Emerald Article: A fast and accurate method to solve the incompressible Navier-Stokes equations

Sergio Rodolfo Idelsohn, Norberto Marcelo Nigro, Juan Marcelo Gimenez, Riccardo Rossi, Julio Marcelo Marti

A fast and accurate method to solve the incompressible Navier-Stokes equations

Sergio Rodolfo Idelsohn
Intitució Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Spain and
Centro Internacional de Métodos Numéricos en Ingeniería (CIMNE), Polytechnic University of Catalonia (UPC), Barcelona, Spain

Norberto Marcelo Nigro and Juan Marcelo Gimenez
Centro Internacional de Métodos Computacionales en Ingeniería, CIMEC-INTEC-CONICET-UN, Santa Fe, Argentina, and

Riccardo Rossi and Julio Marcelo Marti
Centro Internacional de Métodos Numéricos en Ingeniería (CIMNE), Polytechnic University of Catalonia (UPC), Barcelona, Spain

Received 1 March 2012
Revised 27 July 2012
3 September 2012
Accepted 4 December 2012

Abstract

Purpose – The purpose of this paper is to highlight the possibilities of a novel Lagrangian formulation in dealing with the solution of the incompressible Navier-Stokes equations with very large time steps.

Design/methodology/approach – The design of the paper is based on introducing the origin of this novel numerical method, originally inspired on the Particle Finite Element Method (PFEM), summarizing the previously published theory in its moving mesh version. Afterwards its extension to fixed mesh version is introduced, showing some details about the implementation.

Findings – The authors have found that even though this method was originally designed to deal with heterogeneous or free-surface flows, it can be competitive with Eulerian alternatives, even in their range of optimal application in terms of accuracy, with an interesting robustness allowing to use large time steps in a stable way.

Originality/value – With this objective in mind, the authors have chosen a number of benchmark examples and have proved that the proposed algorithm provides results which compare favourably, both in terms of solution time and accuracy achieved, with alternative approaches, implemented in in-house and commercial codes.

Keywords Fluids, Flow, Fluid dynamics, Navier-Stokes equations, Particle methods, Large time-steps, Incompressible fluid flows, Updated Lagrangian formulations, Real time CFD

Paper type Research paper

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 thank CONICET, Universidad Nacional del Litoral and ANPCyT for their financial support (grants PICT 1645 BID (2008), CAI + D 65-333 (2009)). Riccardo Rossi acknowledges the support of the MEC through the eDams project (BIA2010-21350-C03-00).

Engineering Computations: International Journal for Computer-Aided Engineering and Software
Vol. 30 No. 2, 2013
pp. 197-222
© Emerald Group Publishing Limited
0264-4401
DOI 10.1108/02644401311304854

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 a 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 (Donea and Huerta, 2003). Despite such large effort, the solution of large 3D problems involving free-surfaces, moving interfaces or complex fluid-structure interaction problems is still very demanding.

Lagrangian formulations have been used to determine the position of the free surface in fixed mesh approximations. Over the years, several proposals were made to use the Lagrangian approach to solve the Navier-Stokes equations. For compressible flows the main references may be found in the particle-in-cell methods (PIC or FLIP) (Harlow, 1964; Brackbill *et al.*, 1988) or the material point methods (MPM) (Sulsky and Kaul, 2004), whereas for incompressible flows the main reference are in the particle finite element method (PFEM and PFEM-2) (Idelsohn *et al.*, 2004, 2008; Oñate *et al.*, 2008). The difference between the MPM and the PIC methods with respect to the particle finite element methods (PFEM and PFEM-2) (Idelsohn *et al.*, 2004, 2008a, b, 2012; Oñate *et al.*, 2008; Larese *et al.*, 2008) is that in the first ones the particles are material points, with an associated mass, which is preserved during the calculations. On the contrary in the PFEMs, the particles are immaterial points that are used only to evaluate the convective terms.

The advantages of these solutions in solving problems featuring free-surfaces or multi-fluids with complicated internal interfaces have been demonstrated extensively (Idelsohn *et al.*, 2004, 2008a, b, 2012; Oñate *et al.*, 2008; Larese *et al.*, 2008). In general, these formulations are more expensive than Eulerian alternatives when applied to the solution of homogeneous flows. They justify their popularity due to their efficiency in solving problems where standard Eulerian formulations are inaccurate or cannot be applied.

When attempting to classify Navier-Stokes solvers it is important to take into account the level of locality of the information needed. One may define as "implicit" a solution strategy in which a change in the solution in any part of the domain can potentially influence the solution in any other part of it. "Explicit" strategies can hence be understood as strategies in which the solution at a point, within a time step, is only influenced by a portion of the domain around the point. A fundamental feature of implicit solver is thus that they enforce a strong coupling between time and space, while in explicit solvers this coupling is somewhat relaxed. In the second class we include not only the explicit time integration schemes but also methods that lead to a linear system of equations that may be factorized once and integrated in time with the same factorized matrix.

Even though implicit time integration schemes are often preferred in the literature against explicit ones, the latter may be advantageous on recent hardware and in particular on general purpose graphic processor units (GPGPU) (Mossaiby *et al.*, 2012).

The main objective 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. In this sense, we shall also observe that the proposed algorithms were implemented by the authors in two distinct computational codes, one of them within a general-purpose framework (Dadvand *et al.*, 2010, 2012).

The paper will be divided as follows: in a first section we will describe the explicit integration following the velocity and acceleration streamlines (X-IVAS) time integration to be used for de convective terms. This time integration was described before in a previous work of the authors (Idelsohn *et al.*, 2012) but it is included here for completeness. The PFEM for fixed mesh is then refreshed and its combination with the X-IVAS time integration of the convective term is presented. We denote as "PFEM-2" the resulting methodology. Finally, before showing the numerical examples, the strategy to obtain a fast algorithm, suitable for parallelization is described.

2. The X-IVAS time integration of the convective terms

2.1 General description

Let $\mathbf{x}_p$ be the vector defining the position of a particle in a 3D space, function of the time t. For simplicity we will use the notation $\mathbf{x}_p^t$ to denotes such point. Thus, at time $t = t^n$ we will write $\mathbf{x}_p^n$, at time $t = t^n + \Delta t = t^{n+1}$ we will denote the position as $\mathbf{x}_p^{n+1}$ and in general, in any time between $t = t^n$ and $t = t^{n+1}$ we will write $\mathbf{x}_p^{n+t}$.

Let $\mathbf{V}^{n+t}(\mathbf{x}_p^{n+t})$ and $\mathbf{A}^{n+t}(\mathbf{x}_p^{n+t})$ be two vectors defining the velocity and the acceleration of the particle $\mathbf{x}_p$ at any time t^{n+t}:

$$\begin{cases} \mathbf{V}^{n+t}\left(\mathbf{x}_p^{n+t}\right) = \dfrac{D\mathbf{x}_p^{n+t}}{Dt} & (2.1) \\[2ex] \mathbf{A}^{n+t}\left(\mathbf{x}_p^{n+t}\right) = \dfrac{D\mathbf{V}^{n+t}\left(\mathbf{x}_p^{n+t}\right)}{Dt} & (2.2) \end{cases}$$

where $D\phi/Dt$ represents the material (Lagrangian) derivative in time of any function ϕ. The material derivative is connected with the spatial derivative by the convective terms:

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + V_i \frac{\partial \phi}{\partial x_i} = \frac{\partial \phi}{\partial t} + \mathbf{V}^T \nabla \phi$$

In all initial value problems like the transient Navier-Stokes equations, the time solution of a problem consists in: given all the variables at time $t = t^n$, find the same variables at time $t = t^{n+1}$. In other words, to integrate in time equations (2.1) and (2.2):

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \displaystyle\int_n^{n+t} \mathbf{V}^\tau\left(\mathbf{x}_p^\tau\right) d\tau & (2.3) \\[2ex] \mathbf{V}^{n+t}\left(\mathbf{x}_p^{n+t}\right) = \mathbf{V}^n\left(\mathbf{x}_p^n\right) + \displaystyle\int_n^{n+t} \mathbf{A}^\tau\left(\mathbf{x}_p^\tau\right) d\tau & (2.4) \end{cases}$$

The accuracy of the results will depend to a great extent on the accuracy of the discretization of the velocity and acceleration in the space, but also in the approximation introduced in the integration of equations (2.3) and (2.4).

Time integration of the velocity. The idea of X-IVAS method is to use the velocity streamlines obtained at time step t^n to approximate the final position of a particle $\mathbf{x}_p^{n+1}$.

Let then:

$$\mathbf{x}_p^{n+t} \approx \mathbf{y}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n\left(\mathbf{x}_p^\tau\right) d\tau \tag{2.5}$$

Equation (2.5) is explicit because we are using only information at time step t^n. In this case, we are using neither a constant nor a linear approximation of the velocity field. Instead, we are using the same high order approximation the velocity field has at time t^n. The only difference with the exact integration (2.3) is that here we are performing the integral (within each time step) following a pseudo trajectory of the particles calculated with the velocity streamline, instead of following the true trajectory (Figure 1).

Once discretized, equation (2.5) may be integrated analytically or using any standard time integration scheme like explicit Runge-Kutta or alternatively by any substepping technique. The way to integrate analytically equation (2.5) inside each triangular element is explained in Idelsohn *et al.* (2012).

Time integration of the acceleration. The idea proposed in equation (2.5) may be also used for the acceleration. That is, for improving the time integration of the acceleration while remaining explicit in time. This means to approximate equation (2.4) by:

$$\mathbf{V}^{n+t}\left(\mathbf{x}_p^{n+t}\right) \approx \mathbf{V}^n\left(\mathbf{x}_p^n\right) + \int_n^{n+t} \mathbf{A}^n\left(\mathbf{x}_p^\tau\right) d\tau \tag{2.6}$$

Equation (2.6) represents an integration following the acceleration streamlines (Figure 2) obtained at time t^n. This may be solved using any of the particle position integrations described before:

$$\begin{cases} \mathbf{x}_p^{n+t} \approx \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n\left(\mathbf{x}_p^\tau\right) d\tau \\ \mathbf{V}^{n+t}\left(\mathbf{x}_p^{n+t}\right) \approx \mathbf{V}^n\left(\mathbf{x}_p^n\right) + \int_n^{n+t} \mathbf{A}^n\left(\mathbf{x}_p^\tau\right) d\tau \end{cases} \tag{2.7}$$

We must note that equations (2.7) are still explicit because they are using the velocity and acceleration at time t^n (Figure 2). The way to integrate analytically equation (2.7) inside each triangular element is explained in the Donea and Huerta (2003).

