1 + 4\pi)R \quad \text{and} \quad -\pi R \leqslant z \leqslant \pi R.$$

We used $256 \times 128 \times 128$ grid points. The ratio grid spacing versus particle spacing was always unity. When the wake was fully developed, particles occupied roughly 25% of the computational box. This size, which allows to follow four rolls (see Fig. 10), is in general thought as sufficient for accurate computation of body forces, especially the drag coefficient, provided the outflow boundary conditions do not create spurious vorticity. This is confirmed by our calculations.

Indeed, Table 1 and Figs. 8 and 9 shows the evolution of a few 2D diagnostics with respect to the domain size. These diagnostics are the mean drag coefficient $\overline{C_D}$, the mean top lift coefficient $\widehat{C_L}$ and Strouhal number S_t. The domain size is successively chosen at $(R_\infty - R_b)/R_b = 2\pi$, 4π, 8π and 16π.

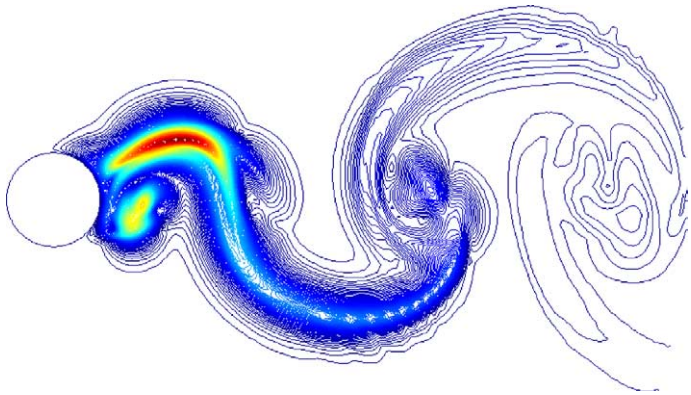

Fig. 11. Direct Numerical Simulation of an unstable 3D wake for $Re = 300$: typical spectral profile of main-growth wavelength (mode B).

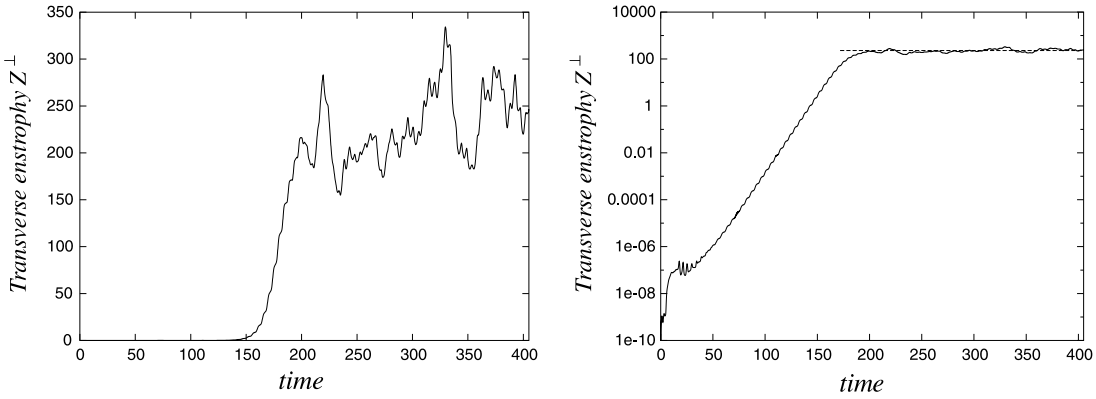

Fig. 12. Evolution of the 3D part of enstrophy, at $Re = 300$, using natural scale (left picture) and logarithmic scale (right picture).

