Cluster Comput (2014) 17:231–241
DOI 10.1007/s10586-013-0323-2

A general purpose parallel block structured open source incompressible flow solver

Mariana Mendina · Martin Draper · Ana Paula Kelm Soares · Gabriel Narancio · Gabriel Usera

Received: 1 March 2013 / Accepted: 26 September 2013 / Published online: 27 November 2013

Abstract A general purpose incompressible flow solver, called caffa3d.MBRi, is presented which features a block structured framework to accommodate both a flexible approach to geometry representation and a straightforward implementation of parallel capabilities through the MPI library. Representation of complex geometries can be handled semi automatically through a combination of body fitted blocks of grids and the immersed boundary condition strategy over both Cartesian and body fitted grid blocks. The parallelization strategy is based on the same block structured framework, by means of encapsulated MPI calls performing a set of conceptually defined high level communication tasks. A set of real world applications ranging from bioengineering to microclimate scenarios is presented to demonstrate the capabilities of the solver, which is open source and freely available through the web page.

Keywords Finite Volume · Immersed Boundary · Blocks Structured · MPI

M. Mendina · M. Draper · G. Narancio · G. Usera (✉)
IMFIA, Udelar, J. Herrera y Reissig 565, 11300 Montevideo, Uruguay
e-mail: gusera@fing.edu.uy

M. Mendina
e-mail: mmendina@fing.edu.uy

M. Draper
e-mail: mdraper@fing.edu.uy

G. Narancio
e-mail: gnaranci@fing.edu.uy

A.P. Kelm Soares
LEMMA, Universidade Federal do Paraná, Curitiba, Brasil
e-mail: anapaulaks@ufpr.br

1 Introduction

In this work the general purpose incompressible flow solver caffa3d.MBRi is presented and described. The solver aims at providing a useful tool for numerical simulation of real world fluid flow problems that require both geometrical flexibility and parallel computation capabilities to afford tens and hundreds million cells simulations.

Instead of tackling geometrical flexibility through fully unstructured grids, a block structured grid approach is followed combined with the immersed boundary condition method [6] to address even the most complex geometries with little meshing effort and preserving the inherent numerical efficiency of structured grids [3, 7].

The same block structured framework provides the basis for parallelization through domain decomposition under a distributed memory model using the MPI library. A compact set of encapsulated calls to MPI routines provide the required high level communication tasks between processes, or domain regions, each comprised of one or several grid blocks.

Sections 2 through 5 present different aspects of the solver implementation, while in Sect. 6 selected application examples are briefly described to illustrate the solver capabilities. Final conclusions are drawn in Sect. 7.

It has been suggested by other authors that fluid dynamics research would benefit from the availability of more public and open source codes [18]. The authors agree with this view and welcome the use and further development of caffa3d.MBRi model by other researchers. For this purpose the code is freely available through the website.[1]

Several comprehensive open source flow solver packages have been developed in the last decade, like OpenFoam,

[1] www.fing.edu.uy/imfia/caffa3d.MB.

Gerris and Palabos among others, each with its own rich set of features. While sharing various features with some of these packages, we believe that caffa3d.MBRi solver has a few distinct characteristics. It evolved from a family of 2D flow solvers contained in the classical text book by Ferziger and Peric [1] sharing many characteristics with them, thus making it appealing for young scientists looking for a base code to master and further build onto it their own developments. Moreover it is coded in Fortran 90 and thus requires less programming skills to master than many C based open source packages, making it especially appealing to scientists less familiar with computer science. Finally, it incorporates some specific capabilities related to micrometeorological modelling, as for example atmospheric radiation models, not commonly found in other packages.

This article is an extended version of "A general purpose parallel block structured open source flow solver" presented by the authors at the "1st International Workshop on Soft Computing Techniques in Cluster and Grid Computing Systems SCCG 2012" held in conjunction with the "Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2012)". The solver is described in greater detail in the present work and its parallel performance is further analyzed.

2 Mathematical model

The mathematical model comprises the mass balance equation (1) and momentum balance equation (2) for a viscous incompressible fluid, together with generic non-reacting scalar transport equation (3) for scalar field ϕ with diffusion coefficient Γ. Note that (2) has been written only for the first Cartesian direction here.

$$\int_S (\boldsymbol{v}\cdot\hat{\boldsymbol{n}}_S)dS = 0 \tag{1}$$

$$\begin{aligned}\int_\Omega \rho\frac{\partial u}{\partial t}d\Omega + \int_S \rho u(\boldsymbol{v}\cdot\hat{\boldsymbol{n}}_S)dS \\ = \int_\Omega \rho\beta(T-T_{ref})\boldsymbol{g}\cdot\hat{\boldsymbol{e}}_1 d\Omega + \int_S -p\hat{\boldsymbol{n}}_S\cdot\hat{\boldsymbol{e}}_1 dS \\ + \int_S (2\mu\boldsymbol{D}\cdot\hat{\boldsymbol{n}}_S)\cdot\hat{\boldsymbol{e}}_1 dS\end{aligned} \tag{2}$$

$$\int_\Omega \rho\frac{\partial\phi}{\partial t}d\Omega + \int_S \rho\phi(\boldsymbol{v}\cdot\hat{\boldsymbol{n}}_S)dS = \int_S \Gamma(\nabla\phi\cdot\hat{\boldsymbol{n}}_S)dS \tag{3}$$

In these equations, $\boldsymbol{v} = (u, v, w)$ is the fluid velocity, ρ is the density, β is the thermal expansion factor, T the fluid temperature and T_{ref} a reference temperature, g is the gravity, p the pressure, μ the dynamic viscosity of the fluid and $\boldsymbol{D}$ the strain tensor, defined in (4). The balance equations are written for a region Ω, limited by a closed surface S, with unit outward pointing normal $\hat{\boldsymbol{n}}_S$. Finally $\hat{\boldsymbol{e}}_1$ is the first Cartesian direction.