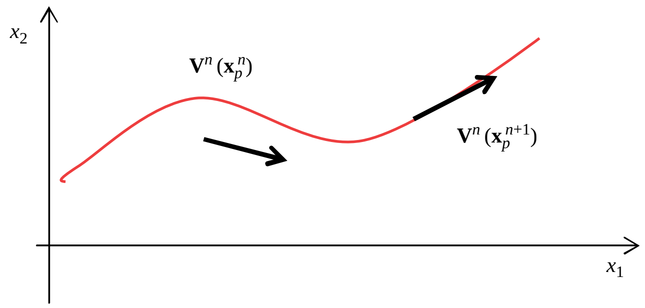

x_2
$\mathbf{V}^n(\mathbf{x}_p^n)$
$\mathbf{V}^n(\mathbf{x}_p^{n+1})$
x_1

Figure 1.
Integration following the velocity streamlines

2.2 The X-IVAS integration applied to the incompressible Navier-Stokes equations

In the Navier-Stokes equation, the acceleration is obtained from the momentum conservation equation:

$$\mathbf{A}^t\left(\mathbf{x}_p^t\right) = \frac{1}{\rho^t\left(\mathbf{x}_p^t\right)}\left[\nabla\cdot\sigma^t\left(\mathbf{x}_p^t\right) + \mathbf{b}^t\left(\mathbf{x}_p^t\right)\right] \tag{2.8}$$

with the stress tensor:

$$\sigma^t\left(\mathbf{x}_p^t\right) = \tau^t\left(\mathbf{x}_p^t\right) - p^t\left(\mathbf{x}_p^t\right)\mathbf{I} \tag{2.9}$$

and the deviatoric tensor:

$$\tau^t\left(\mathbf{x}_p^t\right) = \mu\left[\nabla\mathbf{V}^t\left(\mathbf{x}_p^t\right) + \nabla^T\mathbf{V}^t\left(\mathbf{x}_p^t\right)\right] \tag{2.10}$$

where ρ is the density, μ the viscosity, p the pressure, $\mathbf{b}$ a volumetric force and $\mathbf{I}$ the identity tensor.

The mass conservation reads:

$$\frac{\partial\rho^t\left(\mathbf{x}_p^t\right)}{\partial t} + \nabla\cdot\left[\rho^t\left(\mathbf{x}_p^t\right)\mathbf{V}^t\left(\mathbf{x}_p^t\right)\right] = 0 \tag{2.11}$$

Since for an incompressible flow:

$$\rho^t\left(\mathbf{x}_p^t\right) = \rho(\mathbf{x}_p) = \rho_p = cte > 0 \tag{2.12}$$

Equation (2.11) becomes:

$$\nabla\cdot\mathbf{V}^t\left(\mathbf{x}_p^t\right) = 0 \tag{2.13}$$

Using the X-IVAS method presented before, the Navier-Stokes equations between the two time stations t^n and $t = t^{n+1}$ read:

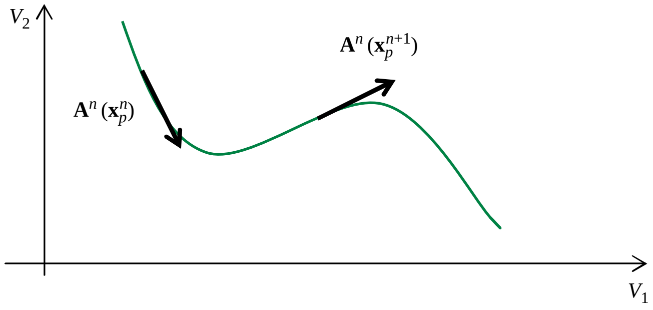

V_2
$\mathbf{A}^n(\mathbf{x}_p^{n+1})$
$\mathbf{A}^n(\mathbf{x}_p^n)$
V_1

Figure 2.
Integration following the acceleration streamlines

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n \left(\mathbf{x}_p^\tau \right) d\boldsymbol{\tau} \\ \mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \mathbf{V}^n \left(\mathbf{x}_p^n \right) + \frac{1}{\rho_p} \int_n^{n+1} \left[\nabla \cdot \boldsymbol{\sigma}^n \left(\mathbf{x}_p^t \right) + \mathbf{b}^n \left(\mathbf{x}_p^t \right) \right] dt \end{cases} \tag{2.14}$$

The incompressibility constraint in equation (2.13) shows no time dependence and hence needs to be treated implicitly, unless a certain degree of compressibility is admitted.

On the contrary, the solution of equation (2.14) by explicit techniques is possible and the use of the X-IVAS scheme guarantees the possibility of using large time steps.

The presence of the viscous term implies however a practical limitation to the step size, namely that the Fourier number ($Fo = 2\mu\Delta t/\rho h^2$) must remain smaller than one. For this reason we have modified equation (2.14) in order to remain implicit not only for the pressure but also for the viscous terms. The new explicit-implicit integration scheme is described next. Through the paper, we will refer as "implicit-diffusion algorithm" to the algorithm that considers implicitly the contribution of the viscosity, and as "explicit-diffusion algorithm" to the explicit alternative.

Equation (2.14) is replaced by the following:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n \left(\mathbf{x}_p^\tau \right) d\tau \\ \rho_p \mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \mathbf{V}^n \left(\mathbf{x}_p^n \right) + \int_n^{n+1} \Big\{ \nabla \cdot \tau^n \left(\mathbf{x}_p^t \right) + \nabla \cdot \delta\tau \left(\mathbf{x}_p^t \right) - \nabla p^n \left(\mathbf{x}_p^t \right) \\ \qquad - \nabla \delta p \left(\mathbf{x}_p^t \right) + \mathbf{b}^n \left(\mathbf{x}_p^t \right) \Big\} dt \end{cases} \tag{2.15}$$

We remark the implicit terms for stress and the pressure in the second of the equation (2.15), i.e.:

$$\begin{cases} \delta\tau\left(\mathbf{x}_p^t\right) = \tau^t \left(\mathbf{x}_p^t \right) - \tau^n \left(\mathbf{x}_p^t \right) = \mu \left[\nabla \delta \mathbf{V} \left(\mathbf{x}_p^t \right) + \nabla^T \delta \mathbf{V} \left(\mathbf{x}_p^t \right) \right] \\ \delta \mathbf{V} \left(\mathbf{x}_p^t \right) = \mathbf{V}^t \left(\mathbf{x}_p^t \right) - \mathbf{V}^n \left(\mathbf{x}_p^t \right) \end{cases} \tag{2.16}$$

$$\delta p \left(\mathbf{x}_p^t \right) = p^t \left(\mathbf{x}_p^t \right) - p^n \left(\mathbf{x}_p^t \right) \tag{2.17}$$

The integral of the $\delta\tau$ and δp terms in equation (2.15) will be approximated by a fully implicit backward integration scheme:

$$\int_n^{n+1} \left\{ \nabla \cdot \delta\tau \left(\mathbf{x}_p^t \right) - \nabla \delta p \left(\mathbf{x}_p^t \right) \right\} dt \approx \frac{\Delta t}{2} \left[\nabla \cdot \delta\tau \left(\mathbf{x}_p^{n+1} \right) - \nabla \delta p \left(\mathbf{x}_p^{n+1} \right) \right] \tag{2.18}$$

Then, the second equation of equation (2.15) will be split in the following two steps:

$$\begin{cases} \rho_p \widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \mathbf{V}^n\left(\mathbf{x}_p^n\right) + \int_n^{n+1}\left\{\nabla\cdot\tau^n\left(\mathbf{x}_p^t\right) - \nabla p^n\left(\mathbf{x}_p^t\right) + \mathbf{b}^n\left(\mathbf{x}_p^t\right)\right\}dt \\ \qquad\qquad\qquad + \dfrac{\Delta t}{2}\left\{\nabla\cdot\delta\tau\left(\mathbf{x}_p^{n+1}\right)\right\} \\ \rho_p \mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) - \frac{\Delta t}{2}\nabla\left[\delta p\left(\mathbf{x}_p^{n+1}\right)\right] \end{cases} \tag{2.19}$$

Applying the divergence operator to both sides of the second equation of equation (2.19) and taking into account equation (2.13) gives:

$$\rho_p \nabla\cdot\widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \nabla\cdot\left\{\nabla\left[\delta p\left(\mathbf{x}_p^{n+1}\right)\right]\right\}\frac{\Delta t}{2} \tag{2.20}$$

Then the four steps using the X-IVAS technique remains:

- *Step I.* Evaluate explicitly the velocity $\widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ and the new particle position $\mathbf{x}_p^{n+1}$:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t}\mathbf{V}^n\left(\mathbf{x}_p^\tau\right)d\tau \\ \rho_p \widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \mathbf{V}^n\left(\mathbf{x}_p^n\right) + \int_n^{n+1}\left\{\nabla\cdot\tau^n\left(\mathbf{x}_p^t\right) - \nabla p^n\left(\mathbf{x}_p^t\right) + \mathbf{b}^n\left(\mathbf{x}_p^t\right)\right\}dt \end{cases} \tag{2.21}$$

- *Step II.* Solve implicitly the first equation (2.19) to obtain the velocity correction $\delta\widehat{\mathbf{V}}\left(\mathbf{x}_p^{n+1}\right)$:

$$\left[\rho_p - \mu\frac{\Delta t}{2}\nabla\cdot\left(\nabla + \nabla^T\right)\right]\delta\hat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p\delta\widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) \tag{2.22}$$

where $\delta\widehat{\widehat{\mathbf{V}}}\left(\mathbf{x}_p^{n+1}\right) = \widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) - \mathbf{V}^n\left(\mathbf{x}_p^{n+1}\right)$

- *Step III.* Solve implicitly the Laplace equation to obtain the pressure increment $\delta p\left(\mathbf{x}_p^{n+1}\right)$:

$$\rho_p \nabla\cdot\widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \nabla\cdot\left\{\nabla\left[\delta p\left(\mathbf{x}_p^{n+1}\right)\right]\right\}\frac{\Delta t}{2} \tag{2.23}$$

where $\widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) + \delta\widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$

- *Step IV.* Evaluate the new incompressible velocity $\mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ and pressure $p^{n+1}\left(\mathbf{x}_p^{n+1}\right)$:

$$\rho_p \mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) - \nabla\left[\delta p\left(\mathbf{x}_p^{n+1}\right)\right]\frac{\Delta t}{2} \tag{2.24}$$

$$p^{n+1}\left(\mathbf{x}_p^{n+1}\right) = p^n\left(\mathbf{x}_p^{n+1}\right) + \delta p\left(\mathbf{x}_p^{n+1}\right) \tag{2.25}$$

It must be noted that for homogeneous fluid flows (e.g. fluid with constant viscosity and constant density) both implicit parts, equations (2.22) and (2.23) lead to a matrix with constant coefficients. Such matrix may be factorized once during the whole calculation (or an expensive pre-conditioner could be computed) thus allowing a much faster solution phase for the following steps. In an Eulerian formulation, the presence of the convective terms in the tangent matrix make impossible to factorize the whole tangent matrix once. This fact, combined with the possibility of using much larger time steps, provides a competitive lead of the X-IVAS algorithm with respect to the Eulerian counterparts.