One can notice on Fig. 9 that convergence of lift coefficient and Strouhal number is first-order, while drag coefficient is third-order. Computations can be considered as converged for $(R_\infty - R_b)/R_b = 8\pi$. For a distance of 4π, the error in the diagnostics is of the order of 1%. Given that we were mostly interested by the relationship between the dimension of the flow and the drag values, and that the difference between the 3D and 2D drag values at this Reynolds number was about ten times bigger, this level of accuracy was considered as satisfactory. A truncation radius corresponding to $(R_\infty - R_b)/R_b = 4\pi$ allows 3D calculations with good enough spanwise resolution to capture the desired wavelengths. Note that most finite-differences calculations need to extend the domain much further in the radial direction to avoid spurious wave reflection, sometimes at the expense of an insufficient spanwise resolution (for instance [20]). One explanation of the good behavior of the method even for relatively small domain in the radial direction is that the truncation of the domain only affects the field reconstruction, while the Lagrangian treatment of the vorticity advection equation does not rely on any artificial boundary condition.

Table 2 shows a comparison of drag values obtained in our computations (see also Fig. 10) and in other reference simulations [11,13,18]. More numerical results on this flow, and in particular new results concerning the effect of cylinder rotations on the topology of the wake, can be found in [7,26,27].

Table 2
Mean drag coefficients and Strouhal numbers for various Reynolds numbers, compared to reference diagnostics from [11], except ∗ from [18] and † from [13]

Re	Dim.	$\overline{C_D}$	S_t	Ref. $\overline{C_D}$	Ref. S_t
300	2D	1.382	0.2110	1.377	0.211
300	3D	1.262	0.2027	1.28∗	0.203∗
400	2D	1.408	0.2237	1.414	0.220
400	3D	1.198	0.210	1.2†	–

3. Immersed boundary VIC methods

The concept of immersed boundaries is an attempt to free numerical computations of flows around complex geometries from technically difficult and time-consuming grid generation algorithms. One may distinguish two broad classes of such methods. In the first class, reminiscent to volume of fluid (VOF) methods, computational cells close to the boundary are given a special treatment, depending on the way they intersect the boundary. For the inertial terms for instance, this approach typically leads in 2D finite-volume methods to modified flux formulas that seem rather involved to implement in 3D. In this class of methods, let us also mention the recent paper of Ploumhans and Winckelmans [23] and Ploumhans et al. [24], where vortex methods are designed to handle complex 2D and 3D geometries. In these papers, particles are given different treatments depending on there distance to or amount of overlapping with the body.

In the second class of methods, the flow equations are discretized in a unique way throughout the computational domain, which includes the immersed body, and boundary conditions appear in the form of localized forcing terms in the right-hand side of the flow equations. Our efforts belong to this class. We actually believe that since a non body-fitted method cannot take advantage, at least not in a straightforward way, of refinement potential that grid generation methods in general offer near boundaries, immersed boundary techniques must remain very simple and economical to compete with ever improving body-fitted techniques.

Immersed boundary methods can be traced back to Peskin's original idea of treating elastic fibers in biological flows by forces acting on the flow [22]. This idea, which was actually proposed together with a vortex method, although physically appealing, did not rely on a clear-cut treatment of boundary conditions. More recent efforts in the context of finite-difference methods aimed at giving a more conventional numerical definition of boundary conditions imposed on an immersed boundary. The general idea is to enforce boundary conditions through the addition of a singular source term on the boundary. This source term can be written most simply and efficiently directly at the discrete level as a forcing that at every time-step drives the flow back to rest on the boundary. A key point is then to interpolate this singular forcing on the grid points next to the boundary. Of particular interest is the reference [9] where the accuracy of the particular form of the interpolation function which distributes the forcing term on the grid is discussed. It seems that the optimal interpolation scheme has to be chosen carefully in function of the particular finite-difference method used to discretize the Navier–Stokes equations. For centered second-order finite-difference methods a linear interpolation allows to retain second-order accuracy up to the boundary.

