V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal,14-17 June 2010

COMPARISON OF BODY-FITTED AND IMMERSED BOUNDARY METHODS FOR BIOMECHANICAL APPLICATIONS

Bruno Tayllamin[*], Simon Mendez[*], Ramiro Moreno[†], Ming Chau[††], Franck Nicoud[*]

[*]I3M, University Montpellier 2, France
bruno.tayllamin@math.univ-montp2.fr, simon.mendez@math.univ-montp2.fr, franck.nicoud@univ-montp2.fr
[†]INSERM U 858 I2MR, University Hospital Toulouse-Rangueil, France
moreno.r@chu-toulouse.fr
[††]Advanced Solutions Accelerator, Montpellier, France
mchau@advancedsolutionsaccelerator.com

Key words: CFD, Immersed boundary, blood flow, stenosis, aorta

Abstract. *New Immersed Boundary methods for a staggered grid are developed and compared with a reference Body-Fitted method in the context of biomedical blood flow simulations. The Immersed Boundary methods developed consist in reconstructing the velocity value close to the wall using different interpolation procedures while keeping a homogeneous Neumann boundary condition for pressure. The methods are devised to be robust enough in order to cope with complex geometry arising in the biomedical context. The capability of the Immersed Boundary methods to achieve mass conservation is assessed when considering the case of the flow through a stenotic vessel. To demonstrate the potential of these Immersed Boundary methods to cope with more physilogical flow, simulation of a pulsatile flow through an aortic cross was performed. In both flow cases, the flow solution obtained using the Immersed Boundary methods is compared with that obtained using a Body-Fitted method.*

1 INTRODUCTION

The most common approach in Computational Fluid Dynamics (CFD) in complex geometry is to make use of a body-fitted (BF) method based either on finite volume or finite element methodology. This approach has led to accurate simulations of blood flow in arteries [5,9]. However, the generation of the body-fitted grid is time consuming and requires an engineering knowledge from the user. Thus, the use of a body-fitted method for blood flow simulation is restrained to research area and has limited benefit for clinical purposes.

The Immersed Boundary Method (IBM) has emerged as an alternate method[1,4] which does not require any grid generation task. Simulations of flow into complex geometries[4] are done on a Cartesian grid which can be automatically generated, the presence of solid boundaries being accounted for by appropriate forcing added to the classical flow equations. The method is thus more appropriate for a practical use of blood flow simulations in clinical routine.

Using a cartesian grid has the disadvantage that, in general, the physical boundary of the body does not conform to the grid; it is possible that not even a single node belonging to the grid matches the physical boundary. The difficulty is thus to set onto the cartesian grid the boundary conditions needed to solve the flow equations, altough these normally apply at the body surface. The IBM permits to cope with this difficulty by enforcing the effects of the physical boundary conditions over the flow solution computed on the cartesian grid. Several formulations of the IBM have been proposed for the forcing procedure (see Mittal et Iaccarino[4] for a review). These formulations belong either to a discrete or a continuous approach. In the former, the forcing is applied directly on the discretized flow equations while in the latter the forcing generally takes the form of a body force term added to the continous flow equations.

In the reconstruction formulation the value of the flow variables is directly enforced on the cartesian grid nodes closest to the physical boundary instead of being computed by solving the fluid flow equations[1]. The enforced value of a flow variable usually takes the form of a linear combination between the value of that variable at the physical boundary and its value further inside the flow domain. Fadlun *et al.*[1] used a linear interpolation for enforcing the velocity value on nodes closest to the physical boundary.

The enforced variables serve as boundary conditions for solving the flow equations inside the flow domain of the cartesian grid. In Fadlun *et al.*[1] only velocity was enforced. Thus the flow equations were numerically solved using a Dirichlet boundary condition for velocity over the nodes closest to the physical boundary. However, the pressure boundary condition over the boundary nodes was let undefined; pressure was computed on every node of the grid including the nodes where velocity was enforced. When using a pressure correction scheme, the pressure gradient at boundary nodes can modify the enforced velocity value, possibily decreasing the accuracy of the reconstruction procedure[2]. Kang *et al.*[3] show another drawback of letting the pressure boundary condition undefined. Due to the discretizing procedure, the pressure derivatives at the nodes closest to the body surface are computed using both physical pressure on nodes in the fluid domain and unphysical pressure on nodes outside. An unphysical "pressure coupling"[3] between the flow domain and the outside can then take place. According to these authors, this "pressure coupling" can affect the whole flow solution depending on the flow case considered. These previous studies demonstrate the need to restrain the pressure computation to the part of the cartesian grid where the fluid flows. This implies to enforce both a velocity and a pressure boundary condition at nodes in the fluid domain closest to the body surface.

The question of making velocity and pressure boundary conditions consistent has been raised by Ikeno et Kajishima[2]. This question is of particular relevance when a staggered grid is used since pressure nodes do not coincide with velocity nodes. Ikeno et Kajishima[2] assessed the effect of different treatments of the pressure boundary condition while using an interpolation scheme to enforce velocity on a staggered grid. In the present study, different velocity reconstruction methods are compared, while enforcing a homogenous Neumann pressure boundary condition on a staggered grid. We follow the strategy of Kang *et al.*[3] originally formulated on a collocated mesh, where pressure is only computed over the flow domain of the grid. We propose new forcing methods for velocity on staggered grids and compare them with existing ones.

The numerical method is described in section 2, where the different formulations for velocity reconstruction are detailed. An academic test case corresponding to a locally constricted pipe is then discussed in section 3. The occlusion mimics the presence of an atherosclerotic plaque narrowing the lumen of a human vessel. Results from our IBM are notably compared with a reference solution obtained using a high-order body-fitted (BF) numerical tool. Special attention is paid to the error in mass conservation, which will serve as a quality criterion. Eventually, the application of the method to a pulsatile flow through an in vitro model of an aorta cross is presented in section 4. The general satisfactory agreement between the IBM and the BF results demonstrate the potential of the former for clinical routine applications.

2 FORMALISM

We assume blood to be a Newtonian fluid and the flow to be incompressible. Thus the governing equations are the 3-D Navier-Stokes (N.S.) equations (with summation convention):

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j \partial x_j}, \tag{1}$$

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{2}$$

over Ω_{fluid} (see Fig. 1) where u_i is the i^{th} component of the flow velocity vector $\boldsymbol{U}$, x_i the i^{th} space variable, t is the time variable, ρ is the density, p is the pressure, and ν is the kinematic viscosity.

Assuming a one-step explicit time integration scheme, the time discretized equations are at time step n:

$$u_i^{n+1} = u_i^n + \Delta t(-u_j^n \frac{\partial u_i^n}{\partial x_j} - \frac{1}{\rho}\frac{\partial p^{n+1}}{\partial x_i} + \nu \frac{\partial^2 u_i^n}{\partial x_j \partial x_j}), \tag{3}$$

$$\frac{\partial u_i^{n+1}}{\partial x_i} = 0, \tag{4}$$

where u_i^{n+1} and p^{n+1} are to be computed.

