Recent Advances in the Particle Finite Element Method Towards More Complex Fluid Flow Applications

Norberto M. Nigro, Juan M. Gimenez and Sergio R. Idelsohn

Abstract This paper presents a state of the art in the Particle Finite Element Method, normally called PFEM, its emphasis in the new ideas oriented to extend its application not only to solve fluid structure interaction and multifluid problems, also bring new opportunities to shorten the gap between engineering design times and computational simulation times for general problems when Eulerian formulation were typically chosen. In order to reduce the long history of this method here the starting point begins with the reformulation of the method to solve academic and real problems in real time or at least in drastically reduced computational times. The main topics involved in this paper are around the stability and the accuracy of Lagrangian formulations against its Eulerian counterpart shown through several academic benchmarks and a deep analysis of the efficiency revealing that the original method needs some new features. The former brought out a new integration method called X-IVAS and the later has produced a new version of the method called PFEM in fixed Mesh. Once the method had shown its good performance and how the new features impact on the final efficiency the last developments had been done in extending the application of this new method in multifluids and other complex fluid mechanics problems like turbulence and reactive flows.

N. M. Nigro (✉) · J. M. Gimenez
Centro de Investigacion en Metodos Computacionales (CIMEC), Consejo Nacional de Investigaciones Cientificas y Tecnicas (CONICET), Santa Fe, Argentina
e-mail: nnigro@intec.unl.edu.ar

N. M. Nigro
Centro de Investigacion en Metodos Computacionales (CIMEC), Universidad Nacional del Litoral (UNL), Santa Fe, Argentina

S. R. Idelsohn
Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE), Institució Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Spain

S. R. Idelsohn
Centro de Investigaciones en Mecánica Computacional (CIMEC), Universidad Nacional del Litoral (UNL), Santa Fe, Argentina

S. R. Idelsohn (ed.), *Numerical Simulations of Coupled Problems in Engineering*, Computational Methods in Applied Sciences 33, DOI: 10.1007/978-3-319-06136-8_12,

1 Introduction

Standard formulations for the solution of the incompressible Navier-Stokes equations may be split in two classes depending on the approach chosen for the description of the inertial terms, namely Eulerian and Lagrangian approaches. In the first class, the acceleration is described as the sum of the spatial derivative of the velocity plus the convective term. In the second approach, the acceleration is simply described as the total derivative of the velocity. Over the last 30 years, computer simulation of incompressible flows has been mainly based on the Eulerian formulation (see [18] for references on this topic). However, with this formulation, it is still difficult to analyze large 3D problems in which the shape of the free-surfaces or internal interfaces changes continuously or in fluid structure interactions where complex contact problems are involved. In all these problems the computing time is sometimes so high that makes the method unpractical. In the last few years, several solutions using the Lagrangian formulation to solve the compressible and incompressible Navier-Stokes equations have been developed [3, 9, 11]. The advantages of these solutions to solve problems with free-surfaces or multi-fluids with complicated internal interfaces have been demonstrated [15]. In general, these formulations are more expensive than a standard Eulerian approaches if they are used in homogeneous flows, but they justify their popularity in solving free- surface flows or complicated multi-fluids flows in which the standard Eulerian formulations are inaccurate or, sometimes, impossible to be used [15].

When attempting to compare the efficiency of Eulerian codes against Lagrangian ones the conclusions were not so definite. Even though Lagrangian solvers are simpler than Eulerian ones the very small time steps normally employed in the former has reduced its application only to specific examples.

Only few attempts in the past thought in using Lagrangian formulation for homogeneous fluid flow. To cite here only a few contributions we can mention the work of Joe Stam [23]. In these work the author solve the Navier-Stokes equations in the context of video games leaving the message that it is possible to design simpler numerical methods that may be applied on this challenge context where the efficiency is the key point.

One of the reasons why there are so few jobs using Lagrangian methods for homogeneous fluid flow applications may be the important computational cost involved in the remeshing. Lagrangian solvers are principally based on moving particles, and after that some sort of mesh should be built depending on the specific method the way to do that. *PFEM* is one of the most popular Lagrangian methods with the particular fact that the moving particles define a mesh where the discrete equations are solved. Its origins go back to the early 2,000. For brevity reasons its state of the art is given up here. Readers interested in the basis of the method may see http://www.cimne.com/pfem/ where there are most of the publications of the method. Among the most cited publications here we can mention [3, 12, 13, 19].

This feature obliged the method to deform the mesh up to certain limit where for geometric reasons some sort of remeshing should be done. As the remeshing was only

limited to some special time intervals the deforming mesh added another ingredient to the time step selection, to avoid the mesh inversion. This severe limitation together with another imposed for the non-linearities and those proper of explicit schemes made the efficiency of original *PFEM* a serious problem. Lately the method evolved thanks to the progress done in parallel mesh generation and remeshing avoiding this serious limitation in some measure.

Even though these limitations and the large community that normally employ Eulerian codes the Lagrangian formulation contains some nice features that need to be reviewed here.

One of the most important rests on the missing of the convective term in the balance equations, converting the non-symmetric equations in symmetric and positive definite. For Navier-Stokes equations this fact has a by-product, converting in linear the original non-linear momentum equation. These two facts avoid the usage of stabilization terms with the strong consequence of not adding the typical numerical diffusion needed to stabilize it. Not having convective terms, for constant coefficient problems as for laminar and homogeneous fluid flow and also for DNS (Direct Numerical Simulation), the system matrices may be factorized at the beginning and reusing them all the time steps, with an important saving in cpu-time. For convection dominated flows the time step in Eulerian formulations needs to be limited attending non-linearities and stability reasons. On the contrary, the Lagrangian formulations do not suffer from this inconvenience when the equations are integrated with good accuracy. This is a key point that deserves much more attention.

In particular *PFEM* has evolved considerably over the past few years, incorporating new ideas seeking enlarge the time steps largely in stable and accurate way.

In this sense *PFEM* has incorporated a novel time integration scheme called X-IVS and its extension X-IVAS. This form of integrating based in following the streamlines of the flow in the present time step is to some extent a better way to solve the non-linearities of the equations of the flow.

In this way it is possible to solve the complex flow situations allowing to extend the time steps in a significant way.

On the other hand being the information carried by the particles using the mesh only for computing secondary fields confers to the method of high accuracy.

Therefore the goals of accuracy, robustness (stability) and efficiency are significantly improved by these new ideas included in the last version of *PFEM*, called Fixed Mesh PFEM.

One of the main target of this work is to show that Lagrangian formulations are not only valuable to solve heterogeneous fluid flows with free-surfaces. We will prove on the contrary that even for homogeneous fluid flows, without free-surfaces or internal interfaces, they are able to yield accurate solutions while being competitively fast when compared to state-of-the-art eulerian solvers. Also, another goal of this paper is to update the state of the art of *PFEM* joining some basis published before [6, 7, 10, 11, 22], with new findings discovering more and more nice features of the method to become a competitive tool in the future for high performance computations.

The paper starts with a mathematical review of the problems to be treated writing them in an Eulerian and a Lagrangian formalism.

Next, the time integration schemes are presented where it is possible to understand the novelty introduced by X-IVS and X-IVAS.

It is followed by a section dedicated to two examples that have served as inspiring muses for the development of new ideas which were then applied to PFEM. In these examples may be understood the benefits of using Lagrangian solvers. While these examples solved by Eulerian codes needs a lot of numerical artifacts, they are trivially solved by Lagrangian ones. The next section presents the two versions of PFEM. The first called Mobile Mesh Version is an extension to the original PFEM with permanent remeshing and a X-IVAS time integration scheme included. Showing the pros and cons the rest of the section is devoted to the novel idea of mixing two view points, one based on particles and the other based on the background and fixed mesh. This idea allowed to increase the efficiency in a very important way. Even though some earlier attempts had been done in using the duality of particle and mesh, for example [8], at the moment of designing the idea this information was not on the knowledge of the authors and moreover, both ideas have only few things in common.

The next section presents some details about the Fixed Mesh version of PFEM, how to manage the particle inventory, how to share the information between particles and mesh. It is followed by a section where the focus is on the treatment of the diffusive terms. Contrary to what may be a prior assumed, the Lagrangian behavior has been superior to the Eulerian one, in regard to precision being that this part of the calculation is of Eulerian nature. The last section is devoted to show some examples solved numerically by PFEM where it is possible to realize that in the present status PFEM is able to solve turbulent flows, multifluids and multiphase flows, general multiphysics, among others. Finally some conclusions are included.

2 X-IVAS: A New Integration Method to Enhance Accuracy and Stability

In this section the emphasis is placed on the main features that produce the big advantages of *PFEM* against any other method. In general the interesting problem to be solved is the general transport equation that is very widespread in the engineering applications. Both, the passive scalar transport equation and the incompressible flow represented by the Navier-Stokes equations will be considered in the rest of the paper.

In order to understand the evolution of *PFEM* the Eulerian and Lagrangian formulations are introduced first.

2.1 Scalar Transport Equation

In the Eulerian framework a fixed coordinate system is considered as the reference for the physical quantities. The scalar transport equation is written as:

$$\frac{\partial T}{\partial t} + \nabla \cdot (\mathbf{v}T) = \nabla \cdot (\alpha \nabla T) + Q \tag{1}$$

where $T(\mathbf{x}, t)$ is the dependent variable (passive scalar), for example the temperature, $\mathbf{v}$ is the velocity vector, for this problem a given data and α the diffusivity, with $\nabla\cdot$ the divergence operator, ∇ the gradient operator and $\frac{\partial}{\partial t}$ the temporal derivative. In this problem $\mathbf{x}$ represents a fixed coordinate.

Normally this equation may be rewritten in the following form:

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T = \nabla \cdot (\alpha \nabla T) + Q - (\nabla \cdot \mathbf{v})T \tag{2}$$

where the first order derivative is split in two terms, one for the convective transport and the other for the source term generated by the non free divergence velocity field. Normally the incompressible flow satisfies the free divergence and in this case this source term may be neglected.

On the other hand in the Lagrangian framework the same problem is written as:

$$\frac{DT}{Dt} = \nabla \cdot (\alpha \nabla T) + Q \tag{3}$$

where $\frac{DT}{Dt} = \frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T$ is the material derivative. The convective term works like a variable transformation between that measured in a fixed coordinate system and that measured in the moving coordinate system that travels with the fluid velocity $\mathbf{v}$. In this transformation the velocity field is incorporated in the dependent variable itself being the unknown variable $T = T(\mathbf{x}_p, t)$ with $\mathbf{x}_p$ the location of a fluid parcel included within the material volume. This location is at the same time another variable, so it is needed to solve an additional equation like:

$$\frac{D\mathbf{x_p}}{Dt} = \mathbf{v} \tag{4}$$

Finally the problem in Lagrangian framework is:

$$\begin{aligned} \frac{DT}{Dt} &= \nabla \cdot (\alpha \nabla T) + Q \\ \frac{D\mathbf{x_p}}{Dt} &= \mathbf{v} \end{aligned} \tag{5}$$

2.2 Incompressible Viscous Fluid Flow: Navier-Stokes Equations

The other problem that in this paper deserves special attention is the fluid dynamics of an incompressible and viscous flow. It is very well known that this problem is governed by the Navier-Stokes equations that presents the balance of the linear momentum equation and the continuity equation or the mass balance.

Both equations normally written together in an Eulerian framework look like:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\mathbf{v}\rho) &= 0 \\ \frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v}) &= \nabla \cdot (\sigma) + \mathbf{F} \end{aligned} \tag{6}$$

Being σ the stress tensor which definition may be split in two parts, one for the spherical (isotropic) component being proportional to the fluid pressure and the other, the deviatoric or viscous component normally written as τ. The operator $\otimes$ means the tensor or dyadic product between two vectors. $\mathbf{F}$ represents the external force, for example the gravity, and finally ρ is the density. For incompressible flows the density is constant, therefore, the continuity equation becomes a constrain over the velocity field, as:

$$\nabla \cdot (\mathbf{v}) = 0 \tag{7}$$

Applying the above restriction also in the linear momentum equation produces a simplified and non-conservative version like

$$\begin{aligned} \nabla \cdot (\mathbf{v}) &= 0 \\ \rho(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v}) &= \nabla \cdot (\sigma) + \mathbf{F} \end{aligned} \tag{8}$$

For the Lagrangian formulation the above system is written as:

$$\begin{aligned} \nabla \cdot (\mathbf{v}) &= 0 \\ \rho \frac{D\mathbf{v}}{Dt} &= \nabla \cdot (\sigma) + \mathbf{F} \\ \frac{D\mathbf{x_p}}{Dt} &= \mathbf{v} \end{aligned} \tag{9}$$

2.3 Time Integration

In this section the time integration of both frameworks, the Eulerian and the Lagrangian is presented.

For simplicity the scalar transport equation is chosen first leaving for some special topics the extension to the vector equation system governing the fluid dynamics of one phase incompressible viscous flow.

2.3.1 Scalar Transport Equation

For the Eulerian framework represented by Eq. (2) the integration is normally done as

$$\begin{aligned}
\int_{t^n}^{t^{n+1}} \frac{\partial T}{\partial t} dt &= \int_{t^n}^{t^{n+1}} (-\mathbf{v} \cdot \nabla T + \nabla \cdot (\alpha \nabla T) + Q) dt \\
T^{n+1}(\mathbf{x}) - T^n(\mathbf{x}) &= \int_{t^n}^{t^{n+1}} (-\mathbf{v} \cdot \nabla T(\mathbf{x}) + \nabla \cdot (\alpha \nabla T(\mathbf{x})) + Q(\mathbf{x}, t)) dt \\
T^{n+1}(\mathbf{x}) - T^n(\mathbf{x}) &= (-\mathbf{v} \cdot \nabla T(\mathbf{x}) + \nabla \cdot (\alpha \nabla T(\mathbf{x})) + Q(\mathbf{x}, t))^{n+\theta} \Delta t
\end{aligned} \tag{10}$$

For some $\theta \in (0, 1)$ the last expression in (10) gives the exact solution. As this parameter is unknown and problem dependent some fixed values for θ are adopted, $\theta = 0$ for explicit schemes, $\theta = 1$ for implicit schemes and $\theta = \frac{1}{2}$ for Crank-Nicholson among others.

$$\begin{aligned}
(-\mathbf{v} \cdot \nabla T + \nabla \cdot (\alpha \nabla T) + Q)^{n+\theta} = \theta \Big(&- \mathbf{v} \cdot \nabla T + \nabla \cdot (\alpha \nabla T) + Q \Big)^{n+1} \\
&+ (1 - \theta) \Big(- \mathbf{v} \cdot \nabla T + \nabla \cdot (\alpha \nabla T) + Q \Big)^{n}
\end{aligned} \tag{11}$$

Replacing (11) in (10)

$$\begin{aligned}
T^{n+1}(\mathbf{x}) - T^n(\mathbf{x}) = \theta \Big(&- \mathbf{v} \cdot \nabla T + \nabla \cdot (\alpha \nabla T) + Q \Big)^{n+1} \Delta t \\
&+ (1 - \theta) \Big(- \mathbf{v} \cdot \nabla T + \nabla \cdot (\alpha \nabla T) + Q \Big)^{n} \Delta t
\end{aligned} \tag{12}$$

The right hand side in (12) is evaluated using only the information of the nodal point $\mathbf{x}$ at the two extremes of the time interval, t^n and $t^{n+1} = t^n + \Delta t$.

For the Lagrangian framework a similar integration scheme is applied.

$$
\begin{aligned}
\int_{t^n}^{t^{n+1}} \frac{DT}{Dt} dt &= \int_{t^n}^{t^{n+1}} (\nabla \cdot (\alpha \nabla T) + Q)^t dt \\
\int_{t^n}^{t^{n+1}} \frac{D\mathbf{x_p}}{Dt} dt &= \int_{t^n}^{t^{n+1}} \mathbf{v}^t dt \\
T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^n, t^n) &= \int_{t^n}^{t^{n+1}} \Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^t dt \\
\mathbf{x_p}^{n+1} - \mathbf{x_p}^n &= \int_{t^n}^{t^{n+1}} \mathbf{v}^t \ dt
\end{aligned}
\tag{13}
$$

In a straightforward way it is possible to apply (11) in (13) producing the following standard Lagrangian form:

$$
\begin{aligned}
T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^n, t^n) &= \int_{t^n}^{t^{n+1}} \Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^t dt \\
&= \theta \Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^{n+1} \Delta t + \\
&\quad (1-\theta)\Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^n \Delta t \\
\mathbf{x_p}^{n+1} - \mathbf{x_p}^n &= \theta \mathbf{v}^{n+1} \Delta t + (1-\theta)\mathbf{v}^n \Delta t
\end{aligned}
\tag{14}
$$

2.3.2 Navier-Stokes

The extension of the time discretization to the Navier-Stokes equations needs to solve the pressure-velocity coupling.