3. The PFEM with X-IVAS integration (PFEM-2) applied to the incompressible Navier-Stokes equations

A method using a Lagrangian formulation may be solved using either a moving mesh approach or relying on a fixed "background" mesh. The idea is the following: once the velocity at the new position $\mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ has been found by solving the Lagrangian equation, a new discretization of the continuum is needed. This can be achieved either by building a new mesh to connect the new particle positions or by devising a technique to project the velocity results onto the fixed background mesh. The previous versions of the PFEM (Oñate *et al.*, 2008; Idelsohn *et al.*, 2008a, b; Larese *et al.*, 2008; Ryzhakov *et al.*, 2010, 2012) relied on continuous re-meshing thus implying reforming the finite element data structures and solving a different linear algebra problem at each time step. These tasks were identified to be very expensive in terms of CPU time and very hard to parallelize satisfactorily, thus limiting the performance of the algorithm. The fixed mesh algorithm was thus designed to avoid the re-meshing step and appeared to provide, as an extra bonus, the definition of linear algebra problems with constant coefficients.

We should remark that the viscous problem to be solved in the implicit viscosity correction step, has constant coefficients exclusively in the case of constant viscosity distribution. Even though most of the flows are turbulent and viscosity depends on the flow itself, thus implying that this case is often not met, there is some interest in laminar flows as in microfluidics and also in direct numerical simulation (DNS) or LES with an explicit treatment of the Reynolds stress tensor. An extended description of the PFEM with moving mesh and with fixed mesh may be obtained in Idelsohn *et al.* (2012). We include in the following section a brief description of the algorithm with fixed mesh.

3.1 The PFEM-2 with fixed mesh

Suppose a fixed "background" mesh is given. Since such mesh covers the domain of interest, several of the Lagrangian particles will fall within each of the elements that compose the mesh. At the beginning of each step the velocity at the mesh nodes and at the internal particles is known. Once the algorithm explained before is applied to all the particles to obtain $\mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$, we project on the fixed mesh nodes these velocities using an assembly of all the particles surrounding each mesh node. Different approaches are available to perform such projections, for example SPH or moving least square (MLS) techniques could be used for the interpolation, as well as weights based on the position on the top of the underlying mesh. While a complete discussion of this problem falls beyond the scope of current work, as a general observation we should remark that, in order to avoid unwanted artificial numerical diffusion of the results, many particles are needed around each node of the fixed mesh.

Figure 3 shows one possible approach to carry out the projection step. We are interested in computing the projection of a given particle field on the background mesh,

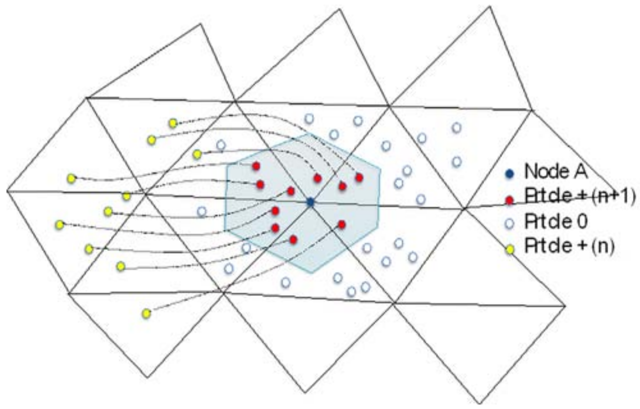


Figure 3.
Fixed mesh and Lagrangian moving particles: how to compute the projection from particle-to-mesh nodes

we hence propose to compute for each node of the fixed mesh the contribution of the subset of particles placed "in its vicinity". In this work we consider as "close", the particles that belong to the shadow region highlighted in Figure 3 and corresponding to the node called node A. These particles, shown as red circles and called *prtcle* + (n + 1), represent the particles that at the time step t^{n+1} fall inside the shadow region and come from another part of the domain. Their original position is represented in the same figure by yellow circles and are identified as *prtcle* + (n). In our work we define the shadow region around a given node A as the union of "sub-elements" that cover the domain around A. Such sub-elements are formed by the vertex A, the centroid of the triangle and the mid of the edges. Their union associates to a given point A an area which is equivalent to one-third of the total area of the triangles around the node of interest. The particles marked in the figure as empty circles represent those particles that belong to the elements surrounding A but do not contribute because they are out of the shadow region formed by the sub-elements just introduced. The streamlines corresponding to the subset of particles that at the present time are inside the shadow region and contribute to the value of the node A in the mesh at the current time step are drawn as the paths between the stars and the filled circles.

3.2 Spatial discretization of the incompressible Navier-Stokes equations using X-IVAS integration

Given a fixed mesh we can define the approximation on the component of velocity field and pressure as:

$$\mathbf{V}^n(\mathbf{x}) = \vec{\vec{\mathbf{N}}}^T(\mathbf{x})\vec{\mathbf{V}}^n \tag{3.1}$$

$$p^n(\mathbf{x}) = \vec{\mathbf{N}}_p^T(\mathbf{x})\vec{\mathbf{p}}^n \tag{3.2}$$

where $\vec{\vec{\mathbf{N}}}^T(\mathbf{x})$ and $\vec{\mathbf{N}}_p(\mathbf{x})$ identify the finite element (FE) shape functions for the velocity and the pressure, respectively.

Starting with first step of the algorithm described in Section 2, the terms $\Delta \cdot \{\mu(\mathbf{x}) \times [\nabla \mathbf{V}^n(\mathbf{x}) + \nabla^T \mathbf{V}^n(\mathbf{x})]\}$ and $\nabla p^n(\mathbf{x})$ in equation (2.21) will be approximated by a continuous field. The two contributions will be named $\mathbf{g}^n$ ($\mathbf{x}$) and Π^n ($\mathbf{x}$), respectively, such that:

$$\mathbf{g}^n(\mathbf{x}) = \vec{\vec{\mathbf{N}}}^T(\mathbf{x})\vec{\mathbf{g}}^n \text{ and } \Pi^n(\mathbf{x}) = \vec{\vec{\mathbf{N}}}^T(\mathbf{x})\vec{\Pi}^n. \tag{3.3}$$

To evaluate $\vec{\mathbf{g}}^n$ and $\vec{\Pi}^n$ we will use the same approximation used in the standard FEM, that is:

$$\int_\Omega \vec{\vec{\mathbf{N}}}\{\nabla\cdot[\mu(\nabla\mathbf{V}^n + \nabla^T\mathbf{V}^n)] - \mathbf{g}^n\}d\Omega\vec{\mathbf{V}}^n = 0 \tag{3.4}$$

and:

$$\int_{\Omega^n} \vec{\vec{\mathbf{N}}}\{\nabla p^n - \Pi^n\}d\Omega = 0 \tag{3.5}$$

Integrating by parts the first term in both equations (3.4) and (3.5):

$$-\int_\Omega \left\{\nabla\vec{\vec{\mathbf{N}}}\mu\left[\nabla\vec{\vec{\mathbf{N}}}^T + \nabla^T\vec{\vec{\mathbf{N}}}^T\right]\right\}d\Omega\,\vec{\mathbf{V}}^n + \int_\Gamma \vec{\vec{\mathbf{N}}}q^n\,d\Gamma - \int_\Omega \vec{\vec{\mathbf{N}}}\vec{\vec{\mathbf{N}}}^T\,d\Omega\,\vec{\mathbf{g}}^n = 0 \tag{3.6}$$

and:

$$-\int_\Omega \left\{\nabla\vec{\vec{\mathbf{N}}}\vec{\mathbf{N}}_p^T\right\}d\Omega\,\vec{\mathbf{p}}^n + \int_\Gamma \vec{\vec{\mathbf{N}}}p^n\mathbf{I}\,\mathbf{n}\,d\Gamma - \int_\Omega \vec{\vec{\mathbf{N}}}\vec{\vec{\mathbf{N}}}^T\,d\Omega\,\vec{\Pi}^n = 0 \tag{3.7}$$

we obtain:

$$\vec{\mathbf{g}}^n - \vec{\Pi}^n = \left(\vec{\vec{\mathbf{M}}}\right)^{-1}\left(\vec{\vec{\mathbf{K}}}\vec{\mathbf{V}}^n - \vec{\vec{\mathbf{B}}}\vec{\mathbf{p}}^n - \vec{\sigma}^n\right) \tag{3.8}$$

with:

$$\vec{\vec{\mathbf{K}}} = \int_\Omega \left\{\nabla\vec{\vec{\mathbf{N}}}\mu\left[\nabla\vec{\vec{\mathbf{N}}}^T + \nabla^T\vec{\vec{\mathbf{N}}}^T\right]\right\}d\Omega \tag{3.9}$$

$$\vec{\vec{\mathbf{M}}} = \int_\Omega \vec{\vec{\mathbf{N}}}\vec{\vec{\mathbf{N}}}^T d\Omega \tag{3.10}$$

$$\vec{\vec{\mathbf{B}}} = \int_\Omega \nabla\vec{\vec{\mathbf{N}}}\vec{\mathbf{N}}_p^T\,d\Omega \tag{3.11}$$

and:

$$\begin{aligned}\vec{\sigma}^n &= \int_\Gamma \vec{\vec{\mathbf{N}}}q^n d\Gamma - \int_\Gamma \vec{\vec{\mathbf{N}}}p^n\mathbf{I}\,\mathbf{n}\,d\Gamma = \int_\Gamma \vec{\vec{\mathbf{N}}}[\mu(\nabla\mathbf{V}^n + \nabla^T\mathbf{V}^n) - p^n\mathbf{I}]\mathbf{n}\,d\Gamma \\ q^n &= \mu(\nabla\mathbf{V}^n + \nabla^T\mathbf{V}^n)\mathbf{n}\end{aligned} \tag{3.12}$$

Then, after discretization in space, the first step (equation (2.21)) will read:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \vec{\vec{\mathbf{N}}}^T \left(\mathbf{x}_p^\tau\right) d\tau \vec{\mathbf{V}}^n \\ \rho_p \hat{\hat{\vec{\mathbf{V}}}}^{n+1} \left(\mathbf{x}_p^{n+1}\right) = \rho_p \mathbf{V}^n \left(\mathbf{x}_p^n\right) + \int_n^{n+1} \vec{\vec{\mathbf{N}}}^T \left(\mathbf{x}_p^t\right) dt\, \vec{\mathbf{g}}^n \\ \qquad - \int_n^{n+1} \vec{\vec{\mathbf{N}}}^T \left(\mathbf{x}_p^t\right) dt\, \vec{\Pi}^n + \int_n^{n+1} \mathbf{b}^n \left(\mathbf{x}_p^t\right) dt \end{cases} \tag{3.13}$$

Note 1. All the time integrals in equation (3.13) may be computed by different methods. For linear shape function elements it may be computed analytically inside each finite element. It may also be evaluated dividing the time step Δt in several small sub-steps δt. This is not an expensive operation taking into account that computations are explicit and each particle may be evaluated separately from each other in parallel.

