Comput Visual Sci 3: 67–75 (2000)

Computing and Visualization in Science

Regular article

A finite point method for incompressible flow problems

Eugenio Oñate[1], **Carlos Sacco**[1], **Sergio Idelsohn**[1,2]

[1] International Center for Numerical Methods in Engineering, Universidad Politécnica de Cataluña Gran Capitán s/n, 08034 Barcelona, Spain
[2] Universidad Nacional del Litoral, Santa Fe, Argentina

Received: 30 June 1999 / Accepted: 21 September 1999

Communicated by: M. Espedal and A. Quarteroni

Abstract. A stabilized finite point method (FPM) for the meshless analysis of incompressible fluid flow problems is presented. The stabilization approach is based in the finite increment calculus (FIC) procedure developed by Oñate [14]. An enhanced fractional step procedure allowing the semi-implicit numerical solution of incompressible fluids using the FPM is described. Examples of application of the stabilized FPM to the solution of two incompressible flow problems are presented.

1 Introduction

Mesh free techniques have become quite popular in computational mechanics. A family of mesh free methods is based on smooth particle hydrodynamic procedures [1,2]. These techniques, also called free lagrangian methods, are typically used for problems involving large motions of solids and moving free surfaces in fluids. A second class of mesh free methods derive from generalized finite difference (GFD) techniques [3,4]. Here the approximation around each point is typically defined in terms of Taylor series expansions and the discrete equations are found by using point collocation. Among a third class of mesh free techniques we find the so called diffuse element (DE) method [5], the element free Galerking (EFG) method [6,7] and the reproducing kernel particle (RKP) method [8,9]. These three methods use local interpolations for defining the approximate field around a point in terms of values in adjacent points, whereas the discretized system of equations is typically obtained by integrating the Galerkin variational form over a suitable background grid.

The *finite point method* (FPM) proposed in [10–13] is a truly meshless procedure. The approximation around each point is obtained by using standard moving least square techniques similarly as in DE and EFG methods. The discrete system of equations is obtained by sampling the governing differential equations at each point as in GFD methods.

The basis of the success of the FPM for solid and fluid mechanics applications is the *stabilization* of the discrete differential equations. The stable form found by the *finite element calculus* procedure presented in [14–17] corrects the errors introduced by the point collocation procedure, mainly next to the boundary segments. In addition, it introduces the necessary stabilization for treating high convection effects and it also allows equal order velocity-pressure interpolations in fluid flow problems [17].

The content of the paper is structured as follows. In the next section the basis of the FPM approximation is described. The stabilized governing equations for incompressible flows derived using the finite increment calculus (FIC) approach are presented next. A three step semi-implicit fractional solution scheme using the FPM approximation is described in some detail. Two examples of the efficiency and accuracy of the stabilized FPM for numerical solution of incompressible flow problems are presented, namely the analysis of a driven cavity flow and the solution of a backwards facing step.

1.1 Interpolation in the FPM

Let Ω_i be the interpolation domain (cloud) of a function $u(x)$ and let s_j with $j = 1, 2, \cdots, n$ be a collection of n points with coordinates $x_j \in \Omega_i$. The unknown function u may be approximated within Ω_i by

$$u(x) \cong \hat{u}(x) = \sum_{l=1}^{m} p_l(x)\alpha_l = p(x)^T\alpha \tag{1}$$

where $\alpha = [\alpha_1, \alpha_2, \cdots \alpha_m]^T$ and vector $p(x)$ contains typically monomials, hereafter termed "base interpolating functions", in the space coordinates ensuring that the basis is complete. For a 2D problem we can specify

$$p = [1, x, y]^T \quad \text{for } m = 3 \tag{2}$$

and

$$p = [1, x, y, x^2, xy, y^2]^T \quad \text{for } m = 6 \quad \text{etc.} \tag{3}$$

Function $u(x)$ can now be sampled at the n points belonging to Ω_i giving

$$u^h = \begin{Bmatrix} u_1^h \\ u_2^h \\ \vdots \\ u_n^h \end{Bmatrix} \cong \begin{Bmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_n \end{Bmatrix} = \begin{Bmatrix} p_1^T \\ p_2^T \\ \vdots \\ p_n^T \end{Bmatrix} \alpha = C\alpha \tag{4}$$

where $u_j^h = u(x_j)$ are the unknown but sought for values of function u at point j, $\hat{u}_j = \hat{u}(x_j)$ are the approximate values, and $p_j = p(x_j)$.

In the FE approximation the number of points is chosen so that $m = n$. In this case C is a square matrix. The procedure leads to the standard shape functions in the FEM [19].

If $n > m$, C is no longer a square matrix and the approximation can not fit all the u_j^h values. This problem can be simply overcome by determining the $\hat{u}$ values by minimizing the sum of the square distances of the error at each point weighted with a function $\varphi(x)$ as

$$J = \sum_{j=1}^{n} \varphi(x_j)\left(u_j^h - \hat{u}(x_j)\right)^2 = \sum_{j=1}^{n} \varphi(x_j)\left(u_j^h - p_j^T\alpha\right)^2 \tag{5}$$

with respect to the α parameters. Note that for $\varphi(x) = 1$ the standard least square (LSQ) method is reproduced.

Function $\varphi(x)$ is usually built in such a way that it takes a unit value in the vecinity of the point i typically called "star node" where the function (or its derivatives) are to be computed and vanishes outside a region Ω_i surrounding the point. The region Ω_i can be used to define the number of sampling points n in the interpolation region. A typical choice for $\varphi(x)$ is the normalized Gaussian function and this has been chosen in the examples shown in the paper. Of course $n \geq m$ is always required in the sampling region and if equality occurs no effect of weighting is present and the interpolation is the same as in the LSQ scheme.