It is well known that the velocity vector unknown arises solving the vector momentum equation. Being the pressure the scalar unknown for which the continuity equation might be the natural choice, in this equation the pressure does not appear. Moreover, this equation is not a time evolution equation, it works like a constraint over the velocity field, choosing only those velocity field that satisfy a free divergence. To discover the equation associated with the pressure several alternatives are possible. Among them, segregated or projection methods like fractional step appear as good candidates. The idea behind the fractional step is to write the momentum equation discretized in time in such a way to firstly predict a velocity using the old value of the pressure (the pressure at the old time step) and after correcting it with the updated pressure that arises from applying the divergence operator to the correction equation getting a Poisson like equation for the pressure.

In synthesis the fractional step may be viewed as:

$$
\begin{aligned}
\mathbf{v}^{n+1} - \mathbf{v}^n &= \Big(\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}\Big)^{n+\theta}\Delta t \\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n &= \theta\Big(\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}\Big)^{n+1}\Delta t + (1-\theta)\Big(\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}\Big)^{n}\Delta t \\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n &= \theta\Big(-\nabla p + \nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t + \\
&\qquad (1-\theta)\Big(-\nabla p + \nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t \\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n &= \theta(-\nabla p^{n+1} + \nabla p^n)\Delta t - \theta\nabla p^n\Delta t + \theta\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t \\
&\qquad + (1-\theta)\Big(-\nabla p + \nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t \\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n &= \theta(-\nabla p^{n+1} + \nabla p^n)\Delta t - \nabla p^n\Delta t \\
&\qquad + \theta\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t \\
&\qquad + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t \\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n &= \theta(-\nabla p^{n+1} + \nabla p^n)\Delta t - \nabla p^n\Delta t + \theta\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t \\
&\qquad + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t \\
\underbrace{\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1}}_{CORRECTOR} + \underbrace{\widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n}_{PREDICTOR} &= \underbrace{\theta(-\nabla p^{n+1} + \nabla p^n)\Delta t}_{CORRECTOR} - \underbrace{\nabla p^n\Delta t}_{PREDICTOR} \\
&\qquad + \underbrace{\theta\Big(\nabla\cdot\mathbf{v}(\widehat{\mathbf{v}}^{n+1}) + \mathbf{f}\Big)^{n+1}\Delta t + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t}_{PREDICTOR}
\end{aligned}
\tag{15}
$$

Spiting the predictor and corrector parts of the equation in two steps and applying the divergence to the corrector step using the constraint that $\nabla\mathbf{v}^{n+1} = 0$,

PREDICTOR

$$
\begin{aligned}
\widehat{\mathbf{v}}^{n+1} - \mathbf{v}^n &= -\nabla p^n\Delta t + \theta\Big(\nabla\cdot\mathbf{v}(\widehat{\mathbf{v}}^{n+1}) + \mathbf{f}\Big)^{n+1}\Delta t \\
&\qquad + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t
\end{aligned}
$$

PRESSURE EQUATION

$$
\begin{aligned}
\nabla\cdot\Big(\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} &= -\theta\Delta t(\nabla p^{n+1} - \nabla p^n)\Big) \\
\nabla\cdot\widehat{\mathbf{v}}^{n+1} &= \theta\Delta t\nabla\cdot(\nabla\delta p) = \theta\Delta t\nabla^2\delta p
\end{aligned}
$$

CORRECTOR

$$
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} = -\theta\Delta t\nabla\delta p \tag{16}
$$

with $\delta p = p^{n+1} - p^n$ and ∇^2 the Laplacian operator. The equation for the pressure is interposed between the predictor and corrector equations for the momentum equation as it is normally found in the algorithm.

For the Eulerian formulation the above three steps may be applied straightforward only changing $\mathbf{v}$ by $\mathbf{v}(\mathbf{x})$ and p by $p(\mathbf{x})$.

For the Lagrangian formulation the above algorithm may be summarized as:

PREDICTOR
Explicit part

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^{n} + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x_p}^{\tau}) d\tau$$

$$\hat{\hat{\mathbf{v}}}^{n+1}(\mathbf{x_p}^{n+1}) - \mathbf{v}^n(\mathbf{x_p}^n) = \int_{t^n}^{t^{n+1}} -\nabla p^n(\mathbf{x_p}^{\tau}) + (1-\theta)\Big(\nabla \cdot \tau_{\mathbf{v}}^n(\mathbf{x_p}^{\tau}) + \mathbf{f}^n(\mathbf{x_p}^{\tau})\Big) d\tau$$

Implicit part

$$\hat{\mathbf{v}}^{n+1}(\mathbf{x}) - \hat{\hat{\mathbf{v}}}^{n+1}(\mathbf{x}) = \theta\Big(\nabla \cdot \tau_{\mathbf{v}}(\hat{\mathbf{v}}^{n+1}(\mathbf{x})) + \mathbf{f}^{n+1}\Big)\Delta t$$

PRESSURE EQUATION

$$\nabla \cdot \hat{\mathbf{v}}^{n+1}(\mathbf{x}) = \theta \Delta t \nabla^2 \delta p(\mathbf{x})$$

CORRECTOR

$$\mathbf{v}^{n+1}(\mathbf{x_p}) - \hat{\mathbf{v}}^{n+1}(\mathbf{x_p}) = -\theta \Delta t \nabla \delta p(\mathbf{x_p}) \tag{17}$$

It should be noted that for the whole procedure of Lagrangian formulation two coordinates are used, one for the particles ($\mathbf{x_p}$) and the other for the mesh ($\mathbf{x}$). The relation between them is presented in a next section.

2.4 X-IVS [X-IVAS]: Explicit Integration Velocity [Acceleration] Scheme

In the scalar transport equation the velocity field is a given data, known not only for its spatial variation also for its time variation. Therefore it is possible to include this information explicitly in the Lagrangian formulation. Using a high accurate particle tracking integration scheme it is possible to solve simple and complex pathlines normally present in fluid flows (Figs. 1 and 2).

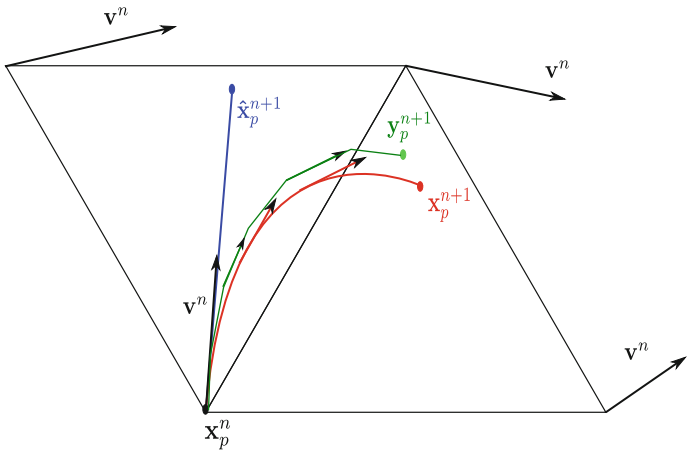

Fig. 1 Time integration

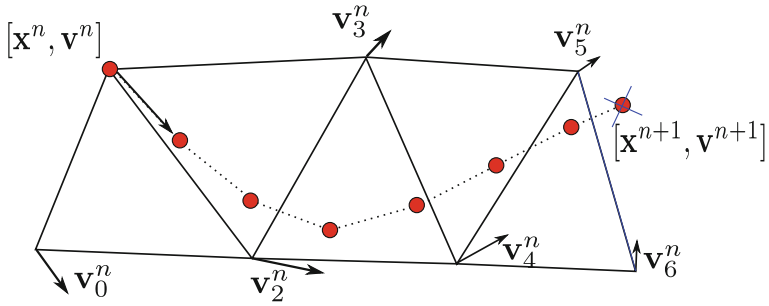

Fig. 2 Streamline integration updating the particle position and state

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^\tau(\mathbf{x_p}^\tau) d\tau \tag{18}$$

- Real trajectory:

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^n + \int_n^{n+1} \mathbf{v}^\tau(\mathbf{x_p}^\tau)\, d\tau. \tag{19}$$

- Simple approximation:

$$\widehat{\mathbf{x_p}}^{n+1} = \mathbf{x_p}^n + \mathbf{v}^n(\mathbf{x_p}^n)\Delta t \tag{20}$$

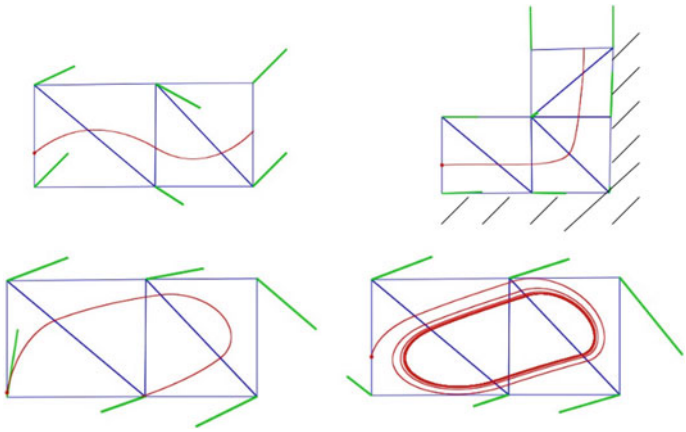

Fig. 3 Streamline integration updating the particle position and state

- Streamlines approximation:

$$\mathbf{y}_p^{n+1} = \mathbf{x_p}^n + \sum_{i=0}^{N-1} \mathbf{v}^n(\mathbf{y}_p^{n+\frac{i}{N}})\,\delta t \tag{21}$$

2.4.1 Remarks

- Adaptive substep: $\delta t = \frac{\Delta t}{N} \propto Co_{ELE}$. These integration substeps may be replaced by an almost exact integration [18]. However for practical applications a very little difference in accuracy is observed for the former with an extra cost for the later that push the decision on the former.
- For vector systems where normally the velocity field is also an unknown the particle velocity $\mathbf{v_p}$ may also be updated following the same ideas changing the name of the method to **X-IVAS**.
- for fluxes depending on the spatial derivatives (like diffusive) **X-IVAS** method is also applied.
- This novel integration in Lagrangian context allows to a significantly better particle trajectory integration **(X-IVS)** following streamlines resolving more difficult details of the flow with high accuracy. Figure 3 shows some details that may be resolved without a drastically reduction in the time step, as it is normally done by standard Lagrangian integration schemes.
- In general this integration scheme does not suffer for strong time step restrictions caused by the non-linearities present in the flow field (Fig. 3).

As it was mentioned for the unknown field, the temperature, only known for the old time step t^n, an approximation to the future time step should be done using the **X-IVAS** method.

$$T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^{n}, t^{n}) = \int_{t^n}^{t^{n+1}} Q(\mathbf{x_p}^{\tau}, \tau) d\tau + \int_{t^n}^{t^{n+1}} \Big(\nabla \cdot (\alpha \nabla T(\mathbf{x_p}^{\tau}, \tau)) \Big) d\tau \tag{22}$$

In the *PFEM* method the last term at the right hand side is approximated in the following form:

$$\begin{aligned} T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^{n}, t^{n}) \approx & \int_{t^n}^{t^{n+1}} Q(\mathbf{x_p}^{\tau}, \tau) d\tau \\ & + \underbrace{(1-\theta) \int_{t^n}^{t^{n+1}} \nabla \cdot (\alpha \nabla T(\mathbf{x_p}^{\tau}, t^{n})) d\tau}_{explicit} \\ & + \underbrace{\theta \nabla \cdot (\alpha \nabla T(\mathbf{x_p}^{n+1}, t^{n+1})) \Delta t}_{implicit} \end{aligned} \tag{23}$$

The last integration is only one possibility to choose among others, the explicit part is solved simultaneously with the particle pathline computation, while the implicit one is solved using the final position of the particles. However other choices may be done in order to improve this computation, that for brevity reasons are not included in this paper.

Comparing (12) with (23) the main difference between both may be written as:

$$\begin{aligned} \Delta_{EUL \to LAG} = & T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^{n}, t^{n}) \\ & - \Big(T^{n+1}(\mathbf{x}) - T^{n}(\mathbf{x}) + \theta \mathbf{v}(\mathbf{x}, t^{n+1}) \cdot \nabla T(\mathbf{x}, t^{n+1}) \Delta t \\ & + (1-\theta) \mathbf{v}(\mathbf{x}, t^{n}) \cdot \nabla T(\mathbf{x}, t^{n}) \Delta t \Big) \end{aligned} \tag{24}$$

This difference is due to the error produced by the transformation between both frames, an Eulerian or fixed one and the Lagrangian or mobile one. This difference should tend to zero when the time step goes to zero. However, for large time steps normally needed to speed up the computation, the fact of evaluating the velocity field placed on a fixed position ($\mathbf{x}$) for Eulerian formulation in two different time intervals may introduce large errors. Moreover the spatial stabilization needed for advection dominant problems introduce also some extra errors that tends to dissipate the solution much more than the physics, specially at large time steps.

2.5 A Simple Coupled System, Solving the Coupled Flow Field and a Passive Scalar

One of the main purposes of this development is its application to solve coupled problems where the flow and several other fields are solved simultaneously with some sort of interaction. For brevity a simple case is here presented. It deals with the coupling of an incompressible viscous flow with a scalar transport like temperature using the Boussinesq approach.

2.5.1 Physical Equations to Solve

- Navier-Stokes Equation System for Newtonian and incompressible fluids:

$$\begin{aligned} \rho \frac{D\mathbf{v}}{Dt} &= -\nabla p + \nabla \cdot \Big(\mu(\nabla \mathbf{v}^T + \nabla \mathbf{v})\Big) + \mathbf{f} \\ \nabla \cdot \mathbf{v} &= 0 \end{aligned} \tag{25}$$

- Scalar-Transport Equations:

$$\frac{D\phi_j}{Dt} = \nabla \cdot (\alpha_j \, \nabla \phi_j) + Q_j \quad \forall j \in (1 : N_{fields}) \tag{26}$$

- Example of coupling. For $\phi = T \Rightarrow$ Boussinesq approximation:

$$\mathbf{f} = \rho \mathbf{g} \beta (\phi - \phi_c) \tag{27}$$

2.5.2 Discretization

The key of the PFEM algorithm is transporting the information with particles following the streamlines. Although the field $\mathbf{v_p}$ is not stationary, streamlines are taken as stationary on each time step ($\mathbf{v_p}^n$), the particle position follows that field and the particle state is updated by the rate of change determined by the balance equation.

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x_p}^\tau)\, d\tau \tag{28a}$$

$$\mathbf{v_p}^{n+1} = \mathbf{v_p}^n + \int_{t^n}^{t^{n+1}} (\mathbf{a}^n(\mathbf{x_p}^\tau) + \mathbf{f}^\tau)\, d\tau \tag{28b}$$

$$\phi_p^{n+1} = \phi_p^n + \int_{t^n}^{t^{n+1}} (\mathbf{g}^n(\mathbf{x_p}^\tau) + Q^\tau)\, d\tau \tag{28c}$$

where $\mathbf{a}^n = -\nabla p^n + \nabla \cdot \left(\mu(\nabla^T \mathbf{v}^n + \nabla \mathbf{v}^n)\right)$ and $\mathbf{g}^n = \nabla \cdot (\alpha\, \nabla \phi^n)$, which are nodal variables.

3 Two Examples to Show the Benefits of the New Ideas

The following two examples serve as the starting point of new ideas behind high accurate and stable convective transport equations.

- Pure advection of a passive scalar field.
- Inviscid transport of a vortex.

The first example was the proof in showing the advantages of using Lagrangian formulations when a pure advective problem is between hands. It is a Gaussian hill profile imposed as an initial condition advected by a pure rotation motion. For this problem the profile shape and its amplitude should be conserved all the way. The second one is one extension of the first example applied to a vector system like Navier-Stokes equations. It consists of an initial vortex that is transported in an inviscid flow. For this problem the intensity of the vortex should be conserved.

3.1 Pure Advection of a Passive Scalar Field

This well known problem normally serves as a benchmark for the spatial stability of Eulerian numerical schemes. The first scope in this benchmark is to show that no spurious oscillations appear and the second one focus on minimizing the numerical diffusion introduced by the stabilization schemes. Also the time integration numerical scheme is responsible for extra numerical dissipation, being the first order explicit ($\theta = 0$) or implicit ($\theta = 1$) schemes not recommended for their high dissipation. Crank-Nicholson ($\theta = \frac{1}{2}$) is preferred in this sense. However looking at the solution it is always present a reduction in the original amplitude that may be improved only reducing the mesh size and the time step.

