Asociación Argentina de Mecánica Computacional

Mecánica Computacional Vol XXX, págs. 451-483 (artículo completo)
Oscar Möller, Javier W. Signorelli, Mario A. Storti (Eds.)
Rosario, Argentina, 1-4 Noviembre 2011

A NEW APPROACH TO SOLVE INCOMPRESSIBLE NAVIER-STOKES EQUATIONS USING A PARTICLE METHOD

Norberto Nigro[a] , Juan Gimenez[a] , Alejandro Limache[a], Sergio Idelsohn[b], Eugenio Oñate[c], Nestor Calvo[d], Pablo Novara[d], Pedro Morin[e]

[a]*CONICET Researcher at Centro Internacional de Métodos Computacionales en Ingenieria (CIMEC-INTEC), Santa Fe, Argentina.(http://www.cimec.gov.ar/)*

[b]*ICREA Research Professor at the International Center for Numerical Methods in Engineering (CIMNE), Barcelona, Spain.(http://www.cimne.edu.es/)*

[c]*Professor at the Polytechnic University of Catalonia and Director of CIMNE, Barcelona, Spain.*

[d]*Facultad de Ingeniería y Ciencias Hídricas (FICH) de la Universidad Nacional del Litoral, Santa Fe, Argentina. (http://www.fich.unl.edu.ar)*

[e]*CONICET Researcher at Instituto de Matemática Aplicada del Litoral (IMAL), Santa Fe, Argentina.*

Keywords: particle method, incompressible flows, computational fluid dynamics

Abstract. The goal of this work is to present a new methodology to solve computational fluid dynamics (CFD) problems based on a particle method minimizing the usage of mesh based solvers in order to get a potential computational efficiency to get rid of the new challenges of engineering and science. Thanks to the recent advances in hardware, in particular the possibility of using graphic processors (GPGPU), high performance computing is now available if and only if software development gives an important jump to incorporate this technology. Due to the complexity in programming over such a platform the best way to take advantage of its performance rests on the design of numerical methods able to be viewed as cellular automata. In this sense explicit methods seem to be an attractive choice. However it is well known that explicit methods have a severe stability limitation. On the other hand, the spatial discretization of particle methods offers some advantages against others like finite elements or finite volumes in terms of computational costs. The main reasons of this rest on the low dimensionality of this method (particle methods are a zero dimensional representation of the solution of a given set of PDE's while finite elements are 3D and finite volume are part 2D and part 3D). In addition particle methods are generally written in Lagrangian formulation avoiding the necessity of defining a spatial stabilization in convection dominated flows. Finite elements and finite volume are generally designed using an Eulerian formulation with some extra diffusion in the solution due to this stabilization requirement. However, some particle methods often require a mesh to interpolate and also to solve the problem losing some of their advantages in terms of efficiency. Even though there are some methods that do not use any mesh in their formulation, the interpolation methods become very complex introducing errors in the computation with noticeable extra diffusion. Having detected two main limitations of particle methods to solve Navier-Stokes equations for viscous incompressible flows, we propose in this work the following: • to enhance the time integration using an explicit streamline based scheme computed with the old velocity vector allowing to enlarge the time steps of standard explicit schemes in advection dominated flows • in order to minimize the use of mesh based solvers, the velocity predictor and its correction is formulated purely on the particles as any spatial collocation method using a gradient recovery technique to include the pressure gradient and the viscous terms. This method is written in a Lagrangian formulation in a segregated way like a fractional step method. The computation of the predicted fractional velocity and its correction is done using our proposal explained above, i.e. streamline in time collocation in space scheme. On the other hand the pressure correction (Poisson solver) is carried out using a FEM like method. This method may be implemented in two ways: • the mobile mesh version: where the particles represent the mesh nodes and a permanent remeshing is needed in order to avoid the severe restriction in the time steps imposed by the mesh motion. Remember that the mesh is only used to solve the Poisson problem for the pressure correction. • the fixed mesh version: where there is a background mesh to do some computations, mainly for the pressure, and a certain amount of particles that move in a Lagrangian way transporting the velocity. Some interpolation between the particles and the fixed mesh is needed but the remeshing is completely avoided. This paper presents this novel approach built with the above mentioned features and shows some results to demonstrate its ability in terms of stability and accuracy with a high potential to be optimized in order to solve the challenging engineering problems of the next decades.

INTRODUCTION

The main target of this work is to look for algorithms to simulate CFD problems as fast as possible, improving the current performance of general-purpose commercial codes. This goal does not mean yet obtaining Real Time CFD solution but this goal is on this way, with the aim of changing days of simulations for hours of simulations making feasible the present challenging demands of engineering design.

Even though implicit time integration schemes are preferred in the literature against explicit ones, the latter are in a better position attending to the present of hardware technology that is oriented to the usage of general purpose graphic processor units (GPGPU).

On the other hand it is not obvious that an Eulerian approach is better in terms of efficiency and accuracy than a Lagrangian one. At least in this paper we try to put into question this asseveration.

One of the main drawbacks of the explicit integration using an Eulerian formulation is the restricted stability of the solution with the time steps and the spatial discretization (Zienkiewicz, et al. 2006; Donea, J. and Huerta, A. 2003). For the incompressible Navier-Stokes equations, it is well known that the time step to be used in the solution of the momentum equations is stable only for time steps smaller than two critical values: the Courant-Friedrichs-Lewy (CFL) number and the Fourier number. The first one concerns the convective terms and the second one the diffusive ones. Both numbers must be less than one to have stable algorithms. For convection dominant problems like high Reynolds number flows, the condition CFL<1 becomes crucial and limits the use of explicit methods or makes the solution scheme far from being efficient.

The possibility to perform parallel processing and the recent upcoming of new processors like GPGPU (Ehrenstein, G.2008) increase the possibilities of the explicit integration in time due to the facility to parallelize explicit methods having results with speed-up closed to one.

Although the incompressible condition cannot be solved explicitly, except by introducing a small compressibility in the flow, the momentum conservation equations with an explicit integration together with a parallel processing may reduce drastically the computing time to solve the whole problem provided that a large time step may be preserved independently to the discretization in space. This paper is concerned with this objective: to perform explicit integration of the momentum equations without the CFL<1 restriction.

Over the last 30 years, computer simulation of incompressible flows has been mainly based on the Eulerian formulation of the fluid mechanics equations on continuous domains (Hughes, T.J.R. et al 1986). 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 (Tezduyar, T.E. et al 1992a) 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.

Standard formulations to solve the incompressible Navier-Stokes equations may be split in two classes. In a first class there are those methods in which all the equations (momentum and mass conservation) are solved together. An implicit integration in time is performed in both the momentum equations and the mass balance equation. They are called Monolithic Methods. In a second-class we find those methods called Fractional-Step Methods, or Pressure-Segregation Methods (Chorin, A.J. ,1967) because they solve the problem in two steps: one explicit step for the momentum equations and a second implicit step for the mass balance equation. The advantage of monolithic methods is the possibility to have stable solutions with large time steps. Their main disadvantage is the need to solve very large non-linear system of ill-conditioned coupled equations. On the other hand, the big advantage of explicit integration is the simplicity and the scalability in parallel processors. Until now, the big disadvantage was to be conditionally stable for relative small time steps and also to have stability with spatial discretization dependency.

In this paper we will present a pressure-segregation method with an explicit time integrator without the CFL restriction. This allows large time steps independent of the spatial discretization having equal or better precision that an implicit integration.

The idea is to use the information we have at time $t = t^n$ in the velocity streamlines as well as in the acceleration streamlines to update the particle position as well as the velocity in a Lagrangian frame. The method may be used with moving or fixed meshes.

1 EXPLICIT INTEGRATION FOLLOWING THE VELOCITY AND THE ACCELERATION STREAMLINES.

Let $\mathbf{x}_p$ be the vector defining the position of a particle in a 3D space, function of the time t that we will write for simplicity $\mathbf{x}_p^t$. At time $t = t^n$ we will write $\mathbf{x}_p^n$, at time $t = t^n + \Delta t = t^{n+1}$ we will write $\mathbf{x}_p^{n+1}$ 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 $\mathbf{A}^{n+t}(\mathbf{x}_p^{n+t})$ vectors defining the velocity and the acceleration respectively of a particle $\mathbf{x}_p$ at any time t^{n+t}

$$\begin{cases} \mathbf{V}^{n+t}(\mathbf{x}_p^{n+t}) = \dfrac{D\mathbf{x}_p^{n+t}}{Dt} & (1.1) \\ \mathbf{A}^{n+t}(\mathbf{x}_p^{n+t}) = \dfrac{D\mathbf{V}_p^{n+t}}{Dt} & (1.2) \end{cases}$$

where $\dfrac{D\phi}{Dt}$ represents the material (Lagrangian) derivative in time of any function f. The material derivative is connected with the spatial derivative by the convective terms: $\dfrac{D\phi}{Dt} = \dfrac{\partial\phi}{\partial t} + V_i \dfrac{\partial\phi}{\partial x_i} = \dfrac{\partial\phi}{\partial t} + \mathbf{V}^T \nabla\phi$

In all initial value problems like the transient Navier-Stokes equations, the time solution of a problem consists in: knowing 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 (1.1) and (1.2):

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \displaystyle\int_n^{n+t} \mathbf{V}^\tau(\mathbf{x}_p^\tau)\, d\tau & (1.3) \\ \mathbf{V}^{n+t}(\mathbf{x}_p^{n+t}) = \mathbf{V}^n(\mathbf{x}_p^n) + \displaystyle\int_n^{n+t} \mathbf{A}^\tau(\mathbf{x}_p^\tau)\, d\tau & (1.4) \end{cases}$$

The accuracy of the results will depend to a great extent in the accuracy of the discretization of the velocity and acceleration in the space, but also in the approximation introduced in the integration of (1.3) and (1.4). In this paper we will be concern with the time integration only, being possible to use any space discretization to achieve analogous results.

1.1 Time integration of the velocity:

Equation (1.3) may be approximated in different ways. The simplest one is the constant velocity explicit integration in which the velocity is considered constant in the whole time interval with the value of the velocity at the initial position:

$$\mathbf{x}_p^{n+1} \approx \mathbf{x}_p^{n} + \mathbf{V}^{n}(\mathbf{x}_p^{n})\,\Delta t \tag{1.5}$$

Another possibility is the linear velocity implicit integration in which the velocity is considered with a linear variation between t^n and t^{n+1}:

$$\mathbf{x}_p^{n+1} \approx \mathbf{x}_p^{n} + \mathbf{V}^{n}(\mathbf{x}_p^{n})\,(1-\theta)\Delta t + \mathbf{V}^{n+1}(\mathbf{x}_p^{n+1})\,\theta\,\Delta t \tag{1.6}$$

with θ being a parameter varying between 0 (explicit) and 1 (fully implicit).

The difficulty of (1.6) is the fact that velocity $\mathbf{V}^{n+1}$ is unknown. This means that is part of the variables to be solved. For this reason implicit methods introduce a non-linear system of equations.

The question is: are there other explicit formulations better than the constant velocity approximation used in (1.5)? Of course there are. We can use previous time steps, like t^{n-1}, t^{n-2}, etc. to approximate high order time curves. We propose here a different way to improve the explicit integration. The idea 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

$$\mathbf{x}_p^{n+t} \approx \mathbf{x}_p^{n} + \int_{n}^{n+t} \mathbf{V}^{n}(\mathbf{x}_p^{\tau})\,d\tau \tag{1.7}$$

Equation (1.7) is explicit because we are using only information at time step t^n. In this case we are not using a constant else a linear, approximation of the velocity field. We are using the same high order approximation the velocity field has at time t^n. The only difference with the exact integration (1.3) is that here we are performing the integral (inside each time step) following a pseudo trajectory of the particles calculated with the velocity streamline, instead of following the true trajectory (Fig. 1.1). It must be noted that in the stationary case, the particle position evaluated with the velocity streamlines and the trajectory are coincident. This time integration will be named *Explicit Integration following the Velocity Streamlines* **(X-IVS).**

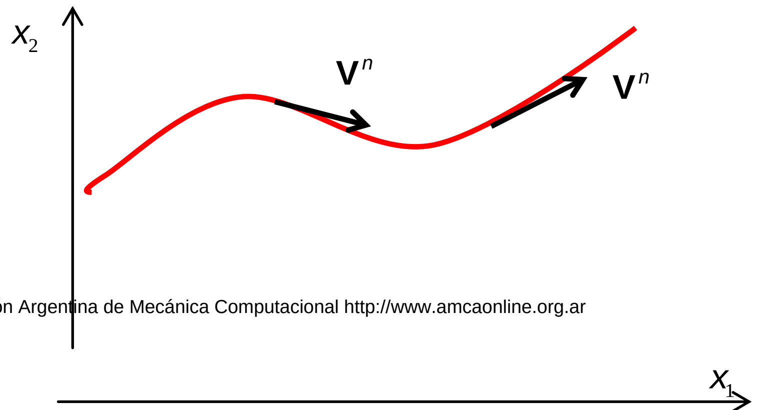


Fig. 1.1 Integration following the Velocity Streamlines

1.2 Time integration acceleration:

In classical explicit integration, equation (1.4) is replaced by:

$$\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \mathbf{A}^n(\mathbf{x}_p^n)\Delta t \tag{1.8}$$

The value of $\mathbf{x}_p^{n+1}$ may be evaluated with any of the possibility described before:

a) the fully explicit case, that is using (1.5), reduce to:

$$\begin{cases} \mathbf{x}_p^{n+1} \approx \mathbf{x}_p^n + \mathbf{V}^n(\mathbf{x}_p^n)\,\Delta t \\ \mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \mathbf{A}^n(\mathbf{x}_p^n)\Delta t \end{cases} \tag{1.9}$$

b) the X-IVS case, that is using (1.7), remains:

$$\begin{cases} \mathbf{x}_p^{n+1} \approx \mathbf{x}_p^n + \int_n^{n+1} \mathbf{V}^n(\mathbf{x}_p^t)\,dt \\ \mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \mathbf{A}^n(\mathbf{x}_p^n)\Delta t \end{cases} \tag{1.10}$$

In implicit (linear) integration equation (1.4) is replaced by

$$\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \mathbf{A}^n(\mathbf{x}_p^n)(1-\theta)\Delta t + \mathbf{A}^{n+1}(\mathbf{x}_p^{n+1})\theta\,\Delta t \tag{1.11}$$

which also may be used with any of the previous time integrations for the particle position.

