Immersed boundary technique for turbulent flow simulations

Gianluca Iaccarino

Center for Turbulence Research, Stanford University, CA 94305-3030; jops@ctr.stanford.edu

Roberto Verzicco

*DIMeG and CEMeC, Politecnico di Bari, Via Re David, 200, 70125, Bari, Italy;
verzicco@poliba.it*

The application of the Immersed Boundary (IB) method to simulate incompressible, turbulent flows around complex configurations is illustrated; the IB is based on the use of non-body conformal grids, and the effect of the presence of a body in the flow is accounted for by modifying the governing equations. Turbulence is modeled using standard Reynolds-Averaged Navier-Stokes models or the more sophisticated Large Eddy Simulation approach. The main features of the IB technique are described with emphasis on the treatment of boundary conditions at an immersed surface. Examples of flows around a cylinder, in a wavy channel, inside a stirred tank and a piston/cylinder assembly, and around a road vehicle are presented. Comparison with experimental data shows the accuracy of the present technique. This review article cites 70 references. [DOI: 10.1115/1.1563627]

1 CONTEXT

The continuous growth of computer power strongly encourages engineers to rely on computational fluid dynamics (CFD) for the design and testing of new technological solutions. Numerical simulations allow the analysis of complex phenomena without resorting to expensive prototypes and difficult experimental measurements.

The basic procedure to perform numerical simulation of fluid flows requires a discretization step in which the continuous governing equations and the domain of interest are transformed into a discrete set of algebraic relations valid in a finite number of locations (computational grid nodes) inside the domain. Afterwards, a numerical procedure is invoked to solve the obtained linear or nonlinear system to produce the local solution to the original equations. This process is simple and very accurate when the grid nodes are distributed uniformly (Cartesian mesh) in the domain, but becomes computationally intensive for disordered (unstructured) point distributions.

For simple computational domains (a box, for example) the generation of the computational grid is trivial; the simulation of a flow around a realistic configuration (a road vehicle in a wind tunnel, for example), on the other hand, is extremely complicated and time consuming since the shape of the domain must include the wetted surface of the geometry of interest. The first difficulty arises from the necessity to build a smooth surface mesh on the boundaries of the domain (body conforming grid). Usually industrially relevant geometries are defined in a CAD environment and must be translated and *cleaned* (small details are usually eliminated, overlapping surface patches are trimmed, etc) before a surface grid can be generated. This mesh serves as a starting point to generate the volume grid in the computational domain.

In addition, in many industrial applications, geometrical complexity is combined with moving boundaries and high Reynolds numbers. This requires regeneration or deformation of the grid during the simulation and turbulence modeling, leading to a considerable increase of the computational difficulties. As a result, engineering flow simulations have large computational overhead and low accuracy owing to a large number of operations per node and high storage requirements in combination with low order dissipative spatial discretization. Given the finite memory and speed of computers, these simulations are very expensive and time consuming with computational meshes that are generally limited to around one million nodes.

In view of these difficulties, it is clear that an alternative numerical procedure that can handle the geometric complexity, but at the same time retains the accuracy and high efficiency of the simulations performed on regular grids, would represent a significant advance in the application of CFD to industrial flows.

Transmitted by Associate Editor P Orlandi

In order to use regular Cartesian-like meshes, the requirement that the grid is conforming to the domain boundary must be relaxed. The basic idea is to consider the discretization of a simple, fictitious computational domain obtained by eliminating the complex object of interest (ie, the road vehicle in the example cited above). In order to obtain a realistic simulation, the effect of the presence of the object on the flow must be included in the problem. This can be achieved by introducing appropriate extra terms in the governing equations. Several approaches have been developed by using this concept and the main differences are related to the definition of the additional terms or, more in general, to the discretization schemes used in the vicinity of the *immersed* surfaces; in the next section an overview of different flavors of this approach is presented. The details of various forcing schemes are then described together with two approaches used for the simulations of high Reynolds number turbulent flows. In addition, a proposed scheme for grid enrichment in the neighborhood of an immersed surface is described.

Several examples of the application are reported to demonstrate the accuracy and flexibility of this technique.

2 BACKGROUND

The Immersed Boundary (IB) technique allows the solution of differential equations in complex geometric configurations on *simple* meshes by introducing *forcing* conditions on certain surfaces corresponding to the physical location of the complex boundaries; the simulations are then performed on a much simpler domain. This idea has been pursued by many researchers in the last three decades. To the authors knowledge, the first example is due to Vieceli [1] that extended the Marker and Cell (MAC) method [2,3] to include boundaries of arbitrary shape. The basic idea consisted of treating the fluid-boundary interface as a free surface and to impose there pressure boundary conditions so that particles could move only along the tangent to the boundary line. This procedure led to an iteration between pressure and velocity fields until flow incompressibility and boundary impermeability were both satisfied. The method, referred to as ABMAC (Arbitrary Boundary MAC), was generalized in a successive paper [4] to cope with moving walls and in this case, in addition to the pressure, velocity boundary conditions were also imposed at the interface. This technique allowed the treatment of walls moving with a prescribed law or moving as a consequence of the forces exerted by the fluid on the surface.

Peskin [5,6] reports, at the beginning of the 1970s, simulations of the blood flow in the heart/mitral-valve system assuming a very low Reynolds number and 2D flow. Three-dimensional heart flows that also included the contractile and elastic nature of the boundary were considered successively by Peskin [7] and McQueen and Peskin [8,9]. In Peskin's formulation, the fluid equations (incompressible Navier-Stokes equations) are solved on uniform Cartesian grids and the elastic fibers of the heart walls are *immersed* in the flow: fluid and fibers exert time varying forces on one another. A Lagrangian coordinate system moving with the local fluid velocity is attached to the fibers and tracks their location in space; the information about the position of the fibers and their forcing on the fluid is transferred to the Eulerian underlying mesh where the flow solution is obtained. In this procedure, the resulting forcing consists of delta functions located on the first cells external to the immersed body which, therefore, can not be adequately represented on a finite size mesh. For this reason, a smooth transition between the external fluid and internal body cells is introduced which is equivalent to spreading the delta function over a narrow band (typically three or four nodes) across the boundary.

The problem of heart modeling is complicated by the fact that the boundaries of the computational domain are moving and respond to forces (typically the pressure and viscous stresses) depending on the local flow conditions. In contrast, if the boundary configuration is fixed and known, the computation of the interaction between the fluid and the immersed surfaces is much simpler. In principle, Peskin's approach can be applied directly by decreasing the deformability of the elastic fiber; in practical terms, this will result in a numerically stiff problem.

The first applications of the IB approach to problems with solid, indeformable immersed surfaces were carried out by Basdevant and Sadourny [10], Briscolini and Santangelo [11], and Goldstein, Handler and Sirovich [12].

Briscolini and Santangelo [11] used an immersed boundary approach (referred to as mask method, which was a modified version of that by Basdevant and Sadourny [10]) to compute the unsteady 2D flow around circular and square cylinders at Reynolds numbers up to 1000 whereas Goldstein, Handler and Sirovich [12] considered the 2D start up flow around a circular cylinder and 3D plane- and ribbed-turbulent channel flow. In these works, the IB approach is used in conjunction with spectral methods and the forcing is applied in a band (consisting of three to four computational nodes) around the interface. This was required to reduce spurious oscillations appearing in the solutions. On the other hand, Saiki and Biringen [13] used the forcing of [12] to compute the flow around steady and rotating circular cylinders using fourth order central finite-difference approximations. The use of finite-differences avoided the appearance of spurious flow oscillations at the boundary even if in that case the forcing was also spread across the boundary using a procedure that the authors refer to as *first order accurate* similar to the delta function of Peskin [5].

The main drawback of the forcing introduced in [12] is that it contains two free constants that need to be tuned according to the problem being solved; in particular, for unsteady flows this forcing introduces a time step limitation that reduces the efficiency and the applicability of the method (see Section 3.2.1 for a detailed description). Another disadvantage of the described methods is that, in order to avoid equation stiffening and unphysical flow oscillations, the boundary forcing terms are spread across the boundary which therefore is smeared over the grid, thus decreasing the solution accuracy.

In the framework of general PDEs, the mathematical formulation of the problem has been described by LeVeque and Calhoun [14]; a simple 1D unsteady diffusion equation with a moving interface was considered:

$$\phi_t = \phi_{xx} + c(t)\,\delta(x - \alpha(t)) \qquad (1)$$

where $\alpha(t)$ represents the interface location (immersed boundary) and $c(t)$ the solution value in $\alpha(t)$.

The discretization of Eq. (1) requires the use of an approximation to δ: δ_h. The authors derived constraints to impose on δ_h to ensure high order accuracy at the immersed boundary. No such guidelines are available in general, ie, for the 3D Navier-Stokes equations. Nevertheless, *a posteriori* analysis has shown that it is possible to achieve second order accuracy in practical applications [15].

Mohd-Yusof [16] derived an alternative formulation of the forcing that does not affect the stability of the discrete time equations and does not require forcing smoothing. In addition, no user-defined parameters were used in the formulation of the forcing function, making it flow independent (see Section 3.2.3). In [16] the new forcing was combined with B-splines to compute the laminar flow over a 3D ribbed channel, showing substantial improvements with respect to the previous formulations. This discrete time forcing scheme was originally developed in a spectral context and has also been successfully applied to flows around cylinders and spheres, at moderate Re.

Recently, the same idea of forcing has been used by [17] and [18] in the framework of a finite-difference Large-Eddy Simulation code. The applications included the flow around simple and complicated geometries in a large range of Reynolds numbers. Some of these results are reported in the following.

Before concluding this section, we wish to stress that the above described techniques can all be gathered under the name of Immersed Boundary methods which are the object of the present review. There are other procedures, however, that rely on a similar philosophy and differ in technical details. For the sake of brevity, we will not review these approaches, but we will give in the following a short description in order to provide the interested reader with additional references.

A class of methods called *penalty methods* or *fictitious domain methods* or *domain embedding methods* assumes the immersed body as a porous medium and solves the Navier-Stokes-Brinkman equations; these are the Navier-Stokes equations with the addition of a term of volume drag, called Darcy drag, which accounts for the action of the porous medium on the flow. When the medium permeability K tends to zero, the medium behaves as a solid body and the flow around an arbitrary shaped object can be studied. A detailed description of the method can be found in Khadra *et al* [19], Khadra *et al* [20], Angot *et al* [21], Kevlahan and Ghidaglia [22], and references therein; in this context we only wish to stress that when viewed from the perspective of the immersed boundary methods, the penalty methods can be considered as a particular case of the forcing by [12] and [13], even if with a completely different physical meaning. This will be briefly shown in Section 3.2.2.

A different technique, called *Cartesian Grid Method*, also relies on non-body-fitted meshes to describe the flow around complex geometries. In this case, however, instead of adding forcing terms to the governing equations, the grid cells are modified at the body interface according to its intersections with the underlying grid. Given the large number of possible intersections, this generates a wide variety of *cut interface cells* which must be handled in a separate way depending on their topology; nevertheless, finite volume methods have been successfully used for the calculation of complex 3D flows [23]. In the framework of finite volume discretization, the transfer of information from the embedded surface to the underlying Cartesian grid needs special treatments to satisfy conservation laws on the physical domain. As already mentioned, this results in irregularly shaped cells (Cartesian mesh cells cut by the interface) at the boundaries in which a flux balance must be performed. Various flavors of *cell cut* algorithms have been developed by Aftosmis [24] and Forrer [25]. The pacing item in applying finite volume schemes on Cartesian meshes has been the stiffness associated with small cells cut at the boundaries; recently Ye *et al* [26] proposed cell merging schemes that alleviate this problem by agglomerating the smallest cell to their closest neighbors. Initial accuracy assessments have been carried out by Coirier [27] for the compressible Navier-Stokes equations and by Almgren *et al* [28] for the Poisson equations; full second order accuracy was only obtained considering the cells not intersected by the boundaries. In the more recent analysis by Ye *et al* [26], second order accuracy has also been reported at the boundary.