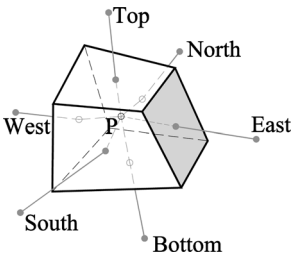

Fig. 1 Single non regular hexahedra with six neighbours

$$\boldsymbol{D} = \frac{1}{2}\left(\nabla\boldsymbol{v} + \nabla\boldsymbol{v}^T\right) \tag{4}$$

The generic transport equation (3) for non-reacting scalars can be used to implement in a straightforward manner further physical models like heat transport required for the temperature field, both Reynolds Averaged and Large Eddy Simulation turbulence closures, wet air processes which include evaporation and condensation, etc. An arbitrary number of scalar fields can be solved simultaneously, with coupling between them as for the case of temperature field influencing both momentum equations through buoyancy and wet air process equations through condensation and evaporation conditions. The use of equations in their global balance form together with the finite volume method, as opposed to the differential form, favors enforcing conservation laws for fundamental quantities such as mass and momentum into the solving procedure [1].

3 Geometry representation strategy

The global grid is made up from structured grid blocks, which can be either orthogonal Cartesian grid blocks or curvilinear body fitted grid blocks. Nevertheless geometrical properties are always expressed in a Cartesian coordinate system, as well as flow properties which are expressed in primitive variables in the same Cartesian coordinate system, like velocities for example. To provide greater geometrical flexibility the immersed boundary method [6] can be combined with both Cartesian and body fitted grid blocks.

Each grid block is composed by an arrangement of non-regular hexahedra as the one of Fig. 1. Within each block, the hexahedra have each exactly six neighbors, do not overlap and leave no empty space in between. The approximation of the integral expressions of (1), (2), and (3) can be done efficiently using pre-computed geometrical properties. the volume $\Delta\Omega$ of the element, the coordinates of its baricenter P, and the surface normal vectors $\boldsymbol{S}_c$ and coordinates of baricenter c for each face ($c = w, e, s, n, b, t$); see Fig. 2. The normal surface vector $\boldsymbol{S}_c$ is defined with its norm equal to the surface area of the face. Each volume element face is a quadrilateral, *not necessarily planar*, so its properties are

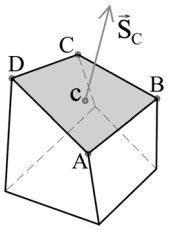

Fig. 2 Volume element with normal surface vector $\boldsymbol{S}_c$ at baricenter c of *top face*

computed from the composition of two triangular elements as ABC or ACD on Fig. 2, through simple algebra.

4 Discretization and solver scheme

Each equation in the mathematical model is discretized and linearized at each cell to obtain a discrete approximation in the form of (5), written again for the u velocity component, where the variable value at cell P is related to the values at the six neighbors.

$$A_P^u \cdot u_P + A_W^u \cdot u_W + A_E^u \cdot u_E + A_S^u \cdot u_S + A_N^u \cdot u_N + A_B^u \cdot u_B + A_T^u \cdot u_T = Q_P^u \tag{5}$$

Complete details for discretization of each term will not be given here but can be found in [17], together with various validations of the solver [16, 17]. As an example, for the convective u flux in (2) using an implicit first order upwind approximation combined with an explicit second order linear interpolation deferred correction in the source term, results:

$$\begin{aligned} F_e^{cu} &= \int_{S_e} \rho u(\boldsymbol{v} \cdot \hat{\boldsymbol{n}}_S) dS \approx \dot{m}_e u_e \\ &= \max(\dot{m}_e, 0) u_P + \min(\dot{m}_e, 0) u_E \\ &\quad + \gamma_{CDS}\big(\dot{m}_e u_e - \max(\dot{m}_e, 0) u_P - \min(\dot{m}_e, 0) u_E\big) \\ &= A_P^{cue} \cdot u_P + A_E^{cue} \cdot u_E - Q_P^{cue} \end{aligned} \tag{6}$$

In (6), F_e^{cu} is the convective u flux through the *east* face of cell element P, $\dot{m}_e$ is the mass flux through the face, and γ_{CDS} is the blending factor between upwind and linear interpolation approximation.

Further, the SIMPLE [10] method for pressure-velocity coupling is used to obtain a discretized equation for the pressure in the form of (5), from the mass balance equation (1). Refined methods for pressure-velocity coupling can also be incorporated [5], together with improved linear interpolations [4]. Also different implicit time stepping schemes can be combined for the momentum equations, like first order backwards Euler or second order Cranck-Nicholson.

The usual lexicographical order in 3D implies that the resulting segregated linear equation system is hepta-diagonal in each grid block, and thus globally block hepta-diagonal. Either a block structured variant of the Stone-SIP solver [7] or a block structured Algebraic Multigrid (AMG) solver with SIP as a smoother [8, 17] are used for iterative solution

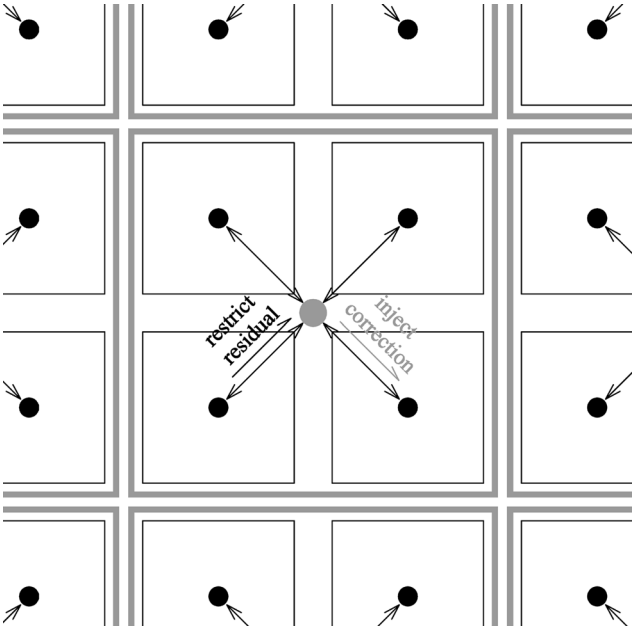

Fig. 3 AMG restriction and injection process at linear system level, shown for the 2D case

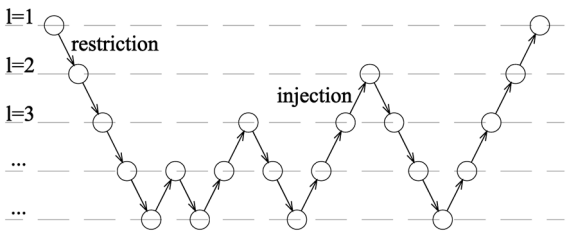

Fig. 4 AMG W cycles involving a sequence of restriction and prolongation operations that effectively reduce error components at coarser levels

of each linear system. The SIP solver algorithm accommodates well the block structure inherited from the grid, allowing efficient parallelization as described below. The AMG solver can profit from the multigrid approach at the linear equation system level by grouping matrix coefficients according to a pattern of $2 \times 2 \times 2$ cells in 3D, and the solution and linear term arrays accordingly, through a restriction process involving increasingly coarser levels. Corrections are then propagated backwards to the finner levels through a sequence of injection or prolongation operations, Figs. 3 and 4 illustrate this process.