However, we will propose something new for evaluating the velocity: the idea proposed in (1.7) 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 (1.4) by:

$$\mathbf{V}^{n+t}(\mathbf{x}_p^{n+t}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \int_n^{n+t} \mathbf{A}^n(\mathbf{x}_p^\tau)\,d\tau \tag{1.12}$$

Equation (1.12) represents an integration following the acceleration streamlines (See Fig. 1.2) obtained at time t^n. This may be solved using any of the particle position integrations described before. For consistence, we will use the X-IVS method described in (1.7):

$$\begin{cases} \mathbf{x}_p^{n+t} \approx \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n(\mathbf{x}_p^\tau)\, d\tau \\ \mathbf{V}^{n+t}(\mathbf{x}_p^{n+t}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \int_n^{n+t} \mathbf{A}^n(\mathbf{x}_p^\tau)\, d\tau \end{cases} \tag{1.13}$$

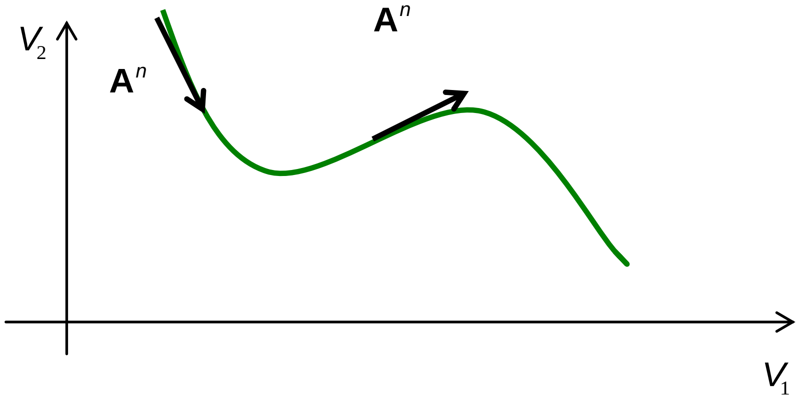


Fig. 1.2 Integration following the Acceleration Streamlines

We must note that equations (1.13) are still explicit because they are using the velocity and acceleration at time t^n (Fig.1.2). Nevertheless, any constant or linear variation in time neither the velocity nor the acceleration are not assumed, as it is the standard assumption in classical explicit or implicit integration scheme. This approach will be named *Explicit Integration following the Velocity and Acceleration Streamlines* **(X-IVAS).**

Another possibility to perform the integration is via the so-called *Characteristic Methods* (Jos Stam 1999; O. Pironneau and M. Tabata. 2010). We can summarize the method using $\theta = 1$ in (2.11):

$$\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) \approx \mathbf{V}^n(\mathbf{x}_p^n) + \mathbf{A}^{n+1}(\mathbf{x}_p^{n+1})\ \Delta t \tag{1.14}$$

and then evaluating the term $\mathbf{V}^n(\mathbf{x}_p^n)$ by backward integration over the streamlines of equation (1.3):

$$\mathbf{x}_p^{n+1-t} \approx \mathbf{x}_p^{n+1} - \int_{n+1}^{n+1-t} \mathbf{V}^n(\mathbf{x}_p^\tau)\, d\tau \tag{1.15}$$

Finally the value of $\mathbf{V}^n(\mathbf{x}_p^n)$ is obtained by interpolating the velocity $\mathbf{V}^n$ at the position $\mathbf{x}_p^n$.

It must be noted that in spite of the similarity of idea between the Characteristic Methods and the X-IVAS method presented here, the concept and the results are totally different. The Characteristic Method is a X-IVS integration with an implicit integration of the acceleration for $\theta = 1$. Therefore, the velocity is not integrated following the acceleration streamline and, furthermore, the need of an interpolation over the velocity field to evaluate $\mathbf{V}^n(\mathbf{x}_p^n)$ introduces important errors as shown in the examples.

2 TOWARDS AN ANALYTICAL STREAMLINE BASED TIME INTEGRATION.

This section deals with the goal of getting algorithms as closed as possible to follow particles minimizing the integration error and also reducing to a minimum the number of substeps to complete the enlarged time step proposed in the PFEM-2 method. The ideal situation should be to go on jumping between elements with a minimum amount of computation inside them during each time step. In this way we proposed to find the time and the location of the particle when it is just crossing between two elements, the element in which it was at the beginning of the time step and its neighbor found during its path. Inside each element the analytical expression for the particle trajectory is used reducing the error to a minimum induced by the time integration and by the space approximation of the variables. The former is due to the time at which the fields are assessed, normally the old one for explicit schemes, and the later for the linear interpolation normally used in massively computation.
Briefly, the algorithm presented in this section should be able to solve for each particle the following ODE :

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{V} \tag{2.1}$$

over a background mesh where the velocity field is defined $\mathbf{V}(\mathbf{x})$ with initial conditions like $\mathbf{x}_p(t=0)$.The following figure shows a typical problem where the mesh and the velocity field over it is observed, with dot lines following the trajectory of a given particle.

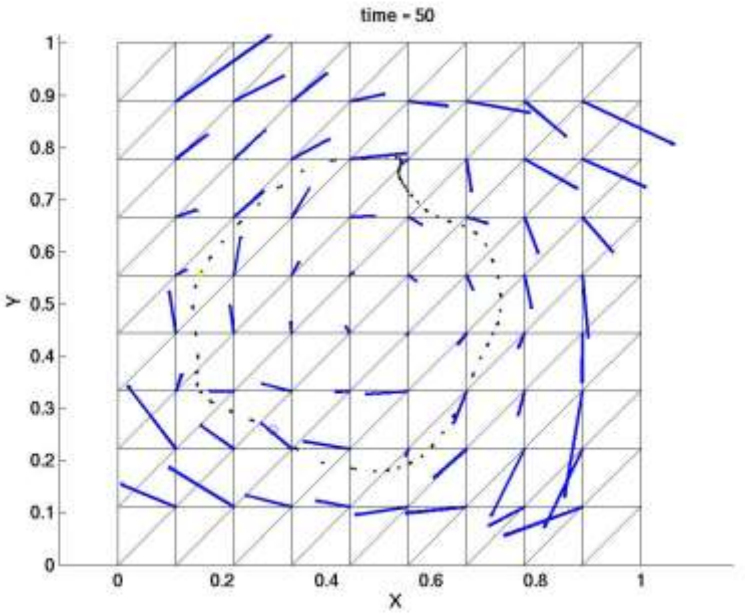


Fig. 2.1: Analytical computation of the particle trajectory assuming a linear velocity field interpolation inside each element in the mesh

Inside each triangle the velocity field is interpolated linearly so it is possible to get a closed analytical expression taking this assumption. The main idea of our strategy consists in detecting the event for which the particle initially placed at P at time t^n crosses the boundary of the triangle e entering to the element e'. This event is marked as Q in the following figure and it is computed using the analytic expression of the particle trajectory that moves with the linear velocity field given by:

$$
\begin{aligned}
&\mathbf{v}(\mathbf{y}^t) = \sum_{j=a,b,c} \mathbf{v}_j \mathbf{N}_j(\mathbf{y}^t) \\
&\mathbf{N}_j(\mathbf{y}^t) = \lambda_j(\mathbf{y}^t) \\
&t \in (t^n, t^n + \Delta t)
\end{aligned}
\tag{2.2}
$$

As it was mentioned the strategy is based on the first occurrence of crossings of particle paths through element boundaries.

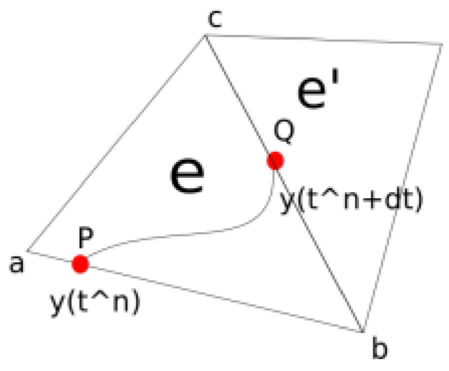


Fig. 2.2: Detecting the crossing event in the trajectory of a particle

Several situations have been taken into account. One of them is plotted in the following figure.

The particle seems to go out of the current element e through the edge b-c but before reaching this boundary and very close to it, the particle changes its direction exiting through the edge c-a. To look for these events a bisection non-linear root finding algorithm was used. The following subsection enters in some details about the mathematics involved in the quasi-analytical time integration of particle path.

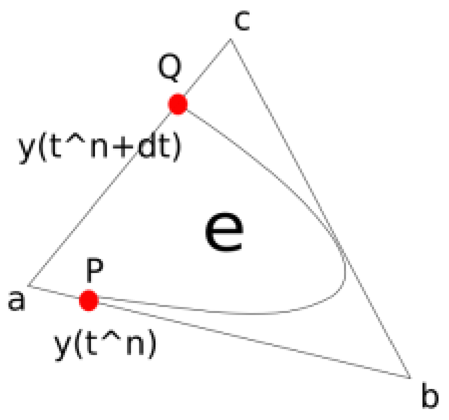


Fig 2.3: Bypassing pathologies in the crossing event detection.

2.1 Exact computation of particle path

We describe here how we computed the path followed by a massless particle starting at a point in an element, moved by a piecewise linear velocity. More precisely, we consider a triangle *T*, described by the coordinates of its vertices *X* (each column representing one vertex), a linear velocity *V* (each column representing the velocity at each vertex), and an initial point λ^0 (column vector) in baricentric coordinates.
That is, the real world coordinates of the initial point are obtained multiplying X by λ^0, i.e., $\mathbf{x}^0 = \lambda^0\mathbf{X}$.
The goal is to compute the exact path of the particle starting at $\mathbf{x}^0$ given by the velocity field **V**. If **v(x)** denotes the velocity at the point **x**, then the particle path is the solution to the following system of ODEs:

$$\dot{\mathbf{x}}(t) = \mathbf{v}(\mathbf{x}(t)), \qquad \mathbf{x}^0 = \lambda^0\mathbf{X} \tag{2.3}$$

It is more convenient to work with the baricentric coordinates. Since $\mathbf{x}(t) = \lambda(t)\mathbf{X}$ and $\mathbf{v}(\mathbf{x}(t)) = \lambda(t)\mathbf{V}$ the system reads

$$\mathbf{X}\dot{\lambda}(t) = \mathbf{V}(\lambda(t)), \qquad \lambda(t=0) = \lambda^0 \tag{2.4}$$

Since $\lambda_1(t) + \lambda_2(t) + \lambda_3(t) = 1$ we replace $\lambda_3(t) = 1 - \lambda_1(t) - \lambda_2(t)$ and obtain the following system for $\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}$:

$$\mathbf{M_X}\dot{\lambda}(t) = \mathbf{M_V}\lambda(t) + \mathbf{V}_3\,; \qquad \lambda(0) = \lambda^0 \tag{2.5}$$

where $\mathbf{M_X} = [\mathbf{X}_1 - \mathbf{X}_3 \quad \mathbf{X}_2 - \mathbf{X}_3]$ and $\mathbf{M_V} = [\mathbf{V}_1 - \mathbf{V}_3 \quad \mathbf{V}_2 - \mathbf{V}_3]$ with $\mathbf{X}_\mathrm{i}$, $\mathbf{V}_\mathrm{i}$ denoting the i-th column of $\mathbf{X}$,$\mathbf{V}$ respectively. If the element is not degenerate $\mathbf{M_X}$ is invertible and the problem is equivalent to:

$$\dot{\lambda}(t) = \underbrace{\mathbf{M_X}^{-1}\mathbf{M_V}}_{\mathbf{A}}\lambda(t) + \underbrace{\mathbf{M_X}^{-1}\mathbf{V}_3}_{\mathbf{b}}\,; \qquad \lambda(0) = \lambda^0 \tag{2.6}$$

or

$$\dot{\lambda}(t) = \mathbf{A}\lambda(t) + \mathbf{b}\,; \qquad \lambda(0) = \lambda^0 \tag{2.7}$$