Standard minimization of (5) with respect to α gives

$$\alpha = \bar{C}^{-1}u^h \qquad , \qquad \bar{C}^{-1} = A^{-1}B \tag{6}$$

$$A = \sum_{j=1}^{n} \varphi(x_j)\, p(x_j)\, p^T(x_j)$$

$$B = \left[\varphi(x_1)p(x_1),\, \varphi(x_2)p(x_2),\, \cdot\varphi(x_n)p(x_n)\right] \tag{7}$$

The final approximation is obtained by substituting α from (6) into (1) giving

$$\hat{u}(x) = p^T\bar{C}^{-1}u^h = N^T u^h = \sum_{j=1}^{n} N_j^i u_j^h \tag{8}$$

where the "shape functions" for the i-th star node are

$$N_j^i(x) = \sum_{l=1}^{m} p_l(x)\bar{C}_{lj}^{-1} = p^T(x)\bar{C}^{-1} \tag{9}$$

It must be noted that accordingly to the least square character of the approximation

$$u(x_j) \simeq \hat{u}(x_j) \neq u_j^h \tag{10}$$

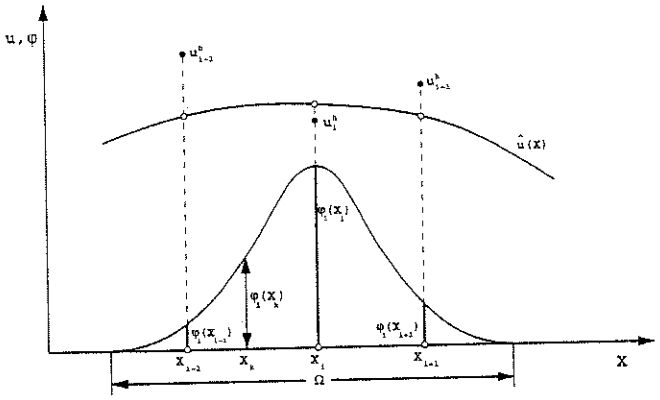

Fig. 1. Fixed weighting least square procedure

i.e. the local values of the approximating function do not fit the nodal unknown values. Indeed $\hat{u}$ is the true approximation for which we shall seek the satisfaction of the differential equation and the boundary conditions and u_j^h are simply the unknown parameters sought.

The weighted least square approximation described above depends on a great extend on the shape and the way to apply the weighting function. The simplest way is to define a fixed function $\varphi(x)$ for each of the Ω_i interpolation domains [11, 12].

Let $\varphi_i(x)$ be a weighting functions satisfying (Fig. 1)

$$\begin{aligned} &\varphi_i(x_i) = 1 \\ &\varphi_i(x) \neq 0 \qquad x \in \Omega_i \\ &\varphi_i(x) = 0 \qquad x \notin \Omega_i \end{aligned} \tag{11}$$

Then the minimization square distance becomes

$$J_i = \sum_{j=1}^{n} \varphi_i(x_j)\left(u_j^h - \hat{u}(x_j)\right)^2 \qquad \text{minimum} \tag{12}$$

The expression of matrices A and B coincide with (7) with $\varphi(x_j) = \varphi_i(x_j)$.

Note that according to (1), the approximate function $\hat{u}(x)$ is defined in each interpolation domain Ω_i. In fact, different interpolation domains can yield different shape functions N_j^i. As a consequence a point belonging to two or more overlapping interpolation domains has different values of the shape functions which means that $N_j^i \neq N_j^k$. The interpolation is now multivalued within Ω_i and, therefore for any useful approximation a decision must be taken limiting the choice to a single value. Indeed, the approximate function $\hat{u}(x)$ will be typically used to provide the value of the unknown function $u(x)$ and its derivatives in only specific regions within each interpolation domain. For instance by using point collocation we may limit the validity of the interpolation to a single point x_i. It is precisely in this context where we have found this meshless method to be more useful for practical purposes [10–13].

2 Stabilized governing equations for incompressible flows

The stabilized governing equations for incompressible viscous flows are obtained by applying the standard conserva-

tion laws expressing balance of momentum and mass over a control domain. Assuming that the control domain has finite dimensions and representing the variation of mass and momentum over the domain using Taylor series expansions of one order higher than those used in the standard infinitesimal theory, the following expressions are found [14, 17]:

Momentum.

$$r_{m_i} - \underline{\frac{1}{2} h_{mj} \frac{\partial r_{m_i}}{\partial x_j}} = 0 \qquad \text{in } \Omega \tag{13}$$

Mass balance.

$$r_d - \underline{\frac{1}{2} h_{dj} \frac{\partial r_d}{\partial x_j}} = 0 \qquad \text{in } \Omega \tag{14}$$

where for the steady state case

$$r_{m_i} = \rho \frac{\partial (u_i u_j)}{\partial x_j} + \frac{\partial p}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \tag{15}$$

$$r_d = \frac{\partial u_i}{\partial x_i} \tag{16}$$

with $i, j = 1, 2$ for a two dimensional flow.

In (15) ρ is the fluid density (here assumed to be constant), u_i is the velocity component in the i-th direction, p the pressure, b_i the body forces and τ_{ij} the viscous stress components related to the velocity gradients through the fluid viscosity μ by

$$\tau_{ij} = 2\mu \left(\varepsilon_{ij} - \frac{1}{3} \frac{\partial u_k}{\partial x_k} \delta_{ij} \right) \tag{17a}$$

with

$$\varepsilon_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{17b}$$

Einstein summation convention for repeated indexes in products and derivatives is used, i.e. $h_{dj} \frac{\partial r_d}{\partial x_j} = \sum_j h_{dj} \frac{\partial r_d}{\partial x_j}$.