Finally, an entire class of numerical techniques has been developed to handle moving interfaces (of various physical nature) on a fixed grid. Two of these techniques are the Volume of Fluid (VOF) approach [29] and the Level-Set technique [30]. An example of applications for the first technique is the tracking of interfaces between different fluids (ie, free surface flows) and for the second is the propagation of flames in combustion modeling [31]. These techniques might be adapted to deal with solid interfaces (representing and object in a fluid stream) and therefore used in a context similar to the IB method.

3 NUMERICAL TECHNIQUE FOR THE NAVIER-STOKES EQUATIONS

In this section, a numerical technique for the solution of Navier-Stokes equations based on the IB approach is described. The simulation of the flow around a solid object requires the generation of a computational grid covering the domain of interest without the object and a geometrical description of the object (in general, a surface S). The governing equations are solved on the *underlying* mesh and a *forcing* term $\mathbf{f}$ is introduced to model the effect of the presence of the body on the flow evolution. This term is evaluated such that a desired velocity distribution $\mathbf{V}_S$ can be assigned over the boundary S. We wish to stress that the presence of $\mathbf{f}$ in Eq. (2) does not imply, necessarily, a force acting on the fluid; in fact, while the expression of $\mathbf{f}$ by [5,12], and [13] indeed represents the action of the surface on the flow, the formulation by [16] is equivalent to a velocity boundary conditions inside the domain. The Navier-Stokes equations for an incompressible fluid (with density ρ and viscosity ν) in 3D are:

$$\frac{D\mathbf{u}}{Dt}=\frac{\partial\mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}=\nu\nabla^2\mathbf{u}-\rho^{-1}\nabla p+\mathbf{f},\quad \nabla\cdot\mathbf{u}=0 \tag{2}$$

where $\mathbf{u}$ and p are the velocity components and the pressure, respectively. In principle, there are no restrictions for the velocity distribution $\mathbf{V}_S$ and for the shape and motion of S; therefore, a variety of boundary conditions can be imposed. The main advantage of this approach is that $\mathbf{f}$ can be prescribed on a regular mesh so that the accuracy and efficiency of the solution procedure on simple grids are maintained. The geometrical description of the object is based on a CAD representation so that the CFD solver can be directly linked to the CAD environment without the need for surface translations and modeling.

The main components of the present solver are:

1) accurate and reliable discretization schemes and turbulence models
2) a computational-geometry algorithm to locate the object onto the grid and to transfer information between the mesh and the object surface (forcing and reconstruction)
3) a mesh-enrichment approach to increase the grid resolution in the vicinity of the immersed surface

These components will be described in the following sections.

3.1 Discretization schemes and turbulence modeling

Most of the nonbody conforming techniques described in the literature are based on the direct solution of Eqs. (2) and, therefore, simulations performed at relatively low Reynolds number. In these conditions the treatment of the immersed boundary condition is somewhat less critical because the viscous effects are dominant and stabilizing for the numerical procedure. On the other hand, high Reynolds numbers are challenging because the boundary layers tend to be extremely thin and inaccuracies introduced at the (immersed) boundary can completely modify the development of the flow. In this review we examine the IB technique developed in the framework of Large Eddy Simulation (LES) and Reynolds Averaged Navier-Stokes (RANS) approaches.

3.1.1 Large eddy simulation technique

The relevant equations for LES are obtained from the Navier-Stokes Eqs. (2) after the application of a space filter (filtered variables are denoted with a bar):

$$\frac{D\overline{\mathbf{u}}}{Dt}=\nu\nabla^2\overline{\mathbf{u}}+\nabla\cdot\{\nu+\nu_t[\nabla\overline{\mathbf{u}}+(\nabla\overline{\mathbf{u}})^T]\}-\rho^{-1}\nabla\overline{P}+\overline{\mathbf{f}},\quad \nabla\cdot\overline{\mathbf{u}}=0 \tag{3}$$

The difference between (2) and (3) consists of the second term on the right hand side of (3) that is obtained when the filter is applied to the nonlinear term $\mathbf{u}\cdot\nabla\mathbf{u}$ and the Boussinesq hypothesis of stress-strain linear relationship is invoked.

The eddy viscosity ν_t is modeled as $\nu_t=C\Delta^2(2\overline{S}_{ij}\overline{S}_{ij})^{1/2}$ where $\overline{S}_{ij}$ is the filtered rate of strain tensor, and Δ is the filter width (typically the grid size).

The value of the model coefficient C in the subgrid scale turbulent viscosity ν_t is determined by a dynamic procedure [32,33] that does not require direct specification of any model constant. Essentially, this involves filtering at two different length scales (or filter widths). The first filtering operation is implicit in the numerical method, and corresponds to a filter width that is equal to the local grid spacing.

The second filter (the *test filter*) is implemented by averaging over nearest neighbor nodes; this corresponds to a filter width of twice the local grid spacing. Finally, an average over statistically homogeneous directions is needed to determine the local value of C. Provided that grid resolution is adequate in the vicinity of solid walls, this form of the dynamic model properly accounts for wall proximity without explicit damping functions (eg, the van Driest function in the case of a Smagorinsky model [34]).

The prescription of the term $\mathbf{f}$ is described in the next section.

Details of the numerical method are given in Verzicco and Orlandi [35] and Orlandi (2000) [36], the latter also providing the source code and several advanced tutorials; only the main features are summarized here. Spatial derivatives in Eqs. (3) are discretized in a Cartesian or polar cylindrical coordinate system (depending on the application) using staggered central second order finite difference approximations. The discretized system is integrated in time using a fractional step method where the viscous terms are computed implicitly and the convective terms explicitly. The large, sparse matrix resulting from the implicit terms is inverted by an approximate factorization technique. At each time step the momentum equations are provisionally advanced (predictor step) using the pressure at the previous time step, giving an intermediate non-solenoidal velocity field. Afterward, an elliptic equation (obtained combining the continuity and the momentum equations) is solved to enforce the divergence-free condition on the velocity field.

3.1.2 Reynolds averaged Navier-Stokes technique

The RANS equations closely resemble the Eqs. (3) with the difference that the dependent variables are now time averaged (and not space averaged). Within this context, therefore, $\overline{\mathbf{u}}$ and P denote the mean velocity and pressure, respectively. From a physical point of view, the main difference is that turbulence is modeled completely (at all length scales) and for this reason turbulence models are more complex and crucial for accurate simulations.

In addition to Eqs. (3), transport equations are solved to model selected turbulent quantities (for example, the turbulent kinetic energy and the turbulence dissipation rate in the standard k-ϵ models [37]) with the objective being to build the eddy viscosity ν_t. This is a substantial difference between LES and RANS when used in conjunction with the IB approach for an immersed solid object; turbulent quantities exhibit steep gradients (and usually local maxima) in the near vicinity of solid boundaries. This imposes strict requirements to the accuracy of the numerical scheme and to the grid resolution in the region near the immersed boundary.

One of the simplest (yet accurate) differential turbulence model is the one equation model due to Spalart and Allmaras [38]; the additional transport equation is:

$$\frac{D\tilde{\nu}}{Dt}=G_\nu+\frac{1}{\sigma_\nu}\nabla\cdot\{(\nu+\nu_t)\nabla\tilde{\nu}\}+C_{b2}(\nabla\tilde{\nu})^2-Y_\nu+f_\nu \quad (4)$$

where G_ν is the production of turbulent viscosity and Y_ν is the destruction of turbulent viscosity that occurs in the near wall region due to wall blocking and viscous damping; σ_ν and C_{b2} are model constants and f_ν is the forcing term. The eddy viscosity is computed as:

$$\nu_t=\tilde{\nu}\frac{\chi^3}{\chi^2+C_{b1}^3} \quad (5)$$

where χ is the ratio between $\tilde{\nu}$ and the molecular diffusion and C_{b1}, another model constant. The production term is proportional to the vorticity magnitude whereas the destruction term contains the dependency on $\tilde{\nu}$ and the distance from the closest wall to provide eddy viscosity damping in the near-wall viscous dominated region. This model has been developed to capture appropriately boundary layers flows subjected to adverse pressure gradients; it is the simplest model available to accurately capture separation [38]. It is useful to point out that the modified eddy viscosity $\tilde{\nu}$ varies linearly in the boundary layer approaching a solid wall (whereas the variation of the eddy viscosity is nonlinear according to Eq. (5)). This property will be useful in conjunction with the treatment of the immersed boundaries. Differently from the SGS model in the LES approach, this model requires for each computational grid point the distance from the wall to evaluate the damping functions.

The equations are discretized on structured grids using a collocated finite difference high order upwind scheme. The code solves the equations in a segregated manner, with the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) algorithm used to achieve the pressure-velocity coupling for stability. In the SIMPLE algorithm, the continuity equation is converted into a discrete Poisson equation for pressure. The differential equations are linearized and solved implicitly in sequence: starting with the pressure equation (predictor stage), followed by the momentum equations and the pressure correction equation (corrector stage). The equations for the scalars (turbulent quantities) are solved after the updating of both pressure and velocity components. Within this loop, the linearized equations for each variable, as they arise, are treated using an algebraic multigrid solver.

Other turbulence models can be formulated starting from transport equations of different quantities; for example, the equations for the turbulent kinetic energy k and the turbulence dissipation rate ϵ lead to the well-known k-ϵ approach. In general, any turbulence model can be used in the framework of a RANS/IB solver [39] but the particular behavior of the unknown quantities in the close vicinity of solid walls will influence the quality and accuracy of the predictions in relation to the immersed boundary treatment (see Section 3.2.4).

3.2 Definition of the forcing term

The accuracy of the IB approach depends on the specification of the forcing term in the governing equations. In this section, different approaches are analyzed. In addition, the problem of defining the forcing in the neighboring of the immersed surface and eventually inside the object are discussed.

3.2.1 *Feedback forcing*

Supposing that the immersed surface S coincides locally with the grid node (i,j,k) and a Dirichlet boundary condition, possibly time dependent, $\phi(t)=\phi_S(t)$ has to be applied (ϕ is either one of the velocity components or the eddy viscosity in the RANS model). The forcing term *f*, according to Goldstein, Handler and Sirovich [12] and Saiki and Biringen [13], is:

$$f_{ijk}(t)=\alpha_f\int_0^t[\phi_{ijk}(t')-\phi_S(t')]dt'+\beta_f[\phi_{ijk}(t)-\phi_S(t)] \quad (6)$$

α_f and β_f are *negative* constants (whose dimensions are, respectively, $1/T^2$ and $1/T$, T being the time). The above quantity is a feedback to the difference $\phi_{ijk}-\phi_S$ that asymptotically enforces $\phi_{ijk}=\phi_S$ on the immersed boundary. In fact, the first term of Eq. (6) will decrease in time (become more negative) as the integrand increases, thus tending to annihilate any difference between ϕ_{ijk} and ϕ_S. The second term, on the other hand, can be interpreted as the resistance opposed by the surface element to assume a boundary value different from ϕ_S.

