ELSEVIER

Comput. Methods Appl. Mech. Engrg. 139 (1996) 315–346

Computer methods in applied mechanics and engineering

A stabilized finite point method for analysis of fluid mechanics problems

E. Oñate*, S. Idelsohn[1], O.C. Zienkiewicz[2], R.L. Taylor[3], C. Sacco

International Center for Numerical Methods in Engineering (CIMNE), Universitat Politècnica de Catalunya, Edificio C1, Gran Capitán s/n, 08034, Barcelona, Spain

Received 24 April 1996

Abstract

In this paper a meshless procedure termed 'the finite point method' for solving convection–diffusion and fluid flow type problems is presented. The method is based on the use of a weighted least-square interpolation procedure together with point collocation for evaluating the approximation integrals. Special emphasis is given to the stabilization of the convective terms and the Neumann boundary condition which has been found to be essential to obtain accurate results. Some examples of application to diffusive and convective transport and compressible flow problems using quadratic FP interpolations are presented.

1. Introduction

It is widely acknowledged that 3D mesh generation remains one of the big challenges in both Finite Element (FE) [1] and Finite Volume (FV) [2–4] computations. Thus, given enough computer power, even the most complex problems in computational mechanics, such as the 3D solution of Navier–Stokes equations in fluid flow can be tackled accurately providing an acceptable mesh is available. The generation of 3D meshes, however, is despite major recent advances in this field, certainly the bottleneck in most industrial FE and FV computations and, in many cases, it can absorb far more time and cost than the numerical solution itself.

Different authors have recently investigated the possibility of deriving numerical methods where meshes are unnecessary. The first attempts were reported by some finite difference (FD) practitioners deriving FD schemes in arbitrary irregular grids [5–8]. Here, typically the concept of 'star' of nodes was introduced to derive FD approximations for each central node by means of local Taylor series expansions using the information provided by the number and position of nodes contained in each star [9].

An alternative class of methods named Smooth Particle Hydrodynamics (SPH), sometimes called the Free Lagrange methods, depend only on a set of disordered point or particles and has enjoyed considerable popularity in computational physics and astrophysics to model the motion and collision of stars [9].

Nayroles et al. [10] proposed a technique which they call the Diffuse Element (DE) method, where

* Corresponding author.

[1] Professor at Universidad Nacional del Litoral, Santa Fe, Argentina. Visiting Professor, CIMNE.

[2] Professor of Civil Engineering, University College, Swansea, UK. UNESCO Professor, UPC.

[3] Professor of Civil Engineering, University of California, Berkeley. Visiting Professor, CIMNE.

PII S0045-7825(96)01088-2

only a collection of nodes and a boundary description is needed to formulate the Galerkin equations. The interpolating functions are polynomials fitted to the nodal values by a weighted least-squares (WLS) approximation. Although no finite element mesh is explicitly required in this method, still some kind of 'auxiliary grid' was used in [10] in order to compute numerically the integral expressions derived from the Galerkin approach. Belytschko et al. [11,12] have proposed an extension of the DE approach which they call the element-free Galerkin (EFG) method. This provides additional terms in the derivatives of the interpolant considered unnecessary by Nayroles et al. [10]. In addition, a regular cell structure is chosen as the 'auxiliary grid' to compute the integrals by means of high-order quadratures. Duarte and Oden [13], Babuška and Melenk [14] and Taylor et al. [15] have recently formalized this type of approximation as a subclass of the so-called 'partition of unity' (PU) methods and they propose meshless and enhanced FE procedures using hierarchical PU interpolations.

Liu et al. [16–19] have developed a different class of 'gridless' multiple scale methods based on reproducing kernel and wavelet analysis. This technique termed Reproducing Kernel Particle (RKP) method introduces a new type of shape functions using an integral window transform. The window function can be *translated* and *dilated* around the domain, thus replacing the need to define elements and providing refinement.

In a recent work, Oñate et al. [20–23] have found that the weighted least-square interpolation with a simple point collocation technique for evaluating the approximation integrals is a promising Finite Point Method (FPM) for the numerical solution of a wide range of problems in computational mechanics. The advantages of this FPM compared with standard FEM is to avoid the necessity of mesh generation and compared with classical FDM is the facility to handle the boundary conditions and the non-structured distribution of points. Moreover, the FPM proposed seems to be as accurate as other numerical methods and the computing time to solve the differential equation is of the same order as for FE and FV methods using non-structured grids. Examples of application of the FPM to the solution of elliptic equations as well as to those governing convective transport and compressible fluid flow using linear WLS interpolations were reported in [20–23].

Another interesting conclusion reported in [22] is the comparison with other meshless techniques presented in the literature. Firstly, the use of a Gaussian weighting function improves considerably the results with respect to the standard least-square (LSQ) approach. Secondly, the sensitivity of any point data interpolation based procedure to a variable number of points in each interpolation domain (cloud) must be low enough to preserve the freedom of adding, moving or removing points. This sensitivity is very high in meshless techniques using the LSQ approximation, it is large in WLS methods with linear interpolations and it is quite low in WLS methods using quadratic interpolations which indicates some advantage of the latter for practical applications [22].

In this paper the FPM proposed in [20–23] is further extended to the solution of the advective–convective transport equations as well as those governing the flow of compressible fluids using a *quadratic WLS interpolation*. Here, the stabilization of the numerical algorithm is crucial to guarantee acceptable results similarly as it occurs in FD, FV and FE methods for fluid flow problems. A residual stabilization procedure, adequate for the FPM, is proposed in the paper. It is shown that the stabilization of both the convective terms and the Neumann boundary condition is necessary to ensure a correct solution in all cases. Examples of application to a number of fluid mechanics problems including the solution of some two-dimensional convective transport and compressible flow situations are also given.

In the next section some basic concepts of mesh-free techniques including some details of the different weighted least-square interpolations typically used are briefly described.

2. Basic concepts of mesh-free techniques

Let us assume a scalar problem governed by a differential equation

$$A(u) = b \quad \text{in } \Omega \tag{1}$$

with Neumann boundary conditions

$$B(u) = t \quad \text{in } \Gamma_t \tag{2a}$$

and Dirichlet (essential) boundary conditions

$$u - u_p = 0 \quad \text{in } \Gamma_u \tag{2b}$$

to be satisfied in a domain Ω with boundary $\Gamma = \Gamma_t \cup \Gamma_u$. In the above, A and B are appropriate differential operators, u is the problem unknown and b and t represent external forces or sources acting over the domain Ω and along the boundary Γ_t, respectively. Finally, u_p is the prescribed value of u over the boundary Γ_u.

The most general procedure of solving numerically the above system of differential equations is the weighted residual method in which the unknown function u is approximated by some trial approximation $\hat{u}$ and Eqs. (1) and (2) are replaced by [1]

$$\int_{\Omega} W_i[A(\hat{u}) - b]\,\mathrm{d}\Omega + \int_{\Gamma_t} \bar{W}_i[B(\hat{u}) - t]\,\mathrm{d}\Gamma + \int_{\Gamma_u} \bar{\bar{W}}_i[\hat{u} - u_p]\,\mathrm{d}\Gamma = 0 \tag{3}$$

with the weighting functions W_i, $\bar{W}_i$ and $\bar{\bar{W}}_i$ defined in different ways FE, FV and FD methods can be considered as particular cases of (3) and indeed so can all the meshless approximation procedures.

In order to keep a local character of the problem (leading to a banded matrix), function u must be approximated by a combination of locally defined functions as

$$u(x) \cong \hat{u}(x) = \sum_{i=1}^{n_p} N_i(x) u_i^h = \boldsymbol{N}^{\mathrm{T}}(x)\boldsymbol{u}^h \tag{4}$$

with n_p being the total number of points in the domain and the interpolation functions $N_i(x)$ satisfy

$$\begin{aligned} N_i(x) &\neq 0 \quad \text{if } x \in \Omega_i \\ N_i(x) &= 0 \quad \text{if } x \notin \Omega_i \end{aligned} \tag{5}$$

Here, Ω_i is a subdomain of Ω containing n points, $n \ll n_p$. In (4) u_i^h is the *approximate value* of u at point i such that $u(x_i) \simeq u_i^h$.

In FE and FV methods the Ω_i subdomains are divided into elements and the N_i function may have some discontinuities (in the function itself or in its derivatives) on the element interfaces. In the FE method the weighting functions W_i are defined in 'weighting domains' which usually coincide precisely with the interpolating domains Ω_i. In cell vertex FV the interpolation and integration domains also coincide, however in the cell centered case they are different [2–4,22].

A common feature of FE and FV methods is that they both require a mesh for interpolation purposes and also to compute the integrals in Eq. (3). FE methods define the shape functions N_i over non-overlapping regions (elements) the assembly of which constitutes the domain Ω_i [1]. Different interpolations are therefore possible for a given number of points simply by changing the orientation or the form of these regions. Although FV techniques do not explicitly define an interpolation of the form (4), it is well known that they are equivalent to using linear shape functions over domains Ω_i defined in the same manner as in the FE method [2–4].

The way to define the shape functions N_i is classical in the FEM [1]. The process will be repeated here in order to compare and better understand the differences with the FPM.

Let Ω_i 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 $x_j \in \Omega_i$. The unknown function u may be approximated within Ω_i by

$$u(x) \cong \hat{u}(x) = \sum_{l=1}^{m} p_l(x)\alpha_l = \boldsymbol{p}(x)^{\mathrm{T}}\boldsymbol{\alpha} \tag{6}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_m]^{\mathrm{T}}$ and vector $\boldsymbol{p}(x)$ 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

$$p = [1, x, y]^{\mathrm{T}} \quad \text{for } m = 3 \tag{7a}$$

and

$$p = [1, x, y, x^2, xy, y^2]^{\mathrm{T}} \quad \text{for } m = 6 \quad \text{etc.} \tag{7b}$$

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

$$\boldsymbol{u}^h = \begin{Bmatrix} u_1^h \\ u_2^h \\ \vdots \\ u_n^h \end{Bmatrix} \cong \begin{Bmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_n \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{p}_1^{\mathrm{T}} \\ \boldsymbol{p}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{p}_n^{\mathrm{T}} \end{Bmatrix} \boldsymbol{\alpha} = \boldsymbol{C}\boldsymbol{\alpha} \tag{8}$$

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 $\boldsymbol{p}_j = \boldsymbol{p}(x_j)$.

In the FE approximation the number of points is chosen so that $m = n$. In this case $\boldsymbol{C}$ is a square matrix and we can obtain *after equaling $\boldsymbol{u}^h$ with $\boldsymbol{C}\boldsymbol{\alpha}$* in (8)

$$\boldsymbol{\alpha} = \boldsymbol{C}^{-1}\boldsymbol{u}^h \tag{9}$$

and

$$u \cong \hat{u} = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{u}^h = \boldsymbol{N}^{\mathrm{T}}\boldsymbol{u}^h = \sum_{j=1}^{n} N_j^i u_j^h \tag{10}$$

with

$$\boldsymbol{N}^{\mathrm{T}} = [N_1^i, \ldots, N_n^i] = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{C}^{-1} \quad \text{and} \quad N_j^i = \sum_{l=1}^{m} p_l(x) C_{lj}^{-1} \tag{11}$$

The shape functions $N_j^i(x)$ satisfy the standard condition [1]

$$\begin{aligned} N_j^i(x_i) &= 1 \quad j = i \\ &= 0 \quad j \neq i \quad i, j = 1, \ldots, n \end{aligned} \tag{12}$$

Furthermore, for two differents interpolating domains Ω_i and Ω_k the corresponding shape functions are the same, i.e.

$$N_j^i = N_j^k$$

The development of the N_j^i can often be performed directly using interpolation methods and/or isoparametric concepts [1].

If $n > m$, $\boldsymbol{C}$ is no longer a square matrix and the approximation cannot 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

$$J = \sum_{j=1}^{n} \varphi(x_j)(u_j^h - \hat{u}(x_j))^2 = \sum_{j=1}^{n} \varphi(x_j)(u_j^h - \boldsymbol{p}_j^{\mathrm{T}}\boldsymbol{\alpha})^2 \tag{13}$$

with respect to the $\boldsymbol{\alpha}$ 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 vicinity of the point i (typically called 'star node' [8]) 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. Of course $n \geqslant m$ 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 Eq. (13) with respect to $\boldsymbol{\alpha}$ gives

$$\boldsymbol{\alpha} = \bar{\boldsymbol{C}}^{-1}\boldsymbol{u}^h , \qquad \bar{\boldsymbol{C}}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{B} \tag{14}$$

with matrices A and $\boldsymbol{B}$ given by

$$\boldsymbol{A} = \sum_{j=1}^{n} \varphi(x_j)\boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{15a}$$

$$\boldsymbol{B} = [\varphi(x_1)\boldsymbol{p}(x_1), \varphi(x_2)\boldsymbol{p}(x_2), \ldots , \varphi(x_n)\boldsymbol{p}(x_n)] \tag{15b}$$

Matrix A may be written as

$$\boldsymbol{A} = [\boldsymbol{p}_1, \boldsymbol{p}_2 \cdots \boldsymbol{p}_n]\begin{pmatrix} \varphi(x_1) & 0 & \cdots & \cdots \\ 0 & \varphi(x_2) & 0 & \cdots \\ \cdots & \cdots & \ddots & \cdots \\ \cdots & \cdots & 0 & \varphi(x_n) \end{pmatrix}\begin{bmatrix} \boldsymbol{p}_1^{\mathrm{T}} \\ \boldsymbol{p}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{p}_n^{\mathrm{T}} \end{bmatrix} \tag{16}$$

where

$$\boldsymbol{p}_j = \boldsymbol{p}(x_j) = \begin{Bmatrix} p_1(x_j) \\ p_2(x_j) \\ \vdots \\ p_m(x_j) \end{Bmatrix} \tag{17}$$

The final approximation is still given by Eq. (10) now however substituting matrix $\boldsymbol{C}$ by $\bar{\boldsymbol{C}}$. The new shape functions are therefore

$$N_j^i(x) = \sum_{l=1}^{m} p_l(x)\bar{C}_{lj}^{-1} = \boldsymbol{p}^{\mathrm{T}}(x)\bar{\boldsymbol{C}}^{-1} \tag{18}$$

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

$$u(x_j) \simeq \hat{u}(x_j) \neq u_j^h \tag{19}$$

i.e. the local values of the approximating function do not fit the nodal unknown values (Fig. 1). Indeed, $\hat{u}$ is the true approximation for which we shall seek the satisfaction of the differential equation and boundary conditions and u_j^h are simply the unknown parameters sought!

However, if $n = m$ the FEM type approximation is recovered. Then, $\hat{u}(x_j) = u_j^h$ and once again conditions (12) are satisfied.