Equations (13) and (14) are the *stabilized forms* of the governing differential equations for an incompressible flow. The terms underlined in (13) and (14) introduce naturally the necessary stabilization at the discretization level. The so called *characteristic length* vectors $\boldsymbol{h}_m$ and $\boldsymbol{h}_d$ are defined as (for 2D problems)

$$\boldsymbol{h}_m = \begin{Bmatrix} h_{m_1} \\ h_{m_2} \end{Bmatrix}, \quad \boldsymbol{h}_d = \begin{Bmatrix} h_{d_1} \\ h_{d_2} \end{Bmatrix} \tag{18}$$

where h_{m_1} and h_{m_2} are the dimensions of the finite control domain where balance of momentum is enforced. Similarly h_{d_1} and h_{d_2} represent the dimensions of the domain where mass conservation is expressed. The components of vectors $\boldsymbol{h}_m$ and $\boldsymbol{h}_d$ introduce the necessary stabilization along the streamline and transverse directions to the flow in the discrete problem.

The method to derive the modified differential equations (13) and (14) incorporating the stabilization terms was termed in [14] *finite increment calculus* as a reference to the standard infinitesimal calculus techniques where the size of the domain where balance of mass and momentum is enforced is assumed to be negligible. Note that for $\boldsymbol{h}_m = \boldsymbol{h}_d \to 0$ the standard infinitesimal form of the momentum and mass balance equations is recovered [14–18].

Equations (13) and (14) are complemented by the following boundary conditions [14, 17].

Balance of momentum at the boundary Γ_t.

$$n_j \sigma_{ij} - t_i + \underline{\frac{1}{2} h_{m_j} n_j r_{m_i}} = 0 \qquad \text{on } \Gamma_t \tag{19}$$

where n_i is the i-th component of the unit normal vector to the boundary and t_i are the prescribed tractions at the Neumann boundary Γ_t of the analysis domain Ω.

Prescribed velocity at the boundaries.

$$u_t = u_t^p \quad \text{on } \Gamma_{u_t} \tag{20}$$

$$u_n - \underline{\frac{1}{2} h_{d_i} n_i r_d} = u_n^p \qquad \text{on } \Gamma_{u_n} \tag{21}$$

In (20) u_t and u_t^p denote the tangential velocity to the boundary and its prescribed value, respectively.

Equation (21) expresses the balance of mass on an arbitrary domain next to the boundary. u_n and u_n^p denote the velocity normal to the boundary and its prescribed value, respectively. The value of u_n^p is zero on solid walls and stationary free surfaces.

Also in (20) and (21) Γ_{u_t} and Γ_{u_n} are the parts of the boundary Γ of Ω where the tangential and normal velocities are prescribed, respectively. The Dirichlet boundary is defined as $\Gamma_u = \Gamma_{u_t} \cup \Gamma_{u_n}$.

The underlined terms in (19) and (21) introduce the necessary stabilization at the boundaries in a form consistent with that of (13) and (14). These terms are obtained by invoking balance of momentum and mass at a domain of finite size next to the boundary. Details of the derivation of (19) to (21) are found in [14, 17].

2.1 Alternative form of stabilized governing equations

Let us express the components of the characteristic vector $\boldsymbol{h}_d$ for the mass balance equation as

$$h_{d_i} = -2\rho \tau_{d_i} u_i \tag{22}$$

where the τ_{d_i} parameters are termed "intrinsic times" per unit mass. The negative sign in (22) is necessary to introduce a positive stabilization in the mass balance equation at the discrete level as it will be shown later.

From simple differentiation rules we can write

$$u_i \frac{\partial}{\partial x_i} \left(\frac{\partial u_j}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left(u_i \frac{\partial u_j}{\partial x_j} \right) - \left(\frac{\partial u_k}{\partial x_k} \right)^2 \tag{23}$$

Substituting (22) into (14) and making use of (23), (14) and (16) we can rewrite the mass balance equation (neglecting higher order terms) as

$$r_d - \tau_{di} \frac{\partial \bar{r}_{m_i}}{\partial x_i} = 0 \tag{24a}$$

where

$$\bar{r}_{m_i} = \rho u_j \frac{\partial u_i}{\partial x_j} + \frac{\partial p}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \tag{24b}$$

Following a similar process, equation (21) expressing balance of mass at the boundary can be rewritten using (13) and (22) as

$$u_n - \tau_{d_i} n_i \bar{r}_{m_i} = u_n^p \qquad \text{on } \Gamma \tag{25}$$

We summarize next for the sake of clarity the set of governing equations to be solved.

Momentum.

$$\boxed{r_{m_i} - \frac{1}{2} h_{m_j} \frac{\partial r_{m_i}}{\partial x_j} = 0} \qquad \text{in } \Omega \tag{26}$$

Mass balance.

$$\boxed{r_d - \tau_{d_i} \frac{\partial \bar{r}_{m_i}}{\partial x_i} = 0} \qquad \text{in } \Omega \tag{27}$$

Boundary conditions.

$$\boxed{n_j \sigma_{ij} - t_i + \frac{1}{2} h_{m_j} n_j r_{m_i} = 0} \qquad \text{on } \Gamma_t \tag{28}$$

$$\boxed{u_t - u_t^p = 0} \qquad \text{on } \Gamma_{u_t} \tag{29}$$

$$\boxed{u_n - \tau_{d_i} n_i \bar{r}_{m_i} - u_n^p = 0} \qquad \text{on } \Gamma_{u_n} \tag{30}$$

where r_{m_i} and r_d are defined in (24a) and (24b) respectively.

A similar form of the modified differential equations for momentum and mass balance ((26) and (27)) has been recently proposed by Ilinca et al. [20]. They express the exact solution as sum of the numerical approximation and a perturbation. The modified equations are derived by expanding the original differential equations for momentum and mass balance in Taylor series and elliminating the perturbation terms. However, the boundary conditions remain unchanged and thus the stabilizing terms in (28) and (30) are omitted in [20]. This leads to the appearance of additional boundary integrals in the Galerkin formulation. These terms vanish naturally if the full stabilized expressions (26)–(30) emanating from the FIC method are used as shown in [17].