An intuitive argument for understanding the action of the above forcing is the following. Consider the forcing applied to the velocity components in Eqs. (2) to impose the condition $\mathbf{u}(t)=\mathbf{V}_S$ on the surface; if we retain only the first term on the left hand side and the last term on the right hand side in the Navier-Stokes Eq. (2):

$$\frac{d\mathbf{q}}{dt}\approx\mathbf{f}=\alpha_f\int_0^t\mathbf{q}dt'+\beta_f\mathbf{q} \quad (7)$$

with $\mathbf{q}=\mathbf{u}-\mathbf{V}_S$. Equation (7) represents a simple damped oscillator. This implies that as $\mathbf{u}$ on the boundary becomes different from $\mathbf{V}_S$ the forcing $\mathbf{f}$ *brings* $\mathbf{u}$ back to $\mathbf{V}_S$. In an unsteady flow, the magnitude of α_f must be large enough so that the restoring force can react with a frequency which is bigger than any frequency in the flow. Unfortunately, the value of the constants is flow dependent and, even if, when α_f and β_f are big enough, the flow becomes independent of their value, there is not a general rule for their determination. The major drawback of this forcing, however, is that big values of α_f and β_f make Eqs. (2) stiff and its time integration requires very small time steps. Goldstein, Handler and Sirovich [12] performed the stability analysis and they found that, when all the forcing terms are computed explicitly, a one or two orders of magnitude decrease in the time step size was required to ensure stability. This is clearly unacceptable for large scale calculations of turbulent flows.

A partial improvement to the stability limit can be obtained by treating the second term in the forcing term of Eq.

(7) implicitly [17]; this modification only alleviates the severe time step limitation of a fully explicit treatment of the forcing.

It should be stressed that the stability of the calculation (and, therefore, the time step size) depends not only on the values of α_f and β_f but also on the flow, ie, on the details of the geometry to be mimicked. We have observed, for example, that the presence of sharp corners prevents the adoption of small values (in magnitude) of α_f and β_f. On the other hand, for smooth geometries, small values of the constants can be used and simulations up to $CFL=0.5$ can be run. It is also possible to relax the values of α_f and β_f during the *quiet* phases of the flow evolution, but there is no unique criterion for this and *ad hoc* judgments are needed.

3.2.2 Penalty methods

In the family of the penalty methods, the forcing term f_{ijk} assumes the form

$$f_{ijk}=\frac{(\phi_{ijk}-\phi_S)}{\beta} \tag{8}$$

where the parameter β is a function both of spatial position $\mathbf{x}$ and time t. Equation (8) is a particular case of the feedback forcing but it can be interpreted in a different way. If we consider the Navier-Stokes Eqs. (2) and we introduce the Darcy number ($Da=K_0/L^2$ with K_0 a reference permeability and L a reference length), the above forcing can be rewritten for the velocity as

$$\mathbf{f}_{ijk}=\frac{\nu(\mathbf{u}_{ijk}-\mathbf{V}_S)}{\rho DaK} \tag{9}$$

K being the new free parameter. If $K\rightarrow\infty$, the forcing vanishes and Eqs. (2) recover the standard Navier-Stokes. In contrast, if $K\rightarrow 0$, the forcing becomes dominant in the equation yielding the solution $\mathbf{u}_{ijk}=\mathbf{V}_S$.

For $0<K<\infty$, the forcing can be modulated to provide a momentum loss in a desired region, thus simulating porous media. In this case, Eqs. (2) become the Navier-Stokes/Brinkman equations that can be solved over the whole domain with different values of K depending on the zonal characterization (fluid, solid, or porous medium).

A drawback of the method is that in computer simulations the value of K can be neither 0 nor ∞, therefore solid and fluid regions are approximated by finite, user-defined values; they tend to be dependent on the problem and Reynolds number and must be tuned for each simulation. The main problem is related to the use of very small values of K that increase the stiffness of the governing equations and, therefore, the convergence properties of the solution procedure. The final values of K must therefore be a compromise between the need of approximating solid boundaries and the preservation of the numerical stability at a reasonable computational cost.

3.2.3 Direct forcing

This approach consists of an imposition of the velocity boundary conditions on the immersed surface without introducing or computing any forcing term. Nevertheless, in order to make the notation consistent with the previous section we will treat the direct forcing in the context of an extra force in the Navier-Stokes equations.

Mohd-Yusof [16] first proposed to consider the problem of forcing directly in the context of the discretized equation to drive the numerical solution towards the required boundary values. In his approach, the forcing can be explicitly defined so that appropriate boundary values are specified at the immersed surface. In other words, if ϕ^n_{ijk} is the approximation of the solution of the governing equation, the discrete Navier-Stokes equation can be written as:

$$\frac{\phi^{n+1}_{ijk}-\phi^n_{ijk}}{\Delta t}=RHS+f_{ijk} \tag{10}$$

where Δt is the time step and RHS represents the discretized form of the convective, diffusive, and source terms in the LES or RANS equations. Supposing, as before, that the immersed surface coincides with the node (i,j,k) and a Dirichlet boundary condition ($\phi=\phi_S$) has to be applied, the forcing term f can be directly obtained by:

$$f_{ijk}=\frac{\phi_s-\phi^n_{ijk}}{\Delta t}-RHS \tag{11}$$

being ϕ either $\mathbf{u}$ or $\tilde{\nu}$.

Once more, it is worth noting that the combination of Eqs. (10) and (11) yields $\phi^{n+1}_{ijk}=\phi_S$ which corresponds to a boundary condition within the flow; this implies that (in practice) in a calculation, the forcing term (11) is never computed. The main advantage in this case is that no additional terms are introduced in the equations, thus avoiding stiffness problems as in the previous approaches.

3.2.4 Boundary reconstruction

The expressions previously given for the forcing are formally derived in the case that the position of the unknowns on the grid coincides with the immersed boundary; this would require the boundary to lay on coordinate lines or surfaces which is not the case for complex curvilinear geometries. In particular, in the case of a staggered solution algorithm, even if the boundary were coincident with the position of one unknown, this would not be so for the others; therefore, an interpolation procedure would be needed anyway.

As for the choice of the forcing scheme, many different techniques have been adopted to overcome this difficulty. We can classify the available techniques in two groups: a) schemes that spread the forcing function in the vicinity of the immersed surface and b) schemes that produce a local reconstruction of the solution based on the target boundary values.

Peskin [6] proposed the first approach by substituting a *discrete* Dirac δ function in (1) as discussed before. The main drawback of this approach is that this spreading acts as an extra dissipation in the close vicinity of the immersed boundary; this can lead to inaccurate predictions of the boundary layer development.

On the other hand, local reconstructions of the solution in the vicinity of the immersed boundary can be built with high degree of accuracy. Initial work by [18] was based on a simple linear, 1D operator (Fig. 1*a*) and this approach proved to be accurate for boundaries largely aligned with grid lines. On each grid segment intersecting the immersed surface, a linear velocity reconstruction is obtained using the *interior* value (point 2 in Fig. 1*a* and the wall value point 0); the value close to the interface inside the body (point 1) can therefore be reconstructed. In the general case of curved boundaries on Cartesian grids a more sophisticated reconstruction scheme must be used. In Figs. 1*b* and 1*c* linear and quadratic 2D stencils are shown. In the linear case, two *interior* values (2 and 3) are used together with the wall value (0) to evaluate the solution close to the interface (1). Note that the triangles in Fig. 1*b* are built such that two vertices always lay in the fluid part. By increasing the support for the interpolation stencil higher order reconstructions can be obtained. We point out that in this case the word *interior* always refers to the fluid side, therefore in the node 1 the velocity is reversed in order to prevent the flow from penetrating and slipping on the immersed boundary. The above linear reconstruction has been implemented in [17] in a slightly different way, with the node 1 being the first interior and 2 the second interior; only minor differences were found with respect to the above described procedure, even if the latter is preferable since reconstructed quantities are not assigned inside the flow domain.

It is worthwhile to notice that this approach can be considered as a generalization of the *ghost cell* approach [39], where the boundary conditions are imposed by fixing suitable values of the solution outside the computational domain (ghost cells).

The stencils reported earlier are suitable to reconstruct variables that are smoothly varying without exhibiting large maxima; it is well known that high order polynomial interpolations are keen to introduce wiggles and spurious extrema. For this reason, more elaborate schemes may be used; the inverse distance weighted method proposed by [40] has the property of preserving local maxima and producing smooth reconstruction. The interpolation at a certain location (x,y,z) is:

$$\phi(x,y,z)=\sum_{m=1}^{n} w_m\phi_m/q \tag{12}$$

$$w_m=\left(\frac{R-h_m}{Rh_m}\right)^p \tag{13}$$

$$q=\sum_{l=1}^{n}\left(\frac{R-h_l}{Rh_l}\right)^p \tag{14}$$

where ϕ_m represents the solution at a certain location, w_m represents the weight, and h_m the distance between the location (x,y,z) and the location of ϕ_m; R represents the maximum h_m.

The method of inverse distance has been successfully used also for the velocity reconstruction at the immersed boundary by Tessicini *et al* [41] for the axisymmetric and 3D simple and coaxial jets forming from a curvilinear nozzle.

3.2.5 Internal treatment of the body

A few words should be said about the internal treatment of the bodies since the forcings described in Sections 3.2.1–3.2.3 are only valid at the boundary. For the internal treatment of the body, there are several possibilities, even if in our simulations we have found that the external flow is essentially independent of the internal conditions.

A first possibility is to apply the forcing inside the body without any smoothing. This is equivalent to imposing the velocity distribution inside the body with the pressure that adjusts accordingly.

An alternative approach consists of leaving the interior of the body free to develop a flow without imposing anything. Of course, in this case the flow pattern inside the body will be different from the previous case, but the external flow is unchanged.

The last possibility we have investigated is to reverse the velocity at the first point inside the body in such a way that it still results in $\mathbf{u}=\mathbf{V}$ on the boundary. Again, only the internal flow pattern is different. Note that this internal treatment was required by [16] in spectral simulations to alleviate the problem of spurious oscillations near the boundary; this procedure was used also by Goldstein, Handler and Sirovich [12] for their simulations.

Extensive testing of these procedures has been performed to check the influence of the internal treatment of the body

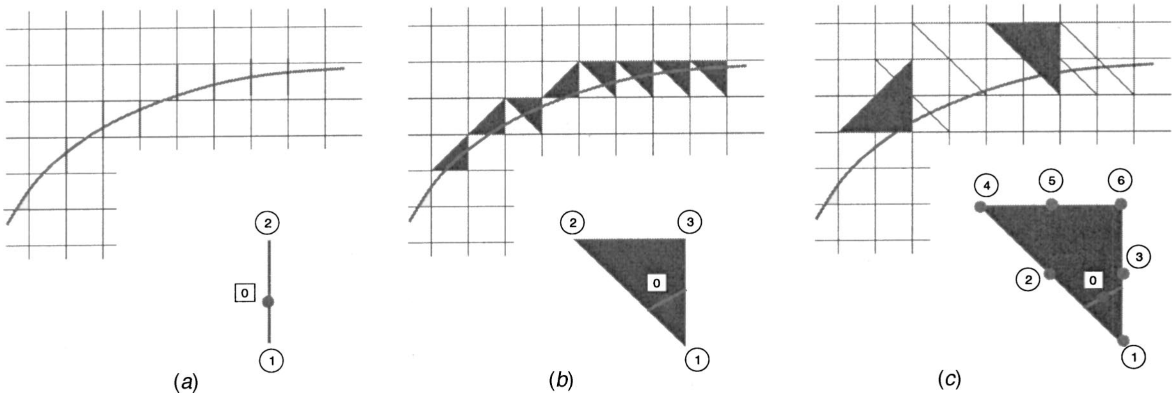

Fig. 1 Reconstruction stencils in the vicinity of the immersed boundary: *a*) Linear one-dimensional scheme, *b*) linear multi-dimensional scheme, and *c*) quadratic multi-dimensional scheme

on the accuracy and the efficiency of the scheme. We have found that, when using the direct forcing of Section 3.2.3, there is essentially no influence. Therefore, depending on the particular flow, the easiest treatment can be used. On the other hand, the feedback forcing of Section 3.2.1 required smaller values of α_f and β_f (in absolute value) when the velocity distribution inside the body was prescribed. Although this did not affect the external flow, lower values of α_f and β_f allowed the use of bigger time steps, thus improving the efficiency.

