Comput. Methods Appl. Mech. Engrg. 191 (2002) 2337–2370

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Boundary cloud method: a combined scattered point/boundary integral approach for boundary-only analysis

Gang Li[a], N.R. Aluru[b,*]

[a] Department of Mechanical and Industrial Engineering, 3265 Beckman Institute, MC-251, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana–Champaign, 405 N. Mathews Avenue, Urbana, IL 61801, USA

[b] Department of General Engineering, 3265 Beckman Institute, MC-251, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana–Champaign, 405 N. Mathews Avenue, Urbana, IL 61801, USA

Received 7 February 2001; received in revised form 2 October 2001

Abstract

A boundary cloud method (BCM), for boundary-only analysis of partial differential equations, is presented in this paper for solving potential equations in two dimensions (2-D). The BCM combines a weighted least-squares approach for construction of interpolation functions with a boundary integral formulation for the governing equations. Given a set of scattered points on the surface of an arbitrary object, Hermite-type interpolation functions are constructed by developing a weighted least-squares approach. We also introduce truncated Hermite-type interpolation functions. The boundary integrals are evaluated by using a cell structure. We propose various schemes for evaluating the singular, nearly singular and nonsingular integrals. We also propose a true meshless approach for boundary-only analysis of potential equations. Numerical results, comparing the classical boundary element method and the BCM, are presented for several 2-D potential problems. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Boundary cloud method; Meshless; Boundary integral method; Scattered point interpolation; Hermite interpolation; Boundary-only analysis

1. Introduction

Boundary integral formulations and boundary element methods (BEMs) [1] are attractive computational techniques for linear and exterior problems as they reduce the dimensionality of the original problem. For example, for 3-D problems, the BEM requires discretization of the 2-D surface of the 3-D object and for 2-D problems, the BEM requires discretization of the 1-D boundary. For exterior problems, the use of classical methods, such as finite difference [2] or finite element methods [3], requires discretization of the entire exterior, whereas with a BEM only the surface needs to be discretized. Since only the surface is discretized in BEMs, the mesh generation process is not as intensive compared to interior methods such as finite element methods; however, for complex surfaces, mesh generation can still be a bottleneck in BEMs.

[*] Corresponding author.
E-mail address: aluru@uiuc.edu (N.R. Aluru).
URL: http://www.staff.uiuc.edu/~aluru.

Two key steps in finite element and boundary element methods are (i) the construction of interpolation functions and (ii) discretization and integration. If these two steps can be implemented by using only a scattered set of points instead of using elements (as commonly used in finite element methods) or panels (as commonly used in boundary element methods), then the complex mesh generation process can be alleviated. A popular approach for construction of interpolation functions over a scattered set of points is to employ a moving least-squares technique [4]. Recently, several mesh-free methods have been proposed in the literature (see e.g. [5] for an overview) combining moving least-squares approach with Galerkin [6] as well as collocation techniques [7,8] for interior problems.

A boundary node method [9], for boundary-only analysis, combining the moving least-squares approach with a boundary integral equation has been proposed. Recently, the boundary node method has been extended to 3-D potential equations [10] and elasticity [11]. A key difficulty in the boundary node method is the construction of interpolation functions using moving least-squares methods. For 2-D problems, where the boundary is 1-D, (x, y) coordinates cannot be used to construct interpolation functions (this issue is explained in detail in Section 5). Instead, a cyclic coordinate (s) is used in the moving least-squares approach to construct interpolation functions. For 3-D problems, where the boundary is 2-D, curvilinear coordinates (s_1, s_2) are used to construct interpolation functions. The definition of these coordinates is not trivial for complex geometries. The boundary integrals in the boundary node method are integrated by employing a cell structure. The concept of a cell is quite different from that of a panel in BEMs. Cells are used to sprinkle integration points or Gauss points and the boundary integrals are evaluated using the integration points. In the boundary node method, the integration is performed by using a regular Gauss quadrature except for the cell where the integration is singular.

In this paper we introduce a boundary cloud method (BCM), which combines a scattered point approach for constructing interpolation functions with boundary integral equations for governing partial differential equations. The key ideas in the BCM are:

1. We introduce a least-squares approach to construct Hermite-type interpolation functions. This approach overcomes the difficulty encountered with the classical weighted least-squares approach to construct interpolation functions. Regular Cartesian coordinates ((x, y) for 2-D or (x, y, z) for 3-D) can be employed in the new approach without resorting to cyclic (s) or curvilinear (s_1, s_2) coordinates.
2. A cell structure is employed for numerical integration. Depending on the nature of the boundary integrals (classified as singular, nearly singular and nonsingular), different techniques are employed to evaluate the boundary integrals: (1) direct analytical integration is employed when the boundary integrals are singular, (2) a Nyström scheme [12,13] combined with a singular value decomposition technique [14] is employed to construct the quadrature rules when the boundary integrals are nearly singular and (3) a regular Gauss quadrature is employed when the boundary integrals are nonsingular. The proposed integration scheme achieves high accuracy and improves the convergence of the method.

In many emerging application areas, such as nano- and micro-electromechanical systems (MEMS) [15], exterior electrostatic analysis is a common requirement. Currently, the exterior electrostatic analysis is performed by BEMs [16]. Since MEMS are geometrically very complicated, meshing the surface for exterior electrostatic analysis can be quite involved. A BCM is attractive for electrostatic and other exterior analysis problems as it reduces the dimensionality of the problem and alleviates the meshing requirement. In addition, since we include the normal derivative of the unknown in the weighted least-squares minimization approach, the BCM produces more accurate results in capturing discontinuities in the normal derivative of the solution.

The rest of the paper is outlined as follows: Section 2 summarizes the governing equations, the boundary integral formulations and the classical BEM, Section 3 describes the construction of interpolation functions using a weighted least-squares approach, Section 4 introduces a truncated Hermite-type interpolation approach, Section 5 discusses the properties of the Hermite-type interpolation functions, Section 6 describes the discretization and numerical integration of the boundary integrals, Section 7 summarizes the global matrix assembly and imposition of boundary conditions, Section 8 introduces a true meshless approach for boundary-only analysis of potential problems, numerical results are shown in Section 9 and conclusions are given in Section 10.

2. Governing equations and boundary-integral formulation

We will focus on only 2-D problems in this paper, but the approach can be extended for 3-D problems. The governing equations along with the boundary integral formulations are summarized below.

Consider an arbitrary domain Ω, as shown in Fig. 1, on which the potential equation along with the boundary conditions is to be solved, i.e.

$$\nabla^2 u = 0 \quad \text{in } \Omega, \tag{1}$$

$$u = g \quad \text{on } \Gamma_g, \tag{2}$$

$$q = \frac{\partial u}{\partial n} = h \quad \text{on } \Gamma_h, \tag{3}$$

where u is the unknown potential, Γ_g is the portion of the boundary where Dirichlet boundary conditions are specified, Γ_h is the portion of the boundary where Neumann boundary conditions are specified, g and h are specified Dirichlet and Neumann boundary conditions, respectively, q is the normal derivative of u and n is the outward normal vector.

A boundary integral equation for the potential problem is given by (see e.g. [1] for details)

$$\int_{\mathrm{d}\Omega} \left\{ \ln r(P,Q) q(Q) - \frac{\partial \ln r(P,Q)}{\partial n_Q} [u(Q) - u(P)] \right\} \mathrm{d}S_Q = 0, \tag{4}$$

where $\ln(r)$ is Green's function for the 2-D potential equation, P and Q are source and field points, respectively, r is the distance between P and Q and n_Q is the outward normal vector at a field point Q on boundary $\mathrm{d}\Omega$.

A classical approach to solve Eq. (4) is to use a BEM [1]. In a BEM, the surface is discretized into panels and the potential and its derivative are assumed to be constant on each panel. The centroid of each panel is taken as the collocation point and the value of the potential and its derivative at the collocation point represent the value of the potential and its derivative on the panel. The boundary integral equation for a source point P can be written as

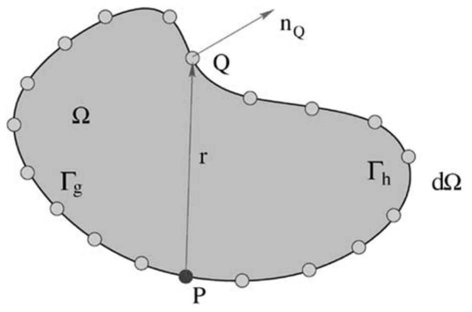

Fig. 1. An arbitrary domain on which the potential equation is to be solved.

$$\sum_{k=1}^{K} \int_{\mathrm{d}S_k} \ln r(P, Q_k) q(Q_k) \, \mathrm{d}S_Q = \sum_{k=1}^{K} \left\{ \int_{\mathrm{d}S_k} \frac{\partial \ln r(P, Q_k)}{\partial n_{Q_k}} [u(Q_k) - u(P)] \, \mathrm{d}S_Q \right\}, \tag{5}$$

where K is the number of panels, $\mathrm{d}S_k$ is the length of kth panel and Q_k is the field point on the kth panel. Eq. (5) can be rewritten in a matrix form as

$$\widehat{\mathbf{H}}\mathbf{u} = \widehat{\mathbf{G}}\mathbf{q}, \tag{6}$$

where $\widehat{\mathbf{H}}$ and $\widehat{\mathbf{G}}$ are $K \times K$ coefficient matrices, and $\mathbf{u}$ and $\mathbf{q}$ are $K \times 1$ potential and its normal derivative vector, respectively. The entries of the coefficient matrices are given by

$$\widehat{G}(i,j) = \int_{\mathrm{d}S_j} \ln r(P_i, Q_j) \, \mathrm{d}S_Q, \tag{7}$$

$$\widehat{H}(i,j) = \int_{\mathrm{d}S_j} \frac{\partial \ln r(P_i, Q_j)}{\partial n_{Q_j}} \mathrm{d}S_Q + \frac{1}{2}\delta_{ij}, \quad i,j = 1, \ldots, K. \tag{8}$$

By substituting boundary conditions into Eq. (6), the unknown potential or its normal derivative can be computed. In Section 9, the convergence of the BEM is compared with the convergence of the BCM.

3. Hermite-type interpolation

In a BCM, the surface of the domain is discretized into scattered points (see Fig. 1). The points can be sprinkled randomly covering the boundary of the domain. Interpolation functions are constructed by centering a weighting function at each point or node. The unknown quantities, u and q, are approximated by a Hermite-type interpolation. For a 2-D problem, as shown in Fig. 2, given a point t, the unknown and its normal derivative in the vicinity of the point t are approximated by

$$u(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\mathbf{a}_t, \tag{9}$$

$$q(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\mathbf{a}_t, \tag{10}$$

where $\mathbf{p}$ is the base interpolating polynomial, $\mathbf{a}_t$ is the unknown coefficient vector for point t and n is the direction of the outward normal to the boundary (note that n can be different at every point). For the interpolation region shown in Fig. 2, $\mathbf{a}_t$ is constant. When the interpolation region changes, $\mathbf{a}_t$ can change.