The exact solution of (2.7) is given by

$$\lambda(t) = \left(e^{t\mathbf{A}} - I\right)\mathbf{A}^{-1}\mathbf{b} + e^{t\mathbf{A}}\lambda^0 \tag{2.8}$$

with

$$e^{t\mathbf{A}} = I + t\mathbf{A} + \frac{t^2\mathbf{A}^2}{2!} + \frac{t^3\mathbf{A}^3}{3!} + \frac{t^4\mathbf{A}^4}{4!} + \cdots = \sum_{n=1}^{\infty} \frac{t^n\mathbf{A}^n}{n!} \tag{2.9}$$

The formula $\left(e^{t\mathbf{A}} - I\right)\mathbf{A}^{-1}$ does not make sense when A is not invertible, but it should be interpreted in the following sense:

$$\left(e^{t\mathbf{A}} - I\right)\mathbf{A}^{-1} = t + \frac{t^2\mathbf{A}}{2!} + \frac{t^3\mathbf{A}^2}{3!} + \frac{t^4\mathbf{A}^3}{4!} + \cdots = \sum_{n=1}^{\infty} \frac{t^n\mathbf{A}^{n-1}}{n!} \tag{2.10}$$

The question now arises on how to compute these two series in a fast way. The answer is not so complicated and we resort to the eigenvalues $\sigma(\mathbf{A})$ of $\mathbf{A}$. There are three cases:

- The matrix A has two real eigenvalues of multiplicity 1

$$\mathbf{A} = \mathbf{PDP}^{-1} \qquad \mathbf{D} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}$$

- The matrix A has only one real eigenvalue with multiplicity 2

$$\mathbf{A} = \mathbf{PDP}^{-1} \qquad \mathbf{D} = \begin{bmatrix} \sigma & 1 \\ 0 & \sigma \end{bmatrix}$$

- The matrix A has two complex conjugate eigenvalues. In this case the (real) Jordan decomposition looks like

$$\mathbf{A} = \mathbf{PDP}^{-1} \qquad \mathbf{D} = \begin{bmatrix} \alpha & \beta \\ -\beta & \alpha \end{bmatrix}$$

Next, the root finding algorithm applied to the three baricentric coordinates lambda is applied with some implementation details not included here for brevity reasons.

2.2 A measure of accuracy and efficiency of this analytical particle path computation.

In this subsection we focus on the accuracy and the efficiency of this novel methodology. Even though a lot of work should be done in order to confirm the usefulness of this methodology here an academic example is presented with the target put on the comparison against other standard integration techniques. Here we compare this integration technique with a standard 4th order Runge-Kutta and the current time integrator used in PFEM2 that is a sub-cycled Forward Euler.

The example chosen is the Archimedes spiral, a mathematical function defined as:

$$x = r\cos(\theta) \quad ; \quad y = r\sin(\theta)$$

$$\dot{\theta} = \frac{1}{4} \quad ; \quad \dot{r} = \frac{1}{80}\dot{\theta}$$

$$\theta = \theta(t=0) + \int_0^T \dot{\theta}\,dt = \theta(t=0) + \frac{1}{4}t$$

$$r = r(t=0) + \int_0^T \dot{r}\,dt = r(t=0) + \frac{1}{320}t \tag{2.11}$$

The following figure shows the error for the three methods compared using different time steps. It is evident for this example the advantages of using the strategy proposed in this section. As an additional remark the current time integrator used in PFEM-2 is far enough in accuracy respect to this quasi-analytical method, therefore, the future incorporation of this technique seems to be very promising.

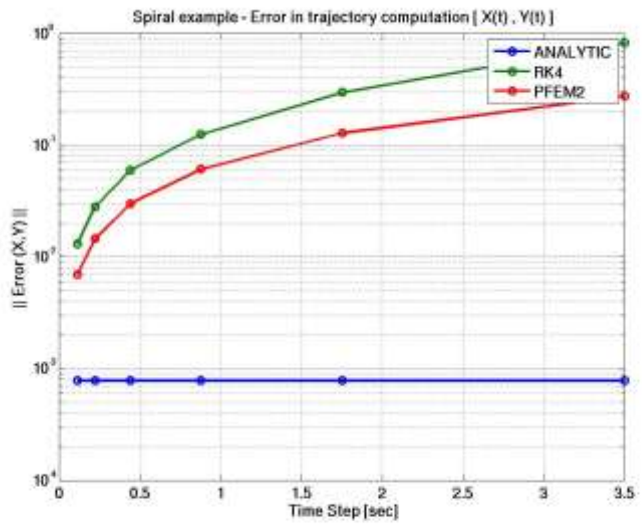


Fig 2.4: Sensitivity of the error integration with time step. Comparison among the analytical technique presented here against 4th order Runge-Kutta and current PFEM2 integrator.

The following figure shows a comparison between the analytic technique presented in this section and the current PFEM2 time integrator when the latter is refined with an increasing number of sub-cycles.
It should be noted the constant error for the analytic path computation with the choice of the time step. This behavior is an evidence of the feature of this time integration scheme, it may be defined as a Discrete Event System Specification (DEVS). Therefore the error does not depend on the time step selection else in the accuracy of the crossing event detection.

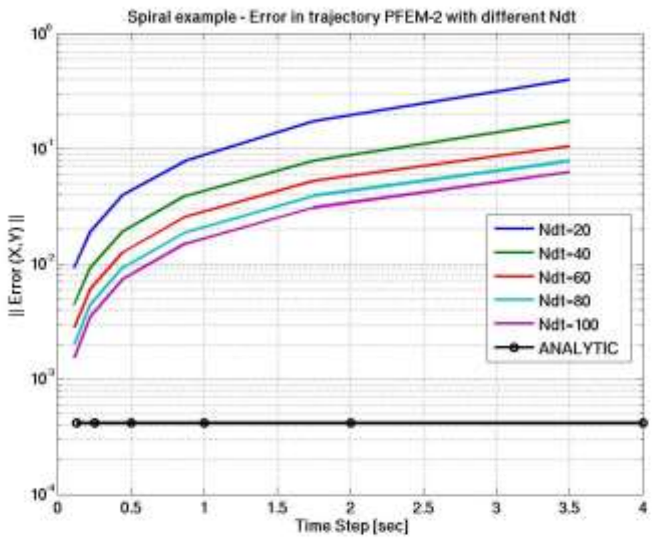


Fig 2.5: Sensitivity of the PFEM2 time integration error with time step. Comparison with the analytic technique presented here.

In the above figure it may be noted that even though the number of sub-cycles increases up to 5 times its normal value the accuracy is far from that of the analytical technique introduced in this section. Another statement to be addressed is the relation of the time step with the Courant number. According to the original goal of PFEM2 of increasing the Courant number as much as possible behind the normal stability criterion of Explicit Eulerian formulation it should be mentioned that the time step of 4 second correspond to Courant number of approximately 10. Therefore the advantage of switching the time integration scheme for that presented here is very important.

The following figure shows the comparison in terms of the CPU time among the above mentioned techniques.

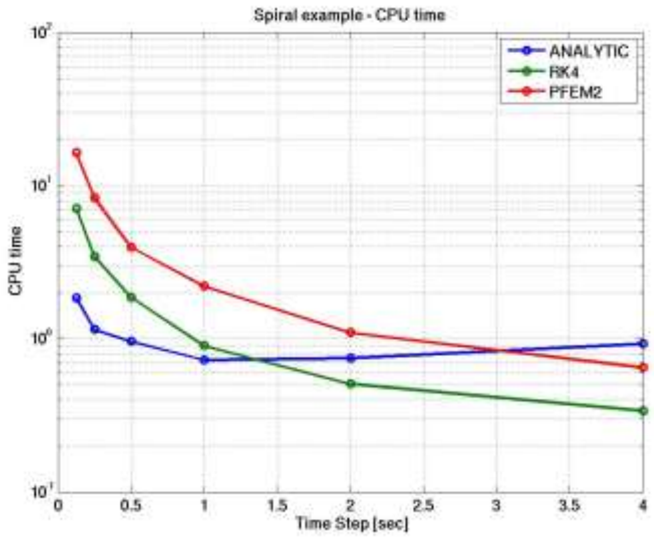


Fig 2.6: CPU time comparison among against 4th order Runge-Kutta, the current PFEM2 integrator and the analytic version.

It may be concluded that even though the analytic time integration seems to be more expensive mainly for large time steps the ratio among the three techniques is not so high and probably with further work in improving the efficiency of this promising technique it may be possible to reduce its computational costs below that of the current PFEM2 version.

2.3 Some preliminary conclusions of this particle path computation technique.

The design of an analytic time integration scheme of particle path and its extension to the computation of analytic velocity field based on a linear interpolation of the acceleration field, not presented here, not only show to be possible also gives some nice features. One of them is the characteristic of being a DEVS technique that put this computation in competitive terms towards real time applications. Another item to be highlighted is the large improvement in terms of accuracy with a little extra computational cost. Finally the extension to 3D problems is possible and specially promising is the incorporation of much more equations to the flow model, like in multi-physics applications.

3 THE EXPLICIT INTEGRATION FOLLOWING THE VELOCITY AND ACCELERATION STREAMLINES FOR THE CONVECTION-DIFFUSION EQUATION

In order to test the validity of the X-IVAS method, we will start with the scalar convection-diffusion equation:
In Eulerian frame

$$\frac{\partial T}{\partial t}+\mathbf{V}^T\nabla T=\nabla\bullet(\kappa\nabla T)+Q \qquad (3.1)$$

In Lagrangian frame ((((((((

$$\frac{DT}{Dt}=\nabla\bullet(\kappa\nabla T)+Q \qquad (3.2)$$

(3.2)

For a finite time step both equations become:

$$T^{n+1}(\mathbf{x})=T^n(\mathbf{x})+\int_n^{n+1}\left\{-\mathbf{V}^T\nabla T^t(\mathbf{x})+\nabla\bullet[\kappa\nabla T^t(\mathbf{x})]+Q^t(\mathbf{x})\right\}]dt \qquad (3.3)$$

(3.3)

in the Eulerian version and

$$\begin{cases} T^{n+1}(\mathbf{x}_p^{n+1})=T^n(\mathbf{x}_p^n)+\displaystyle\int_n^{n+1}\left\{\nabla\bullet[\kappa\nabla T^t(\mathbf{x}_p^t)]+Q^t(\mathbf{x}_p^t)\right\}dt \\ \mathbf{x}_p^{n+t}=\mathbf{x}_p^n+\displaystyle\int_n^{n+t}\mathbf{V}^\tau(\mathbf{x}_p^\tau)d\tau \end{cases} \qquad (3.4)$$

in the Lagrangian frame.

The explicit integration of the convection-diffusion equation in the Eulerian version is:

$$T^{n+1}(\mathbf{x})=T^n(\mathbf{x})+\left\{-\mathbf{V}^T\;\nabla T^n(\mathbf{x})+\nabla\bullet[\kappa\;\nabla T^n(\mathbf{x})]+Q^n(\mathbf{x})\right\}\Delta t$$

and the implicit θ version reads:

$$T^{n+1}(\mathbf{x})=T^n(\mathbf{x})+\left\{-\mathbf{V}^T\nabla T^{n+\theta}(\mathbf{x})+\nabla\bullet[\kappa\nabla T^{n+\theta}(\mathbf{x})]+Q^{n+\theta}(\mathbf{x})\right\}\Delta t \qquad (3.5)$$

In the convection-diffusion equation, the acceleration vector is not present, at least in its standard definition $\mathbf{A}=\frac{D\mathbf{V}}{Dt}$, but the term $h=\frac{DT}{Dt}$ has the form of an acceleration. Using the previous ideas of the X-IVAS method the following explicit integration is proposed for the convection-diffusion equations using a Lagrangian frame: (3.6)

$$\begin{cases} T^{n+1}(\mathbf{x}_p^{n+1})=T^n(\mathbf{x}_p^n)+\displaystyle\int_n^{n+1}\left\{\nabla\bullet[\kappa\nabla T^n(\mathbf{x}_p^t)]+Q^n(\mathbf{x}_p^t)\right\}dt \\ \mathbf{x}_p^t=\mathbf{x}_p^n+\displaystyle\int_n^{t}\mathbf{V}^n(\mathbf{x}_p^\tau)d\tau \end{cases} \qquad (3.7)$$

Note that both equations are totally explicit.