To deal with the linearization and subsequent coupling of linear systems for each equation in the mathematical model, an outer-inner iteration scheme for each time step is employed, as shown in Fig. 5. Linear systems for each equation in the mathematical model are assembled and undergo inner iterations with SIP or AMG-SIP linear solvers, one after the other so that the coupling between equations is updated after iterating each equation. The outer loop is re-

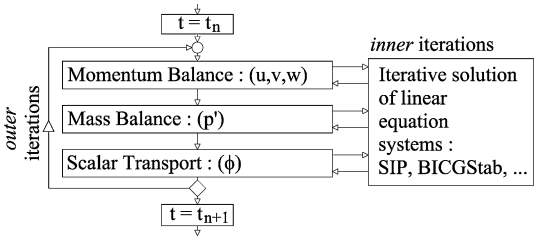

Fig. 5 Iteration scheme for one time step (adapted from [9])

peated within each time step until the desired level of convergence is achieved before continuing to the next time step.

5 Parallelization strategy

Parallelization strategy relies onto the block structured approach for driving domain decomposition and for organizing computations and communications so that they overlap as much as possible to reduce communication overhead. Grid blocks are organized into regions, with one or several grid blocks assigned to each region, which in turn are mapped biunivocally to MPI processes. Distribution of blocks among regions is devised to maximize load balance and minimize communications, and it is currently static and resolved at compile time. However, dynamically adjusting block distributions among regions is also possible and currently under development. While the grid is structured within each grid block, the relationships between grid blocks are of unstructured nature, since each grid block might interface with any number of other grid blocks [7]. Thus, indexed lists are constructed for so called intra region interfaces and inter region interfaces. Intra region interfaces are those between blocks of the same region and require no explicit communication or additional storing locations for halo. Inter region interfaces on the other hand relate cells belonging to blocks assigned to different regions or MPI processes, and thus require explicit communication to take place and also require extra storing locations for halo. To minimize communications, at the expense of some extra halo regions, each field is provided with its own halo. Figure 6 shows a schematic block structured grid made of four grid blocks distributed in two regions. Blocks (1) and (2) belong to one region (black), while blocks (3) and (4) belong to another region (gray). Examples of intra region and inter region block interfaces are identified in the figure.

This scheme enables easy encapsulation of required MPI calls to update halos for the required field, and widespread use of non-blocking MPI communication calls which in turn favor overlapping between communications and computations. These MPI encapsulation routines are organized in a single Fortran90 module within the package, so that no other module in the package requires direct use of MPI calls. For

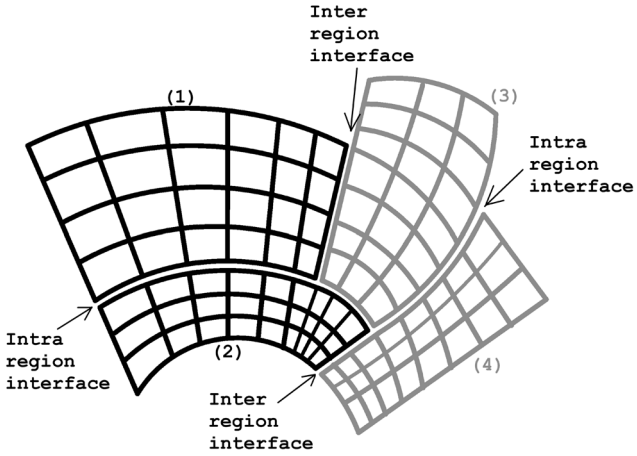

Fig. 6 Schematic representation of a block structured grid made of four blocks (*1*, *2*, *3* and *4*), organized in two regions (*black* and *gray*), with examples of intra region block interfaces and intra region block interfaces identified

example, updating halo regions in all inter region interfaces for the pressure field P can be achieved with a single call to the SwapFiMPI routine with later calls to the TestCommunicationsMPI routine for testing completion of the non-blocking MPI calls:

```
...
CALL SwapFiMPI(P,MpiRequestSend,MpiRequestRecv)
...
...
CALL TestCommunicationsMPI(MpiRequestSend)
CALL TestCommunicationsMPI(MpiRequestRecv)
...
```

The same call would be appropriate to update a three component vector field like the velocity field or even a nine component tensor field like the turbulence stress tensor for a turbulence model, hiding the detailed sintaxis required for standard MPI calls.

The most time consuming operations in the code, both to assemble the equation coefficients or to iterate them with the linear solver, comprise either cell centered operations or cell face operations. The former require no information other than that locally available within each grid block, and thus involve no communications. Cell face operations on the other hand can be classified in two groups. The first group comprises all inner cell faces in each grid block and also those cell faces involved in intra region interfaces. For this group no communications are required either. The second group of cell face operations involves cell faces that correspond to inter region interfaces. In this case explicit communications do need to take place, generally before and after the corresponding computations. Carefully interleaving inter region interfaces communications and computations with other cell face operations and all cell centered operations, allows to effectively minimize the impact of communication

overhead. It is noted that for an N^3 grid block there can be at most $6N^2$ inter region cell face interfaces, while there are N^3 inner cell faces operations required plus N^3 cell centered operations. Thus, as N increases there is an increasingly larger opportunity for inter region related communications to overlap with computations. Preliminary speed up tests, mainly in x86 architectures, in fact show that memory bandwidth for processes sharing the same machine provides a more serious bottleneck than communication overhead between processes allocated in different machines, even when run on relatively slow Gigabit networks. Significant performance gains in fact were observed for a given number of processes and cores, distributing processes among different nodes rather than concentrating processes in fewer nodes.

6 Application examples and results

Selected application cases will be presented next, demonstrating some of the solver features. These cases have been selected to cover a wide range of application contexts, from biological scale flows up to micro meteorological scales, as well as a range of parallelization levels, from a few cores up to 68 cores.