In [10] several Figs. 9.1–9.4 are shown where it is possible to realize how the amplitude is drastically reduced using large Courant numbers with Eulerian formulations. Even though the spatial stabilization is reduced to a minimum and the time integration is chosen as second order the numerical dissipation is highly noticeable. Only reducing the Courant number with finer meshes it is possible to reduce it but never annihilate.

On the other hand Lagrangian formulations are better positioned for this kind of problems if only the amplitude is observed. This remains exactly constant all the way regardless the Courant number. However with standard first order integration, the problem arises in the definition of the pathlines that are shifted inwards or outwards depending on the explicit or implicit character of the time integration. Only with second order time integration is possible to reduce this pathology but some sort of iteration is needed. See Fig. 9.7 in [10].

Using the X-IVAS integration is possible to fix both errors simultaneously producing a high accurate solution regardless the Courant number. This is also shown in Fig. 9.7 at left in [10].

A final remark about this important result achieved on such a basic example, showing the great capabilities of Lagrangian formulations over Eulerian ones for convective transport, is related to some more accurate Eulerian schemes that are currently being published for transporting signals without causing spurious diffusion. Called as High Resolution Schemes [14, 16], these numerical methods have a robust control to suppress the wiggles with the minimum amount of numerical dissipation. According to the Godunov theorem [16] this is only possible in a nonlinear way. Even though this way circumvents the drawbacks it is important to realize that Lagrangian formulations achieved the same or better results without doing nothing special saving the extra cost normally experienced by such schemes.

3.2 Inviscid Transport of a Vortex

Having found the good benefits of Lagrangian formulations for transporting scalar fields in a stable and accurate way the following step was to extend the same to vector systems. Here the incompresible viscous fluid flow model was taken. The equivalent example in this context is the transport of a vortex in an inviscid flow. It is well known that looking at the hyperbolic part of the whole system, neglecting the diffusion and not considering the role of the pressure, the problem looks like the convection of vorticity waves. If a vortex ring is imposed as initial condition, neglecting the viscosity with slip boundary conditions on the walls, the vortex should conserve its kinetic energy as much as possible.

Figure 4 shows how the Eulerian and Lagrangian formulation transports this vorticity. For both formulations the mesh is kept fixed and the simulation had run with the same time step, with a high enough Courant number in order to highlight the performance and precision comparison. It is possible to conclude that the Lagrangian formulation is more energy conservative than the Eulerian counterparts with the evidence that the vortex may be transported much better. This example confirms the advantages of Lagrangian formulation respect to Eulerian ones not only for advective transport of scalar fields, also for non linear vector fields (Fig. 4).

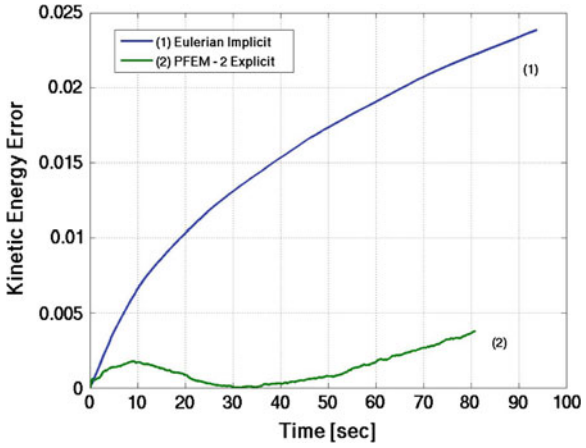

Fig. 4 Inviscid vortex ring dynamics. Conservation of momentum

4 Moving and Fixed Mesh PFEM Versions

The natural evolution of *PFEM* method employed only one mesh built from a cloud of points defined by the moving particle position. There was a one to one relation between mesh nodes and particles. At each time step the original *PFEM* method moved the nodes following the updated particle positions as long as the mesh was not deformed in such a way that an invalid grid appears. Remeshing was only used when the deformation of the mesh was so large that the time step suffered a drastic reduction making the computation too much expensive. At that times, the remeshing was by-passed at extreme for cpu times reasons. Summarizing the stability of the original *PFEM* was mainly affected by:

- critical time step for explicit advective terms ($Co < O(1)$).
- critical time step for explicit diffusive terms ($Fo < O(1)$).
- critical time step for the deforming mesh limited by the inversion of some elements in the mesh (invalid)
- non linearities

The sequence of the problems above defined may be summarized as:

- To solve only the passive scalar transport Eq. (28a) and (28c) are used. If you want to transport N passive scalars, you only have to solve (28c) for each one of the N variables.
- To solve the Navier Stokes equation system, (28a) and (28b) are used, and p must be calculated. A typical Fractional Step Method is used to solve the coupling between the pressure and the velocity (see Eq. 16).
- To solve the thermal and fluid flow coupling (natural convection) (28a), (28c) and (28b) are used and a constitutive law for the buoyancy term should be added (Boussinesq approach)

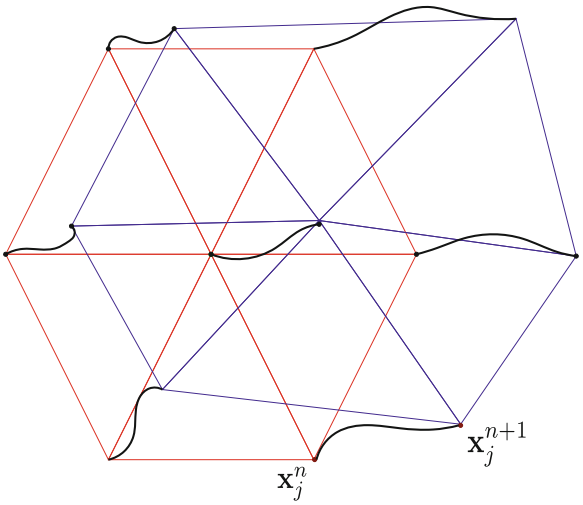

Fig. 5 *PFEM* —moving mesh version

All these steps need a mesh update and again the remeshing returns to the scenario. During the last years a lot of progress was done in terms of more efficient mesh generation and regeneration exploiting the parallelism, making the remeshing affordable. A permanent remeshing circumvents the severe time step restriction produced by the invalid mesh condition. In this sense the *PFEM* had experienced a significant progress increasing the time steps with stable solutions.

- It is necessary to update the mesh states with the particles states. There are two approaches which have generated two versions of the method:
  - Remesh the geometry with new particles positions (particles↔nodes): **PFEM Mobile Mesh**
  - Project states from particles to nodes, preserving the mesh as fixed: **PFEM Fixed Mesh**

The Mobile Mesh version has the following features:

- a 1-1 relation between Particles and Nodes.
- Remeshing at each time step.
- Need permanent assembling, profiling and solving of the algebraic linear system.

Figure 5 shows how the particle motion change the mesh definition at each time step.

The first tests showed very good features in terms of stability and accuracy getting a drastic reduction in the cpu times involved for solving some benchmarks compared with the original version of *PFEM*. This improved performance, added to the possibility of using very large time steps were the first evidences that the permanent

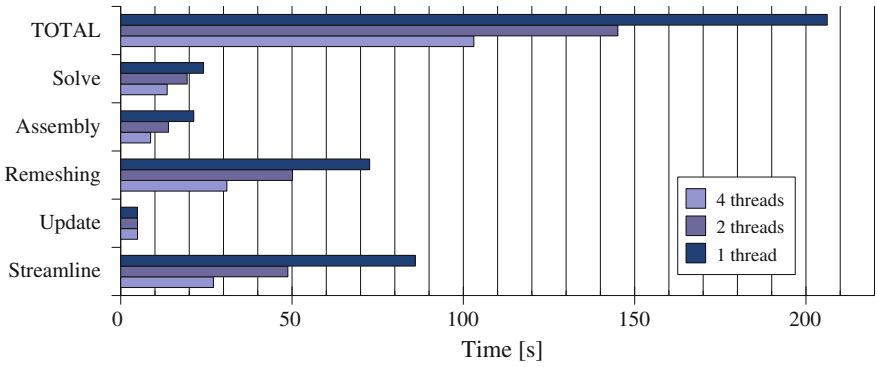

Fig. 6 *PFEM* —moving mesh version—profiling

remeshing using X-IVAS integration were two important numerical ingredients for exploiting the good features evidenced by the Lagrangian formulation.

Even though moving mesh *PFEM* version has several advantages against its Eulerian counterpart, it has some limitations in terms of efficiency. Mainly the permanent remeshing and the assemble/solving of implicit problems limit its scalability The Fig. 6 shows a profiling of the moving mesh version of the *PFEM*.

As it is evident from this figure most of the time is spent in remeshing, assembling and solving the implicit linear systems, with a performance similar to mesh based methods because the particle update only consumes a small part of the whole cost.

In order to reduce the computational cost added by these two stages a novel idea was presented: *the Fixed Mesh Version* of *PFEM* .

This new method combines particles with a background mesh. Particles carry the information along the whole process using the mesh only for secondary computations, those needed to update the particle position and their states. It is normally understood as an hybrid method or dual method where particles act like the master in the computation and the mesh is the slave. The idea does not only avoid the permanent remeshing, using a fixed background mesh it is possible to integrate all the implicit part of the computations with an important and favorable impact on the computational efficiency, the possibility of re-using the matrix profile and for linear diffusion problems also its factorization.

This fact may be exploited only by Lagrangian formulation because the Eulerian counterpart always has the convective term proportional to the changing unknown velocity inside the system matrix to be inverted. Therefore it is not possible to take advantage of it.

The fixed mesh version has the following features :

- Particles cloud over a Fixed Background Mesh.
- No need remeshing.
- It needs Projections and Interpolations between particles and mesh nodes.
- It needs only one LU or Cholesky factorization for implicit calculations.

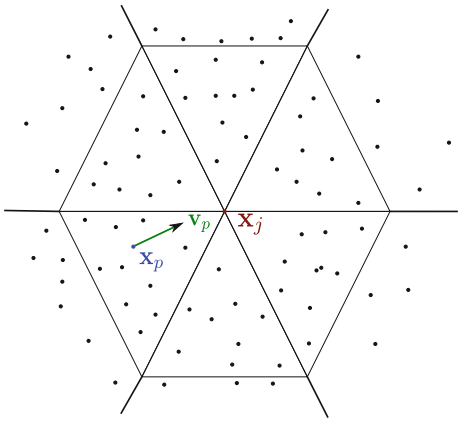

Fig. 7 *PFEM* —fixed mesh version

For constant coefficients problems an initial factorization may be built, reusing it along the whole computation. In this way it is possible to reduce the number of iterations of the preconditioned conjugate gradient used for solving the symmetric and positive definite linear systems involved in the computation (Poisson for the pressure and the heat equation for the implicit part of the diffusive terms ($\theta > 0$)).

However, the fixed mesh version has some cons, specially that concerned with the projection error. This operation allows to reconstruct some fields transported by the particles in a very accurate way on a mesh where the spatial derivatives should be computed. The better the reconstruction the less the error in the computation of the fluxes depending on the spatial derivatives (Fig. 7).

Several numerical experiments have shown that the advection part of the computation is governed by particles and the diffusive part is governed by the mesh. Therefore increasing the number of particles allows to transport very complex initial conditions or represents complex fields produced by its time evolution under complex flow fields. On the other hand refining the mesh allows to improve the diffusive fluxes computation. But this is not the only way to get it, higher order reconstruction is also an alternative to explore in the future.

First order reconstruction may become inconsistent, i.e. refining in the number of particles may produce higher diffusive flux errors. To avoid it one possibility could be to split the elements in subdomains, one around each element node, using only that division to project particles values on the node associated with that subelement.

Instead of using all the particles belonging to the elements connected to a given node, only those particles belonging to the subelements connected to the node are chosen. This subdivision also serve to seed particles when that subelement is void of particles. In a next section some details about it are presented.

In terms of efficiency the Fig. 8 shows as the fixed mesh version has changed its profiling, now with the most important computational cost component concentrated in the particle computation, typical of a Lagrangian formulation.

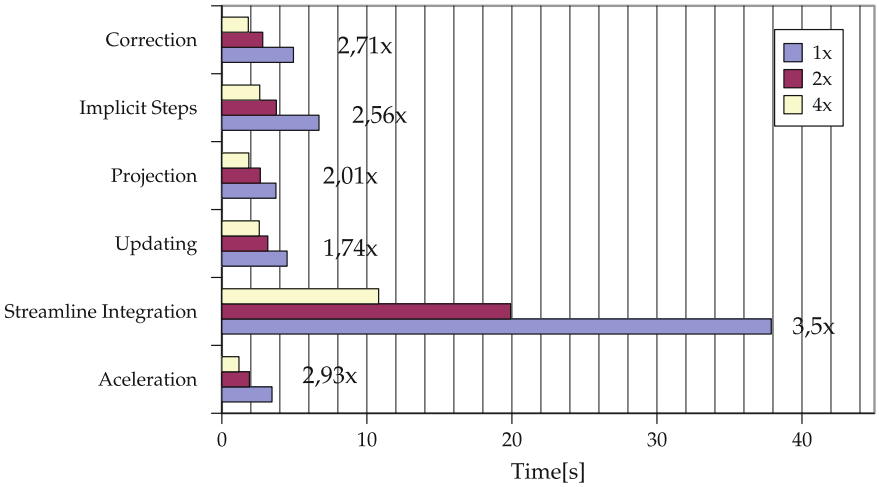

Fig. 8 *PFEM* —fixed mesh version—profiling. *Case* flux around a cylinder 2D—CPU: Intel i7-2600k (4 cores)

Being particle computations cheaper than mesh computations it put the *PFEM* method in a very good condition for high performance computing stuffs, specially for scalability.

Finally this section ends with some review of the two algorithms that were firstly developed in the context of the *PFEM*, one for the scalar transport and the other for the incompressible viscous flow.

Algorithm 1 - Time Step PFEM Scalar Transport

1. Calculate scalar change rate on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\, \mathbf{g}^n\, d\Omega = -\int_\Omega \nabla \boldsymbol{N}\, \alpha \nabla \phi^n\, d\Omega + \int_\Gamma \boldsymbol{N}\, \nabla \phi^n \cdot \eta\, d\Gamma$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\, d\tau$
   $\phi_p^{n+1} = \phi_p^n + \int_{t^n}^{t^{n+1}} \mathbf{g}^n(\mathbf{x}_p^{n+\tau}) + Q^{n+\tau}\, d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\phi_j^{n+1} = \boldsymbol{\pi}(\phi_p^{n+1})$

Algorithm 2 - Time Step PFEM Incompressible Flow

1. Calculate acceleration on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\ \nabla\cdot\tau_v\ d\Omega = -\int_\Omega \nabla\boldsymbol{N}\cdot(\mu\nabla\mathbf{v}^n)\ d\Omega + \int_\Gamma \boldsymbol{N}\ \nabla\mathbf{v}^n\cdot\eta\ d\Gamma$
   $\int_\Omega \boldsymbol{N}\ \nabla p^n\ d\Omega = -\int_\Omega \nabla\boldsymbol{N}\ p^n\ d\Omega + \int_\Gamma \boldsymbol{N}\ p^n\cdot\eta\ d\Gamma$
   $\mathbf{a}^n = \nabla\cdot\sigma = -\nabla p^n + \nabla\cdot\tau_v$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\ d\tau$
   $\hat{\mathbf{v}}_p^{n+1} = \mathbf{v}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{a}^n(\mathbf{x}_p^{n+\tau}) + \mathbf{f}^{n+\tau}\ d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\mathbf{v}_j^{n+1} = \boldsymbol{\pi}(\mathbf{v}_p^{n+1})$
5. Find the pressure value solving the Poisson equation system using FEM:
   $\rho\nabla\cdot\hat{\mathbf{v}}_j^{n+1} = \Delta_t\Delta[\delta p^{n+1}]$
6. Update the velocity value with the new pressure:
   $\rho\mathbf{v}_j^{n+1} = \rho\hat{\mathbf{v}}_j^{n+1} - \Delta_t(\nabla p^{n+1} - \nabla p^n)$
   $\rho\mathbf{v}_p^{n+1} = \rho\hat{\mathbf{v}}_p^{n+1} - \Delta_t\boldsymbol{\pi}^{-1}(\nabla p^{n+1} - \nabla p^n)$

It is important to realize that in the fixed mesh version of *PFEM* there is an operation called projection that deserves some attention. In a next section some comments about it will be presented.

5 Particle Inventory Managment

As mentioned before, in the *PFEM* algorithm, after the streamline integration the state variables are placed on the particles. Both, for reasons of incompressibility (pressure) as for the treatment of the diffusion (viscous stress tensor) require that the information should be located on the grid. While for the Mobile Mesh version the mesh is done with the particles themselves, in the Fixed Mesh approach particles and grid are decoupled and a projection $\boldsymbol{\pi}$ from particle states to nodal states should be done.