It is useful to point out that the solution in the interior of the domain is discarded and does not influence the physical solution outside; in other words, calculations performed in the inside are an unnecessary overhead. On the other hand, for specific applications (like the study of the conjugate solid/fluid heat transfer), the ability to compute a solution inside the solid might be an additional advantage of the present technique.

3.3 Description of the boundary immersion

The immersed objects can be described using CAD primitives directly, thus eliminating completely the need for CAD/CFD translations. The widely used Stereo-LiThography (STL) format is herein employed; the STL representation of a surface is a collection of unconnected triangles of sizes inversely proportional to the local curvature of the original surface. This format is already the standard for the *Rapid Prototyping* community and all the CAD systems have the ability to export a given surface in STL format automatically. This allows the treatment of any complex geometry without the need to generate a surface mesh; the *only* requirement for the object description is that the given surface must be a closed manifold. This is the same restriction enforced by rapid prototyping tools and guarantees that the final objects can be produced.

The geometrical preprocessor uses the CAD surface description and the underlying grid to generate all the interpolation data required by the IB Flow Solver. The geometrical module performs the separation (*tagging*) of the computational cells in *dead* (inside the body), *alive* (outside the body), and *interface* (partially inside). This procedure is based on a simple Ray Tracing (RT) technique normally used in computer graphics. A random ray which originates from the location to be checked (grid nodes) is considered and the intersections between this ray and the given surface are counted; if the total number is even (odd) the point is outside (inside) the object. The intersection between a ray (a 3D segment) and the surface (a collection of polygons) is carried out using the geometrical algorithms reported in O'Rourke [42]. The RANS flow solver requires only information at the nodes whereas the LES solver, due to the staggering of the variables, requires the same tagging performed at the faces of each control volumes; this is performed by analyzing the nodes belonging to each face. If all the nodes are dead (alive) the face is tagged as dead (alive); otherwise, it is considered as an interface.

The RT may fail due to incorrect surface representation (overlapping or missing triangles in the STL file); to overcome this difficulty, before tagging any location three perpendicular rays are cast (on a structured grid there are six edges from every node): if the corresponding result (odd or even intersections) is the same, the point is tagged, otherwise, up to 20 additional random rays are traced, and the most probable result is accepted. This *healing* process can be also approached in a different way by testing the STL file for inconsistency and by regenerating the surface triangulation [43]. After completing the tagging, some additional geometrical quantities are evaluated to perform the interpolation explained in the previous section.

Two example of the application of this geometrical tagging are reported. The first one represents a shark and it is shown in Fig. 2; the model is made up of 40,000 triangles and the solution on a coarse mesh is shown in terms of the stream traces. The second example shows a very realistic sports car (Fig. 3); this model is made up of more than half a million surface triangles, with all the details of the original geometry preserved; in this case, the computed pressure distribution on the surface is reported.

The total cost of the tagging and generation of the inter-

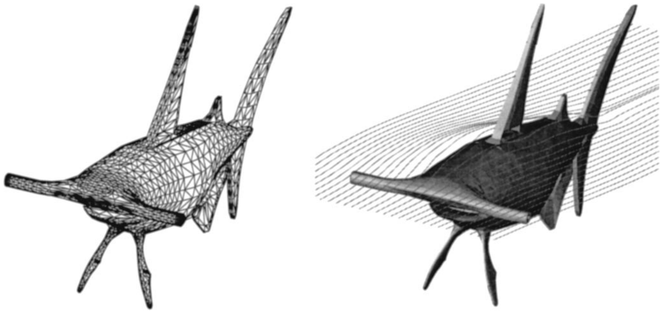

Fig. 2 STL model of a hammer-head shark and streamtraces of the computed flow field at Reynolds number of 1,000

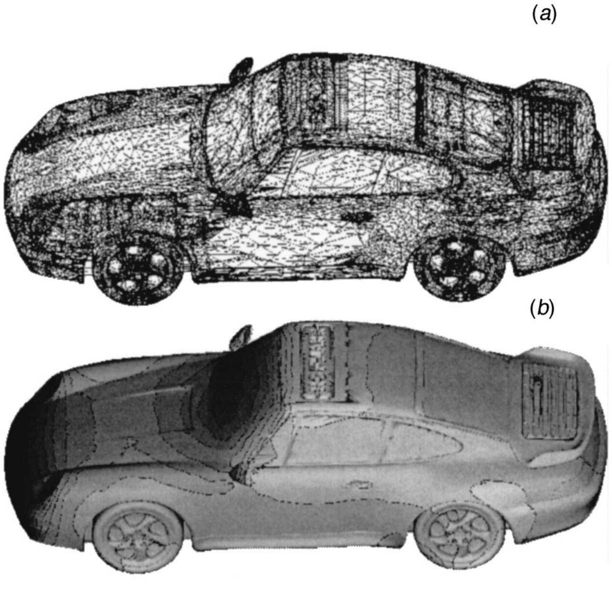

Fig. 3 STL model of a Porsche 911 and computed pressure distribution on the surface at Reynolds number of 100,000

polation data requires less than 20 seconds for the first example and about nine minutes for the second one by using an underlying grid of about one million grid points on a SGI R12K workstation.

It is useful to point out that the STL surface triangulation is not well suited as surface mesh for unstructured body fitted volume grid generation; this is due to the presence of highly skewed triangles in regions of low surface curvature.

3.4 Grid refinement technique

Cartesian methods are extremely effective in capturing smoothly varying solutions, but have difficulties in dealing with steep gradients because of the overhead associated with local resolution: even if fine grids are only required in a limited region, grid lines must be extended to the boundary of the domain. In the context of the IB techniques, very refined grids are often required close to high curvature immersed surfaces to properly represent the details of the geometry.

Grid resolution can be easily increased in the framework of unstructured meshes by locally inserting grid points and regenerating the connectivity between the points; the same procedure for Cartesian grids is not straightforward as it will destroy the implicit ordering of the mesh points. Many approaches for local grid refinement of Cartesian meshes have been proposed; the most successful is based on the work by Powell and De Zeeuw [44] and Pember *et al* [45]. It is based on the idea of *octree* data structure in which every computational cell can be subdivided in four (eight in three dimensions) children-cells. The main difficulty associated with this technique is the inherent complexity of the solution algorithm, which is reflected in high computational costs and memory requirements.

On the other hand, for Cartesian grids, mesh coordinate directions can be identified and used to number the cells and nodes. Thus, in two dimensions a cell is identified by its two indices i and j in the mesh coordinate system, and its neighbors are located by incrementing one of these indices. For unstructured meshes this is no longer possible, since, in principle, the cells and nodes are not ordered. The use of unstructured meshes requires the storage of connectivity information along with the use of an indirect addressing system.

If we consider the discretization of an elliptic operator (for example, the diffusion terms in the 2D Navier-Stokes Eqs. (2)), we can write a central second order finite difference discretization on a Cartesian grid as:

$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=\frac{1}{\Delta x}\left[\frac{\phi_{i+1,j}-\phi_{i,j}}{\Delta x}-\frac{\phi_{i,j}-\phi_{i-1,j}}{\Delta x}\right]+\frac{1}{\Delta y}\left[\frac{\phi_{i,j+1}-\phi_{i,j}}{\Delta y}-\frac{\phi_{i,j}-\phi_{i,j-1}}{\Delta y}\right] \tag{15}$$

where Δx and Δy are the mesh spacing in the two directions; Δx and Δy can depend, respectively, on x and y and in this case the mesh is nonuniform. This discretization is obviously limited to regular brick shaped physical domains. A more general representation can be obtained using curvilinear grids, where three sets of coordinate lines intersect in each point of the computational domain [46]. In this case the grid point locations must be explicitly defined (as the distance between two neighboring points is not constant) but the grid connectivity is still implicit in the initial ordering (the grid is said *structured*). Using unstructured grids, the computational domain is discretized by a disordered cloud of points and then the differential operator cannot be approximated by simple formulas as before. Polynomial or least square reconstructions in the neighbors of each vertex are built to evaluate the differential operators.

If we go back to the Eq. (15), we can generalize it to deal with local grid refinement. The idea is to introduce a certain degree of explicit connectivity but force any grid vertex to have two neighbors on each grid line. The differential operator can be discretized as:

$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=\frac{1}{\Delta x_{ij}}\left[\frac{\phi_{i^+_{ij},j}-\phi_{i,j}}{\Delta x^+_{ij}}-\frac{\phi_{i,j}-\phi_{i^-_{ij},j}}{\Delta x^-_{ij}}\right]+\frac{1}{\Delta y_{ij}}\left[\frac{\phi_{i,j^+_{ij}}-\phi_{i,j}}{\Delta y^+_{ij}}-\frac{\phi_{i,j}-\phi_{i,j^-_{ij}}}{\Delta y^-_{ij}}\right] \tag{16}$$

where i^-_{ij}, i^+_{ij}, j^-_{ij}, and j^+_{ij} are the only connectivity information required at each location (i,j). Δx^-_{ij}, Δx^+_{ij}, Δy^-_{ij}, and Δy^+_{ij} are the corresponding grid spacings and Δx_{ij}, Δy_{ij} their averages, respectively.

This approach is based on the observation that in a Cartesian mesh the locally refined grid can be viewed as a finer globally refined mesh with some grid lines partially deleted. In Fig. 4 the central region of the grid is refined but the grid lines are extended to the boundary of the computational domain; in this way, a (i,j) ordering of the vertices (unknowns) can be retained. In order to exploit the savings occurring with the local refinement strategy, unknowns located outside the central zone on the dashed lines are not considered in the problem. The discretization formula (16) can be used to link active grid points across the dashed lines, for example at the location $(i-1,j-1)$ where the point $(i-1,j+1)$ will be used instead of $(i-1,j)$. This procedure can be repeated for all the locations reported in Fig. 4, but not for the point (i,j) which is missing the left neighboring point. This location is called (in the unstructured context) a *hanging* node and the local solution is obtained via interpolation (reconstruction) from surroundings points. A detailed discussion of the implementation and accuracy issues related to the local grid refinement is outside the scope of the present review but it is reported in [47].

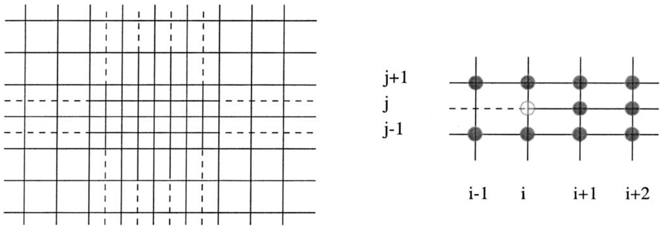

Fig. 4 Cartesian mesh with local mesh refinement: dashed lines represent grid lines that are partially deleted

As discussed in the previous section, the correct representation of curved immersed boundaries requires fine underlying grids; an automatic grid enrichment technique has been developed using the geometrical tagging introduced in the previous section and the ability to treat hanging nodes. In Fig. 5, a curved boundary (representing the letter F) is immersed on an underlying uniform grid (Fig. 5*a*). The tagging function T is shown for the initial coarse mesh; the dark area corresponds to internal cells $(T=-1)$ whereas the white area corresponds to fluid cells $(T=1)$. The inverse of the numerical gradient of this function is also reported: its value is proportional to the local grid size. By successively halving the cells until this gradient exceeds a prescribed value, the grid and the corresponding sharper geometrical representation in Fig. 5*d* is obtained. Note that the values of the gradient are increasing (the inverse is reported) from left to right in Fig. 5 because of the decreasing grid size. As an example, in Fig. 6, a calculation is carried out around the letters FPC using a grid adapted with the same procedure.

4 EXAMPLES

The IB approach in conjunction with the LES or the RANS simulation code has been applied to several problems; in the following, five test cases are reported. The first two represent fairly classical problems and the objective is to demonstrate the accuracy of the present technique as compared to standard body fitted approaches. The third and fourth problems illustrate the ability of the IB technique to handle problems with moving boundaries; more in detail, the third example has a truly moving boundary while the fourth as an impeller rotating at a constant angular velocity which, therefore, is fixed in the rotating frame of reference. The last one is an industrial-like problem and shows that the IB approach can be used as a design tool to evaluate the effect of different geometrical configurations without the need to regenerate computational grids.