Note 2. On the boundaries Γ_V where the velocity is known (Dirichlet boundary conditions), it is also known the acceleration. Let us call it $D\mathbf{V}/Dt|_{\Gamma_V}$. On these boundaries, it is necessary to impose the value of $(\mathbf{g} - \Pi)|_{\Gamma_V}$ such that: $\mathbf{g} - \Pi = \rho(D\mathbf{V}/Dt) - \mathbf{b}$ on Γ_V.

After the first step, we project the $\hat{\hat{\vec{\mathbf{V}}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ on the fixed mesh to obtain a vector with $\delta\hat{\hat{\vec{\mathbf{V}}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ at the nodes of the fixed mesh $\delta\hat{\hat{\vec{\mathbf{V}}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) \Rightarrow \delta\hat{\hat{\vec{\mathbf{V}}}}^{n+1}(\mathbf{x})$.

The next step is to solve implicitly equation (2.22) which using a finite element approximation reads:

$$\left[\mathbf{M}(\rho) + \mathbf{K}\frac{\Delta t}{2}\right] \delta\hat{\vec{\mathbf{V}}}^{\mathbf{n+1}}(\mathbf{x}) = \mathbf{M}(\rho)\delta\hat{\hat{\vec{\mathbf{V}}}}^{n+1}(\mathbf{x}) \tag{3.14}$$

where:

$$\vec{\vec{\mathbf{M}}}(\rho) = \int_\Omega \vec{\vec{\mathbf{N}}} \rho_{(\mathbf{x})} \vec{\vec{\mathbf{N}}}^T d\Omega \tag{3.15}$$

The following step is to solve also implicitly (equation (2.23)). Using a classical FEM approximation reads:

$$\int_\Omega \vec{\mathbf{N}}_p \rho_{(\mathbf{x})} \nabla \cdot \hat{\mathbf{V}}^{n+1} d\Omega - \int_\Omega \vec{\mathbf{N}}_p \nabla \cdot \frac{\Delta t}{2} \nabla[\delta p^{n+1}] d\Omega = 0 \tag{3.16}$$

Integrating by parts both terms in equation (3.16) gives:

$$-\int_\Omega \nabla\vec{\mathbf{N}}_p \rho_{(\mathbf{x})} \vec{\vec{\mathbf{N}}}^T d\Omega \hat{\vec{\mathbf{V}}}^{n+1} + \int_\Omega \nabla\vec{\mathbf{N}}_p \frac{\Delta t}{2} \nabla\vec{\mathbf{N}}_p^T d\Omega\, \delta\vec{\mathbf{p}}^{n+1} + \int_{\Gamma_V} \vec{\mathbf{N}}_p \bar{\mathbf{V}}^{n+1} d\Gamma = 0 \tag{3.17}$$

or in matrix form:

$$-\vec{\vec{\mathbf{B}}}(\rho)\hat{\vec{\mathbf{V}}}^{\mathbf{n+1}} + \frac{\Delta t}{2}\vec{\vec{\mathbf{L}}}\, \delta\vec{\mathbf{p}}^{\mathbf{n+1}} + \bar{\vec{\mathbf{V}}}^{\mathbf{n+1}} = 0 \tag{3.18}$$

with:

$$\vec{\vec{\mathbf{B}}}(\rho) = \int_\Omega \nabla\vec{\mathbf{N}}_p \rho_{(\mathbf{x})} \vec{\vec{\mathbf{N}}}^T d\Omega \tag{3.19}$$

$$\vec{\vec{\mathbf{L}}} = \int_\Omega \nabla\vec{\mathbf{N}}_p \nabla\vec{\mathbf{N}}_p^T d\Omega \tag{3.20}$$

$$\bar{\vec{\mathbf{V}}}^{n+1} = \int_{\Gamma_V} \vec{\mathbf{N}}_p \bar{\mathbf{V}}^{n+1} d\Gamma = \int_{\Gamma_V} \vec{\mathbf{N}}_p \left[\rho_{(\mathbf{x})} \hat{\mathbf{V}}^{n+1} - \frac{\Delta t}{2} \nabla(\delta p)\right] d\Gamma \tag{3.21}$$

The momentum equation does not need any stabilization, since a Lagrangian formulation is used. Equation (3.18) on the contrary must be stabilized in space since an unstable velocity-pressure pair is chosen. Any spatial stabilization method can be used in principle (Oñate *et al.*, 2007; Codina, 2000), leading, in general, to a term of the type $\vec{\vec{\mathrm{S}}}(\tau)\vec{\mathbf{p}}^{n+1}$ (where $\vec{\vec{\mathrm{S}}}(\tau)$ is a Laplacian-like matrix scaled by a suitable stabilization parameter τ) that must be added to equation (3.18) (Codina, 2000):

$$\left[\frac{\Delta t}{2}\vec{\vec{\mathbf{L}}} + \vec{\vec{\mathbf{S}}}(\tau)\right]\delta\vec{\mathbf{p}}^{n+1} = \vec{\vec{\mathrm{B}}}(\rho)\hat{\vec{\mathbf{V}}}^{n+1} - \bar{\vec{\mathbf{V}}}^{n+1} + \vec{\vec{\mathbf{S}}}(\tau)\vec{\mathbf{p}}^n \tag{3.22}$$

After solving equation (3.21) the pressure at time $t = t^{n+1}$ may be evaluated.

The last step is the evaluation of the final velocity at time $t = t^{n+1}$. Using equation (2.24):

$$\rho_p \vec{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \widehat{\vec{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) - \frac{\Delta t}{2}\,\delta\Pi^{n+1}\left(\mathbf{x}_p^{n+1}\right) \tag{3.23}$$

The Vector $\delta\vec{\Pi}^{n+1}$ may be calculated using equation (3.5). In matrix notation:

$$\int_\Omega \vec{\vec{\mathbf{N}}}\,\nabla\vec{\mathbf{N}}_p^T d\Omega\;\delta\vec{\mathbf{p}}^{n+1} = \int_\Omega \vec{\vec{\mathbf{N}}}\,\vec{\mathbf{N}}^T d\Omega\;\delta\vec{\Pi}^{n+1} \tag{3.24}$$

or:

$$\delta\Pi^{n+1} = \left(\vec{\vec{\mathbf{M}}}\right)^{-1}\vec{\vec{\mathbf{D}}}\,\delta\vec{\mathbf{p}}^{n+1} \tag{3.25}$$

with:

$$\vec{\vec{\mathbf{D}}} = \int_\Omega \vec{\vec{\mathbf{N}}}\,\nabla\vec{\mathbf{N}}_p^T d\Omega \tag{3.26}$$

The stabilization parameter used here may be found in several publications (Hughes and Tezduyar, 1984; Tezduyar *et al.*, 1992; Badia and Codina, 2008; Codina *et al.*, 2007).

As a resume, the PFEM with the X-IVAS integrations with fixed mesh reads:

(I) Evaluate the viscous and pressure gradients vector ($\vec{\mathbf{g}}^n - \vec{\Pi}^n$):

$$\vec{\mathbf{g}}^n - \vec{\Pi}^n = \left(\vec{\vec{\mathbf{M}}}\right)^{-1}\left(\vec{\vec{\mathbf{K}}}\vec{\mathbf{V}}^n - \vec{\vec{\mathbf{B}}}\vec{\mathbf{p}}^n - \vec{\sigma}^n\right)$$

(II) Evaluate (explicitly) the fractional velocity $\widehat{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ and the position $\mathbf{x}_p^{n+1}$ for each of the particles in the model:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n\left(\mathbf{x}_p^\tau\right) d\tau \\ \rho_p \widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \mathbf{V}^n\left(\mathbf{x}_p^n\right) + \int_n^{n+1} \vec{\mathbf{N}}^T\left(\mathbf{x}_p^t\right) dt\, \vec{\mathbf{g}}^n - \int_n^{n+1} \vec{\mathbf{N}}^T\left(\mathbf{x}_p^t\right) dt\, \vec{\Pi}^n \\ \qquad + \int_n^{n+1} \mathbf{b}^n\left(\mathbf{x}_p^t\right) dt \end{cases}$$

(III) Project the fractional velocity $\widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$ to the underlying mesh on a vector $\delta\widehat{\widehat{\vec{\mathbf{V}}}}^{n+1}$:

$$\widehat{\widehat{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) - \mathbf{V}^n\left(\mathbf{x}_p^{n+1}\right) \Rightarrow \delta\widehat{\widehat{\vec{\mathbf{V}}}}^{n+1}(\mathbf{x})$$

(IV) Solve implicitly the linear system of equations to obtain $\delta\widehat{\vec{\mathbf{V}}}^{n+1}$:

$$\left[\mathbf{M}(\rho) + \mathbf{K}\frac{\Delta t}{2}\right] \delta\widehat{\vec{\mathbf{V}}}^{n+1} = \mathbf{M}(\rho)\delta\widehat{\widehat{\vec{\mathbf{V}}}}^{n+1}$$

(V) Solve implicitly the linear system of equations to obtain the pressure increment $\delta\vec{\mathbf{p}}^{n+1}$ on the nodes of the fixed mesh:

$$\left[\frac{\Delta t}{2}\vec{\vec{\mathbf{L}}} + \vec{\vec{\mathbf{S}}}(\tau)\right] \delta\vec{\mathbf{p}}^{n+1} = \vec{\vec{\mathbf{B}}}(\rho)\hat{\vec{\mathbf{V}}}^{n+1} - \bar{\vec{\mathbf{V}}}^{n+1} + \vec{\vec{\mathbf{S}}}(\tau)\vec{\mathbf{p}}^n$$

(VI) Evaluate the new pressure gradients vector $\delta\vec{\Pi}^{n+1}$:

$$\delta\vec{\Pi}^{n+1} = \left(\vec{\vec{\mathbf{M}}}\right)^{-1} \vec{\vec{\mathbf{D}}}\, \delta\vec{\mathbf{p}}^{n+1}$$

(VII) Evaluate at each particle the new velocity $\mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$:

$$\rho_p \vec{\mathbf{V}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) = \rho_p \widehat{\vec{\mathbf{V}}}^{n+1}\left(\mathbf{x}_p^{n+1}\right) - \frac{\Delta t}{2}\, \delta\Pi^{n+1}\left(\mathbf{x}_p^{n+1}\right)$$

(VIII) Evaluate at each node of the mesh the new velocity vector $\vec{\mathbf{V}}^{n+1}$:

$$\vec{\mathbf{V}}^{n+1} = \hat{\vec{\mathbf{V}}}^{n+1} - \delta\vec{\Pi}^{n+1} \frac{\Delta t}{2\rho(\mathbf{x})}$$

(IX) Evaluate at each node of the mesh the new pressure vector $\vec{\mathbf{p}}^{n+1}$:

$$\vec{\mathbf{p}}^{n+1} = \vec{\mathbf{p}}^n + \delta\vec{\mathbf{p}}^{n+1}$$

(X) Increase the time $t^{n+1} = t^n + \Delta t$ and update the variables at the nodes:

$$\vec{\mathbf{V}}^n \Leftarrow \vec{\mathbf{V}}^{n+1}$$

$$\vec{\mathbf{p}}^n \Leftarrow \vec{\mathbf{p}}^{n+1}$$

Update also the velocity at the particles:

$$\mathbf{V}^n\left(\mathbf{x}_p^n\right) \Leftarrow \mathbf{V}^{n+1}\left(\mathbf{x}_p^{n+1}\right)$$

go to (I)

It must be noted that for the case of homogeneous fluids (e.g. μ = constant and ρ = constant) and under the hypothesis that a fixed background mesh is used, the matrixes $\mathbf{K}$, $\mathbf{M}$, $\mathbf{L}$, $\mathbf{B}$ and $\mathbf{D}$ are constant during all the time increments. However, for heterogeneous fluid flows, when the fluid characteristics vary, some of them (e.g. $\mathbf{K}(\mu)$, $\mathbf{M}(\rho)$, $B(\rho)$ and $\mathbf{S}(\tau)$) will require recomputation.