5.1 Projection Algorithms

Different approaches are available to perform projections, for example SPH or MLS (Moving Least Square) techniques could be used for the interpolation, as well as weights based on the position on the top of the underlying mesh. A brief review of the actual techniques for projection in PFEM are presented (the equations presented are for scalar projection. They are also valid for each component of a vector state variable):

- Mean weighted by Shape Function (P-1):

$$\phi_j = \frac{\sum_{p=1}^{P} \mathbf{N}_j(\mathbf{x}_p)\phi_p}{\sum_{p=1}^{P} \mathbf{N}_j(\mathbf{x}_p)} \tag{29}$$

  where P is the number of particles inside a certain region around the node j.
- Mean weighted by Distance (P-2):

$$\phi_j = \frac{\sum_{p=1}^{P} ||\mathbf{x}_p - \mathbf{x}_j||_2 \phi_p}{\sum_{p=1}^{P} ||\mathbf{x}_p - \mathbf{x}_j||_2} \tag{30}$$

  with the same definition of P as just above.
- Weighted Polynomial Least Squares (P-3):

$$\phi_j = h_\theta(\mathbf{x}_j) = \boldsymbol{\theta}^T \mathbf{P}(\mathbf{x}_j) \tag{31}$$

  where:
  $\mathbf{P} = [1\ x\ x^2 \ldots]$ on 1d, $\mathbf{P} = [1\ x\ y\ xy\ x^2\ y^2 \ldots]$ on 2d (truncating at the polynomial order required) and $\boldsymbol{\theta} = (\mathbf{X}^T\ \mathbf{W}\ \mathbf{X})^{-1}(\mathbf{X}^T\ \mathbf{W}\ \mathbf{y})$. It must be noted that to invert the matrix in the calculation of $\boldsymbol{\theta}$, it is required $P >= n$, where n is the number of terms used on P.

5.2 Particle Seeding

For accuracy reasons each one of the presented projection methods require a certain number of particles in certain region near to each node. Some considerations must be taken into account: the *region around the node* must be defined precisely, and is not assured that there were particles inside each region (specially when high Co numbers are used). Then, new particles must be created at these empty regions. In this section several algorithms attending this issue are presented.

The first approach (S-1), used originally in PFEM, consists on setting the states to a new particle interpolating from the nodal states at the previous time step n:

$$\phi_p^{n+1}(\mathbf{x}_p^{n+1}) = \sum_j \lambda_j(\mathbf{x}_p^{n+1})\phi_j^n \tag{32}$$

being λ_j the area coordinates of the particle in the element. Other algorithm (S-2) searches the state following the streamline but in backward direction, thinking in

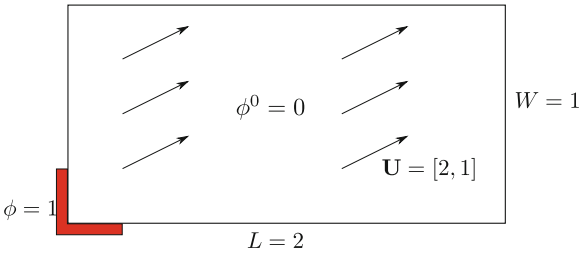

Fig. 9 Pure-convection step problem

finding the particle location that, if it had existed at the beginning of the time step, at the end of it would have arrived at the seed position:

$$\phi_p^{n+1}(\mathbf{x}_p^{n+1}) = \phi_p^n(\mathbf{x}_p^n) + \int_0^{\Delta t} \mathbf{g}^n(\mathbf{x}^{n+\alpha})\, d\alpha \tag{33}$$

The utility of the backward integration to search the state of the new particle is shown in the next example: the pure-convection step problem, which is defined in Fig. 9.

A boundary condition with a sharp discontinuity enters the domain transported with a velocity vector field not aligned with the mesh. Figure 10 shows the results, projected on the mesh, achieved when the particles are created using the criterion defined as (S-1) (left) and also when their states are found using backward integration criterion (S-2) (right).

The results show that S-1 criterion fails for this example putting S-2 criterion as a much better selection for seeding particles when it is necessary.

5.2.1 Particle Removing

On other hand, using a lot of particles increases computing times. During the simulation the seeding is frequent and we need to control the amount of particles inside the domain for computational cost reasons. So, the removal action should be defined following some criteria.

Although it is known that particle which leaves the geometry must be deleted, inside the geometry is not clear when particles should be removed and how to do that. Removing particles decreases computing times of the algorithm but also decreases the quality of the solution because it introduces numerical diffusion in an indirect way.

In the first approach (R-1), the particles are not removed unless that two or more of them are in almost the same position. This approach obtained accurate results solving the pure-advective case of the rotating Gaussian signal [18].

A second approach (R-2), consists on requiring minimum number and a maximum number of particles at each sub-element that must be conserved at each time step.

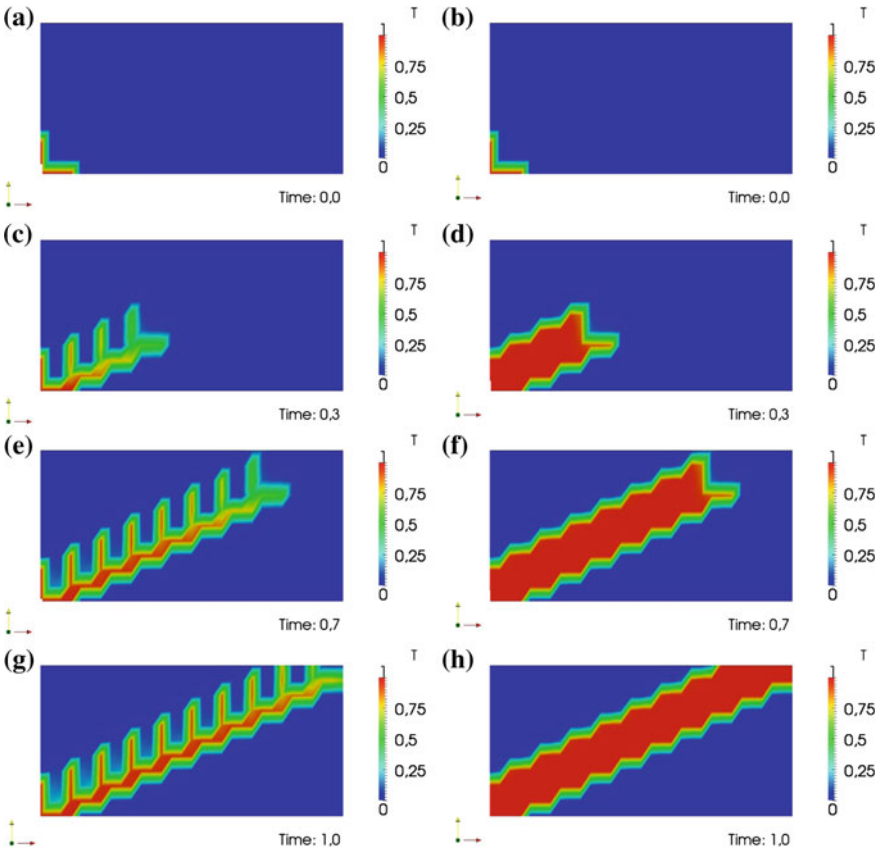

Fig. 10 Pure convection transport. Results for different strategies to new particles states. *Left* S-1. *Right* S-2

The sub-element i is the third parts of the triangle (fourth parts of the tetrahedral in three dimensions) where the area coordinate corresponding to the vertex i is larger than the rest. The idea is to think that if each node has enough number of particles around, the projected state from particles to the node will be accurate. However, as will be demonstrated in the next example, the continuous intrusion to the system could decrease the quality of the solution, specially in scalar problems. This criterion shows good results in solving incompressible flows.

The example of the *rotating Gaussian signal without diffusion* shows that the numerical diffusion is important when a frequent creation and removing of particles is performed following this criterion. The polar mesh (4390 elements) is the same in all cases, the case consists of a Gaussian transported by a rotating flow without diffusion term. Figure 11 compares the value of the maximum through two laps. The best option for this problem is R-1 (in Fig. 1 Tmax old), while R-2 requires a large range ($[min\_subele; max\_subele]$) of particles not to spread: comparing

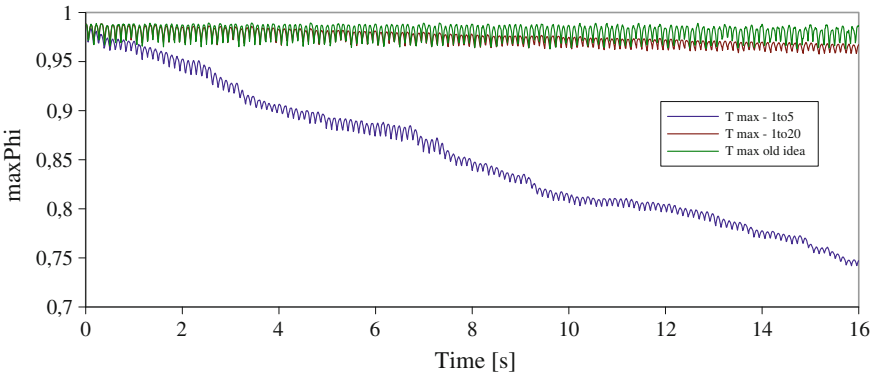

Fig. 11 Rotating Gaussian. Evolution of $\phi_{\max}$ for different update particle techniques

[1; 20] with [1; 5], in the second option the intrusion in the system is greater and the solution decreases its quality.

Regarding the computing times also R-1 is the best, because requires 40 s (mean 40,000 particles), whereas [1; 5] needs 50 s (mean 46,000 particles) and [1; 20] needs 100 s (mean 120,000 particles).

However, R-1 does not work fine for the step's problem (defined in previous subsection), unless it creates new particles using backward integration (criterion S-2). Also, due to the type of projection of the algorithms developed, which searches particles in the sub-elements to send data to nodes, creating new particles in empty elements does not ensure that there will be particles in the region around the node (their sub-elements), so another type of selection of the position of the new particles must be developed.

5.3 Converging into a New Algorithm to Update Particle Inventory

Taking into account the previous discussion, an algorithm to update particle inventory has been implemented, that includes the benefits of the methods discussed previously and follows the principle of not modifying the system unless it be really necessary.

It was mentioned that the condition *empty-element*, except for some particular problems, shows to work not so fine. The main reason is the lack of particles close enough to nodes. So, the new algorithm must not search *empty-elements*, instead of that, it searches *empty-regions*.

The region j is defined as the geometric space where the node j takes particle information to update its state using some projection operator; therefore, if this region is empty, the node does not have enough information to be updated. Once detected an *empty-region*, only one particle is created in exactly the same position of its node and with a state found using backward integration.

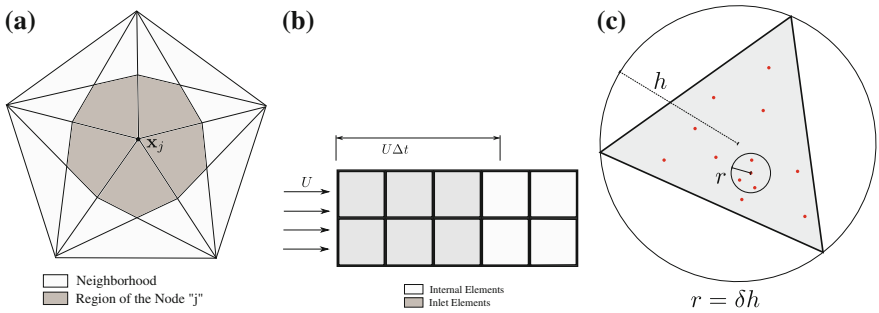

Fig. 12 Graphical scheme to understand the new algorithm proposed

Although to create new particles on the nodes is the least intrusive way to maintain good solution on the nodes, if the problem is not of *confined flow*, the number of particles will decrease while the simulation runs. Typically in the inlet flow boundary the boundary condition is imposed. Then, creating particles in the *Inlet Elements* and doing backward integration to search the states no error is committed, and more than one particles can be created. The position does not have to be on the nodes and its state will not have error. This approach allows to keep approximately the same numbers of particles during the simulation, which preserves the accuracy of the method.

Particle removing is carried out when two or more particles are in a circle (2d) or sphere (3d) with a radius proportional to the size of the element ($r = \delta$ h). This approach allows to use different δs over the geometry, being a new tool to control the number of particles. Graphic representation is presented in the Fig. 12c.

Figure 12a shows the neighbor elements and the region belonging to the node j. It must be there at least one particle in the gray zone to have a good projection, else a new particle must be created in the same position of the node and searching its state with backward integration. Figure 12b shows which elements are considered as *inlet elements* and, if they are empty, they must create internal particles.

Finally, this algorithm allows to solve all tests presented in this paper while other approaches have shown to fail: the *Gaussian rigid rotation* and the *step-2d*.

The last example consists on testing this algorithm in the Navier-Stokes solver (*PFEM Fixed Mesh*). The case chosen is the *Flow Around a Cylinder*, because it presents different zones of refinement and patches of inlet and outlet flow. The results are presented in the Fig. 13a and b. Similar accuracy in the amplitude and frequency can be observed, but R-2 obtains better definition of the forces signal, specially for Cd. For more details see [10, 11].

6 Diffusive-Dominant Problems

When the problem is diffusive-dominant, the advantages of the method PFEM are not as clear as in the advective-dominant case. The explicit calculation of the diffusion traditionally used by PFEM is limited by the dimensionaless Fourier number and,

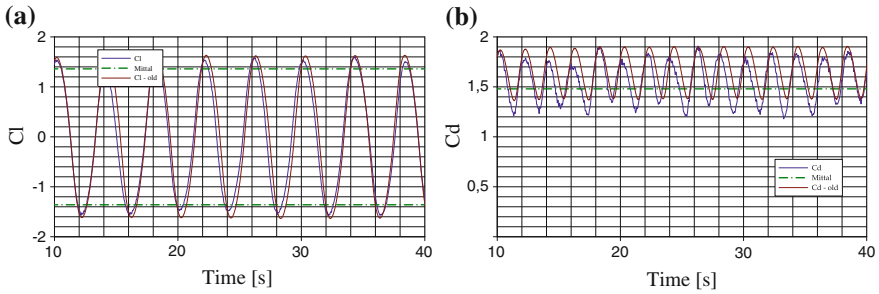

Fig. 13 Lift coefficient and drag coefficient for flow around a cylinder solved using the new updating algorithm and comparing it with R-2 (called *old* in the graphic)

in some particular cases, the temporal change of the transported variables vanishes due to the shape of its own solution. To relax these restrictions, in the Sect. 6.1 a new model to calculate the diffusion is presented. Several tests are presented to confirm improvements in the solution.

6.1 Diffusive Implicit Correction

Simulations solving the diffusive term in an explicit way are restricted by $Fo < 0.5$. This is a strong limitation for the time-step, specially on very refined mesh and in diffusive dominant problems (where $Fo > Co$). Due to explicit PFEM suffers this stability constraint the possibility of enlarging the time step may be lost when the flow locally turns to be diffusive. As we have mentioned normally in the vicinity of bodies some refinement is done to capture boundary layers and flow separations and locally the Fourier number increases. Also, in some particular cases, the temporal change of the transported variables vanishes due to the shape of the own solution: when the time-step is chosen such that the integral of the curvature of the function ϕ_j^n vanishes, the method will not apply diffusion on ϕ, so that the solution will be wrong. This case may be present for traveling waves with diffusion.

A new approach to solve the diffusive term is based on the *theta method* which consists on discretizing the non-stationary variable using a weighted mixture between an explicit prediction and an implicit correction.

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = \theta \mathbf{g}^{n+1} + (1-\theta)\mathbf{g}^n \tag{34}$$

Doing a first step in explicit way

$$\frac{\hat{\phi}^{n+1} - \phi^n}{\Delta t} = (1-\theta)\mathbf{g}^n \tag{35}$$

and doing the correction in an implicit way, this is subtracting (35) from (34), it follows

$$\frac{\phi^{n+1} - \hat{\phi}^{n+1}}{\Delta t} = \theta \mathbf{g}^{n+1} \tag{36}$$

Algorithm 3 - Time Step PFEM Scalar Transport Explicit Diffusion - Implicit Correction

1. Calculate scalar change rate on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\ \mathbf{g}^n\ d\Omega = -\int_\Omega \nabla \boldsymbol{N}\ \alpha \nabla \phi^n\ d\Omega + \int_\Gamma \boldsymbol{N}\ \nabla \phi^n \cdot \eta\ d\Gamma$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\ d\tau$
   $\hat{\phi}_p^{n+1} = \phi_p^n + \int_{t^n}^{t^{n+1}} \mathbf{g}^n(\mathbf{x}_p^\tau) + Q^{n+\tau}\ d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\hat{\phi}_j^{n+1} = \boldsymbol{\pi}(\hat{\phi}_p^{n+1})$
5. Implicit correction:
   $\phi_j^{n+1} = \hat{\phi}_j^{n+1} + \Delta t\ \theta \mathbf{g}_j^{n+1}$.
6. Interpolate state to particles:
   $\phi_p^{n+1} = \hat{\phi}_p^{n+1} + \boldsymbol{\pi}^{-1}(\delta \phi_j^{n+1})$.