3 Fractional step solution

The stabilization formulation above presented is naturally extended to the transient case. The stabilized form of the momentum and mass balance equations are writen now as [14, 18]

Momentum.

$$\left(r_{m_i} - \frac{h_{m_j}}{2} \frac{\partial r_{m_i}}{\partial x_j}\right) - \frac{\delta}{2} \frac{\partial}{\partial t} \left(r_{m_i} - \frac{h_{m_j}}{2} \frac{\partial r_{m_i}}{\partial x_j}\right) = 0 \tag{31}$$

Mass balance.

$$\left(r_d - \frac{h_{d_j}}{2} \frac{\partial r_d}{\partial x_j}\right) - \frac{\delta}{2} \frac{\partial}{\partial t} \left(r_d - \frac{h_{d_j}}{2} \frac{\partial r_d}{\partial x_j}\right) = 0 \tag{32}$$

In above δ is a time stabilization parameter. Transient effects are also included in the term r_{m_i} given by

$$r_{m_i} = \rho \left(\frac{\partial u_i}{\partial t} + \frac{\partial (u_i u_j)}{\partial x_j}\right) + \frac{\partial p}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \tag{33}$$

Equation (31) and (32) are obtained by expressing the balance of momentum and mass in space-time domains of finite dimensions $[\boldsymbol{h}_m \times \delta]$ and $[\boldsymbol{h}_d \times \delta]$, respectively. Details of the derivation can be found in [14, 18].

Equations (31) and (32) can be used to derive a number of stabilized numerical schemes for the transient solution of the Navier–Stokes equations.

3.1 Three steps splitting scheme

It is interesting to derive a splitting algorithm starting with the new stabilized equations. For the sake of clarity the time stabilization terms involving δ will be neglected in (31) and (32). Also the stabilized mass balance equation will be written in the more convenient form given by (27).

A time marching solution scheme for (31) can be written as (for $\delta = 0$)

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\rho} \left[\rho \frac{\partial (u_i u_j)^n}{\partial x_j} + \frac{\partial p^{n+1}}{\partial x_i} - \frac{\partial \tau_{ij}^n}{\partial x_j} - b_i^n - \left(\frac{h_{m_j}}{2} \frac{\partial r_{m_i}}{\partial x_j}\right)^n \right] \tag{34}$$

The analogy of (34) with that found using the so called characteristic integration schemes [21, 22] is clear if vector $\boldsymbol{h}_m$ is chosen aligned with the velocity field, i.e. $\boldsymbol{h}_m = \tau \boldsymbol{u}$ where τ is an intrinsic time parameter. Indeed the arbitrary form of vector $\boldsymbol{h}_m$ in (34) provides a more general procedure where the components of vector $\boldsymbol{h}_m$ can be freely chosen.

A semi-implicit time splitting or "fractional step" [21, 22] algorithm can now be obtained as follows. Equation (34) is split as

$$u_i^* = u_i^n - \frac{\Delta t}{\rho} \left[\rho \frac{\partial (u_i u_j)}{\partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} - b_i - \frac{h_{m_j}}{2} \frac{\partial r_{m_i}}{\partial x_j} \right]^n \tag{35}$$

$$u_i^{n+1} = u_i^* - \frac{\Delta t}{\rho} \frac{\partial p^{n+1}}{\partial x_i} \tag{36}$$

Note that the sum of (35) and (36) gives the original form of (34).

Substituting (35) into (27) gives

$$r_d^* - \frac{\Delta t}{\rho} \frac{\partial^2 p^{n+1}}{\partial x_i \partial x_i} - \tau_{d_i} \left[\frac{\partial \bar{r}_{m_i}}{\partial x_i} \right]^{n+1} = 0 \tag{37}$$

where

$$r_d^* = \frac{\partial u_i^*}{\partial x_i} \tag{38}$$

$$\left[\frac{\partial \bar{r}_{m_i}}{\partial x_i}\right]^{n+1} = \frac{\partial}{\partial x_i}\left[\rho\left(\frac{\partial u_i}{\partial t} + u_j\frac{\partial u_i}{\partial x_j}\right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i\right]^n - \frac{\partial^2 p^{n+1}}{\partial x_i \partial x_i} \tag{39}$$

The solution steps are the following:

Step 1

Solve explicitely for the so called "fractional" velocities u_i^* [21,22] using (35).

Step 2

Compute the pressure field p^{n+1} by solving the equation for the Laplacian of pressure derived from (37). Note that this equation has the following form

$$\Delta t\frac{\partial^2 p^{n+1}}{\partial x_i \partial x_i} + \tau_{d_i}\frac{\partial^2 p^{n+1}}{\partial x_i \partial x_i} = r_d^* - \tau_{d_i}\frac{\partial}{\partial x_i} \times \left[\rho\left(\frac{\partial u_i}{\partial t} + u_j\frac{\partial u_i}{\partial x_j}\right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i\right]^n \tag{40}$$

Clearly for $\tau_{d_i} = \tau$ above equation simplifies to

$$\left(\frac{\Delta t}{\rho} + \tau\right)\Delta p^{n+1} = \bar{r}_d^* \tag{41}$$

where Δ is the Laplacian operator and

$$\bar{r}_d^* = r_d^* - \tau\frac{\partial}{\partial x_i}\left[\rho\left(\frac{\partial u_i}{\partial t} + u_j\frac{\partial u_i}{\partial x_j}\right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i\right]^n \tag{42}$$

Step 3

Compute the velocities u_i^{n+1} by using (36).

Equation (41) differs slighty from the form typically used in fractional step schemes where the term involving τ does not appear [21,22]. This term, however, is essential to preserve the stability of the mixed FPM formulation.

Obviously, other forms of above three steps transient solution scheme involving the implicit computation of u_i^{n+1} are also possible.

Extension of this transient solution method to the simpler Stokes problem are straightforward. The same scheme can be applied to derive enhanced algorithms for transient non linear structural dynamic problems allowing equal order interpolation for velocities and pressure as described in [23].