4 THE EXPLICIT INTEGRATION FOLLOWING THE VELOCITY AND ACCELERATION STREAMLINES FOR THE INCOMPRESSIBLE NAVIER-STOKES EQUATIONS

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

$$\mathbf{A}^t(\mathbf{x}_p^t) = \frac{1}{\rho^t(\mathbf{x}_p^t)}[\nabla\bullet\sigma^t(\mathbf{x}_p^t) + \mathbf{b}^t(\mathbf{x}_p^t)] \tag{4.1}$$

with the stress tensor

$$\sigma^t(\mathbf{x}_p^t) = \tau^t(\mathbf{x}_p^t) - p^t(\mathbf{x}_p^t)\mathbf{I} \tag{4.2}$$

and the deviatoric tensor

$$\tau^t(\mathbf{x}_p^t) = \mu[\nabla\mathbf{V}^t(\mathbf{x}_p^t) + \nabla^T\mathbf{V}^t(\mathbf{x}_p^t)] \tag{4.3}$$

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

The mass conservation reads

$$\rho^t(\mathbf{x}_p^t)\nabla\bullet\mathbf{V}^t(\mathbf{x}_p^t) = 0 \tag{4.4}$$

For an incompressible flow:

$$\rho^t(\mathbf{x}_p^t) = \rho(\mathbf{x}_p) = \rho_p = cte \neq 0 \tag{4.5}$$

and then (4.4) has the single form:

$$\nabla\bullet\mathbf{V}^t(\mathbf{x}_p^t) = 0 \tag{4.6}$$

Using the X-IVAS method presented before, the Navier-Stokes equations between two times t^n and t^{n+1} reads:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n(\mathbf{x}_p^\tau)\,d\tau \\ \mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) = \mathbf{V}^n(\mathbf{x}_p^n) + \frac{1}{\rho_p}\int_n^{n+1}[\nabla\bullet\sigma^n(\mathbf{x}_p^t) + \mathbf{b}^n(\mathbf{x}_p^t)]\,dt \end{cases} \tag{4.7}$$

Equation (4.7) is explicit not only for the velocity terms but also for the pressure. As explained before, for incompressible flows the pressure must remain implicit in order that the time integration scheme remains unconditionally stable. For this reason, we will slightly modify (4.7) in order to allow the pressure to remain implicit. In fact the method will not be anymore a fully explicit integration, but will be a semi-explicit (or semi-implicit) method. However, we will continue calling the method X-IVAS in order to emphasize the explicit feature of the momentum equation and the integration over the streamlines for both the velocity and the acceleration.

The new semi-explicit integration scheme is:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^{n} + \displaystyle\int_n^{n+t} \mathbf{V}^n(\mathbf{x}_p^{\tau})\,d\tau \\ \mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) = \mathbf{V}^n(\mathbf{x}_p^{n}) + \dfrac{1}{\rho_p}\displaystyle\int_n^{n+1}[\nabla\bullet\tau^n(\mathbf{x}_p^t) - \nabla p^t(\mathbf{x}_p^t) + \mathbf{b}^n(\mathbf{x}_p^t)]\,dt \end{cases} \tag{4.8}$$

in the second equation of (4.8) we remark the implicit term for the pressure, i.e.:

$$\int_n^{n+1}[\nabla p^t(\mathbf{x}_p^t)]\,dt \tag{4.9}$$

This term will be split in the following way:

$$\int_n^{n+1}[\nabla p^t(\mathbf{x}_p^t)]\,dt = \int_n^{n+1}\left\{\nabla p^n(\mathbf{x}_p^t) + \nabla[\delta p\ \ (\mathbf{x}_p^t)]\right\}dt \tag{4.10}$$

with

$$\delta p(\mathbf{x}_p^t) = p^{n+1}(\mathbf{x}_p^t) - p^n(\mathbf{x}_p^t) \tag{4.11}$$

The term $\int_n^{n+1}\left\{\nabla[\delta p\ \ (\mathbf{x}_p^t)]\right\}dt$ in (4.10) is the implicit term introduced to stabilize the pressure integration in time.

In order to solve separately the pressure field from the velocity field, we will use a Pressure Segregation Method as the Fractional Step Method (FSM). Then the second equation of (4.8) will be split in the following two ones:

$$\begin{cases} \widehat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) = \mathbf{V}^n(\mathbf{x}_p^{n}) + \displaystyle\int_n^{n+1}\dfrac{1}{\rho_p}\left\{\nabla\bullet\tau^n(\mathbf{x}_p^t) - \nabla p^n(\mathbf{x}_p^t) + \mathbf{b}^n(\mathbf{x}_p^t)\right\}dt \\ \mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) = \widehat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) - \displaystyle\int_n^{n+1}\dfrac{1}{\rho_p}\left\{\nabla[\delta p(\mathbf{x}_p^t)]\right\}dt \end{cases} \tag{4.12}$$

The integral in the second equation, that is the term used for the stabilization of the pressure integration in time, will be approximated by a fully implicit backward integration scheme:

$$\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) = \widehat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) - \frac{1}{\rho}\nabla[\delta p(\mathbf{x}_p^{n+1})]\,\frac{\Delta t}{2} \tag{4.13}$$

Applying the divergence operator to both sides of (4.13) and taking into account (4.6) gives:

$$\nabla\bullet\hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) = \nabla\bullet\left\{\frac{1}{\rho_p}\nabla[\delta p(\mathbf{x}_p^{n+1})]\right\}\frac{\Delta t}{2} \qquad (4.14)$$

Then the three steps FSM using the X-IVAS technique remains:

Step I) Evaluate explicitly the fractional velocity $\hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1})$ and the new particle position $\mathbf{x}_p^{n+t}$:

$$\begin{cases} \mathbf{x}_p^{n+t} = \mathbf{x}_p^n + \int_n^{n+t} \mathbf{V}^n(\mathbf{x}_p^\tau)\,d\tau \\ \hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) = \mathbf{V}^n(\mathbf{x}_p^n) + \int_n^{n+1}\frac{1}{\rho_p}\left\{\nabla\bullet\tau^n(\mathbf{x}_p^t) - \nabla p^n(\mathbf{x}_p^t) + \mathbf{b}^n(\mathbf{x}_p^t)\right\}dt \end{cases} \qquad (4.15)$$

Step II) Solve implicitly the Laplace equation to obtain the pressure increment $\delta p(\mathbf{x}_p^{n+1})$:

$$\nabla\bullet\hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) = \nabla\bullet\left\{\frac{1}{\rho_p}\nabla[\delta p(\mathbf{x}_p^{n+1})]\right\}\frac{\Delta t}{2} \qquad (4.16)$$

Step III) Evaluate the new incompressible velocity $\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1})$ and pressure $p^{n+1}(\mathbf{x}_p^{n+1})$:

$$\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1}) = \hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1}) - \frac{1}{\rho}\nabla[\delta p(\mathbf{x}_p^{n+1})]\frac{\Delta t}{2} \qquad (4.17)$$

$$p^{n+1}(\mathbf{x}_p^{n+1}) = p^n(\mathbf{x}_p^{n+1}) + \delta p(\mathbf{x}_p^{n+1}) \qquad (4.18)$$

5 SPATIAL DISCRETIZATION OF THE CONVECTION-DIFFUSION EQUATION USING THE X-IVAS METHOD.

Lagrangian formulations are more naturally solved using moving meshes, this is the case of Particle Methods like SPH (Monaghan, J.J. 1988) or PFEM (Idelsohn, S.R.; Oñate, E. and Del Pin, F. 2004), but may be also solved using fixed meshes as in the case of MPM (D. Sulsky and A. Kaul, 2004), or PFEM-2 as it will be explained in the second part of this section.

5.1 Discretization with moving meshes

At each time step a new mesh is generated with the position of the particles. Let $Mesh^n$ be the coordinates and the connectivity of a mesh at time $t = t^n$. The temperature at the nodes of the mesh is known from previous time steps.

Inside each element, the spatial approximation of the temperature is:

$$T^n(\mathbf{x}) = \vec{\mathbf{N}}^{nT}(\mathbf{x})\vec{\mathbf{T}}^n \qquad (5.1)$$

where $\vec{\mathbf{N}}^{nT}(\mathbf{x})$ represent the vector containing the standard continuous FEM shape functions and $\vec{\mathbf{T}}^n$ the vector with the local values of the temperature at time $t = t^n$.

The term $\nabla\bullet[\kappa\nabla T^n(\mathbf{x}_p^t)]$ in (3.4), will be discretized continuously through the domain with the same shape functions used for the temperature:

$$\nabla\bullet[\kappa\nabla T^n(\mathbf{x})] = g^n(\mathbf{x}) = \vec{\mathbf{N}}^{nT}(\mathbf{x})\vec{\mathbf{g}}^n \qquad (5.2)$$

(5.2)

where $\vec{\mathbf{g}}^n$ represent the vector containing the local value of the $\nabla\bullet[\kappa\nabla T^n(\mathbf{x}_p^t)]$ function.
To evaluate $\mathbf{g}^n$ we will use the same approximation used in standard FEM, that is:

$$\int_{\Omega^n} \vec{\mathbf{N}}^n(\mathbf{x})\left\{\nabla\bullet[\kappa\nabla T^n(\mathbf{x})]-\vec{\mathbf{N}}^{nT}(\mathbf{x})\vec{\mathbf{g}}^n\right\}d\Omega=0 \tag{5.3}$$

(5.3)

Integrating by part the first term:

$$-\int_{\Omega^n}\left\{\nabla\vec{\mathbf{N}}^n\kappa\nabla\vec{\mathbf{N}}^{nT}\right\}d\Omega\vec{\mathbf{T}}^n+\int_{\Gamma}\vec{\mathbf{N}}^n\overline{q^n}d\Gamma-\int_{\Omega}\vec{\mathbf{N}}^n\vec{\mathbf{N}}^{Tn}d\Omega\vec{\mathbf{g}}^n=0 \tag{5.4}$$

(5.4)

where $\overline{q^n}$ is the known normal flux to the boundary Γ:

$$\overline{q^n}=-\kappa\nabla T^n\mathbf{n} \tag{5.5}$$

Finally the value of $\vec{\mathbf{g}}^n$ remains:

(5.5)

$$\vec{\mathbf{g}}^n=(\vec{\vec{\mathbf{M}}}^n)^{-1}(\vec{\vec{\mathbf{K}}}^n\vec{\mathbf{T}}^n+\overline{\vec{\mathbf{q}}^n}) \tag{5.6}$$

and

$$g^n(\mathbf{x})=\vec{\mathbf{N}}^{nT}(\mathbf{x})(\vec{\vec{\mathbf{M}}}^n)^{-1}(\vec{\vec{\mathbf{K}}}^n\vec{\mathbf{T}}^n+\overline{\vec{\mathbf{q}}^n}) \tag{5.7}$$

(5.6)

(5.7)

where the matrix $\vec{\vec{\mathbf{M}}}^n,\vec{\vec{\mathbf{K}}}^n$ and $\overline{\vec{\mathbf{q}}^n}$ are the standard mass matrix, stiffness matrix and imposed flux vector defined in FEM built with $Mesh^n$.

$$\overrightarrow{\overrightarrow{\mathbf{M}}}^n=\int_{\Omega^n}\left\{\vec{\mathbf{N}}^n\vec{\mathbf{N}}^{nT}\right\}d\Omega \quad;\quad \vec{\vec{\mathbf{K}}}^n=\int_{\Omega^n}\left\{\nabla\vec{\mathbf{N}}^n\kappa\nabla\vec{\mathbf{N}}^{nT}\right\}d\Omega \quad \vec{\vec{\mathbf{q}}}=\int_{\Gamma}\vec{\mathbf{N}}^n\,\overline{\mathbf{q}}^n d\Gamma \tag{5.8}$$

; ; (5.8)

The mass matrix may be lumped, as it is usual in explicit methods.
The X-IVAS method for one time step may be summarized as:

$$\begin{cases}\vec{\mathbf{g}}^n=(\vec{\vec{\mathbf{M}}}^n)^{-1}(\vec{\vec{\mathbf{K}}}^n\vec{\mathbf{T}}^n+\overline{\vec{\mathbf{q}}^n})\\ \mathbf{x}_p^t=\mathbf{x}_p^n+\int_n^t\mathbf{V}^n(\mathbf{x}_p^\tau)d\tau\\ T^{n+1}(\mathbf{x}_p^{n+1})=T^n(\mathbf{x}_p^n)+\int_n^{n+1}\left\{\vec{\mathbf{N}}^{nT}(\mathbf{x}_p^t)\right\}dt\vec{\mathbf{g}}^n+\int_n^{n+1}Q^n(\mathbf{x}_p^t)dt\\ \textit{with }\ \mathbf{x}_p^{n+1}\ \textit{ generate a new mesh}\end{cases} \tag{5.9}$$

It must be noted that the time integrals in (5.9) may be evaluated for different methods, for instance, dividing the time step Δt in small sub-steps δt. This is not an expensive operation taking into account that computations are explicit and then each particle may be evaluated separately from each other using a parallel computer.