An standard FEM formulation is used to compute the implicit correction. This problem may be solved either with the approaches *absolute* (37) or *incremental* (38).

$$[\mathbf{M} + \theta\ \Delta t\ \mathbf{K}]\ \phi^{n+1} = \mathbf{M}\ \hat{\phi}^{n+1} + \Delta t\ \mathbf{F}^{n+\frac{1}{2}} \tag{37}$$

$$[\mathbf{M} + \theta\ \Delta t\ \mathbf{K}]\ \delta\phi^{n+1} = -\theta\ \Delta t\ \mathbf{K}\ \hat{\phi}^{n+1} + \Delta t\ \mathbf{F}^{n+\frac{1}{2}} \tag{38}$$

where $\mathbf{M}$ is a mass matrix, $\mathbf{K}$ is the stiffness matrix and $\mathbf{F}$ is the load vector of a standard FEM discretization. It must be noted that the matrix $[\mathbf{M} + \theta\ \Delta t\ \mathbf{K}]$ for $\mathbf{K} \neq \mathbf{K}(t)$ and $\Delta t = cte$ does not depend on the time, then it can be factorized at the beginning of the computation and used as a preconditioner afterward with a significant cpu-time reduction.

6.2 Analytic Diffusion 1D: Sinusoidal Signal

A diffusive-dominant problem with analytic solution is presented. It is solved using explicit, implicit and semi-implicit schemes for diffusion and using different Fo values.

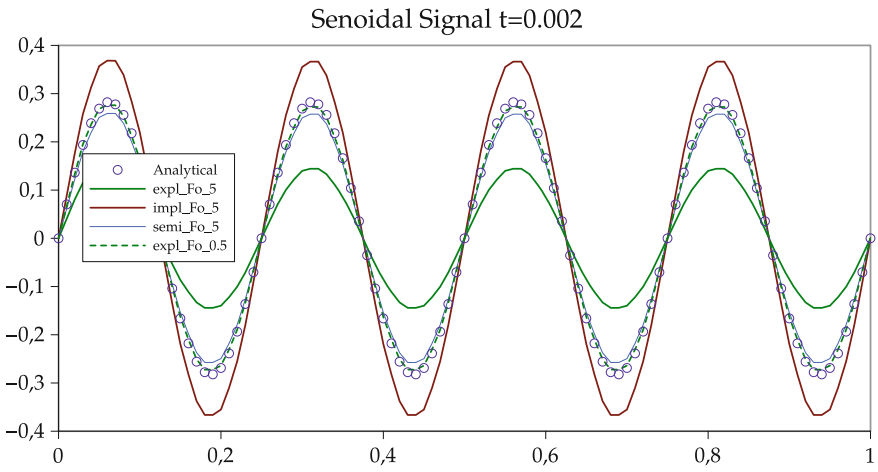

Fig. 14 Comparison between Explicit ($\theta = 0$), Implicit ($\theta = 1$) and Semi-Implicit ($\theta = 0.5$) schemes for diffusion in PFEM with the analytic solution at $t = 0.002$

The problem is:

$$\frac{\partial \phi}{\partial t} = \alpha \frac{\partial^2 \phi}{\partial^2 x} \quad \forall x \in (0, 1) \tag{39}$$

$$\phi(x = 0, t) = \phi(x = L, t) = 0; \quad t > 0 \tag{40}$$

$$\phi(x, t = 0) = \sin(kx); \quad t = 0 \tag{41}$$

with its analytic solution as:

$$\phi(x, t) = \sin(kx) e^{-(2\pi kx)^2 t} \tag{42}$$

where $\alpha = 1$ is chosen, the wave number of the problem is $k = \frac{2\pi}{\lambda} = 4$ and the mesh size is $\Delta x = 0.02$.

Figure 14 shows that using $Fo = 5$ more accurate results are obtained choosing a semi-implicit scheme. While explicit simulations are not accurate with $Fo = 5$, the solution with $Fo = 0.5$ is useful. These results show that temporal integration error in PFEM behaves as usual, i.e. semi-implicit or Crank Nicholson schemes are more accurate than first order explicit or fully implicit schemes. However for semi-implicit solvers we need to give up the idea of an explicit solver with severe consequences on the efficiency. Do not forget that one of the main goals in PFEM development is having a robust (stable) solver that allows to switch between accuracy and efficiency with greater freedom. But, due to the matrix for the implicit part of the computation of the diffusion can be factorized once at the beginning, it is possible to run simulations using big time-steps without loosing accuracy and efficiency. This idea of combining explicit schemes with efficient implicit schemes gives Fixed Mesh

PFEM its stronghold. Efficient implicit schemes means solving linear systems in an iterative way with good preconditioners.

6.3 A Pathological Case: Sinusoidal Signal Travelling

In this section, a pathological case is presented. The explicit calculation of the diffusion updates the state variable with the integral of the second derivative of the variable itself, i.e. the integral of the curvature. When certain conditions are accomplished, that integral vanishes and the explicit diffusion is null generating wrong new states. However, an implicit calculation of the diffusion solves that problem.

The problem consists on a sinusoidal wave transported by a field $\mathbf{v}$ with a non negligible diffusive term. The idea is to force numerical and physical parameters searching that the integral of the curvature of the function vanishes at each time-step.

If the length traveled by a particle is multiple of the length wave of the signal ($U\Delta t = m\lambda$), then $x_p^{n+1} = x_p^n + m\lambda$, hence, the rate of change of the variable (its curvature κ) will be null because

$$\kappa = \frac{d^2\phi}{dx^2} = \frac{d^2}{dx^2}[\sin(\frac{2\pi}{\lambda}x)] = C\ \sin(\frac{2\pi}{\lambda}x) = \mathbf{g}$$

and $\int_{x^n}^{x^{n+1}} \mathbf{g}\, dx = 0$.

This pathological situation has a very low probability and only is present in Lagrangian formulations where advection and diffusion are weakly coupled.

The problem to solve consists on:

$$\frac{\partial\phi}{\partial t} + U\frac{\partial\phi}{\partial x} = \alpha\frac{\partial^2\phi}{\partial^2 x} \quad \forall x \in (0,\infty) \tag{43}$$

$$\phi(x=0,t) = \sin(\omega t) \quad t > 0 \tag{44}$$

$$\phi(x,t=0) = \sin(\frac{2\pi}{\lambda}x) \quad t = 0 \tag{45}$$

where the advection and the diffusion can be analytically solved in an uncoupled way, allowing to determinate the decay of the signal.

$$\phi(x,t) = \sin(\frac{2\pi}{\lambda}\,[x-(x_0+Ut)])\, e^{-(\frac{2\pi}{\lambda})^2 t} \tag{46}$$

Using the parameters:

- $U = 5{,}000$
- $\alpha = 1$
- $L_x = 10$

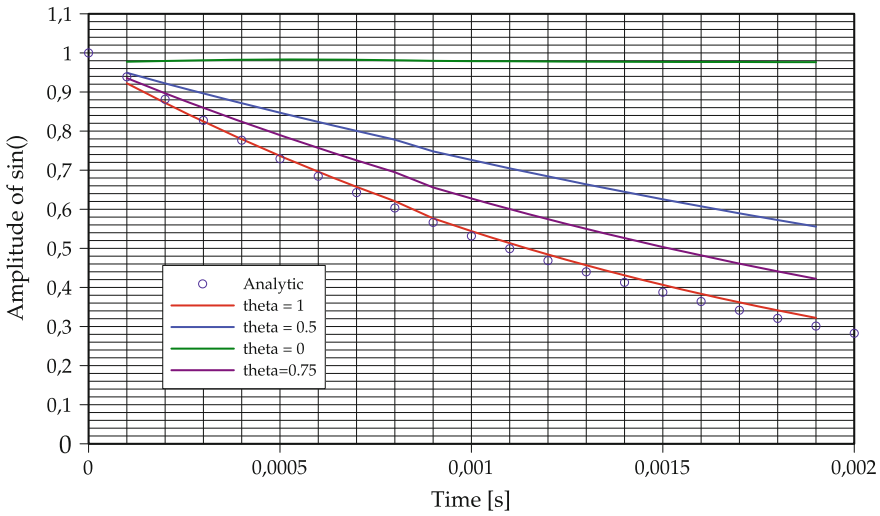

Fig. 15 Temporal evolution of sinusoidal amplitude

- $\lambda = 0.25$
- $\Delta x = 0.025$
- $\Delta t = 0.0001$ ($Fo = 0.16$).

In Fig. 15 results obtained with different values of θ are presented comparing with analytic decay. The most accurate simulation is using $\theta = 1$, using $\theta = 0$ decay is not observed and with other values for intermediate θ solutions are obtained. Finally, a corrective step of an erroneous explicit prediction does not ensure accurate results due to the bad performance of explicit schemes for this very special case.

It must be emphasized that the presented case rarely appears in non-academic problems, but it allows to demonstrate another reason to choose an implicit calculation of the diffusion instead of an explicit.

6.4 Implicit Calculation of the Viscous Diffusion

The theta method can also be adopted to calculate the viscous effects on incompressible flow problems. Again, this strategy allows to extend the maximum time-step without the limitation of the Fourier number. The expressions are similar to the scalar case presented in Eqs. (34)–(36), but replacing ϕ with $\mathbf{v}$ and $\mathbf{g}$ with $\nabla \cdot \tau_{\mathbf{v}}$ where $\tau_{\mathbf{v}} = \mu(\nabla \mathbf{v} + \nabla \mathbf{v}^T)$.

Finally, the algorithm for the implicit correction of the viscous stress tensor is presented in the following algorithm:

Algorithm 4 - Time Step PFEM Incompressible Flow with Implicit Correction of the viscous diffusion.

1. Calculate acceleration on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\ \nabla\cdot\tau_v\ d\Omega = -\int_\Omega \nabla\boldsymbol{N}\cdot(\mu\nabla\mathbf{v}^n)\ d\Omega + \int_\Gamma \boldsymbol{N}\ \nabla\mathbf{v}^n\cdot\eta\ d\Gamma$
   $\int_\Omega \boldsymbol{N}\ \nabla p^n\ d\Omega = -\int_\Omega \nabla\boldsymbol{N}\ p^n\ d\Omega + \int_\Gamma \boldsymbol{N}\ p^n\cdot\eta\ d\Gamma$
   $\mathbf{a}^n = \nabla\cdot\sigma = -\nabla p^n + \nabla\cdot\tau_v$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\ d\tau$
   $\hat{\mathbf{v}}_p^{n+1} = \mathbf{v}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{a}^n(\mathbf{x}_p^\tau) + \mathbf{f}^{n+\tau}\ d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\hat{\mathbf{v}}_j^{n+1} = \boldsymbol{\pi}(\hat{\mathbf{v}}_p^{n+1})$
5. Implicit correction:
   $\hat{\mathbf{v}}_j^{n+1} = \hat{\mathbf{v}}_j^{n+1} + \Delta t\ \theta\nabla\cdot\tau_j^{n+1}$.
6. Find the pressure value solving the Poisson equation system using FEM:
   $\rho\nabla\cdot\hat{\mathbf{v}}_j^{n+1} = \Delta_t\Delta[\delta p^{n+1}]$
7. Update the velocity value with the new pressure:
   $\rho\mathbf{v}_j^{n+1} = \rho\hat{\mathbf{v}}_j^{n+1} - \Delta_t(\nabla p^{n+1} - \nabla p^n)$
   $\rho\mathbf{v}_p^{n+1} = \rho\hat{\mathbf{v}}_p^{n+1} - \Delta_t\boldsymbol{\pi}^{-1}(\nabla p^{n+1} - \nabla p^n)$

The equation system for the implicit correction of the viscous diffusion can be solved using the same strategy as presented in (37) or (38). Also, for $\mu \neq \mu(t)$ and $\Delta t = cte$ the matrix does not depend on the time, then it can be factorized only once.

6.5 A Simple Test for a Diffusive Dominant Problem. The Mesh Size and the Time Step Dependency

6.5.1 Advective-Diffusive Transport of a Gaussian Hill

The transport of a Gaussian Hill problem was used to demonstrate the goodness of PFEM method to solve a scalar transport problem [18]. This case also made evident the pathology that explicit Eulerian approaches suffer in solving a pure advective transport problem with $CFL > 1$. The problem consists of a Gaussian hill signal used as initial condition transported with physical diffusion. The velocity field is a flow rotating around the center of a square domain. The Gaussian signal is displaced from the center of the domain at a certain radius and its shape makes the transported signal have a non-zero value in a limited region of the domain initially. The signal should be transported following circular path lines. Figure 16 shows the problem definition.

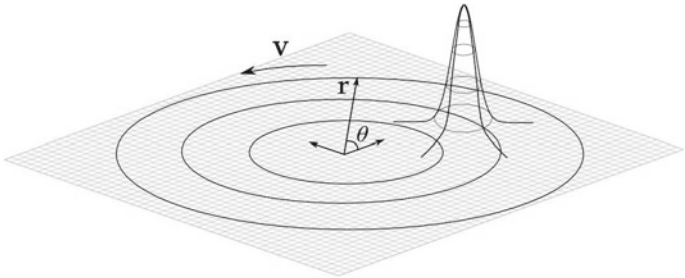

Fig. 16 Initial temperature distribution for the advective-diffusive transport problem

6.5.2 Problem Parameter Definition

This problem is taken from Donea and Huerta [4]. The initial condition is:

$$\phi(\mathbf{x},0) = \begin{cases} \frac{1}{4}(1+cos(\pi X))(1+cos(\pi Y)) & if\ X^2+Y^2 \leq 1 \\ 0 & otherwise \end{cases} \tag{47}$$

where $\mathbf{X} = (\mathbf{x} - \mathbf{x}_0)/\sigma$ with a boundary condition $\phi = 0$ for all the nodes lying on the boundary. The initial position of the center of the signal and its radius are $\mathbf{x}_0$ and σ respectively. In this example $\mathbf{x}_0 = (\frac{1}{6}, \frac{1}{6})$ and $\sigma = 0.2$ are taken. The velocity field corresponds to a rigid rotation with angular velocity $\omega = 2$, therefore $\mathbf{v}(\mathbf{x}) = (-\omega y, \omega x)$. The diffusivity chosen is $\alpha = 0.0001$.

Three different meshes were employed, all defined over a unit square $[-\frac{1}{2}, -\frac{1}{2}] \times [\frac{1}{2}, \frac{1}{2}]$. The coarse mesh called $M1$, with 30×30 quadrangular elements split in triangles. Another finer called $M2$, with 100×100 and finally the finest mesh called ($M3$), with 500×500, in order to define a reference solution for comparison playing the role of an almost exact solution.

6.5.3 FEM and PFEM Simulations

Next the results are compared. They were obtained using:

- an Eulerian FEM+SUPG code using different time integration θ schemes, where $\theta = 1$ is the first order implicit Backward-Euler and $\theta = 0.5$ is the second order Crank-Nicholson. They are labeled as `FEM-1order` and `FEM-2order` respectively.
- a Lagrangian code called PFEM using different number of particles per element seeded at the initial time step, (3,9,15,24), labeled as `PFEM-3p`, `PFEM-9p`, `PFEM-15p` and `PFEM-24p` respectively. For the seeding and the removing at least one particle for each subelement was fixed as the minimum limit and 20 as the maximum. The strategy for the diffusion treatment was fully implicit, i.e. X-IVS method was employed.