6.1 Wind flow over topography

The first case corresponds to that of wind flow over topography for which the well known case of the Askervein Hill Project [14, 15] has been considered. This measurement campaign took place at an elliptically shaped hill, 120 m high and about 1 km wide across the minor axis. A block structured curvilinear grid was step up, fitting the topography of the hill within a square domain 6 km in side. The grid was split into 16 grid blocks with $48 \times 48 \times 96$ cells each, for the sake of parallelization, with uniform horizontal resolution of about 30 m, and vertical resolution near ground of 0.2 m to resolve the boundary layer. Overall the grid comprised about 3.5 million cells. A simple Smagorinsky large eddy turbulence model was attached to the solver. Figure 7 presents the results for the wind speed-up across de hill profile, comparing current results with those from [11] and from the measurements [14, 15]. A fairly good agreement with measurements and previous simulations is seen in Fig. 7, which is encouraging considering the relatively coarse mesh used and the simple turbulence closure applied. Performance test results for this case are presented in Table 1, for 2, 4, 8 and 16 cores, distributed in up to two nodes. Tests were performed in dual processor servers, equipped with two Xeon E5430 processors each, 8 GB RAM per node, and connected through a Gigabit network. The servers were dedicated to these tests, which means that cores not used in the test remained unloaded. Results show that cores distribution among nodes has a strong effect on performance,

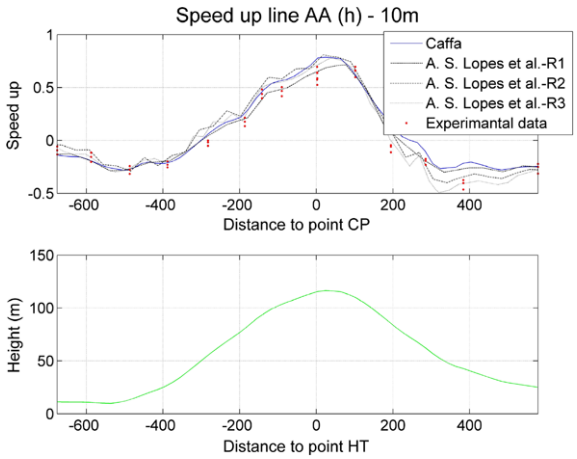

Fig. 7 Speed up (*top*) and hill profile (*bottom*) across Askervein Hill Project domain. Speed up reported at 10 m comparing results from current simulations, those from reference [11] and measurements [14, 15]

Table 1 Performance test results for Askervein Hill case. Number of cores and distribution among servers are reported in the first three columns

Nodes	Cores per node	Total cores	Elapsed time (s)
1	2	2	16620
1	4	4	10154
1	8	8	8188
2	4	8	6085
2	8	16	4505

probably due to a memory bandwidth bottleneck within each node. Distributing 8 cores among two servers resulted in a performance increase of about 1.35X with respect to the case in which the 8 cores were assigned to the same server. Thus, for the present case and available hardware, inter node communication seems to be less of a limiting factor than local memory access bandwidth. Further scaling results are presented in Sect. 6.4.

6.2 Wind flow through built environments

The next application incorporates the representation of buildings geometries into the flow domain by means of immersed boundary method, eventually on top of the topography already captured by the body fitted grid. A square domain, 0.9 km^2 in area, is selected corresponding to a coastline region of Montevideo city, as depicted in Fig. 8. This site was chosen to include the city thermal power plant and to study the dispersion of its emissions. Atmospheric conditions leading to maximum pollutant concentration at floor level have been previously identified to be associated to light winds, about 3.0 m/s velocity at 30 m height. Figure 9 shows a side view of the domain with the topography and the building outlines incorporated by the immersed

Fig. 8 Coastline region of Montevideo city selected for the simulation covering approximately 0.9 km^2

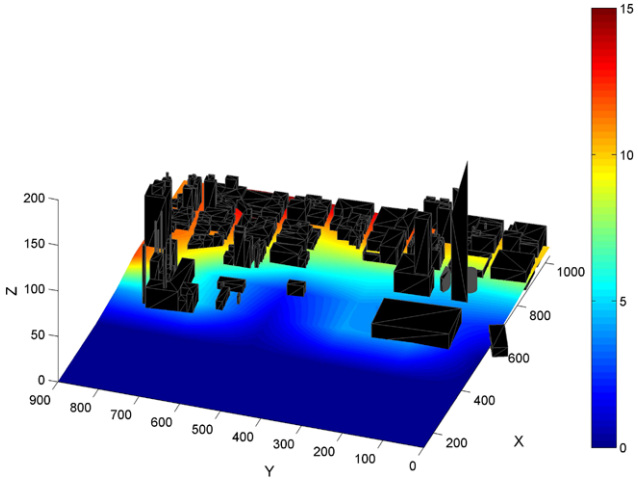

Fig. 9 Side view of simulation domain with topographic height shown in color and building outlines incorporated by immersed boundary method

boundary method. The domain of interest was covered in this case with an array of 64 grid blocks of $96 \times 96 \times 96$ cells each, with an uniform horizontal resolution of about 1 m. Another four blocks were setup with periodic boundary conditions to produce a developed boundary layer, the output of which was then input into the main domain at each time step, Fig. 10. The outflow from this subdomain is input in to the main domain. The overall grid comprises about 60 million cells and uses 68 cores. Flow streamlines are shown in Fig. 11 evidence of the effect of both the topography and the buildings. Also the plume from pollutants emitted at the city's thermal power plant is shown in gray shade. A contours plot for pollutant concentration at the ground level is given in Fig. 12. The effect of building wakes in trapping the pollutants towards the floor is evident. In greater detail in Fig. 13, from a related simulation, streamlines show the occurrence of wakes behind each individual building. The occurrence of a horseshoe vortex in front of the tallest building can be spotted from the streamline pattern. These simulations can be profited from, for example, for urban comfort studies and for analyzing the dispersion of urban pollutants, together with wind tunnel studies.

6.3 Blood flow within a patient specific cerebral aneurysm

Numerical simulations of hemodynamics in a patient specific aneurysm geometry, located at a cerebral artery conjunction called the circle of Willis, were performed under different stenting treatment options. This case corresponds to a recent CFD challenge organized in 2011 by Universitat Pompeu Fabra (UPF), for which most participating research groups used commercial CFD software, while few groups applied open source CFD packages, one of which was caffa3d.MBRi. Figure 14 shows the aneurysm geometry with the specified inlet and outlet flow distributions, for a bulk Reynolds number of $\mathrm{Re} = 50$. An idealized setup was specified for the challenge with steady flow conditions and rigid vessel walls. An immersed boundary approach was used, with the aneurysm geometry laid over an automatically generated block-structured orthogonal mesh with 143 grid blocks and over 40 million cells globally, with a uniform 75 μm resolution. The resulting grid topology is shown in Fig. 15, where the grid blocks are depicted over the aneurysm shape. This procedure leads to an almost automatic meshing strategy for a geometry in which developing a body fitted block structured grid would be seldom feasible. Unstructured grids would be better suited for a body fitted approach, but still would require considerable meshing effort, especially considering the intricate stents geometry.