2.1. Fixed least-square approximation (FLS)

The weighted least-square approximation described above, depends to a great extent 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 (see Fig. 1(a)). Let $\varphi_i(x)$ be a weighting functions satisfying

$$\begin{cases} \varphi_i(x_i) = 1 \\ \varphi_i(x) \neq 0 & x \in \Omega_i \\ \varphi_i(x) = 0 & x \not\in \Omega_i \end{cases} \tag{20}$$

Then, the minimization square distance becomes

$$J = \sum_{j=1}^{n} \varphi_i(x_j)(u_j^h - \hat{u}(x_j))^2 \quad \text{minimum} \tag{21}$$

The expression of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ in Eq. (14) are now

$$\boldsymbol{A} = \sum_{j=1}^{n} \varphi_i(x_j)\boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{22a}$$

$$\boldsymbol{B} = [\varphi_i(x_1)\boldsymbol{p}(x_1), \varphi_i(x_2)\boldsymbol{p}(x_2), \ldots , \varphi_i(x_n)\boldsymbol{p}(x_n)] \tag{22b}$$

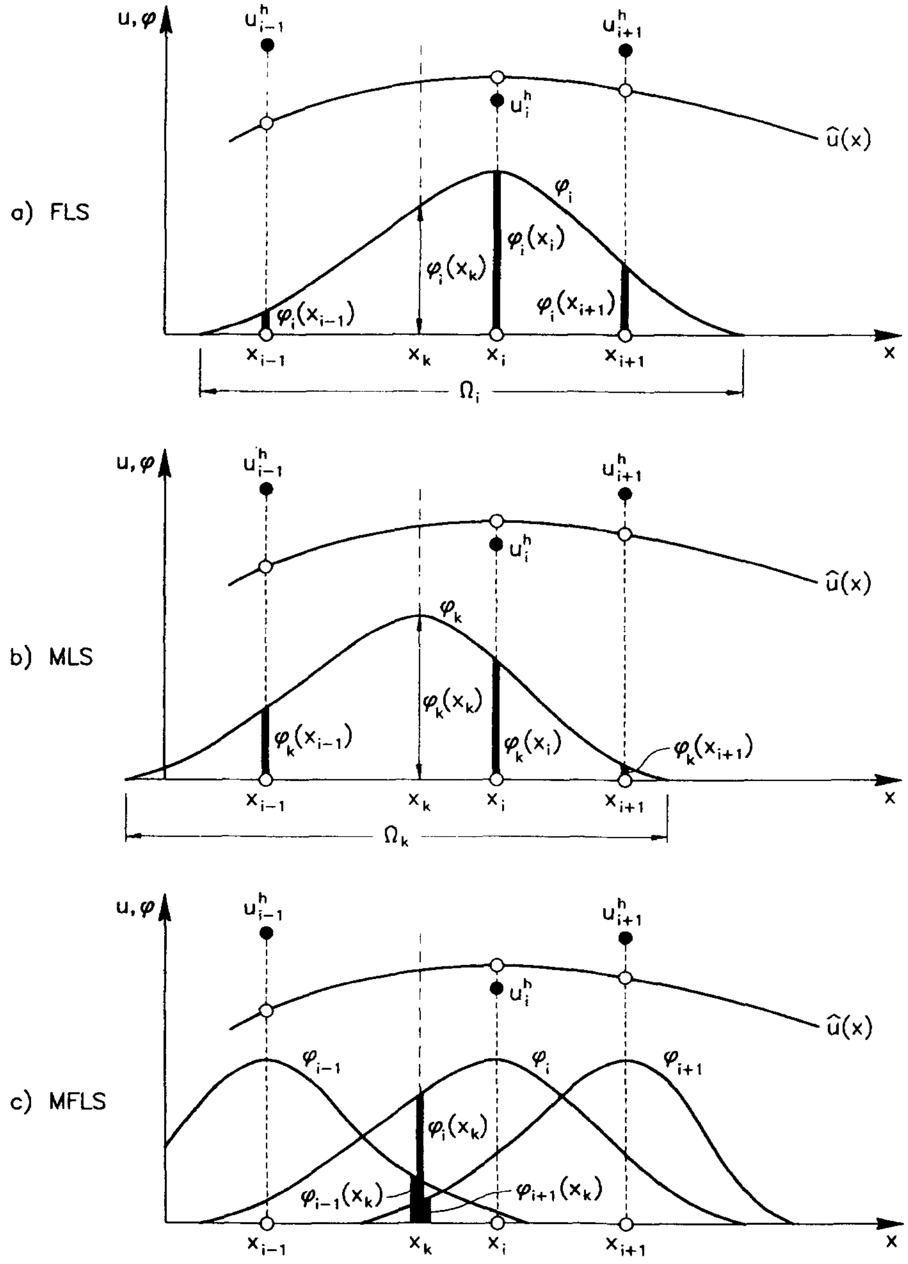

Fig. 1. Differents weighting least-square procedures. (a) Fixed least square (FLS); (b) moving least square (MLS); (c) multiple fixed least square (MFLS).

Note that according to (6), 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^i_j. As a consequence a point belonging to two or more overlapping interpolation domains has different values of the shape functions which means that $N^i_j \neq N^k_j$. 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 [22].

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.

2.2. Moving least-square (MLS) approximation

In the moving least-square (MLS) approach the weighting function φ is defined in shape and size and is translated over the domain so that it takes the maximum value over the position identified by the coordinate x_k where the unknown function $\hat{u}$ is to be evaluated. Note that x_k is an arbitrary position and not necessarily coincident with one of the x_i points defined in Eq. (4).

As shown in Fig. 1(b), we now minimize for any arbitrary coordinate x_k the following functional

$$J(x_k) = \sum_{j=1}^{n} \varphi_k(x_j)(u_j^h - \boldsymbol{p}_j^{\mathrm{T}} \boldsymbol{\alpha})^2 \tag{23}$$

where φ_k can in general change its shape and span depending on the position of point x_k. Note again that x_k is now an arbitrary coordinate position and it can be simply replaced by the global coordinate x. We will however retain the form $\varphi_k(x_j - x_k)$ to emphazise the possibility of changing the function φ at each position within the approximation domain. Function φ_k is defined by

$$\begin{cases} \varphi_k(x_k) = 1 & \\ \varphi_k(x) \neq 0 & x \in \Omega_k \\ \varphi_k(x) = 0 & x \not\in \Omega_k \end{cases} \tag{24}$$

where Ω_k is a subdomain of Ω around an arbitrary position x_k. Observe that now J is a function of the position x_k, and then the $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices are also a function of x_k. This must be taken into account when computing the derivatives of the shape functions, i.e.

$$\boldsymbol{A}(x_k) = \sum_{j=1}^{n} \varphi_k(x_j)\boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{25a}$$

$$\boldsymbol{B}(x_k) = [\varphi_k(x_1)\boldsymbol{p}(x_1), \varphi_k(x_2)\boldsymbol{p}(x_2) \cdots, \varphi_k(x_n)\boldsymbol{p}(x_n)] \tag{25b}$$

Note also that the parameters α_i are no longer constants but vary continuously with the position x_k and that inversion of matrices is required at every point where $\hat{u}$ is to be evaluated.

Furthermore, a unique global definition of the shape functions can now be obtained provided:

(a) the weighting function φ_k is continuous and differentiable in Ω_k,

(b) the weighting function φ_k vanishes on the boundary of Ω_k and outside,

(c) the number of points n within Ω_k is equal or greater than the parameters m at all points in Ω_k.

In this case the MLS shape function of an arbitrary position x_k is

$$N_j^k(x) = \sum_{l=1}^{m} p_l(x)\bar{C}_{lj}^{-1}(x_k) \tag{26}$$

with

$$\bar{\boldsymbol{C}}^{-1}(x_k) = \boldsymbol{A}^{-1}(x_k)\boldsymbol{B}(x_k) \tag{27}$$

2.3. Multiple fixed least-square (MFLS) approximation

In general, with an arbitrary definition of points the problem of specifying φ_k at every position x_k is very difficult and presents an infinite number of possibilities. In order to avoid this difficulty but preserving the unique global definition of the shape functions obtained in the MLS, Oñate et al. have recently proposed a new possibility [22]. The idea is to define first a weighting function φ_i at any x_i point as in the FLS approximation. These functions are subsequently used to weight the square distances at an arbitrary position x_k. For a constant grid spacing with an invariant shape of the weighting function φ_k, it is possible to write

$$\varphi_k(x_j) = \varphi_j(x_k) \tag{28}$$

Introducing this equality in (23) we have

$$J(x_k) = \sum_{j=1}^{n} \varphi_j(x_k)(u_j^h - \boldsymbol{p}_j^{\mathrm{T}}\boldsymbol{\alpha})^2 \tag{29}$$

The least-square problem leads to the values of $\boldsymbol{\alpha}$ given by Eq. (14) with

$$\boldsymbol{A}(x_k) = \sum_{j=1}^{n} \varphi_j(x_k)\boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{30}$$

and

$$\boldsymbol{B}(x_k) = [\varphi_1(x_k)\boldsymbol{p}(x_1), \varphi_2(x_k)\boldsymbol{p}(x_2), \cdots \varphi_n(x_k)\boldsymbol{p}(x_n)] \tag{31}$$

This method will be termed multiple fixed least-square (MFLS) approximation since it makes use of different fixed functions φ_j to weight the square distances in $J(x_k)$ (Fig. 1(c)).

This approximation is coincident with the MLS method just for an invariant shape of the weighting functions (i.e. $\varphi_k = \varphi_l = \varphi$). This algorithm also produces solutions for $\boldsymbol{\alpha}$ which depend on the position x_k. Note that the definition of the shape function is still *unique* and coincides with Eq. (26) providing the three conditions of continuity and differentiability described for the MLS method are fulfilled.

2.4. Order of the approximation of the shape functions

In the FLS method the $\boldsymbol{\alpha}$ parameters are constant in each Ω_i domain and the order of the approximation is directly the order included in the definition of $\boldsymbol{p}(x)$. On the other hand, in both the MLS and MFLS methods, the $\boldsymbol{\alpha}$ parameters are functions of the position x_k, and the interpolation to the $\hat{u}$ unknown function may include higher-order shapes. Nevertheless, it is important to show that in both cases, the interpolation defined can approximate at least all the functions used to define $\boldsymbol{p}(x)$. The following proof follows the arguments given in [15].

Consider the set of approximations

$$\boldsymbol{u} = \sum_{j=1}^{n} N_j^k(x)\boldsymbol{u}_j \tag{32}$$

where

$$\boldsymbol{u} = [\hat{u}_1(x), \hat{u}_2(x), \ldots, \hat{u}_n(x)] \tag{33}$$

and

$$\boldsymbol{u}_j = [u_{j1}^h, u_{j2}^h, \ldots, u_{jn}^h] \tag{34}$$

are the local values of the approximation, i.e. u_{ji}^h are the sought parameters corresponding to the approximate function $\hat{\boldsymbol{u}}_i$.

Assigning to each u_{jl}^h the value of the polynomial $p_l(x_j)$ (i.e. the lth entry in $\boldsymbol{p}$) gives

$$\boldsymbol{u}_j = \boldsymbol{p}^{\mathrm{T}}(x_j) \tag{35}$$

Now, using the definition of the interpolation functions (see Eq. (16)) we have

$$\boldsymbol{u} = \sum_{j=1}^{n} N_j^k(x)\boldsymbol{p}^{\mathrm{T}}(x_j) = \boldsymbol{p}^{\mathrm{T}}(x)\boldsymbol{A}^{-1}(x_k)\boldsymbol{B}(x_k)\boldsymbol{P} \tag{36}$$

where

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{p}_1^{\mathrm{T}} \\ \boldsymbol{p}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{p}_n^{\mathrm{T}} \end{bmatrix} \quad \text{with} \quad \boldsymbol{p}_i^{\mathrm{T}} = \boldsymbol{p}(x_i)^{\mathrm{T}} \tag{37}$$

After substitution of the definition of $\boldsymbol{B}(x_k)$ of Eq. (31) yields

$$\boldsymbol{u} = \boldsymbol{p}^{\mathrm{T}}(x)\boldsymbol{A}^{-1} \sum_{j=1}^{n} \varphi_j(x_k)\boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{38}$$

and

$$\boldsymbol{u} = \boldsymbol{p}^{\mathrm{T}}(x)\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{p}^{\mathrm{T}}(x) \tag{39}$$

which shows that the form can interpolate exactly any function included as part of the definition of $\boldsymbol{p}(x)$. Similar results can be found if the definitions of $\boldsymbol{A}$ and $\boldsymbol{B}$ in the MLS is used. If polynomials are used to define the functions, the interpolation always includes exact representations for each included polynomial. Inclusion of the zero-order polynomial (i.e. $m = 1$), implies that

$$\sum_{j=1}^{n} N_j^k(x) = 1$$

In mathematics this is called a *partition of unity* (provided it is true for all points, x, in the domain) [14]. It is easy to recognize that this is the same requirement as applied to most finite element shape functions.

2.5. Derivatives of the shape functions

From Eqs. (10) and (14)

$$u(x) \cong \hat{u}(x) = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{u}^h = \boldsymbol{N}^{\mathrm{T}}\boldsymbol{u}^h \tag{40}$$

with

$$\boldsymbol{N}^{\mathrm{T}} = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{B} \tag{41}$$

In the FLS approximation the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are constant over the Ω_i domain, and then the derivatives of the shape functions are directly

$$\frac{\partial \boldsymbol{N}^{\mathrm{T}}}{\partial x} = \frac{\partial \boldsymbol{p}^{\mathrm{T}}}{\partial x}\boldsymbol{A}^{-1}\boldsymbol{B} \tag{42}$$

In the MLS and MFLS, the derivatives may be obtained using the next procedure which follows the ideas presented in [15]. Let us write

$$\boldsymbol{N}^{\mathrm{T}} = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{W} \tag{43}$$

where matrix $\boldsymbol{W}$ is defined in such a way that

$$\boldsymbol{A}\boldsymbol{W} = \boldsymbol{B} \tag{44}$$

For example, the first derivative in $x = x_k$ is given by

$$\frac{\partial \boldsymbol{N}^{\mathrm{T}}}{\partial x} = \frac{\partial \boldsymbol{p}^{\mathrm{T}}}{\partial x}\boldsymbol{W} + \boldsymbol{p}^{\mathrm{T}}\frac{\partial \boldsymbol{W}}{\partial x} \tag{45}$$

and

$$\boldsymbol{A}\frac{\partial \boldsymbol{W}}{\partial x} + \frac{\partial \boldsymbol{A}}{\partial x}\boldsymbol{W} = \frac{\partial \boldsymbol{B}}{\partial x} \tag{46}$$

where for the MLS approximation is

$$\frac{\partial \boldsymbol{A}}{\partial x} = \sum_{j=1}^{n} \frac{\partial \varphi_k(x_j)}{\partial x}\boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{47}$$

and

$$\frac{\partial \boldsymbol{B}}{\partial x} = \left[\frac{\partial \varphi_k(x_1)}{\partial x}\boldsymbol{p}(x_1), \frac{\partial \varphi_k(x_2)}{\partial x}\boldsymbol{p}(x_2) \cdots, \frac{\partial \varphi_k(x_n)}{\partial x}\boldsymbol{p}(x_n)\right] \tag{48}$$

and for the MFLS is

$$\frac{\partial \boldsymbol{A}}{\partial x} = \sum_{j=1}^{n} \frac{\partial \varphi_j(x_k)}{\partial x} \boldsymbol{p}(x_j)\boldsymbol{p}^{\mathrm{T}}(x_j) \tag{49}$$

and

$$\frac{\partial \boldsymbol{B}}{\partial x} = \left[\frac{\partial \varphi_1(x_k)}{\partial x} \boldsymbol{p}(x_1), \frac{\partial \varphi_2(x_k)}{\partial x} \boldsymbol{p}(x_2) \cdots, \frac{\partial \varphi_n(x_k)}{\partial x} \boldsymbol{p}(x_n) \right] \tag{50}$$

Higher derivatives may be computed by repeating the above process to define the higher derivatives of $\boldsymbol{W}$.