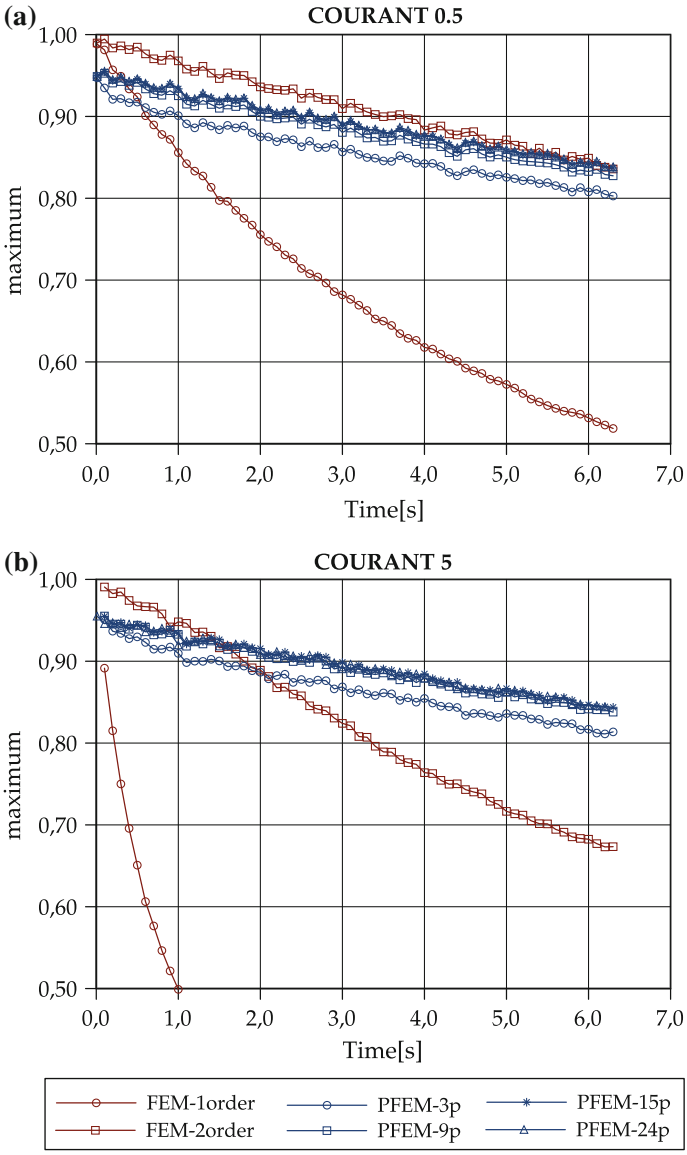

Fig. 17 Amplitude evolution on mesh M1 for $Co = 0.5$ (**a**) with $Co = 5$ (**b**). PFEM results are shown projected on the mesh

Figure 17 presents the evolution of the amplitude of the signal for the different simulations. It is confirmed again the fact that Eulerian simulations introduce a lot of numerical diffusion mainly due to the temporal scheme and the spatial stabilization.

It is noted that *PFEM* simulations start with an amplitude $\phi_{\max} \neq 1$, it is due to the projection of the maximum value on particles over the grid nodes. It is not an error because the particle information does not suffer for any type of numerical dissipation when is transported. The only error source is when the information is projected on the mesh for secondary computations. To confirm this, the signal amplitude over the particles may be viewed in Fig. 18. Here, another important result arises: the maximum value over the particles is independent of the number of particles initially seeded. This fact depends on the particle removing limits chosen during the computation and also on the projection operator design.

6.5.4 Simulation on a Finer Mesh *M*2

Figure 19 presents the evolution of the signal amplitude for different simulations. In this case the maximum on the mesh are almost the same as the maximum over the particles. It is due to the finer mesh involved, making the projection operation less diffusive.

7 Some Applications for More Complex Problems

This section finally end the paper showing some brief details about new promising and challenging applications of PFEM method. In some sense all theses applications present some sort of coupling problems. The first is a typical natural convection heat transfer problem in both, laminar and turbulent regime. The following example is the well known benchmark of turbulence modeling proposed by Rodi and Ferziger, a cube mounted on a channel floor and finally one example of multifluids flow, going towards the multiphase flow problems a very demanding need of the industry.

7.1 Thermal and Fluid Dynamics Coupled Problems

7.1.1 Natural Convection in a Square Cavity

The problem presented deals with the two dimensional flow with a Prandtl number $Pr = 0.71$ in a square cavity of side $H = 1$[m]. The boundary conditions for the momentum equation are non slip at all boundaries. Horizontal walls are isolated, and the vertical sides are at different temperatures $T_c < T < T_h$ ($\phi = T$ for natural convection problems). Figure 20 exhibits the geometry of the cavity. Simulations were carried out using a mesh of triangular elements with 100×100 nodes and refinement towards the walls. The wide range of Ra numbers (48) was obtained by a constant temperature difference of $\Delta T = 1\,K$ adjusting the thermal expansion coefficient β to supply the desired Ra.

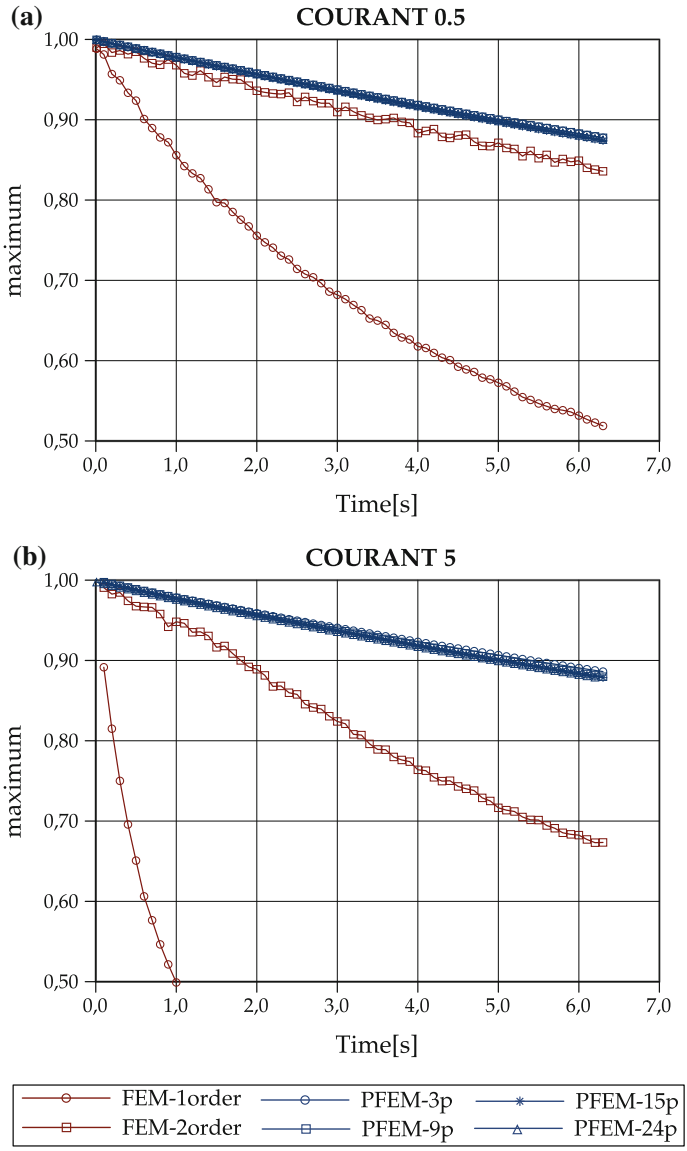

Fig. 18 Amplitude evolution on mesh M1 for $Co = 0.5$ (**a**) and for $Co = 5$ (**b**). PFEM results are shown on the particles, not projected on the mesh

$$Ra = \frac{g\beta H^3(\phi_h - \phi_c)}{\alpha\nu} \tag{48}$$

where α is the thermal diffusivity corresponding to air with the above mentioned Pr in standard temperature and pressure conditions.

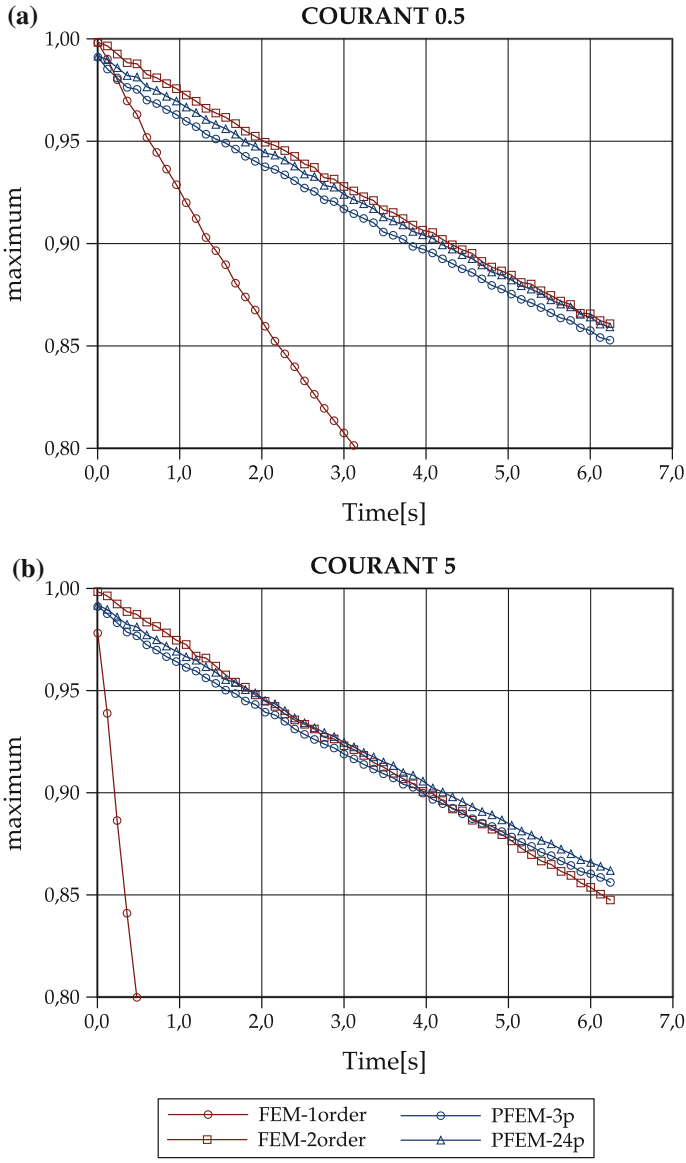

Fig. 19 Amplitude evolution on mesh $M2$ for $Co = 0.5$ (**a**) and for $Co = 5$ (**b**). PFEM results are shown on the mesh

7.1.2 Results and Discussion

This section provides a set of solutions at low Ra number. The quantities under study are the following:

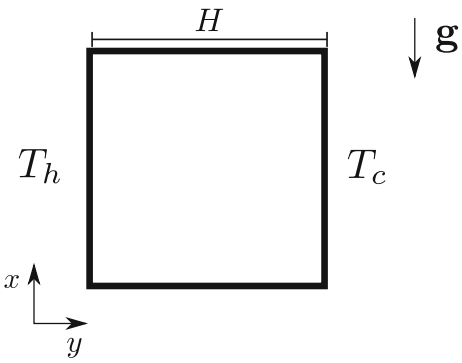

Fig. 20 Detail of cavity simulated, *left* wall at T_h, *right* wall at T_c, *top* and *bottom* walls are insulated

Table 1 Numerical solution for thermal square cavity with PFEM comparing with reference data

Ra	Data	PFEM2	Corzo [1]	Davis [2]
10^3	$u_{\max}$ (x = 0.5)	3.605	3.640	3.634
10^3	$y_{\max}$ (x = 0.5)	0.814	0.812	0.813
10^3	$v_{\max}$ (y = 0.5)	3.650	3.700	3.679
10^3	$x_{\max}$ (y = 0.5)	0.183	0.177	0.179
10^4	$u_{\max}$ (x = 0.5)	15.982	16.281	16.182
10^4	$y_{\max}$ (x = 0.5)	0.824	0.822	0.823
10^4	$v_{\max}$ (y = 0.5)	19.378	19.547	19.509
10^4	$x_{\max}$ (y = 0.5)	0.116	0.123	0.120
10^6	$u_{\max}$ (x = 0.5)	64.483	64.558	65.330
10^6	$y_{\max}$ (x = 0.5)	0.845	0.851	0.851
10^6	$v_{\max}$ (y = 0.5)	218.054	221.572	216.750
10^6	$v_{\max}$ (y = 0.5)	0.037	0.067	0.039

$u_{\max}(\frac{1}{2})$: The maximum horizontal velocity on the vertical mid-plane of the cavity (together with its location).

$v_{\max}(\frac{1}{2})$: The maximum vertical velocity on the horizontal mid-plane of the cavity (together with its location).

Table 1 shows PFEM results for $Ra = 10^3$, 10^4 and 10^6 compared with the [1, 2] solutions. Excellent agreement to experimental data in both results for momentum and energy equations prove the accuracy of this approach for this low Ra number range. The horizontal velocity component in the vertical mid-plane is shown in Fig. 21. Here is worthy to note that when Ra number increases the boundary layer becomes thinner and the maximum values in the velocity get closer to the walls. Finally Fig. 22 presents the temperature profiles for the three cases.

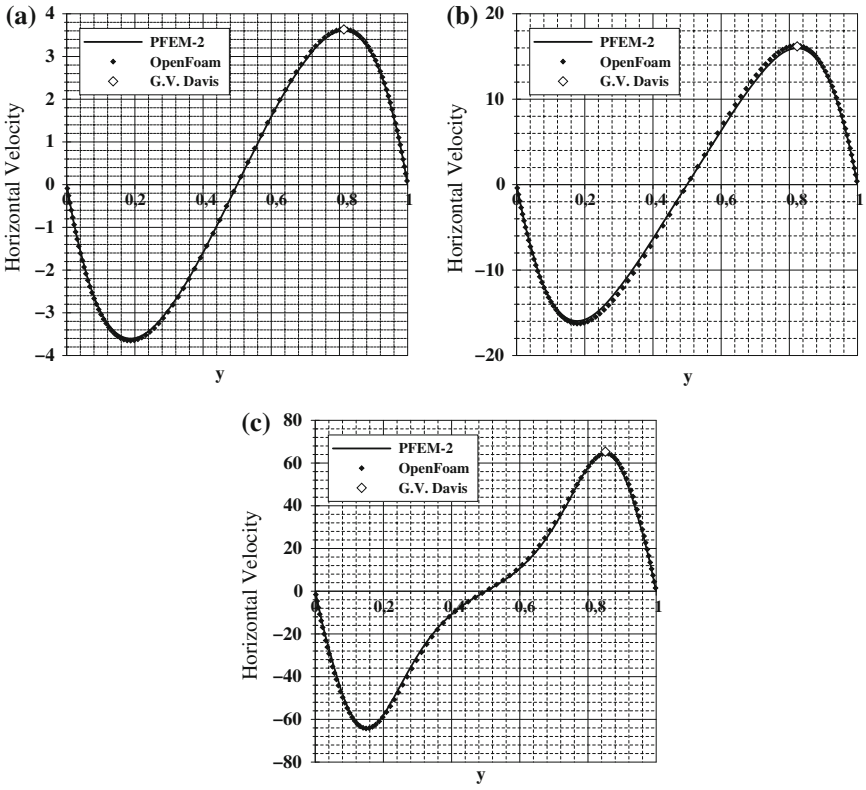

Fig. 21 Horizontal velocity profiles at x mid-plane to **a** $Ra = 10^3$, **b** $Ra = 10^4$ and **c** $Ra = 10^6$

7.1.3 Natural Convection in a Cubic Cavity

The schematic model for the problem is shown in Fig. 23. The cubic cavity is one meter length with an aspect ratio of unity and is filled with air as working fluid. The Prandtl number is fixed at $Pr = 0.71$. All surrounding walls are rigid and impermeable. The vertical walls located at $x = 0$ and $x = 1$ are retained to be isothermal but at different temperatures of T_h and T_c, respectively. The buoyancy force due to gravity works downwards (i.e., in negative z-direction).

7.1.4 Results and Discussion

For the present range of Ra numbers, solutions were obtained on a mesh with 81,000 tetrahedral elements and around of eighteen thousand nodes, and with refinement towards the walls. The following characteristic quantities are presented:

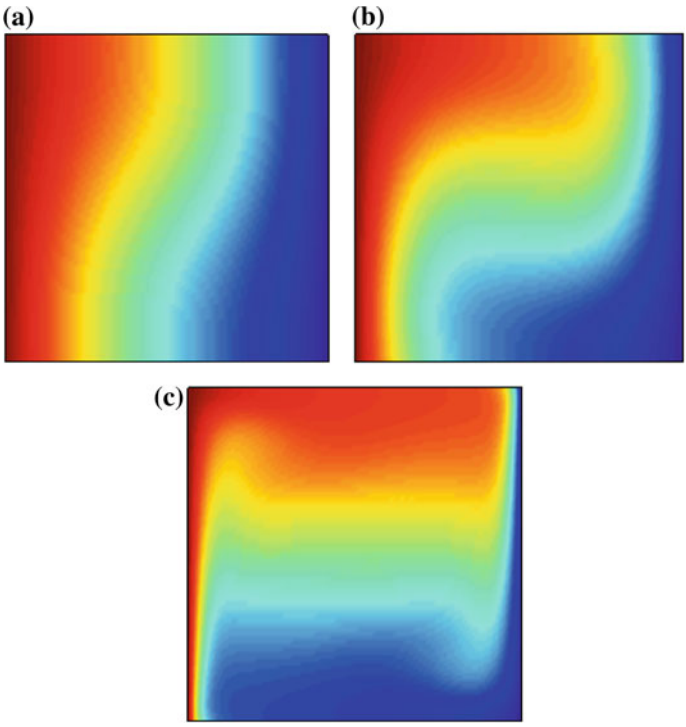

Fig. 22 Temperature field ϕ to **a** $Ra = 10^3$, **b** $Ra = 10^4$ and **c** $Ra = 10^6$

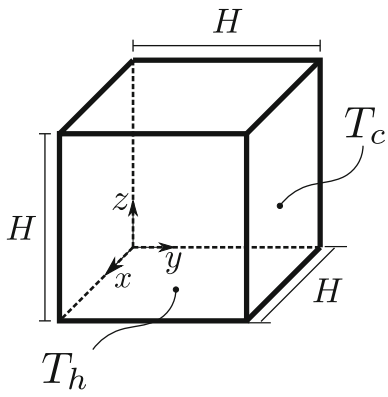

Fig. 23 Schematic model for the natural convection in a cubical cavity

$u_{\max}(\frac{1}{2})$: The maximum horizontal velocity for x-direction on center line (x = 0.5, y = 0.5) of the cavity and its location.