3.2 Numerical solution using the FPM

The implementation of the three step scheme described in previous section in the context of the FPM is straight forward. Equation (8) is used to define the approximation of velocities and pressures within each cloud Ω_i as

$$\hat{u}_m = \sum_{j=1}^{n} N_j^i u_{m_j}^h; \quad m = 1,2,3 \quad \text{for 3D} \tag{43}$$

$$\hat{p} = \sum_{j=1}^{n} N_j^i p_j^h \tag{44}$$

where $(\hat{\cdot})$ denote approximate values and the shape functions N_j^i were defined in (9).

Direct substitution of (43) and (44) into the stabilized governing equations described in previous section gives the following numerical scheme for computation of the parameters $u_{m_j}^h$ and p_j^h.

Step 1

Compute explicitely the fractional velocities at each point k in the domain as

$$(\hat{u}_i^*)_k = (\hat{f}_i^n)_k; \quad k = 1,\ldots,N; \quad i = 1,2,3 \tag{45}$$

in which N is the total number of points in the domain and

$$(\hat{f}_i^n)_k = \left\{\hat{u}_i^n - \frac{\Delta t}{\rho}\left[\rho\frac{\partial(\hat{u}_i\hat{u}_j)}{\partial x_j} - \frac{\partial \hat{\tau}_{ij}}{\partial x_j} - b_i - \frac{h_{m_j}}{2}\frac{\partial \hat{r}_{m_j}}{\partial x_j}\right]^n\right\}_k \tag{46}$$

where $(\hat{\cdot})$ denotes approximate values.

Once the values of $\hat{u}_i^*$ have been obtained, the parameters $u_{m_j}^h$ can be computed by solving the following system of equations

$$(\hat{u}_m^*)_k = \sum_{j=1}^{n} N_j^k u_{m_j}^h, \quad k = 1,\ldots,N \tag{47}$$

Equation (47) is a system of N equations with N unknowns from where the parameters $u_{m_j}^h$, $j = 1,\ldots,N$ can be found. These parameters are needed to compute the derivatives of the velocity field in steps 2 and 3. Indeed the solution of (47) must be repeated for every component of the velocity vector (i.e. $m = 1,2,3$ for 3D problems).

Step 2

Compute the pressure field at time $n+1$ by solving (40). Substituting (43) and (44) into (40) and sampling this equation at each point in the domain gives (for $\tau_{d_i} = \tau$)

$$K(p^h)^{n+1} = \hat{r}_d^* \tag{48}$$

where (for 2D problems)

$$K_{kj} = \left(\frac{\Delta t}{\rho} + \tau\right)\left(\frac{\partial^2 N_j^k}{\partial x_1^2} + \frac{\partial^2 N_j^k}{\partial x_2^2}\right) \tag{49}$$

$$\hat{r}_{d_k}^* = \hat{r}_{d_k}^* - \tau\frac{\partial}{\partial x_i}\left[\rho\left(\frac{\partial \hat{u}_i}{\partial t} + \hat{u}_j\frac{\partial \hat{u}_i}{\partial x_j}\right) - \frac{\partial \hat{\tau}_{ij}}{\partial x_j} - b_i\right]_k^n \tag{50}$$

As usual $(\cdot)_k^n$ denotes values within brackets evaluated at point k and the n-th time step.

Equation (48) provides a system of equations from which the pressure parameters $(p_k^h)^{n+1}$ can be found at each point k.

Step 3

The final step is the explicit computation of the velocities in each point at time $n+1$. Substituting (43) and (44) into (36) and sampling this equation at each point gives

$$(\hat{u}_i^{n+1}) = \left[\hat{u}_i^* - \frac{\Delta t}{\rho}\frac{\partial \hat{p}^{n+1}}{\partial x_i}\right]_k ; \quad k = 1, \ldots, N \tag{51}$$

Note that the derivatives of the approximate functions $\hat{u}_i$ and $\hat{p}$ in (50) and (51) are computed by direct differentiation of the expressions (43) and (44), i.e.

$$\frac{\partial \hat{u}_m}{\partial x_l} = \sum_{j=1}^{n} \frac{\partial N_j^i}{\partial x_l} u_{m_j}^h$$
$$\frac{\partial \hat{p}}{\partial x_l} = \sum_{j=1}^{n} \frac{\partial N_j^i}{\partial x_l} p_j^h \tag{52}$$

The steps 1–3 described above are repeated for every new time increment.

A local time step size for each point in the domain can be used to speed up the search of the steady state solution. The local time step is defined as $\Delta t_i = \frac{d_i}{2|u_i|}$, where d_i is the minimum distance from a star point to any of its neighbourghs in the cloud. Note however that the full transient solution requires invariably the use of a global time step Δt_g equal for all nodes and defined as $\Delta t_g = \min(\Delta t_i), \quad i = 1, \ldots, N$.

4 Boundary conditions

Prescribed tractions on the Neumann boundary Γ_t, (19) or prescribed velocities at the Dirichlet boundaries Γ_{u_t} or Γ_{u_n}, ((20) and (21)) may be imposed.

During the fractional step solution, the first explicit step is solved without imposing any boundary conditions. During the second step, two kinds of boundary conditions may be imposed: on boundaries where the normal velocity is imposed, (21) reads using (36)

$$u_n^p = u_i^* n_i - \frac{\Delta t}{\rho}\frac{\partial p^{n+1}}{\partial x_i} n_i - \frac{1}{2} h_{d_i} n_i r_d \tag{53}$$

Taking into account (27), (37) and (41) leads to

$$u_n^p = u_i^* n_i - \frac{\Delta t}{\rho}\frac{\partial p^{n+1}}{\partial x_i} n_i - \frac{1}{2} h_{d_i} n_i [(\Delta t + \tau)\Delta p^n - \bar{r}_d^*] \tag{54}$$

Equation (54) represents a stabilized Neumann boundary condition for the pressure equation (48). This equation is imposed in the FPM during the pressure computation step as a new equation for all points k belonging to the Γ_{u_n} boundary.