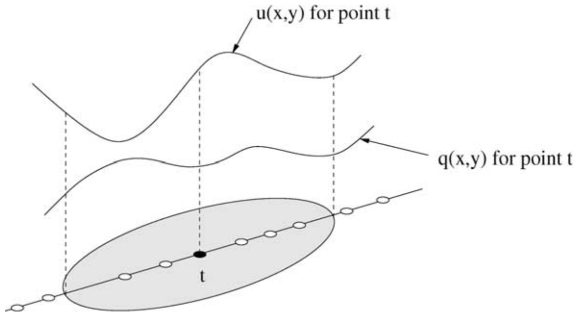

Fig. 2. The interpolation region for point t. The weighting function is centered at point t and vanishes outside the shaded region.

In this paper, we use a linear polynomial basis. The base interpolating polynomial and its normal derivatives are given by

$$\mathbf{p}^{\mathrm{T}}(x,y) = [1 \quad x \quad y], \quad m = 3, \tag{11}$$

$$\frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n} = \left[0 \quad \frac{\partial x}{\partial n} \quad \frac{\partial y}{\partial n}\right] = [0 \quad \cos(\widehat{nx}) \quad \cos(\widehat{ny})], \quad m = 3, \tag{12}$$

where $(\widehat{nx})$ and $(\widehat{ny})$ are the angles between the outward normal direction and the positive x- and y-axis, respectively.

For a point t, the unknown coefficient vector $\mathbf{a}_t$ is computed by minimizing

$$J_t = \sum_{i=1}^{NP} w_i(x_t,y_t)\left[\mathbf{p}^{\mathrm{T}}(x_i,y_i)\mathbf{a}_t - \hat{u}_i\right]^2 + \sum_{i=1}^{NP} w_i(x_t,y_t)\left[\frac{\partial \mathbf{p}^{\mathrm{T}}(x_i,y_i)}{\partial n}\mathbf{a}_t - \hat{q}_i\right]^2, \tag{13}$$

where NP is the number of nodes, and $w_i(x_t,y_t)$ is the weighting function centered at (x_t,y_t) and evaluated at node i whose coordinates are (x_i,y_i). $\hat{u}_i$ and $\hat{q}_i$ are nodal parameters. The weighting function is nonzero when the location of node i is within a certain distance from point (x_t,y_t). Thus, for a point (x_t,y_t), the weighting function is nonzero only for a few other nodes in the vicinity of it. The region where the weighting function is nonzero is called a cloud. The weighting function is typically a cubic spline or a Gaussian function. In this paper, a cubic spline is used whose definition is given by

$$w_i(x_t) = \frac{1}{d_x}\hat{w}\left(\frac{x_t - x_i}{d_x}\right), \tag{14}$$

$$\hat{w}(z) = \begin{cases} 0, & z < -2, \\ (z+2)^3/6, & -2 \leqslant z \leqslant -1, \\ 2/3 - z^2(1+z/2), & -1 \leqslant z \leqslant 0, \\ 2/3 - z^2(1-z/2), & 0 \leqslant z \leqslant 1, \\ -(z-2)^3/6, & 1 \leqslant z \leqslant 2, \\ 0, & z > 2, \end{cases} \tag{15}$$

where $z = (x_t - x_i)/d_x$ and d_x denotes the support size of the weighting function in the x-direction. A multi-dimensional weighting function can be constructed as a product of 1-D weighting functions. In 2-D, the weighting function is given by

$$w_i(x_t,y_t) = \frac{1}{d_x}\hat{w}\left(\frac{x_t - x_i}{d_x}\right)\frac{1}{d_y}\hat{w}\left(\frac{y_t - y_i}{d_y}\right), \tag{16}$$

where d_y is the support size in the y-direction. The stationary of J_t leads to

$$(\mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\mathbf{P}')\mathbf{a}_t = \mathbf{P}^{\mathrm{T}}\mathbf{W}\hat{\mathbf{u}} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\hat{\mathbf{q}}. \tag{17}$$

Eq. (17) can be rewritten as

$$\mathbf{C}_t\mathbf{a}_t = \mathbf{A}_t\hat{\mathbf{u}} + \mathbf{B}_t\hat{\mathbf{q}}, \tag{18}$$

$$\mathbf{a}_t = \mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}, \tag{19}$$

where $\mathbf{C}_t$ is an $m \times m$ matrix, $\mathbf{A}_t$ is an $m \times NP$ matrix and $\mathbf{B}_t$ is an $m \times NP$ matrix, whose definitions are given by

$$\mathbf{C}_t = \mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\mathbf{P}', \tag{20}$$

$$\mathbf{A}_t = \mathbf{P}^{\mathrm{T}}\mathbf{W}, \tag{21}$$

$$\mathbf{B}_t = \mathbf{P}'^{\mathrm{T}}\mathbf{W}, \tag{22}$$

$\mathbf{P}$ and $\mathbf{P}'$ are $NP \times m$ matrices, $\hat{\mathbf{u}}$ and $\hat{\mathbf{q}}$ are $NP \times 1$ vectors, $\mathbf{W}$ is a square $NP \times NP$ diagonal matrix and their definitions are given by

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}^{\mathrm{T}}(x_1, y_1) \\ \mathbf{p}^{\mathrm{T}}(x_2, y_2) \\ \vdots \\ \mathbf{p}^{\mathrm{T}}(x_{NP}, y_{NP}) \end{bmatrix}, \qquad \mathbf{P}' = \begin{bmatrix} \frac{\partial \mathbf{p}^{\mathrm{T}}(x_1, y_1)}{\partial n_1} \\ \frac{\partial \mathbf{p}^{\mathrm{T}}(x_2, y_2)}{\partial n_2} \\ \vdots \\ \frac{\partial \mathbf{p}^{\mathrm{T}}(x_{NP}, y_{NP})}{\partial n_{NP}} \end{bmatrix}, \tag{23}$$

$$\hat{\mathbf{u}} = \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_{NP} \end{bmatrix}, \qquad \hat{\mathbf{q}} = \begin{bmatrix} \hat{q}_1 \\ \hat{q}_2 \\ \vdots \\ \hat{q}_{NP} \end{bmatrix}, \tag{24}$$

$$\mathbf{W} = \begin{bmatrix} w_1(x_t, y_t) & 0 & 0 & 0 \\ 0 & w_2(x_t, y_t) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & w_{NP}(x_t, y_t) \end{bmatrix}, \tag{25}$$

where $n_1, n_2, \ldots, n_{NP}$ are the outward normal directions of nodes $1, 2, \ldots, NP$, respectively. The unknowns, u and q, can be expressed as

$$u(x, y) = \mathbf{p}^{\mathrm{T}}(x, y)\mathbf{a}_t = \mathbf{p}^{\mathrm{T}}(x, y)(\mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}), \tag{26}$$

$$q(x, y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x, y)}{\partial n}\mathbf{a}_t = \frac{\partial \mathbf{p}^{\mathrm{T}}(x, y)}{\partial n}(\mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}). \tag{27}$$

In short form,

$$u(x, y) = \mathbf{M}(x, y)\hat{\mathbf{u}} + \mathbf{N}(x, y)\hat{\mathbf{q}}, \tag{28}$$

$$q(x, y) = \mathbf{S}(x, y)\hat{\mathbf{u}} + \mathbf{T}(x, y)\hat{\mathbf{q}}, \tag{29}$$

where $\mathbf{M}(x, y)$, $\mathbf{N}(x, y)$, $\mathbf{S}(x, y)$ and $\mathbf{T}(x, y)$ are $1 \times NP$ interpolation vectors and they are defined by

$$\mathbf{M}(x, y) = \mathbf{p}^{\mathrm{T}}(x, y)\mathbf{C}_t^{-1}\mathbf{A}_t, \tag{30}$$

$$\mathbf{N}(x, y) = \mathbf{p}^{\mathrm{T}}(x, y)\mathbf{C}_t^{-1}\mathbf{B}_t, \tag{31}$$

$$\mathbf{S}(x, y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x, y)}{\partial n}\mathbf{C}_t^{-1}\mathbf{A}_t, \tag{32}$$

$$\mathbf{T}(x, y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x, y)}{\partial n}\mathbf{C}_t^{-1}\mathbf{B}_t. \tag{33}$$

It can be shown that $\mathbf{M}(x, y)$, $\mathbf{N}(x, y)$, $\mathbf{S}(x, y)$ and $\mathbf{T}(x, y)$ are multivalued, i.e. $\mathbf{M}(x, y)$, $\mathbf{N}(x, y)$, $\mathbf{S}(x, y)$ and $\mathbf{T}(x, y)$ change when the weighting function is centered at different points. To construct a useful approximation, the interpolation function should be limited to a single value. In this paper, we limit the interpolation function to a single value by computing the interpolations at the point where the weighting function is centered, i.e. when the weighting function is centered at (x_t, y_t), the only useful values are $\mathbf{M}_i(x_t, y_t)$, $\mathbf{N}_i(x_t, y_t)$, $\mathbf{S}_i(x_t, y_t)$ and $\mathbf{T}_i(x_t, y_t)$, $i = 1, 2, \ldots, NP$. For the vector of unknowns, $\mathbf{u}$ and $\mathbf{q}$, Eqs. (28) and (29) can be rewritten in a matrix form as

$$\mathbf{u} = \mathbf{M}\hat{\mathbf{u}} + \mathbf{N}\hat{\mathbf{q}}, \tag{34}$$

$$\mathbf{q} = \mathbf{S}\hat{\mathbf{u}} + \mathbf{T}\hat{\mathbf{q}}, \tag{35}$$

where $\mathbf{M}$, $\mathbf{N}$, $\mathbf{S}$ and $\mathbf{T}$ are $NP \times NP$ interpolation matrices. At any point (x, y), the unknowns are evaluated by

$$u(x,y) = \sum_{I=1}^{NP} M_I(x,y)\hat{u}_I + \sum_{I=1}^{NP} N_I(x,y)\hat{q}_I, \tag{36}$$

$$q(x,y) = \sum_{I=1}^{NP} S_I(x,y)\hat{u}_I + \sum_{I=1}^{NP} T_I(x,y)\hat{q}_I, \tag{37}$$

where

$$M_I(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\mathbf{C}_t^{-1}\mathbf{p}(x_I, y_I)w_I(x_t, y_t), \tag{38}$$

$$N_I(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\mathbf{C}_t^{-1}\mathbf{p}'(x_I, y_I)w_I(x_t, y_t), \tag{39}$$

$$S_I(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\mathbf{C}_t^{-1}\mathbf{p}(x_I, y_I)w_I(x_t, y_t), \tag{40}$$

$$T_I(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\mathbf{C}_t^{-1}\mathbf{p}'(x_I, y_I)w_I(x_t, y_t). \tag{41}$$

Remarks

1. If the second term in Eq. (13) is neglected, the weighted least-squares approach is identical to the fixed weighted least-squares approach introduced in [8]. A variant of the second term considered in Eq. (13) has been added in previous works [17,18] to moving least-squares approximation to handle Neumann boundary conditions.
2. In a fixed least-squares approach, Eq. (17) becomes

$$\mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P}\mathbf{a}_t = \mathbf{P}^{\mathrm{T}}\mathbf{W}\hat{\mathbf{u}}. \tag{42}$$

As discussed in Section 5.2, $\mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P}$ becomes singular when all the nodes lie along a straight line.