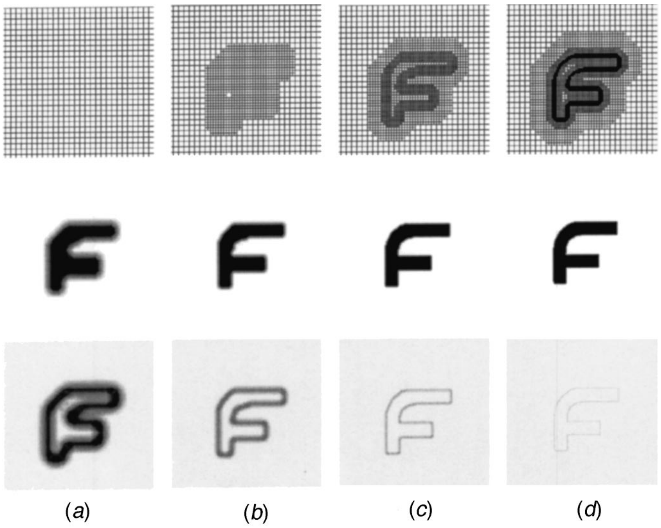

Fig. 5 Example of the automatic grid refinement strategy for immersed boundaries: Computational grids (top), heavyside tagging function (middle), and numerical derivative of the tagging function (bottom); *a* to *d* represent successive levels of refinement

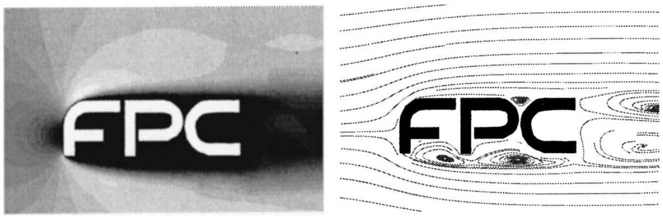

Fig. 6 Flow simulation around the FPC letters at Reynolds number 10,000

4.1 Flow around a cylinder

The flow around a circular cylinder has been extensively studied both numerically and experimentally for several decades. This flow, in fact, from one hand, is sufficiently simple to be analyzed in great detail while, on the other hand, still retains the physics of more complex flows. The flow regime is strongly dependent on the Reynolds number, Re (based on the diameter): it develops a steady symmetric recirculation for Re⩽40. while it sheds counter rotating 2D laminar vortices up to Re=190. From 190⩽Re⩽260 the shedding becomes 3D with span-wise perturbations of four cylinder diameter wavelength while for higher Reynolds values the perturbations develop on a finer scale, around one cylinder diameter. Given the low Reynolds values, all the described regimes are amenable to direct numerical simulation and they have been studied by many researchers (see the review by Williamson [48] and references therein). This flow is also an interesting benchmark for the immersed boundary methods since the flow dynamics is governed by the separation of the boundary layer from the cylinder surface. This implies that inaccurate treatments of the (immersed) body surface would result in unphysical perturbations on the layer and, therefore, in altered dynamics. The direct numerical simulation of this flow up to Re=300 with an immersed boundary method has been carried out by Orlandi *et al* [49]

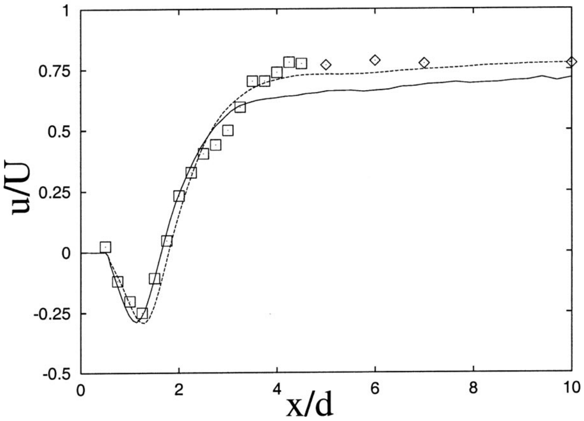

Fig. 7 Mean streamwise velocity on the center line of the wake of a circular cylinder at Re=3900: □ experimental results by Lourenco and Shih [50], ◇ experimental results by Ong and Wallace [51], ——dns on a 49×129×193 grid, – – –les with a dynamic SGS model on a 49×129×193 grid

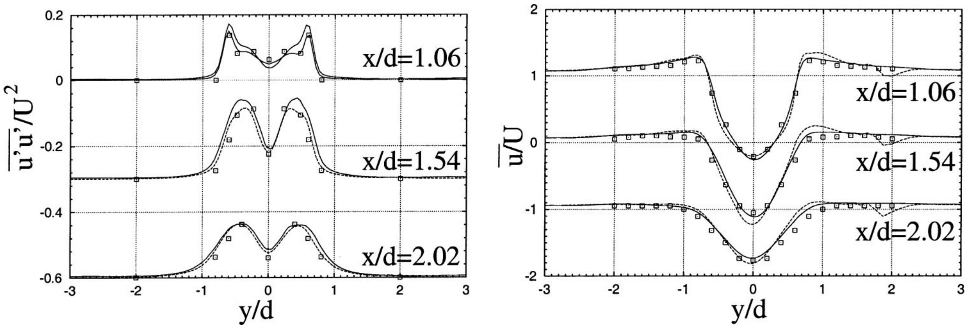

Fig. 8 Mean (left) and rms (right) cross-stream profiles of streamwise velocity in the wake of the cylinder. Sections are sampled at 1.06, 1.54, and 2.02 cylinder diameters downstream. The symbols are the results by Kravchenko and Moin [52], ——dns on a 49×129×193 grid, – – –les with a dynamic SGS model on a 49×129×193 grid.

confirming the above described findings and showing the capability of the method to capture the viscous boundary layer separation.

When the Reynolds number is increased up to 1200 (even if in the literature scattered values in between 300 and 1200 are reported [50]) the shear layers separating from the cylinder become unstable and the shed vortices contain fine scale structures. As a consequence, the cylinder wake experiences a dynamics different from that at lower Reynolds as experimentally shown in [51] and [52] at Re=3900. At this Reynolds number, direct numerical simulation, although affordable by modern supercomputers, has never been attempted, while large eddy simulation has been successfully employed. Kravchenko and Moin [53] have used B-splines on body fitted meshes obtaining a good agreement with experiments. These results were fully confirmed by Breuer [54] which also addressed the effect of the span-wise domain dimension.

The simulation of this flow by an immersed boundary approach poses a two-fold challenge, since the boundary layer developing at the cylinder surface is very thin and this must be captured with a non-body-fitted grid. In addition, the flow separation strongly depends on the wall dynamics which, in turn, requires the appropriate behavior of the sub-grid scale model at the wall. The results of Figs. 7 and 8 have been obtained by Tessicini [55] by a second order central finite difference code on a Cartesian mesh with a dynamic sub-grid scale model. The grid had 49×129×193 points, respectively, in the span-wise, cross-stream, and stream-wise directions and the computational domain was πD, $30D$, $30D$ in the same directions, D being the cylinder diameter; the mesh was uniform in the span-wise and nonuniform in the other directions in such a way as to cluster the nodes around the cylinder and downstream in the wake. These results show a satisfactory agreement with experiments and numerical simulations available from the literature and, similarly to Breuer's [54] findings, the flow in the region around the cylinder has little sensitivity to the turbulence model while it has a positive effect for the downstream wake evolution (Fig. 7).

One interesting feature of these results is the discrepancy of the LES simulation with respect to the simulation without sub-grid scale model and the results by [53] in the region $y/d=2$ (Fig. 8). This was found to be due to the uneven distribution of the grid points owing to an enlargement of the computational domain obtained by *patching* an extra strip to an old computational domain. In that region, therefore, the metric of the mesh was not smooth enough and this introduced numerical errors when computing finite difference derivatives. This problem was more severe for the LES simulation than for DNS since in the former case the sub-grid scale model requires the computation of the strain field in order to parameterize the turbulent stresses. This finding was unexpected since inaccuracies in a dynamically passive region were supposed to have negligible influence on the solution and, if any, certainly not on the mean flow statistics.

We wish to stress that, although accidental, the above mistake was instructive since, while it is clear that a symmetric grid must be used for the flow around a circular cylinder, the choice is not obvious for an arbitrary object; the sensitivity of the LES solutions to the *grid quality* is therefore an important factor to be accounted for in the LES of complex flows.

4.2 Flow in a wavy channel

The flow in a wavy channel (Fig. 9) is characterized by increased turbulence levels close to the wall with improved heat and mass transfer performance. Laminar and weakly turbulent flows have been analyzed using DNS by Leonardi and Orlandi [56] in order to investigate the physics of wall turbulence in the presence of surface roughness. In the present case, the Reynolds number is in the range of 10,000 to 100,000, therefore still relatively low, but nevertheless the problem is challenging for turbulence modeling. Kuzan's experimental measurements [57] are used to evaluate the accuracy of RANS simulations performed using the IB technique. Additional simulations performed using the body fitted approach (with the original code developed by Rogers and Kwak [58]) are also presented.

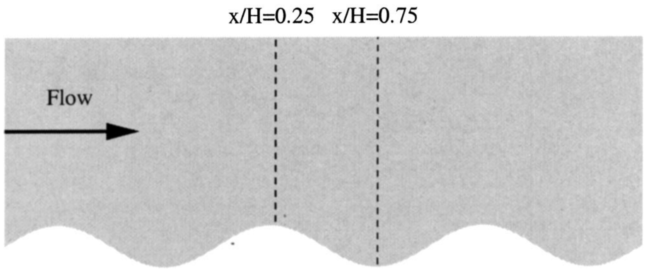

Fig. 9 Sketch of the wavy channel problem with the location of the measured velocity profiles

The wavy bottom wall has a sinusoidal shape whose amplitude and wave length are 0.1 m and 1.0 m, respectively. Since the flow is periodic, the computational domain can be chosen to cover only one period of the wavy channel. The length of the periodic domain is 1 m. Experimental data are available in two sections corresponding to the valley and the peak in Fig. 9. The Reynolds number (based on mean channel height and the mass flow) is about 8000.

The computational grids are reported in Fig. 10. The first one (Fig. 10*a*) is a body fitted mesh made up of 100×80 grid points with strong clustering at the channel walls. The second is a Cartesian mesh (100×60) underlying the sinusoidal bottom wall (Fig. 10*b*), whereas the third is locally refined in the region close to the immersed bottom wall (Fig. 10*c*).