Nayroles et al. [10] suggest that approximations ignoring the derivatives of $\boldsymbol{\alpha}$ may be used to define the derivatives of the interpolation functions. Accordingly, as an approximation they propose the use of

$$\frac{\partial \boldsymbol{N}}{\partial x} \cong \frac{\partial \boldsymbol{p}^{\mathrm{T}}}{\partial x} \boldsymbol{A}^{-1}\boldsymbol{B} \tag{51}$$

for the first derivative. This approximation simplifies the construction of derivatives as it is no longer necessary to compute the derivatives for $\boldsymbol{A}$ and $\boldsymbol{B}$.

3. Derivation of the discretized equations

The selection of different weighting functions in the general weighted residual form of Eq. (3) yields different sets of discretized equations. In order to preserve the mesh-free character of the method, the weighting domain must be defined independently of any mesh. All approximation methods of integral type (i.e. Galerkin, area collocation, etc.) necessitate the introduction of complex procedures for integration (i.e. background grid, etc. [10–12,16–19]). Some of these procedures are reviewed in [20,22]. In this paper we shall therefore limit the choice to *point collocation* methods where we feel the advantages of meshless procedures are best realized.

Such point collocation has recently been used with success for solution of inviscid/viscous flows by Batina [24] who however limited his work to the use of LSQ method and linear approximation. We have found that considerable improvement can be gained using weighted least-square procedures [20–23].

Point collocation implies making $W_i = \bar{W}_i = \bar{\bar{W}}_i = \delta_i$ in Eq. (3) where δ_i is the Dirac delta. This gives the set of equations

$$[A(\hat{u})]_i - b_i = 0 \quad \text{in } \Omega \tag{52}$$

$$[B(\hat{u})]_i - t_i = 0 \quad \text{in } \Gamma_t \tag{53}$$

$$\hat{u}_i - u_p = 0 \quad \text{in } \Gamma_u \tag{54}$$

Any of the previous shape functions may be used now to approximate $\hat{u}$ leading in all cases to the system of equations

$$\boldsymbol{K}\boldsymbol{u}^h = \boldsymbol{f} \tag{55}$$

with $K_{ij} = [A(N_j)]_i + B(N_j)]_i$ and where the symmetry of the 'coefficient' matrix $\boldsymbol{K}$ is not generally achieved. Vector $\boldsymbol{u}^h$ contains the problem unknowns, u_i^h, and $\boldsymbol{f}$ is a vector containing the contributions from the force terms b and t and the prescribed values u_p.

Taking a particular set of nodes and shape functions, this method is coincident with the generalized Finite Difference Methods of the type described in [5,7,8]. However, we feel that the approach proposed here offers more possibilities. Indeed, any of the interpolation techniques described in Section 2 can be used.

4. Stabilizing the finite point method in convection–diffusion problems

For non-self adjoint problems such as occur in fluid mechanics special treatment is needed to stabilize the numerical approximation [1]. As a typical example we shall outline here the special feature on the convection–diffusion equation given by

$$A(\phi) = c\frac{\partial \phi}{\partial t} + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{\nabla}\phi - \boldsymbol{\nabla}^{\mathrm{T}}(k\boldsymbol{\nabla}\phi) - Q = 0 \quad \text{in } \Omega \tag{56a}$$

$$B(\phi) = \boldsymbol{n}^{\mathrm{T}}k\boldsymbol{\nabla}\phi + \bar{q}_n = 0 \quad \text{in } \Gamma_t \tag{56b}$$

$$\phi - \phi_p = 0 \quad \text{in } \Gamma_u \tag{56c}$$

with the initial condition $\phi = \phi_0(\boldsymbol{x})$ for $t = t_0$.

In (56) $\boldsymbol{\nabla}$ is the gradient operator, c and k are known physical parameters, $\boldsymbol{u}$ is the velocity vector, ϕ the unknown field and Q a source term. $\bar{q}_n$ and ϕ_p are known values of the flux and the unknown function at the boundaries Γ_t and Γ_u, respectively, and $\boldsymbol{n}$ is the normal to the boundary.

Among the techniques typically used to stabilize FD, FE and FV methods we can list upwind finite difference derivatives, anisotropic balancing diffusion, Petrov–Galerkin weighting functions and characteristic time integration [1]. an alternative stabilization procedure appropriate for FPM which reproduces the best features of some of these techniques is described next.

4.1. Residual stabilization technique

A simple stabilization method can be derived by writing the flow balance equations in a finite domain [25]. The process is described in the Appendix for the simple 1D convection-diffusion problem leading to the following stabilized form of the steady state conservation equation

$$A(\phi) = r - \frac{h}{2}\frac{\partial r}{\partial x} = 0 \tag{57}$$

where

$$r = -u\frac{\partial \phi}{\partial x} + \frac{\partial}{\partial x}\left(k\frac{\partial \phi}{\partial x}\right) + Q \tag{58}$$

in Eq. (57) h is the length of the domain (here onwards termed *characteristic length*) where flow balance is enforced. This length can be written as $h = 2\tau u$ where τ is the so-called *intrinsic time scale* in the FE convective transport and fluid flow literature [27]. The transient form of Eq. (57) is readily obtained as (see Appendix)

$$A(\phi) = c\frac{\partial \phi}{\partial t} - r + \frac{h}{2}\frac{\partial r}{\partial x} = 0 \tag{59}$$

The extension to 2D problems is straightforward as shown in [25] and gives

$$\text{steady state:} \quad A(\phi) = r - \frac{h}{2|\boldsymbol{u}|}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{\nabla}r = 0 \tag{60}$$

$$\text{transient:} \quad A(\phi) = c\frac{\partial \phi}{\partial t} - r + \frac{h}{2|\boldsymbol{u}|}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{\nabla}r = 0 \tag{61}$$

where in general

$$r = -\boldsymbol{u}^{\mathrm{T}}\boldsymbol{\nabla}\phi + \boldsymbol{\nabla}^{\mathrm{T}}k\boldsymbol{\nabla}\phi + Q \tag{62}$$

The stabilization term in Eqs. (60) and (61) can be expressed in terms of the intrinsic time simply making $h = 2\tau|\boldsymbol{u}|$.

The discretization in time of Eq. (61) can be written using a backward integration scheme as

$$\Delta\phi = \Delta t\left[r - \frac{h}{2|\boldsymbol{u}|}\boldsymbol{u}^{\mathrm{T}}\nabla r\right]^n \tag{63}$$

where $\Delta\phi = \phi^{n+1} - \phi^n$ and $(\cdot)^n$ denotes values computed at time t_n.

Eq. (63) can be found to be *identical* to that obtained using a *characteristic approximation* as described in [28] if the distance h is expressed by $h = |\boldsymbol{u}|\,\Delta t$.

The expression of the balancing term arising in the above equations can be simplified if a linear or quadratic approximation is used as the cubic derivatives of ϕ in the diffusive term are zero. A further simplification arises if the source term is constant and the balancing term in Eqs. (60) and (61) is then simply given by

$$\frac{h}{2|\boldsymbol{u}|}\boldsymbol{u}^{\mathrm{T}}(\boldsymbol{u}^{\mathrm{T}}\nabla\phi) \tag{64}$$

Note that the balancing term may be interpreted now as an additional diffusion as typically occurs in the FE literature [1].

The value of the optimal characteristic length h can be found by using the same arguments of standard upwinding and Petrov–Galerkin FE procedures, i.e. by searching exact nodal values for the simple 1D problem with $Q = 0$. Application of this concept to the FPM gives [20,22]

$$\begin{aligned} h &= \frac{\alpha\bar{h}}{2} \quad \text{for linear interpolations } (m = 2) \\ &= \frac{\alpha\bar{h}}{4} \quad \text{for quadratic interpolations } (m = 3) \end{aligned} \tag{65}$$

where

$$\alpha = \coth|\mathrm{Pe}| - \frac{1}{|\mathrm{Pe}|} \tag{66}$$

with the Peclet number defined as $\mathrm{Pe} = (|\boldsymbol{u}|\bar{h})/2k$.

In Eq. (66) $\bar{h}$ is the distance measured along the streamline between the end points for a particular cloud as shown in Fig. 2.

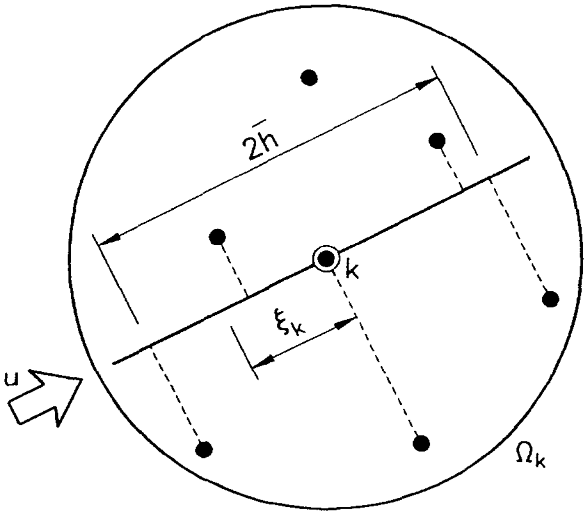

Fig. 2. Definition of the characteristic distance $\bar{h}$.

4.2. Treatment of boundary conditions

The essential boundary conditions are simply satisfied pointwise on points x_i placed on Γ_u as

$$\phi(x_i) = \phi_p(x_i) \quad x_i \text{ on } \Gamma_u \tag{67}$$

Other techniques to impose the essential boundary conditions are discussed in [22]. The authors have found that the straightforward satisfaction of the Neumann boundary condition in the FPM via Eq. (53) leads to unstable results. This problem can be overcome by using the same residual stabilization technique as described above, applied now to the Neumann boundary condition.

This can be achieved by rewriting Eq. (56b) (for the 1D case) as

$$B(\phi) = k\frac{\partial \phi}{\partial x} + \bar{q}_n - \frac{h}{2}r = 0 \tag{68}$$

where h is the *characteristic distance* defined in the previous section.

Eq. (68) can be derived from the flow balance condition in a boundary domain of length $h/2$ as described in the Appendix where the extension to 2D problems is also shown. For further details see [25].

The new stabilized governing differential equations in the domain and the Neumann boundary conditions are discretized in space using the FP approximation as previously described.

Oñate [25] has shown that the application of the standard Galerkin FE weighted residual approach to the stabilized Eqs. (60) and (68) leads to a system of discretized equations analogous to those obtained with the so-called Petrov–Galerkin and SUPG FE procedures [1,27]. Indeed for certain values of the characteristic distance h all stabilization methods typically used in FE computations can be reproduced [25].

4.3. Convection–diffusion examples

The first example shows the importance of using a weighted interpolation. Fig. 3 shows the numerical results for the solution of the one-dimensional convection–diffusion equation for Pe = 1. The numerical results have been obtained with the FPM using 1D quadratic base polynomials ($m = 3$) and five point

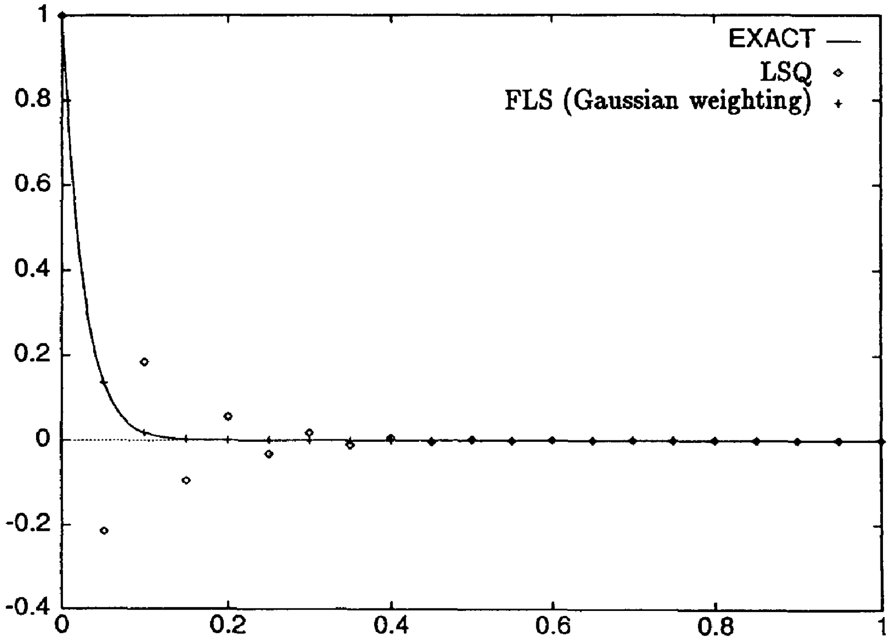

Fig. 3. One-dimensional convection–diffusion problem. Numerical results obtained with the FPM using FLS and LSQ methods with quadratic interpolations ($m = 3$) and five point clouds ($m = 5$). $n_p = 21$.

clouds ($n = 5$). The problem has been solved with a WLS interpolation using fixed Gaussian weighting functions (FLS) and also with a standard LSQ interpolation. Results obtained with the FLS approach reproduce the exact solution whereas the LSQ method leads to oscillations. Further 1D examples of this type using linear FP interpolations can be found in [20,22,23].