4. Numerical examples

This section is devoted to show by numerical examples the reliability and the efficiency of the PFEM-2 method. While Idelsohn *et al.* (2012) focused mostly on the moving mesh implementation of the algorithms, we concentrate here on the fixed mesh version of the code. The examples chosen were the circular cylinder and a typical NACA 0012, often used in aeronautical engineering. Even though the formulation is also implemented in 3D as shown in Idelsohn *et al.* (2012) here only 2D simulations are shown. A similar 3D analysis will be performed in the future.

In order to assess the accuracy and performance of the proposed method, which represent the main goal of the current paper, we include for each example two sub-sections, one to discuss the accuracy and the other to address the performance in terms of CPU time. The proposed algorithm was benchmarked internally using the same mesh against OpenFOAM (OpenCFD, 2009) and against Kratos (Dadvand *et al.*, 2010, 2012), an in-house solver that implements the implicit fractional step method. It was verified empirically that for the selected examples the nonlinearities in the computation did not allow the use of large CFLs. As a consequence, since performance results measured for Kratos were similar to the ones measured for OpenFOAM we only report the latter.

For all the examples we report the CPU-times obtained treating implicitly the viscous term (implicit diffusion) as described at the beginning of the paper and, for reference, the CPU-times obtained by treating it explicitly (explicit diffusion).

For the examples chosen, we plot drag, lift and moment curves, which represent the typical quantities of interest for engineering use and we compare them quantitatively to reference values.

4.1 Flow around a circular cylinder at Re = 1,000

This example involves the flow past a circular cylinder as shown in Figure 4. The computational domain is rectangular and extended five cylinder diameters upwards and in both transverse directions and 15 diameters downwards, with the cylinder diameter D of 1 and centered at the origin.

The velocity is imposed to a value of U = 1, identified as the free-stream velocity both at the inlet (right side of the domain) and on the top and bottom boundaries.

A no-slip boundary condition is applied on the sides of the cylinder. Pressure is set to zero at the outflow. Viscosity is set so to obtain a Reynolds number (Re) of 1,000, based on the cylinder diameter and on the prescribed inflow velocity.

The finite element mesh employed consists of 88,000 linear triangles, with 44,520 nodal points, being refined near the cylinder.

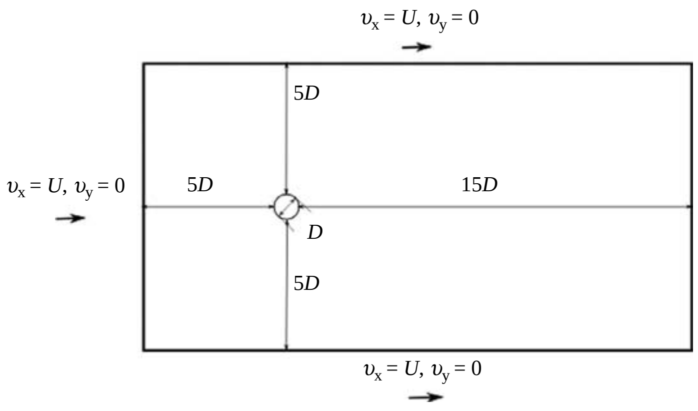

$\upsilon_x = U,\ \upsilon_y = 0$
$\upsilon_x = U,\ \upsilon_y = 0$
$\upsilon_x = U,\ \upsilon_y = 0$
5D
5D
15D
D
5D

Figure 4.
Flow around a cylinder: computational domain

The results achieved here were compared against those obtained by Mittal and Kumar (2001).

The number of particles inside each mesh element is controlled to be in the range of 1-3 by sub-element. The sub-element is defined as the region inside each triangle round each vertex, i.e. there are three sub-elements per triangle, defined as the region bounded by the vertex, both half edges converging at the vertex and the centroid of the triangle. In this way a more controlled interpolation error is obtained.

Accuracy results. Drag and the lift curves are compared against the results of Mittal in Figure 5. Comparison to the reference shows the drag where a good agreement in the oscillation frequency is observed with a drag mean value 10 percent above the reference mean value. The amplitude of this oscillated drag is in good agreement with the reference.

Although this is out of scope in the current paper, we would like to observe that the method used in projecting from the particles to the mesh affects to some extent the

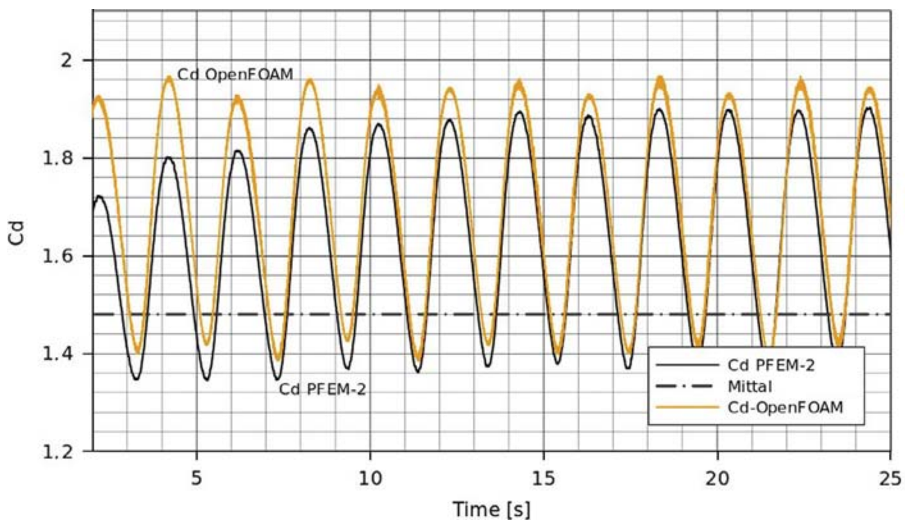


Figure 5.
Comparison drag coefficient Re = 1,000 PFEM-2 vs OpenFOAM® vs Mittal

level of "noise" in the lift and drag evaluation. This aspect will be discussed in detail in a future publication.

As shown in Figure 6, the lift force oscillates (as expected) around the zero with amplitude 20 percent above the reference value. Figures 5 and 6 also show a comparison to the results of OpenFOAM® (OpenCFD, 2009).

The analysis of the results shows how the results of PFEM2 are very similar to the ones obtained with using OpenFOAM® using the same mesh.

Figure 6 shows the lift curves from which we can draw similar conclusions.

The results shown in Table I summarize the results obtained proving that a good quantitative agreement is achieved both for the drag and lift and for the Strouhal number.

Performance results. The OpenFOAM® solution was used as a benchmark for the computational time. 1 s of simulation time was used as the reference interval. A maximum Courant of around 10 was used for fixed mesh version with implicit diffusion.

The other versions were used with a more conservative time step. OpenFOAM® was found to have a practical limitation for the Courant number around 1 as for larger values the nonlinear iterations diverged. In order to get accurate solutions two PISO corrections and two non-orthogonal corrections were used, with a tolerance about 10^{-6}. Here a summary of the main parameters is included:

(1) CPU: Intel I7-2600, 3.4 GHz.

(2) Real time simulated: 1 s.

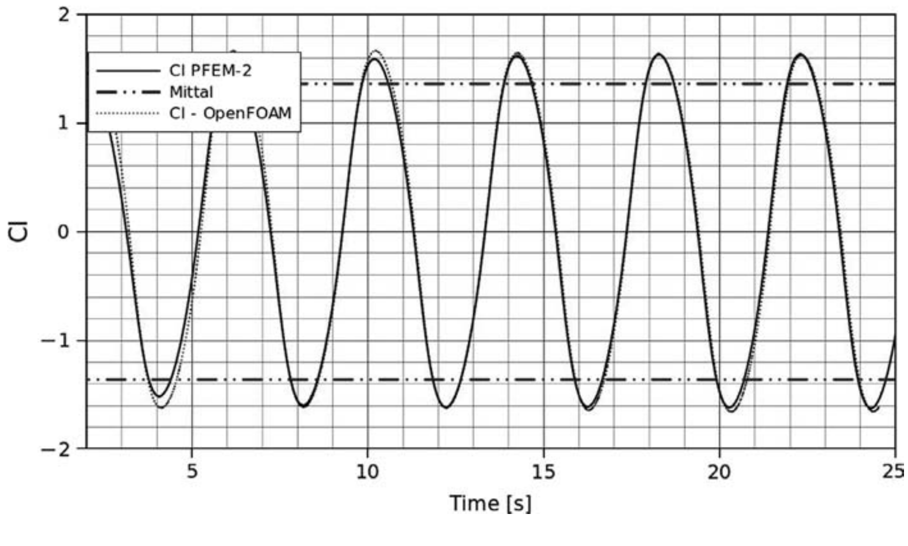


Figure 6.
Comparison lift coefficient Re = 1,000 PFEM-2 vs OpenFOAM® vs Mittal

Table I.
A brief summary of the comparison among OpenFOAM®, Mittal and Kumar (2001) and PFEM-2

	Strouhal	$\overline{C_d}$	C_d *amplitude*	C_L *amplitude*
Mittal	0.25	1.48	0.21	1.36
PFEM-2	0.2475	1.639	0.245	1.63
OpenFOAM	0.26	1.696	0.25	1.62

(3) PFEM-2 fixed mesh implicit diffusion: $\Delta t = 0.025$ s (mean Courant 2, maximum Courant 9.5).
(4) PFEM-2 fixed mesh explicit diffusion: $\Delta t = 0.0125$ s.
(5) PFEM-2 moving mesh: $\Delta t = 0.01$ s.
(6) OpenFOAM®: $\Delta t = 0.002$ s:
   - Solver: icoFoam.
   - Selected maximum Δt to get maximum Courant = 1.
   - Number of PISO correctors: 2.
   - Number of non-orthogonal correctors: 2.
   - U tolerance: $\epsilon_U = 10^{-6}$.
   - p tolerance: $\epsilon_p = 10^{-6}$.