In view of their robustness and reasonable cost when used with Cartesian grids, Vortex In Cell methods should clearly benefit from immersed boundaries approaches. The accuracy of vortex methods is largely dependent on accurate regridding techniques that in general require the use of global mappings to Cartesian geometries – as it is the case for cylinder wakes. Although it is possible to combine several local mappings in domain decomposition-like methods that can facilitate their implementation for complex geometries [6],

incorporating the concept of immersed boundaries in VIC methods would certainly add a great deal of flexibility in their application. The field computations through Poisson solvers is also clearly faster in Cartesian geometries than for more general cases, in particular due to the coupling generally appearing in the computations of all stream function components.

As it turns out, the treatment of immersed boundaries is very natural in the context of vortex methods [3]. Even in a body-fitted vortex method, vorticity flux formulas used to satisfy the no-slip condition can indeed be seen as a forcing term in the flow equation. As we will demonstrate below, the extension of this technique to immersed boundary is at the same time immediate and accurate. In the rest of this section, we successively describe how we handle no-through flow an no-slip boundary conditions, then we show some numerical validations of the method.

3.1. No-through flow boundary condition

When the Biot–Savart law is used to compute velocities in grid-free vortex methods, the no-through flow boundary condition is enforced by using single or double layer potentials. These potentials are evaluated on source points distributed along the boundary and through integral equations which translate the condition $\mathbf{u}\cdot\mathbf{n}=0$ on these points, along the lines of the classical panel method [12]. This procedure does not require particles in the flow to be initialized and remeshed on body-fitted grid. It thus gives a simple and elegant way to deal with immersed boundaries. Details of this method and numerical illustrations on impulsively started 2D cylinder are given elsewhere [3,7]. Here, for cost considerations already mentioned, we are interested in velocity evaluations based on grid Poisson solver. This leads to a slightly more involved method to account for the no-through flow boundary condition.

Let us assume that, at a given time-step, $\overline{\omega}$ is an extended vorticity field (that may simply be the extension by 0 of the flow vorticity) in a computational box $\overline{\Omega}$ containing the body (typically we will use a square box). Going back to the Helmholtz decomposition of the velocity, we have to solve, for the extended stream functions $\overline{\boldsymbol{\psi}}$ and potential $\overline{\phi}$, successively

$$-\Delta\overline{\boldsymbol{\psi}}=\overline{\boldsymbol{\omega}}\quad\text{in }\overline{\Omega}, \tag{25}$$

$$\nabla\cdot\overline{\boldsymbol{\psi}}=0\quad\text{in }\overline{\Omega}, \tag{26}$$

then

$$\Delta\overline{\phi}=0\quad\text{in }\Omega\text{ and in }\overline{\Omega}-\Omega, \tag{27}$$

$$\frac{\partial\overline{\phi}}{\partial\mathbf{n}}=-(\nabla\times\overline{\boldsymbol{\psi}})\cdot\mathbf{n}\quad\text{on }\Gamma_{\mathrm{b}}. \tag{28}$$

The above boundary condition on $\overline{\phi}$ has to be understood in the sense of outer normal derivative – assuming the flow domain is outside the obstacle.

Let us first point out that, if the domain $\overline{\Omega}$ is simple enough, the condition (25) is much simpler to complement with appropriate boundary conditions that enforce the divergence-free condition (26) than for a general domain. For a square box with sides parallel to the axis for instance, it suffices to use homogeneous Neumann boundary condition on the side perpendicular to the z-axis for the z-component of $\overline{\boldsymbol{\psi}}$ and homogeneous Dirichlet condition for the two other components, and similar conditions on the other sides of the box. This is definitely one advantage in using a Cartesian mesh rather than a body-fitted mesh.