The second example is the solution of a steady state *pure diffusive* problem in a square unit domain with $c = k = 1$, a sinusoidal source $Q = \sin(\pi x)$ and the following boundary conditions: $\phi = 0$ for $x = 0$ and $x = 1$ and $\partial\phi/\partial y = 0$ for $y = 0$ and $y = 1$. The problem has been solved using 2D quadratic base polynomials ($m = 6$), clouds with a minimum of 9 points ($n \geqslant 9$) and a FLS interpolation with a Gaussian weighting function. The solution has been obtained with the two different point distributions

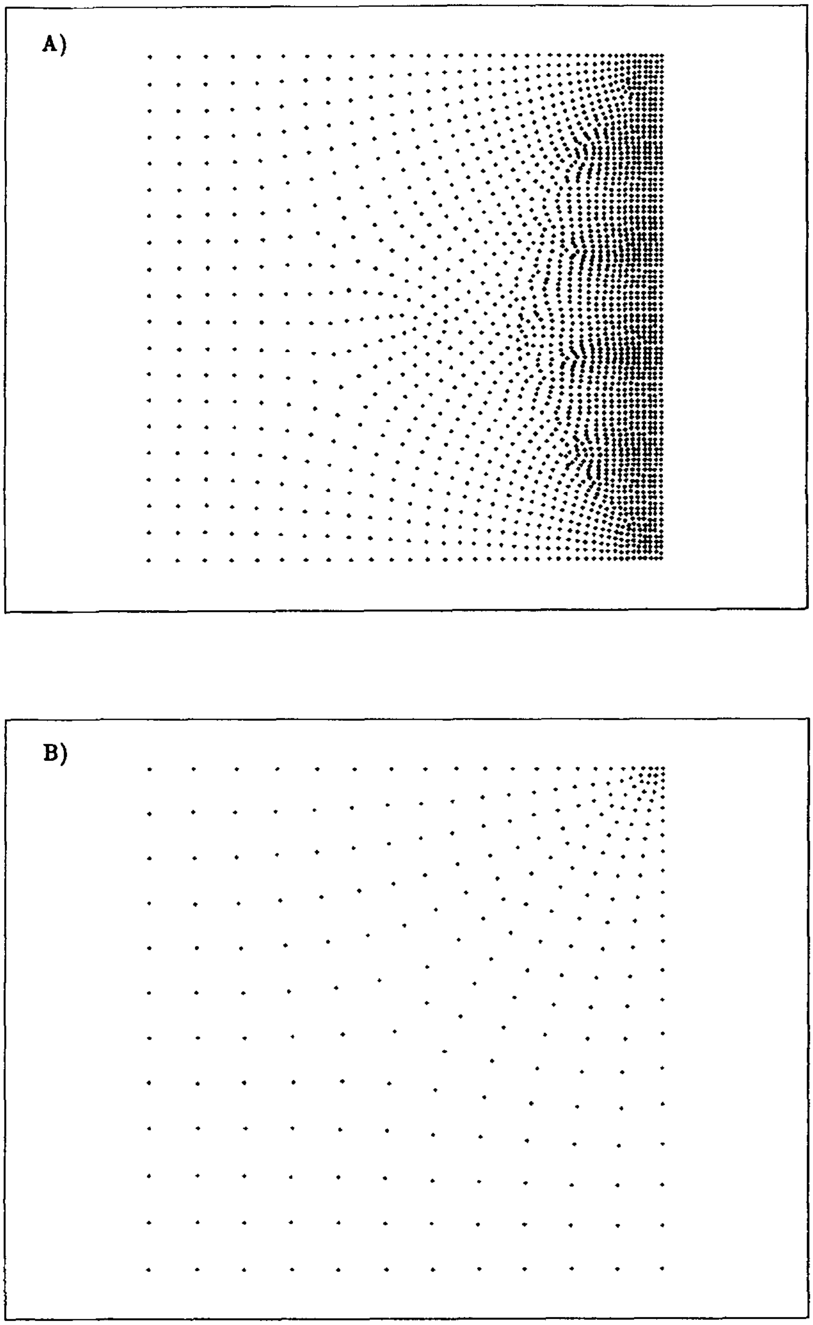

Fig. 4. Point distributions for the solution of two-dimensional steady state convection–diffusion problem in a square domain. (A) $n_p = 1832$; (B) $n_p = 254$.

shown in Fig. 4. The distribution of points in this and the following examples has been generated using a standard advancing front technique [1,26]. The selection of the points within each cloud has been performed using a simple four quadrant selection technique where the closest points to the star node are selected while trying to ensure a uniform distribution of points among the quadrants.

The contours of equal ϕ obtained with the two point distributions are shown in Fig. 5. Note the accuracy of the results in both cases which shows a promising insensitivity of the method to the distribution of points.

The effect of the balancing term $h/2r$ in the Neumann boundary condition is clearly seen in Fig. 6 as erroneous results are obtained when this term is not taken into account.

The next example is the solution of the same problem including convection effects due to an incoming velocity. The same quadratic FP approximation is again used. Fig. 7 shows the results for the case $u = 1$, $v = 0$. Both point distributions give acceptable results although the point distribution B is a wrong one for an horizontal incoming flow. The distribution of ϕ along three horizontal lines is shown in Fig. 8. Note the absence of oscillations and the accuracy of the results computed for both point distributions.

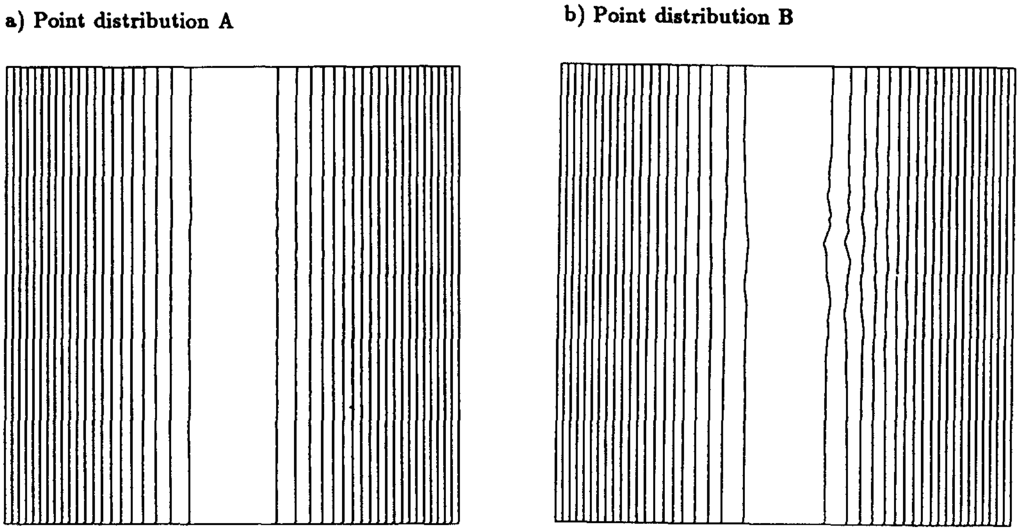

Fig. 5. Two-dimensional diffusion problem with a sine source. Numerical results obtained for the point distributions A and B of Fig. 4 with the stabilized FPM using quadratic FLS interpolation and nine point clouds.

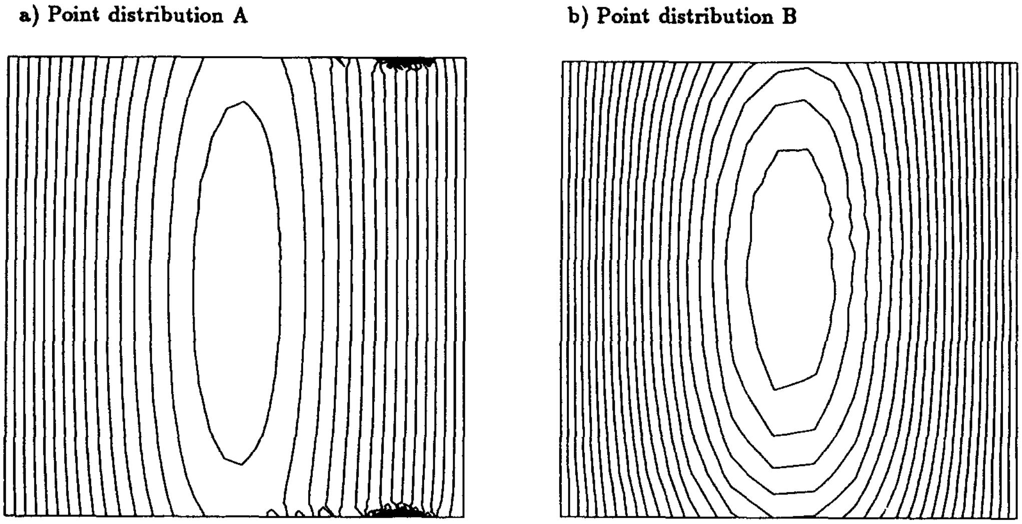

Fig. 6. Two-dimensional diffusion problem with a sine source. Wrong results obtained neglecting the stabilizing term in the Neumann boundary condition.

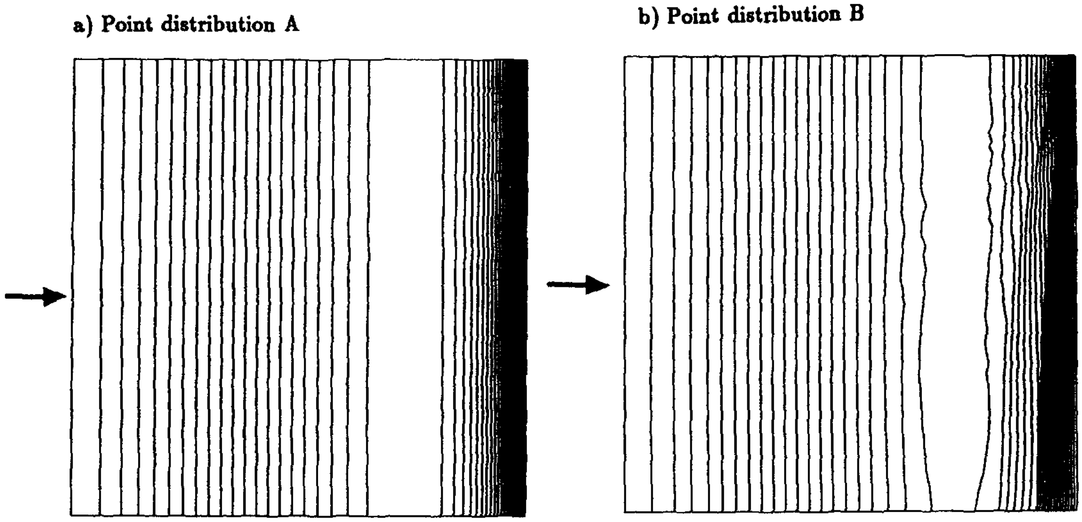

Fig. 7. Convection–diffusion problem with a sine source and horizontal incoming velocity. Numerical results obtained for the point distributions *A* and *B* of Fig. 4 with the stabilized FPM using quadratic FLS interpolation and nine point clouds.

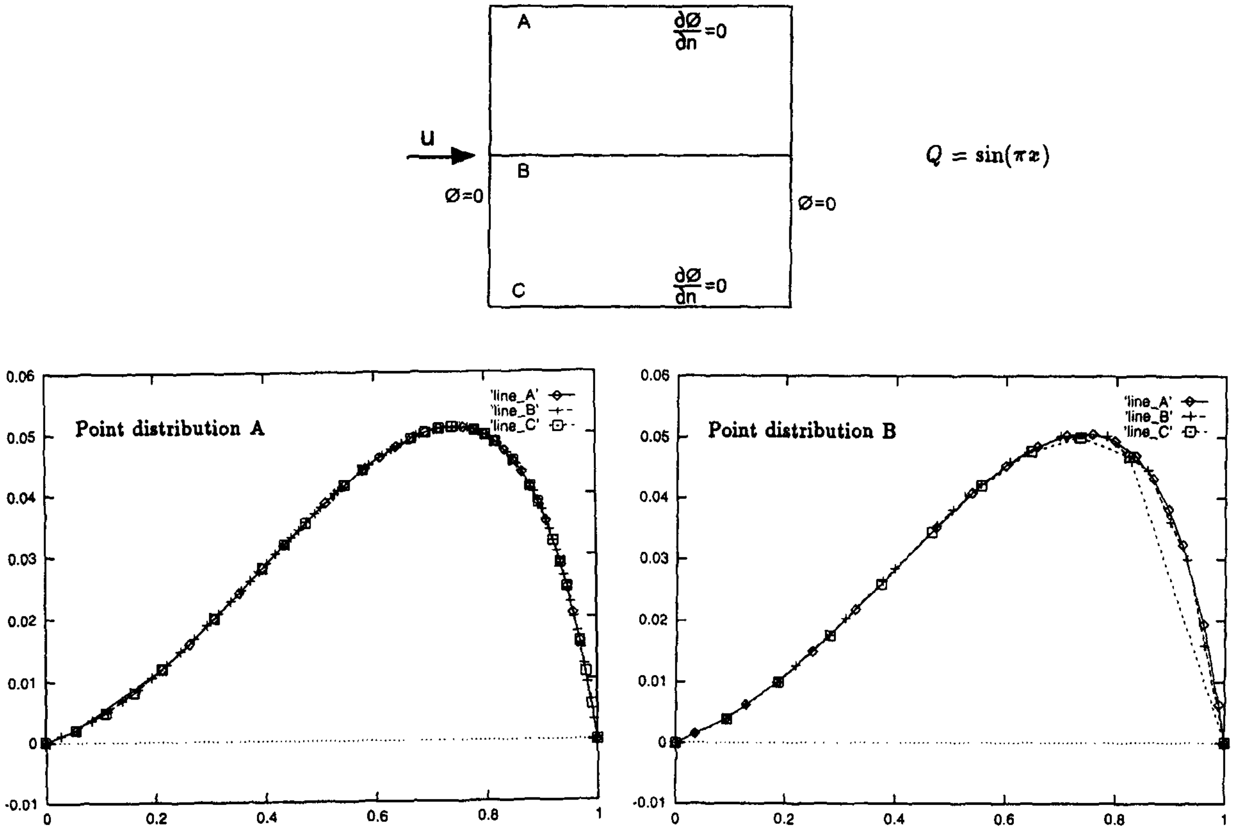

Fig. 8. Two-dimensional convection problem with a sine source ($u = 1$, $v = 0$). Distribution of ϕ along three horizontal lines. Numerical results obtained for the point distributions of Fig. 4 with the stabilized FPM using quadratic FLS interpolation and nine point clouds.

Fig. 9 shows the inaccurate results obtained when the balancing term is not included in the Newmann boundary condition as explained. The contours of ϕ obtained for the case $u = v = 1$ for both point distributions are shown in Fig. 10. Note again the accuracy of the results for the 'wrong' point distribution A, which shows the insensitivity of the method to not very well adapted point distributions.