2.1 Numerical method

The projection scheme on a staggered cartesian grid initialy proposed by Harlow et Welch[8] is used in this study. It is recalled in the following for completness and to introduce useful notations. A classical way to solve the coupled system Eqs. (3) and (4) is to use a projection method[8] where the time stepping is split in two steps: the prediction and the correction steps. During the prediction step, equation

$$\bar{u}_i = u_i^n + \Delta t(-u_j^n \frac{\partial u_i^n}{\partial x_j} - \frac{1}{\rho}\frac{\partial p^n}{\partial x_i} + \nu \frac{\partial^2 u_i^n}{\partial x_j \partial x_j}) \tag{5}$$

is used to compute a predicted velocity $\bar{u}_i$ which is not divergence free. In the correction step, velocity and pressure are advanced in time using:

$$u_i^{n+1} = \bar{u} - \frac{\Delta t}{\rho}\frac{\partial \delta p}{\partial x_i} \tag{6}$$

$$p^{n+1} = p^n + \delta p, \tag{7}$$

where the pressure update δp is computed so that the velocity field at time $n+1$ is divergence free:

$$\frac{\partial \bar{u}_i}{\partial x_i} = \frac{\Delta t}{\rho}\frac{\partial^2 \delta p}{\partial x_i \partial x_i} \tag{8}$$

Equations (5) to (8) can easily be embedded into a three-stage Runge-Kutta scheme in order to increase accuracy. The only change compared to the previous equations is to replace Δt by $\alpha_k \Delta t$ where $(\alpha_1, \alpha_2, \alpha_3) = (0.5, 0.5, 1)$. A staggered cartesian grid is then used to discretize the spatial derivatives of (5) to (8) thanks to finite differences:

$$\bar{u}_i = u_i^n + \Delta t(-u_j^n \frac{\delta^1 u_i^n}{\Delta x_j} - \frac{1}{\rho}\frac{\delta^1 p^n}{\Delta x_i} + \nu \frac{\delta^2 u_i^n}{\Delta x_j \Delta x_j}) \tag{9}$$

$$\frac{\delta^1 \bar{u}_i}{\Delta x_i} = \frac{\Delta t}{\rho}\frac{\delta^2 \delta p}{\Delta x_i \Delta x_i} \tag{10}$$

$$u_i^{n+1} = \bar{u}_i - \frac{\Delta t}{\rho}\frac{\delta^1 \delta p}{\Delta x_i} \tag{11}$$

where $\frac{\delta^1}{\Delta x_j}$ stands for centered 1st order derivative in the j direction and $\frac{\delta^2}{\Delta x_j^2}$ for the centered 2nd order derivative. Algebraic equations (9) to (11) are to be solved for u_i^{n+1} and p^{n+1} at each nodes included in Ω_{fluid}.

The numerical method being fully explicit, the time step must satisfy both the convective and diffusive stability criteria. In the present study, the CFL and Fourier numbers were set to 0.5 and 0.4 respectively, well below their theoretical limit values. Still, some numerical instabilities were sometimes observed for complex flows solved on coarse meshes. For stability improvement, a 4^{th}-order Artificial Viscosity (AV4) term is thus added to (9) yielding:

$$\bar{u} = u_i^n + \Delta t(-u_j^n \frac{\delta^1 u_i^n}{\Delta x_j} - \frac{1}{\rho}\frac{\delta^1 p^n}{\Delta x_i} + \nu \frac{\delta^2 u_i^n}{\Delta x_j \Delta x_j}) - sC^*_{av4}\Delta x_j^4 \frac{\delta^4 u_i^n}{\Delta x_j^4}, \tag{12}$$

where $\frac{\delta^4}{\Delta x_j^4}$ stands for the 4^{th}-order derivative in the j direction. The amount of AV4 depends on the value of s, a user parameter set between 0 and 1. Besides, C^*_{av4} is the value of the AV4 coefficient for which the 4^{th}-order artificial term is comparable to the physical 2^{nd}-order diffusive term. This value is obtained by considering a 1-D flow model and assuming the complex exponential

$$u = e^{ikx} \tag{13}$$

to be part of the numerical solution, where k is the wave number and x the space variable. The coefficient C^*_{av4} is assessed by writing:

$$\Delta t \nu \frac{\delta^2 u}{\Delta x^2} \sim C^*_{av4}\Delta x^4 \frac{\delta^4 u}{\Delta x^4}. \tag{14}$$

Considering $k = \frac{2\pi}{L}$ to be a physically relevant wave number, where L is the characteristic length of the computational domain, Eqs. (13) and (14) lead to:

$$C^*_{av4} \sim \Delta t \frac{\nu N_x^4}{(2\pi L)^2}, \tag{15}$$

where N_x is the characteristic number of grid nodes. Using coefficient C^*_{av4} as computed in (15) will dissipate even the largest scales of size L. This would be the case when the user parameter s is set to unity. Very small values of s are obviously preferred so that only a small amount of artificial viscosity is added. The AV4 acts as a numerical diffusive term for high spatial frequency variables value that are unphysical noise. Application to the 3D problem is straighforward, the user parameters being the characteristic length of the flow and value of the scaling s.

2.2 Immersed Boundaries

For inlets and outlets, classical Dirichlet and non-reflecting conditions are used. On the other hand, a specific treatment is required to handle solid boundaries. Note that only rigid walls are considered in this study. Calling Γ_{ib} such solid boundary, the physical conditions to impose over Γ_{ib} are obviously:

$$u_{i,\Gamma_{ib}} = 0, \tag{16}$$

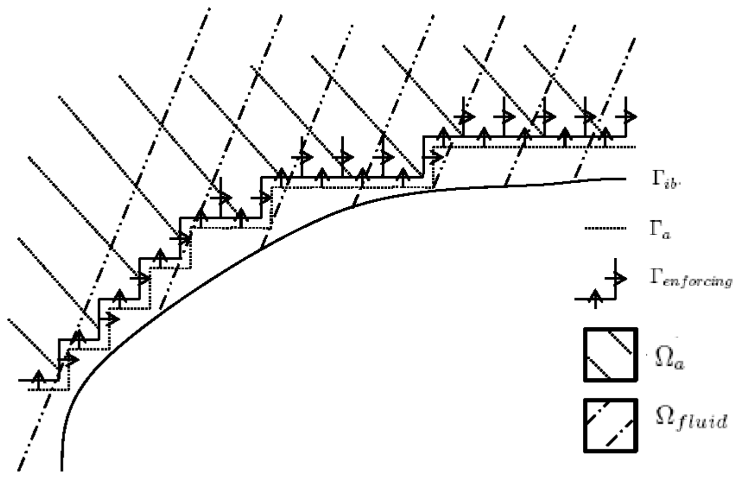

Figure 1: Immersed Boundary

$$\frac{\partial \delta p}{\partial x_i} n_i = 0, \tag{17}$$

where $\boldsymbol{n} = (n_1, n_2, n_3)$ is the unit vector normal to Γ_{ib}. In this paper, $\boldsymbol{n}$ is chosen inward. In order to locate the physical boundary Γ_{ib} on the cartesian grid, we use a distance fonction ϕ defined on each node of the grid. This fonction gives at each node the normal distance to the closest boundary Γ_{ib}. ϕ is defined as:

$$\phi(\boldsymbol{x_i}) = 0 \qquad \boldsymbol{x_i} \in \Gamma_{ib}, \tag{18}$$

$$\phi(\boldsymbol{x_i}) > 0 \qquad \boldsymbol{x_i} \in \Omega_{fluid}, \tag{19}$$

$$\phi(\boldsymbol{x_i}) < 0 \qquad \boldsymbol{x_i} \notin \Omega_{fluid}, \tag{20}$$

where $\boldsymbol{x_i}$ stands for the position vector of a grid node where the i^{th} component of the velocity is defined. Following Kang *et al.*[3] we approximate the physical domain Ω_{fluid} by the computational domain Ω_a[3] (see Fig. 1). Ω_a is defined as containing all the cells entirely lying inside Ω_{fluid} (see Fig. 2). We also define Γ_a as being the boundary of Ω_a and $\Gamma_{enforcing}$ as refering to all the faces which receive a specific treatment through the immersed boundary method. Of course, $\Gamma_{enforcing}$ contains Γ_a since a boundary condition must be given on each face belonging to the boundary of the computational domain. Note that due to the staggered grid arrangement, $\Gamma_{enforcing}$ also contains the faces interior to Ω_a that have at least one neighbour outside (see Fig. 1). Four forcing methods are considered in this paper. Independantly of these methods, the velocity value outside Ω_a is always set to zero.