4. Truncated Hermite-type interpolation

In the Hermite-type interpolation introduced in Eqs. (34) and (35), unknowns $\mathbf{u}$ and $\mathbf{q}$ are approximated in terms of both $\hat{\mathbf{u}}$ and $\hat{\mathbf{q}}$. Four coefficient matrices, $\mathbf{M}$, $\mathbf{N}$, $\mathbf{S}$ and $\mathbf{T}$, need to be computed. As will be shown in the following sections, when the Hermite-type interpolations are used in the boundary integral equation, the final matrix form of the boundary integral equation is complicated and extra computational cost is required to solve the final matrix problem. In this section, we introduce a truncated Hermite-type interpolation which has the same construction as the Hermite-type interpolation except that the definition of the

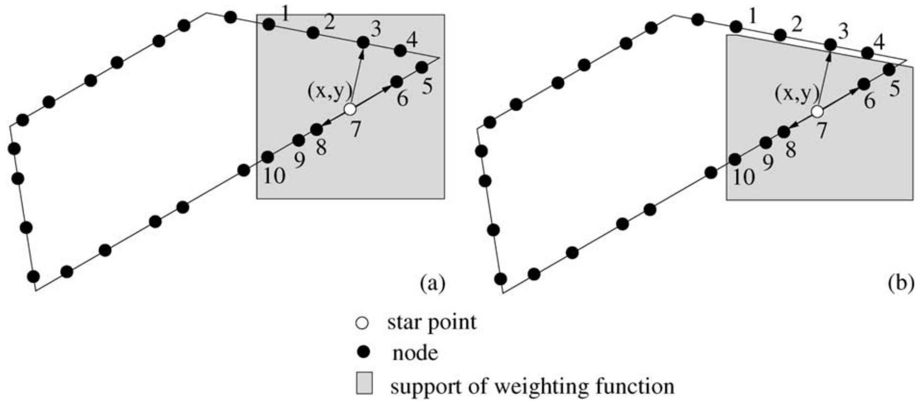

Fig. 3. Definition of weighting function in a (a) regular Hermite-type interpolation; (b) truncated Hermite-type interpolation.

weighting function is different at certain points. The truncated Hermite-type interpolation can significantly reduce the computational cost.

From the definition given in Eq. (16), $w_i(x_t, y_t)$ is a weighting function centered at point (x_t, y_t) (referred to as a star point) and evaluated at node i. $w_i(x_t, y_t)$ is zero when the distance from the star point (x_t, y_t) to node i is larger than the support size of the weighting function. In a regular Hermite-type interpolation, the weighting function extends onto another boundary edge if necessary. For example, as shown in Fig. 3(a), when the weighting function is centered at point 7, the weighting function is nonzero for nodes 1–10 (referred to as a 10-point cloud). Nodes 1–4 lie on one boundary edge while nodes 5–10 lie on a different boundary edge. In a truncated Hermite-type interpolation, nodes lying on a different boundary edge are not included in the cloud. For example, when the weighting function is centered at node 7, the weighting function includes only nodes 5–10 (six-point cloud) and does not include nodes 1–4 as they lie on a different boundary edge.

In a truncated Hermite-type interpolation, Eq. (18) can be written as

$$\overline{\mathbf{C}}_t \overline{\mathbf{a}}_t = \overline{\mathbf{A}}_t \tilde{\mathbf{u}} + \overline{\mathbf{B}}_t \tilde{\mathbf{q}}, \tag{43}$$

where $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{q}}$ are the nodal parameters, $\overline{\mathbf{C}}_t$ is an $m \times m$ matrix, $\overline{\mathbf{A}}_t$ is an $m \times NP$ matrix, and $\overline{\mathbf{B}}_t$ is an $m \times NP$ matrix whose definitions are given by

$$\overline{\mathbf{C}}_t = \mathbf{P}^{\mathrm{T}} \overline{\mathbf{W}} \mathbf{P} + \mathbf{P}'^{\mathrm{T}} \overline{\mathbf{W}} \mathbf{P}', \tag{44}$$

$$\overline{\mathbf{A}}_t = \mathbf{P}^{\mathrm{T}} \overline{\mathbf{W}}, \tag{45}$$

$$\overline{\mathbf{B}}_t = \mathbf{P}'^{\mathrm{T}} \overline{\mathbf{W}}. \tag{46}$$

$\overline{\mathbf{W}}$ is an $NP \times NP$ diagonal matrix

$$\overline{\mathbf{W}} = \begin{bmatrix} \bar{w}_1(x_t, y_t) & 0 & 0 & 0 \\ 0 & \bar{w}_2(x_t, y_t) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \bar{w}_{NP}(x_t, y_t) \end{bmatrix}, \tag{47}$$

$$\bar{w}_i(x_t, y_t) = \begin{cases} w_i(x_t, y_t), & \text{star point } (x_t, y_t) \text{ and node } i \text{ are on the same boundary edge}, \\ 0, & \text{otherwise.} \end{cases} \tag{48}$$

In a truncated Hermite-type interpolation, Eqs. (28) and (29) can be rewritten as

$$u(x,y) = \overline{\mathbf{M}}(x,y)\tilde{\mathbf{u}} + \overline{\mathbf{N}}(x,y)\tilde{\mathbf{q}}, \tag{49}$$

$$q(x,y) = \overline{\mathbf{S}}(x,y)\tilde{\mathbf{u}} + \overline{\mathbf{T}}(x,y)\tilde{\mathbf{q}}, \tag{50}$$

where $\overline{\mathbf{M}}(x,y)$, $\overline{\mathbf{N}}(x,y)$, $\overline{\mathbf{S}}(x,y)$ and $\overline{\mathbf{T}}(x,y)$ are $1 \times NP$ vectors whose definitions are given by

$$\overline{\mathbf{M}}(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\overline{\mathbf{A}}_t, \tag{51}$$

$$\overline{\mathbf{N}}(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\overline{\mathbf{B}}_t, \tag{52}$$

$$\overline{\mathbf{S}}(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\overline{\mathbf{C}}_t^{-1}\overline{\mathbf{A}}_t, \tag{53}$$

$$\overline{\mathbf{T}}(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\overline{\mathbf{C}}_t^{-1}\overline{\mathbf{B}}_t. \tag{54}$$

As described in Section 3, the interpolation functions are only computed at the point where the weighting function is centered so that the interpolation functions are limited to a single value. It is shown in Appendix A that $\overline{\mathbf{N}}(x,y) = \mathbf{0}$ and $\overline{\mathbf{S}}(x,y) = \mathbf{0}$ in a truncated Hermite-type interpolation. As a result, Eqs. (49) and (50) can be rewritten as

$$\mathbf{u} = \overline{\mathbf{M}}\tilde{\mathbf{u}}, \tag{55}$$

$$\mathbf{q} = \overline{\mathbf{T}}\tilde{\mathbf{q}}. \tag{56}$$

At any point (x,y), the unknowns are evaluated by

$$u(x,y) = \sum_{I=1}^{NP} \overline{M}_I(x,y)\tilde{u}_I, \tag{57}$$

$$q(x,y) = \sum_{I=1}^{NP} \overline{T}_I(x,y)\tilde{q}_I, \tag{58}$$

where

$$\overline{M}_I(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\mathbf{p}(x_I,y_I)\bar{w}_I(x_t,y_t), \tag{59}$$

$$\overline{T}_I(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\overline{\mathbf{C}}_t^{-1}\mathbf{p}'(x_I,y_I)\bar{w}_I(x_t,y_t). \tag{60}$$

5. Properties of Hermite-type and truncated Hermite-type interpolations

The Hermite-type interpolation functions presented in the previous section have several interesting properties. First, unlike the construction of the interpolation functions in the classical BEM, the approach presented here does not need to know the connectivity information among the points or nodes. Points can be sprinkled randomly covering the boundary of the domain. This property greatly alleviates the task of mesh generation, which can be difficult and time consuming for complex geometries. Second, the classical weighted least-squares approach has difficulty computing interpolation functions when points are placed

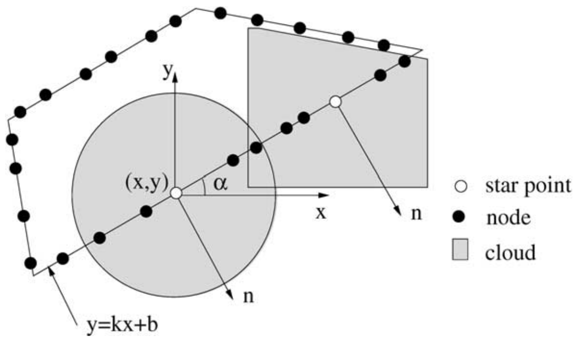

Fig. 4. Illustration of boundary clouds. A circular and a truncated rectangular cloud are shown.

only along the boundary. For example, when all the points lie along a straight line, the moment matrix in a weighted least-squares approach becomes singular. Hermite-type interpolation functions overcome this disadvantage by including the normal derivatives of the base interpolating polynomial for the boundary nodes. Third, Hermite-type interpolation functions satisfy the consistency conditions, i.e. they reproduces any linear combination of the base interpolating polynomial exactly. Each of these three aspects is discussed below in detail.

5.1. Meshless property of interpolation functions

The Hermite-type interpolation functions can be constructed by simply sprinkling points without knowing the connectivity information among the nodes or by creating elements which connect various nodes. As shown in Fig. 4, the boundary of the domain is first discretized into points. Interpolation functions are constructed by centering a weighting function at each point. When a weighting function is centered at a point (called star point), there are a few other points in the vicinity of the star point for which the weighting function does not vanish. The vicinity of the star point for which the weighting function does not vanish is referred to as a cloud. The points outside the cloud do not have any influence on the interpolation function for the star point. The parameters in the weighting function (for example d_x and d_y in Eq. (16)) determine the cloud size. The weighting functions used in defining clouds provide versatility in the construction of interpolation functions. For a uniform point distribution, we can use a circular cloud with the value of the weighting function determined only by Euclidean distance from the star node. For a uniform point distribution with unequal spacing along each axis (for example for 3-D problems where the boundary is 2-D), a rectangular-shaped cloud can be used with the value of the 2-D weighting function determined by the product of two 1-D weighting functions aligned with the coordinate axis. For random point distributions, the size and shape of each cloud can be constructed independent of each other.