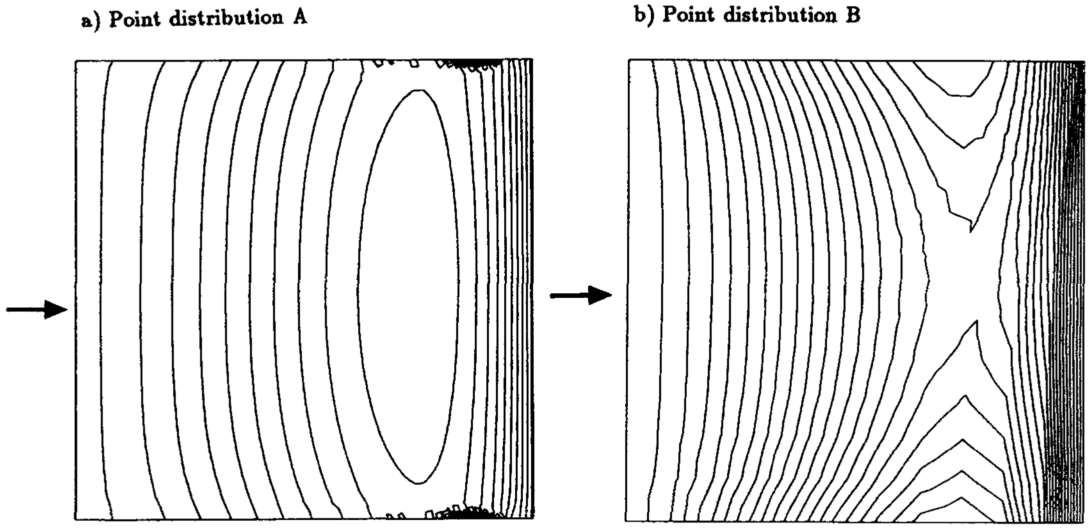

Fig. 9. Convection–diffusion problem with a sine source and horizontal incoming velocity. Wrong results obtained neglecting the stabilizing term in the Neumann boundary condition.

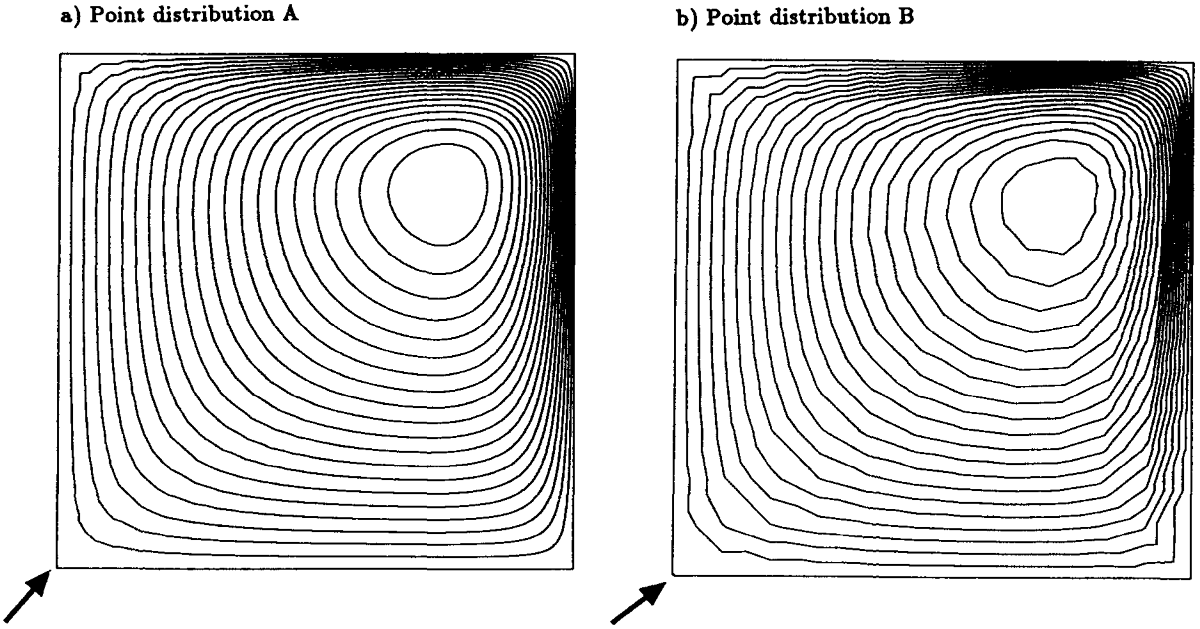

Fig. 10. Convection–diffusion problem with a sine source and diagonal incoming velocity. Numerical results obtained for the point distributions A and B of Fig. 4 with the stabilized FPM using quadratic FLS interpolation and nine point clouds.

Other examples of convection–diffusion problems solved by the authors with the FPM using a linear approximation can be found in [20–22].

4.4. Example of potential flow

The method has been also tested for solving the potential flow equation. Fig. 11 shows the point distribution and the results for the potential function contours for a flow around a cylinder. Here again, a quadratic FLS interpolation and a minimum of nine point clouds have been used.

A pseudo-temporal explicit integration have been used to reach the steady state solution. Results shown in Fig. 11(b) are those obtained using the stabilization technique in the Neumann boundary conditions described in Eq. (68). Without this stabilizing term, no convergence was achieved for any Δt time step increment.

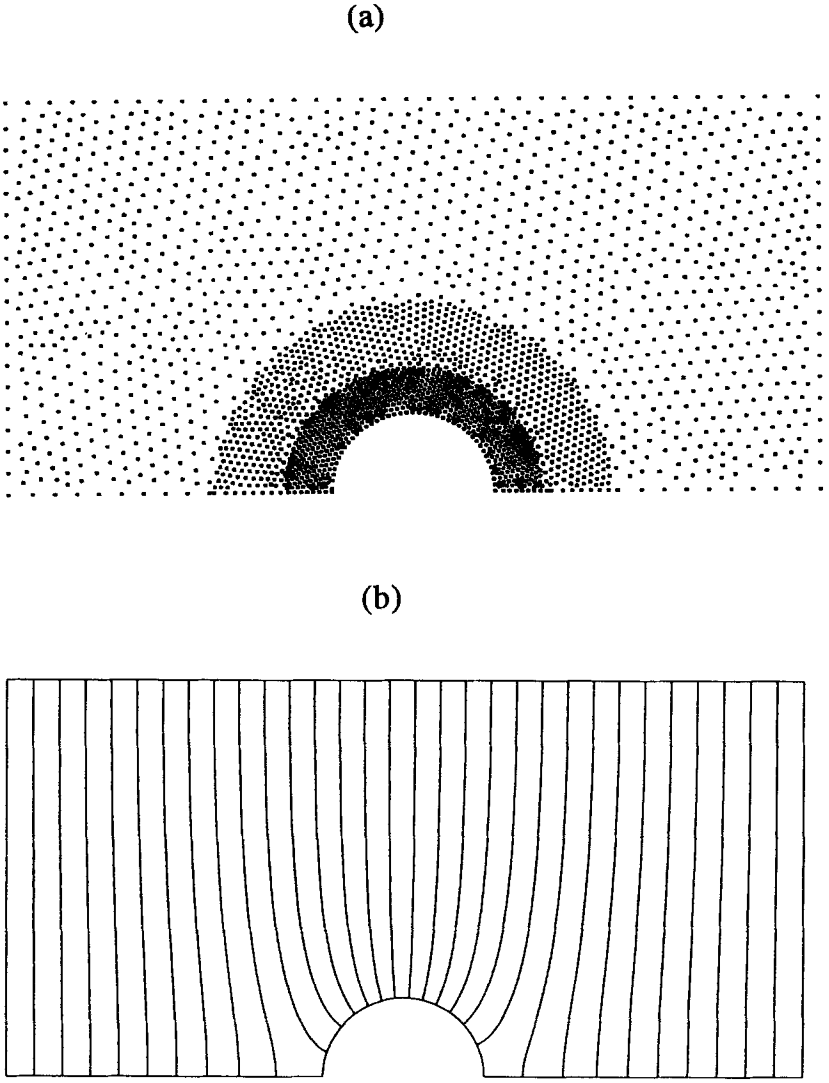

Fig. 11. Potential flow example. Flow around a cylinder. (a) Point distribution. $n_p = 2856$; (b) potential function contours. Solution obtained with the FPM using a quadratic FLS approach ($m = 6$) and clouds with $n \geq 9$.

5. Compressible fluid flow

The approach proposed in the previous section will be generalized now to solve two-dimensional compressible fluid mechanics problems governed by the Navier–Stokes equations written in conservation form as [1]

$$A(\boldsymbol{v}) = \frac{\partial \boldsymbol{v}}{\partial t} + \frac{\partial \boldsymbol{f}_i}{\partial x_i} + \frac{\partial \boldsymbol{g}_i}{\partial x_i} + \boldsymbol{q} = \boldsymbol{0} \quad i = 1, 2 \tag{69}$$

where

$$\boldsymbol{v} = [\rho, \rho u_1, \rho u_2, \rho E]^{\mathrm{T}} \tag{70}$$

$$\boldsymbol{f}_i = [\rho u_i, \rho u_1 u_i + \delta_{1i} p, \rho u_2 u_i + \delta_{2i} p, u_i(\rho E + p)]^{\mathrm{T}} \tag{71}$$

defines the convective flux vector and

$$\boldsymbol{g}_i = \left[0, -\tau_{1i}, -\tau_{2i}, -k\frac{\partial T}{\partial x_i} - \tau_{ij}u_j\right]^{\mathrm{T}} \tag{72}$$

defines the diffusion flows. Finally

$$\boldsymbol{q} = [0, g_1, g_2, 0]^{\mathrm{T}} \tag{73}$$

gives the source terms due to (minus) the gravity acceleration. In the above, ρ is the density, u_k is the velocity in the kth direction, p is the pressure and E is the energy [1]. The stress components τ_{ij} are related to velocity gradients by

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) \tag{74}$$

where μ is the viscosity. The equations are completed by the universal gas law

$$p = pRT \tag{75}$$

where R is the gas constant and T the temperature.

5.1. Residual stabilization technique

Eqs. (69)–(75) have been previously solved using the FPM with a Lax–Wendroff type scheme and linear base interpolations [22,23]. The solution will now be attempted using an extension of the residual stabilization approach presented in the previous section and a *quadratic FLS interpolation*.

Following the same arguments of flow balance over a finite region given in the previous section the following stabilized form of Eq. (69) can be written as [25]

$$A(\boldsymbol{v}) = \frac{\partial \boldsymbol{v}}{\partial t} - \boldsymbol{r} + \frac{h}{2|\boldsymbol{u}|}u_k\frac{\partial \boldsymbol{r}}{\partial x_k} = 0 \tag{76}$$

with

$$\boldsymbol{r} = -\frac{\partial \boldsymbol{f}_i}{\partial x_i} - \frac{\partial \boldsymbol{g}_i}{\partial x_i} - \boldsymbol{q} \tag{77}$$

where h is again the characteristic distance of the flow problem. A generalization of Eq. (76) accounting for different values of h for each equation is proposed in [25]. A simple backward integration scheme leads to

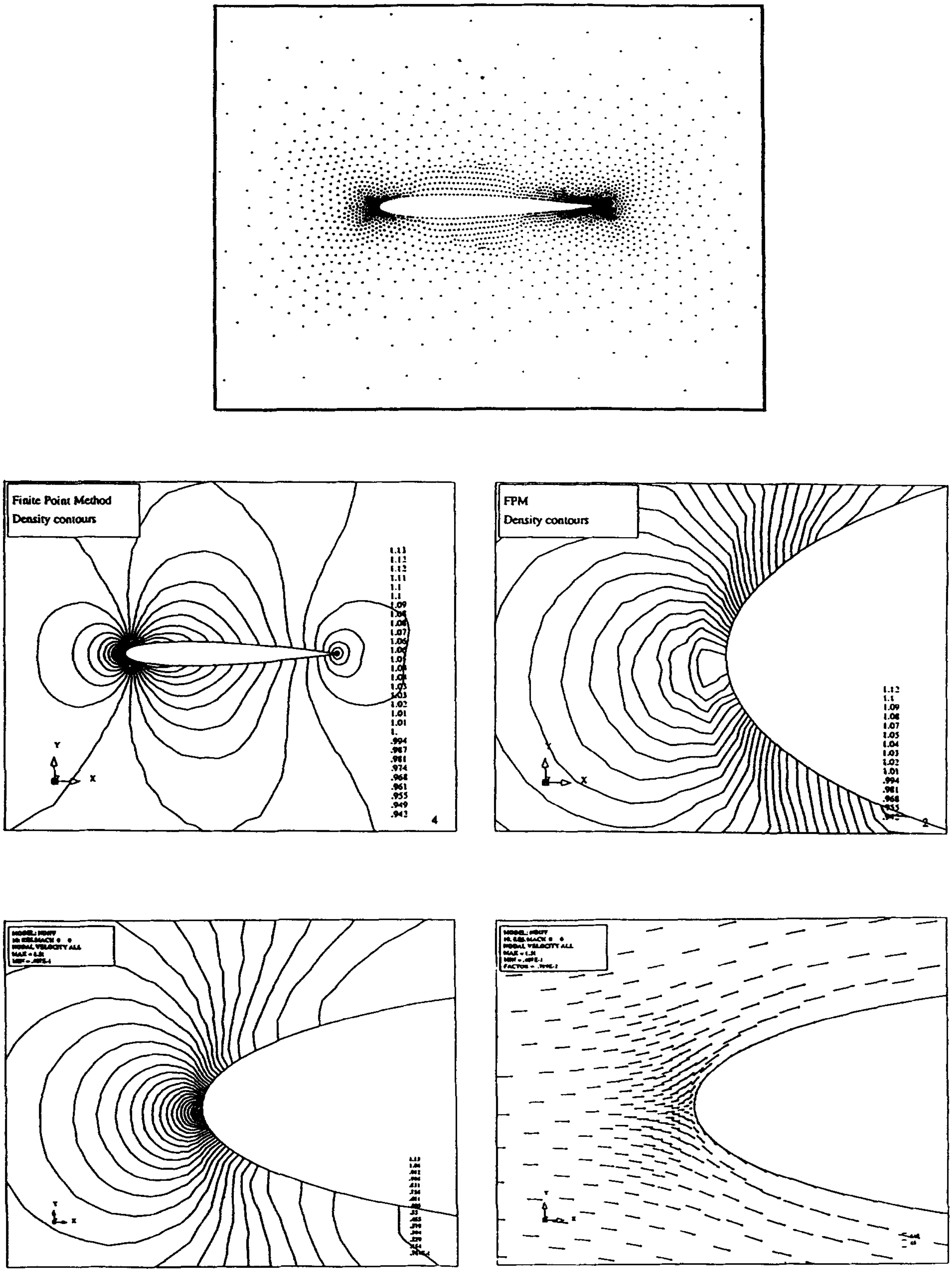

Fig. 12. Inviscid compressible flow around a NACA 0012 profile. Horizontal incoming flow for Mach = 0.5. Point distribution and plots of density and velocity fields. Numerical solution obtained with the FPM using a quadratic FLS approach ($m = 6$) and clouds with $n \geq 9$.

$$\Delta \boldsymbol{v} = \Delta t \left[\boldsymbol{r} - \frac{h}{2|\boldsymbol{u}|} u_i \frac{\partial \boldsymbol{r}}{\partial x_i} \right]^n \tag{78}$$

where $\Delta \boldsymbol{v} = \boldsymbol{v}^{n+1} - \boldsymbol{v}^n$. Eq. (78) can be also derived using a characteristic approximation following the ideas described in [25]. A simpler form of Eq. (78) neglecting the contribution of diffusive flows contribution in the stabilization term has been used by Zienkiewicz et al. [28,29] to derive a fractional step procedure to solve compressible and incompressible flows with the FEM.