6 SPATIAL DISCRETIZATION OF THE INCOMPRESSIBLE NAVIER-STOKES EQUATIONS USING THE X-IVAS METHOD.

As in the previous section we can divide the algorithms into moving mesh and fixed mesh ones. However, for simplicity we will describe only the moving mesh algorithm and the other may be easily obtained with the same ideas previously described, that means: many particles at each element and projection on a fixed mesh at the end of each time step.

We have a mesh $Mesh^n$ at time $t=t^n$ and in this mesh we define the approximation for each component of the velocity field and the pressure using the FEM shape functions:

$$V_i^n(\mathbf{x}) = \vec{\mathbf{N}}_i^{nT}(\mathbf{x})\vec{\mathbf{V}}_i^n; \quad p^n(\mathbf{x}) = \vec{\mathbf{N}}_p^{nT}(\mathbf{x})\vec{\mathbf{p}}^n \tag{6.1}$$

or in the three component of velocity

$$\mathbf{V}^n(\mathbf{x}) = \vec{\mathbf{N}}^{nT}(\mathbf{x})\vec{\mathbf{V}}^n \tag{6.2}$$

Starting with first step of the FSM, the terms $\frac{1}{\rho_p}\nabla\bullet\tau^n(\mathbf{x})$ and $\frac{1}{\rho_p}\nabla p^n(\mathbf{x})$ in (5.15) will be also approximated in a continuous field and will be called $\mathbf{g}^n(\mathbf{x})$ and $\Pi^n(\mathbf{x})^n$ respectively, such that:

$$\mathbf{g}^n(\mathbf{x}) = \vec{\mathbf{N}}^{nT}(\mathbf{x})\vec{\mathbf{g}}^n$$

$$\Pi^n(\mathbf{x}) = \vec{\mathbf{N}}^{nT}(\mathbf{x})\overrightarrow{\Pi}^n \tag{6.3}$$

where $\vec{\mathbf{g}}^n$represent the vector containing the local values of the $\frac{1}{\rho_p}\nabla\bullet\tau^n(\mathbf{x}_p^n)$ function at the nodes and the local values of $\frac{1}{\rho_p}\nabla p^n(\mathbf{x}_p^n)$.

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

$$\int_{\Omega^n}\vec{\mathbf{N}}^n(\mathbf{x})\left\{\frac{1}{\rho_p}\nabla\bullet\left[\tau^n(\mathbf{x})\right] - \vec{\mathbf{N}}^{nT}(\mathbf{x})\,\vec{\mathbf{g}}^n\right\}d\Omega = 0 \tag{6.4}$$

(6.4)

$$\int_{\Omega^n}\vec{\mathbf{N}}\;(\mathbf{x})\left\{\frac{1}{\rho_p}\nabla p^n(\mathbf{x}) - \vec{\mathbf{N}}^{nT}(\mathbf{x})\,\overrightarrow{\mathbf{\Pi}}^n\right\}d\Omega = 0 \tag{6.5}$$

(6.5)

Integrating by parts the first term in both equations (6.4 and 6.5):

$$-\int_{\Omega^n}\nabla\vec{\mathbf{N}}^n(\mathbf{x})\frac{\mu_p}{\rho_p}\left[\nabla\vec{\mathbf{N}}^n + \nabla\vec{\mathbf{N}}^{nT}\right]d\Omega + \int_{\Gamma^n}\vec{\mathbf{N}}^n(\mathbf{x})\bar{q}^n\,d\Gamma - \int_{\Omega^n}\vec{\mathbf{N}}^n(\mathbf{x})\vec{\mathbf{N}}^{nT}d\Omega\vec{\mathbf{g}}^n = 0 \tag{6.6}$$

and (6.6)

$$-\int_{\Omega^n}\nabla\vec{\mathbf{N}}^n(\mathbf{x})\frac{1}{\rho_p}\vec{\mathbf{N}}^{nT}d\Omega\,\overrightarrow{\mathbf{p}}^n + \int_{\Gamma^n}\vec{\mathbf{N}}^n(\mathbf{x})\bar{p}\,\boldsymbol{\eta}d\Gamma - \int_{\Omega^n}\vec{\mathbf{N}}^n(\mathbf{x})\vec{\mathbf{N}}^{nT}d\Omega\overrightarrow{\Pi}^n = 0 \tag{6.7}$$

we obtain: (6.7)

$$\vec{\mathbf{g}}^n - \overrightarrow{\mathbf{\Pi}}^n = \left(\overrightarrow{\overrightarrow{\mathbf{M}}}^n\right)^{-1}\left(\vec{\vec{\mathbf{K}}}^n\,\frac{\mu}{\rho}\overrightarrow{\mathbf{V}}^n - \overrightarrow{\mathbf{B}}^n\,\frac{1}{\rho}\vec{p}^n - \overrightarrow{\overrightarrow{\sigma}}^n\right) \tag{6.8}$$

with (6.8)

$$\vec{\vec{\mathbf{K}}}^n = \int_{\Omega^n}\nabla\vec{\mathbf{N}}^n\left(\frac{\mu}{\rho}\right)_p\left[\nabla\vec{\mathbf{N}}^n + \nabla\vec{\mathbf{N}}^{nT}\right]d\Omega \tag{6.9}$$

(6.9)

$$\overrightarrow{\overrightarrow{\mathbf{M}}}^n = \int_{\Omega^n}\left\{\vec{\mathbf{N}}^n\vec{\mathbf{N}}^{nT}\right\}d\Omega \tag{6.10}$$

(6.10)

$$\vec{\vec{\mathbf{B}}}^n\left(\frac{1}{\rho}\right)=\int_{\Omega^n}\left\{\nabla\vec{\mathbf{N}}^n\frac{1}{\rho_p}\vec{\mathbf{N}}^{nT}\right\}d\Omega \qquad (6.11)$$

and (6.11)

$$\overline{\sigma^n}=\int_{\Gamma_\sigma}\vec{\mathbf{N}}^n\overline{q^n}d\Gamma-\int_{\Gamma_\sigma}\vec{\mathbf{N}}^n\overline{p^n}\mathbf{In}d\Gamma=\int_{\Gamma_\sigma}\vec{\mathbf{N}}^n[\frac{\mu}{\rho_p}(\nabla\mathbf{V}^n+\nabla^T\mathbf{V}^n)-p^n\mathbf{I}]\mathbf{n}d\Gamma=\int_{\Gamma_\sigma}\vec{\mathbf{N}}^n\overline{\sigma^n}\mathbf{n}d\Gamma \qquad (6.12)$$

It must be noted that Γ_σ is the boundary line where the surface stresses are known. Only in the case of free-surface flows we have $\Gamma_\sigma\neq 0$. However, on a free-surface typically is $\overline{\sigma^n}=0$. (6.12) This means that the last integral in (6.12) in general is always equal to zero. Nevertheless it is interesting to note the integration by parts of the pressure gradients in (6.7). This integration is performed in order to have $\overline{\sigma^n}=0$ as a natural boundary condition, which is the standard boundary condition on a free-surface boundary.

Then, after discretization in space, the first step of the FSM (Eqs. 4.15) will read:

$$\begin{cases}\mathbf{x}_p^{n+t}=\mathbf{x}_p^n+\int_n^{n+t}\mathbf{V}^n(\mathbf{x}_p^\tau)d\tau\\ \hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1})=\mathbf{V}^n(\mathbf{x}_p^n)+\int_n^{n+1}\vec{\mathbf{N}}^{nT}(\mathbf{x}_p^t)dt(\vec{\mathbf{g}}^n-\vec{\Pi}^n)+\frac{1}{\rho_p}\int_n^{n+1}\mathbf{b}^n(\mathbf{x}_p^t)dt\end{cases} \qquad (6.13)$$

It is necessary at this step to evaluate (before changing the mesh), vector $\vec{\mathbf{p}}^n(\mathbf{x}_p^{n+1})$. This vector will be useful later.

After the first step, we move the nodes to the new position $\mathbf{x}_p^{n+t}$ and we generate a new mesh $Mesh^{n+1}$ (or we project the $\hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1})$ on the old mesh in the case of a fixed mesh method).

The next step is to solve implicitly (Eq. 4.16). Using a classical FEM approximation reads:

$$\int_{\Omega^{n+1}}\vec{\mathbf{N}}_p^{n+1}\nabla\bullet\hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1})d\Omega-\int_{\Omega^{n+1}}\vec{\mathbf{N}}_p^{n+1}\nabla\bullet\left\{\frac{\Delta t}{2\rho_p}\nabla[\delta p(\mathbf{x}_p^{n+1})]\right\}d\Omega=0 \qquad (6.14)$$

The field $\delta p(\mathbf{x}_p^{n+1})$ will be also discretized with the same FEM shape functions:

$$\delta p(\mathbf{x}_p^{n+1})=p^{n+1}(\mathbf{x}_p^{n+1})-p^n(\mathbf{x}_p^{n+1})=\vec{\mathbf{N}}_p^{Tn+1}(\mathbf{x})\delta\vec{\mathbf{p}}^{n+1} \qquad (6.15)$$

Integrating by parts both terms in (7.14) gives:

$$-\int_{\Omega^{n+1}}\nabla\vec{\mathbf{N}}_p^{n+1}\vec{\mathbf{N}}^{Tn+1}d\Omega\,\hat{\vec{\mathbf{V}}}^{n+1}+\int_{\Omega^{n+1}}\nabla\vec{\mathbf{N}}_p^{n+1}\frac{\Delta t}{2\rho_p}\nabla\vec{\mathbf{N}}_p^{Tn+1}d\Omega\,\delta\vec{\mathbf{p}}^{n+1}+\int_{\Gamma_V}\vec{\mathbf{N}}_p^{n+1}\overline{\mathbf{V}}^{n+1}d\Gamma=0 \qquad (6.16)$$

$$-\vec{\vec{\mathbf{B}}}^{n+1}\hat{\vec{\mathbf{V}}}^{n+1}+\vec{\vec{\mathbf{L}}}^{n+1}(\frac{\Delta t}{2\rho_p})\delta\vec{\mathbf{p}}^{n+1}+\overline{\vec{\mathbf{V}}}^{n+1}=0 \qquad (6.17)$$

$$\vec{\vec{\mathbf{B}}}^{n+1}=\int_{\Omega^{n+1}}\nabla\vec{\mathbf{N}}_p^{n+1}\vec{\mathbf{N}}^{Tn+1}d\Omega \qquad (6.18)$$

$$\vec{\vec{\mathbf{L}}}^{n+1}(\frac{\theta\Delta t}{\rho_p})=\int_{\Omega^{n+1}}\nabla\vec{\mathbf{N}}_p^{n+1}\frac{\Delta t}{2\rho_p}\nabla\vec{\mathbf{N}}_p^{Tn+1}d\Omega \qquad (6.19)$$

$$\overline{\overline{\vec{\mathbf{V}}}}^{n+1} = \int_{\Gamma_V} \vec{\mathbf{N}}_p^{n+1} \overline{\mathbf{V}}^{n+1} d\Gamma = \int_{\Gamma_V} \vec{\mathbf{N}}_p^{n+1} [\hat{\mathbf{V}}^{n+1} - \frac{\Delta t}{2\rho} \nabla(\delta p)] d\Gamma \tag{6.20}$$

It must be noted the integration by parts of the first term in (6.14). This integration allows us have as natural boundary condition the term $\int_{\Gamma_V} \vec{\mathbf{N}}_p^{n+1} \overline{\mathbf{V}}^{n+1} d\Gamma = 0$ which is the standard boundary condition in confined flows.