5.2. Boundary-only property of Hermite-type interpolation

For boundary-only problems, where points are distributed only along the boundary of the domain, neither the classical 2-D kernel approximations [7] nor the weighted least-squares approximations [4,8] can be used due to the singularity of the moment matrix. Since only the boundary is considered in boundary integral formulations, for 2-D problems the boundary is 1-D, and when all the points in a cloud lie along a straight line, the moment matrix in weighted least-squares and kernel approximations loses its rank. However, the Hermite-type weighted least-squares interpolation restores the full rank of the moment matrix and can be used to compute interpolation functions for any distribution of points without any difficulty.

To illustrate the problem with the classical weighted least-squares approximation, consider a boundary segment as shown in Fig. 4. Without losing generality, we will assume that the equation of the boundary segment is of the form $y = kx + b$. The moment matrix with a classical weighted least-squares approximation is given by

$$\mathbf{C}_t = \mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} = \begin{bmatrix} \sum_{j=1}^{NP} w_j(x_t, y_t) & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)y_j \\ \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j^2 & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j y_j \\ \sum_{j=1}^{NP} w_j(x_t, y_t)y_j & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j y_j & \sum_{j=1}^{NP} w_j(x_t, y_t)y_j^2 \end{bmatrix}. \tag{61}$$

As shown in Fig. 4, if all the points in the cloud are on the same boundary segment, we have $y_j = kx_j + b$ for $(x_j, y_j) \in$ cloud. In this case, the moment matrix becomes

$$\mathbf{C}_t = \begin{bmatrix} \sum_{j=1}^{NP} w_j(x_t, y_t) & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & k\left\{\sum_{j=1}^{NP} w_j(x_t, y_t)x_j\right\} + b\left\{\sum_{j=1}^{NP} w_j(x_t, y_t)\right\} \\ \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j^2 & k\left\{\sum_{j=1}^{NP} w_j(x_t, y_t)x_j^2\right\} + b\left\{\sum_{j=1}^{NP} w_j(x_t, y_t)x_j\right\} \\ \sum_{j=1}^{NP} w_j(x_t, y_t)y_j & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j y_j & k\left\{\sum_{j=1}^{NP} w_j(x_t, y_t)x_j y_j\right\} + b\left\{\sum_{j=1}^{NP} w_j(x_t, y_t)y_j\right\} \end{bmatrix}. \tag{62}$$

It is obvious that the third column of the moment matrix is the linear combination of the first two columns; thus the rank of the above matrix is 2. If $\mathbf{C}_t$ is singular, $\mathbf{a}_t$ cannot be computed uniquely (see Eq. (42)). Nevertheless, in Hermite-type weighted least-squares interpolation, the moment matrix for a star point t is given by

$$\begin{aligned}\mathbf{C}_t = \mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\mathbf{P}' &= \begin{bmatrix} \sum_{j=1}^{NP} w_j(x_t, y_t) & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)y_j \\ \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j^2 & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j y_j \\ \sum_{j=1}^{NP} w_j(x_t, y_t)y_j & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j y_j & \sum_{j=1}^{NP} w_j(x_t, y_t)y_j^2 \end{bmatrix} \\ &+ \begin{bmatrix} 0 & 0 & 0 \\ 0 & \sum_{j=1}^{NP} w_j(x_t, y_t)\cos^2(\widehat{nx}) & \sum_{j=1}^{NP} w_j(x_t, y_t)\cos(\widehat{nx})\cos(\widehat{ny}) \\ 0 & \sum_{j=1}^{NP} w_j(x_t, y_t)\cos(\widehat{nx})\cos(\widehat{ny}) & \sum_{j=1}^{NP} w_j(x_t, y_t)\cos^2(\widehat{ny}) \end{bmatrix}.\end{aligned} \tag{63}$$

For the boundary segment under consideration,

$$\frac{\cos(\widehat{nx})}{\cos(\widehat{ny})} = -k. \tag{64}$$

Thus the moment matrix is given by

$$\begin{aligned}\mathbf{C}_t &= \mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\mathbf{P}' \\ &= \begin{bmatrix} \sum_{j=1}^{NP} w_j(x_t, y_t) & \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)(kx_j + b) \\ \sum_{j=1}^{NP} w_j(x_t, y_t)x_j & \sum_{j=1}^{NP} w_j(x_t, y_t)\{x_j^2 + k^2\cos^2(\widehat{ny})\} & \sum_{j=1}^{NP} w_j(x_t, y_t)\{kx_j^2 + bx_j - k\cos^2(\widehat{ny})\} \\ \sum_{j=1}^{NP} w_j(x_t, y_t)(kx_j + b) & \sum_{j=1}^{NP} w_j(x_t, y_t)\{kx_j^2 + bx_j - k\cos^2(\widehat{ny})\} & \sum_{j=1}^{NP} w_j(x_t, y_t)\{(kx_j + b)^2 + \cos^2(\widehat{ny})\} \end{bmatrix}.\end{aligned} \tag{65}$$

From Eq. (65), it is clear that the rank deficiency in classical weighted least-squares is compensated by the normal derivatives of the basis functions. The moment matrix in the new weighted least-squares approximation has full rank. For special cases, such as vertical or horizontal segments, where $k = \infty$ or $k = 0$, respectively, it can be shown that the new weighted least-squares approximation provides full rank for the moment matrix.

By following the steps outlined above, it can be shown that the truncated Hermite-type interpolation functions can also be computed using Cartesian coordinates, i.e. the moment matrix $\overline{\mathbf{C}_t}$ of the truncated Hermite-type interpolation has full rank.

5.3. Consistency of Hermite-type boundary interpolation

In this section, we show that the interpolation functions can approximate at least all the functions included in the definition of $\mathbf{p}(x,y)$. To show this, consider the approximation given in Eqs. (26) and (27)

$$u(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)(\mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}), \tag{66}$$

$$q(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}(\mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}), \tag{67}$$

where

$$\hat{\mathbf{u}} = [\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_{NP}]^{\mathrm{T}}, \tag{68}$$

$$\hat{\mathbf{q}} = [\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_{NP}]^{\mathrm{T}}, \tag{69}$$

$\hat{u}_i$ and $\hat{q}_i$ are nodal parameters corresponding to node i.

Assigning to each $\hat{u}_i$ and $\hat{q}_i$ the values of the basis functions vector gives

$$\hat{\mathbf{u}}_i = \mathbf{p}^{\mathrm{T}}(x_i,y_i), \tag{70}$$

$$\hat{\mathbf{q}}_i = \mathbf{p}'^{\mathrm{T}}(x_i,y_i), \tag{71}$$

i.e.

$$\hat{\mathbf{u}} = \mathbf{P}, \tag{72}$$

$$\hat{\mathbf{q}} = \mathbf{P}'. \tag{73}$$

Substituting Eqs. (72) and (73) into Eqs. (66) and (67), respectively, gives

$$\begin{aligned}\mathbf{u}(x,y) &= \mathbf{p}^{\mathrm{T}}(x,y)(\mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}) = \mathbf{p}^{\mathrm{T}}(x,y)(\mathbf{C}_t^{-1}\mathbf{A}_t\mathbf{P} + \mathbf{C}^{-1}\mathbf{B}_t\mathbf{P}') \\ &= \mathbf{p}^{\mathrm{T}}(x,y)\mathbf{C}_t^{-1}(\mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\mathbf{P}') = \mathbf{p}^{\mathrm{T}}(x,y)\mathbf{C}_t^{-1}\mathbf{C}_t = \mathbf{p}^{\mathrm{T}}(x,y),\end{aligned} \tag{74}$$

which shows that the approximation in Eq. (66) can interpolate any function exactly as part of the definition of $\mathbf{p}(x,y)$. Similarly,

$$\begin{aligned}\mathbf{q}(x,y) &= \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}(\mathbf{C}_t^{-1}\mathbf{A}_t\hat{\mathbf{u}} + \mathbf{C}_t^{-1}\mathbf{B}_t\hat{\mathbf{q}}) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}(\mathbf{C}_t^{-1}\mathbf{A}_t\mathbf{P} + \mathbf{C}_t^{-1}\mathbf{B}_t\mathbf{P}') \\ &= \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\mathbf{C}_t^{-1}(\mathbf{P}^{\mathrm{T}}\mathbf{W}\mathbf{P} + \mathbf{P}'^{\mathrm{T}}\mathbf{W}\mathbf{P}') = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\mathbf{C}_t^{-1}\mathbf{C}_t = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\end{aligned} \tag{75}$$

Eqs. (74) and (75) show that Hermite-type interpolation can interpolate exactly any member of the basis polynomial and, consequently, any linear combination of the basis polynomial.

It can easily be shown that the truncated Hermite-type interpolation also satisfies the consistency conditions. However, unlike the Hermite-type interpolation, in a truncated Hermite-type interpolation, $\mathbf{u}$ is only interpolated by $\hat{\mathbf{u}}$ and $\mathbf{q}$ is only interpolated by $\hat{\mathbf{q}}$. The shape functions $\overline{\mathbf{M}}$ and $\overline{\mathbf{T}}$ have the property of partion of unity which are given by

$$\sum_{I=1}^{NP} \overline{M}_I(x,y) = 1, \tag{76}$$

$$\sum_{I=1}^{NP} \overline{T}_I(x,y) = 1. \tag{77}$$

5.4. Comparison of interpolation functions

Consider a portion of the boundary shown in Fig. 5. Points are uniformly distributed on the boundary. The horizontal segment of the boundary has 20 evenly distributed points. Interpolation functions $M(x,y)$, $N(x,y)$, $S(x,y)$ and $T(x,y)$ are computed for four points on the horizontal segment using a five-point cloud. The coordinates of the four points are point $1 = (0.025, 0)$, point $2 = (0.475, 0)$, point $3 = (0.525, 0)$, point $4 = (0.975, 0)$. A comparison of all the interpolation functions for the four points is shown in Figs. 6–9.

In the truncated Hermite-type approach, the only nonzero interpolation functions are $\overline{M}(x,y)$ and $\overline{T}(x,y)$. These two interpolation functions are shown Figs. 10 and 11 for the four points shown in Fig. 5.

Remarks

1. For the regular Hermite-type interpolation functions shown in Figs. 6–9, note that the interpolation functions N and S are zero for interior nodes. For nodes on the edge of the boundary or close to the edge of the boundary, N and S are not zero.

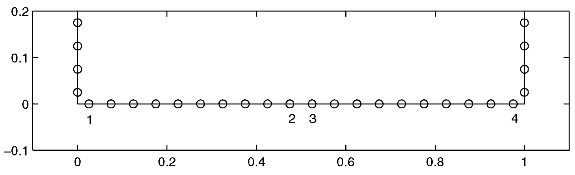

Fig. 5. A portion of boundary where various interpolation functions are computed and compared.