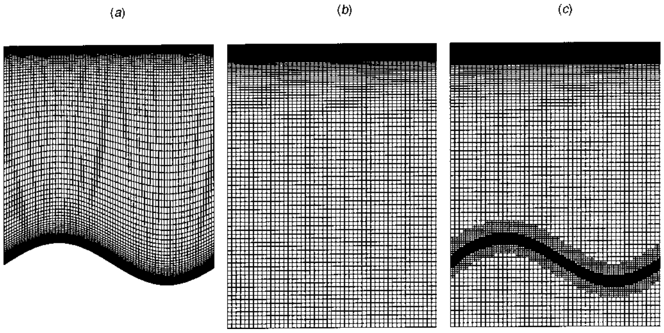

Fig. 10 Computational grids for the wavy channel RANS simulations: *a*) body fitted grid, *b*) Cartesian mesh, and *c*) locally refined Cartesian mesh

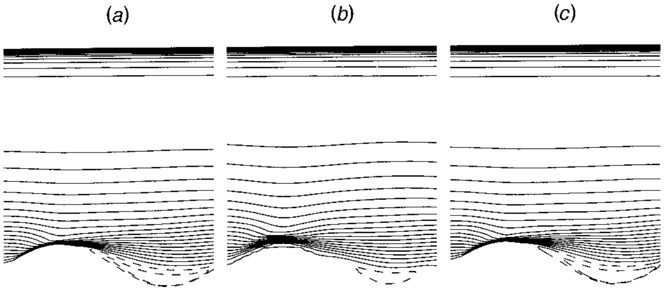

Fig. 11 Streamwise velocity component at Re=8000 (dashed line represent negative values): *a*) body fitted grid, *b*) Cartesian mesh, and *c*) locally refined Cartesian mesh

The solution is reported in Fig. 11 in terms of the streamwise velocity component; a large recirculation region is present downstream of the wavy peak (dashed lines); this is captured by the body fitted and the locally refined grid in a very similar way, while somewhat underpredicted by the uniform Cartesian mesh. This is confirmed by the analysis of the velocity profiles reported in Fig. 12. Very good agreement between the experiments and the calculations performed on the body fitted and the locally refined grid is observed; the calculation on the uniform Cartesian mesh captures the qualitative behavior of the flow but fails to capture the correct amount of separation. This clearly indicates the need for a very highly refined grid close to curved immersed boundaries to correctly represent sharp flow gradients.

4.3 Flow in a piston/cylinder assembly

The flow in an axisymmetric piston/cylinder assembly with a fixed valve has been simulated using both the LES and the RANS solver.

The configuration is reported in Fig. 13 and experimental measurements (phase averaged mean and RMS radial profiles of axial velocity) are available [59] for the validation of the numerical results. In the experiment, the piston was externally driven so that the fluid flowed into the cylinder from outside during the downward piston motion and vice versa when the piston moved up. Since the valve was fixed and a tiny annular gap was left open between the valve and the cylinder head, the compression phase is not included in the flow dynamics (the working fluid is still assumed to be incompressible). The piston was driven by a simple harmonic motion at a speed of 200 rpm≃21 rad/s which for the present geometry yields a mean piston speed of $\overline{V_p}=0.4\,\text{m/s}$ (when averaged over a half cycle). The Reynolds number of the flow based on $\overline{V_p}$ and on the piston radius is Re=2000 in air.

In Verzicco *et al* [18], all the details of the computation and the boundary conditions are given. In this paper we only mention that at the lower surface a prescribed mass flow is assigned with a constant axial velocity profile in such a way as to preserve the free divergence in the region between the

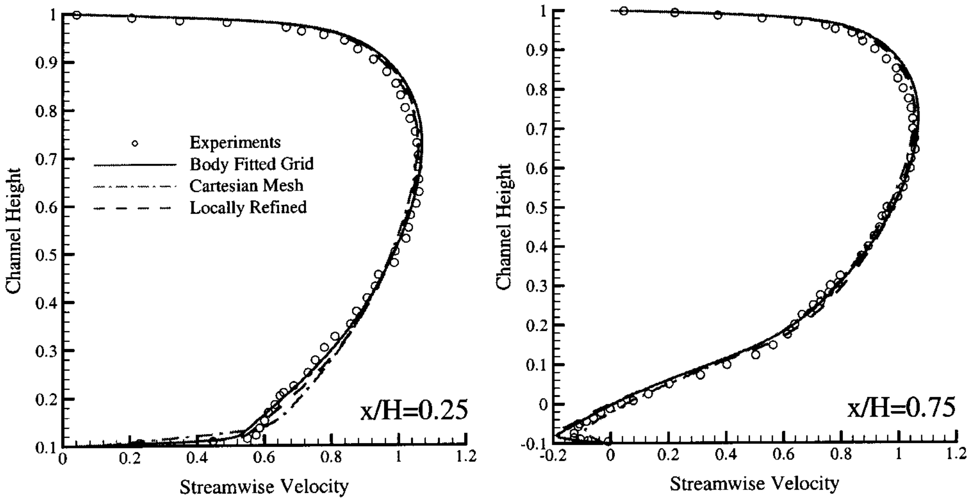

Fig. 12 Streamwise velocity component profiles compared with the experimental data

lower boundary and the piston. At the upper boundary, convective boundary conditions are used as extensively explained in [18].

In Fig. 13, snapshots during one instant of the oscillating cycle are given and the high three-dimensionality of the flow can be appreciated from the vector plots in orthogonal sections. Radial profiles of axial velocity were obtained by phase averaging the fields over four cycles and then averaged in the azimuthal direction. Three profiles at different axial locations are shown in Fig. 14. The comparison with the experimental data shows that the LES results are always in better agreement than the RANS, but overall both solutions represent the flow appropriately. In Verzicco *et al* [18], LES velocity profiles in additional sections and RMS profiles of axial velocity are also reported consistently showing a very good agreement with the measurements. In Haworth [60], the present LES results are also compared to simulations obtained using an unstructured boundary fitted, deformable mesh; the quality of the results is comparable even though the immersed boundary technique is much less expensive.

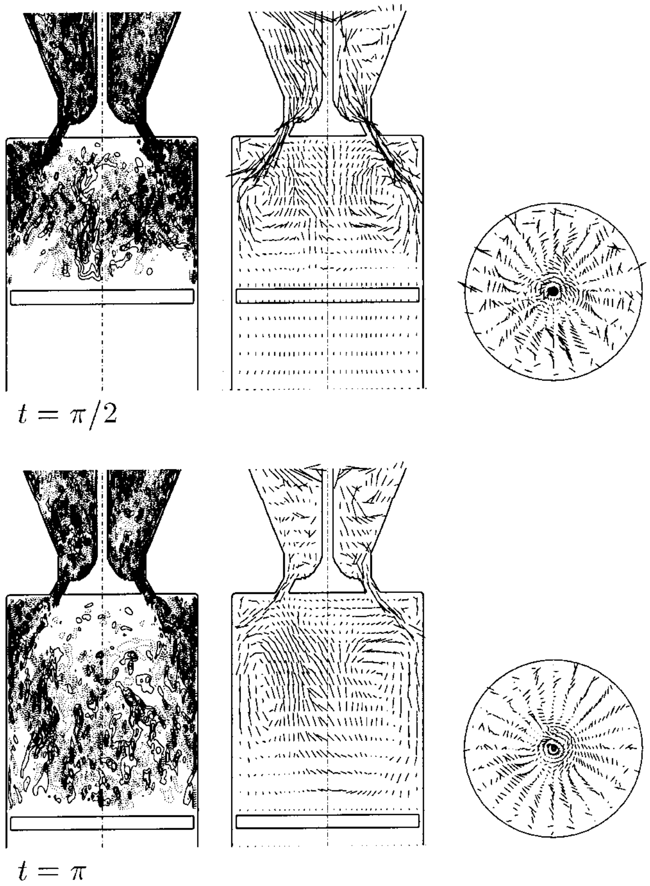

Fig. 13 Contour plots of azimuthal vorticity and velocity vectors projected onto 2D cutting planes for a 3D case with azimuthal perturbation at Re=2,000, 65×65×151 ($\theta\times r\times z$) grid, dynamic Smagorinsky subgrid-scale turbulence model. Vorticity scale solid lines (——) indicate positive values, dotted lines (· · · ·) indicate negative values, and the increment between adjacent isocontours is $\Delta\omega=\pm 2.5\bar{V}_p/b$: *a*) $t=\pi/2$, azimuthal vorticity; *b*) $t=\pi/2$, projected velocity vectors, meridional plane; *c*) $t=\pi/2$, projected velocity vectors, 15 mm below the head; *d*) $t=\pi$, azimuthal vorticity; *e*) $t=\pi$, projected velocity vectors, meridional plane; and *f*) $t=\pi$, projected velocity vectors, 15 mm below the head.

4.4 Flow in a stirred tank

The LES and RANS solvers have been used to investigate the flow in a cylindrical unbaffled tank stirred by an impeller located at mid-height of the tank, rotating at constant velocity Ω [61]. The impeller has eight blades equispaced over the whole azimuthal span; a sketch of the device is given in Fig. 15.

A computational grid made up of 192×102×97 nodes (in the vertical, radial and azimuthal direction respectively) has been used. The grid is uniform in the azimuthal direction and a section of it is reported in Fig. 15. The Navier-Stokes Eqs. (2) are solved in a frame of reference fixed with the impeller and therefore rotating with constant angular velocity Ω. The extra terms **f** are prescribed at each time step to establish the desired velocity $\mathbf{V}_b=0$ (in the rotating frame) on the impeller and shaft surfaces. On the external cylindrical wall of the tank, a constant azimuthal velocity is imposed $V_\theta(R)=-\Omega R$ (where R is the radius of the tank), while the bottom horizontal no-slip surface moves according to $V_\theta(r)=-\Omega r$ with r the radial coordinate. A slip boundary condition is imposed on the upper boundary of the computational domain. The Reynolds number based on the rotational speed and the blade radius (R_b) is Re=1636.

Indeed, we have verified that the moderate value of the Reynolds number and the fine grid used were such that the sub-grid scale turbulence model was essentially inactive and LES and DNS results were indistinguishable. The results are presented in Fig. 16 in terms of azimuthally averaged velocity vectors, instantaneous velocity magnitude and turbulent kinetic energy. The meridional plane is roughly divided into two halves by the radial jet emanating from the impeller. The two recirculation regions are not symmetric owing to the different boundary conditions on the upper and lower horizontal surfaces. The flow is strongly unsteady (as evident from Fig. 16*b*); in the meridional recirculation, it is essentially dominated by the rotation period of the impeller, while the flow in the jet has a wide frequency content, ranging from the slow vertical flapping up to the fast vortex shedding from the impeller blade tips [62]. In addition, the flow is strongly inhomogeneous in space with quiet and laminar-like recirculation far from the impeller and a turbulent jet region at the center of the tank (Fig. 16*c*).

Quantitative comparisons between the present simulations and experimental data is reported in Fig. 17 in terms of radial profiles of azimuthal, radial, and vertical velocity components. The present results are in very good agreement with the measurements; in particular, the peaks of the azimuthal and radial velocity close to the impeller are very well captured. RANS predictions show large discrepancies, especially in terms of the radial velocity, which is strongly overestimated. This is mainly due to the presence of large scale unsteadiness and inhomogeneous turbulence.

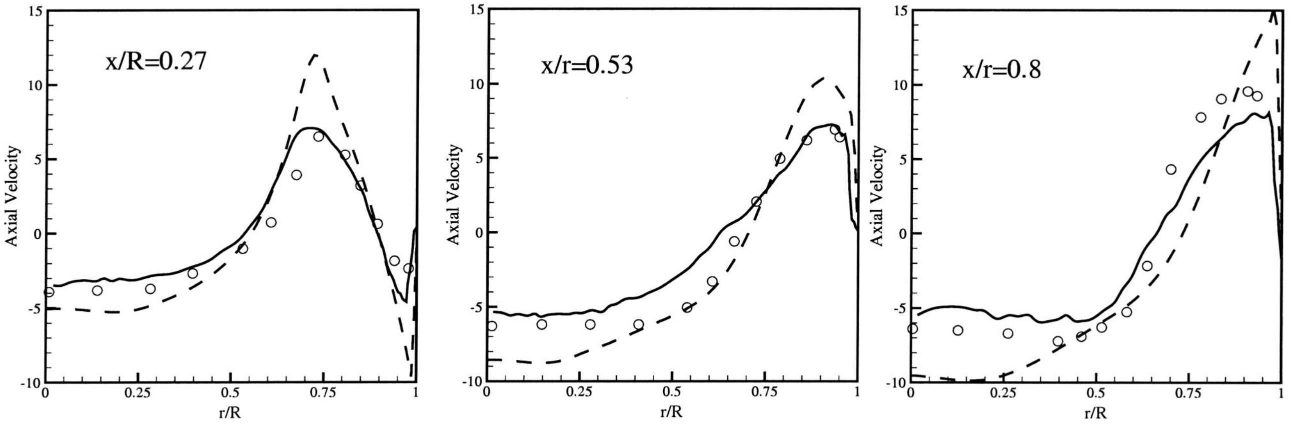

Fig. 14 Radial profiles of averaged axial velocity components at different locations in the cylinder. Symbols: Experiments [58], Solid Line: LES simulation; Dashed line: RANS simulations

4.5 Flow around road vehicles

The LES solver with the IB approach has been used to simulate the flow around a square back road vehicle with drag reduction appendices attached to its base. The objective is to study the unsteady dynamics of the wake and the modifications induced by the drag reduction devices; experimental data are available for comparison [63].