Figure 7 shows that, for the settings described, PFEM-2 is around five times faster than OpenFOAM® using four cores. A scaling factor of around 3 is found, corresponding to a parallel efficiency around 75 percent. At the end of this section a summary of profiling and scalability of PFEM-2 is presented.

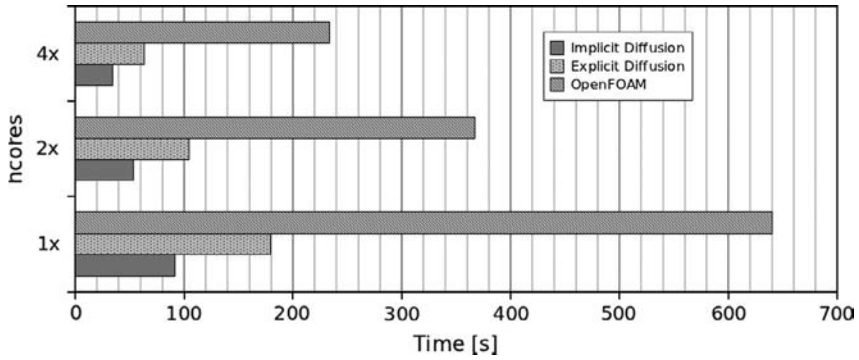
4.2 Flow around a NACA 0012 airfoil

The second example chosen is the popular NACA 0012 airfoil, which was selected to take advantage of the large body of experimental data available.

The first study case corresponds to the profile with an angle at attack of 0° and a Re of 6 millions. The second one is the same profile with an angle of attach of 4° and a Reynolds of 10,000. No turbulence model was included in the simulation.

The airfoil chord is of unit length and the computational domain is large in order not to interfere the boundary conditions with the airfoil. The computational domain and a detail of the mesh for the latter case are shown in Figure 8. For the former case the domain and the mesh employed were similar.

NACA 0012 at zero angle of attack and Re = 6 millions. The reference for this test was the experimental set-up reported in Sheldhal and Klimas (1981).



Note: CPU-times comparison between different PFEM-2 algorithms and OpenFOAM® for one, two and four cores

Figure 7.
Cylinder – Re = 1,000

37,568 linear triangles and 18,989 nodes compose the mesh used. Approximately 212,000 particles were seeded along the entire domain.

The mean pressure coefficient compared with the reference value is shown in Figure 9. This coefficient is obtained as the time average of the ratio between the relative pressure and the reference dynamic pressure as:

$$\overline{C_p} = \frac{\bar{p} - p_\infty}{(1/2)p_\infty U_\infty^2}$$

As expected a good agreement to the experimental results is achieved.

The corresponding drag curve is shown in Figure 10 where it can be verified that the numerical results lay within a 8 percent of difference to the experimental result.

The lift in Figure 11 oscillate around zero, as experimentally observed.

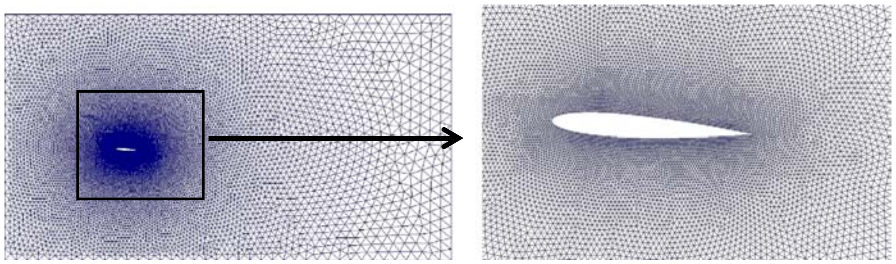


Figure 8.
Computational domain and mesh for four degrees of angle of attack

Note: General view

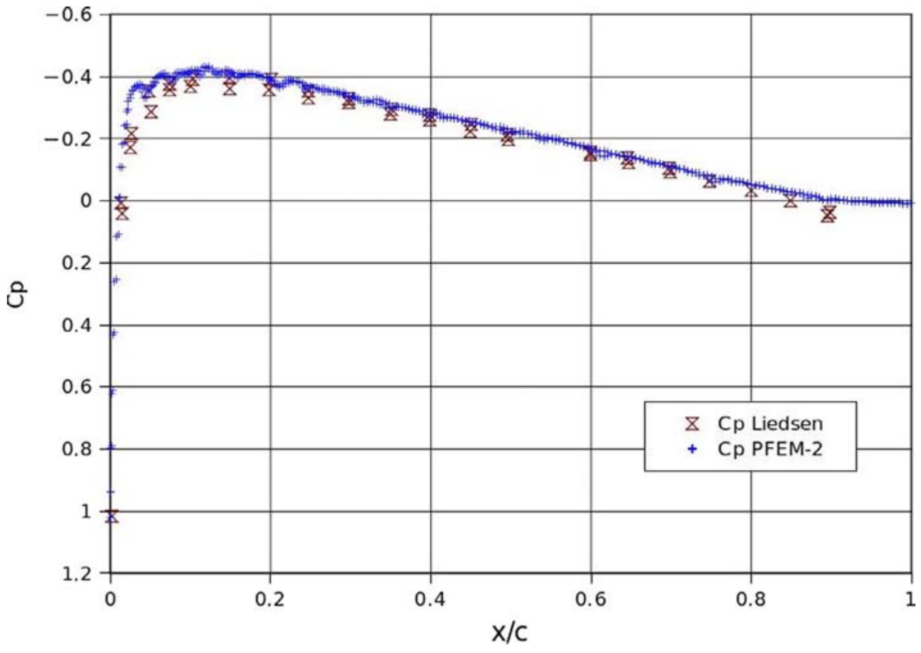
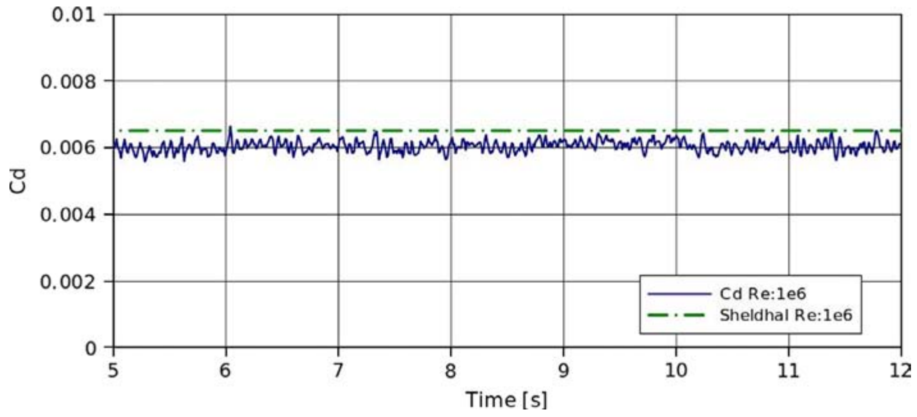


Figure 9.
Mean pressure coefficient for Re = 6 million



Note: The dotted line represents Sheldhal's experimental $\overline{C_d}$

Figure 10.
Drag coefficient for Re = 6 million

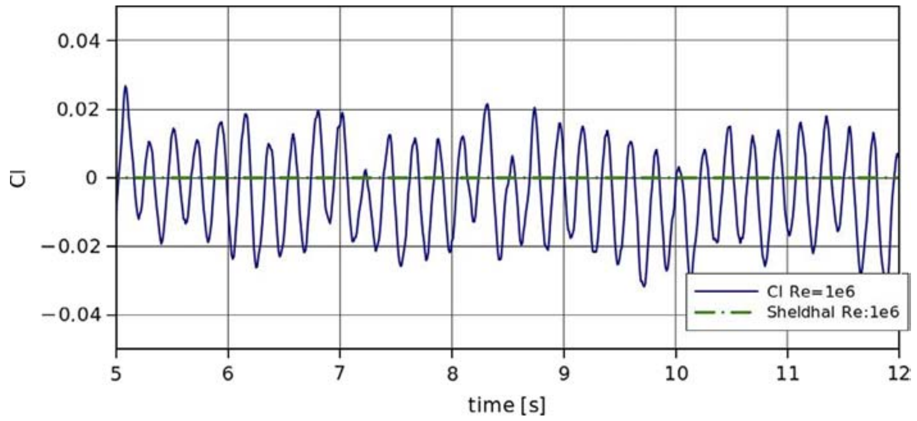


Figure 11.
Lift coefficient for Re = 6 million

The results obtained, appear to provide sufficient accuracy for engineering applications, particularly when the Re is not too high. A turbulence model will be added in the future to improve the accuracy at high Re.

NACA 0012 at 4° angle of attack and Re = 10,000. This example allows enforce the results obtained in the previous sub-section especially when the angle of attack is different from zero as it is normally expected in aeronautical and wind turbine applications. Here the reference is Srinath and Mittal (2010). The mesh is similar to that defined above.

Accuracy results. The results in terms of accuracy show that the mean pressure coefficient follows the reference results in an acceptable way as shown in Figure 12.

The lift is bounded by the limits taken from the reference as shown in Figure 13.

The drag is a little underestimated with its amplitude similar to that published in the reference but with a mean value that is about 9 percent away from that (Figure 14).