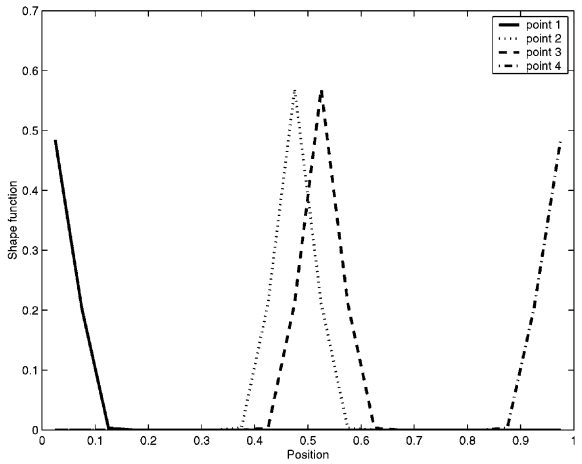

Fig. 6. Interpolation function M for the four points shown in Fig. 5.

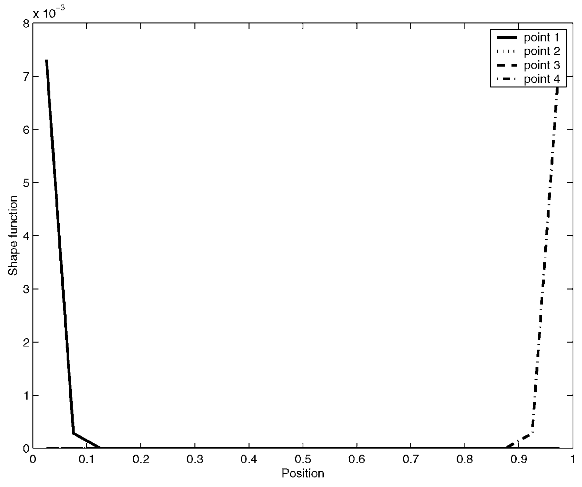

Fig. 7. Interpolation function N for the four points shown in Fig. 5.

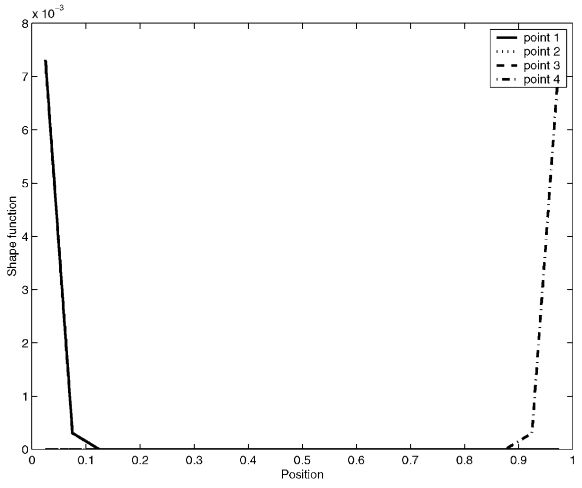

Fig. 8. Interpolation function S for the four points shown in Fig. 5.

2. Note that the interpolation functions for the interior nodes are identical with both truncated and non-truncated approaches. However, for nodes close to the edge of the boundary, both approaches give different interpolation functions.

6. Discretization and integration

The boundary of the domain is discretized into cells for integration purpose. As shown in Fig. 12, each cell contains a certain number of nodes and the number of nodes can vary from cell to cell. The concept of cell is quite different from that of an element or a panel in BEMs. The cell can be of any shape or size and the only restriction is that the union of all the cells equal the boundary of the domain. Assuming that the

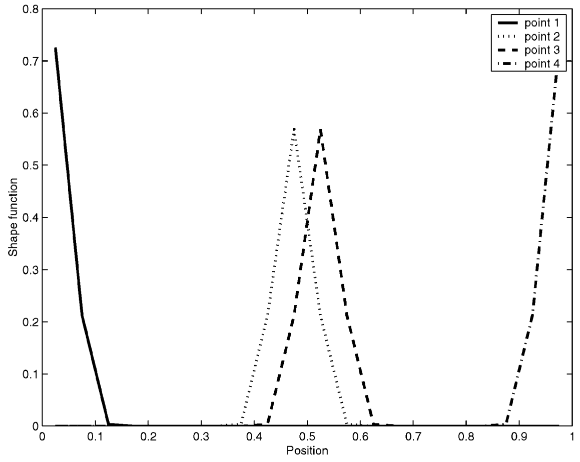

Fig. 9. Interpolation function T for the four points shown in Fig. 5.

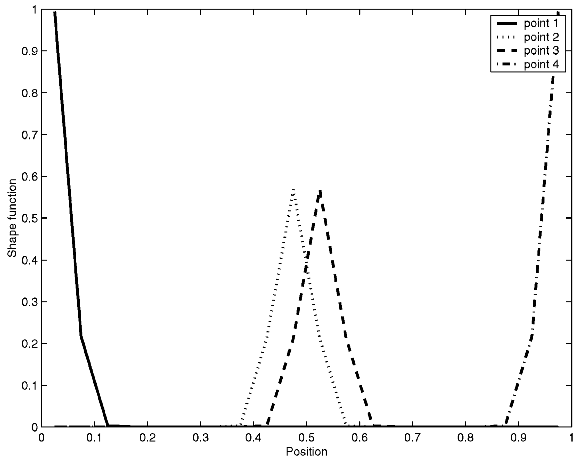

Fig. 10. Interpolation function $\overline{M}$ for the four points shown in Fig. 5.

boundary is discretized into NC cells, the boundary integral equation for the potential problem given in Eq. (4) can be rewritten as

$$\sum_{k=1}^{NC} \int_{\mathrm{d}S_k} \left\{ \ln r(P,Q)q(Q) - \frac{\partial \ln r(P,Q)}{\partial n_Q}[u(Q) - u(P)] \right\} \mathrm{d}S_Q = 0, \tag{78}$$

where $\mathrm{d}S_k$ is the boundary length of cell k. The integral over a single cell k can be expressed as

$$I_k = \int_{\mathrm{d}S_k} \ln r(P,Q)q(Q)\,\mathrm{d}S_Q - \int_{\mathrm{d}S_k} \frac{\partial \ln r(P,Q)}{\partial n_Q} u(Q)\,\mathrm{d}S_Q + u(P)\int_{\mathrm{d}S_k} \frac{\partial \ln r(P,Q)}{\partial n_Q}\,\mathrm{d}S_Q. \tag{79}$$

Consider the first integral in Eq. (79) where the integrand is log singular. When the source point P and the field point Q are far away from each other, the singularity is weak. On the contrary, when the source

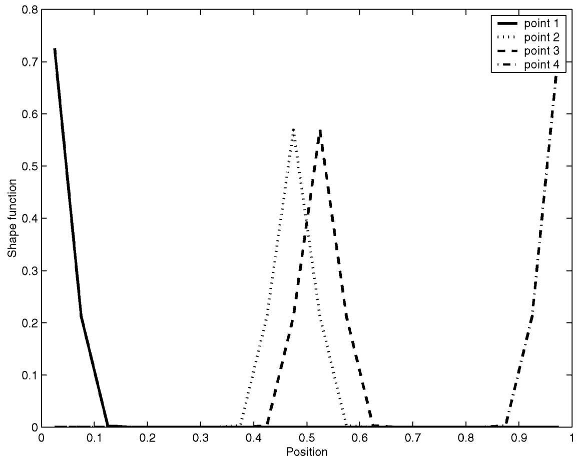

Fig. 11. Interpolation function $\overline{T}$ for the four points shown in Fig. 5.

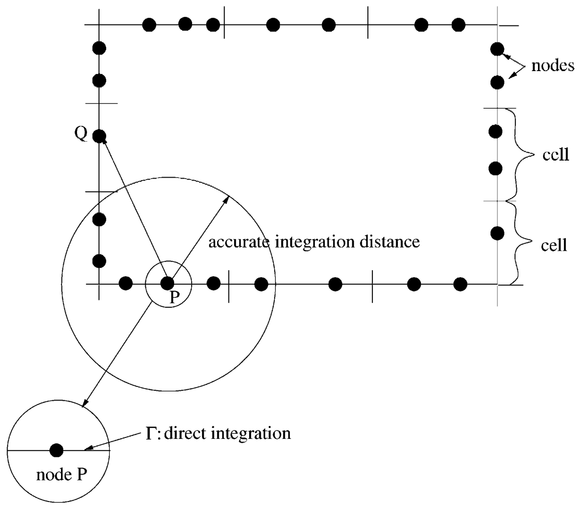

Fig. 12. For integration purpose, the boundary of the domain is broken into cells.

point P and the field point Q are close to each other or coincide, the integral becomes nearly singular or singular, respectively. The use of regular Gauss quadrature would produce significant error in such cases and special treatment needs to be applied. As shown in Fig. 12, given a source point P, cells are classified into three categories: (1) cells which are far away from the source point; (2) cells which are near the source point; (3) a cell which includes the source point. As shown in Fig. 12, given a source point and a distance, defined as accurate integration distance in this paper, a circle centered at the source point and with the radius of a given distance can be drawn. Any cell which is either completely or partially inside the circle is considered to be a nearby cell for the source point. If a cell is completely outside of the circle, it is defined as a far-away cell. Varying numerical integration schemes are applied depending on the classification of the cell and these are described below.

6.1. Cells are far away from the source point

When a cell is far away from the source point P, regular Gauss quadrature is sufficient for numerical integration. For a given cell k, the boundary integral in Eq. (79) can be written as

$$I1_k = \sum_{j=1}^{NG_k} \eta_j \ln r(P,Q_j) q(Q_j) - \sum_{j=1}^{NG_k} \eta_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}} u(Q_j) + u(P) \sum_{j=1}^{NG_k} \eta_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}, \tag{80}$$

where NG_k is the number of Gauss points in cell k and η_j is the weight of Gauss point j. Substituting the approximations for u and q at the Gauss points into Eq. (80), we obtain

$$\begin{aligned} I1_k = {} & \sum_{i=1}^{NP} \sum_{j=1}^{NG_k} \eta_j \ln r(P,Q_j) \left[S_i(Q_j)\hat{u}_i + T_i(Q_j)\hat{q}_i\right] - \sum_{i=1}^{NP} \sum_{j=1}^{NG_k} \eta_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}} \left[M_i(Q_j)\hat{u}_i + N_i(Q_j)\hat{q}_i\right] \\ & + \sum_{i=1}^{NP} \left[M_i(P)\hat{u}_i + N_i(P)\hat{q}_i\right] \sum_{j=1}^{NG_k} \eta_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}. \end{aligned} \tag{81}$$