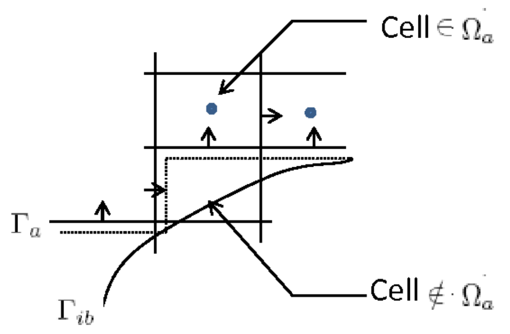

Figure 2: The Approximated Domain on the Staggered grid

The first method, denoted by "A", consists in forcing the velocity value over Γ_a to its value over Γ_{ib} given by (16). In other words, the boundary condition (16), which normally applies on Γ_{ib}, is replaced by $u_i = 0$ on Γ_a. This method is rather similar to the "step wise way" of Fadlun *et al.*[1]. The remaining nodes of $\Gamma_{enforcing}$ that do not belong to Γ_a do not receive any specific treatment. Instead, velocity at these nodes is obtained by solving the flow equations, as it is done further inside Ω_a. Velocity is forced to zero outside Ω_a. Note that this makes the effective zero velocity boundary somewhat different than the stair step like boundary Γ_a.

In the second forcing method "B", the boundary condition (16), which normally applies on Γ_{ib}, is replaced by a Dirichlet condition on $\Gamma_{enforcing}$:

$$\bar{u}_{i,enforcing} = \alpha \bar{u}_{i,\Omega_a}. \tag{21}$$

This is a linear interpolation where $\alpha = \dfrac{\phi_{\boldsymbol{x_{i,enforcing}}}}{\phi_{\boldsymbol{x_{i,\Omega_a}}}}$ and where $\bar{u}_{i,\Omega_a}$ is the velocity solution to the discretized N.S. equations at $\boldsymbol{x_{i,\Omega_a}}$. The node $\boldsymbol{x_{i,\Omega_a}}$ is chosen among the neighbours of $\boldsymbol{x_{i,enforcing}}$ belonging to Ω_a in such a way to maximize the quantity:

$$\frac{\boldsymbol{x_{i,\Omega_a}} - \boldsymbol{x_{i,enforcing}}}{\|\boldsymbol{x_{i,\Omega_a}} - \boldsymbol{x_{i,enforcing}}\|} \cdot \boldsymbol{n}, \tag{22}$$

where

$$\boldsymbol{n} = \frac{\boldsymbol{\nabla}\phi(\boldsymbol{x_{i,enforcing}})}{\|\boldsymbol{\nabla}\phi(\boldsymbol{x_{i,enforcing}})\|} \tag{23}$$

is an approximation to the local normal to the body surface Γ_{ib}. In doing so, the inner node which is the most aligned to the immersed boundary normal is selected. Clearly, this approach is only 1^{st}-order accurate and could be improved by using several internal nodes $\boldsymbol{x_{i,\Omega_a}}$ to define $u_{i,enforcing}$. However, this simple approach was preferred in this study where the main focus is put on mass conservation issues and biomedical applications, where rather large uncertainties regarding the geometry definition are present.

In the last forcing methods "C" and "D", a correction is added to the forcing "B". The first step is to compute, before the interpolation procedure "B", the velocity $\boldsymbol{U}$ at the center of the neighbour cells of $\boldsymbol{x_{i,enforcing}}$. The second step consists in applying the forcing "B" as described above. Finally, a correction to the enforced value $\bar{u}_{i,enforcing}$ is computed and added. The correction is meant to remove the normal component of the flow velocity at $\boldsymbol{x_{i,enforcing}}$:

$$\bar{u}_{i,enforcing} = \alpha \bar{u}_{i,\Omega_a} - (\bar{u}_{j,cell} n_j) n_i \tag{24}$$

where $\bar{u}_{j,cell}$ is computed at the center of neighbour cells before the forcing "B" is applied. In method "C", $u_{j,cell}$ is computed in the neighbour cell that belongs to the fluid domain Ω_a. In "D", $u_{j,cell}$ is computed at exact location of $\boldsymbol{x_{i,enforcing}}$ using the two neighbour cells, should they belong to Ω_a or not. For all the approaches "A", "B", "C", and "D" the homogeneous Neumann boundary condition for pressure (17) is enforced over Γ_a. Thus, on theses nodes, the pressure correction (10) does not act. In other words, the velocity values enforced on Γ_a by the IBM remain unchanged after the projection step has been completed.

2.3 The BF numerical tool : the YALES2 solver

YALES2 (http://nonpremixed.insa-rouen.fr/∼moureau/yales2.html) is an unstructured, body-fitted Computational Fluid Dynamics solver, covering a wide range of applications[6]. Thanks to its object-oriented structure, using modules in Fortran 90, it is very modular and includes multi-physics solvers: solvers for incompressible and compressible flows, level-set, magneto-hydrodynamic, spray, lagrangian particle tracking solvers, among others. YALES2 relies on central finite-volume schemes of fourth order in space. A fourth-order time integration is used. For the present application, the incompressible Navier–Stokes equations are solved using a fractional step method. The deflated PCG method is used for the Poisson solver. YALES2 is a massively parallel CFD solver, based on MPI.

3 IDEALIZED STENOSIS CONFIGURATION

A first simple configuration is studied to evaluate the different IBM implementations, presented earlier. Grid-converged YALES2 results are generated in the same configuration, and are used as a reference.

3.1 Presentation of the computations

3.1.1 Geometry

The configuration studied in this section is an idealized stenosis. It is one of the most popular geometries in computational fluid dynamics studies for cardio-vascular applications. To mimic stenotic flows in arteries, a rigid straight pipe is considered, with a constriction. The geometry is perfectly axisymmetric. The diameter of the pipe is $D = 0.05$ m far from the stenosis, and the diameter is halved at the throat of the stenosis. The axis of the geometry is the z axis. The origin of the axes is located at the center of the throat. The radius of the stenosis is

$$r(z) = \frac{D}{2}[1 - 0.25(1 + cos(2\pi z/L))] \quad \text{if}|\text{z}| \leq \text{L}/2, \tag{25}$$

$$r(z) = \frac{D}{2} \quad \text{if}|\text{z}| \geq \text{L}/2. \tag{26}$$

The length of the constriction is $L = 4\,D$. The domain inlet is located at $z = -10\,D$, while the outlet is located at $z = 10\,D$ in the computations performed with the BF solver and $z = 30\,D$ in the computations performed with the IBM solver. This extension of the domain has been decided for the IBM computations to observe the portion downstream of the constriction, where the flow goes back to the Poiseuille solution.

3.1.2 Grid

The grid used in the computations depends on the solver: for the computations using the IBM solver, a cartesian grid with constant spacing $\Delta x/D = 0.03$, $\Delta y/D = 0.03$ and $\Delta z/D = 0.2$ is used. The BF solver uses body-fitted unstructured grids. In the computations presented here, two grids are used. Grid 1 contains 3,194,042 tetrahedral cells. Typical cell sizes are $0.025\,D$ at the throat and $0.05\,D$ far from the throat. Grid 2 is a direct refinement of grid 1 by automatic refinement: each tetrahedral cell is cut into 8 cells by YALES2. Grid 2 thus contains 25,552,336 tetrahedral cells, with typical cell sizes $0.0125\,D$ at the throat and $0.025\,D$ far from the throat. The use of the two grids allows to check the grid convergence of the solution. Figure 3 shows the geometry of the configuration and of a view of the surface mesh of grid 1 in the throat region.