The strategy to implement the boundary condition (28) when Γ_b does not coincide with grid points starts with the following observation: if $\overline{\phi}$ was a continuous harmonic extension of the exact flow potential across the boundary, in view of its gradient discontinuity we would get

$$\Delta\overline{\phi} = \left[\frac{\partial\overline{\phi}}{\partial\mathbf{n}}\right]_{\Gamma_b} \otimes \delta_{\Gamma_b},$$

where $[\cdot]$ means the jump across Γ_b and δ_{Γ_b} is the 2D Dirac mass supported by Γ_b. The goal is to determine $[\frac{\partial\overline{\phi}}{\partial\mathbf{n}}]_{\Gamma_b}$ and to distribute it on grid points. We proceed as follows: we first tag grid points which are at a distance less than the grid-size from the boundary. We denote by $\widetilde{\Gamma}$ the set made by these N grid-points. We then are looking for a function g, with support on $\widetilde{\Gamma}$ (see Fig. 13), such that the solution to the system

$$\Delta\overline{\phi} = g \quad \text{in } \overline{\Omega}, \tag{29}$$

$$\frac{\partial\overline{\phi}}{\partial\mathbf{n}} = 0 \quad \text{on } \overline{\Gamma}, \tag{30}$$

satisfies

$$\frac{\partial\overline{\phi}}{\partial\mathbf{n}} = -(\nabla\times\overline{\boldsymbol{\psi}})\cdot\mathbf{n} \quad \text{on } \widetilde{\Gamma}.$$

This constitutes a linear system for the unknown function g over $\widetilde{\Gamma}$ of size N. We use a GMRES type iterative solver to solve this system. The vector–matrix product involved in the iterative method consists of the solution of a Poisson equation followed by the evaluations of potential derivatives on the tagged grid-points. In the numerical examples given below, we use a standard Fishpack Poisson solver on a Cartesian uniform mesh, and second-order one-sided finite-differences for the gradient evaluation.

The method just outlined is very simple-minded and one expected drawback is that since the boundary condition is only fulfilled "near" the boundary, at most first-order can be achieved, with the risk of flow leaking outside Ω. However the incompressibility of the flow has a nice side effect here: the no-slip condition implies that the normal derivative of the normal velocity component vanishes on Γ_b. Therefore one has

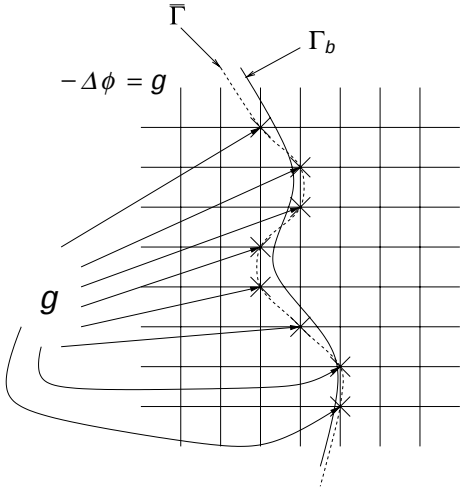

Fig. 13. Enforcement of no through flow condition for immersed boundary technique.

$$\mathbf{u}\cdot\mathbf{n}(\mathbf{x}) = \mathrm{O}\Big(d(\mathbf{x},\Gamma_{\mathrm{b}})^2\Big).$$

As a result, enforcing no-through flow at a distance less than a grid-size from the boundary yields second-order accuracy. In all our numerical experiment, no-through flow at the boundary was indeed satisfied well beyond the truncation errors introduced by the other approximations made in the vortex method. At every time-step, the GMRES method was initialized by the result of the previous time-step and two to three iterations were in general sufficient to reach the chosen residual error – fixed to 10^{-5} in our calculations.