Despite the momentum equations do not need any spatial stabilization in the Lagrangian formulation, equation (6.17) must be stabilized in space for equal order velocity-pressure formulations like this one. Any space stabilization method may be used given, in general, a term like $\tau \vec{\vec{\mathbf{S}}}^{n+1} \vec{\mathbf{p}}^{n+1} (\mathbf{x}_p^{n+1})$ that must be added to (6.17):

$$\left[\vec{\vec{\mathbf{L}}}^{n+1} \left(\frac{\Delta t}{2\rho_p} \right) + \tau \vec{\vec{\mathbf{S}}}^{n+1} \right] \delta \vec{\mathbf{p}}^{n+1} = \vec{\vec{\mathbf{B}}}^{n+1} \vec{\hat{\mathbf{V}}}^{n+1} - \vec{\vec{\mathbf{V}}}^{n+1} - \tau \vec{\vec{\mathbf{S}}}^{n+1} \vec{\mathbf{p}}^{n} (\mathbf{x}_p^{n+1}) \tag{6.21}$$

After solving (6.21) the pressure at time $t = t^{n+1}$ and position $\mathbf{x}_p^t = \mathbf{x}_p^{n+1}$ may be evaluated as:

$$\vec{\mathbf{p}}^{n+1} (\mathbf{x}_p^{n+1}) = \vec{\mathbf{p}}^{n} (\mathbf{x}_p^{n+1}) + \delta \vec{\mathbf{p}}^{n+1} \tag{6.22}$$

The last step in the FSM is the evaluation of the final velocity at time $t = t^{n+1}$ and position $\mathbf{x}_p^t = \mathbf{x}_p^{n+1}$. Using (4.17):

$$\vec{\mathbf{V}}^{n+1} (\mathbf{x}_p^{n+1}) = \hat{\vec{\mathbf{V}}}^{n+1} (\mathbf{x}_p^{n+1}) - \delta \vec{\Pi}^{n+1} \frac{\Delta t}{2} \tag{6.23}$$

with

$$\delta \vec{\Pi}^{n+1} = \frac{1}{\rho_p} \nabla \delta \vec{\mathbf{p}}^{n+1} = \frac{1}{\rho_p} \nabla [\vec{\mathbf{p}}^{n+1} (\mathbf{x}_p^{n+1}) - \vec{\mathbf{p}}^{n} (\mathbf{x}_p^{n+1})] \tag{6.24}$$

The Vector $\delta \vec{\Pi}^{n+1}$ may be calculated in the same way as for $\vec{\Pi}^n$. This means discretizing $\delta \Pi^{n+1}$ on the domain Ω by the same FEM shape functions, $\delta \Pi^{n+1} = \mathbf{N}^{Tn+1} \delta \vec{\Pi}^{n+1}$ and imposing the equality

$$\int_{\Omega^{n+1}} \vec{\mathbf{N}}^{n+1} [\frac{1}{\rho_p} \nabla \delta p^{n+1} - \delta \Pi^{n+1}] d\Omega = 0 \tag{6.25}$$

In matrix notation:

$$\int_{\Omega^{n+1}} \vec{\mathbf{N}}^{n+1} \frac{1}{\rho_p} \nabla \vec{\mathbf{N}}^{Tn+1} d\Omega \delta \vec{\mathbf{p}}^{n+1} = \int_{\Omega^{n+1}} \vec{\mathbf{N}}^{n+1} \vec{\mathbf{N}}^{Tn+1} d\Omega \delta \vec{\Pi}^{n+1} \tag{6.26}$$

and

$$\delta \vec{\Pi}^{n+1} = (\vec{\vec{\mathbf{M}}}^{n+1})^{-1} \vec{\vec{\mathbf{D}}} (\frac{1}{\rho_p}) \delta \vec{\mathbf{p}}^{n+1} \tag{6.27}$$

with

$$\vec{\vec{\mathbf{D}}}^{n+1}(\frac{1}{\rho_p}) = \int_{\Omega^{n+1}} \vec{\mathbf{N}}^{n+1} \frac{1}{\rho_p} \nabla \vec{\mathbf{N}}^{Tn+1} d\Omega$$

$$\vec{\vec{\mathbf{M}}}^{n+1} = \int_{\Omega^{n+1}} \vec{\mathbf{N}}^{n+1} \vec{\mathbf{N}}^{Tn+1} d\Omega \qquad (6.28)$$

It is interesting to note here the difference between $\vec{\vec{\mathbf{D}}}^{n+1}(\frac{1}{\rho_p})$ and $\vec{\vec{\mathbf{B}}}^{n}(\frac{1}{\rho_p})$. They are different due to the integration by parts in (6.7) and not performed in (6.26). However it is convenient to evaluate both matrices in order to satisfy the natural boundary conditions.

The X-IVAS method for the N-S equations with moving mesh reads:

I) On $Mesh^n$ evaluate explicitly the viscous and pressure gradients terms $\vec{\mathbf{g}}^n - \vec{\Pi}^n$:

$$\vec{\mathbf{g}}^n - \vec{\Pi}^n = (\vec{\vec{\mathbf{M}}}^n)^{-1} (\vec{\vec{\mathbf{K}}}^n (\frac{\mu}{\rho}) \vec{\mathbf{V}}^n - \vec{\vec{\mathbf{B}}}^n (\frac{1}{\rho}) \vec{\mathbf{p}}^n - \overline{\vec{\sigma}^n})$$

II) Evaluate explicitly the fractional velocity $\hat{\mathbf{V}}^{n+1}(\mathbf{x}_p^{n+1})$ and the new particle position $\mathbf{x}_p^{n+1}$:

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

III) Evaluate $\mathbf{p}^n(\mathbf{x}_p^{n+1})$:

$$\mathbf{p}^n(\mathbf{x}_p^{n+1}) = \vec{\mathbf{N}}^{nT}(\mathbf{x}_p^{n+1}) \vec{\mathbf{p}}^n$$

IV) Generate a new $Mesh^{n+1}$ with $\mathbf{x}_p^{n+1}$

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

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

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

$$\delta\vec{\Pi}^{n+1} = (\vec{\vec{\mathbf{M}}}^{n+1})^{-1} \vec{\vec{\mathbf{D}}}^{n+1} (\frac{1}{\rho_p}) \delta\vec{\mathbf{p}}^{n+1}$$

VII) Evaluate explicitly the new velocity $\mathbf{V}^{n+1}(\mathbf{x}_p^{n+1})$:

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

VIII) Evaluate the new pressure vector $\mathbf{p}^{n+1}(\mathbf{x}_p^{n+1})$:

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

IX) Increase the time $t^{n+1} = t^n + \Delta t$, and update the variables:

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

go to I).

7 NUMERICAL EXAMPLE FOR CONVECTION-DIFFUSION PROBLEMS.

7.1 Advective transport of a Gaussian hill

There is a simple problem that put in evidence the pathology that Eulerian approaches suffer solving a pure advective transport problem. This is the case of a Gaussian hill signal used as initial condition with no 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 to have a non-zero value in a limited region of the domain initially. Time to time the signal should be transported following circular path lines preserving its original shape and also its original amplitude. Figure 7.1 shows the problem definition.

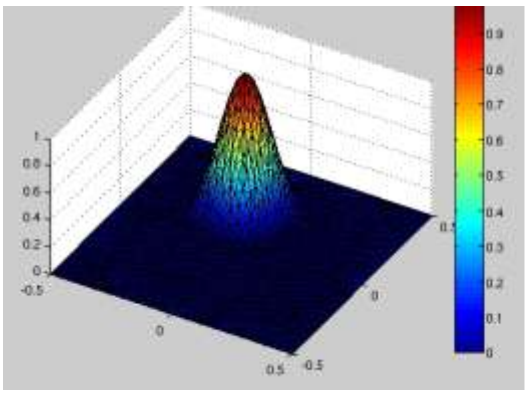


Fig. 7.1 Initial temperature distribution and mesh used

A 2D finite element mesh will be used with 12880 nodes and 25362 triangular elements for represent the fluid velocity and the temperature distributions. The time step integration will be chosen in order to have a CFL number bigger than 5 in all the cases tested. This means that at each time step, a particle may move thought 5 elements. For the Eulerian test a second order integration in time was performed and the convective terms were stabilized using a Streamline Upwind Petrov-Galerkin (SUPG) method (T. Tezduyar and T.J.R. Hughes 1986).

The following cases will be tested:

1- Euler Implicit with a Fixed Mesh (EIFM)
2- Lagrangian with Fixed Mesh using the Characteristic Methods (LFM-C)
3- Lagrangian with Moving Mesh (LMM-1) and particle position evaluated by
$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{V}^{n+1}\Delta t$
4- Lagrangian with Moving Mesh (LMM-2) and particle position evaluated by
$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{V}^{n+1}\theta\Delta t + \mathbf{V}^n(1-\theta)\Delta t$
5- Lagrangian with Moving Mesh and particle position following the streamlines (X-IVAS method) (LMM-X)
6- Lagrangian with Fixed Mesh and particle position following the streamlines (X-IVAS method) (LFM-X)

Figure 7.2 shows the results obtained with the Eulerian formulation (EIFM). The top temperature that initially was equal to one, it is equal to 0.7 after a complete turn of the flow, it is equal to 0.6 after 2 turns and equal to 0.5 after 3 turns.

Due to the numerical diffusion introduced by Eulerian schemes this signal strongly decays during the first two or three turns dissipating almost all its content. The Eulerian solution improves if the grid is refined but a very fine mesh is needed to get an acceptable solution. Even though using Cranck-Nicolson schemes the solution does not improve significantly.

This example shows the unacceptable numerical diffusion that the Euler formulation with large time steps and a relative poor mesh introduces.

We must also note, that the solution with EIFM is implicit because the explicit solution does not work for a CFL>1. This means that this solution is not only inaccurate but it is also expensive (need to solve a linear system of equation at each time step). Furthermore, the convective terms must be stabilized.

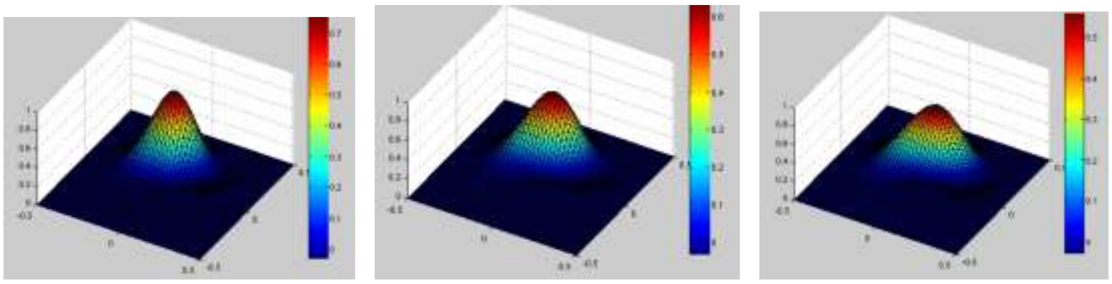


Fig. 7.2 Solution of the Euler Implicit with Fixed Mesh, after 1,2 and 3 flow tours (EIFM)

Despite of all this drawbacks of the fixed mesh Eulerian approach, this is the method more used actually in the literature, not only to solve convection-diffusion problems but also in multi-fluid solution using a level-set method to evaluate the interface position (Sussman, M. et al 1994).

Figure 7.3 shows the solution for a Characteristic Methods as that explained in Section 1 (Eq. 1.14 and 1.15). We can see that this method is also very diffusive. The top temperature that initially was equal to one, it is equal to 0.6 after a complete tour of the flow, it is equal to 0.45 after 2 tours and equal to 0.35 after 3 tours.

Despite that Characteristic Method use the streamlines to follows the particle position, the results are very poor due to the need of a permanent projection of the temperature of each particle on a fixed mesh.

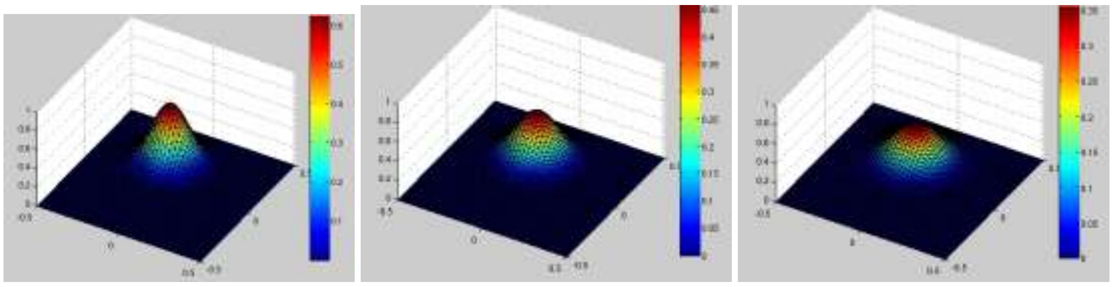


Fig. 7.3 Solution using Fixed Mesh and a Characteristic Method after 1,2 and 3 flow tours. (LFM-C)

We will use now the X-IVAS procedure developed in this paper, which is an explicit method following the velocity and acceleration streamlines. We will use both versions: with moving mesh and with fixed mesh. For the algorithm with fixed mesh, we will introduce particles to improve the projection of the temperature on the fixed mesh at the end of each time step. As explained before, these particles will increase the computing time but, as all the evaluations on the particles are explicit, this computing time increase may not be considered in the case we are using a parallel processing like a GPGPU.

The number of particles introduced at the initial time step was of 10 particles at each finite element.

Figure 7.4 shows the results with a moving mesh and Figure 7.5 with a fixed mesh. We can see that in both cases there is not any numerical diffusion. The result is exactly the initial temperature after 1,2 and 3 turns of the flow. Furthermore, the solution was explicit with a CFL number bigger than 6 in all the time. A plot with the instantaneous value of the CFL number for the moving mesh may be see in Fig. 7.6.

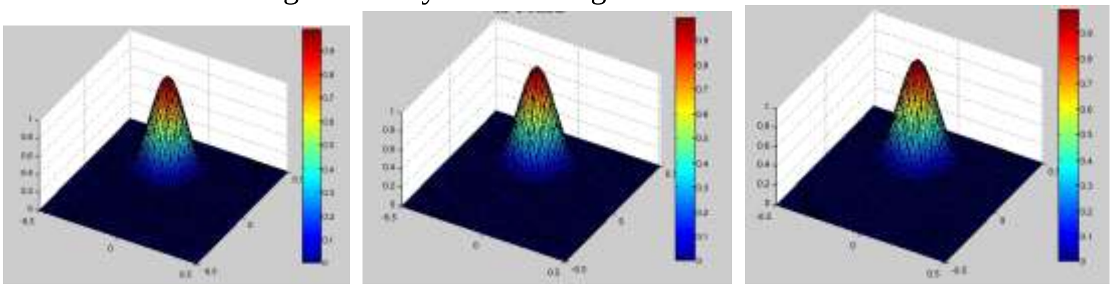


Fig. 7.4 Solution using Moving Mesh and the X-IVAS method after 1,2 and 3 flow tours (LMM-X)

We understand that this is a very particular example with conductivity equal to zero. This means that the streamline integration of the acceleration (Eq.4.7) is identically zero in this case. However these comparisons lead us to think that the X-IVAS integration will produce significant improvements in the solution of all problems dominated by convection.