On outflow boundaries with $n_j \sigma_{ij} = 0$ the pressure is imposed to a constant value, i.e. $p = 0$. In the FPM, essential boundary conditions such as $p = 0$ are imposed using the definition of the function itself via (44) as

$$\hat{p}_i = \sum_{j=1}^{n} N_j^i p_j^h = 0 \tag{55}$$

Equation (55) is sampled at the points located at a boundary where $p = 0$.

During the third step the velocities u_m^{n+1} are computed using (36) for all points within the analysis domain. In points where a velocity is imposed as an essential boundary condition the imposed value is asigned directly. After that, the nodal parameters $u_{m_j}^h$ can be computed by solving the same system of equations described by (47). On points over Neumann boundaries, in particular on boundaries where the tractions are prescribed to zero, equation (19)

$$n_j \sigma_{ij} + \frac{1}{2} h_{m_j} n_j r_{m_i} = 0 \tag{56}$$

is used for computing the velocities at the boundary points using the direct differentiation of the velocity and pressure approximations as described by (52).

5 Computation of the stabilization parameters

Accurate evaluation of the stabilization parameters is one of the crucial issues in stabilized methods. Most of existing methods use expressions which are direct extensions of the values obtained for the simplest 1D case. It is also usual to accept the so called SUPG assumption, i.e. to admit that vector h_m has the direction of the velocity field. This restriction leads to instabilities when sharp layers transversal to the velocity direction are present. This additional defficiency is then corrected by adding a "shock capturing" (SC) stabilization term [24–27].

In our work we will assume for simplicity that the stabilization parameters for the mass balance equations are the same than those for the momentum equations. This implies

$$h_{m_i} = h_{d_i} \tag{57}$$

The problem remains now finding the value of the characteristic length vectors h_{m_i}. Indeed, the components of h_m can introduce the necessary stabilization along the streamline and transversal directions to the flow [14–18].

In this work the SUPG assumption has been chosen for defining h_m as

$$\boldsymbol{h}_m = h_s \frac{\boldsymbol{u}}{|\boldsymbol{u}|} \tag{58}$$

The streamline parameter h_s has been chosen for each cloud as the minimum distance d_i from a star point to any of its neighbourghs. Recall that this distance is also used to define the local time step for each point.

6 Numerical examples

6.1 *Driven cavity flow at* Re = 1000

This is a classical test problem to evaluate the behaviour of any fluid dynamic algorithm. A viscous flow is confined in a square cavity while one of its edges slides tangentially. The boundary conditions are $u = v = 0$ in 3 edges and $u = 1$, $v = 0$ on the upper edge. The problem is solved with the FPM using the distribution of 3329 points shown in Fig. 2. Around

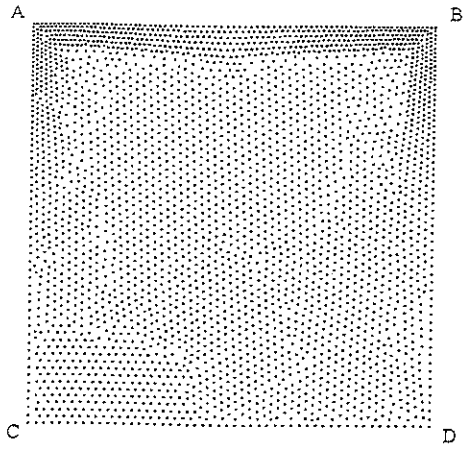

Fig. 2. Driven cavity flow. Distribution of 3329 points. Boundary conditions $u = 0$ at edges AC, CD and BD and points A and B. $u = 1$ and $v = 0$ over the interior of line AB

each point, equal order quadratic based polynomial are used for the velocities and the pressure ($m = 6$). A minimum of six points is selected in each cloud using a combination of minimum distance and quadrant search procedures [11–13]. Typically $n = 9$ is chosen, i.e. two points per each quadrant plus the star node. Initially, except at the edge, the velocity is set to zero everywhere including at the nodes located at the left and right top corners (ramp condition).

Numerical results are shown in Figs. 3, 4 and 5 for Re = 1000. Figures 3 and 4 show the velocity and pressure contours, respectively. The FPM results are compared with experimental results obtained by Ghia et al. [29] showing the velocity x computed along a vertical central cut (Fig. 5). The comparison is satisfactory.

6.2 *Backwards facing step at* Re = 389

In this example, the flow is contrained to move in a 2D domain which presents a backwards step. The domain dimensions are presented in Fig. 6. The step is one half the width of the inflow.

At the inflow a constant velocity profile is prescribed while at the outflow the pressure is prescribed, being the velocity free. The non-slip condition is used at the walls, except