The baseline configuration is reported in Fig. 18; the simulations are performed on a Cartesian grid made up of 220×140×257 points in the stream-wise, vertical, and spanwise directions, respectively.

The experimental Reynolds number based on the free

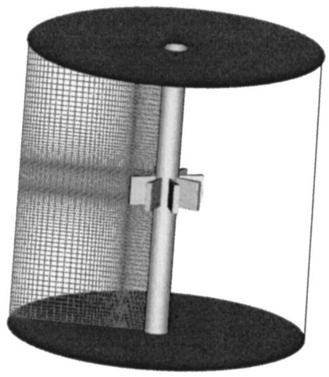

Fig. 15 Tank configuration and computational grid in a meridional plane (only one every six grid-points are shown)

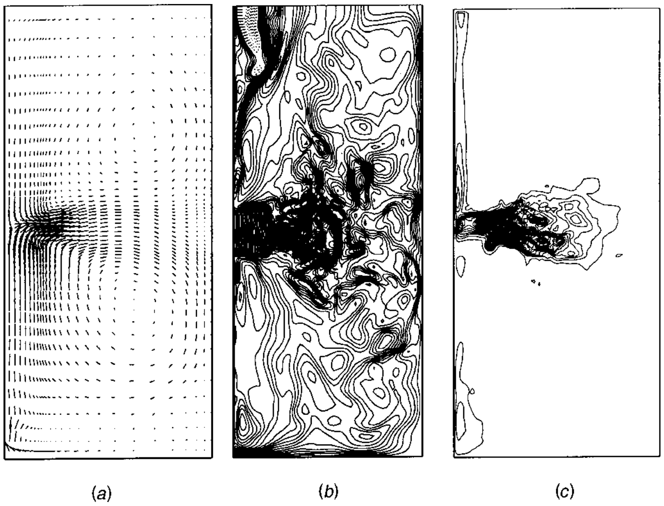

Fig. 16 Contour plots of azimuthally averaged velocity vectors: *a*), instantaneous velocity magnitude, *b*) and turbulent kinetic energy, *c*) in a meridional plane crossing a blade

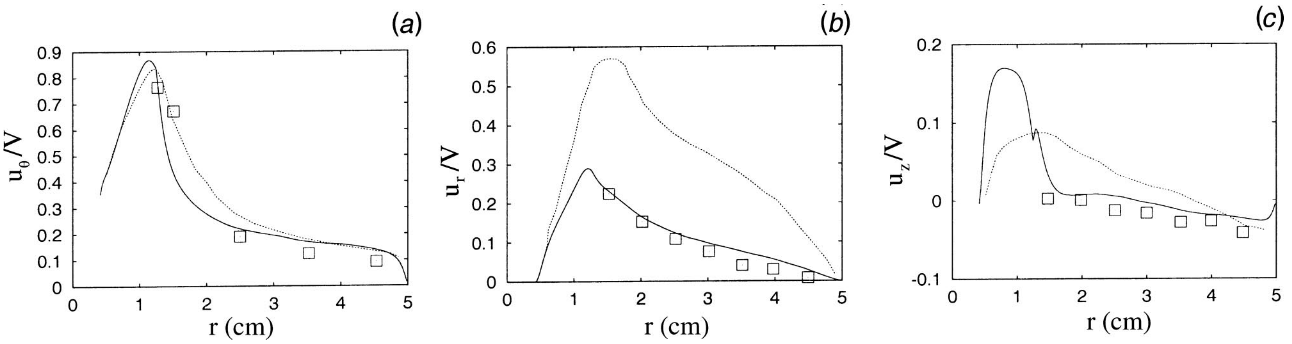

Fig. 17 Radial profiles of averaged velocity components in the middle of the tank. Symbols: Experiments [60], Solid line: Present LES; Dashed line: RANS simulations [61]

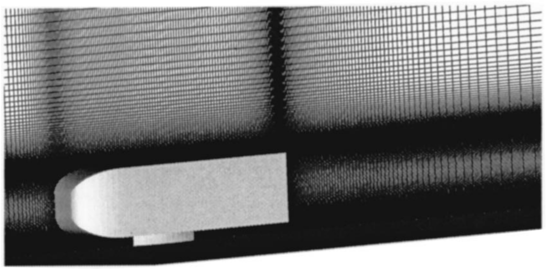

Fig. 18 Road-vehicle configuration and computational grid in the symmetry plane (only one every four grid-points are shown)

stream velocity and the model height (*H*) is Re=170,000. Preliminary simulations were carried out assuming that the main features of the flow and the corresponding trends in the flow dynamics at the back of the body were independent of Reynolds number if this was sufficiently high. Accordingly, the Reynolds number of the numerical simulations was fixed at Re=20,000; it was observed that, indeed, the numerical simulations showed all the trends and the flow features observed in the experiments. However, some quantitative differences were present. For this reason, additional simulations have been performed at Re=100,000 showing a much better quantitative agreement with the experimental data. Quantitative results are reported in Fig. 19 in terms of time averaged stream-wise velocity profiles in two sections downstream of the base for the square back configurations. The measurements are compared with two LES simulations performed at Re=20,000 and Re=100,000; the high Reynolds number simulations agree very well with the experiments. The defect velocity as well as the length of the recirculation region are accurately captured. The low Reynolds number simulations agree qualitatively with the measurements but strongly overpredict the thickness of the bottom wall boundary layer.

The LES flow solver in combination with the IB technique was then used to study the influence of modification of the geometry of the back of the vehicle on the wake dynamics and overall drag coefficients. The use of a the Cartesian mesh reported in Fig. 18 allows to perform the simulations very efficiently without the need to regenerate a computational grid for every configuration.

Qualitative results are reported in Fig. 20 for the three configurations analyzed and for the lower Reynolds number.

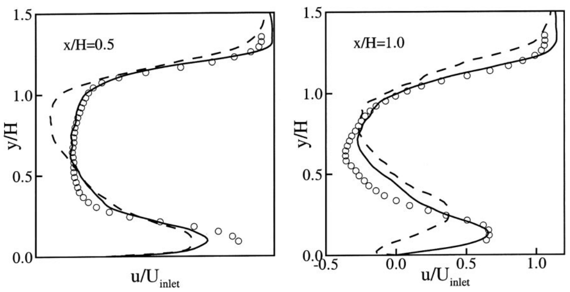

Fig. 19 Streamwise velocity profiles in the wake for the square-back configuration. Symbols: Experiments [62]; Dotted line: LES at Re=20,000; Solid line: LES at Re=100,000.

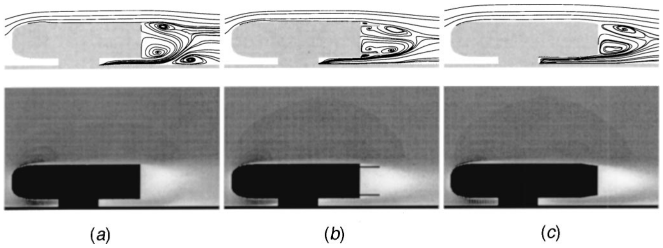

Fig. 20 Flow patterns in the symmetry plane superimposed to contours of time-averaged streamwise velocity: Re=20,000 *a*) Baseline square-back geometry, *b*) Square-back with base plates, *c*) Boat-tail base

The flow patterns in the near wake recirculation region are very different; the results for the baseline square back configuration show a strong interaction between the base recirculation and the boundary layer on the bottom wall. A more top to bottom symmetric wake is obtained with the other two configurations; for these two the base pressure is higher than for the square back and, therefore, the drag is lower.

The high Reynolds number results have also been compared to the experiments in terms of drag co-efficients; a value of 0.291 for the square back and 0.223 for the boat tail were computed from the LES simulations, whereas 0.3 and 0.23 were the corresponding measurements. Additional analyses were carried out to study the unsteady dynamics of the flow (for the three configurations); the wake of the baseline geometry is characterized by a low frequency streamwise *pumping*, a strong vertical flapping, and high frequency vortex pairing in the shear layers detaching from the body. The low frequency modes are strongly suppressed by the drag reduction devices; the same behavior was observed in the experiments. Additional comparison and detailed discussion of the results are reported in Verzicco *et al* [64].

5 CONCLUSIONS

In this review, we have reported some recent results on the application of the Immersed Boundary Method to LES and RANS turbulent flow simulations. The Immersed Boundary approach allows the use of simple and efficient numerical techniques on regular grids for simulations of fluid flow around complex configurations; the complexity of the geometry is accounted for only in the governing equations using additional terms. Details of the forcing definition and the treatment of the immersed surfaces have been discussed and two techniques for predicting turbulent flows have been used, the standard Reynolds-Averaged Navier-Stokes and the more advanced Large Eddy Simulations. Several examples have shown the efficacy of the IB technique in simulating complex, turbulent flows using simple Cartesian-like meshes.

Although the discussed examples encourage the use of the present approach for complex, industrially relevant applications, there are a number of points that still need a detailed investigation.

For example, the local grid refinement technique described in Section 3.4 was shown to work very well in combination with RANS modeling, while nothing is known about its application to LES simulations. This is a crucial point since the need for grid resolution close to solid walls is the main limitation to the application of LES to high Reynolds flows. This observation shifts the focus on the recent wall models for LES described by Balaras and Benocci [65], Balaras *et al* [66], Cabot [67] and Cabot and Moin [68] which have been successfully used with body fitted meshes [69,70], while they have never been integrated in Immersed Boundary solvers. We believe that a combination of LES turbulence modeling with the appropriate wall treatment and the Immersed Boundary approach (possibly with a local grid refinement) would make LES an industrial production tool, similarly to the present RANS simulations.

ACKNOWLEDGMENTS

We wish to thank S Leonardi, F Tessicini, Dr M Fatica, and Prof P Orlandi for providing some of the results shown in the paper and for many fruitful discussions during the preparation of the manuscript.

REFERENCES

[1] Vieceli JA (1969), A method for including arbitrary external boundaries in the MAC incompressible fluid computing technique, *J. Comput. Phys.* **4**, 543–551.

[2] Welch JE, Harlow FH, Shannon JP, and Daly BJ (1966), A computing technique for solving viscous incompressible transient fluid flow problems involving free-surfaces, Report LA-3425, Los Alamos Scientific Lab.

[3] Harlow FH and Welch JE (1965), Numerical calculation of time-dependent viscous incompressible flows of fluid with free surface, *Phys. Fluids* **8**, 2182.

[4] Vieceli JA (1971), A computing method for incompressible flows bounded by moving walls, *J. Comput. Phys.* **8**, 119–143.

[5] Peskin CS (1972), Flow patterns around heart valves: A digital computer method for solving the equations of motion, PhD thesis, Physiology, Albert Einstein College of Medicine, Univ Microfilms 72–30, 378.

[6] Peskin CS (1977), Numerical analysis of blood flow in the heart, *J. Comput. Phys.* **25**, 220–252.

[7] Peskin CS (1982), The fluid dynamics of heart valves: Experimental, theoretical and computational methods, *Annu. Rev. Fluid Mech.* **14**, 235.

[8] Peskin CS and McQueen DM (1989), A three-dimensional computational method for blood flow in the heart: (I) immersed elastic fibers in a viscous incompressible fluid, *J. Comput. Phys.* **81**, 372–405.