3.2. Remeshing, diffusion and no-slip boundary condition

The reason for treating remeshing diffusion and no-slip boundary condition in the same discussion is that these three steps are tightly linked outside the advection step. In our VIC algorithms, we recall that remeshing is performed just before diffusion and vorticity flux formulas. In a body-fitted mesh, as already mentioned, one in general tries to use one-sided formulas. Diffusion for the tangential vorticity components is performed with homogeneous Neumann boundary conditions before vorticity fluxes formulas are applied. Since immersed boundary methods work on extended vorticity, these variants are no longer necessary, and plain remeshing or PSE formulas can be used in a straightforward way. This definitely distinguishes our method from the method proposed in [23] which uses corrected interpolation formulas to account for the overlapping of the boundary and the cells and is more in the spirit of VOF method. Note that the vorticity flux formulas that are essential in any vortex method are designed to correct, on the basis of the slip evaluated at the beginning of that step, any wrong vorticity flux that can have been injected in the flow by diffusion and remeshing. Using centered formulas to remesh vorticity and plain PSE formulas to diffuse vorticity near the boundary have only the effect of introducing spurious vorticity in the flow, something that the vorticity flux formulas are in any case designed to correct. In other words, not only the method described in Section 2 can be used without modification even if the boundary does not coincide with the underlying grid, but the concept of immersed boundary frees us from the need of having to use particular remeshing or diffusion formulas near the boundary.

Another point that must be made is that the vorticity flux is done from source points that are located on the boundary Γ_{b} itself, and not on grid points, and is estimated on the basis of the slip also evaluated on Γ_{b}. Therefore the no-slip boundary condition is enforced, up to the discretization errors, on the body boundary itself, no matter where flow particles are initialized and remeshed. In their ease to handle naturally immersed boundaries, vortex method definitely differ from grid-based methods.

3.3. Numerical examples

We focus here on the case of a ring impinging on a 3D cylinder (comparisons of drag values with reference results for impulsively started 2D cylinders are given elsewhere [3,7]). The initial condition consists of a ring of unit circulation with outer radius 1.4, and a Gaussian core of radius 0.5, located at a distance from the cylinder equal to 2.5 times the cylinder radius.

The computational box is a cube of size corresponding to three cylinder diameters. The Reynolds number is 400. Fig. 14 shows isosurfaces of vorticity magnitude at two successive times of the collision process. One can observe the production of secondary vorticity on the cylinder which rebounds and eventually creates two secondary rings.

We show in Fig. 15 the normal velocity on the cylinder for a rather coarse grid-size, corresponding to a grid resolution of 32^3 points, compared to the normal velocity that would be induced in free space by the ring on the cylinder. The normal velocity is at a level such that particles only exceptionally leak outside the flow domain.

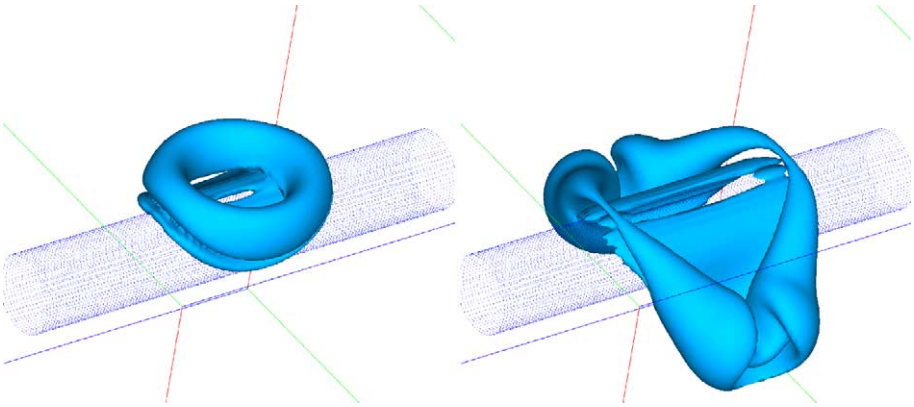

Fig. 14. Cylinder–ring interaction: isosurfaces of vorticity magnitude for times 10 and 40.