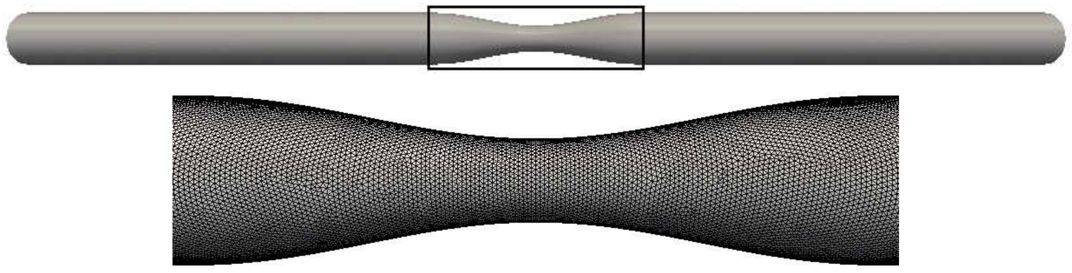
Figure 3: Top: computational domain for the BF calculations S5 and S6 ($-10\,D \leq z \leq 10\,D$). Bottom: view of the surface mesh of grid 1 (S5); zoom in the throat region shown in the top figure.

3.1.3 Run parameters

Simulations are laminar, with the inflow Reynolds number $Re = W_0 D/\nu \approx 250$, with $W_0 = 0.0172$ m s^{-1} the uniform velocity imposed at the inlet condition and $\nu = 3.5 \times 10^{-6}$

$m^2.s^{-1}$ the kinematic viscosity. Table 1 reports the parameters of the simulations presented in the following section. As only one operating point is considered, only numerical parameters change between simulations: solver, grid, time step and implementation for the IBM. In table 1, the time step Δt is defined by comparison with a characteristic time $t_0 = D/W_0 = 2.91$ s.

Simulation	Solver	Grid	IBM	Time step $\Delta t/t_0$
S1	IBM	40x40x200	A	1.93×10^{-2}
S2	IBM	40x40x200	B	1.87×10^{-2}
S3	IBM	40x40x200	C	1.89×10^{-2}
S4	IBM	40x40x200	D	1.88×10^{-2}
S5	BF	grid 1	-	2.63×10^{-3}
S6	BF	grid 2	-	5.50×10^{-3}

Table 1: Operating conditions and numerical characteristics of the simulations presented. The reader is referred to section 2.2 for the definition of the immersed boundaries implementation methods.

3.2 Results

Before comparing IBM and BF results, two preliminary features have to be studied. First, if BF results are considered as reference results, they have to be independent of the grid. This is what is shown in the next paragraph. Concerning IBM results, a critical aspect is mass conservation. Thus, § 3.2.2 is devoted to the evaluation of the different IBM implementations in terms of mass conservation. The last part of this section will compare IBM and BF results in terms of pressure and velocity profiles.

3.2.1 BF simulations: grid convergence results

Grid convergence has been tested thanks to automatic refinement available in YALES2. Grid 1 was refined by cutting each tetrahedral cell into eight cells. This automatic refinement uses the method developed by Rivara[7]. This capacity allows to perform grid refinement studies in a simple and efficient manner, without reusing a grid generator. Velocity and pressure profiles were extracted from simulations S5 and S6, and no major difference is observed between the two calculations, as shown in Fig. 4. Hence, when comparing to IBM simulations, BF results will be extracted only from simulation S6.

3.2.2 IBM simulations: mass conservation

One major issue when using immersed boundaries method is the mass conservation. As solid boundaries are not described by the mesh itself, the impermeable wall condition is not represented by a zero normal velocity at the wall. It is thus of major importance to control mass conservation in the calculations. This paragraph is dedicated to the

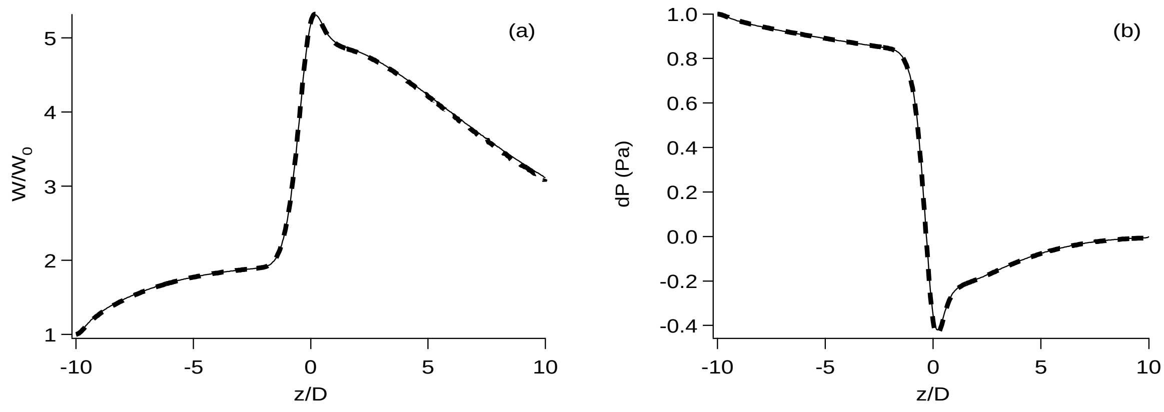

Figure 4: Comparison of runs S5 (- - -) and S6 (——) to show grid convergence in the YALES2 results. Axial velocity (a) and pressure (b) profiles are extracted over the centerline.

comparisons of the IBM implementations regarding the mass (or volume) flow rate along the domain.

The volume flow rate is calculated at each cross-section of the domain described by the mesh (*ie* every $\Delta z/D = 0.2$) by integration over the faces entirely included in the fluid domain. The volume flow rate can be plotted as a function of the axial coordinate z, as in Fig. 5.

Figure 5(a) displays the volume flow rate results by non-dimensionalizing Q by the inlet volume flow rate Q_0. As expected, method A (simulation S1) ensures perfect mass conservation ($Q = Q_0$ along the domain). The three other simulations show changes in the volume flow rate near the inlet boundary condition. This is due to the prescription of constant velocity at the inlet condition. Imposing constant inflow velocity, which ignores the effect of the adjacent non-slipping condition at the wall, leads to important non-zero wall normal velocity near the intersection between the wall and the inflow. Gain or loss of mass is thus observed near the inlet. Further downstream (and upstream of the constriction), the flow becomes parallel to the wall and Q is conserved for all methods. The constriction is the region where the volume flow rate varies most. Whatever the IBM (expect for method A of course), the simulation shows a loss of mass where the cross-section decreases and a gain followed by a small loss where it expands. Variations of volume flow rate are the highest for simulation S2 (IBM B) and the smallest for S4 (IBM D). In the downstream portion of the domain, mass conservation is ensured when the flow reaches a parallel state. Figure 5(b) allows to compare the methods by non-dimensionalizing Q by a value typical of the volume flow rate effectively reaching the constriction. This allows to visualize the net gain/loss of mass due to the constriction (without the inflow effect). Method D (S4) is shown to be the best method for mass conservation (except for method A that is designed to strictly conserve mass).