[9] McQueen DM and Peskin CS (1989), A three-dimensional computational method for blood flow in the heart: (II) contractile fibers, *J. Comput. Phys.* **82**, 289–297.

[10] Basdevant C and Sadourny R (1984), Numerical solution of incompressible flow: the mask method, Lab di Meteorologie Dynamique, Ecole Normale Superieure, Paris (unpublished).

[11] Briscolini M and Santangelo P (1989), Development of the mask method for incompressible unsteady flows, *J. Comput. Phys.* **84**, 57–75.

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

[13] Saiki EM and Biringen S (1996), Numerical simulation of a cylinder in uniform flow: Application of a virtual boundary method, *J. Comput. Phys.* **123**, 450.

[14] LeVeque RJ and Calhoun D (1999), Cartesian grid methods for fluid flow in complex geometries, *Computational Modeling in Biological Fluid Dynamics*, LJ Fauci and S Gueron (eds) IMA Volumes in Mathematics and its Applications, **124**, 117–143.

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

[16] Mohd-Yosuf J (1997), Combined immersed-boundary/B-spline methods for simulations of flow in complex geometries, *Annual Research Briefs, Center for Turbulence Research*, 317–328.

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

[18] Verzicco R, Mohd-Yusof J, Orlandi P, and Haworth D (2000), LES in complex geometries using boundary body forces, *AIAA J.* **38**, 427–433.

[19] Khadra K, Parneix S, Angot P, and Caltagirone JP (1995), Fictitious domain approach for numerical modeling of Navier-Stokes equations, *4th Int Conf on Navier-Stokes Equations and Related Nonlinear Problem*.

[20] Khadra K, Angot P, Parneix S, and Caltagirone JP (2000), Fictitious domain approach for numerical modelling of Navier-Stokes equations, *Int. J. Numer. Methods Fluids* **34**, 651–684.

[21] Angot P, Bruneau CH, and Frabrie P (1999), A penalization method to take into account obstacles in viscous flows, *Numerische Mathematik* **81**, 497–520.

[22] Kevlahan N and Ghidaglia JM (2001), Computation of turbulent flow past an array of cylinders using a spectral method with Brinkman penalization, *Eur. J. Mech. B/Fluids* **20**, 333–350.

[23] Berger M and Aftosmis M (1998), Aspects (and aspect ratios) of Cartesian mesh methods, *16th Int Conf on Numerical Methods in Fluid Dynamics*.

[24] Aftosmis MJ, Berger MJ, and Adomavicius G (2000), A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries, AIAA Paper 2000-0808.

[25] Forrer H (1997), Boundary treatments for Cartesian-grid methods, PhD thesis, Swiss Federal Inst of Tech.

[26] Ye T, Mittal R, Udaykumar HS, and Shyy W (1999), An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries, *J. Comput. Phys.* **156**, 209–240.

[27] Coirier WJ (1994), An adaptively-refined, Cartesian, cell-based scheme for the Euler and Navier-Stokes equations, NASA TM106754.

[28] Almgren AS, Bell JB, Colella P, and Marthaler T (1997), A Cartesian grid projection method for the incompressible Euler equations in complex geometries, *SIAM J. Sci. Comput. (USA)* **18**, 1289–1309.

[29] Hirt CW and Nichols BD (1981), Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys.*, **39**, 201–225.

[30] Sethian JA (1996), *Level Set Methods and Fast Marching Methods, Cambridge Univ Press.*

[31] Duchamp de Lageneste L and Pitsch H (2000), A level-set approach to large eddy simulation of premixed turbulent combustion, Ann Res Briefs-2000, Center for Turbulence Research, Stanford Univ, 105–116.

[32] Germano M, Piomelli U, Moin P, and Cabot WH (1991), A dynamic subgrid-scale eddy viscosity model, *Phys. Fluids A* **3**, 1760–1765.

[33] Lilly DK (1992), A proposed modification of the Germano subgrid-scale closure method, *Phys. Fluids A* **4**, 633–635.

[34] Smagorinsky J (1963), General circulation experiments with primitive equations, *Mon. Weather Rev.* **91**, 99–164.

[35] Verzicco R and Orlandi P (1996), A finite-difference scheme for three-dimensional incompressible flows in cylindrical coordinates, *J. Comput. Phys.* **123**, 402.

[36] Orlandi P (2000), *Fluid Flow Phenomena: A Numerical Toolkit*, Kluwer Acad Pub, 355.

[37] Jones WP, and Launder BE (1972), The prediction of laminarization with a two-equation model of turbulence, *Int. J. Heat Mass Transf.* **15**, 1–32.

[38] Spalart PR and Allmaras SR (1994), A one-equation turbulence model for aerodynamic flows, *La Recherche Aerospatiale*, **1**, 1–23.

[39] Majumdar S, Iaccarino G, and Durbin PA (2001), RANS solver with adaptive structured boundary non-conforming grids, *Annual Research Briefs 2001*, Center For Turbulence Research, 353.

[40] Franke R (1982), Scattered data interpolation: Tests of some methods, *Math. Comput.* **38**, 181–200.

[41] Tessicini F, Verzicco R, and Orlandi P (2001), Effetti della geometria dell'ugello sull'evoluzione di un getto libero, *Atti del* **XV** *Congresso AIMETA*, **SP FL 20**, 2001 (see also Nozzle geometry effects in the near field of a round jet, *Jets in Cross Flow*, CISM, Udine, (2001) Cortelezzi and Karagozian (eds) Springer–Verlag).

[42] O'Rourke J (1998), *Computational Geometry in C*, Cambridge University Press.

[43] Aftosmis MJ, Delanaye M, and Haimes R (1999), Automatic generation of CFD-ready surface triangulations from CAD geometry, AIAA Paper 99–0776.

[44] De Zeeuw D and Powell KG (1993), An adaptive Cartesian mesh method for the Euler equations, *J. Comput. Phys.* **104**, 5668.

[45] Pember RJ, Bell BJ, Colella P, Crutchfield WJ, and Welcome ML

(1995), An adaptive Cartesian grid method for unsteady compressible flow in irregular regions, *J. Comput. Phys.* **1120**, 278–304.

[46] Ferziger JH and Peric M (1999), *Computational Methods for Fluid Dynamics*, Springer-Verlag.

[47] Durbin PA and Iaccarino G (2002), An approach for local grid refinement of structured grids, *J. Comput. Phys.* **181**, 639–653.

[48] Williamson CHK (1996), Vortex dynamics in the cylinder wake, *Annu. Rev. Fluid Mech.* **28**, 477.

[49] Orlandi P, Leonardi S, Tessicini F and Verzicco R (2001), Scie tridimensionali generate da cilindri con e senza ondulazioni, *Atti del* **XV** *Congresso AIMETA*, **SP FL 12**, Taormina, Italy.

[50] Prasad A and Williamson CHK (1996), The instability of separated shear layer from a bluff body, *Phys. Fluids* **8**, 1347.

[51] Lourenco L and Shih C 2000, Characteristics of the plane turbulent near wake of a circular cylinder: A particle image velocimetry study (Unpublished results taken from [53]).

[52] Ong L and Wallace J (1996), The velocity field of a turbulent very near wake of a circular cylinder, *Exp. Fluids* **20**, 441.

[53] Kravchenko A and Moin P (2000), Numerical studies of flow over a circular cylinder at $\mathrm{Re}_D=3900$, *Phys. Fluids* **12**, 403–417.

[54] Breuer M (1997), Numerical modeling influences on large eddy simulations for the flow past a circular cylinder, *Proc of 11th Symp on Turbulent Shear Flows*, Grenoble, France, 26:7–26:12.

[55] Tessicini F (2002), Large eddy simulation of the flow around a circular cylinder using an immersed boundary method, Paper in preparation, (personal communication).

[56] Leonardi S and Orlandi P (2001) DNS of turbulent flows in a channel with roughness, *Proc of Dles4*, Univ of Twente Enschede, Netherlands, 181–189.

[57] Kuzan JD (1986), Velocity measurements for turbulent separated and near-separated flows over solid waves, PhD thesis, Dept of Chem Eng, Univ of Illinois, Urbana IL.

[58] Rogers S and Kwak D (1990), Upwind differencing scheme for the time-accurate incompressible Navier-Stokes equations, *AIAA J.* **28(2)**, 253–262.

[59] Morse AP, Whitelaw JH, and Yianneskis M (1978) Turbulent flow measurements by laser doppler anemometry in a motored reciprocating engine, Report FS/78/24, Imperial College, Dept of Mesh Eng.

[60] Haworth DC and Jansen K (1997), Large-eddy simulation on unstructured deforming meshes: Towards reciprocating IC engines, *Comput. Fluids* (submitted).

[61] Dong L, Johansen ST, and Engh TA (1994), Flow induced by an impeller in an unbaffled tank-I. experimental, *Chem. Eng. Sci.* **49**(4), 549.

[62] Verzicco R, Iaccarino G, Fatica M, and Orlandi P (2000), Flow in an impeller stirred tank using an immersed boundary technique, *Annual Research Briefs 2000*, Center For Turbulence Research, 251 (submitted to *AIChE J*).

[63] Khalighi B, Zhang S, Koromilas C, Balkanyi SR, Bernal LP, Iaccarino G, and Moin P (2001), Experimental and computational study of unsteady wake flow behind a bluff body with a drag reduction device, SAE Paper, 2001-01B-207.

[64] Verzicco R, Fatica M, Iaccarino G, Moin P, and Khalighi B (2002), Large eddy simulation of a road-vehicle with drag reduction devices, *AIAA J* **40**(12), 2447–2455.

[65] Balaras E and Benocci C (1994), Subgrid-scale models in finite-difference simulations in complex wall bounded flows, *AGARD* **CP 551**, 2.1.

[66] Balaras E, Benocci C, and Piomelli U (1996), Two-layer approximate boundary conditions for large-eddy simulations, *AIAA J.* **34**, 1111–1119.

[67] Cabot WH (1995), Large-eddy simulation with wall models, *Ann Res Briefs-1995*, Center for Turbulence Research, Stanford Univ, 41–50.

[68] Cabot WH, and Moin P (1999), Approximate wall boundary conditions in the large-eddy simulation of high Reynolds number flows, *Flow, Turbul. Combust.* **63**, 269–291.

[69] Wang M and Moin P (2002), Dynamic wall modeling for LES of complex turbulent flows, *Phys. Fluids.* **14**(7), 2043–2051.

[70] Wang M, and Moin P (2000), Computation of trailing-edge flow and noise using large-eddy simulation, *AIAA J.* **38**, 2201–2209.

***Gianluca Iaccarino** has been a research associate at the Center for Turbulence Research in Stanford University since 1998. In 1993, after graduating in Aeronautical Engineering at the Università di Napoli (Italy), he worked at CIRA (Italian Aerospace Research Center) in the Aeronautical and Propulsion Department. He has authored or co-authored 15 articles in refereed journals and more than 35 papers in conference proceedings. Iaccarino's research interests are in computational fluid dynamics, turbulence modeling, and grid generation. He is a member of APS and AIAA.*

***Roberto Verzicco** is Associate Professor of Fluid Mechanics in the Dipartimento di Ingegneria Meccanica e Gestionale (DIMeG) and Excellence Center of Computational Mechanics (CEMeC) of Politecnico di Bari, a position he has held since 1998. In 1991, he graduated in Aeronautical Engineering at the Università di Roma. He received the PhD degree in Fluid Dynamics in 1995. He has authored or co-authored over 35 papers in refereed journals and 45 papers in conference proceedings. Verzicco's research interests are in computational fluid dynamics, vortex dynamics, turbulence modeling, and geophysical fluid dynamics. He is a member of APS, EUROMECH, ERCOFTAC, and AIMETA (Associazione Italiana di Meccanica Teorica ed Applicata).*