Eq. (78) can now be discretized using the FPM by substituting the FP interpolation for the convective flux vector within each cloud as $f_i = \Sigma_{j=1}^n N_j f_i^j$. Choosing a point collocation procedure as described earlier leads to a system of equations from which the value of $\Delta \boldsymbol{v}$ (and subsequently those of f_i and g_i) can be obtained and the solution advanced in time in the usual manner. Indeed, the value of the time step increment Δt in Eq. (78) must be adequately chosen to ensure stability of the time integration scheme [1,28,29]. Note that the treatment of the Neumann boundary condition requires again the use of the stabilization procedure described in the previous section. This has however not been necessary for solution of the problems described next where only the essential boundary condition is imposed.

5.2. *Examples of inviscid compressible fluid flow*

Some examples of compressible inviscid flows are shown next. The inviscid solution (i.e. $g_i = 0$ in Eq. (69)) has been obtained in all cases using again a FPM with a quadratic FLS approach ($m = 6$) with Gaussian weighting functions and clouds with a minimum of nine points ($n \geqslant 9$). The steady state solution was sought using the time integration scheme described above. Fig. 12 shows the distribution of the 2556 points used for the analysis of a NACA 0012 profile for $M_\infty = 0.5$ and zero angle of attack and some results for the density and velocity fields obtained. The distribution of pressure coefficient around the airfoil is shown in Fig. 13, where results obtained with the FEM using a Taylor–Galerkin approach and linear triangles with the same nodal distributions of Fig. 12 is also shown for comparison purposes. Fig. 13 also displays the results obtained with the same FPM and $\varphi_j = 1$ (LSQ approach). Note that the accuracy deteriorates in this case as expected.

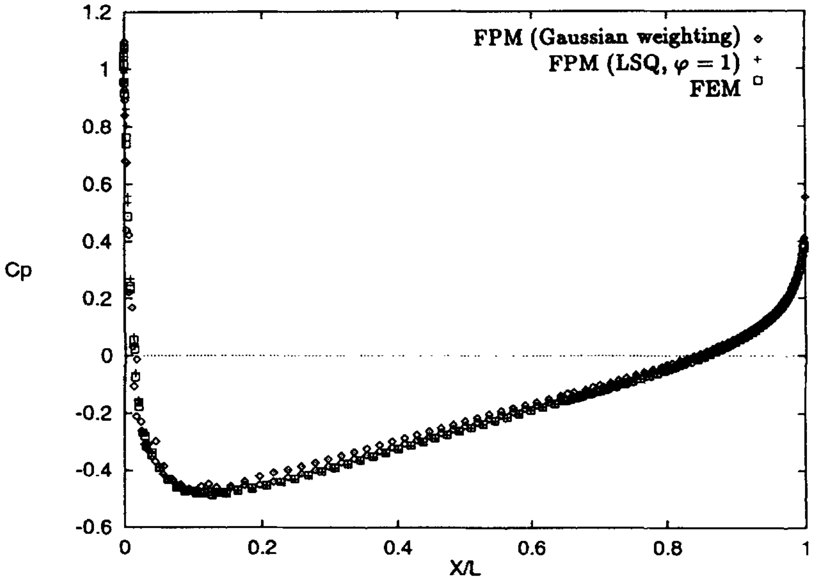

Fig. 13. Inviscid compressible flow around NACA 0012 profile ($M_\infty = 0.5$, $\alpha = 0.0$). Comparison of pressure coefficient around the profile for FPM and FEM solutions.

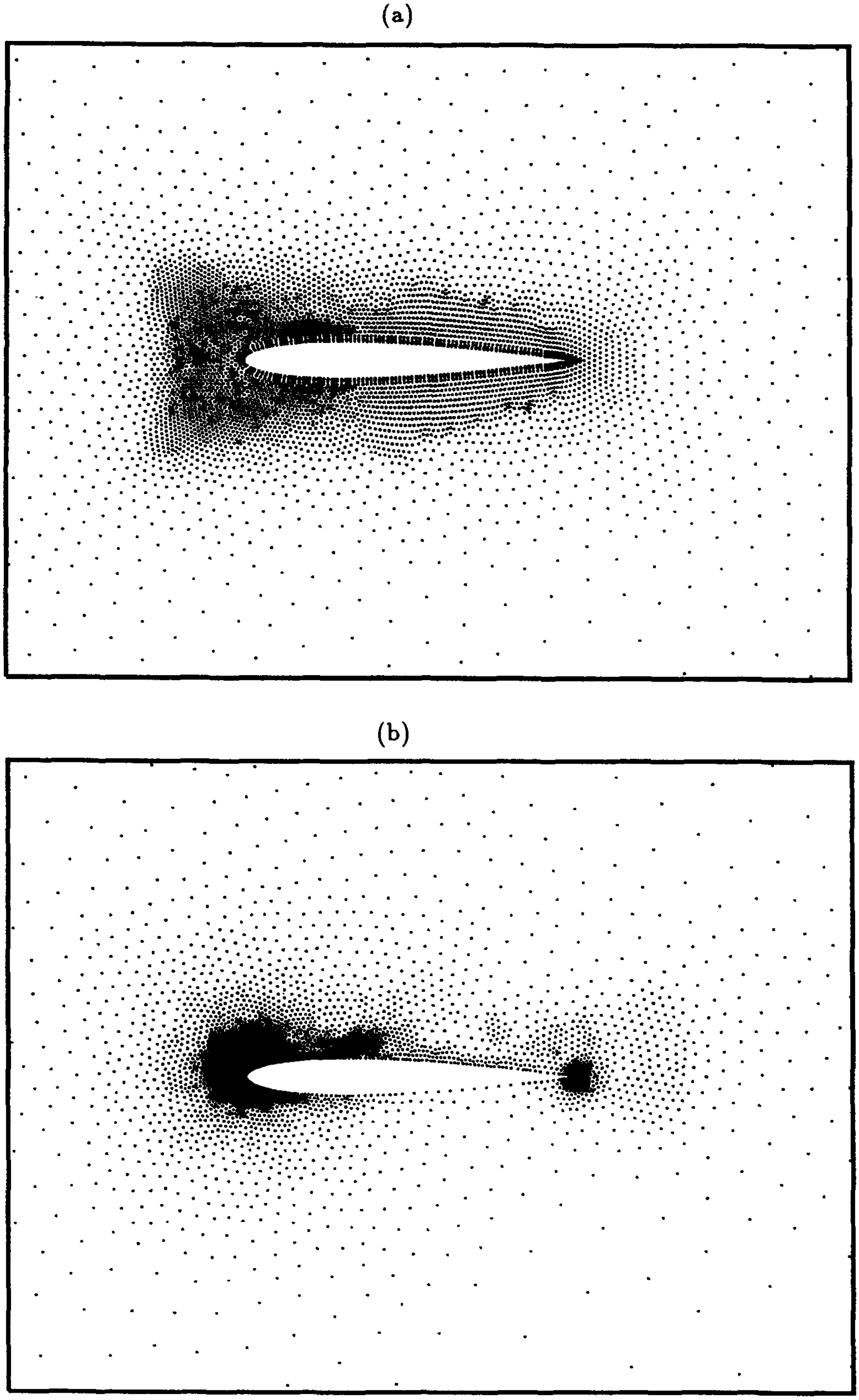

Fig. 14. Inviscid compressible flow around NACA 0012 profile ($M_\infty = 0.3$, $\alpha = 10.0$). Point distributions used in the analysis. (a) 6694 points; (b) 5436 points.

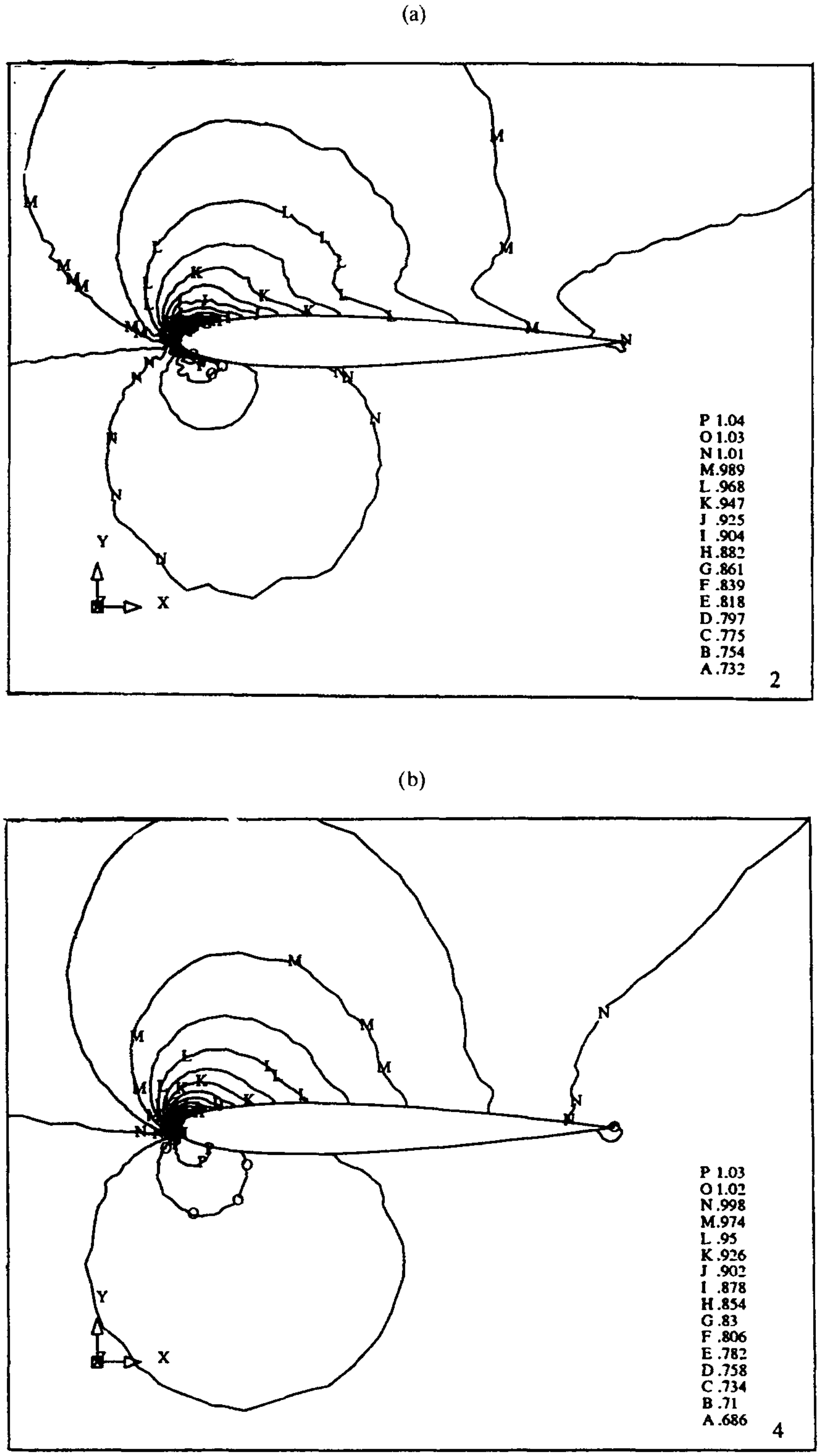

Fig. 15. Inviscid compressible flow around NACA 0012 profile ($M_\infty = 0.5$, $\alpha = 10.0$). Comparison of density contours obtained with the FPM using the two point distributions of Fig. 14. (a) Solution with 6694 points; (b) solution with 5436 points.

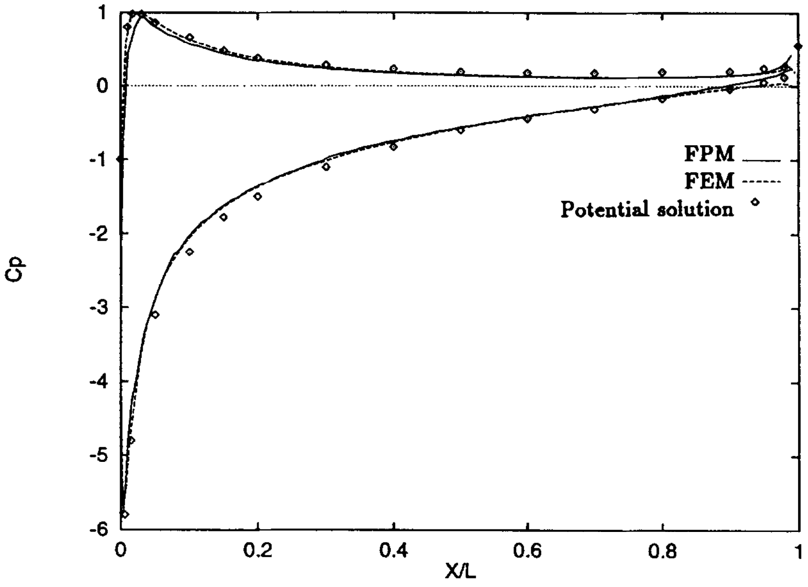

Fig. 16. Inviscid compressible flow around NACA 0012 profile ($M_\infty = 0.5$, $\alpha = 10.0$). Distribution of pressure coefficient around the airfoil.

The second example is a subsonic nearly incompressible flow around the same NACA 0012 profile for $M_\infty = 0.3$ and an angle of attack of 10 degrees. Fig. 14(a) and (b) show the two distributions of 6694 and 5436 points, respectively, used in this analysis. The second point distribution has been obtained by clustering more points in the vicinity of the leading and trailing edges in an adaptive manner. Numerical results for the density contours obtained with the two point distributions are shown in Fig. 15.

Fig. 16 presents the distribution of the pressure coefficient around the airfoil for the adapted point distribution of Fig. 14(b). Results obtained with the FEM using linear triangles and a Taylor–Galerkin formulation and a reference potential solution [30] are also plotted in the figure for comparison purposes.