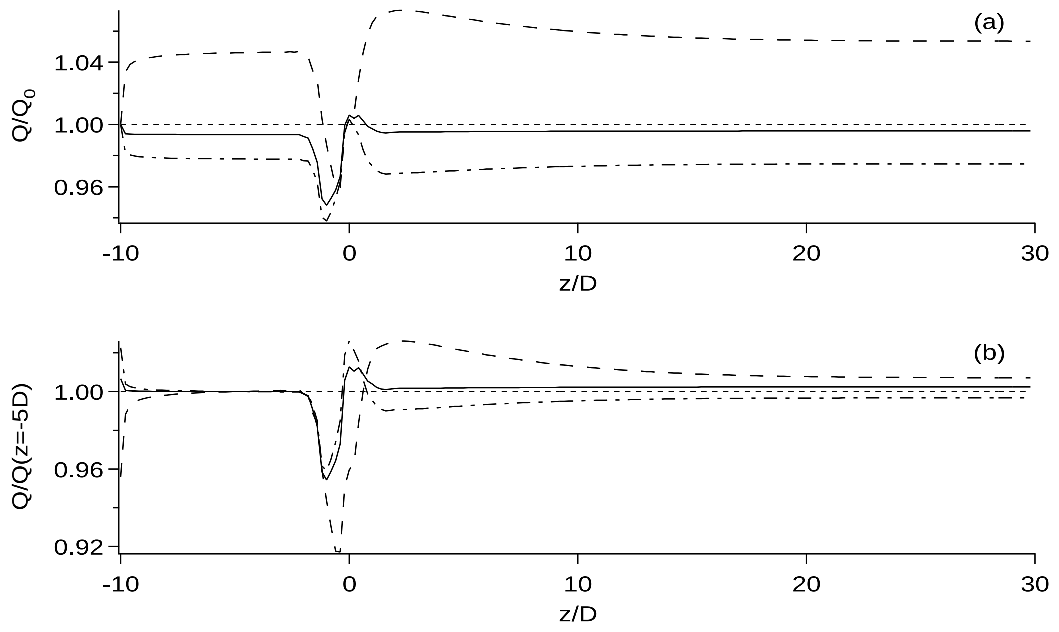

Figure 5: Volume flow rate as a function of the axial coordinate z/D for simulations S1 (- - - -), S2 (— —), S3 (— - —), S4 (——). The volume flow rate is non-dimensionalized by the inlet volume flow rate Q_0 (a) or the volume flow rate at $z = -5D$, in the straight tube portion upstream of the constriction (b).

3.2.3 Comparisons of the IBM and BF results

The aim of this section is to compare results from the IBM and the BF solvers. First, pressure is shown over the centerline (Fig. 6). As the mass flow rate is different in each simulation, direct comparison is not easy. Figure 6 shows the shape of evolution of pressure over the centerline, by comparing $(P - P_o) - (P_i - P_o)$, P_i being the pressure at the inlet condition ($z/D = -10$) and P_o the pressure where the outlet condition is located in simulation S6 ($z/D = 10$). Pressure results from the IBM simulations all reproduce the shape of the results obtained in the BF simulation S6. Better comparison is obtained with method D (simulation S4). The bottom figure of Fig. 6 displays a pressure field from simulation S4, to show the smoothness of the instantaneous field obtained using the IBM solver. This is an advanteage of following the "pressure decoupling" strategy of Kang *et al.*[3]. It allows the flow solution in the computational domain to stay unaffected by unphysical variables lying in the remaining part of the grid that could otherwise, through the pressure computation, cause local and unphysical irregularities.

Streamwise and transverse velocity profiles extracted from the symmetry plane $y = 0$ are now presented. Eight profiles are shown, at $z/D = -3; -2; -1; 0; 1; 2; 3; 4,$ for simulations S1 to S4 (IBM) and S6 (BF). Comparing velocity profiles is not that easy, as the same inlet condition for all simulations ($W_0 = 0.0172$ m.s^{-1}) leads to different

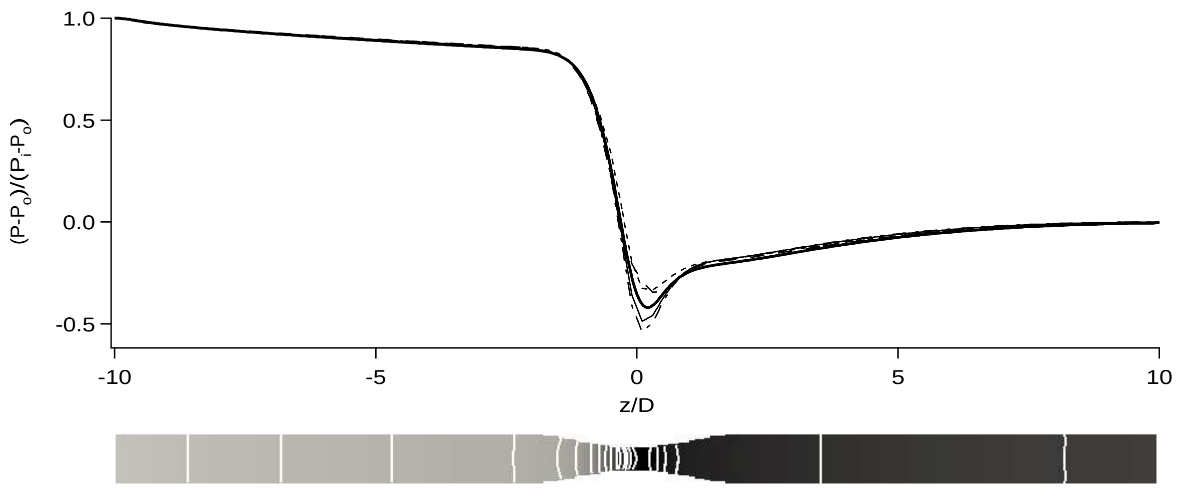

Figure 6: Pressure in the IBM calculations. Top: comparison of non-dimensionalized pressure over the centerline for simulations S1 (- - - -), S2 (— —), S3 (— - —), S4 (——) and S6 (━━). Bottom: pressure field in simulation S4, over the symmetry plane. Scale is from black (low values) to light grey (high values). White pressure isolines allow to show the smoothness of the pressure field.

volume flow rates. Moreover, the volume flow rate varies locally in simulations (as shown in § 3.2.2). For comparisons of the different simulations, velocity profiles have been non-dimensionalized by the local bulk velocity in each calculation (local volume flow rate divided by the real cross-section). Figure 7 displays streamwise velocity profiles. It appears that all IBM simulations give reasonable results. The shape of the streamwise velocity profiles is always well reproduced, except at $z/D = -1$ (Fig. 7c), where IBM simulations show an important contraction of the flow. The size of the recirculation zone downstream of the constriction is well predicted. However, methods B, C and D under-predict the intensity of the back flow. This is expected to be corrected by improving the interpolation method in the IBM implementation (Eqs. (22) and (23)).

Figure 8 displays transverse (wall-normal) velocity profiles (x velocity U against the x coordinate). More differences between the IBM and BF simulations are observed. Curiously, methods A and B show non-physical wall-normal velocity at the first two profiles ($z/D = -3$ (a) and $z/D = -2$ (b)). However, values of velocity are small and these errors are not critical. Method A is shown to anticipate the contraction location, due to the gross geometry description. In the constriction region, velocity profiles in all simulations are very similar, method B being the less accurate in general.

Overall, method B (simulation S2) is the one showing the largest differences with the BF results. Methods C and D give similar results. Method D is preferred to method C for the slightly better mass conservation properties shown in the former section. Method A is rather good, though it is expected to suffer more than the other methods of low resolution.