6.2. Cells are near the source point

When a cell is near the source point, the first integral in Eq. (79) becomes nearly singular. The regular Gauss quadrature becomes inaccurate. The integration is performed by employing a Nyström scheme [12,13] and a singular value decomposition (SVD) [14] technique to compute the quadrature weights directly for the nodes (not Gauss points) within each cell. Consider the integration in Eq. (79) over cell k. Assuming that there are M_k nodes in cell k, Eq. (79) can be rewritten as

$$I2_k = \sum_{j=1}^{M_k} \xi_j \ln r(P,Q_j) q(Q_j) - \sum_{j=1}^{M_k} \bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}} u(Q_j) + u(P) \sum_{j=1}^{M_k} \bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}, \tag{82}$$

where ξ_j is the quadrature weight for node j for the first integral in Eq. (79) and $\bar{\xi}_j$ is the quadrature weight for node j for the second and third integrals in Eq. (79). Note that the quadrature points are the nodes themselves in each cell and no additional quadrature points (such as Gauss points) are used for integration. The weights ξ_j and $\bar{\xi}_j$ are computed such that the integration is exact for a set of basis functions. For example, for the nodes in cell k, the ξ_j weights are computed by integrating the basis functions $[1, x, y]$ along with Green's function, i.e.

$$\sum_{j=1}^{M_k} \left[\xi_j \ln r(P,Q_j)\right] = \int_{\partial S_k} \ln r(P,Q)\, \mathrm{d}S_Q, \tag{83}$$

$$\sum_{j=1}^{M_k} \left[\xi_j \ln r(P,Q_j) x_j\right] = \int_{\partial S_k} \ln r(P,Q) x\, \mathrm{d}S_Q, \tag{84}$$

$$\sum_{j=1}^{M_k} \left[\xi_j \ln r(P,Q_j) y_j\right] = \int_{\partial S_k} \ln r(P,Q) y\, \mathrm{d}S_Q. \tag{85}$$

Similarly the $\bar{\xi}_j$ weights can be computed by integrating the basis functions with the derivative of Green's function

$$\sum_{j=1}^{M_k}\left[\bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}\right] = \int_{\partial S_k} \frac{\partial \ln r(P,Q)}{\partial n_Q}\,\mathrm{d}S_Q, \tag{86}$$

$$\sum_{j=1}^{M_k}\left[\bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}} x_j\right] = \int_{\partial S_k} \frac{\partial \ln r(P,Q)}{\partial n_Q} x\,\mathrm{d}S_Q, \tag{87}$$

$$\sum_{j=1}^{M_k}\left[\bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}} y_j\right] = \int_{\partial S_k} \frac{\partial \ln r(P,Q)}{\partial n_Q} y\,\mathrm{d}S_Q. \tag{88}$$

Eqs. (83)–(85) can be rewritten as

$$\mathbf{K}\xi = \mathbf{R} \tag{89}$$

and Eqs. (86)–(88) can be rewritten as

$$\mathbf{H}\bar{\xi} = \mathbf{P}. \tag{90}$$

In Eqs. (89) and (90), **K** and **H** are matrices of size $3 \times M_k$, ξ and $\bar{\xi}$ are unknown vectors of size $M_k \times 1$, and **R** and **P** are known right-hand side vectors of size 3×1. The right-hand side vectors, **R** and **P**, are computed analytically. Typically, the rank of **K** and **H** matrices is 2 as all the nodes in a cell lie along a linear segment. If only two nodes are contained in a cell (i.e. $M_k = 2$), then there is a unique solution for the linear systems given in Eqs. (89) and (90). However, there is no constraint on the number of nodes that can be in each cell. If the number of nodes in a cell exceeds 2, the linear system of equations in Eqs. (89) and (90) are under-determined. Similarly, if the number of nodes in a cell is less than 2, Eqs. (89) and (90) are over-determined. In such cases, SVD [14] can be used effectively to solve the linear system of equations. Using the SVD technique, the coefficient matrices **K** and **H** are decomposed as

$$\mathbf{K} = \mathbf{U_K W_K V_K^{\mathrm{T}}}, \tag{91}$$

$$\mathbf{H} = \mathbf{U_H W_H V_H^{\mathrm{T}}}, \tag{92}$$

and the weights are obtained by

$$\xi = \mathbf{V_K \overline{W}_K U_K^{\mathrm{T}} R}, \tag{93}$$

$$\bar{\xi} = \mathbf{V_H \overline{W}_H U_H^{\mathrm{T}} P}, \tag{94}$$

where $\mathbf{\overline{W}_K}$ and $\mathbf{\overline{W}_H}$ are diagonal matrices whose diagonal entries are the reciprocal of the diagonal entries of $\mathbf{W_K}$ and $\mathbf{W_H}$, respectively.

Once the weights are computed, by substituting the approximations of u and q into Eq. (82), we obtain

$$\begin{aligned} I2_k = {} & \sum_{i=1}^{NP}\sum_{j=1}^{M_k} \xi_j \ln r(P,Q_j)\left[S_i(Q_j)\hat{u}_i + T_i(Q_j)\hat{q}_i\right] - \sum_{i=1}^{NP}\sum_{j=1}^{M_k} \bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}\left[M_i(Q_j)\hat{u}_i + N_i(Q_j)\hat{q}_i\right] \\ & + \sum_{i=1}^{NP}\left[M_i(P)\hat{u}_i + N_i(P)\hat{q}_i\right]\sum_{j=1}^{M_k}\bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}. \end{aligned} \tag{95}$$

6.3. Cell contains the source point

When the source node P and field node Q fall into the same cell, a small portion of the boundary in the vicinity of P (defined as Γ in Fig. 12), on which the integration of Green's function is singular, is taken out of the cell and is integrated analytically. The rest of the cell is integrated numerically and the weights of the nodes are determined by using the approach described in Section 6.2.

For example, for cell k, the first integral in Eq. (79) can be rewritten as

$$I3_k = I3_k^{\mathrm{d}S_k-\Gamma} + I3_k^{\Gamma}, \tag{96}$$

$$\begin{aligned} I3_k^{\mathrm{d}S_k-\Gamma} &= \sum_{i=1}^{NP}\sum_{j=1}^{M_k-1}\left[\xi_j \ln r(P,Q_j)\right][(S_i(Q_j)\hat{u}_i + T_i(Q_j)\hat{q}_i)] \\ &\quad - \sum_{i=1}^{NP}\sum_{j=1}^{M_k-1}\left[\bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}\right][M_i(Q_j)\hat{u}_i + N_i(Q_j)\hat{q}_i] \\ &\quad + \sum_{i=1}^{NP}\sum_{j=1}^{M_k-1}\left[\bar{\xi}_j \frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}\right][(M_i(P)\hat{u}_i + N_i(P)\hat{q}_i)], \end{aligned} \tag{97}$$

$$\begin{aligned} I3_k^{\Gamma} &= \sum_{i=1}^{NP}\left[\int_\Gamma \ln r(P,Q)\frac{\partial \mathbf{p}(Q)}{\partial n_Q}\,\mathrm{d}S_Q\right][\mathbf{C}_P^{-1}(\mathbf{A}_P\hat{u}_i + \mathbf{B}_P\hat{q}_i)] \\ &\quad - \sum_{i=1}^{NP}\left[\int_\Gamma \frac{\partial \ln r(P,Q)}{\partial n_Q}\mathbf{p}(\mathbf{Q})\,\mathrm{d}S_Q\right][\mathbf{C}_P^{-1}(\mathbf{A}_P\hat{u}_i + \mathbf{B}_P\hat{q}_i)] \\ &\quad + \sum_{i=1}^{NP}\left[\int_\Gamma \frac{\partial \ln r(P,Q)}{\partial n_Q}\,\mathrm{d}S_Q\right][(M_i(P)\hat{u}_i + N_i(P)\hat{q}_i)]. \end{aligned} \tag{98}$$

The weights ξ_j and $\bar{\xi}_j$ ($j = 1,\ldots,M_k - 1$) in Eq. (97) are computed by employing Eqs. (83)–(90). In Eq. (98), the cloud is centered at the source point P and $\mathbf{C}_P$, $\mathbf{A}_P$ and $\mathbf{B}_P$ are evaluated at point P. The integrals on Γ in Eq. (98) are computed analytically (see Appendix B for details).

In summary, for a source point P, the discretized form of Eq. (79) can be written as

$$\sum_{k=1}^{N_1} I1_k + \sum_{k=1}^{N_2} I2_k + I3_k = 0, \tag{99}$$

where N_1 is the number of far-away cells, N_2 is the number of nearby cells and $N_1 + N_2 + 1 = NC$.

The discretization and integration of the boundary integral equation given in Eq. (79) using the truncated Hermite-type interpolation can be carried out along the same lines as the approach described in Sections 6.1–6.3 for the regular Hermite-type interpolations. The only difference is the interpolation itself, which for a truncated Hermite-type approach is given by

$$u(X) = \sum_{i=1}^{NP} \overline{M}_i(X)\hat{u}_i, \quad X = P \text{ or } Q, \tag{100}$$

$$q(X) = \sum_{i=1}^{NP} \overline{T}_i(X)\hat{q}_i, \quad X = P \text{ or } Q. \tag{101}$$

7. Global matrix assembly and imposition of boundary conditions

When the regular Hermite-type interpolation is used, by substituting Eqs. 81, (95)–(98) into Eq. (99) for each node, the final matrix form in the BCM is given by

$$\mathbf{F}\hat{\mathbf{u}} = \mathbf{G}\hat{\mathbf{q}}. \tag{102}$$

In a general mixed boundary value problem, either u or q is specified at each point. By rearranging **u** and **q**, we put all the knowns given by the boundary conditions into a vector **x** and all the unknowns which are to be computed into another vector **y**. By appropriately exchanging the rows of **M**, **N**, **S** and **T**, Eqs. (34) and (35) can be rewritten as

$$\mathbf{x} = \mathbf{M}'\hat{\mathbf{u}} + \mathbf{N}'\hat{\mathbf{q}}, \tag{103}$$

$$\mathbf{y} = \mathbf{S}'\hat{\mathbf{u}} + \mathbf{T}'\hat{\mathbf{q}}. \tag{104}$$

By combining Eqs. (102)–(104), a $2NP \times 2NP$ linear system is obtained:

$$\begin{bmatrix} \mathbf{F} & -\mathbf{G} \\ \mathbf{M}' & \mathbf{N}' \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}} \\ \hat{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{x} \end{bmatrix}. \tag{105}$$

$\hat{\mathbf{u}}$ and $\hat{\mathbf{q}}$ are computed by solving the above linear system. Once $\hat{\mathbf{u}}$ and $\hat{\mathbf{q}}$ are known, the unknown vector **y** is computed from Eq. (104).

When a truncated Hermite-type interpolation is used, the matrix form of the BCM and the interpolations are given by

$$\overline{\mathbf{F}}\hat{\mathbf{u}} = \overline{\mathbf{G}}\hat{\mathbf{q}}, \tag{106}$$

$$\mathbf{u} = \overline{\mathbf{M}}\hat{\mathbf{u}}, \tag{107}$$

$$\mathbf{q} = \overline{\mathbf{T}}\hat{\mathbf{q}}. \tag{108}$$

A procedure, similar to the one described in Eqs. (102)–(105) can be employed to compute the unknowns $\hat{\mathbf{u}}$ and $\hat{\mathbf{q}}$.

