Regular article

A finite point method for incompressible flow problems

Eugenio Oñate\textsuperscript{1}, Carlos Sacco\textsuperscript{1}, Sergio Idelsohn\textsuperscript{1,2}

\textsuperscript{1} International Center for Numerical Methods in Engineering, Universidad Politécnica de Cataluña Gran Capitán s/n, 08034 Barcelona, Spain
\textsuperscript{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 on the finite increment calculus (FIC) procedure developed by Oñate \cite{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 \cite{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 \cite{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 \cite{5}, the element free Galerking (EFG) method \cite{6,7} and the reproducing kernel particle (RKP) method \cite{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 \textit{finite point method} (FPM) proposed in \cite{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 \textit{stabilization} of the discrete differential equations. The stable form found by the \textit{finite element calculus} procedure presented in \cite{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 \cite{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 $\mathcal{D}$ be the interpolation domain (cloud) of a function $u(x)$ and let $s_j$ with $j = 1, 2, \ldots, n$ be a collection of $n$ points with coordinates $x_j \in \mathcal{D}$. The unknown function $u$ may be approximated within $\mathcal{D}$ by

\begin{equation}
  u(x) \approx \hat{u}(x) = \sum_{i=1}^{m} p_i(x) \alpha_i = p(x)^T \alpha
\end{equation}

where $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_m]^T$ 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

\begin{equation}
  p = [1, x, y]^T \quad \text{for } m = 3
\end{equation}

and

\begin{equation}
  p = [1, x, y, x^2, xy, y^2]^T \quad \text{for } m = 6 \quad \text{etc.}
\end{equation}
Function $u(x)$ can now be sampled at the $n$ points belonging to $\Omega_i$ giving

$$u^h = \begin{bmatrix} u^h_1 \\ u^h_2 \\ \vdots \\ u^h_n \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_n \end{bmatrix} = \begin{bmatrix} \rho_1^T \\ \rho_2^T \\ \vdots \\ \rho_n^T \end{bmatrix} \alpha = C\alpha \quad (4)$$

where $u^h_i = u(x_i)$ are the unknown but sought for values of function $u$ at point $j$, $\hat{u}_i = \hat{u}(x_i)$ are the approximate values, and $\rho_j = p(x_j)$.

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 $u^h_i$ values. This problem can be simply overcome by determining the $\hat{u}_i$ values by minimizing the sum of the square distances of the error at each point weighted with a function $\psi(x)$ as

$$J = \sum_{i=1}^{n} \psi(x_i)(u^h_i - \hat{u}(x_i))^2 = \sum_{i=1}^{n} \psi(x_i)(u^h_i - p^T\alpha)^2 \quad (5)$$

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

Function $\psi(x)$ is usually built in such a way that it takes a unit value in the vicinity of the point $i$ typically called “star node” where the function (or its derivatives) are to be computed and vanishes outside a region $\Omega_i$ surrounding the point. The region $\Omega_i$ can be used to define the number of sampling points $s$ in the interpolation region. A typical choice for $\psi(x)$ is the normalized Gaussian function and this has been chosen in the examples shown in the paper. Of course $s \geq m$ 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.

The minimization of (5) with respect to $\alpha$ gives

$$\alpha = C^{-1}u^h \quad , \quad C^{-1} = A^{-1}B \quad (6)$$

$$A = \sum_{j=1}^{s} \psi(x_j)p(x_j)p^T(x_j)$$

$$B = \left[ \psi(x_1)p(x_1), \psi(x_2)p(x_2), \ldots, \psi(x_n)p(x_n) \right] \quad (7)$$

The final approximation is obtained by substituting $\alpha$ from (6) into (1) giving

$$\hat{u}(x) = p^T C^{-1} u^h = N^T u^h = \sum_{j=1}^{n} N^j u^j \quad (8)$$

where the “shape functions” for the $i$-th star node are

$$N^j_i(x) = \sum_{m=1}^{m} p(x) \tilde{C}_{ij}^{-1} = p^T(x) \tilde{C}^{-1} \quad (9)$$

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

$$u(x_i) \simeq \hat{u}(x_i) \neq u^h_i \quad (10)$$

i.e. 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 $u^j$ are simply the unknown parameters sought.