The next example shows the hypersonic inviscid flow around a double ellipse at $M_\infty = 8.15$ and $\alpha = 30°$. Fig. 17(a) shows the uniform distribution of 14 900 points used for this analysis. The Mach number and density contours obtained are displayed in Fig. 17(b–d), whereas the distribution of pressure coefficient is given in Fig. 18. Obviously, substantial gains in accuracy can be obtained by using point adaptivity and an ad hoc shock capturing technique.

5.3. Example of viscous compressible flow

The last example shows the ability of the FPM for analysis of viscous flows. The problem chosen is the viscous compressible flow around the NACA 0012 profile for Re = 5000, $M_\infty = 0.5$ and $\alpha = 0.0$. Fig. 19 shows the distribution of 14.106 points used in the analysis. Note the higher density of points in the vicinity of the boundary layer. Numerical results have been obtained in this case using a linear FLS approach ($m = 3$) with Gaussian weighting functions and clouds with a minimum of 4 points ($n \geq 4$). Fig. 20 displays the Mach number contours. A detail of the velocity field in the vicinity of the trailing

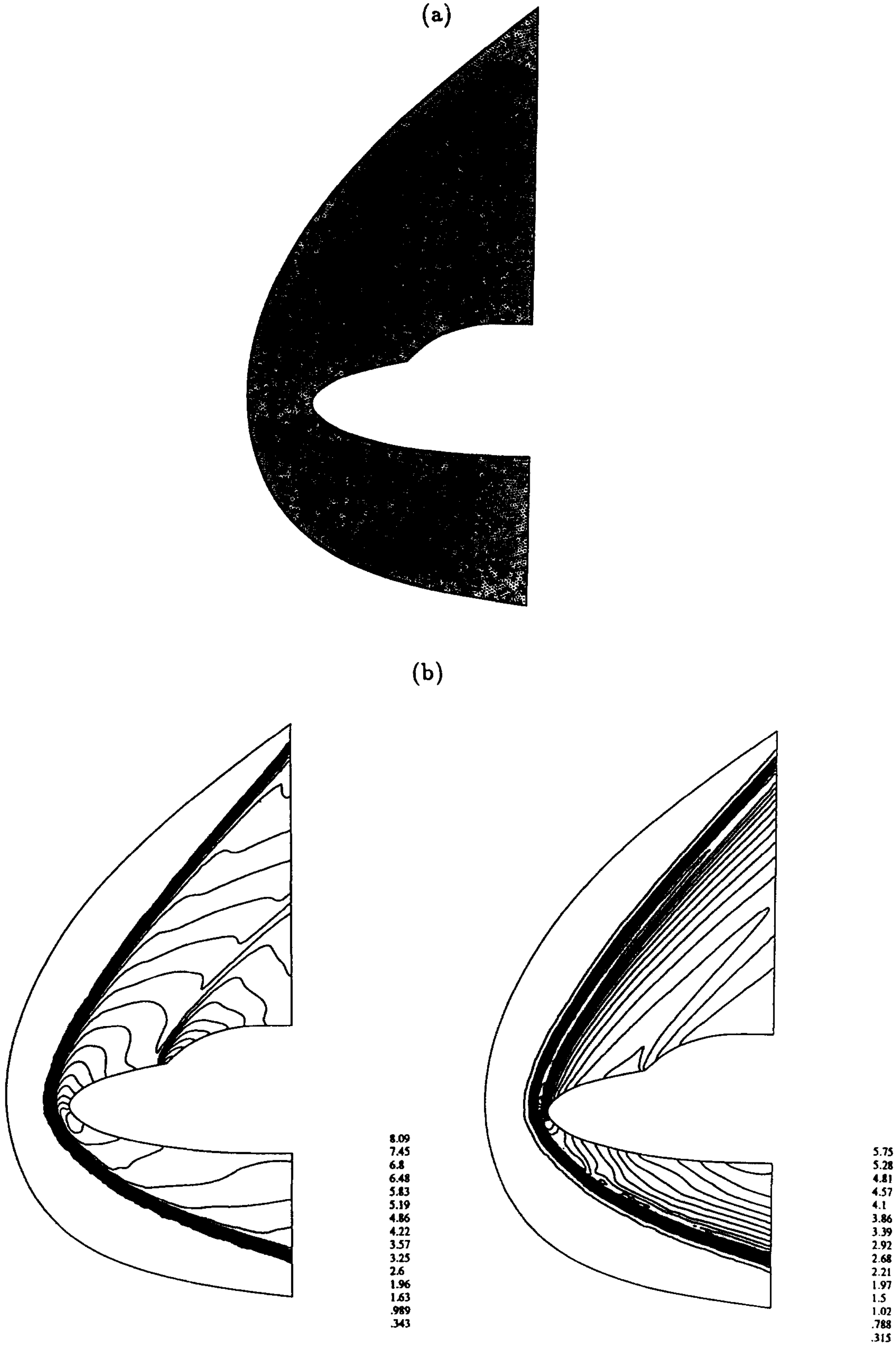

Fig. 17. Hypersonic inviscid flow around a double ellipse ($M_\infty = 8.15$, $\alpha = 30°$). (a) Distribution of 14 900 points used in the analysis; (b) Mach number and density contours; (c) details of density contours in the stagnation zone.

(c)

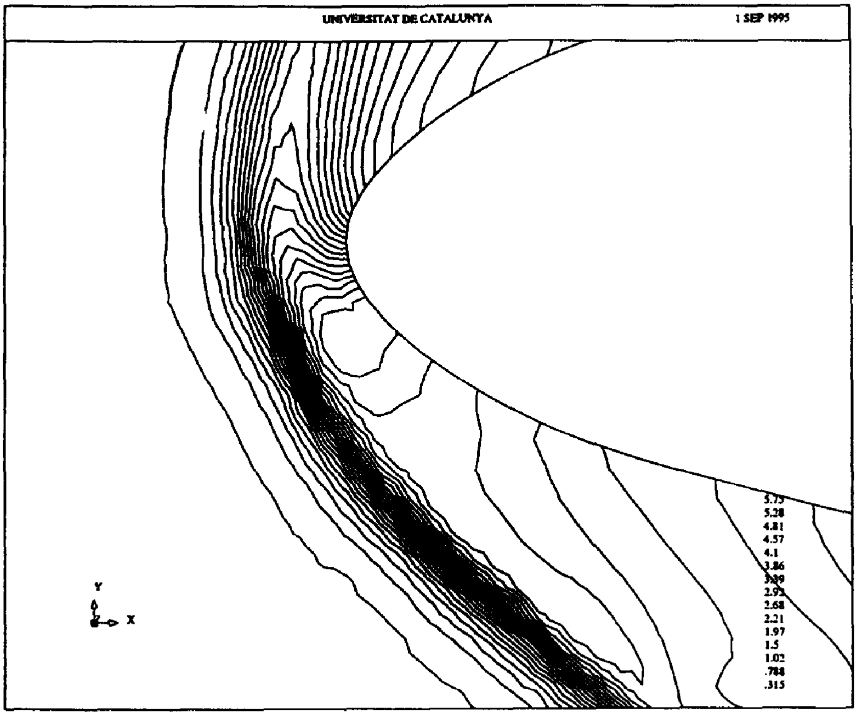

Fig. 17. (Contd.)

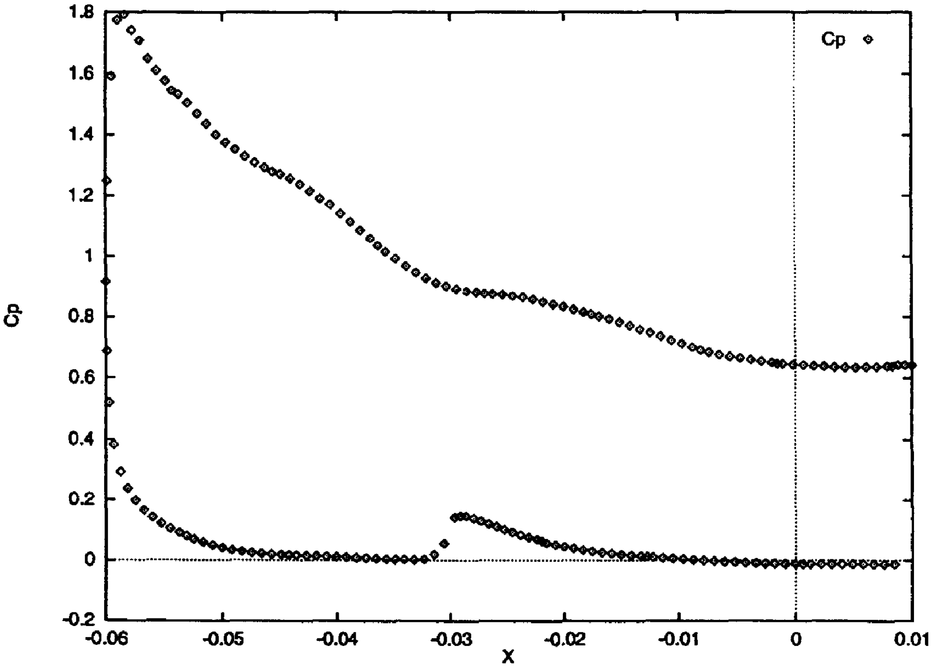

Fig. 18. Hypersonic inviscid flow around a double ellipse ($M_\infty = 8.15$, $\alpha = 30°$). Distribution of pressure coefficient around the surface.

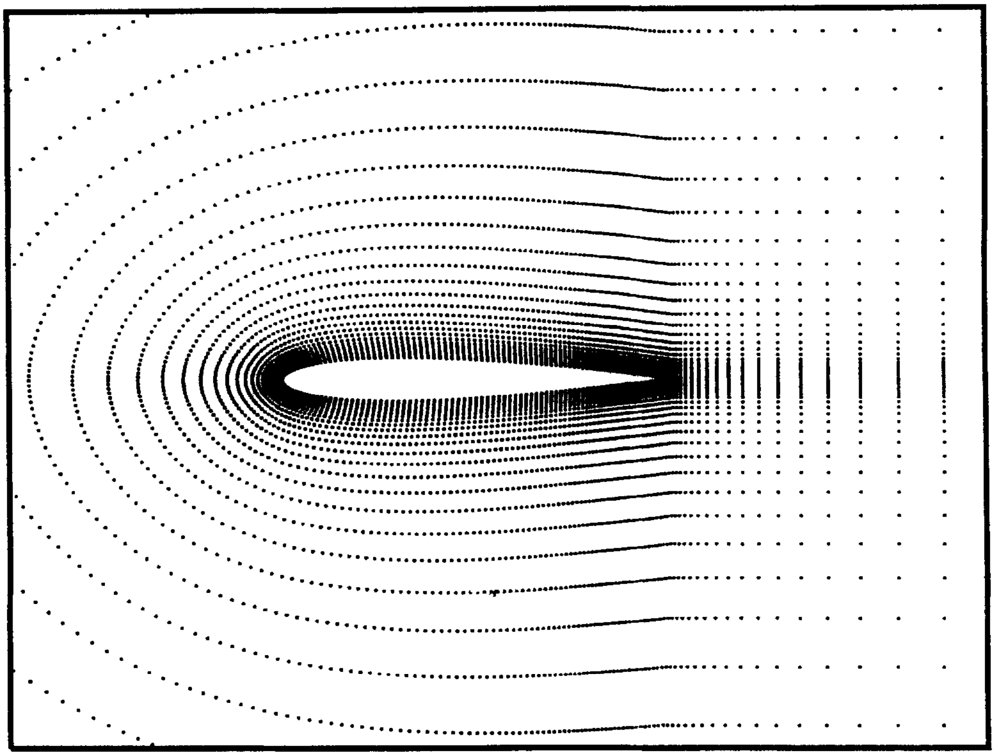

Fig. 19. Distribution of 14 106 points for the solution of viscous flow around a NACA 0012 airfoil (Re = 5000, $M_\infty = 0.5$, $\alpha = 0.0$).

edge is shown in Fig. 20(b). Further examples of this type using linear FP approximation are presented in [20–23].

6. Concluding remarks

The paper shows that the fixed least-square (FLS) interpolation combined with a point collocation technique is a promising Finite Point Method (FPM) for the solution of fluid mechanics problems. A residual stabilization technique which seems very adequate for the FPM has been proposed. It has been shown that in addition to the well-known stabilization requirements for the governing differential equations over the internal domain, the Neumann boundary conditions need also to be stabilized. This has proved to be crucial to obtain acceptable results with the FPM even in cases where convection effects are neglected.

Excellent results have been obtained in all cases with quadratic base interpolations and clouds containing (at least) five and nine points for one- and two-dimensional problems, respectively.

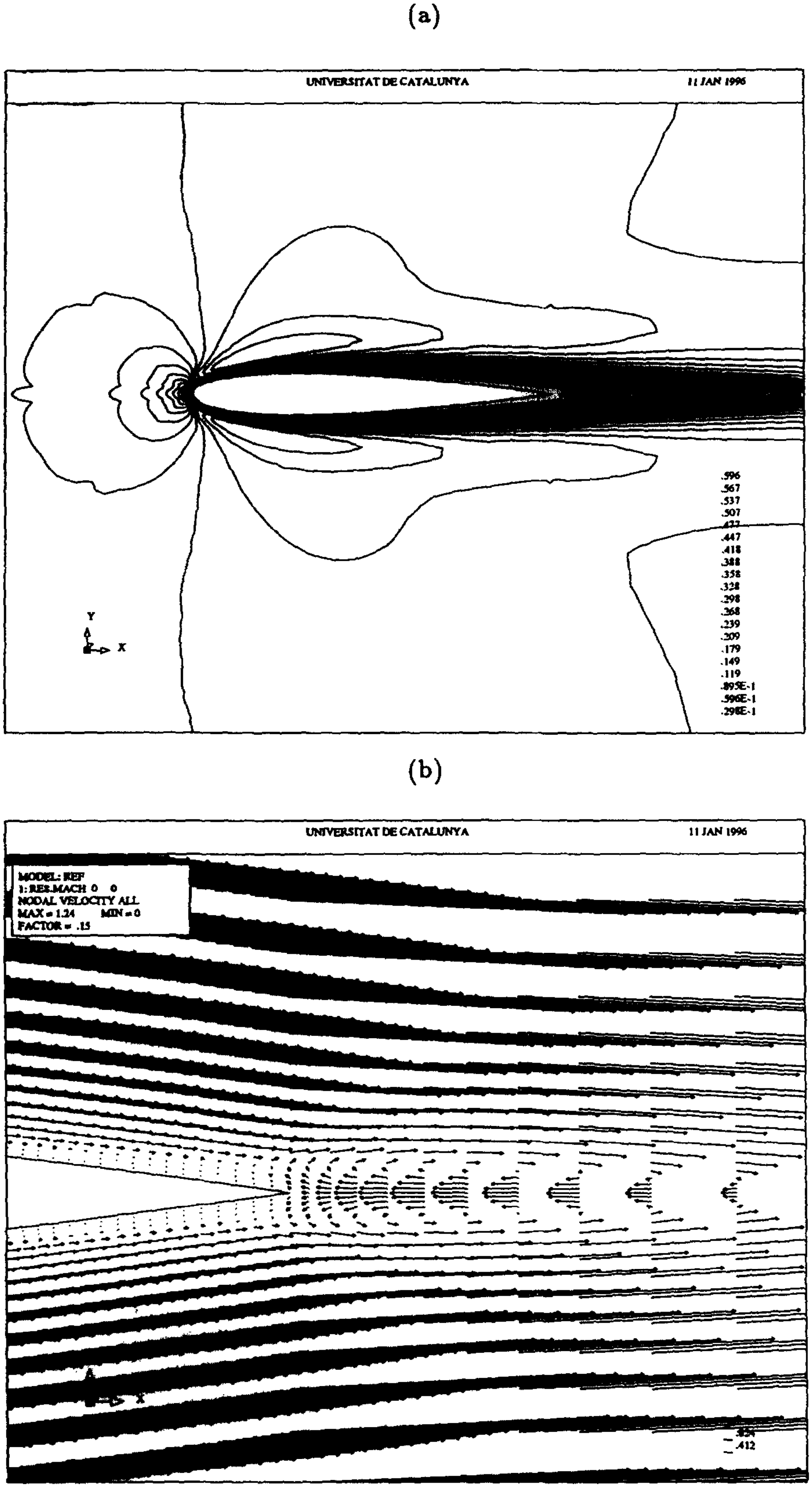

Fig. 20. Viscous flow around a NACA 0012 airfoil (Re = 5000, $M_\infty = 0.5$, $\alpha = 0.0$). (a) Mach number contours; (b) detail of velocity vectors.