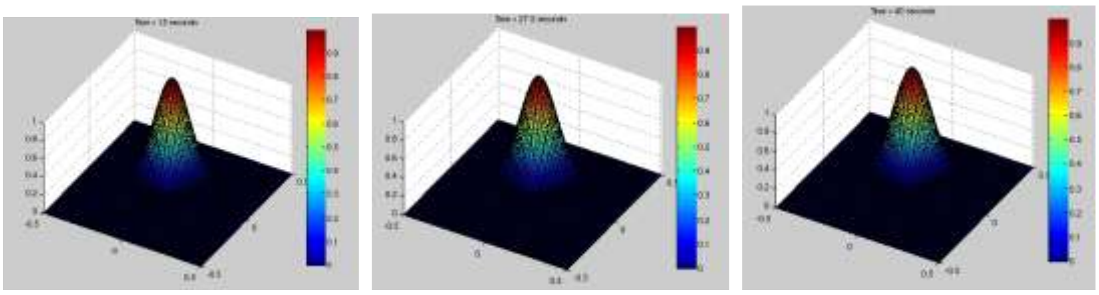


Fig. 7.5 Solution using Fixed Mesh and the X-IVAS method after 1,2 and 3 flow tours. (LFMX)

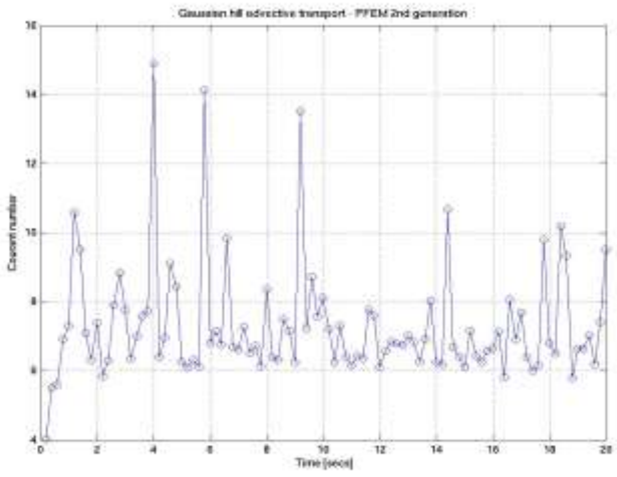


Fig. 7.6 Instantaneous CFL numbers for the solution with X-IVAS method and moving mesh

The other two Lagrangian cases: LMM-1 and LMM-2, give exact solution for the temperature distribution as it is expected but they gives errors in the tracking of the particle position. For instance Figure 7.7 shows the results for LMM-1 view in a 2D representation of the same problem. We can see that the initial temperature distribution moves in a radial way towards the exterior boundary after each flow turn. (Note the change in the draw scale). LMM-2 gives similar results with fewer errors than LMM-1 in the particle position.

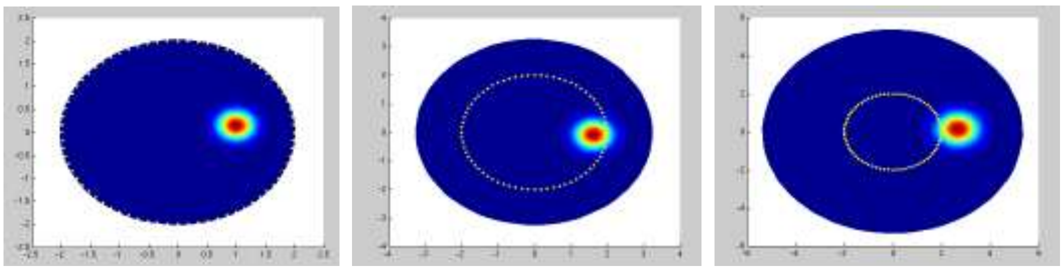


Fig. 7.7 Solution using Moving Mesh with and explicit integration of the particle position after 1,2 and 3 flow tours. (LMM-1)

A first preliminary conclusion looking the results for these simple problems is that the Lagrangian formulation has a strong aptitude to solve advective problems in a more natural way. This can be observed in the intrinsic simplicity of the code used to solve this problem. The only thing to care about is the solution of the particle path lines. The advection step is extremely simple as each particle conserves its original value.

One explanation of the errors introduced in the Eulerian formulation may be the numerical stabilization needed. Other explanation might be related to the inherent interpolation operator representing the Eulerian approach. Comparing both formulae we note that when the streamlines are coincident (or nearly coincident) with the trajectory, the Lagrangian scheme has not errors (or has small errors) while the Eulerian formulation has an intrinsic error depending on the time step and the element mesh size. (See Section 3 and Fig. 3.1 for a detailed explanation of this source of errors).

Even though these conclusions applies to all Lagrangian methods, the time integration of the new X-IVAS approach deserves attention. A rough computation of the time evolution makes the particles to follow wrong path lines corrupting the solution. This fact makes the difference between standard approaches like LMM-1 or LMM-2 against the new proposal following the streamlined time integration as LMM-X or LFM-X.

One of the main drawbacks of Lagrangian methods is the problem that particles crash in between or becomes to close to each other. This limitation is even stricter for moving meshes. This is not a drawback in case of heterogeneous fluids where particles may be sown or may be eliminated at each time step. For multi-fluid flows special strategies may be used in order to save the interface position of the two fluids.

In both, the LMM-X and the LFM-X approaches the diffusion term in the momentum equation is solved explicitly. For this reason the algorithm becomes conditionally stable with the restriction in the time step according to the Fourier limit. As the Fourier number is proportional to the smaller distance between two particles, the explicit integration may become instable when two particles come near to each other. To implement this restriction some sort of update the budget of particles may be included with the following actions:

- One particle is removed when get closer to another at a certain tolerance.
- One particle is removed when get through the boundaries of the domain.
- One particle is added when one element becomes bigger than a mean volume value computed preliminary. This particle is seed at the element centroid.
- One particle is removed when the triangle generated by Delaunay becomes a sliver.

With these actions the remeshing produces a grid with certain minimal quality for warranting acceptable time step to remain stable.

Also the time step is adaptive according to guarantee that the Fourier limit is not exceeded.

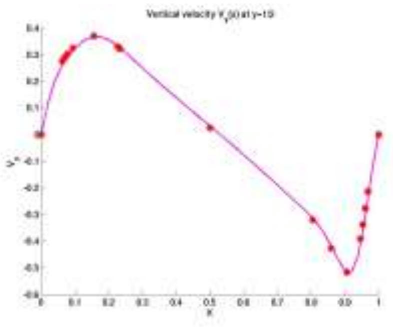
8 NUMERICAL EXAMPLES FOR INCOMPRESSIBLE NAVIER-STOKES PROBLEMS.

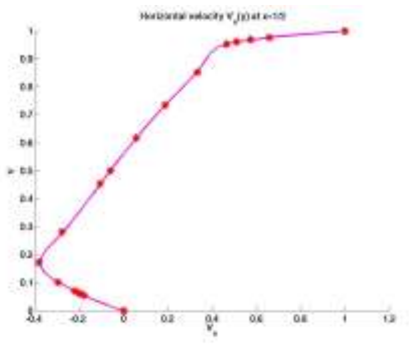
8.1 Lid-driven square cavity

The very popular lid driven square cavity (Ghia U. et al 1982) for Reynolds number of 1000 was tested first using a moving and a fixed mesh method. Even though this example only serves to check it steady solution it is a good benchmark to evidence the behavior of several new ideas put in this work, like the following:

- The velocity and acceleration streamline based time integrations, (Eq. 6.13).
- The point collocation for the whole fractional-step method unless the Poisson solver, (Eqs. 6.13 and 6.14).
- The use of a projection on a continuous field of the pressure gradients and also a double projection on a continuous field of the diffusive terms, (Eqs. 6.4 and 6.5).
- The adaptive time-step limited only by Fourier number restriction for stability but no for CFL number limits.
- The comparison of the results using a moving and a fixed mesh method.

Fig. 8.1 shows the velocity profile at a vertical and horizontal cuts passing through the origin of the square cavity, divided initially with 100x100 equal size triangular elements, compared with Ghia U. et al. 1982, established as the reference for this benchmark where finite differences on 129x129 nodes was used.





a) b)

Fig. 8.1 Lid-driven square cavity. a) Middle line vertical velocity. b) Middle line horizontal velocity

Figure 8.2 shows the evolution in time of both stability limits: CFL and Fourier numbers.

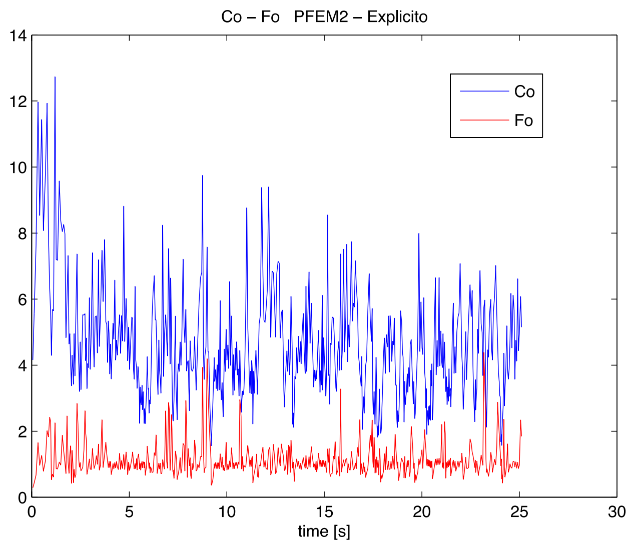


Fig. 8.2 Lid-driven square cavity. CFL and Fourier numbers evolution

We can see that the Fourier number is keep all the time around one by modifying the time step. Nevertheless, the CFL number is much greater than one without any stability problem at all times.

8.2 Backward facing step

This popular test concerns with the flow inside a channel through an expansion of 1:2 of cross section area at Reynolds 389. This problem is well referenced in the bibliography and a deep comparison between results obtained by PFEM2 with other methods is included afterwards.

The next figure shows a comparison between numerical PFEM2 results against experimental data obtained by Armaly et.al 1983 . As it may be noted a general good agreement is achieved with some differences mainly at the middle of the channel.

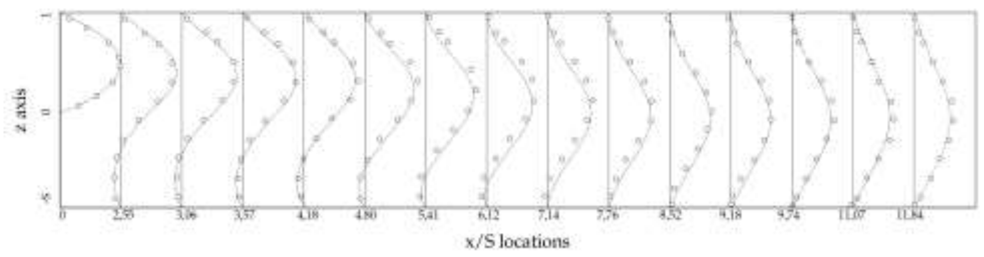


Fig. 8.3 Backward Facing Step benchmark. Velocity profiles against experimental results.

The next figure is similar to the above including also other numerical solution obtained with OpenFOAM (OpenFOAM 2009) where a similar agreement between numerical and experiments is achieved.

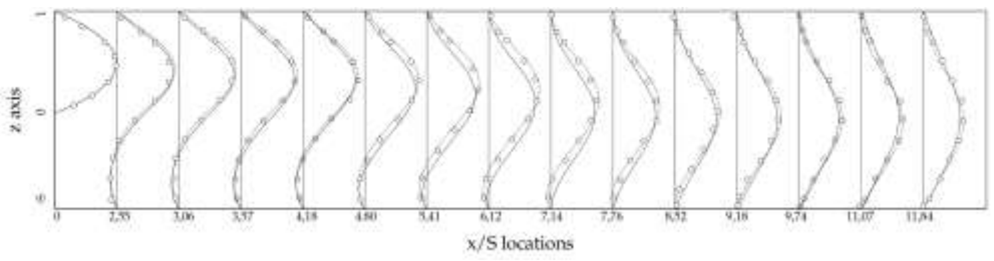


Fig. 8.4 Backward Facing Step benchmark. Velocity profiles against OpenFOAM and experimental results.