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

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

$$\psi_i(x_i) = 1$$

$$\psi_i(x) \neq 0 \quad \text{if} \quad x \in \Omega_i, \quad \psi_i(x) = 0 \quad \text{if} \quad x \notin \Omega_i \quad (11)$$

Then the minimization square distance becomes

$$J = \sum_{i=1}^{n} \psi_i(x)(u^h_i - \hat{u}(x))^2 \quad \text{minimum} \quad (12)$$

The expression of matrices $A$ and $B$ coincide with (7) with

$$C = \sum_{i=1}^{n} \psi_i(x_i)p(x_i)p^T(x_i).$$

Note that according to (1), the approximate function $\hat{u}(x)$ is defined in each interpolation domain $\Omega_i$, in fact, different interpolation domains can yield different shape functions $N^j_i$. As a consequence a point belonging to 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 $\Omega_i$ 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-
A finite point method for incompressible flow problems

...ions 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.**

\[ r_{mi} - \frac{1}{2} \frac{\partial r_{mi}}{\partial x_j} = 0 \quad \text{in } \Omega \tag{13} \]

**Mass balance.**

\[ r_d - \frac{1}{2} \frac{\partial r_d}{\partial x_j} = 0 \quad \text{in } \Omega \tag{14} \]

where for the steady state case

\[ r_{mi} = \rho \left( \frac{\partial u_i u_j}{\partial x_j} + \frac{\partial p}{\partial x_i} - \frac{\partial u_j}{\partial x_j} - h_i \right) \tag{15} \]

\[ r_d = \frac{\partial u_i}{\partial x_i} \tag{16} \]

with \( i, j = 1, 2 \) for a two dimensional flow.

In (15) \( \rho \) is the fluid density (here assumed to be constant), \( u_i \) is the velocity component in the \( i \)-th direction, \( p \) the pressure, \( h_i \) the body forces and \( \tau_{ij} \) the viscous stress components relative to the velocity gradients through the fluid viscosity \( \mu \).

\[ \tau_{ij} = 2 \mu \left( \frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i} \right) \tag{17a} \]

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

Einstein summation convention for repeated indexes in products and derivatives is used, i.e., \( h_{ij} \frac{\partial u_j}{\partial x_i} = \sum_j h_{ij} \frac{\partial u_j}{\partial x_j} \).

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 \( h_m \) and \( h_d \) are defined as (for 2D problems)

\[ h_m = \begin{bmatrix} h_{m1} \\ h_{m2} \end{bmatrix}, \quad h_d = \begin{bmatrix} h_{d1} \\ h_{d2} \end{bmatrix} \tag{18} \]

where \( h_{m1} \) and \( h_{m2} \) are the dimensions of the finite control domain where balance of momentum is enforced. Similarly \( h_{d1} \) and \( h_{d2} \) represent the dimensions of the domain where mass conservation is expressed. The components of vectors \( h_m \) and \( h_d \) 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] as 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 \( h_{m1} = h_{m2} = 0 \) 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 \( \Gamma_t.**

\[ n_i \gamma_i - t_i + \frac{1}{2} h_{mi} n_i r_{mi} = 0 \quad \text{on } \Gamma_t \tag{19} \]

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 \( \Gamma_t \) of the analysis domain \( \Omega \).

**Prescribed velocity at the boundaries.**

\[ u_i = u_i^p \quad \text{on } \Gamma_{u_i} \tag{20} \]

\[ u_n - \frac{1}{2} h_{dn} n_i r_d = u_n^p \quad \text{on } \Gamma_{u_n} \tag{21} \]

In (20) \( u_i \) and \( u_i^p \) 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) \( \Gamma_{u_i} \) and \( \Gamma_{u_n} \) are the parts of the boundary \( \Gamma \) of \( \Omega \) where the tangential and normal velocities are prescribed, respectively. The Dirichlet boundary is defined as \( \Gamma_u = \Gamma_{u_i} \cup \Gamma_{u_n} \).

The underlined terms in (13) and (14) 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 a prescribed boundary. The details on how these terms are found in [14, 17].