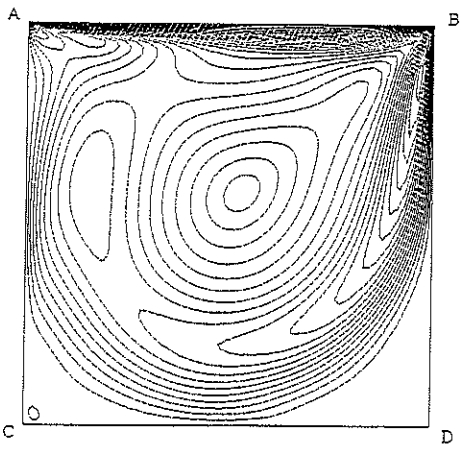

Fig. 3. Driven cavity flow. Velocity contours for Re = 1000

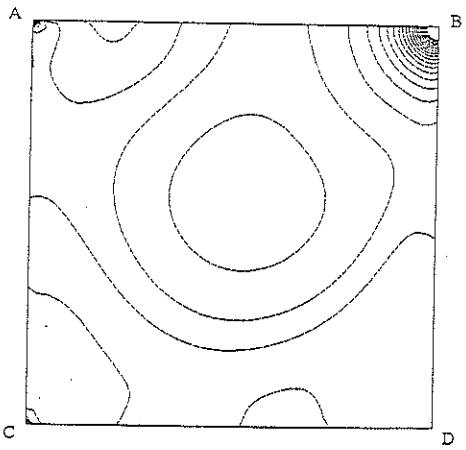

Fig. 4. Driven cavity flow. Pressure contours for Re = 1000

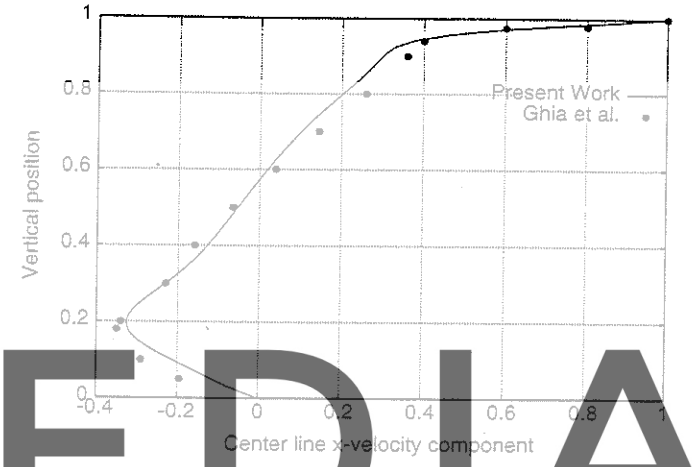

Fig. 5. Driven cavity flow. Horizontal velocity distribution over the center line

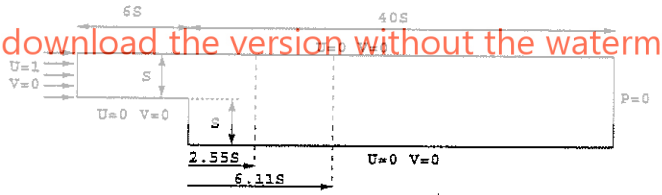

Fig. 6. Backwards facing step. Geometry and boundary conditions

for the two inflow points, where the constant inflow velocity is imposed. No volume forces are present.

The distribution of 8462 points used near the step is represented on Fig. 7. In the rest of the domain a regular distribution of point is used. As on the previous example, equal order

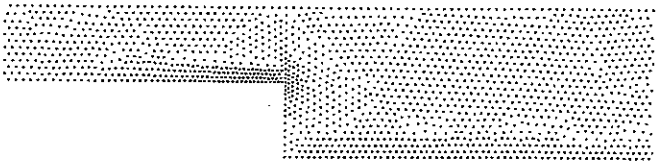

Fig. 7. Backwards facing step. Distribution of 8462 points

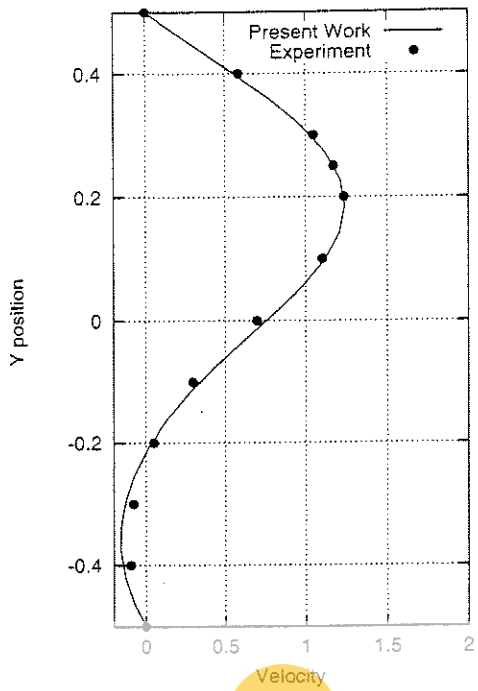

Fig. 8. Backwards facing step. Horizontal velocity distribution along a *vertical line* at $x = 2.55$ S

Fig. 9. Backwards facing step. Horizontal velocity along a *vertical line* at $x = 6.11$ S

quadratic based polynomials are chosen to approximate both velocities and pressure.

Once the stationary state is reached, the solution shows horizontal velocities represented on Figs. 8 and 9 for two planes located at $x = 2.55$ S and $x = 6.11$ S from the step. The FPM results are compared with experimental results presented on [30] showing an excellent agreement.

7 Conclusions

The paper shows that excellent solutions can be reached on incompressible flow problems using the stabilized meshless finite point method.

The following statements must be taken into account in order to achieve correct answers.

1. The adequate stabilization must be used both for the convective terms in the momentum equation and for the incompressibility terms. The necessary stabilization for both terms is naturally introduced by the FIC procedure.
2. Essential boundary conditions may be imposed in the FPM directly by using the equation that approximates the velocity and pressure unknowns.
3. Natural boundary conditions must be introduced explicitely and must be stabilized. The FIC procedure has shown to be also adequate for this purpose.
4. The use of a fractional step algorithm allows the use of equal order approximation for velocities and pressure provided a correct stabilization of the incompressibility terms is introduced. The stabilization provided by the FIC approach has found to be essential to enhance the properties of the standard three step splitting scheme.

Acknowledgements. The authors thank Mr. J. García and Profs. O.C. Zienkiewicz and R.L. Taylor for many useful discussions.