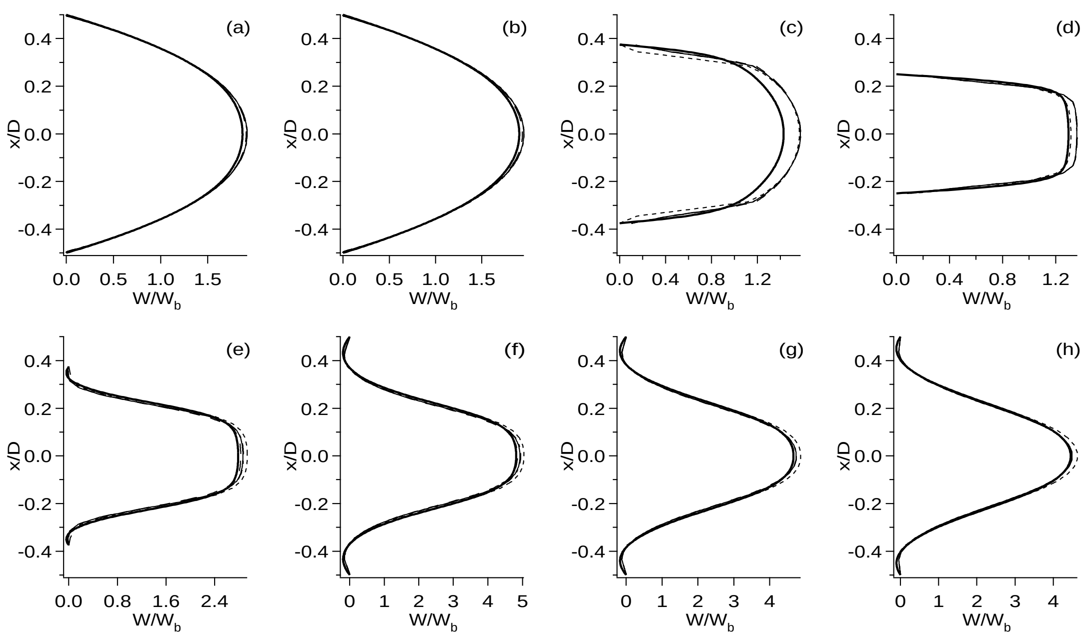

Figure 7: Comparison of non-dimensional streamwise velocity W profiles for simulations S1 (- - - -), S2 (— —), S3 (— - —), S4 (——) and S6 (▬▬). W is non-dimensionalized by the local bulk velocity W_b in each simulation. Profiles extracted on the (x,z) plane, for $y = 0$, at $z = -3\,D$ (a), $z = -2\,D$ (b), $z = -D$ (c), $z = 0$ (d), $z = D$ (e), $z = 2\,D$ (f), $z = 3\,D$ (g) and $z = 4\,D$ (h).

4 IN VITRO MODEL OF AORTIC CROSS

The aim of this section is to provide a preliminary comparison in a more realistic case. The in vitro model of an aortic cross was used in the University Hospital of Toulouse Rangueil (France), to make detailed measurements using Magnetic Resonance Imaging. These measurements were done with a moving geometry (see Nicoud *et al.*[10]). Here, as a first step, the geometry of the aorta is fixed and corresponds to one phase of the moving geometry.

4.1 Presentation of the computations

The geometry of the aorta model is presented in Fig. 9. Compared to a real aorta geometry, the ascending aorta has been lengthened and severely bent. The left subclavian artery is not represented. The boundary conditions (BC) are denoted by numbers in Fig. 9: BC 1 is the main inflow condition, the inlet of the ascending aorta. BCs 2 and 3 represent the brachiocephalic trunk and the common carotid artery, respectively. BC 4 is an outlet condition on the descending aorta. BC 5 is a rigid non-slip wall boundary condition.

Note that the extra pipe section at the inlet of the computational domain (near BC 1 in Fig. 9) is part of the in vitro model. It was designed in such a way to promote the swirl motion of the fluid flow at the inlet of the physiological part of the phantom in order to

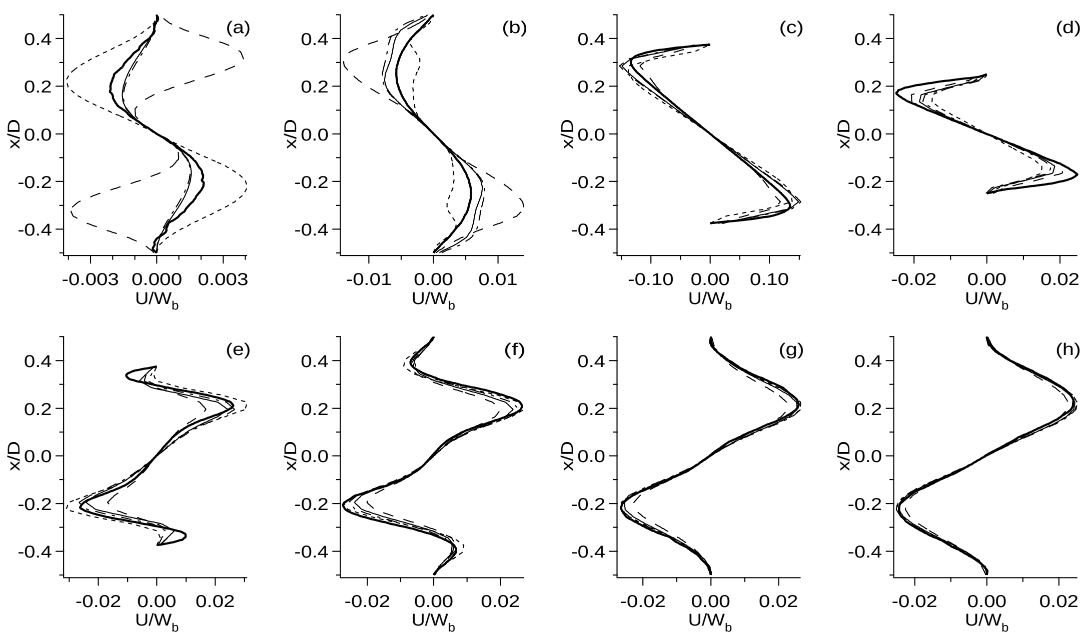

Figure 8: Comparison of non-dimensional transverse velocity U profiles for simulations S1 (- - - -), S2 (— —), S3 (— - —), S4 (——) and S6 (▬). U is non-dimensionalized by the local bulk velocity W_b in each simulation. Profiles extracted on the (x,z) plane, for $y = 0$, at $z = -3\,D$ (a), $z = -2\,D$ (b), $z = -D$ (c), $z = 0$ (d), $z = D$ (e), $z = 2\,D$ (f), $z = 3\,D$ (g) and $z = 4\,D$ (h).

reproduce the in vivo hemodynamic conditions more closely. The simulations began from an initially quiescent flow state and continued for a number of full cardiac cycles in order to allow the development of a fully periodic flow, representative of a regular heartbeat. It was found that the main features of the vascular flow field became periodic within four cycles.

In the calculation, BC 1 is mainly an inflow condition while BCs 2, 3 and 4 are outflow conditions: mass goes out of the domain through 2, 3 and 4. In terms of numerics, the flow is prescribed at BCs 1, 2 and 3 and let free at BC 4. Zero velocity is imposed on the rigid wall (BC 5). In both calculations, the flow rate Q_j is the quantity imposed at BCs j =1, 2 and 3, through a Fourier series in time t: $Q_j = a_{0,j} + \sum_{i=1,5}(a_{i,j}cos(i\omega t) + b_{i,j}sin(i\omega t))$. ω is a pulsation, prescribed at ω = 6.3263 s^{-1}. The cardiac cycle lasts 1 s. $a_{i,j}$ and $b_{i,j}$ coefficients are given in Table 2.

BC	$a_{0,j}$	$a_{1,j}$	$a_{2,j}$	$a_{3,j}$	$a_{4,j}$	$a_{5,j}$	$b_{1,j}$	$b_{2,j}$	$b_{3,j}$	$b_{4,j}$	$b_{5,j}$
j=1	40.15	16.94	-22.59	1.05	-2.15	-1.61	28.50	-0.21	1.56	2.33	-0.72
j=2	-4.23	1.58	2.64	-0.48	-1.27	0.03	2.41	-0.32	-1.67	1.17	0.20
j=3	-1.70	0.63	1.06	-0.19	-0.51	0.01	0.97	-0.13	-0.67	0.47	0.08

Table 2: Coefficients of the time Fourier series defining the volume flow rate at BCs 1, 2 and 3. Values are scaled by 10^{-6} m^3.s^{-1})