$w_{\max}(\frac{1}{2})$: The maximum vertical velocity for z-direction on center line (y = 0.5, z = 0.5) of the cavity and its location.

Table 2 Numerical solution for thermal cubic cavity with PFEM comparing with reference data

Ra	Data	PFEM	Wakashima [24]	Fusegi [5]
10^4	$u_{\max}(x = y = 0.5)$	0.1978	0.1989	0.2013
10^4	$z_{\max}(x = y = 0.5)$	0.8460	0.8250	0.8167
10^4	$w_{\max}(y = z = 0.5)$	0.2190	0.2211	0.2252
10^4	$x_{\max}(y = z = 0.5)$	0.1260	0.1253	0.1167
10^5	$u_{\max}(x = y = 0.5)$	0.1409	0.1423	0.1468
10^5	$z_{\max}(x = y = 0.5)$	0.8460	0.8500	0.8547
10^5	$w_{\max}(y = z = 0.5)$	0.2359	0.2407	0.2471
10^5	$x_{\max}(y = z = 0.5)$	0.0680	0.0751	0.0647
10^6	$u_{\max}(x = y = 0.5)$	0.0766	0.0813	0.0842
10^6	$z_{\max}(x = y = 0.5)$	0.8570	0.8500	0.8557
10^6	$w_{\max}(y = z = 0.5)$	0.2897	0.2382	0.2588
10^6	$x_{\max}(y = z = 0.5)$	0.0280	0.0500	0.0331

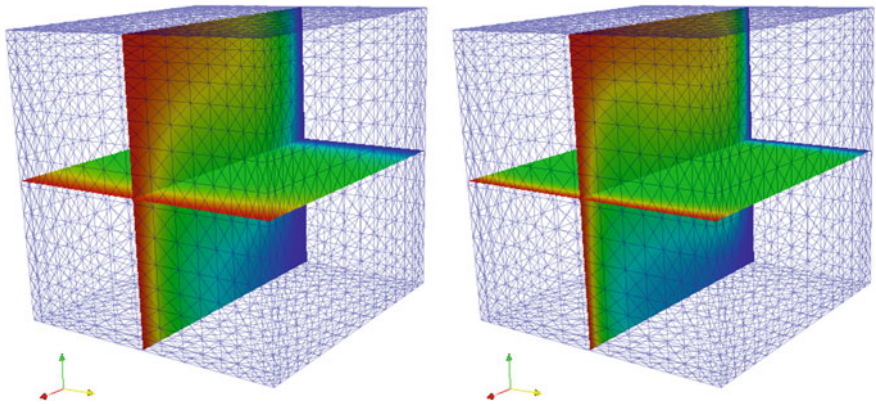

Fig. 24 Mesh with slices of section at mid-planes $y = 0.5$ and $z = 0.5$

Table 2 shows PFEM results for $Ra = 10^4$, 10^5 and 10^6 compared with the [5, 24] solutions. Finally Fig. 24 presents a wireframe of the mesh used with slices of section at mid-planes $y = 0.5$ and $z = 0.5$ respectively.

7.2 Turbulent Flows

7.2.1 Wall Mounted Cube Simulation

Turbulent flows around three-dimensional obstacles are common in nature and occur in many applications including flow around tall buildings, vehicles and computer chips. Understanding and predicting the properties of these flows are necessary for

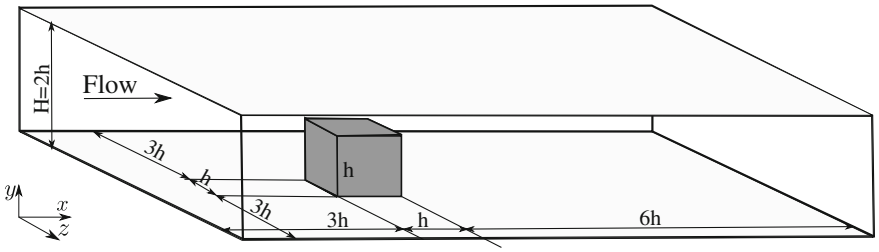

Fig. 25 Geometry for the flow around a cube obstacle

safe, effective and economical engineering designs. Experimental techniques are expensive and often provide data that is not sufficiently detailed. With the advent of supercomputers it has become possible to investigate these flows using numerical simulations.

In this paper the simulation of the turbulent flow around a cube obstacle is presented. This test is known as flow over a wall mounted cube, and it was analyzed experimentally by Martinuzzi and Tropea [17] and numerically by Sha and Ferziger [21], Lakehal and Rodi [15], and Rodi et al. [20] among others. Flow around a cube exhibits characteristics as three dimensionality of the mean flow, separation and large-scale unsteadiness. Quantitative results of this flow are scarce, then flows patters are exhaustively analyzed and compared.

The geometry of the problem is presented in the Fig. 25.

Computational Modeling

In this work, the numerical method used is the Particle Finite Element Method (PFEM) with Large Eddy Simulation (LES) for turbulence modeling. The Sub-Grid Scale (SGS) model used is the Static Smagorinsky model. Regarding to the computational domain, the problem was solved using two grids: the first one is a relatively coarse grid of one million of tetrahedral elements (refined towards the cube and behind it), with a mesh-size of $\delta_h = h/25$ over the cube. On the other hand, the second grid has the same kind of refinement but it has around four million of tetrahedral elements, with $\delta_h = h/40$.

The spanwise boundary condition is slip, the spanwise width is $7\,h$, assuring that blockage effects are small. In the streamwise direction, inflow-outflow boundary conditions are used. A parabolic flow with some perturbations is used at the inlet and fixed pressure condition is applied at the exit. The streamwise length of the domain is $10\,h$.

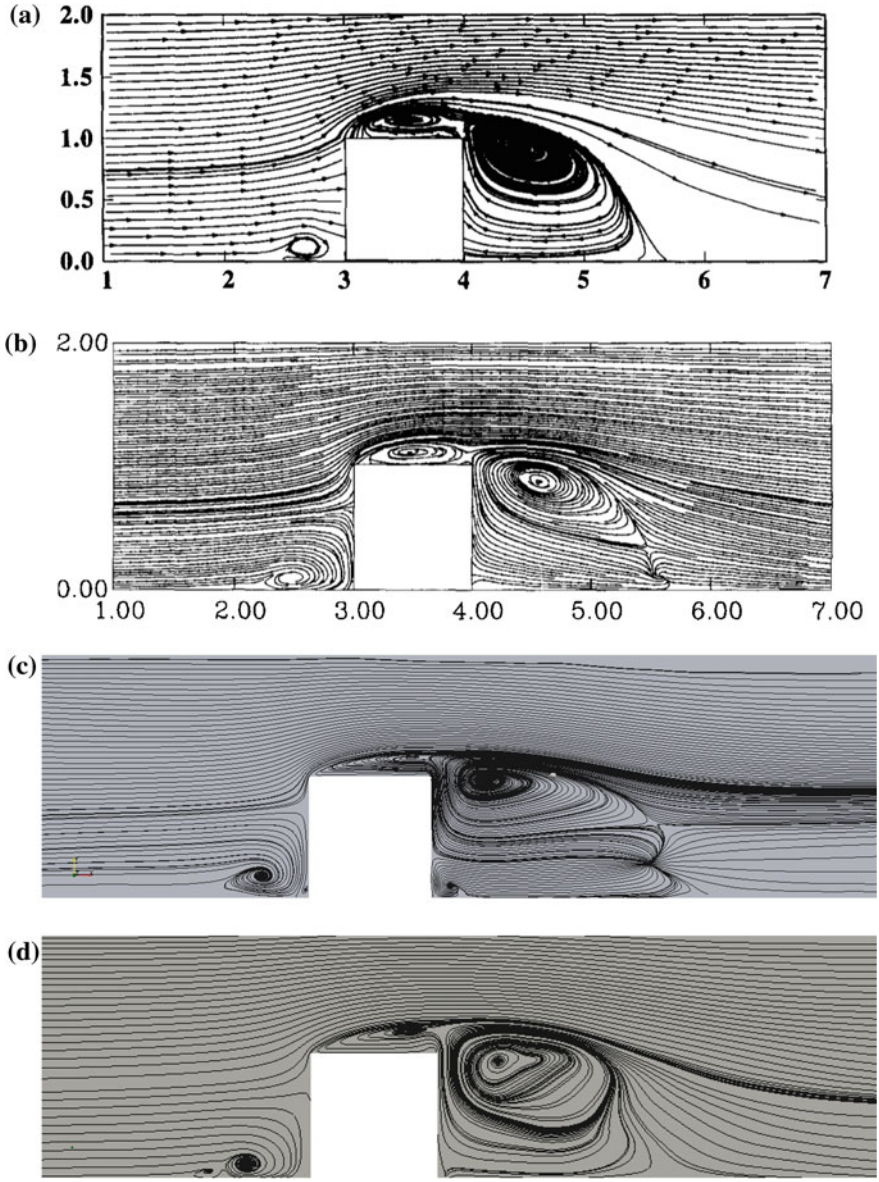

Fig. 26 The streamlines on the symmetry plane at $Re = 40{,}000$. **a** shows the experimental result of Martinuzzi and Tropea [17], **b** result from LES simulation on [21], and **c** and **d** presents the results of PFEM using a coarse and finer mesh respectively

Summary of Results

Large eddy simulations were performed at $Re = 40{,}000$. Figure 26 shows a comparison of time-averaged streamlines on the symmetry plane. The overall prediction of the separation region on the roof and behind the obstacle is quite good even using coarse grids. Shah and Ferziger commented that in its simulations the stagnation point was located high on the front face, and in this work could be arrived the same conclusion. Fluid striking the body above it goes over the obstacle and using the finer mesh we can find a solution where it reattach on the roof, something that Shah could not. Using the finer mesh, the rear recirculation region is not closed, but streamlines originating upstream of the obstacle do not enter this region; fluid enters the rear recirculation region from sides. Near the top of the recirculation region we find the head of the arch vortex. Results using coarse mesh are not accurate, mainly behind the obstacle.

Figure presents the time-averaged streamlines on the floor of the channel. The streamline patterns are consistent with those observed by Martinuzzi and Tropea [17]. These streamlines, which may be viewed as skin friction lines, show the complexity of this 3-D flow. On the reference [21], the primary separation occurs at a saddle point located about one obstacle height (1.05 h) ahead of the obstacle (experimental value = 1.026), whereas PFEM simulation reach approximately ($0.89\,h$) with coarse grid and ($0.92\,h$) with the finer grid. The separation region wraps around the obstacle and forms a strong horseshoe vortex. The converging and diverging streamlines that mark the extent of this vortex are regions of strong upwash and downwash. This horseshoe is better represented by PFEM using the finer mesh, whereas with the coarse mesh the streamlines are too much closed behind the obstacle. Instantaneous pictures (not presented here) of the flow show that the horseshoe vortex is, in fact, highly intermittent; an intact structure is almost never found in these snapshots. The mean flow on the side faces is entirely reversed. In Shah and Ferziger, the primary reattachment length of $1.65\,h$ agrees well with the experimental value of $1.61\,h$, however PFEM reattachment is found in $1.8\,h$.

In the work of Shah and Ferziger [21], it is said that both the primary separation point ahead of the obstacle and the rear reattachment points are singular points (zero skin friction) where the so-called separation lines begin and end. Also, they comment that the owl-face shaped streamlines in the rear recirculation zone of the obstacle correspond to the base of the arch vortex. The arch vortex is formed by quasi-periodic vortex shedding from the upstream vertical corners that resembles a von Karman street. This intact arch vortex exists only in the mean flow and is an artifact of averaging and PFEM can reproduce only approximately this behavior, and strangely with a coarse mesh the result are more accurate. Must be noticed that both grids are not good enough near the floor of the channel, then a better refinement is required to reach the same quality of results as Shah and Ferziger.

Efficiency

In this section the scalability of the current implementation of PFEM is presented. The above mentioned test, using the finer grid, was carried out over a Infiniband

Table 3 CPU-times comparison in seconds between different PFEM2 algorithms and OpenFOAM for one, two and four cores

Cores	$1x$ (s)	$2x$ (s)	$4x$ (s)
OpenFOAM	754	402	286
PFEM2 moving mesh (CIMNE)	484	371	326
PFEM2 fixed mesh (CIMNE)	284	179	138
PFEM2 fixed mesh (CIMEC)	330	176	99

interconnected cluster, which has dual socket nodes with Intel Xeon E5-2600 CPUs and 64 Gb RAM. The interconnection is with IB-QDR 40 Gbps (Fig. 27).

Figure 28 presents the scalability of each PFEM stage and of the entire simulation using an Eulerian weighting strategy, obtaining approximately the same number of degrees of freedom in each partition. Could be noted that the efficiency of the Infiniband cluster is good enough also running with 32 cores, reaching a global $S_{32} \approx 26x$. Using more cores the efficiency decays because there is not enough work for each process to overweight the communication time.

7.3 Multifluids

7.3.1 Sloshing Test

In this section a comparison with the results of the sloshing test is presented. For the experiment, the same mesh and configuration than that presented in Idelsohn et al. [9] have been used (Figs. 29 and 30).

Table 3 shows the computational time necessary to simulate 1 sec. in an Intel(R) Core(TM) i7-3820 CPU 3.60 GHz with OpenFOAM and PFEM2 versions of the International Center for Numerical Methods in Engineering (CIMNE).

On the other hand, the test of the implementation presented in this paper was executed in an Intel(R) Core(TM) i5-3230M CPU 2.60 GHz. To match the heterogeneous platforms a benchmarking factor $\frac{4,007}{9,010}$ (extracted for the web-page http://cpubenchmark.net/high_end_cpus.html) is used, and the final values are presented in the table.

The reported values evidence that, for the settings described, *PFEM* with fixed mesh is more than $2\times$ faster than OpenFOAM.

7.3.2 Dam-Break Test

In this section a comparison with the results of a dam-break test is presented. For the experiment, the same mesh and configuration which is presented in Idelsohn et al. [9] have been used.