References

1. Monaghan, J.J.: Smoothed particle hydrodynamics: Some recent improvement and applications. Annu. Rev. Astron. Physics 30: 543–574 (1992)
2. Randles, P.W., Libersky, L.D.: Smoothed particle hydrodynamics: Some recent improvement and applications. Comp. Meth. Appl. Mech. Eng. 139: 375–408 (1996)
3. Perrone, N., Kao, R.: A general finite difference method for arbitrary meshes. Comp. Struct 5: 45–47 (1975)
4. Liszka, T., Orkisz, J.: The finite difference method at arbitrary irregular grids and its application in applied mechanics. Comp. Struct. 11: 83–95 (1980)
5. Nayroles, B., Touzot, G., Villon, P.: Generalizing the FEM: Diffuse approximation and diffuse elements. Comput. Mechanics 10: 307–318 (1992)
6. Belytschko, T., Lu, Y., Gu, L.: Element free Galerkin methods. Int. J. Num. Meth. Eng. 37: 229–256 (1994)
7. Dolbow, J., Belytschko, T.: An introduction to programming the meshless element free Galerkin method. Archives of Comput. Meth. Eng. 5(3): 207–241 (1998)
8. Liu, W.K., Jun, S., LI, S., Adee, J., Belytschko, T.: Reproducing Kernel particle methods for structural dynamics. Int. J. Num. Meth. Eng. 38: 1655–1679 (1995)
9. Liu, W.K., Y Chen, Jun, S., Chen, J.S., Belytschko, T., Pan, C., Uras, R.A., Chang, C.T.: Overview and applications of the Reproducing Kernel particle method. Archives of Comput. Meth. Eng. 3(1): 3–80 (1996)
10. Oñate, E., Idelsohn, S., Zienkiewicz, O.C., Fisher, T.: A finite point method for analysis of fluid flow problems. Proceedings of the 9th Int. Conference on Finite Element Methods in Fluids. Venize, Italy: 15–21 October, (1995)
11. Oñate, E., Idelsohn, S., Zienkiewicz, O.C., Taylor, R.L.: A finite point method in computational mechanics. Applications to convective transport and fluid flow. Int. J. Num. Meth. Eng. 39: 3839–3866 (1996)
12. Oñate, E., Idelsohn, S., Zienkiewicz, O.C., Taylor, R.L.: A stabilized finite point method for analysis of fluid mechanics's problems. Comput. Meth. in Appl. Eng. 139(1–4): 315–347 (1996)

13. Oñate, E., Idelsohn, S.: A mesh free finite point method for advective-diffusive transport and fluid flow problems. Computational Mechanics 21: 283–292 (1988)
14. Oñate, E.: Derivation of stabilized equations for advective-diffusive transport and fluid flow problems. Comput. Meth. Appl. Mech. Eng. 151(1–2): 233–267 (1998)
15. Oñate, E., García, J., Idelsohn, S.: Computation of the stabilization parameter for the finite element solution of advective-diffusive problems. Int. J. Num. Meth. Fluids 25: 1385–1407 (1997)
16. Oñate, E., García, J., Idelsohn, S.: An alpha-adaptive approach for stabilized finite element solution of advective-diffusive problems with sharp gradients. Ladeveze, P., Oden, J.T. (eds.). New Adv. in Adaptive Comp. Meth. Mech. Elsevier 1998
17. Oñate, E.: A finite element method for incompressible viscous flows using a finite increment calculus formulation.Research Report N. 150, CIMNE, Barcelona, January 1999. Submitted to Comput. Meth. Appl. Mech. Eng.
18. Oñate, E., Manzán, M.: A general procedure for deriving stabilized space-time finite elements for advective-diffussive problems. Int. J. Num. Meth. Fluids 31: 203–221 (1999)
19. Zienkiewicz, O.C., Taylor, R.L.: The finite element method.. 4th Edition, Vol. 1, McGraw Hill 1989
20. Ilinca, F., Hétu, J.-F., Peletier, D.: On stabilized finite element formulations for incompressible advective-diffusive transport and fluid flow problems. AIAA-97-1863. Presented at AIAA CFD Conference, Snowman CO, USA, July 1997. Submitted to Comp. Meth. Appl. Mech. Eng.
21. Zienkiewicz, O.C., Codina, R.: A general algorithm for compressible and incompressible flow. Part I: The split characteristic based scheme. Int. J. Num. Meth. Fluids 20: 869–885 (1995)
22. Zienkiewicz, O.C., Morgan, K., Satya Sai, B.V.K., Codina, R., Vázquez, M.: A general algorithm for compressible and incompressible flow. Part II: Tests on the explicit form. Int. J. Num. Meth. Fluids 20(8–9): 886–913 (1995)
23. Zienkiewicz, O.C., Rojek, J., Taylor, R.L., Pastor, M.: Triangles and tetrahedra in explicit dynamic codes for solids. Int. J. Num. Meth. Eng. 43: 565–583 (1998)
24. Mizakami, A., Hughes, T.J.R.: A Petrov-Galerkin finite element method for convection dominated flows: an accurate upwinding technique for satisfying the maximum principle. Comput. Meth. Appl. Mech. Eng. 50: 181–193 (1985)
25. Hughes, T.J.R., Mallet, M.: A new finite element formulations for computational fluid dynamics: IV. A discontinuity capturing operator for multidimensional advective-diffusive system. Comput. Meth. Appl. Mech. Eng. 58: 329–336 (1986)
26. Galeao, A.C., Dutra do Carmo, E.G.: A consistent approximate upwind Petrov-Galerkin method for convection dominated problems. Comput. Meth. Appl. Mech. Eng. 68: 83–95 (1988)
27. Codina, R.: A discontinuity-capturing crosswind dissipation for the finite element solution of the convection-diffusion equation. Comput. Meth. Appl. Mech. Eng. 110: 325–342 (1993)
28. Oñate, E., García, J.: A stabilized finite element method for incompressible flows involving free surface waves. Publication PI-160, CIMNE, Barcelona 1999
29. Ghia, U., Ghia, K., Shin, C.: High-*Re* solutions for incompressible flow using the Navier-Stokes equation and a multi-grid method. J. Comp. Phys. 48: 387–441 (1982)
30. Armaly, B.F., Durst, F., Pereira, J.C.F., Schönung, B.: Experimental and theoretical investigations of backword-facing step flow. J. Fluid Mech. 127: 473–496 (1983)