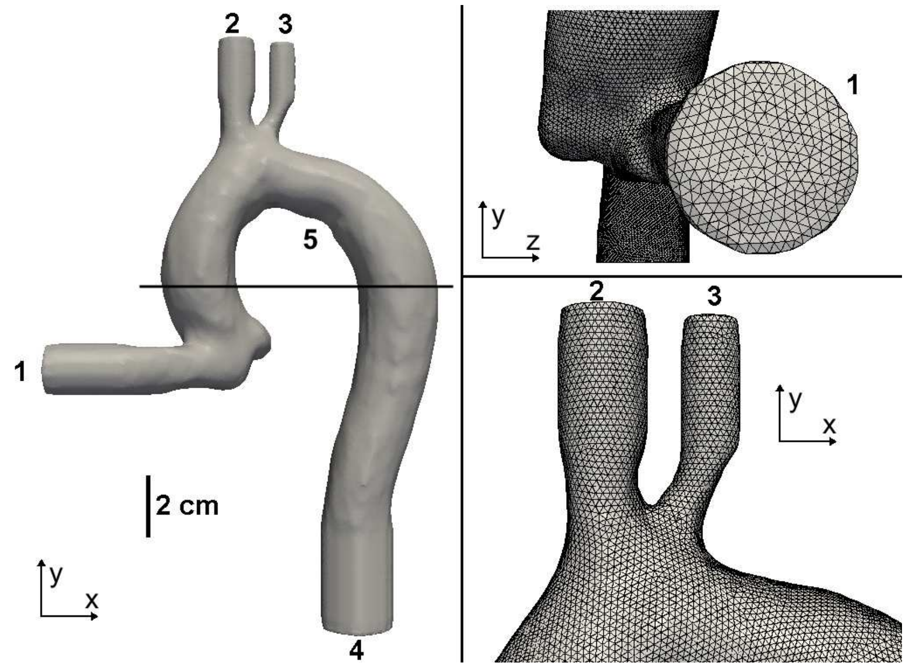

Figure 9: Presentation of the aortic cross model geometry. Left view: computational domain. Right views: presentation of the skin mesh of the BF computation. The scale is shown by the black bar of 2 cm. The horizontal black line cutting the left figure shows the location of the cutting plane used later for flow comparisons in Fig. 12.

Figure 10 presents the volume flow rates imposed at BCs 1, 2 and 3. As the wall is rigid, the flow rate at BC 4 exactly compensates the flow rates at the other BCs. Note that BCs 2 and 3 are always outflow conditions, while BC 1 is an inflow condition most of the time, but experiences negative flow rates during a short period of the cycle (0.5 s $\leq t \leq$ 0.575s).

The grids used in the IBM and BF solvers share the same characteristic discretization size of 1 mm. Figure 9 displays views of the skin mesh in the BF computation: mesh size is sufficient to have a reasonable description of the geometry. The BF grid contains 1,195,791 tetrahedral cells. For the calculation presented in this section method C has been used.

4.2 Results

Numerical results are compared in a sagittal plane cutting through BC 3. The velocity magnitude is displayed in Fig 11. Two instantaneous fields are shown for each simulation: one during systole, at $t_s = 0.3$ s (Fig 11a,b) and one during diastole, at $t_d = 0.5$ s (Fig 11c,d). The velocity magnitude fields are very similar between the BF and the IBM simulations. The global flow structure is well reproduced, even if differences are seen in the details. Some crucial features of the flow are well reproduced by the IBM solver,

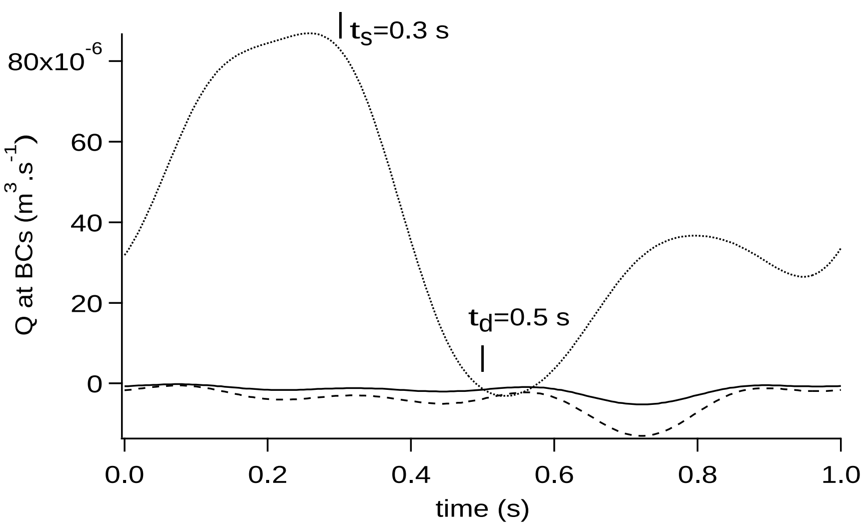

Figure 10: Evolution of volume flow rates Q at BC 1 (⋯⋯), BC 2 (– – – –) and BC 3 (——) during the cardiac cycle of 1 s. Systole (t_s) and diastole (t_d) instants, used in the remainder of the article, are defined in the figure.

notably the detached flow (low-velocity regions) near the internal wall, where curvature is high. Note that the most striking difference between both simulations lies in the small eddies visible in the BF solver. This is due to the difference in the order of the methods: the IBM solver uses a second-order scheme in space, while the BF solver uses a fourth-order scheme. For a fairer comparison, simulations to come with the BF method in future studies will be performed with a second-order scheme.

For more precise comparisons between the two solvers, Fig. 12 allows visualization of the flow in two cross-sections of the aorta model. A horizontal plane is defined (see Fig. 9). It cuts the aorta model at two locations, one in the ascending aorta and the other in the descending aorta. The velocity component in the direction normal to the plane is displayed at the same time instants as for the sagittal plane: $t_s = 0.3$ s (Fig. 12a) and $t_d = 0.5$ s (Fig. 12b).

Scale is the same in BF and IBM results but differ for the systole and diastole instants. Conclusions are similar with what was drawn from the sagittal plane comparisons. The overall structure is reproduced, even if the agreement is not perfect. Again the reason for these difference in the details have yet to be established, though the order of the numerical method is expected to have a major role. In such a dynamical flow, showing strong variations in space and time, the impact of the order of the numerical method can be very important. Recall that the cell size is similar in both calculations.