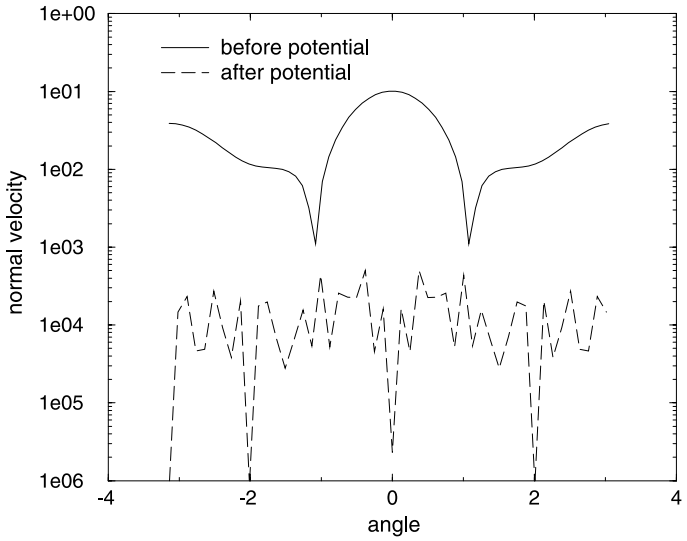

Fig. 15. Cylinder–ring interaction: normal velocity along the cylinder symmetry plane before and after potential correction.

We now turn to the treatment of the no-slip condition. In Fig. 16 we monitor the time evolution of the residual slip velocity together with the location, in the direction of the symmetry axis of the ring, of the center of velocity (for the purpose of this figure, these quantities are not scaled). One can observe that the slip is slightly increasing as the ring approaches the cylinder. It reaches its maximum value at about the time of collision, noticeable in the inflexion visible in the slope of the descent curve. Fig. 17 is a refinement study, at that time, of the accuracy in the treatment of the no-slip condition. In this figure are plotted the residual slip together with the numerical dissipation of the algorithm for several mesh-resolutions. The slip is evaluated in maximum norm, normalized by the slip induced by the initial ring, in absence of vorticity flux at the boundary. The effective diffusion of the algorithm computed by the formula

$$v_{\text{eff}}(t) = \frac{1}{2S(t)} \frac{\mathrm{d}}{\mathrm{d}t} E(t),$$

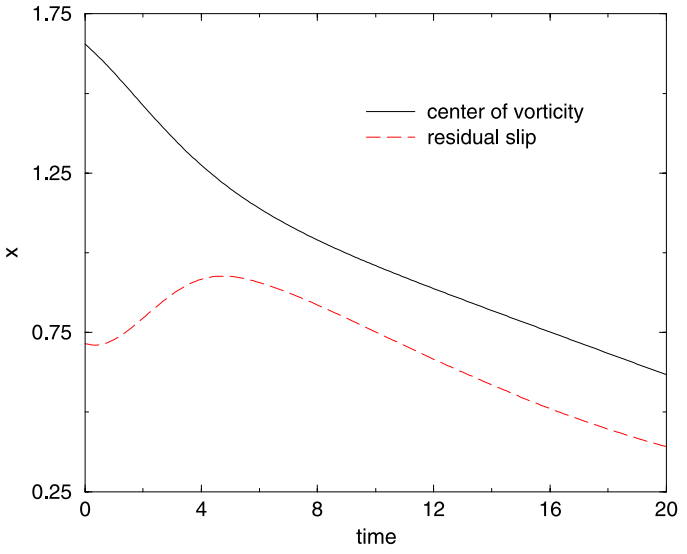

Fig. 16. Cylinder–ring interaction: time history of residual slip and location of center of vorticity along the ring axis.

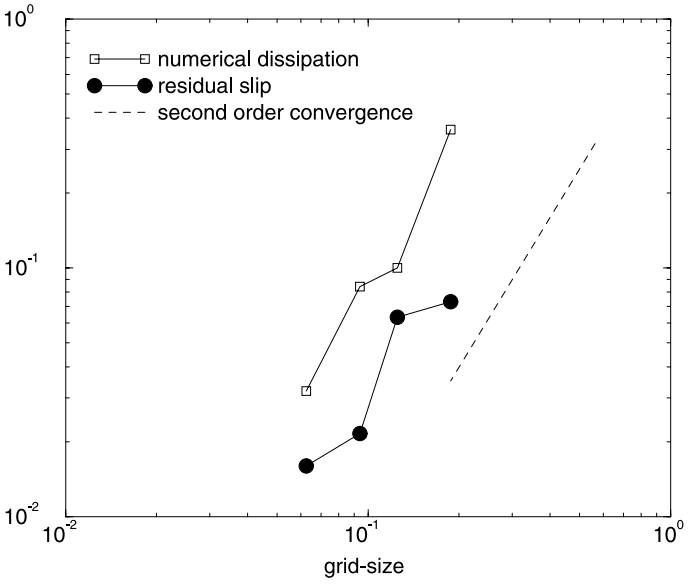

Fig. 17. Residual slip and numerical diffusion for several mesh refinements.