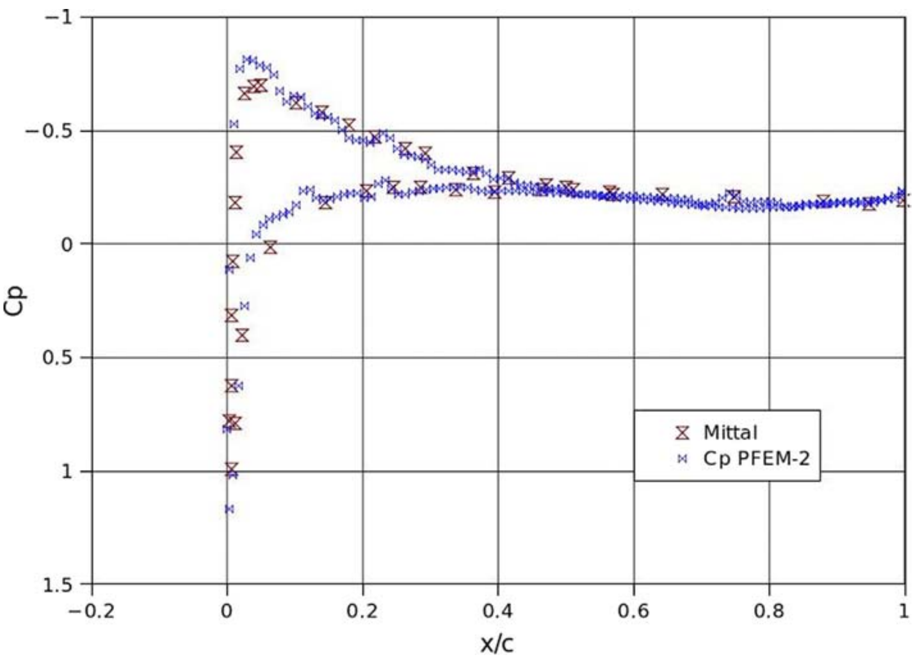


Figure 12.
Mean pressure coefficient for Re = 10 thousands

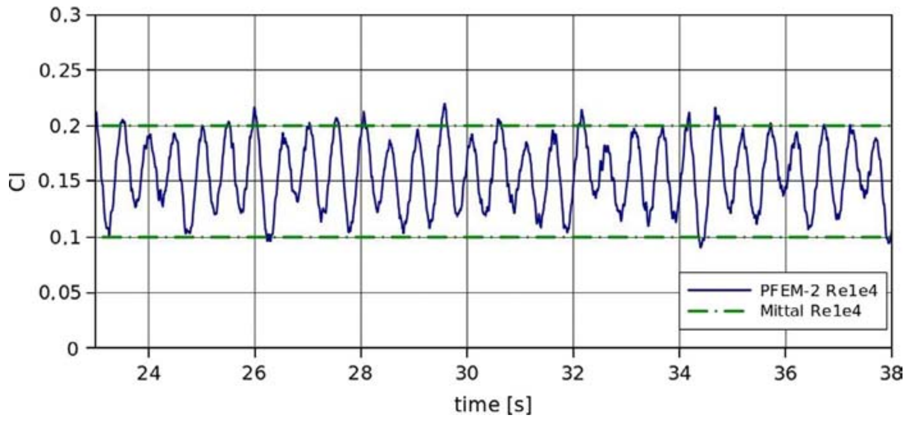
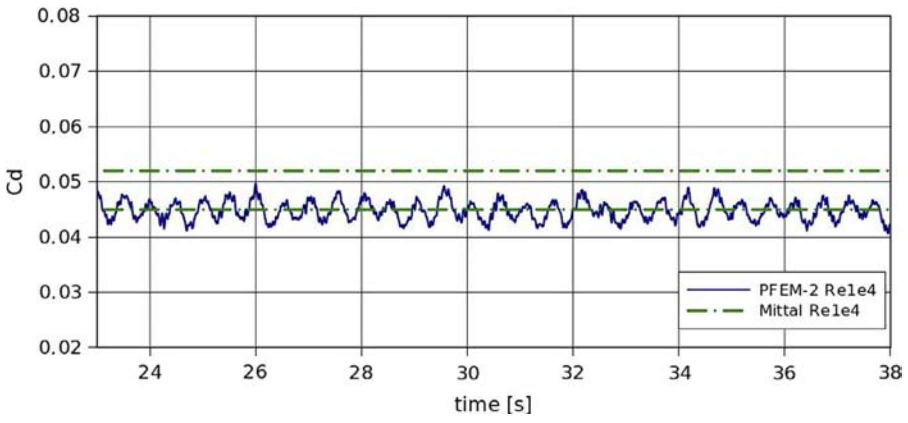


Figure 13.
Lift coefficient for Re = 10 thousands

Note: Mittal's dotted line represents max and min values of lift coefficient

In the Figure 15 some snapshots about the vortex shedding is presented. The frequency of vortex shedding is of 2.1 Hz that compared with that of 2.3 Hz reported in the reference represents a 10 percent of difference.

Time results. From the point of view of efficiency, the present method is faster than OpenFOAM® for both the configurations considered. The speedup factor with respect to OpenFOAM® is around 3 (against the factor 5 obtained for the cylinder test). It is expected that this factor increases with large angle of attack due to an increasing



Note: Mittal's dotted line represents max and min values of drag coefficient

Figure 14.
Drag coefficient for
Re = 10 thousands

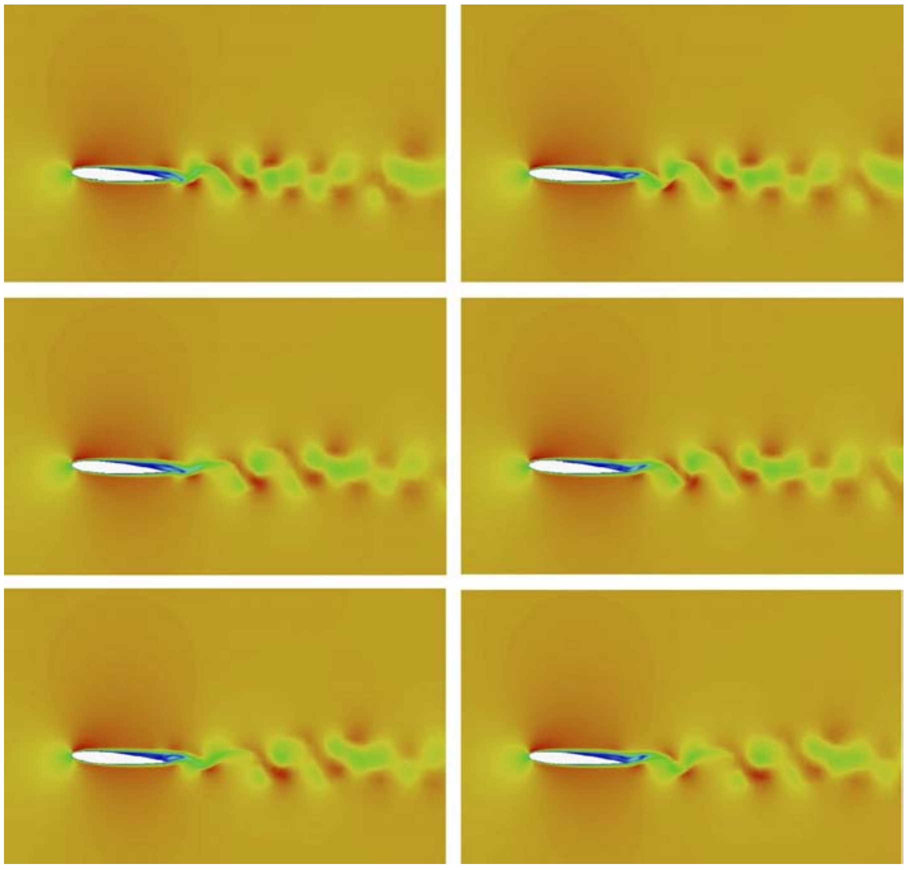


Figure 15.
Snapshots of vortex
shedding for NACA 0012
airfoil at four degrees of
angle of attack and
Re 10 thousands

of the unsteadiness of the flow that introduces more restriction on the time step for OpenFOAM® due to the nonlinearities with no significant restrictions on our Lagrangian method (PFEM-2).

The parameters used in the performance comparison were the following:

(1) CPU: Intel I7-2600 K.
(2) Real time simulated: 1 s.
(3) PFEM-2 fixed mesh implicit diffusion: $\Delta t = 0.015$ s.
(4) PFEM-2 fixed mesh explicit diffusion: $\Delta t = 0.01$ s.
(5) OpenFOAM®: $\Delta t = 0.002$ s:
    - Solver: icoFoam.
    - Selected maximum Δt to get maximum Courant = 1.
    - Number of PISO correctors: 2.
    - Number of non-orthogonal correctors: 2.
    - U tolerance: $\epsilon_U = 10^{-7}$.
    - p Tolerance: $\epsilon_p = 10^{-7}$.

Figure 16 shows the CPU time comparison. The profiling data for the present method is presented graphically in Figure 17. Such data shows that the streamline integration consumes more than half the time. Since this part is highly parallel (thinking in shared memory systems), it is desirable that this portion increases. Similarly, the acceleration computation consumes approximately 6 percent and the correction stage takes 9 percent of the total simulation time.

Therefore, the more scalable part of the code sums about 65 percent of the total computational cost. The remaining part of the code is formed by the updating/projection and the implicit part, i.e. Poisson solver and the diffusive part

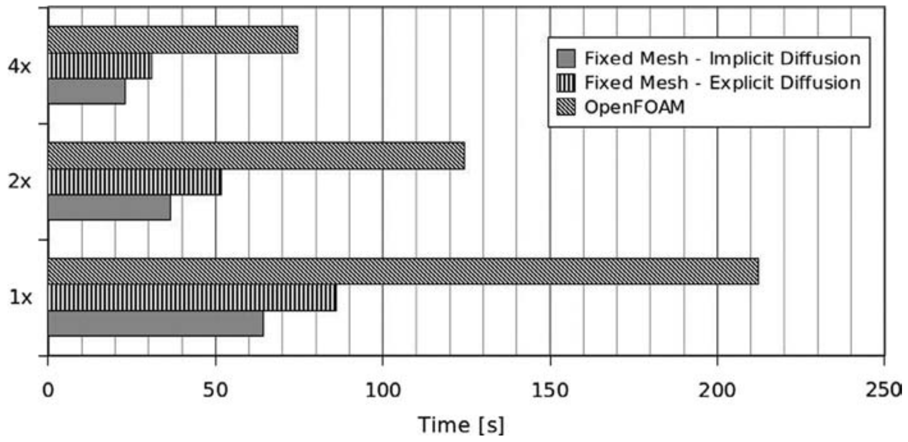
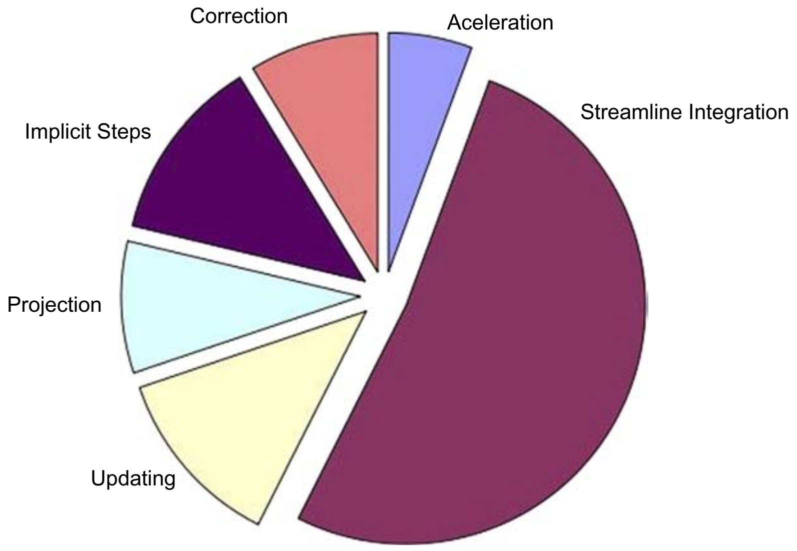


Figure 16.
NACA 0012 – four degrees of attack angle and Re = 10 thousands

Note: CPU-times comparison between different PFEM-2 algorithms and OpenFOAM® for one, two and four cores

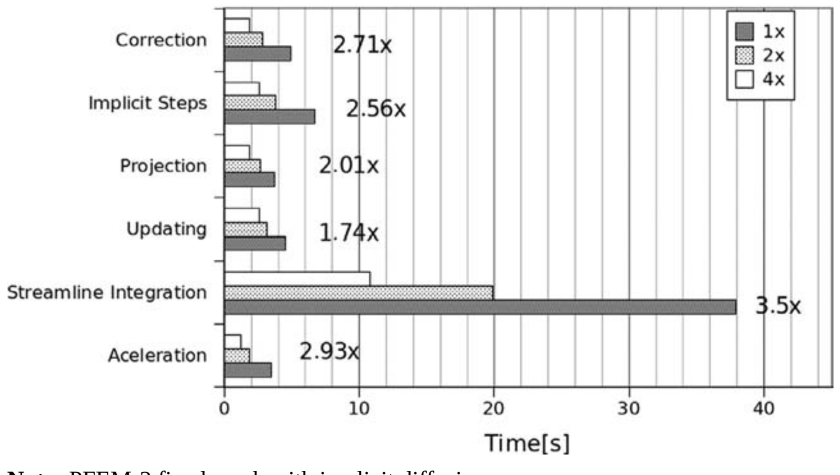


Note: PFEM-2 fixed-mesh implicit diffusion

Figure 17.
Time proportion for each stage of the algorithm

of the momentum equation, here solved using an implicit scheme in order not to be restricted by Fourier number stability considerations.