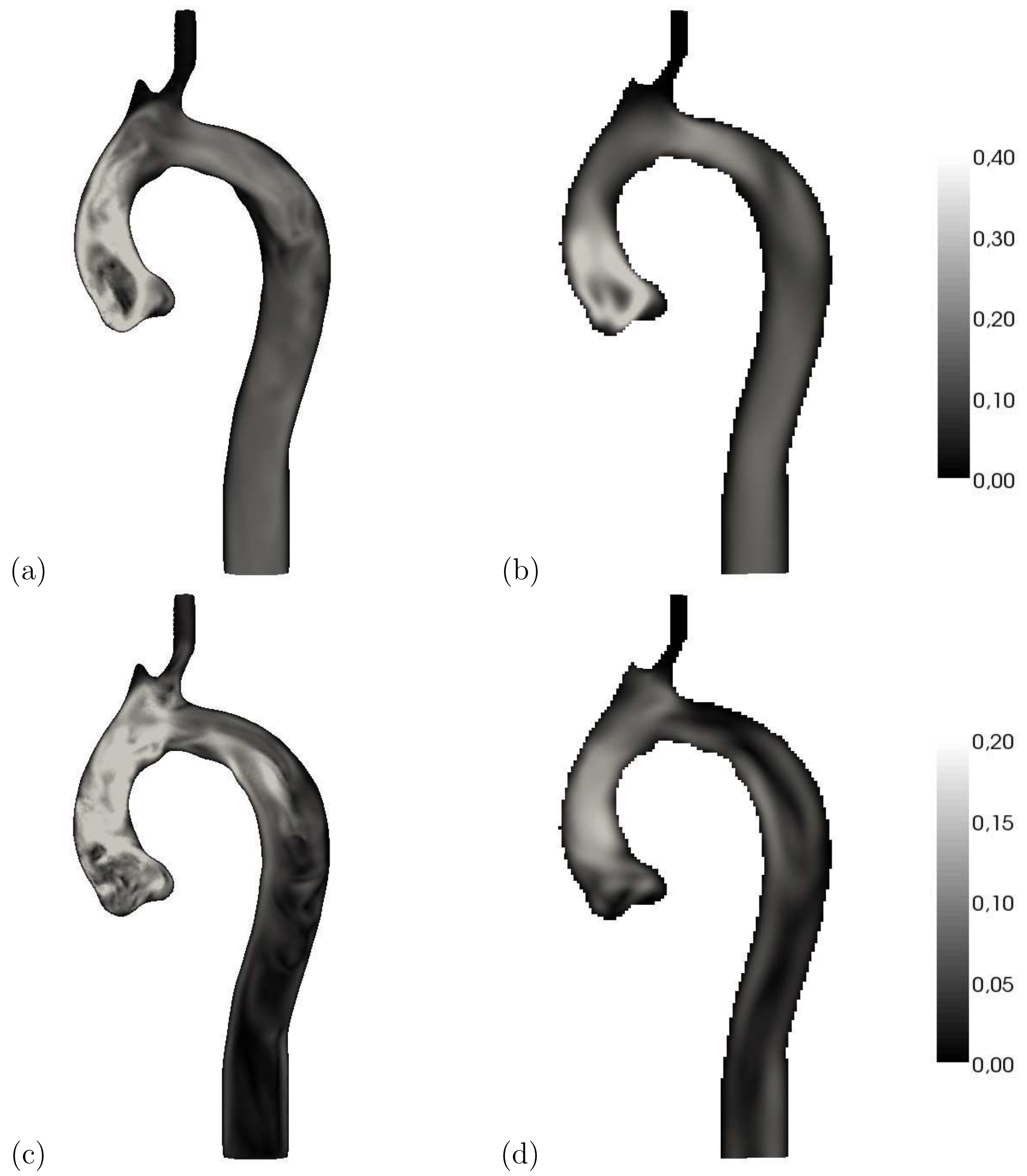

Figure 11: Velocity magnitude over a sagittal plane during systole at time $t_s = 0.3$ s (a,b) and during systole at time $t_d = 0.5$ s (c,d). Comparison of BF (a,c) and IBM (b,d) results.

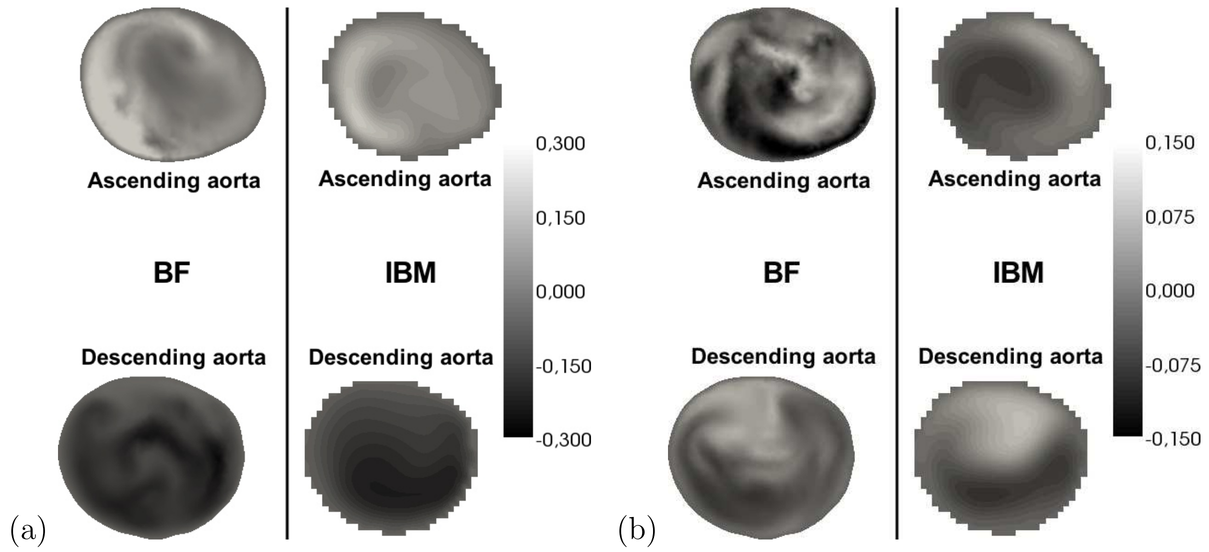

Figure 12: Comparison of BF and IBM results on vertical velocity (normal to the cutting planes) over the cross-section located on the horizontal plane shown in Fig. 9 during systole (a) and diastole (b).

5 CONCLUSION

Here we use the IBM to simulate blood flow into idealized human vessels. We developed new forcing methods for reconstructing the velocity value close to the vessel wall and used a homogeneous Neumann boundary condition for pressure at this location. In section 2.2 the methods were described and formulated for a staggered grid.

In section 3 the case of a stenotic flow was considered to assess the IBM properties. Results obtained using the different IBM implementations were compared with those obtained using a reference BF method. It was shown that, overall, the different IBM are capable to achieve mass conservation with reasonable accuracy. Also, flow solution obtained using the IBM and the BF are quite similar.

Preliminar results of the simulation of a physiological pulsatile flow through a aortic cross model were shown in section 4. Comparisons between the IBM and the BF showed that, again, the general agreement between these methods is satisfactory, altough a more detailed analysis is needed.

Results obtained in this study support the capability of the IBM to cope with complex and physiologically relevant flow whithout the grid generation tasks needed. This demonstrate the potential of the IBM to perform clinical applications of blood flow simulation.

6 ACKNOWLEDGEMENTS

This work was conducted in the framework of the OCFIA research program funded by the Agence Nationale de la Recherche (ANR-07-CIS7-006-02). This work was granted access to the HPC resources of CINES (Centre Informatique National de l'Enseignement

Supérieur) under the allocations c2010035139 and x2010036363 made by GENCI (Grand Equipement National de Calcul Intensif). Dr. V. Moureau is gratefully acknowledged for giving access to the YALES2 solver.

References

[1] E. A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof. Combined Immersed-Boundary Finite-Difference Methods for Three-Dimensional Complex Flow Simulations, *J. Comp. Phys.*, **161**, 35-60 (2000).

[2] T. Ikeno and T. Kajishima, Finite-difference immersed boundary method consistent with wall conditions for incompressible turbulent flow simulations, *J. Comp. Phys.*, **226**, 1485-1508 (2007).

[3] S. Kang, G. Iaccarino, F. Ham and P. Moin, Prediction of wall-pressure fluctuation in turbulent flows with an immersed boundary method, *J. Comp. Phys.*, **228**, 6753-6772 (2009).

[4] R. Mittal and G. Iaccarino, Immersed boundary methods, *Annu. Rev. Fluid Mech.*, **37**, 239-261 (2005).

[5] R. Moreno, M. Chau, S. Jeetoo, F. Nicoud, F. Viart, A. Salvayre, H. Rousseau, Optimized Computational Functional Imaging for Arteries, In proceedings of the *8th International Meeting on High Performance Computing for Computational Science*, Toulouse, France (2008).

[6] V. Moureau and O. Desjardins, A second-order Ghost-Fluid Method for the primary atomization of liquid fuel in air-blast type injectors, In proceedings of the *Summer Program 2008*, Center For Turbulence Research, Stanford University(2008).

[7] M. C. Rivara, Mesh refinement processes based on the generalized bisection of simplices, *SIAM Journal of Numerical Analysis*, **31**, 604-613 (1984).

[8] F. H. Harlow and J. E. Welch, Numerical calculation of the time-dependent viscous incompressible flow of fluid with free surface, *Physics of Fluids*, **8**, 2182-2189 (1965).

[9] D. I. Hollnagel, P. E. Summers, D. Poulikakos, S. S. Kollias, Comparative velocity investigations in cerebral arteries and aneurysms: 3D phase-contrast MR angiography, laser Doppler velocimetry and computational fluid dynamics, *NMR Biomed*, **22**, 795-808 (2009).

[10] F. Nicoud, R. Moreno, B. Tayllamin, M. Chau, H. Rousseau, Computational hemodynamics in moving geometries whithout solving the fluid-structure interaction problem, *Conference on Modelling Fluid Flow*, Hungary (2009).