2.1 Alternative form of stabilized governing equations

Let us express the components of the characteristic vector \( h_d \) for the mass balance equation as

\[ h_d = -2 \rho \gamma_{d} u_i \tag{22} \]

where the \( \gamma_d \) 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

\[ u_i \frac{\partial}{\partial x_i} \left( \frac{\partial u_j}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left( u_j \frac{\partial u_j}{\partial x_j} \right) - \left( \frac{\partial u_j}{\partial x_j} \right)^2 \tag{23} \]

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

\[ r_d - \gamma_d \frac{\partial r_{mi}}{\partial x_i} = 0 \tag{24a} \]
where

\[ \bar{F}_{m_j} = \rho u_j \frac{\partial u_j}{\partial x_j} + \frac{\partial p}{\partial x_j} - \frac{\partial}{\partial x_j} - b_i \]  

(24b)

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

\[ u_n - \tau_d n_i \bar{F}_{m_j} = u_n^0 \quad \text{on} \quad \Gamma \]

(25)

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

\textbf{Momentum.}

\[ r_{m_j} \frac{1}{2} h_{m_j} \frac{\partial \sigma_{m_j}}{\partial x_j} = 0 \quad \text{in} \quad \Omega \]

(26)

\textbf{Mass balance.}

\[ r_d \frac{1}{2} h_{m_j} \frac{\partial \sigma_{m_j}}{\partial x_j} = 0 \quad \text{in} \quad \Omega \]

(27)

\textbf{Boundary conditions.}

\[ n_j \sigma_{m_j} - t_i = \frac{1}{2} h_{m_j} n_j \bar{F}_{m_j} = 0 \quad \text{on} \quad \Gamma_t \]

(28)

\[ n_i u_j = 0 \quad \text{on} \quad \Gamma_{n_i} \]

(29)

\[ u_n - \tau_d n_i \bar{F}_{m_j} - u_n^0 = 0 \quad \text{on} \quad \Gamma_n \]

(30)

A similar form of the modified differential equations for momentum and mass balance ((26) and (27)) has been recently proposed by Linca 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 eliminating 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 written now as [14, 18]

\textbf{Momentum.}

\[ \left( r_{m_j} - \frac{h_{m_j} \sigma_{m_j}}{2} \frac{\partial \sigma_{m_j}}{\partial x_j} \right) - \frac{\delta}{2} \frac{\partial}{\partial t} \left( r_{m_j} - \frac{h_{m_j} \sigma_{m_j}}{2} \frac{\partial \sigma_{m_j}}{\partial x_j} \right) = 0 \]

(31)

\textbf{Mass balance.}

\[ \left( r_d - \frac{h_{m_j} \sigma_{m_j}}{2} \frac{\partial \sigma_{m_j}}{\partial x_j} \right) - \frac{\delta}{2} \frac{\partial}{\partial t} \left( r_d - \frac{h_{m_j} \sigma_{m_j}}{2} \frac{\partial \sigma_{m_j}}{\partial x_j} \right) = 0 \]

(32)

In above  \( \delta \) is a time stabilization parameter. Transient effects are also included in the term \( r_{m_j} \), given by

\[ r_{m_j} = \rho \left( \frac{\partial u_j}{\partial t} + \frac{\partial (u_j u_j)}{\partial x_j} + \frac{\partial p}{\partial x_j} - \frac{\partial}{\partial x_j} - b_i \right) \]

(33)

Equation (31) and (32) are obtained by expressing the balance of momentum and mass in space-time domains of finite dimensions \((h_m \times \delta)\) and \((h_d \times \delta)\), 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 \( \delta \) 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 \( t = 0 \))

\[ u^{n+1} = u^n - \frac{\Delta t}{\rho} \left[ \frac{\partial (u_j u_j)}{\partial x_j} + \frac{\partial p^{n+1}}{\partial x_j} - \frac{\partial}{\partial x_j} - b_i - \frac{h_{m_j} \sigma_{m_j}}{2} \frac{\partial \sigma_{m_j}}{\partial x_j} \right] \]

(34)

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

\[ u_t = u_t^n - \frac{\Delta t}{\rho} \left[ \frac{\partial (u_j u_j)}{\partial x_j} - \frac{\partial}{\partial x_j} - b_i - \frac{h_{m_j} \sigma_{m_j}}{2} \frac{\partial \sigma_{m_j}}{\partial x_j} \right] \]