Five different stenting strategies were tested, using both open mesh (OCS) and closed mesh (CCS) stents, placed at A2-Right, A2-Left or across ACA-A1 arteries. The geometry of the stents was also represented by means of the immersed boundary condition method. The main flow feature is a well-defined jet entering the aneurysm cavity. It's interaction with the stents is shown in Figs. 16 and 17. It is seen in Fig. 17 that the stents configuration affects the intrusion of the jet into the aneurysm cavity, eventually splitting the main jet. Figure 18 shows the streamlines pattern inside the aneurysm cavity, which exhibits strong recirculation.

6.4 Flow at moderate Reynolds number around a sphere sliding over a flat surface

This section describes a water flow simulation around a sphere sliding over a flat surface, which was based on experiments and numerical simulations presented in [12, 13].

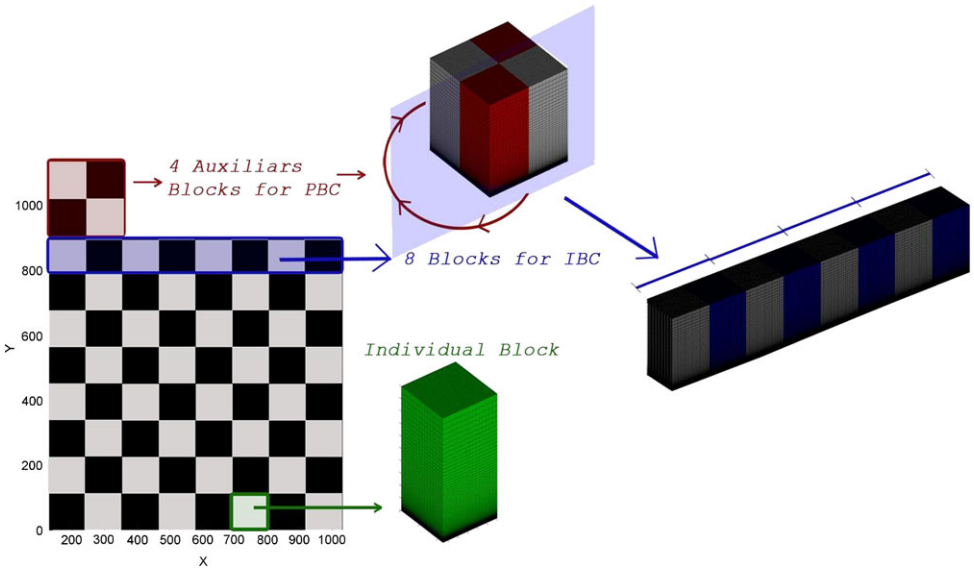

Fig. 10 Schematic representation of the approach to generation of inlet boundary layer conditions. A four block sub-domain is operated with periodic boundary conditions to develop the turbulent boundary layer with instantaneous fluctuations. The outlet from this domain is input at each time through the boundary of the main domain

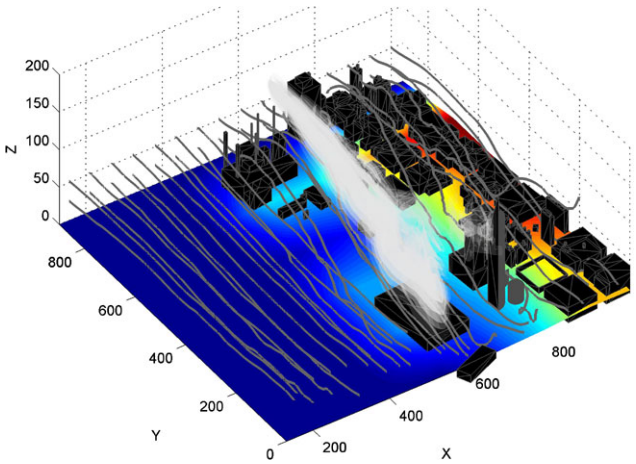

Fig. 11 Streamlines over topography and around building outlines

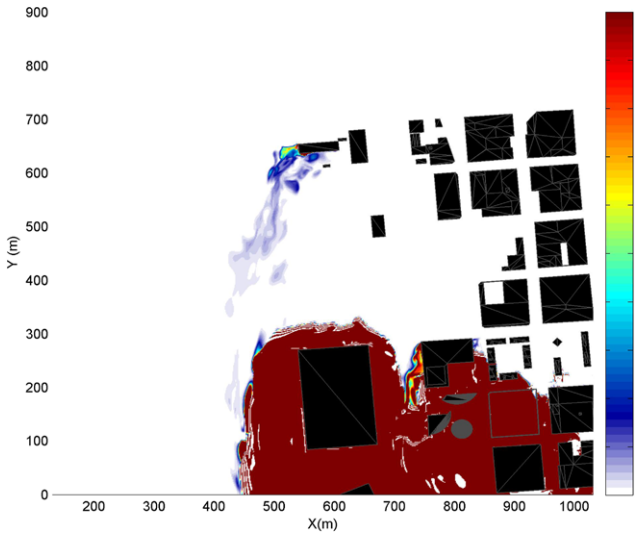

Fig. 12 Contours plot of pollutant concentration at the ground floor, evidence of the effect of building wakes in trapping the pollutants towards the floor

The experiments were carried out in a moving recycling water channel, in which the floor moves with the same velocity as the stream flow. This is analogous to a sphere moving along a fixed wall. The sphere was allowed to be fixed and to rotate forward and backward. The sphere diameter is 9 mm and it is located in the middle of the channel in the XY plane, resting on the floor. The simulations were carried out in a 0.8 m $\times$ 0.12 m $\times$ 0.15 m domain divided in 400 blocks ($25 \times 4 \times 4$). Each block has 32 cells each side, with uniform resolution of approximately 1 mm. The blocks are grouped in 20 regions, thus using 20 cores. The total number of cells is about 13 million, with around 655000 cells per core. Streamlines behind the non-rotating sphere at Re = 50 are given in Fig. 19, while pressure contours for backward rotating sphere are given in Fig. 20. In order to compare the results with [13], the lift and drag coefficients were calculated. The immersed boundary method used provides a straightforward and computationally efficient means to calculate both lift and drag forces over the sphere, since the external forcing imposed by the method to adapt the flow to the sphere geometry corresponds to the force exerted by the sphere onto the flow [6]. Tables 2 and 3 compare the results for the non-rotating sphere obtained by caffa3d.MBRi with those obtained numerically in [13] using a spectral element method with a higher resolution (referred here as Steward). Results are in reasonable agreement, considering the coarse mesh used in the current simulations. The difference is higher for the lift coefficient, probably due to the lift force being more sensitive to the details of the flow near the contact region between the sphere and the floor. This region is quite thin and grid resolution near the contact region might not be high enough.