where $E(t)$ and $S(t)$, respectively, denote the instantaneous energy and enstrophy. The discrepancy $v_{\text{eff}} - v$, plotted in Fig. 13, essentially measures the cumulative dissipative effect of remeshing and vorticity creation. This figure shows that second-order accuracy is reached, which validates the argument given above in favor of a method where remeshing and diffusion are done by standard centered formulas up to the boundary.

Some indications of the cost of the method can further shed some light on the limits and possibilities of the method. For a 96^3 grid and about 500,000 particles, the CPU time was 50 s per time-step on a DEC

alpha workstation running at 500 MHz. This must be compared to the CPU taken by the cylindrical grid. The cost is about the same; in that case the advantage of using a faster Cartesian grid Poisson solver and simpler interpolation formulas in the immersed boundary code is compensated by the cost of the linear system to satisfy the no-through flow. For a cylinder wake, given the natural stretching of the cylindrical grid in the azimuthal direction, it seems unlikely that an immersed boundary vortex method can compete with the body-fitted method, except if a stretched Cartesian grid was used. Such a possibility, which would follow the lines of [6] was not yet implemented.

4. Conclusion

Two classes of VIC methods for the simulation of wall bounded flows have been described and validated. The first class uses body-fitted grids and particle distributions. An Helmholtz decomposition of the velocity fields allows to decouple the calculations of the stream functions which make possible the use of fast Poisson solvers. Numerical validations show that for wake calculations the far field boundary conditions required by these Poisson solvers does not introduce severe limitations over a totally grid-free particle method, as far as body forces are concerned. The resulting method retains the robustness and accuracy of grid-free particle methods while significantly reducing their numerical cost. In passing, we have also given a consistent treatment of 3D vorticity conditions which is not limited to flat boundaries.

The second class of methods deals with bodies as immersed boundaries. No through-flow and no-slip boundary conditions are enforced at two different stages of the algorithm: the no through-flow boundary condition is satisfied in the field calculation via the addition of an appropriate singular component to the potential part of the velocity. The no-slip condition is naturally handled by the vorticity flux formulas that are derived in body-fitted geometries. The method has been validated on the problem of a ring impinging on a cylinder. This problem has been selected as a prototype of 3D vortex–wall interaction which requires to capture accurately the vorticity created at the boundary. A refinement study suggests that the method is second order accurate. This method thus appears to retain the simplicity of particle methods in Cartesian geometries – in particular in the field evaluations and the interpolation formulas – while being general enough to apply to complex geometries.

Acknowledgements

The computational resources were provided by the joint CEA-UJF project CIMENT and the Department of Mathematics of INSA Toulouse.

References

[1] D. Barkley, R.D. Henderson, Three-dimensional Floquet stability analysis of the wake of a circular cylinder, J. Fluid Mech. 322 (1996) 215–241.
[2] A.J. Chorin, Numerical study of slightly viscous flow, J. Fluid Mech. 57 (1973) 785–796.
[3] G.-H. Cottet, Particles and immersed boundaries, Workshop on penalization methods and no-slip flows, Bordeaux, June 2000 (unpublished).
[4] G.-H. Cottet, B. Michaux, S. Ossia, G. Vanderlinden, A comparison of spectral and vortex methods in three-dimensional incompressible flows, J. Comput. Phys. 175 (2002) 702–712.
[5] G.-H. Cottet, P.D. Koumoutsakos, Vortex methods, theory and practice, Cambridge University Press, Cambridge, 2000.
[6] G.-H. Cottet, P.D. Koumoutsakos, M.L. Ould-Sahili, Vortex methods with spatially varying cores, J. Comput. Phys. 162 (2000) 164–185.