Figure 8.5 shows a typical snapshot with the velocity magnitude

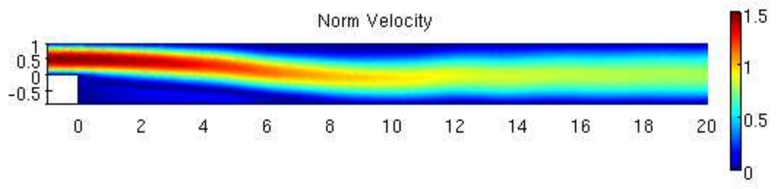


Fig. 8.5 Backward Facing Step benchmark. Velocity magnitude

Next, Fig. 8.6 shows the pressure field where it is possible to visualize that the pressure get developed after 20 length units downwards of the step

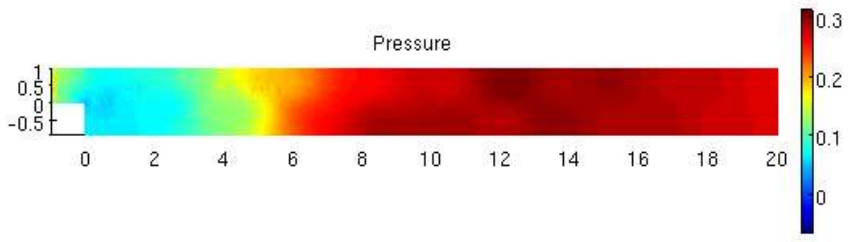


Fig. 8.6 Backward Facing Step benchmark. Pressure

8.3 Vortex shedding behind a cylinder

This example involves the flow past a cylinder, a popular benchmark problem in computational fluid mechanics. A circular cylinder is immersed in a viscous fluid. The Reynolds number is based on the cylinder diameter D and the prescribed uniform inflow velocity U. The geometry and boundary conditions are shown in Fig. 8.7 We set U=1 and D=1.

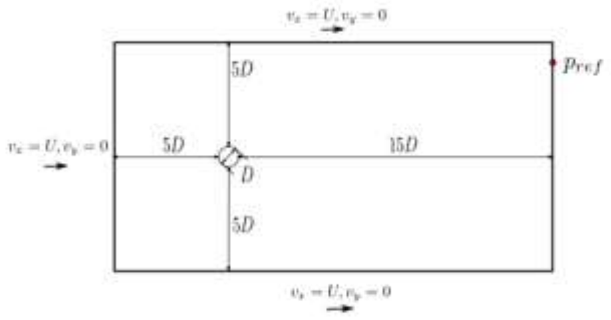


Fig. 8.7. Vortex shedding. Geometry.

For Reynolds approximately less than 40, two symmetrical eddies develop behind the cylinder. These eddies become unstable at higher Reynolds numbers and periodic vortex shedding occurs, leading to the so-called Von Karman vortex street.

The problem was solved with the moving mesh method, starting with 3.500 particles and then increased the number of particle to an average of 5.000 particles with an adaptive particle refinement near the cylinder.

Figure 8.8 shows the velocity profile once the instability occur for the case of Reynolds=1000.

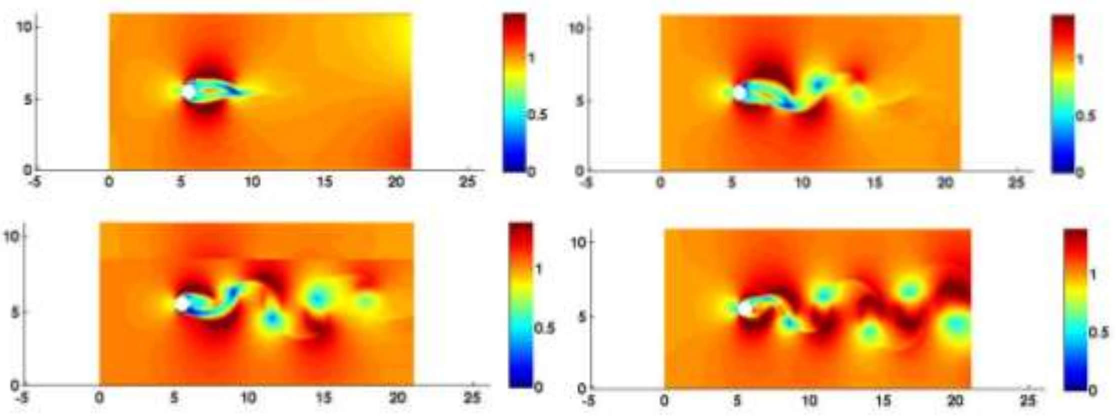


Fig. 8.8: Vortex shedding. Velocity magnitude profile at some time steps.

The time step was regulated in order to keep the Fourier number less than one. The instantaneous Courant number at each time step is shown in Figure 8.9. We can see that the

Courant number is much larger than one for all times.

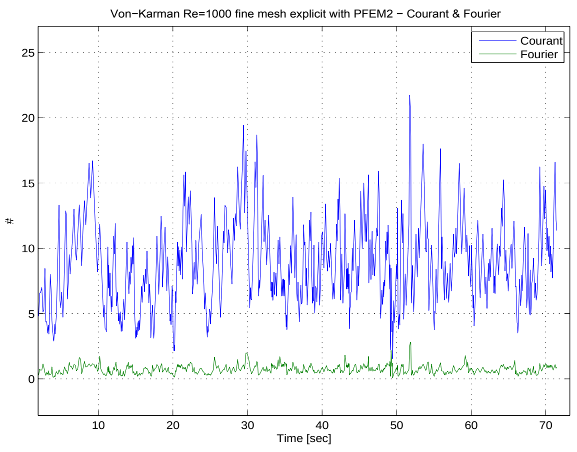


Fig. 8.9. Vortex shedding. CFL number at different time steps.

The vertical forces on the cylinder (Lift) and the horizontal forces (Drag) are represented in Fig. 8.10 and 8.11 respectively. The period of oscillation of these forces is an interesting value to test the accuracy of the method, normally written in a dimensionless form through the Strouhal number. The frequency of the numerical solution is 0.1775 which is in good agreement with the empirical solution that is 0.19 for this Reynolds number.

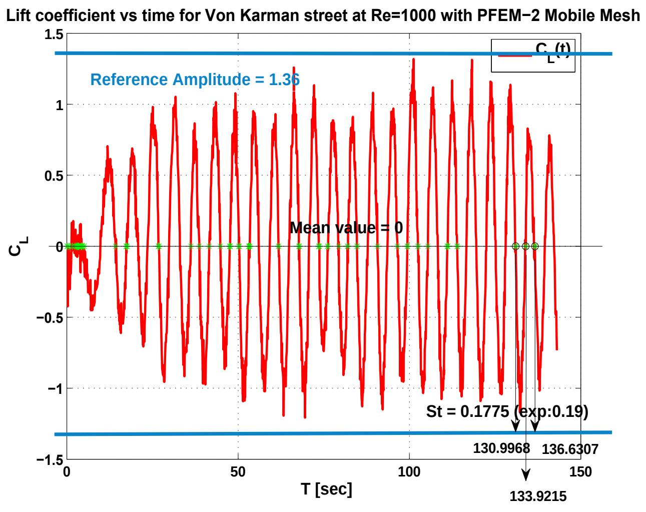


Fig. 8.10. Vortex shedding. Lift and Strouhal number

In Fig. 8.10 we can see that the lift forces have variable amplitude. The experimental amplitude is marked in the figure with a value of 1.35. In Figure 8.11 the time variation of the drag and the amplitude of the drag oscillations are compared to the experimental results. We can see that the experimental amplitude is 0.21 which is in good agreement with the average amplitude obtained with our method. In the same way, the average of the drag forces is in good agreement with the mean experimental value of 1.4754.

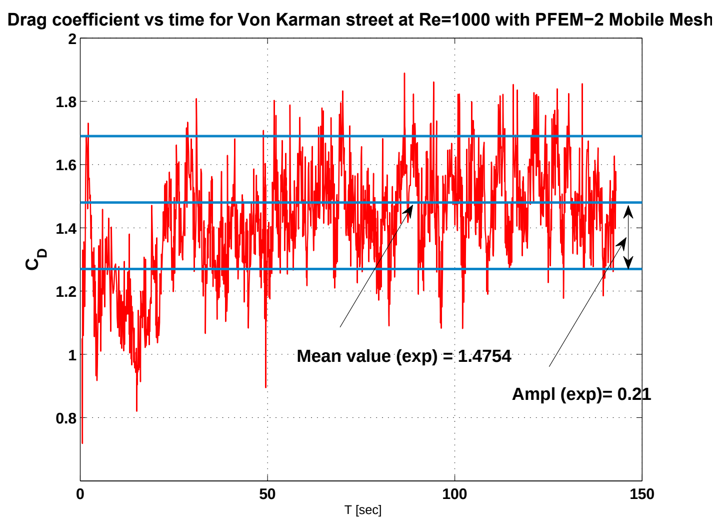


Fig. 8.11. Vortex shedding. Drag Coefficient

9 CONCLUSIONS

Explicit time integration has the big advantage of the easy and efficient scalability. The big disadvantage is the restricted time step for stability. Normally very small time steps are necessary in problems dominated by convection (high Courant number) or in problems where a boundary layer with very small elements is needed to obtain acceptable results (high Fourier number). In this paper we have presented a methodology to keep the time integration explicit independently of the Courant number. The time step is regulated by accuracy needs, but the method remains explicit and stable independently of the mesh size providing that the Fourier number stability limit is not exceeded.

The method is based on moving the particle in a Lagrangian frame following the velocity and acceleration streamlines. The method may be used indistinctly either with a moving mesh or with a fixed mesh. For heterogeneous fluids (multi-fluids) or free-surface flows, certainly, a moving mesh will be more efficient and accurate.

The pressure field is solved implicitly to have the possibility to deal with fully incompressible flows. Nevertheless, a low speed of sound may be introduced, allowing therefore an explicit solution for both the momentum and the mass conservation equations.

We point out once more that the big advantage of the formulation proposed here is the possibility to use large time steps (as in the implicit solutions) while preserving the explicit solution, at least for the momentum equations. This simplifies the implementation of the formulation in parallel computers, improving considerably the CPU time in complex CFD problems.

REFERENCES

Zienkiewicz, O.C. et al.. The finite element method. 6th Edition, 3 Volumes, Elsevier., 2006

Donea, J. and Huerta, A.. *Finite element method for flow problems*. J.Wiley., 2003

Ehrenstein, G., Real time fluid simulation with moving obstacles using GPUs. Diploma Thesis Univ. of Munich, www.ehrenstein.biz. , 2008

Hughes, T.J.R., Franca, L.P., Balestra, M.. A new finite element formulation for computational fluid dynamics. V Circumventing the Babuska-Brezzi condition: A stable Petrov-Galerkin formulation of the Stokes problem accommodating equal order interpolations. *Comput. Meth. Appl. Mech. Engrg*., 59, 85–89, 1986

Tezduyar, T.E., Behr, M. and Liou, J.. A New Strategy for Finite Element Computations Involving Moving Boundaries and Interfaces-The Deforming-Spatial-Domain/Space-Time Procedure: I. The Concept and the Preliminary Numerical Tests. *Comput. Meth. Appl. Mech. Engrg*., 94, 339–351., 1992

Chorin, A.J. , A numerical solution for solving incompressible viscous flows. *J. Comp. Phys*., 2: 12–26.,1967

Stam Jos, Stable Fluids, *SIGGRAPH 99 Conference Proceedings, Annual Conference Series*, August 1999.

Pironneau O. and Tabata M.,. Stability and convergence of a Galerkin- characteristics finite element scheme of lumped mass type. *Int. J. Numer. Meth. Fluids*; 64:1240-1253, 2010

Monaghan, J.J.,. An introduction to SPH. *Computer Physics Communications*, vol 48. 89-96., 1988

Idelsohn, S.R., Oñate, E. and Del Pin, F.. The particle finite element method a powerful tool to solve incompressible flows with free-surfaces and breaking waves. *Int. J. Num. Meth. Engng.,* Vol. 61, 964-989., 2004

Sulsky D. and Kaul A.,   Implicit Dynamics in the Material-Point Method . *Comput. Meth. Appl. Mech. Engrg*, 193:1137-1170 ,2004.

Tezduyar T.E. and Hughes T.J.R., Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations, in: *Proceedings of AIAA 21st Aerospace Sciences Meeting*, AIAA Paper 83-0125, Reno, Nevada ,1983.

Sussman, M., Smereka, P. and Osher, S.,. A level set approach for computing solutions to incompressible two-phase flow, *J. Comp. Phys*., 114, 146-159, 1994

Ghia U., Ghia K., Shin C, High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *Journal of Computational Physics*, 48: 387-411 ,1982

Armaly, B.F., Durst, F., Pereira, J.C.F. and Schonung, B., Experimental and theoretical investigation of backward-facing step flow. *Journal of Fluid Mechanics*, 127, 473-496, 1983

OpenFOAM, The Open Source CFD Toolbox, *User Guide*, OpenCFD Ltd., 2009