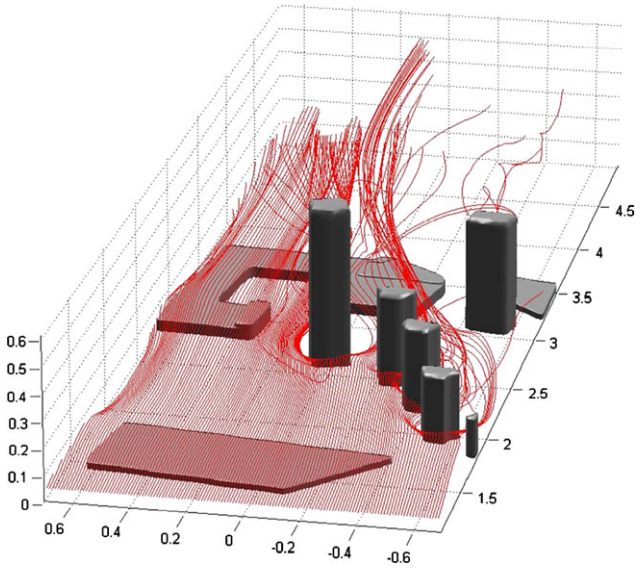

Fig. 13 Flow streamlines around a group of buildings

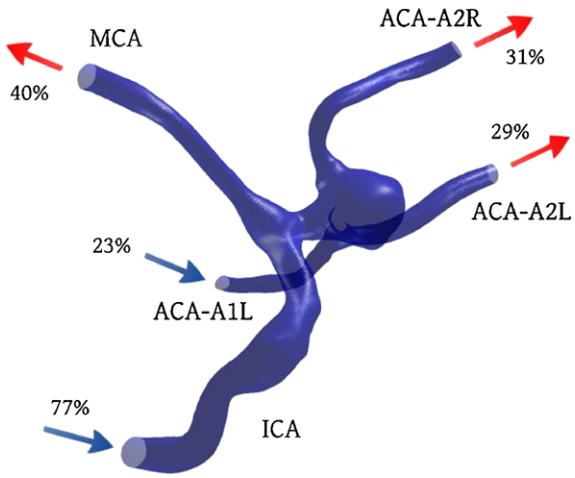

Fig. 14 Patient specific aneurysm geometry with specified flow distribution at inlets and outlets

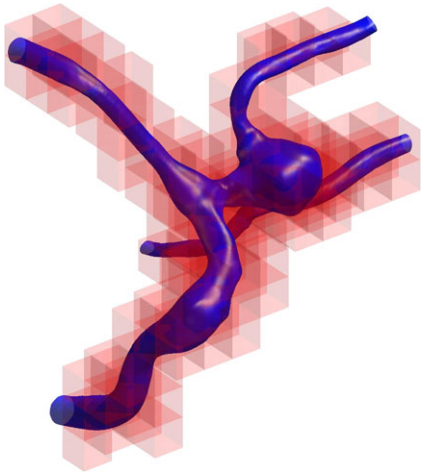

Fig. 15 Block structured grid over laid on the aneurysm domain

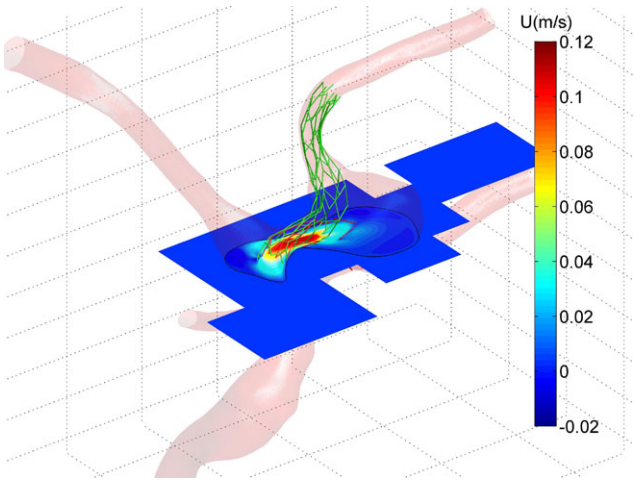

Fig. 16 Detail of flow interaction with stents

Extended strong and weak scaling analysis were performed based on this setup, using up to 128 cores. For the strong scaling analysis, the 400 grid blocks were arranged into 5, 10, 20, 40 and 80 regions with one cpu core assigned to each region. Nodes equipped with two AMD 6172 cpus were used in dedicated mode, thus the remaining cores were left unloaded. For runs with 5, 10 and 20 cores one single server was used, while 2 servers were used for runs with 40 cores and 4 servers were used for runs with 80 cores. Results are presented in Table 4. Note that speed up and efficiency have been computed with reference to the 5 cores setup.

For the weak scaling analysis reduced configurations were setup based on the current case, using increasing number of grid blocks, while keeping the number of blocks assigned to each core fixed at a constant value of four, or about one million cells per core. The RAM requirements per core were also fixed at around 1 GB. Thus, runs were setup with 1, 2, 4, 8, 16, 32, 64 and 128 cores, corresponding to 4, 8, 16, 32, 64, 128, 256 and 512 grid blocks. Results are presented in Table 5.

Strong scaling performance is acceptable for up to 40 cores, with the effect of core distribution among nodes mentioned before being noticeable between runs with 20 and 40 cores. A sharp fall in strong scaling efficiency is observed for the runs with 80 cores. Similarly, weak scaling efficiency decreases at an acceptable rate until about 64 cores, falling more sharply afterwards for the 128 cores runs.