Acknowledgment

The authors acknowledge the help received from Dr. T. Fisher in the generation of data and the analysis of the compressible flow examples.

Appendix A

Let us consider the one-dimensional convection–diffusion problem. Fig. A.1 shows a typical segment AB of length h where balance of fluxes must be satisfied. The values of the flow rate q and the advective term $u\phi$ at the end point A with coordinate $x_B - h$ are expressed in terms of those at point B with coordinate x_B by writing the following third-order Taylor's expansion

$$\begin{aligned} q(x_B - h) &= q(x_B) - h\frac{\mathrm{d}q}{\mathrm{d}x}\bigg|_B + \frac{h^2}{2}\frac{\mathrm{d}^2q}{\mathrm{d}x^2}\bigg|_B + \mathrm{O}(h^3) \\ [u\phi](x_B - h) &= [u\phi](x_B) - h\frac{\mathrm{d}[u\phi]}{\mathrm{d}x}\bigg|_B + \frac{h^2}{2}\frac{\mathrm{d}^2[u\phi]}{\mathrm{d}x^2}\bigg|_B + \mathrm{O}(h^3) \end{aligned} \tag{A.1}$$

The balance equation is now written (for the stationary case) as

$$\sum_{\text{fluxes}} = q(x_B) + [u\phi](x_B) - q(x_B - h) - [u\phi](x_B - h) - \frac{1}{2}[Q(x_B) + Q(x_B - h)]h = 0 \tag{A.2}$$

where a linear distribution of the balance external source Q over the segment has been assumed giving

$$Q(x_B - h) = Q(x_B) - h\frac{\mathrm{d}Q}{\mathrm{d}x}\bigg|_B + \mathrm{O}(h^2) \tag{A.3}$$

Substituting Eqs. (A.1) and (A.3) into Eq. (A.2) and noting that the position of point B is arbitrary, i.e. $x_B = x$, gives

$$-\frac{\mathrm{d}(u\phi)}{\mathrm{d}x} - \frac{\mathrm{d}q}{\mathrm{d}x} + Q + \frac{h}{2}\frac{\mathrm{d}}{\mathrm{d}x}\left[-\frac{\mathrm{d}(u\phi)}{\mathrm{d}x} - \frac{\mathrm{d}q}{\mathrm{d}x} + Q\right] = 0 \tag{A.4}$$

The heat flow rate is related to the unknown variable ϕ by Fourier law; i.e. $q = -k\,\mathrm{d}\phi/\mathrm{d}x$, where, for instance, k is the conductivity in a thermal problem. Substituting this into Eq. (A.4) finally gives

$$r - \frac{h}{2}\frac{\mathrm{d}r}{\mathrm{d}x} = 0 \tag{A.5}$$

where

$$r = -\frac{\mathrm{d}(u\phi)}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}\left(k\frac{\mathrm{d}\phi}{\mathrm{d}x}\right) + Q \tag{A.6}$$

Eq. (A.5) can be reinterpreted as the stabilized form of the advective–diffusive equation and h is the *characteristic distance* governing the amount of the balancing term. Obviously for $h = 0$ the standard form of the advective–diffusive equation is found. A particularly interesting case arises when only the

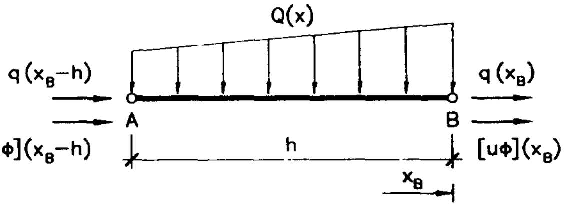

Fig. A.1. Finite domain where balance of fluxes is established.

convective term is included in the balancing term. The stabilized equation (A.5) reads now (for constant u)

$$-u\frac{\mathrm{d}\phi}{\mathrm{d}x}+\frac{\mathrm{d}}{\mathrm{d}x}\left[\left(k+\frac{uh}{2}\right)\frac{\mathrm{d}\phi}{\mathrm{d}x}\right]+Q=0 \tag{A.7}$$

The balancing term contributes now an additional diffusion (usually called 'artificial diffusion') $uh/2$. The form (A.7) has been used as the starting point of many stabilized finite element procedures for numerical solution of convection–diffusion and fluid flow problems [1].

The transient case emerges as a straightforward generalization of Eq. (A.2) to give

$$\sum_{\text{fluxes}}=\frac{\partial\phi}{\partial t} \tag{A.8}$$

Following the same procedure gives

$$\frac{\partial\phi}{\partial t}-r+\frac{h}{2}\frac{\mathrm{d}r}{\mathrm{d}x}=0 \tag{A.9}$$

which is the sought equation.

The same arguments can be applied to derive a stabilized form of 2D convection–diffusion problem as

$$\frac{\partial\phi}{\partial t}-r+\frac{h}{|\boldsymbol{u}|}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{\nabla}r=0 \tag{A.10}$$

with

$$r=-\boldsymbol{\nabla}f+\boldsymbol{\nabla}^{\mathrm{T}}(\boldsymbol{D}\boldsymbol{\nabla}\phi)+Q \tag{A.11}$$

where

$$f=[u\phi,v\phi]^{\mathrm{T}},\qquad \boldsymbol{D}=\begin{bmatrix}k_x & 0\\ 0 & k_y\end{bmatrix}\quad\text{and}\quad \boldsymbol{\nabla}=\left[\frac{\partial}{\partial x},\frac{\partial}{\partial y}\right]^{\mathrm{T}} \tag{A.12}$$

The details of the derivation of (A.10) can be found in [25].

A.1. Stabilization of the Neumann boundary condition

Let us write now the flow balance equation for a finite domain AB of length $h/2$ next to a Neumann boundary point B where the outgoing flux $\bar{q}_n$ is prescribed (Fig. A.2). The choice of the distance $h/2$ is justified by the fact that it allows to recover standard Petrov–Galerkin and SUPG forms in FE procedures as shown in [25].

The balance of fluxes over the domain AB gives

$$\bar{q}_n+[u\phi](x_B)-q\left(x_B-\frac{h}{2}\right)-[u\phi]\left(x_B-\frac{h}{2}\right)-\frac{h}{2}Q=0 \tag{A.13}$$

A simple second-order Taylor expansion (see Eq. (A.1)) leads to

$$\bar{q}_n-q-\frac{h}{2}\left[-\frac{\partial(u\phi)}{\partial x}+\frac{\partial q}{\partial x}+Q\right]=0 \tag{A.14}$$

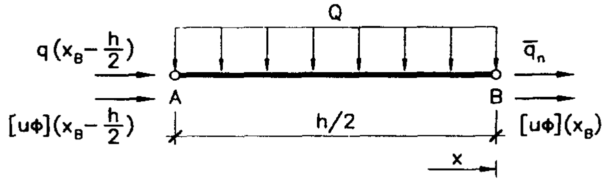

Fig. A.2. Balance domain in the vicinity of a Neumann boundary point B.

Substituting Fourier's law into Eq. (A.14) gives the sought expression

$$k\frac{\mathrm{d}\phi}{\mathrm{d}x}+\bar{q}_n-\frac{h}{2}r=0 \tag{A.15}$$

where the stabilizing term $\bar{r}$ is given by Eq. (A.10).

Similar arguments of flow balance over a two-dimensional finite domain leads to [25].

$$\boldsymbol{n}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\nabla}\phi+\bar{q}_n-\frac{h_n}{2}r=0 \tag{A.16}$$

where $\bar{q}_n$ is the normal flux across the Neumann with normal vector $\boldsymbol{u}$, h_n is a distance along the normal direction, and r is given by Eq. (A.11).

References

[1] O.C. Zienkiewicz and R.L. Taylor, The Finite Element Method (McGraw Hill, Vol. I, 1989, Vol. II, 1991).

[2] O.C. Zienkiewicz and E. Oñate, Finite elements versus finite volumes. Is there a choice?, in: P. Wriggers and W. Wagner, eds., Non Linear Computational Mechanics. State of the Art (Springer-Verlag, 1991).

[3] E. Oñate, M. Cervera and O.C. Zienkiewicz, A finite volume format for structural mechanics, Int. J. Numer. Methods Engrg. 37 (1994) 181–201.

[4] S. Idelsohn and E. Oñate, Finite element and finite volumes. Two good friends, Int. J. Numer. Methods Engrg. 37 (1994) 3323–3341.

[5] G.E. Forsythe and W.R. Wasow, Finite Difference Methods for Partial Differential Equations (Wiley, New York, 1960).

[6] N. Perrone and R. Kao, A general finite difference method for arbitrary meshes, Comput. Struct. 5 (1975) 45–47.

[7] T. Liszka and J. Orkisz, The finite difference method at arbitrary irregular grids and its application in applied mechanics, Comput. Struct. 11 (1980) 83–95.

[8] T. Liszka, An interpolation method for an irregular set of nodes, Int. J. Numer. Methods Engrg. 20 (1984) 1594–1612.

[9] J.J. Moraghan, An introduction to SPH, Comput. Phys. Comm. 48 (1988) 89–96.

[10] B. Nayroles, G. Touzot and P. Villon, Generalizing the FEM: Diffuse approximation and diffuse elements, Comput. Mech. 10 (1992) 307–318.

[11] T. Belytschko, Y. Lu and L. Gu, Element free Galerkin methods, Int. J. Numer. Methods Engrg. 37 (1994) 229–256.

[12] Y.Y. Lu, T. Belytschko and L. Gu, A new implementation of the element free Galerkin method, Comput. Methods Appl. Mech. Engrg. 113 (1994) 397–414.

[13] C.A. Duarte and J.T. Oden, H_p clouds—A meshless method to solve boundary-value problems, TICAM Report 95-05, May 1995.

[14] I. Babuška and J.M. Melenk, The partition of unity finite element method, Technical Note EN-1185, Institute for Physical Science and Technology, Univ. Maryland, April 1995.

[15] R.L. Taylor, O.C. Zienkiewicz, S. Idelsohn and E. Oñate, Moving least square approximations for solution of differential equations, Research Report, 74, International Center for Numerical Methods in Engineering (CIMNE), Barcelona, December 1995.

[16] W.K. Liu, S. Jun and Y.F. Zhang, Reproducing Kernel particle methods, Int. J. Numer. Methods fluids 20 (1995) 1081–1106.

[17] W.K. Liu, S. Jun, S. Li, J. Adee and T. Belytschko, Reproducing Kernel particle methods for structural dynamics, Int. J. Numer. Methods Engrg. 38 (1995) 1655–1679.

[18] W.K. Liu and Y. Chen, Wavelet and multiple scale reproducing Kernel methods, Int. J. Numer. Methods Fluids 21 (1995) 901–933.

[19] W.K. Liu, Y. Chen, S. Jun, J.S. Chen, T. Belytschko, C. Pan, R.A. Uras and C.T. Chang, Overview and applications of the Reproducing Kernel particle methods, Archives Comput. Methods Engrg. 3(1) (1996) 3–80.

[20] E. Oñate, S. Idelsohn and O.C. Zienkiewicz, Finite point methods in computational mechanics, Research Report, 67, CIMNE, Barcelona, July 1995.

[21] E. Oñate, S. Idelsohn, O.C. Zienkiewicz and T. Fisher, 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.

[22] E. Oñate, S. Idelsohn, O.C. Zienkiewicz and R.L. Taylor, A finite point method in computational mechanics. Applications to convective transport and fluid flow, Int. J. Numer. Methods Engrg., to be published.

[23] T. Fisher, S. Idelsohn and E. Oñate, A meshless method for analysis of high speed flows, AGARD Meeting, Seville, October 1995.

[24] J. Batina, A Gridless Euler/Navier–Stokes solution algorithm for complex airfraft applications, AIAA 93-0333, Reno NV, January 11–14, 1993.

[25] E. Oñate, On the stabilization of numerical solution of convective transport and fluid flow problems, Research Report No. 81, International Center for Numerical Methods in Engineering (CIMNE), Barcelona, September 1996.

[26] J. Peraire, J. Peiro, L. Formaggia, K. Morgan and O.C. Zienkiewicz, Finite element Euler computations in three dimensions, Int. J. Numer. Methods Engrg. 26 (1988) 2135–2159.

[27] T.J.R. Hughes and M. Mallet, A new finite element formulation for computational fluid dynamics. III: The generalized streamline operator for multidimensional advective–diffusive systems, Comput. Methods Appl. Mech. Engrg. 58 (1986) 305–328.

[28] O.C. Zienkiewicz and R. Codina, A general algorithm for compressible and incompressible flow. Part I: The split characteristic based scheme, Int. J. Numer. Methods Fluids (1995) 869–885.

[29] O.C. Zienkiewicz, K. Morgan, B.V.K. Satya Sai, R. Codina and M. Vazquez, A general algorithm for compressible and incompressible flow. Part II: Tests on the explicit form, Int. J. Numer. Methods Fluids 20(8–9) (1995) 886–913.

[30] C. Hirsch, Numerical Computations of Internal and External Flow, Vol. 2 (Wiley, 1990).