(35)

\[ u_t^{n+1} = u_t^n - \frac{\Delta t}{\rho} \frac{\partial p^{n+1}}{\partial x_j} \]

(36)

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

Substituting (35) into (27) gives

\[ r_d = \frac{\Delta t}{\rho} \frac{\partial^2 p^{n+1}}{\partial x_j} - \tau_d \frac{\partial \sigma_{m_j}}{\partial x_j}^{n+1} = 0 \]

(37)
A finite point method for incompressible flow problems

where

\[ r^* = \frac{\partial u_i}{\partial x_i} \]

\[ \left[ \frac{\partial \rho}{\partial x_i} \right]_{n+1} = \frac{\partial}{\partial x_i} \left[ \rho \left( \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} \right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \right]^n - \frac{\partial \rho^*}{\partial x_i} \frac{\partial}{\partial x_i} \]  

(38)

The solution steps are the following:

**Step 1**

Solve explicitly for the so called “ fractional” velocities \( u_i^* \) using (35).

**Step 2**

Compute the pressure field \( \rho^* \) by solving the equation for the Laplacian of pressure derived from (37). Note that this equation in the domain has the following form

\[ \Delta t \frac{\partial \rho^*}{\partial x_i} + \frac{\partial ^2 \rho^*}{\partial x_i \partial x_i} = r^* - \tau \frac{\partial \rho}{\partial x_i} \times \left[ \rho \left( \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} \right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \right]^n \]  

(40)

Clearly for \( \tau = \tau \) above equation simplifies to

\[ \Delta t \rho^* + \frac{\partial ^2 \rho^*}{\partial x_i \partial x_i} = r^* \]  

(41)

where \( \Delta \) is the Laplacian operator and

\[ r^* \equiv r^* - \tau \frac{\partial \rho}{\partial x_i} \times \left[ \rho \left( \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} \right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \right]^n \]  

(42)

**Step 3**

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

Equation (41) differs slightly from the form typically used in fractional step schemes where the term involving \( \tau \) 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 viable extensions.

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 straightforward. Equation (8) is used to define the approximation of velocities and pressures within each cloud \( \Omega_i \) as

\[ \bar{u}_m = \sum_{j=1}^{n} N_j^* u_{m_j}; \quad m = 1, 2, 3 \quad \text{for 3D} \]  

(43)

\[ \bar{p} = \sum_{j=1}^{n} N_j^* p_j \]  

(44)

where \( \bar{u}_m \) denote approximate values and the shape functions \( N_j^* \) 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 \( u_{m_j}^* \) and \( p_j^* \).

**Step 1**

Compute explicitly the fractional velocities at each point \( k \) in the domain as

\[ \left( \delta \right) u_k = \left( \delta \right) u_k; \quad k = 1, \ldots, N; \quad i = 1, 2, 3 \]  

(45)

in which \( N \) is the total number of points in the domain and

\[ \left( \delta \right)^* u_k = \left( \delta \right)^* u_k; \quad k = 1, \ldots, N \]  

(46)

where \( \left( \delta \right) \) denotes approximate values.

Once the values of \( u_{m_j}^* \) have been obtained, the parameters \( u_{m_j}^* \) can be computed by solving the following system of equations

\[ \left( \Omega \right) u_k = \sum_{j=1}^{n} N_j^* u_{m_j}; \quad k = 1, \ldots, N \]  

(47)

Equation (47) is a system of \( N \) equations with \( N \) unknowns \( u_{m_j}^* \) and \( u_{m_j}^* \). Each equation contains 4 terms as in (47).