7 Conclusions

The general purpose, open source, flow solver caffa3d.MBRi was presented in this work, with an emphasis in describing

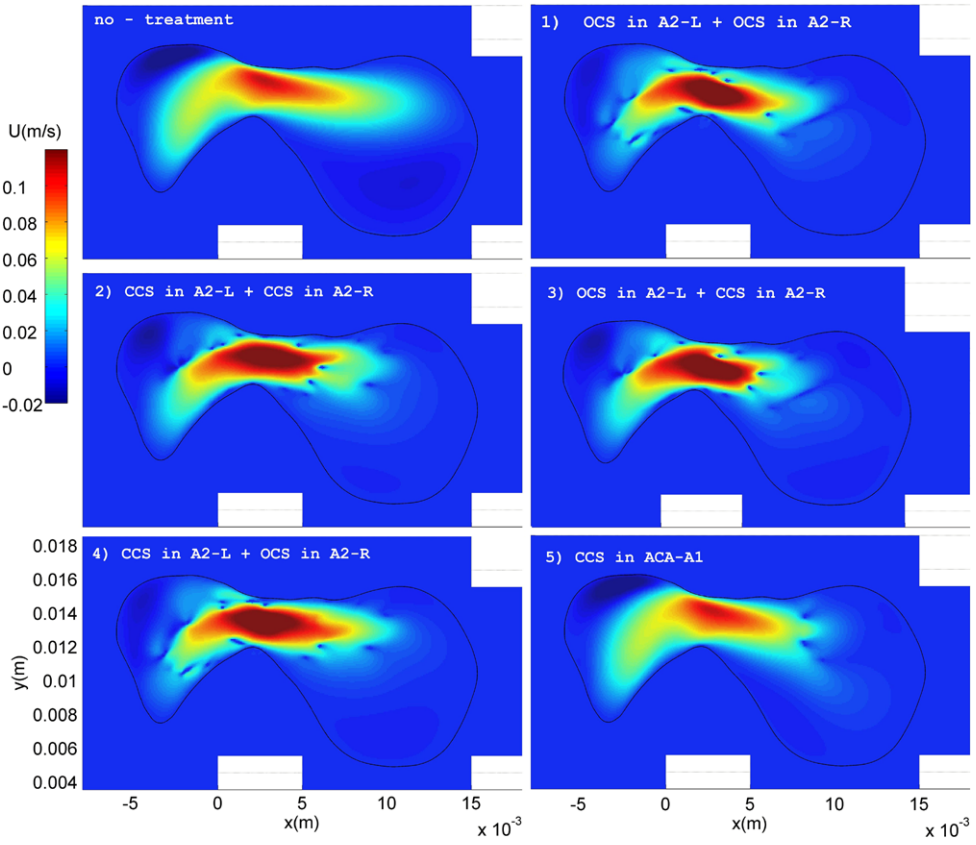

Fig. 17 Flow jet entering the aneurysm cavity with different stenting strategies

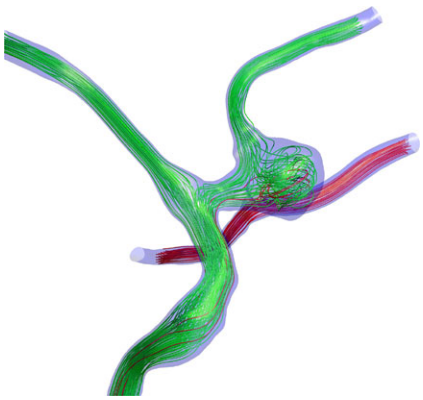

Fig. 18 Streamlines for flow coming from ICA and ACA-A1 arteries. A complex flow pattern with recirculation can be seen inside the aneurysm cavity

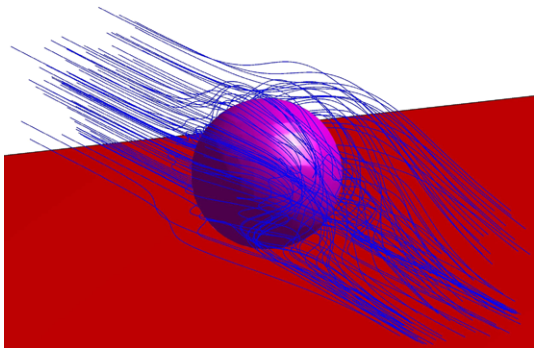

Fig. 19 Streamlines behind non-rotating sphere at Re = 50

how the block structured approach serves well two requirements for real world applications, mainly those of having sufficient geometrical flexibility and enabling parallel computation on modern clusters of multicore nodes for simulations of several million cells. It was shown that, coupled with an immersed boundary method, the block structured approach enables simulation in complex geometries from micro-meteorological scales in urban environments where both topography and building geometry need to be represented, down to bioengineering scales resolving for exam-

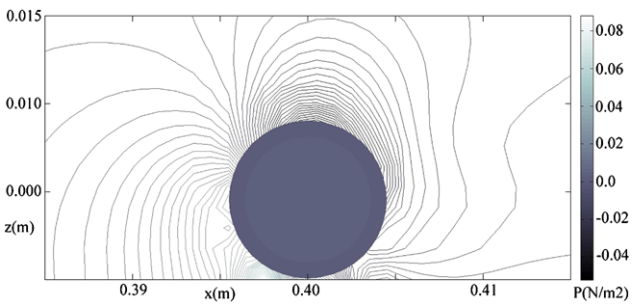

Fig. 20 Pressure contours, Re = 50, backward-rolling sphere

Table 2 Lift coefficient for non-rotating sphere and the relative difference

Re	C_L caffa3d.MBRi	C_L Stewart	Difference (%)
50	0.3650	0.4176	12.6
100	0.2997	0.3626	17.3
150	0.3025	–	–
200	0.3124	0.3297	5.2
250	0.3121	–	–
300	0.3122	0.3022	3.3

Table 3 Drag coefficient for non-rotating sphere and the relative difference

Re	C_D caffa3d.MBRi	C_D Stewart	Difference (%)
50	2.4126	2.4501	1.5
100	1.5671	1.5693	0.1
150	1.2955	1.2523	3.4
200	1.1604	1.1017	5.3
250	1.0655	0.9926	7.3
300	1.0171	0.9252	9.9

Table 4 Strong scaling analysis results

Cores	Nodes	Time (s)	Speed up	Efficiency
5	1	350	1.00	1.00
10	1	200	1.75	0.88
20	1	140	2.50	0.63
40	2	70	5.00	0.63
80	4	61	5.73	0.36