Considering the performance measured in terms of the scalability (Figure 18) we can note that the scalability of the streamline integration stage is almost ideal while the scalability of the acceleration and of the correction stages is less optimal. The three remaining parts, the implicit, updating and projection only enjoy limited scalability due to the nature of the computations. The updating phase is hard to speed up due to the complex memory management required for removing, adding and reordering the



Note: PFEM-2 fixed-mesh with implicit diffusion

Figure 18.
Scalability for each stage of the algorithm

memory storage. The implicit solution phase has also limited scalability due to inherent limitations in the structure of the calculations.

5. Conclusions and future work

A new version of the novel method PFEM-2 originally introduced in Idelsohn *et al.* (2012) was presented in this paper.

As it was mentioned above the restrictions on performance imposed by the parallel re-meshing and the assembling/solver stages of the implicit part of the moving mesh version of PFEM-2 has motivated to review the method incorporating this fixed mesh version that mainly has the advantages of getting away the re-meshing stage and reducing significantly the cost of the implicit part due to the freezing of the factorized matrices involved in those computations.

This new version shows to reduce even more the CPU time as compared with a popular and fast CFD code OpenFOAM® reaching at present a very attractive position in terms of accuracy and efficiency.

The performance characteristics of the method make it appealing as a fast alternative to Eulerian CFD memory, at least on recent shared memory systems.

Some particular conclusions are:

- $\overline{C_p}$, C_d and C_L curves are reasonably shaped and provide acceptable accuracy for engineering usage. These results were compared with the software (OpenFOAM®), numerical results (Mittal and Kumar, 2001; Ladson, 1988; Park, 1998) and experimental results (Sheldhal and Klimas, 1981), obtaining good approximations in all cases.
- The fluid instabilities previous to fully developed turbulence may introduce some changes in the vortex dynamics that currently our PFEM-2 method is unable to capture. This consideration may justify the small differences observed in our results.
- The performance on the PFEM-2 depends on the choice of the particle-to-mesh transfer operator. No "natural" choice exists in performing this step and hence different possibilities are open for testing. This aspect will be subject of future research.
- The efficiency measured in terms of CPU time for reaching a given final time in the simulation (here takes as 1 s) had shown important advantages of the present method against OpenFOAM® being this new fixed mesh version even more fast than the previous moving mesh one. In this paper a factor between 3 and 5 was obtained at the same level of accuracy. A similar speedup was also measured with respect to a in-house implementation of the implicit fractional step method.
- In terms of scalability the present method has in general a 75 percent of efficiency compared with 65 percent of scalability measured for OpenFOAM® on the reference test system. This 75 percent is achieved with an almost ideal speed-up of the streamline computation phase, reduced by a poor scalability of implicit computations, which however in this method tends to have a small impact. Another limitation to scalability was provided by the memory management routines.

Future work will be oriented to the following:

- Extend this method to multi-physics, especially thermally driven flows, RANS turbulence modeling and multispecies computations.
- Since shared memory parallelism is limited by hardware, a distributed memory parallel code should be developed. It is expected that the parallel performance of streamlines computation will be worst on distributed systems due to higher complexity of the algorithm and to the inherent communication needs.
- The application of the algorithm to multi-fluid configurations represents one of the future targets. Some major redesign of the algorithm is however needed to tackle such goal.

References

Badia, S. and Codina, R. (2008), “Pressure segregation methods based on a discrete pressure Poisson equation: an algebraic approach”, *International Journal for Numerical Methods in Fluids*, Vol. 56, pp. 351-82.

Brackbill, J.U., Kothe, D.B. and Ruppel, H.M. (1988), “Flip: a low-dissipation, particle-in-cell method for fluid flow”, *Computer Physics Communications*, Vol. 48 No. 1, pp. 25-38.

Codina, R. (2000), “Stabilization of incompressibility and convection through orthogonal sub-scales in finite element method”, *Computer Methods in Applied Mechanics and Engineering*, Vol. 190, pp. 1579-99.

Codina, R., Principe, J., Guasch, O. and Badia, S. (2007), “Time dependent subscales in the stabilized finite element approximation of incompressible flow problems”, *Computer Methods in Applied Mechanics and Engineering*, Vol. 196, pp. 2413-30.

Dadvand, P., Rossi, R. and Oñate, E. (2010), “An object-oriented environment for developing finite element codes for multi-disciplinary applications”, *Archives of Computational Methods in Engineering*, Vol. 17 No. 3, pp. 253-97.

Dadvand, P., Rossi, R., Gil, M., Martorell, X., Cotela, J., Juanpere, E., Idelsohn, S.R. and Oñate, E. (2012), “Migration of a generic multi-physics framework to HPC environments”, paper presented at 23rd International Conference on Parallel Computational Fluid Dynamics (in press).

Donea, J. and Huerta, A. (2003), *Finite Element Methods for Flow Problems*, Wiley, New York, NY.

Harlow, F.H. (1964), “The particle in cell computing method for fluid-dynamics”, *Methods in Computational Physics*, Vol. 3, pp. 319-43.

Hughes, T. and Tezduyar, T. (1984), “Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations”, *Computer Methods in Applied Mechanics and Engineering*, Vol. 45, pp. 217-84.

Idelsohn, S.R., Oñate, E. and Del Pin, F. (2004), “The particle finite element method a powerful tool to solve incompressible flows with free-surfaces and breaking waves”, *International Journal for Numerical Methods in Engineering*, Vol. 61, pp. 964-89.

Idelsohn, S.R., Marti, J., Limache, A. and Oñate, E. (2008a), “A unified Lagrangian formulation for elastic solids and incompressible fluids: application to fluid-structure interaction problems via the PFEM”, *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, pp. 1762-76.

Idelsohn, S.R., Marti, J., Souto-Iglesia, A. and Oñate, E. (2008b), “Interaction between an elastic structure and free-surface flows: experimental versus numerical comparisons using the PFEM”, *Computational Mechanics*, Vol. 43, pp. 125-32.

Idelsohn, S.R., Nigro, N.M., Limache, A. and Oñate, E. (2012), "Large time-step explicit integration method for solving problems with dominant convection", *Computer Methods in Applied Mechanics and Engineering*, Vol. 217-220, pp. 168-85.

Ladson, C.L. (1988), "Effects of independent variation of Mach and Reynolds numbers on the low-speed aerodynamic characteristics of the NACA 0012 airfoil section", paper presented at NASA TM 4074.

Larese, A., Rossi, R., Oñate, E. and Idelsohn, S. (2008), "Validation of the particle finite element method (PFEM) for simulation of free surface flows", *Engineering Computations*, Vol. 25 No. 4, pp. 385-425.

Mittal, S. and Kumar, V. (2001), "Flow-induced vibrations of a light circular cylinder at Reynolds numbers 1000 to 10000", *Journal of Sound and Vibration*, Vol. 254 No. 5, pp. 923-46.

Mossaiby, F., Rossi, R., Dadvand, P. and Idelsohn, S.R. (2012), "OpenCL-based implementation of an unstructured edge-based finite element convection-diffusion solver on graphics hardware", *International Journal for Numerical Methods in Engineering*, Vol. 89 No. 13, pp. 1635-51.

Oñate, E., Valls, A. and García, J. (2007), "Modeling incompressible flows at low and high Reynolds numbers via finite calculus-finite element approach", *Journal of Computational Physics*, Vol. 224, pp. 332-51.

Oñate, E., Idelsohn, S.R., Celigueta, M.A. and Rossi, R. (2008), "Advances in the particle finite element method for the analysis of fluid-multibody interaction and bed erosion in free surface flows", *Computer Methods in Applied Mechanics and Engineering*, Vol. 197 Nos 19/20, pp. 1777-800.

OpenCFD (2009), *OpenFOAM®, The Open Source CFD Toolbox, User Guide*, OpenCFD Ltd, Berkshire.

Park, J. (1998), "Numerical solutions of flow past a circular cylinder at Reynolds numbers up to 160", *KSME International Journal*, Vol. 12 No. 6, pp. 1200-5.

Ryzhakov, P., Oñate, E., Rossi, R. and Idelsohn, S.R. (2012), "Improving mass conservation in simulation of incompressible flows", *International Journal for Numerical Methods in Engineering*, Vol. 90 No. 12, pp. 1435-548.

Ryzhakov, P., Rossi, R., Idelsohn, S.R. and Oñate, E. (2010), "A monolithic Lagrangian approach for fluid-structure interaction problems", *Computational Mechanics*, Vol. 46 No. 6, pp. 883-99.

Sheldhal, R.E. and Klimas, P.C. (1981), "Aerodynamic characteristics of seven airfoil sections through 180 degrees angle of attack for use in aerodynamic analysis of vertical axis wind turbines", paper presented at SAND80-2114, Sandia National Laboratories, Albuquerque, New Mexico.

Srinath, D.N. and Mittal, S. (2010), "Optimal aerodynamic design of airfoils in unsteady viscous flows", *Computer Methods in Applied Mechanics and Engineering*, Vol. 199, pp. 1976-91.

Sulsky, D. and Kaul, A. (2004), "Implicit dynamics in the material point method", *Computer Methods in Applied Mechanics and Engineering*, Vol. 193, pp. 1137-70.

Tezduyar, T., Mittal, S., Ray, S. and Shih, R. (1992), "Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements", *Computer Methods in Applied Mechanics and Engineering*, Vol. 95, pp. 221-42.

Corresponding author
Sergio Rodolfo Idelsohn can be contacted at: sergio@cimne.upc.edu