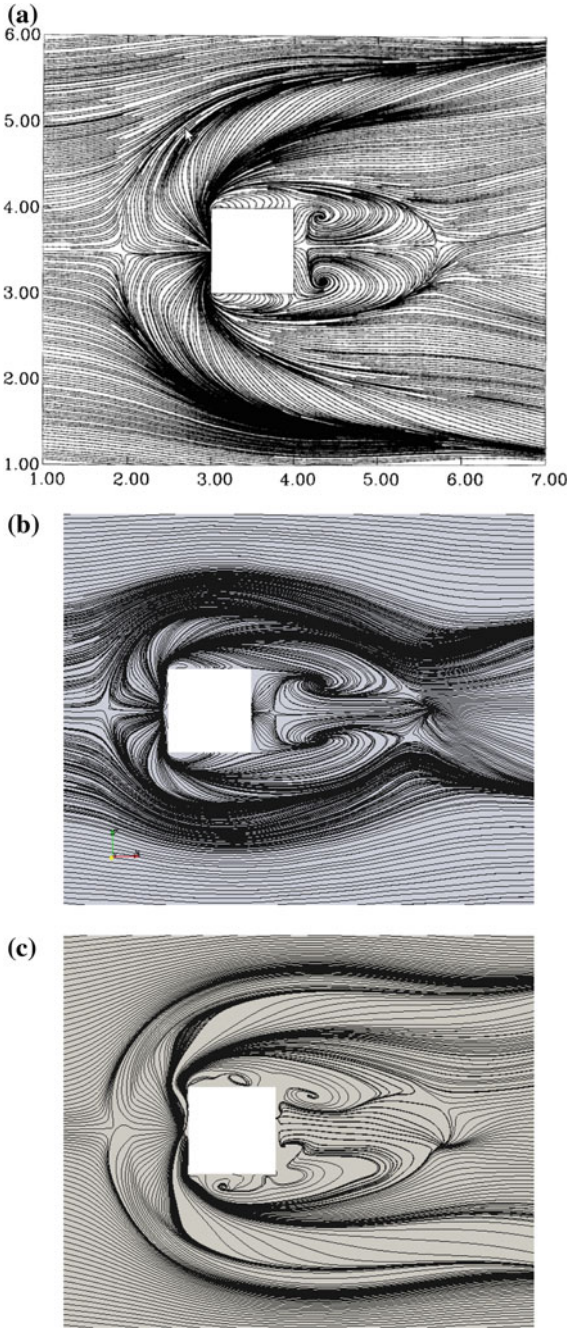

Fig. 27 The streamlines in a plane near to the floor at $Re = 40{,}000$. **a** shows the numerical result of Shah and Ferziger [21], and **b** and **c** present the results of PFEM using a finer mesh respectively

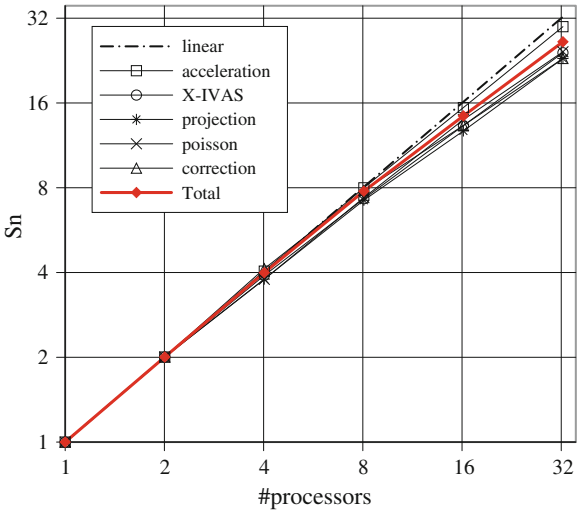

Fig. 28 Speed-up over an Infiniband cluster. Case: flow around a mounted cube in 3d

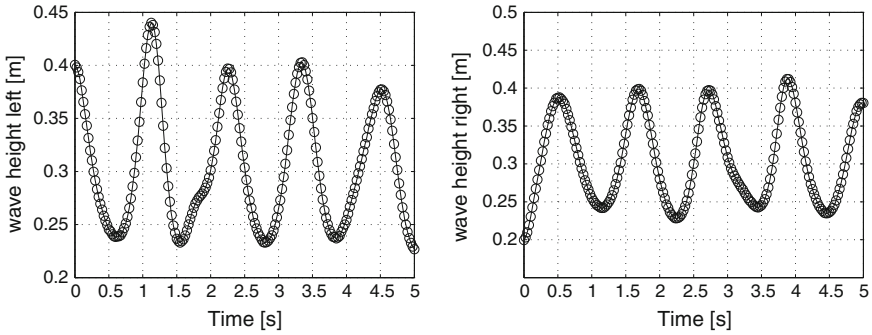

Fig. 29 Interface relative height at the vertical walls (*left side* and *right side*) for *PFEM* fixed mesh

Figure 31 presents snapshots of the simulation. Comparing with the results obtained in [9], good agreement both in the shape of the free surface and in the time evolution with experimental and OpenFOAM results can be observed. The current version is using Courant number larger than 10 (maximum 15), without present any difficulty for this large time-step.

The CPU-Time time needed to simulate 1 s of real time is 274.75 s (running in serial mode). Idelsohn et al. reports 278 s for the CIMNE pfem2 fixed mesh version and 473 s with OpenFOAM.

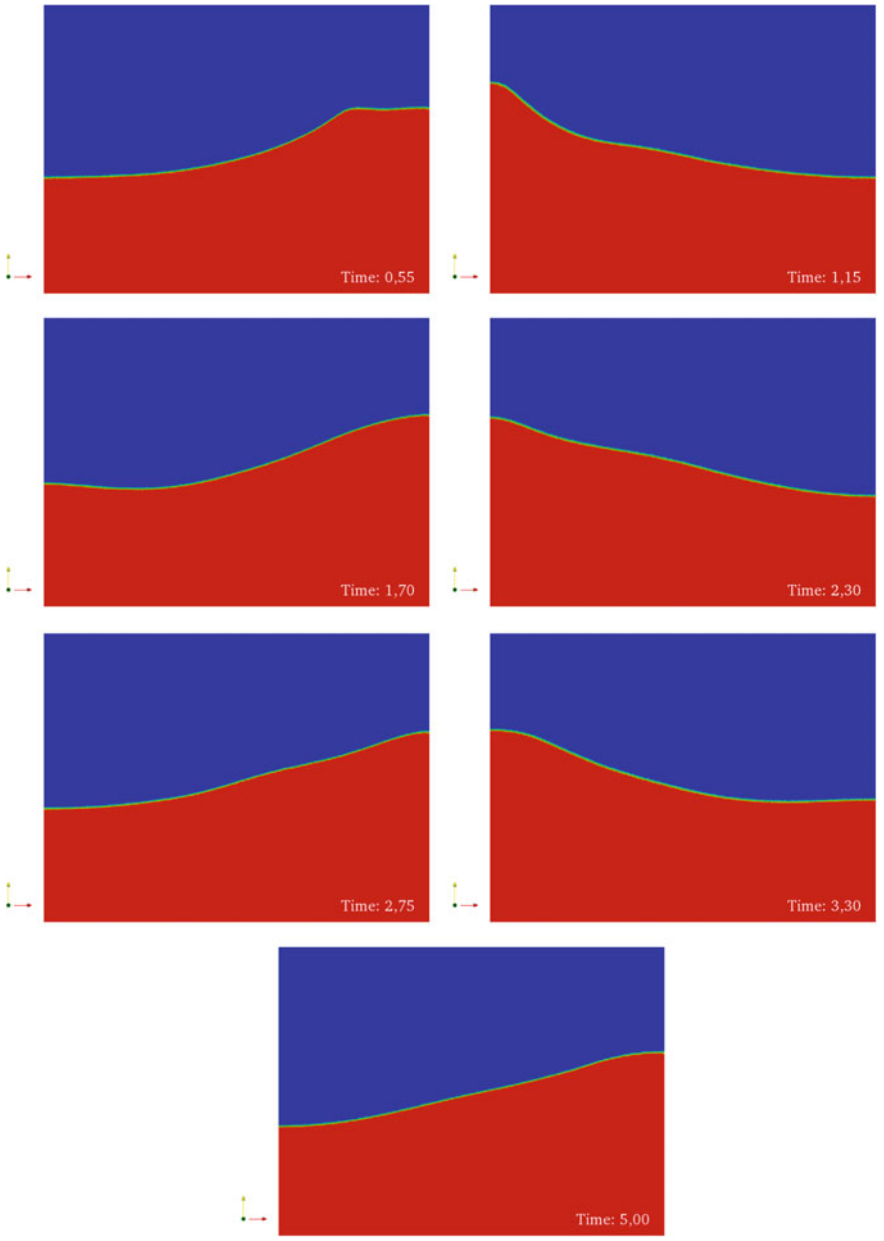

Fig. 30 From *left* to *right* and *top* to *bottom*: sloshing of two immiscible fluids with a large jump in the density: snapshots at different time steps (t = 0.55, 1.15, 1.7, 2.3, 2.75, 3.35 and 5 s.)

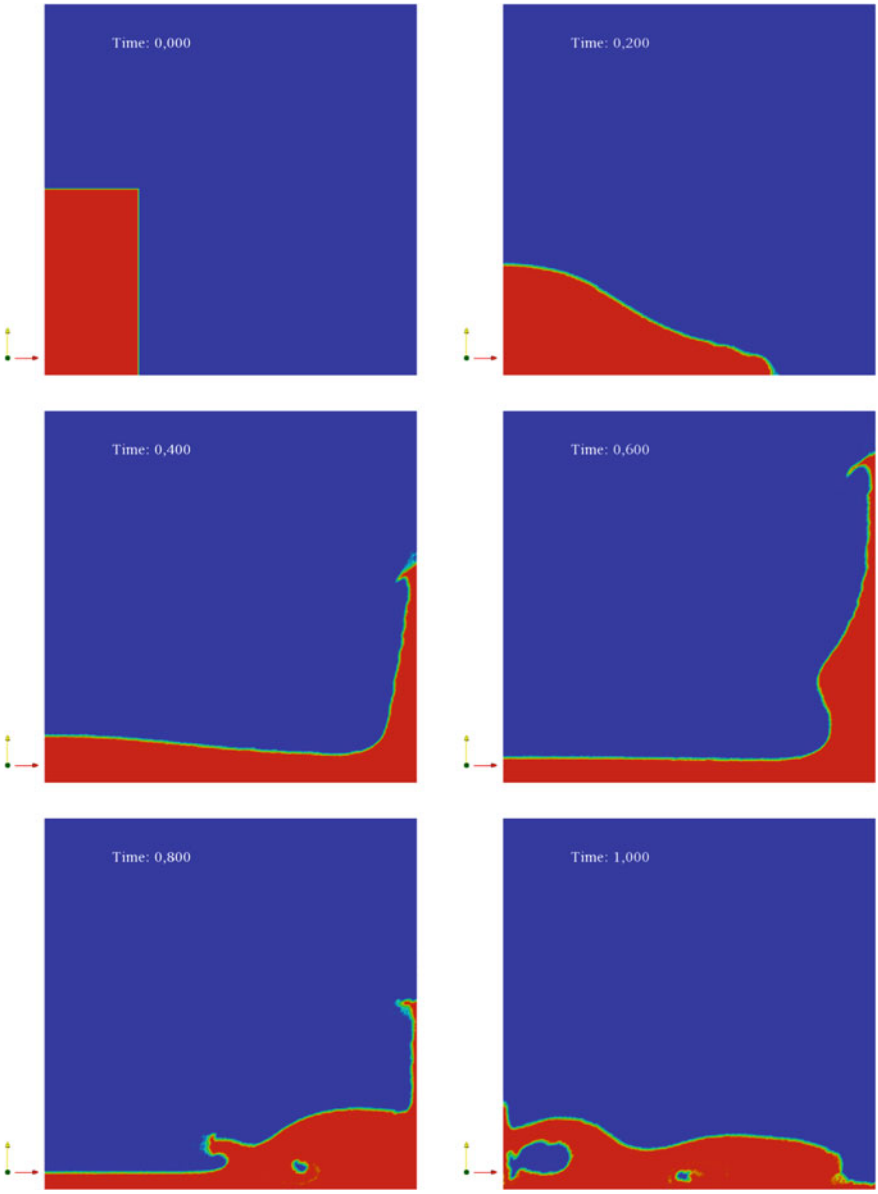

Fig. 31 From *left* to *right* and *top* to *bottom*: snapshots of the dam break without obstacle at t = 0, 0.2, 0.4, 0.6, 0.8 and 1 s

8 Conclusions

In this paper a review and the present developments of PFEM are presented. In recent years much effort has been devoted to improve the performance of this method in order to make it competitive with the rest of the solvers mostly used in computational mechanics. Not only that, but with recent findings that have emerged is thought to be on the gates of a paradigm shift in the way of performing the simulations, especially considering that the community is demanding of methods that are commensurate with the needs of engineering design.

While this paper does not delve into the numerical analysis it establishes the basis to do so in the next few years with the target to demonstrate mathematically the goodness of the Lagrangian methods of this type in front of the very commonly used Eulerian methods.

Finally the last goal has been to show that in addition to the well-known virtues that owns the method to resolve problems with heterogeneous flows, it is also possible to implement complex homogeneous flows, as in the case of turbulence and cases with thermal coupling.

Acknowledgments This work was partially supported by the European Research Council under the Advanced Grant: ERC-2009-AdG Real Time Computational Mechanics Techniques for Multi-Fluid Problems. Norberto Nigro and Juan Gimenez want to thanks to CONICET,Universidad Nacional del Litoral and ANPCyT for their financial support (grants PICT 1645 BID (2008), CAI+D 65-333 (2009)). Also thanks to Santiago Marquez Damian for their invaluable assistance in show the goodness of PFEM vis–vis other solvers available, of whom Santiago is an expert user. To Eugenio Oñate and CIMNE for their unconditional support and his teachings throughout his scientific life. To Pedro Morin, Marta Bergallo for interesting mathematics discussions and to Nestor Calvo and Pablo Novara for sharing some discussions about mesh generation and computational geometry.

References

1. Corzo S, Marquez Damian S, Nigro N (2011) Numerical simulation of natural convection phenomena. Mecánica Computacional XXX:277–296
2. De Vahl Davis G (1983) Natural convection of air in a square cavity: a benchmark numerical solution. Int J Num Meth Fluids 3:249–264
3. Del Pin F (2003) The meshless finite element method applied to a lagrangian particle formulation of fluid flows. Ph.D. Thesis, Facultad de Ingeniería y Ciencias Hídricas (FICH) Instituto de Desarrollo Tecnológico para la Industria Química (INTEC) Universidad Nacional del Litoral
4. Donea J, Huerta A (1983) Finite element method for flow problems. Wiley, Chichester
5. Fusegi T, Hyun J, Kuwahara K, Farouk B (1991) A numerical study of three-dimensional natural convection ina differentially heated cubical enclosure. Int J Heat Mass Transfer 34:1543–1551
6. Gimenez JM, Nigro NM (2011) Parallel implementation of the particle finite element method. Mecánica Computacional XXX:3021–3032
7. Gimenez JM, Nigro NM, Idelsohn SR (2012) Improvements to solve diffusion-dominant problems with pfem-2. Mecánica Computacional XXXI:137–155
8. Hryb D, Cardozo M, Ferro S, Goldschmidt M (2009) Particle transport in turbulent flow using both lagrangian and eulerian formulations. Int Cummun Heat Mass Transfer 36:451–457

9. Idelsohn S, Marti JM, Becker P, Oñate E (2014) Analysis of multi-fluid flows with large time-steps using the particle finite element method. Int J Num Meth in Fluids (in press)
10. Idelsohn S, Nigro NM, Limache A, Oñate E (2012) Large time-step explicit integration method for solving problems with dominant convection. Comput Methods Appl Mech Eng 217–220:168–185
11. Idelsohn SR, Nigro NM, Gimenez JM, Rossi R, Marti J (2013) A fast and accurate method to solve the incompressible navier-stokes equations. Eng Comput 30(2):197–222
12. Idelsohn SR, Oñate E, Calvo N, Del Pin F (2003) The meshless finite element method. Int J Num Meth Eng 58(6):893–912
13. Idelsohn SR, Oñate E, Del Pin F (2004) The particle finite element method a powerful tool to solve incompressible flows with free-surfaces and breaking waves. Int J Numer Meth 61:964–989
14. Jasak H (1996) Error analysis and estimation for the finite volume method with applications to fluid flows. Ph.D. Thesis, London
15. Lakehal D, Rodi W (1997) Calculation of the flow past a surface-mounted cube with two-layer turbulence models. J Wind Eng Ind Aerodyn 67:65–78
16. Leveque R (2002) Finite volume methods for hyperbolic problems, 1st edn. Cambridge University Press, Cambridge
17. Martinuzzi R, Tropea C (1993) The flow around surface-mounted, prismatic obstacles placed in a fully developed channel flow. J Fluids Eng 115:85–92
18. Nigro N, Gimenez J, Limache A, Idelsohn S, Oñate E, Calvo N, Novara P, Morin P (2011) A new approach to solve incompressible navier-stokes equation using a particle method. Mecánica Computacional XXX
19. Oñate E, Idelsohn SR, Del Pin F, Aubry R (2004) The particle finite element method, an overview. Int J Comput Meth 1:267–307
20. Rodi W, Ferziger J, Breuer M, Pourquie M (1997) Status of large eddy simulation: results of a workshop. Trans ASME J Fluid Eng 119:248–262
21. Shah KB, Ferziger JH (1997) A fluid mechanicians view of wind engineering: large eddy simulation of flow past a cubic obstacle. J Wind Eng Ind Aerodyn 67&68:211–224
22. Sklar DM, Gimenez JM, Nigro NM, Idelsohn SR (2012) Thermal coupling in particle finite element method - second generation. Mecánica Computacional XXXI:4143–4152
23. Stam J (1999) Stable fluids. In: SIGGRAPH 99 Conference Proceedings, Annual Conference Series, pp 121–128
24. Wakashima S, Saitoh T (2004) Benchmark solutions for natural convection in a cubic cavity using the high-order time-space method. Int J Heat Mass Transfer 47:853–864