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 \( \tau_k = \tau \))

\[ K (\rho^*) = \rho^* \]  

(48)

where (for 2D problems)

\[ K_{ij} = \left( \frac{\Delta t}{\rho} \right)^* \left( \frac{\partial \rho_{ij}}{\partial x_1} + \frac{\partial \rho_{ij}}{\partial x_2} \right) \]  

(49)

\[ \rho^* = \rho^* \left[ \rho \left( \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} \right) - \frac{\partial \tau_{ij}}{\partial x_j} - b_i \right]^n \]  

(50)

As usual \( (\cdot)^n \) 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 \( \rho \beta_{k}^{p+1} \) 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

\[
\hat{u}_{k}^{n+1} = \left[ \hat{u}_{k}^{n} + \frac{\Delta t}{\rho} \frac{\partial p}{\partial x} \right]_{k} \quad k = 1, \ldots, N
\]  

(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.

\[
\frac{\partial \hat{u}_{i}}{\partial x_{i}} = \sum_{j=1}^{n} \frac{\partial u_{ij}^{i}}{\partial x_{i}} \hat{u}_{m}
\]

\[
\frac{\partial \hat{p}}{\partial x_{i}} = \sum_{j=1}^{n} \frac{\partial p_{ij}^{i}}{\partial x_{i}} \hat{p}_{j}
\]

(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 \( \Delta t = \frac{\Delta t_{k}}{d_{i}} \), where \( d_{i} \) is the minimum distance from a star point to any of its neighbours in the cloud. Note however that the full transient solution requires invariably the use of a global time step \( \Delta t_{k} \) equal for all nodes and defined as \( \Delta t_{k} = \min \{ \Delta t_{i} \} \), \( i = 1, \ldots, N \).

4 Boundary conditions

Prescribed tractions on the Neumann boundary \( \Gamma_{n} \), (19) or prescribed velocities at the Dirichlet boundaries \( \Gamma_{u} \) or \( \Gamma_{v} \), (20) and (21) may be imposed.

During the fractional step solution, the first explicit step (41) is performed at each point using the values of \( \hat{u}_{i} \) and \( \hat{p} \) at the previous time level. In the second step, two kinds of boundary conditions may be imposed: on boundaries where the normal velocity is imposed, (21) reads using (36)

\[
u_{k}^{n} = u_{k}^{n} \nu_{i} - \frac{\Delta t}{\rho} \frac{\partial p}{\partial x_{i}} \nu_{i} - \frac{1}{2} \rho d_{i} n_{i} r_{d}
\]

(53)

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

\[
u_{k}^{n} = u_{k}^{n} \nu_{i} - \frac{\Delta t}{\rho} \frac{\partial p_{k}^{n+1}}{\partial x_{i}} \nu_{i} - \frac{1}{2} \rho d_{i} n_{i} ((\Delta t + \tau) \Delta p + \Delta \hat{p}, \Delta \hat{p})_{T}
\]

(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 \( \Gamma_{u} \) boundary.

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

\[
\hat{p}_{i} = \sum_{j=1}^{N} N_{j}^{i} \hat{p}_{j} = 0
\]

(55)

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

During the third step the velocities \( \hat{u}_{k}^{n+1} \) 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 assigned directly. After that, the nodal parameters \( \hat{u}_{m}^{n+1} \), 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)

\[n_{i} \sigma_{j} + \frac{1}{2} \rho d_{i} n_{i} r_{d} = 0\]

(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 deficiency is then corrected by adding a "shock capturing" (SC) stabilization term [14–27].

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

\[h_{m} = h_{d}\]

(37)

The problem remains now finding the value of the characteristic length vectors \( h_{d} \). Indeed, the components of \( h_{d} \) can be assessed by using some relationship between the streamline and transversal directions to the flow [14–19].

In this work the SUPG assumption has been chosen for defining \( h_{m} \) as

\[h_{m} = h_{d} \frac{u}{|u|}\]

(58)

The streamline parameter \( h_{d} \) has been chosen for each cloud as the minimum distance \( d_{i} \) from a star point to any of its neighbours. 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 33293 points shown in Fig. 2. Around
A finite point method for incompressible flow problems

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 $Re = 589$

In this example, the flow is constricted to move in a 2D do-
main which presents a backwards step. The domain dimen-
sions 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 velo-
city free. The non-slip condition is used at the walls, except

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 rep-
resented on Fig. 7. In the rest of the domain a regular distrib-
ution of point is used. As on the previous example, equal order

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
cut.

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

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

Register for free at https://www.scipedia.com to download the version without the watermark
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 explicitly 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 term 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. Garcia and Profs. O.C. Zienkiewicz and R.L. Taylor for many useful discussions.

References

A finite point method for incompressible flow problems