8. Towards true meshless boundary-only analysis

In the approach described in Section 7, a background cell structure is used to compute weights for the scattered points. Since a background cell structure is used for integration, the technique described in Section 7 may not be considered true meshless even though the interpolation functions are constructed by a true meshless approach.

A trapezoidal rule, which employs the nodal volumes of the nodes as weights, can also be used to evaluate Eq. (79). For 2-D problems, where the boundary is 1-D, the nodal volumes can be computed in a straightforward manner. For example, the weighting function centered at each point can be used to compute the nodal volumes. For 1-D boundaries, as shown in Fig. 13, the nodal volume of node 2 is half of the curve length between node 1 and node 3. The curve length between two nodes is approximated by the distance between the two nodes.

Given a source point P (see Fig. 13), the first integral in Eq. (79) is either singular or close to singular in the vicinity of the source point P. The use of trapezoidal rule in a small region that contains P gives large error. Thus, for any given source point P, the boundary is discretized into three parts: (i) Γ_1 which contains point P and a small neighborhood—in this region the integral is computed analytically (described in Section

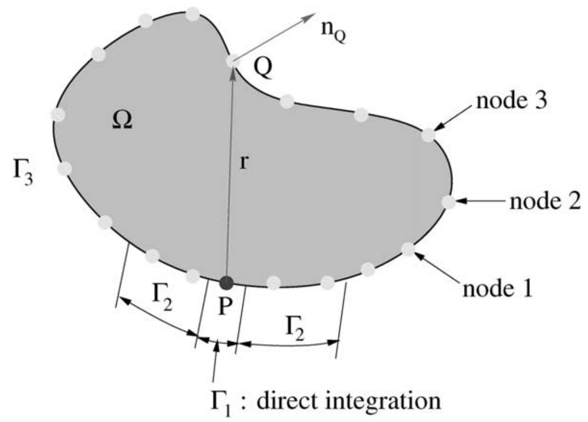

Fig. 13. Discretization scheme of the true meshless boundary-only analysis.

6.3 and Appendix B); (ii) Γ_2, the vicinity of point P, where the integral is close to singular—in this region, the weights of the nodes are computed by the SVD scheme described in Section 6.2; and (iii) Γ_3, which is the rest of the boundary—for nodes on Γ_3, the nodal volume of each node is taken as the weight.

Summarizing, the first integral in Eq. (79) can be rewritten as

$$\begin{aligned}\int_{\partial\Omega}[\ln r(P,Q)q(Q)]\,\mathrm{d}S_Q &= \int_{\Gamma_1+\Gamma_2+\Gamma_3}[\ln r(P,Q)q(Q)]\,\mathrm{d}S_Q \\ &= \sum_{i=1}^{NP}\left[\int_{\Gamma_1}\ln r(P,Q)\frac{\partial \mathbf{p}(Q)}{\partial n_Q}\,\mathrm{d}S_Q\right][\mathbf{C}_P^{-1}(\mathbf{A}_P\hat{u}_i+\mathbf{B}_P\hat{q}_i)] \\ &\quad+\sum_{j=1}^{N_2}\sum_{i=1}^{NP}[\xi_j\ln r(P,Q_j)][(S_i(Q_j)\hat{u}_i+T_i(Q_j)\hat{q}_i)] \\ &\quad+\sum_{j=1}^{N_3}\sum_{i=1}^{NP}[v_j\ln r(P,Q_j)][(S_i(Q_j)\hat{u}_i+T_i(Q_j)\hat{q}_i)],\end{aligned} \tag{109}$$

where ξ_j is the weight for nodes contained on Γ_2 and v_j is the nodal volume of nodes contained on Γ_3.

Similarly, the second integral in Eq. (79) can be rewritten as

$$\begin{aligned}\int_{\partial S_k}\left[\frac{\partial\ \ln r(P,Q)}{\partial n_Q}u(Q)\right]\mathrm{d}S_Q &= \int_{\Gamma_1+\Gamma_2+\Gamma_3}\left[\frac{\partial\ \ln r(P,Q)}{\partial n_Q}u(Q)\right]\mathrm{d}S_Q \\ &= \sum_{i=1}^{NP}\left[\int_{\Gamma_1}\frac{\partial\ \ln r(P,Q)}{\partial n_Q}\mathbf{p}(\mathbf{Q})\,\mathrm{d}S_Q\right][\mathbf{C}_P^{-1}(\mathbf{A}_P\hat{u}_i+\mathbf{B}_P\hat{q}_i)] \\ &\quad+\sum_{j=1}^{N_2}\sum_{i=1}^{NP}\left[\bar{\xi}_j\frac{\partial\ \ln r(P,Q_j)}{\partial n_{Q_j}}\right][M_i(Q_j)\hat{u}_i+N_i(Q_j)\hat{q}_i] \\ &\quad+\sum_{j=1}^{N_3}\sum_{i=1}^{NP}\left[v_j\frac{\partial\ \ln r(P,Q_j)}{\partial n_{Q_j}}\right][M_i(Q_j)\hat{u}_i+N_i(Q_j)\hat{q}_i],\end{aligned} \tag{110}$$

where $\bar{\xi}_j$ is the weight for nodes contained on Γ_2.

Finally, the third integral in Eq. (79) can be rewritten as

$$\sum_{i=1}^{NP}[M_i(P)\hat{u}_i + N_i(P)\hat{q}_i]\int_{\partial S_k}\frac{\partial \ln r(P,Q)}{\partial n_Q}\,\mathrm{d}S_Q$$
$$= \sum_{i=1}^{NP}[(M_i(P)\hat{u}_i + N_i(P)\hat{q}_i)]\left\{\left[\int_{\Gamma_1}\frac{\partial \ln r(P,Q)}{\partial n_Q}\,\mathrm{d}S_Q\right] + \sum_{j=1}^{N_2}\left[\bar{\xi}_j\frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}\right] + \sum_{j=1}^{N_3}\left[v_j\frac{\partial \ln r(P,Q_j)}{\partial n_{Q_j}}\right]\right\}. \tag{111}$$

In Eqs. (109)–(111), N_2 is the number of points on Γ_2, N_3 is the number of points on Γ_3 and $1 + N_2 + N_3 = NP$. Since Γ_1 and Γ_2 are defined locally for each source point, there is no global geometrical discretization. Thus this approach can be considered as a true meshless approach. Even though the assignment of nodal volumes is trivial for 2-D problems (where the boundary is 1-D), the assignment of nodal volumes for randomly distributed points on 3-D geometries (where the boundary is 2-D) can be involved. In Section 9, we compare the performance of the true meshless approach with that of the BCM.

9. Numerical examples

In this section we present results for several problems using the BCM. Unless otherwise mentioned we use the 2-D cubic spline weighting function as given in Eq. (16). The convergence of the method is measured by using a global error measure [7]

$$\epsilon = \frac{1}{|u^{(e)}|_{\max}}\sqrt{\frac{1}{NP}\sum_{I=1}^{NP}\left[u_I^{(e)} - u_I^{(c)}\right]^2} \tag{112}$$

where ϵ is the error in the solution and the superscripts (e) and (c) denote, respectively, the exact and the computed solutions.

Consider the solution of a Laplace equation on various domains as shown in Fig. 14. Fig. 14(a) is a 1×1 square domain where Dirichlet or Neumann boundary conditions are specified along the four edges of the boundary. Five examples with different boundary conditions are presented. In Fig. 14(b), the 1×1 square domain shown in Fig. 14(a) is rotated by 45° and two cubic Dirichlet and Neumann problems are solved over the rotated domain. For a uniform distribution of nodes with a linear basis, the cloud sizes are chosen to be $r_x = r_y = 1.17\Delta s$, where Δs is the spacing between the points. A uniform cell structure is used and each cell contains two Gauss points. Except for the example which uses a random point distribution, each cell contains four nodes in all the other examples. The accurate integration distance, defined in Fig. 12, is set to be 0.3. Fig. 14(c) is a circular domain with a radius of 1.0. Dirichlet and Neumann problems are solved over the circular domain. Nodes are uniformly distributed along the boundary of the domain. A uniform cell structure with two Gauss points per cell is considered. The cloud sizes are chosen to be $r_x = r_y = 1.17\Delta s$, where Δs is the perimeter of the circle divided by the number of points. The accurate integration distance for the circular domain is set to be 0.8. Fig. 14(d) is an ellipse domain and the boundary of the domain is described by

$$x = a\cos(\theta), \qquad y = b\sin(\theta), \tag{113}$$

where $a = 1.0$ and $b = 0.5$. Dirichlet and Neumann problems are solved over the ellipse domain. The cell structure, the cloud size and the accurate integration distance are the same as those used for the circular domain.

The first example is a Dirichlet problem on the square domain with a quadratic solution for u. Dirichlet boundary conditions are specified along the four edges of the boundary. The boundary conditions are

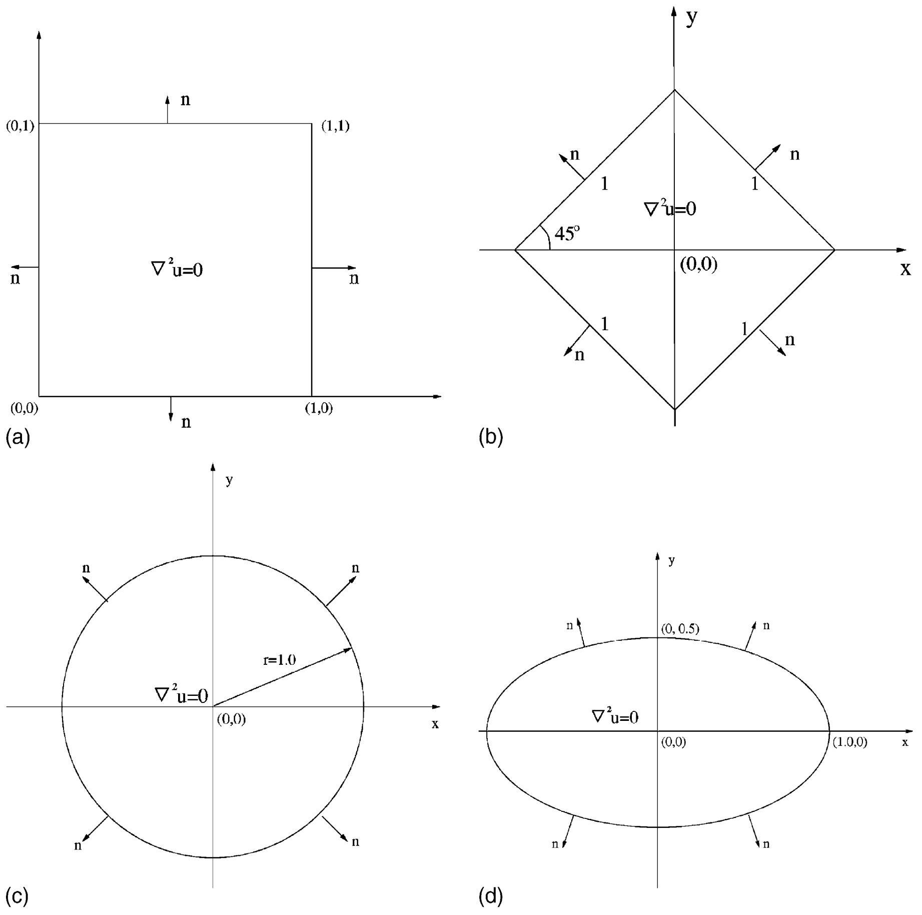

Fig. 14. (a) 1 × 1 square domain on which the potential equation is solved, (b) a rotated square domain, (c) a circular domain, (d) an ellipse domain.