Table 5 Weak scaling analysis results

Cores	Nodes	Time (s)	Efficiency
1	1	221	1.00
2	1	232	0.95
4	1	253	0.87
8	1	319	0.69
16	1	388	0.57
32	2	404	0.55
64	4	498	0.44
128	8	702	0.31

ple patient specific artery geometry together with fine wired stents. It was further shown to be able to cope with sliding surfaces, as in the case of a sphere that slides and rotates along a plane. Current development efforts involve freely moving bodies inside the fluid domain. The same block structured approach enabled dealing with overlapping of computations and communications to reduce the overhead associated with the latter. Simulation cases were presented using up to 128 cores. Systematic ongoing performance analysis are currently being performed, while results reported for up to 64 cores show an acceptable scalability, provided that the local memory bandwidth is not saturated, thus favoring distribution of tasks among different nodes. An experimental partial porting of the code to GPU platforms is also being pursued, with some encouraging results recently presented in [2]. The code is freely available for download through the web page.

References

1. Ferziger, J., Peric, M.: Computational methods for fluid dynamics. Springer, Berlin (2002)
2. Igounet, P., Alfaro, P., Usera, G., Ezzatti, P.: Towards a finite volume model in a many-core platform. Int. J. High Perform. Syst. Archit.
3. Lange, C.F., Schäfer, M., Durst, F.: Local block refinement with a multigrid solver. Int. J. Numer. Methods Fluids **38**, 21–41 (2002)
4. Lehnhauser, T., Schäfer, M.: Improved linear interpolation practice for finite-volume schemes on complex grids. Int. J. Numer. Methods Fluids **38**, 625–645 (2002)
5. Lehnhauser, T., Schäfer, M.: Efficient discretization of pressure-correction equations on non-orthogonal grids. Int. J. Numer. Methods Fluids **42**, 211–231 (2003)
6. Liao, C., Chang, Y., Lin, C., McDonough, J.M.: Simulating flows with moving rigid boundary using immersed-boundary method. Comput. Fluids **39**, 152–167 (2010)
7. Lilek, Z., Muzaferija, S., Peric, M., Seidl, V.: An implicit finite-volume method using nonmatching blocks of structured grid. Numer. Heat Transf. Part B **32**, 385–401 (1997)
8. Mora Acosta, J.: Numerical algorithms for three dimensional computational fluid dynamic problems. Ph.D. Thesis, UPC (2001)
9. Peric, M.: Numerical methods for computing turbulent flows. Course notes (2001)
10. Rhie, C.M., Chow, W.L.: A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation. AIAA J. **21**, 1525–1532 (1983)
11. Silva Lopes, A., Palma, J.M.L.M., Castro, F.A.: Simulation of the Askervein flow. Part 2: Large-eddy simulations. Bound.-Layer Meteorol. **125**, 85–108 (2007)
12. Stewart, B.E., Leweke, T., Hourigan, K., Thompson, M.C.: Wake formation behind a rolling sphere. Phys. Fluids **20**, 071704 (2008)
13. Stewart, B.E., Thompson, M.C., Leweke, T., Hourigan, K.: Numerical and experimental studies of the rolling sphere wake. J. Fluid Mech. **643**, 137–162 (2010)
14. Taylor, P., Teunissen, H.: Askervein '82: report on the September/October 1982 experiment to study boundary layer flow over Askervein, South Uist. Technical Report MSRS-83-8, Meteorological Services Research Branch, Atmospheric Environment Service, Downsview, Ontario, Canada (1983), p. 172
15. Taylor, P., Teunissen, H.: The Askervein Hill Project: report on the September/October 1983, main field experiment. Technical Report MSRS-84-6, Meteorological Services Research Branch, Atmospheric Environment Service, Downsview, Ontario, Canada (1985), p. 300
16. Usera, G., Vernet, A., Ferré, J.A.: Use of time resolved PIV for validating LES/DNS of the turbulent flow within a PCB enclosure model. Flow Turbul. Combust. **77**, 77–95 (2006)
17. Usera, G., Vernet, A., Ferré, J.A.: A parallel block-structured finite volume method for flows in complex geometry with sliding interfaces. Flow Turbul. Combust. **77**, 471–495 (2008)

18. Zaleski, S.: Science and fluid dynamics should have more open sources (2001). http://www.lmm.jussieu.fr/~zaleski/OpenCFD.html

Mariana Mendina received her engineering degree and master degree in applied fluid mechanics from UdelaR. She holds a full time research position since 2001 and lecturer of Fluid Mechanics since 2002 at IMFIA. During this time she has been developing research studies involving different scales within the atmosphere but with emphasis on Planetary Boundary Layer. Her current research interest focus on the development of the CFD model: caffa3d.MBRi, improving the capabilities of this model for working in complex geometries in conjunction with the incorporation of processes that interact with the atmosphere. Her activity is concentrated in the numerical flow simulation in urban environment and dispersion of pollutants into them.

Martin Draper received his engineering degree (2003) and master degree in renewable energies (2009) from Universidad de Montevideo and Universidad de Zaragoza respectively. He has been working as a wind resource analyst since 2009. In addition to this, since 2011 he is doing a PhD in Applied Fluid Mechanics at IMFIA with a scholarship of ANII, focused on CFD in wind energy. His main research interests are the simulation of the wind flow in the atmospheric boundary layer and the interaction between wind turbines through the CFD model caffa3d.MBRi.

Ana Paula Kelm Soares studied at UFPR, where she graduated in Environmental Engineering (2011) and received her Master's degree in Numerical Methods in Engineering (2013). She currently works with environmental modelling, more specifically with water quality and hydrodynamic modelling applied to reservoirs, rivers and estuaries. Her research interests includes environmental and turbulence modelling.

Gabriel Narancio works at IMFIA-UdelaR as Assistant Professor in Fluid Machinery. He received his Mechanical Engineering degree in 2010. Since 2007 he has been working on physical simulations of wind in urban environments in a wind tunnel. Recently he began to work on numerical simulation of the Atmospheric Boundary Layer and its interaction with buildings using the CFD model caffa3d.MBRi.

Gabriel Usera holds a position as Aggregate Professor in Fluid Mechanics at IMFIA-UdelaR. He leads the CFD group there and steers the development of caffa3d.MBRi model. Also runs the nation wide super computing infrastructure Cluster-FING.