[7] G.-H. Cottet, P. Poncet, Particle methods for direct numerical simulations of three-dimensional wakes, J. Turbulence 3 (038) (2002) 1–9.
[8] G.-H. Cottet, P. Poncet, Simulation and control of three-dimensional wakes, to appear in Comput. Fluids (2002).
[9] 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.
[10] A. Friedmann, Partial differential equations of parabolic type, Prentice-Hall, Englewood Cliffs, NJ, 1964.
[11] R.D. Henderson, Details of the drag curve near the onset of vortex shedding, Phys. Fluids 7 (9) (1995) 2102–2104.
[12] J.L. Hess, Panel methods in computational fluid mechanics, J. Fluid Mech. 22 (1990) 255–274.
[13] S.K. Jordan, J.E. Fromm, Oscillatory drag, lift and torque on a circular cylinder in a uniform flow, Phys. Fluids 15 (1972) 371–376.
[14] P.D. Koumoutsakos, Inviscid axisymmetrisation of an elliptical vortex, J. Comput. Phys. 138 (1997) 821–857.
[15] P.D. Koumoutsakos, A. Leonard, High-resolution simulations of the flow around an impulsively started cylinder using vortex methods, J. Fluid Mech. 296 (1995) 1–38.
[16] P.D. Koumoutsakos, A. Leonard, F. Pepin, Boundary conditions for viscous vortex methods, J. Comput. Phys. 113 (1994) 52–61.
[17] K. Lindsay, R. Krasny, A particle method and adaptive treecode for vortex sheet motion in 3-D flow, J. Comput. Phys. 172 (2001) 879–907.
[18] A.G. Kravchenko, P. Moin, K. Shariff, B-spline method and zonal grids for simulations of complex turbulent flows, J. Comput. Phys. 151 (1999) 757–789.
[19] R. Mittal, S. Balachandar, Effects of three-dimensionality on the lift and drag of nominally two-dimensional flows, Phys. Fluids 7 (1995) 1841–1865.
[20] R. Mittal, Large eddy simulation of flow past a circular cylinder, Annual Research Briefs, Center for Turbulence Research, 1995.
[21] M.L. Ould-Salihi, G.-H. Cottet, M. El Hamraoui, Blending finite-differences and vortex methods for incompressible flow computations, SIAM J. Sci. Comput. 22 (2000) 1655–1674.
[22] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271.
[23] P. Ploumhans, G.S. Winckelmans, Vortex methods for high-resolution simulations of viscous flow past bluff bodies of general geometry, J. Comput. Phys. 165 (2000) 354–406.
[24] P. Ploumhans, G.S. Winckelmans, J.K. Salmon, A. Leonard, M.S. Warr en, Vortex methods for high-resolution simulation of three-dimensional bluff- body flows; application to the sphere at $Re = 300$, 500 and 1000, J. Comput. Phys. 178 (2002) 427–463.
[25] P. Poncet, Méthodes particulaires pour la simulation des sillages tridimensionnels, Ph.D. Thesis, University Joseph Fourier, Grenoble, France, 2001.
[26] P. Poncet, Vanishing of mode B in the wake behind a rotating circular cylinder, Phys. Fluids 14 (6) (2002) 2021–2023.
[27] P. Poncet, Topological aspects of the three-dimensional wake behind rotary oscillating circular cylinder, J. Fluid Mech. (2002), under revision.
[28] J.H. Walther, G. Morgenthal, An immersed interface method for the vortex-in-cell algorithm, J. Turbulence 3 (039) (2002) 1–10.
[29] C.H.K. Williamson, Three-dimensional wake behind a cylinder, J. Fluid Mech. 328 (1996) 345.