$$u = x^2 - y^2 \quad \text{for } x = 0,\ y = 0,\ x = 1,\ y = 1. \tag{114}$$

The exact solution for q is given by

$$q = \frac{\partial u}{\partial n} = \begin{cases} 0 & \text{for } y = 0,\ x = 0, \\ 2 & \text{for } x = 1, \\ -2 & \text{for } y = 1. \end{cases} \tag{115}$$

The exact solution of u is quadratic, which is outside the linear base interpolating polynomial. This problem is solved by the BCM with a uniform distribution of 8, 16, 32 and 64 points per edge to study the convergence behavior. A comparison of convergence of the classical BEM (described in Section 2, refer Eqs. (6)–(8)) and BEM is shown in Fig. 15. The convergence rate obtained from BEM is found to be 0.48. The convergence rate of the boundary cloud method is found to be 1.46 for both regular and truncated Hermite-type interpolations.

The second example is a Neumann problem on the square domain with a quadratic solution for u. Neumann boundary conditions are specified on the boundary. Dirichlet boundary condition is specified at one point to make the problem well posed. The boundary conditions are

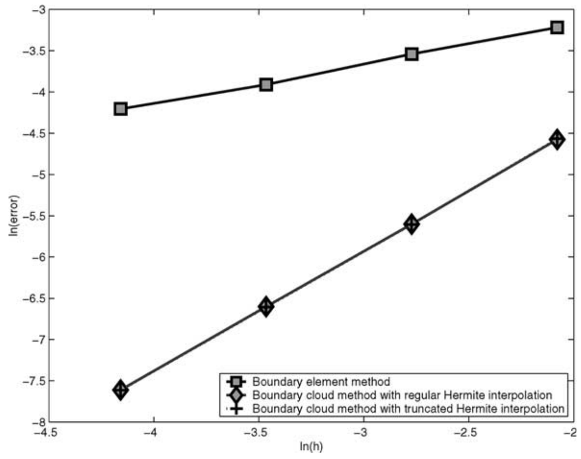

Fig. 15. Comparison of convergence of BEM and BCM for a Dirichlet problem with quadratic exact solution for u.

$$\frac{\partial u}{\partial n} = \begin{cases} 0 & \text{for } y = 0,\ x = 0, \\ 2 & \text{for } x = 1, \\ -2 & \text{for } y = 1, \end{cases} \tag{116}$$

$$u = 0.8789 \quad \text{for } x = 0.9375,\ y = 0. \tag{117}$$

The exact solution for u is given by

$$u = x^2 - y^2 \quad \text{for } x = 0,\ y = 0,\ x = 1,\ y = 1. \tag{118}$$

Comparison between the BEM and BCM is shown in Fig. 16 for a uniform distribution of 8, 16, 32 and 64 points per edge. The convergence rate of the BEM is found to be 1.04. The convergence rate of the BCM is found to be 2.10 for both regular and truncated Hermite-type interpolations.

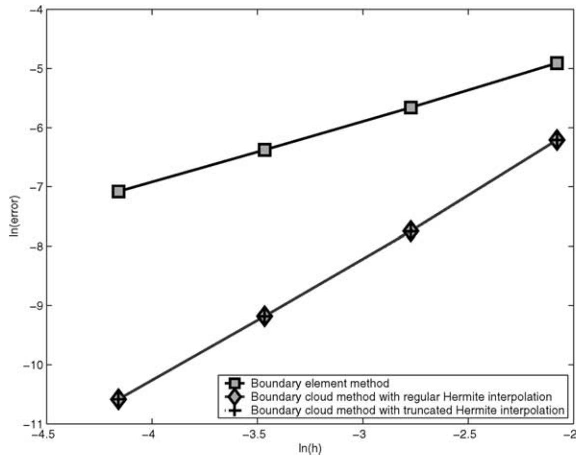

Fig. 16. Comparison of convergence of BEM and BCM for a Neumann problem with quadratic exact solution for u.

The third example is a Dirichlet problem on the square domain with a cubic exact solution for u. The specified Dirichlet boundary conditions are

$$u = -x^3 - y^3 + 3x^2y + 3xy^2 \quad \text{for } x = 0,\ y = 0,\ x = 1,\ y = 1. \tag{119}$$

The exact solution for q is given by

$$q = \frac{\partial u}{\partial n} = \begin{cases} -3y^2 & \text{for } x = 0, \\ -3x^2 & \text{for } y = 0, \\ 3y^2 + 6y - 3 & \text{for } x = 1, \\ 3x^2 + 6x - 3 & \text{for } y = 1. \end{cases} \tag{120}$$

The convergence rate of the BEM and the BCM is shown in Fig. 17 employing a uniform distribution of 8, 16, 32 and 64 points per edge. The convergence rate of the BEM is found to be 0.69 and the convergence rate of the BCM is found to be 1.39 for both regular and truncated Hermite-type interpolations.

The fourth example is a Neumann problem on the square domain with a cubic exact solution for u. The specified boundary conditions are

$$\frac{\partial u}{\partial n} = \begin{cases} -3y^2 & \text{for } x = 0, \\ -3x^2 & \text{for } y = 0, \\ 3y^2 + 6y - 3 & \text{for } x = 1, \\ 3x^2 + 6x - 3 & \text{for } y = 1, \end{cases} \tag{121}$$

$$u = -x^3 - y^3 + 3x^2y + 3xy^2 \quad \text{for } x = 0.97,\ y = 0. \tag{122}$$

The exact solution for u is given by

$$u = -x^3 - y^3 + 3x^2y + 3xy^2 \quad \text{for } x = 0,\ y = 0,\ x = 1,\ y = 1. \tag{123}$$

The convergence rate of the BEM and the BCM is shown in Fig. 18 employing a uniform distribution of 8, 16, 32 and 64 points per edge. The convergence rate of the BEM is found to be 1.21. The convergence rate is found to be 2.06 for regular Hermite-type interpolation and 2.09 for truncated Hermite-type interpolation.

The fifth example is identical to the first example except that the nodes are distributed randomly on the boundary. The convergence plot with a random point distribution of 8, 16, 32 and 64 nodes per edge is shown in Fig. 19. The exact solution and the numerical solution obtained with a random distribution of 32

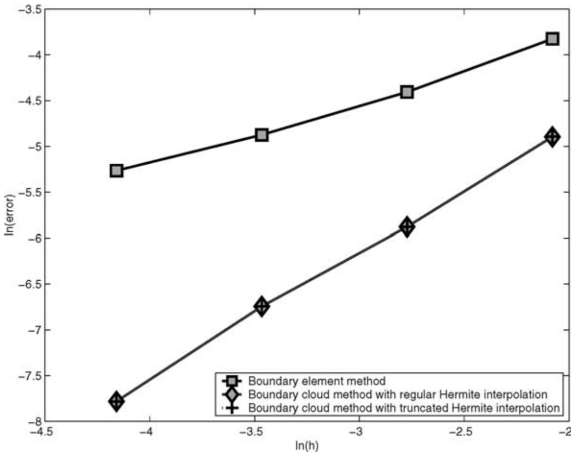

Fig. 17. Comparison of convergence of BEM and BCM for a Dirichlet problem with cubic exact solution for u.

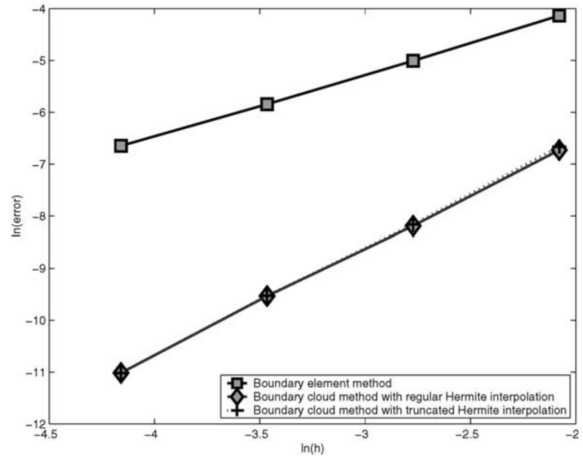

Fig. 18. Comparison of convergence of BEM and BCM for a Neumann problem with cubic exact solution for u.

nodes per edge are compared in Fig. 20. The cloud size in this case is $r_x = r_y = 1.67\Delta s$, where Δs is the spacing between the points. The convergence rate is found to be 1.73 for regular Hermite-type interpolation and 1.82 for truncated Hermite-type interpolation.

The sixth and seventh examples are cubic Dirichlet and Neumann potential problems on the rotated square domain shown in Fig. 14(b). For the Dirichlet problem, the boundary condition is

$$u = -x^3 - y^3 + 3x^2y + 3xy^2 \quad \text{on all the boundary.} \tag{124}$$

The exact solution for q is

$$\frac{\partial u}{\partial n} = (3x^2 - 3y^2)[\cos(\widehat{ny}) - \cos(\widehat{nx})] + 6xy[\cos(\widehat{nx}) + \cos(\widehat{ny})] \quad \text{on all the boundary.} \tag{125}$$

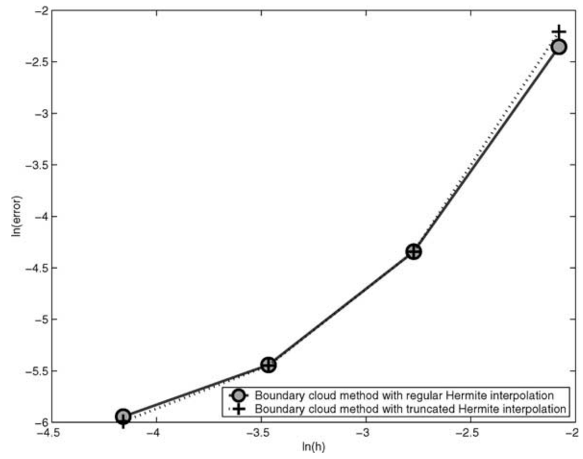

Fig. 19. Convergence of BCM with a random point distribution for a Dirichlet problem with quadratic exact solution for u.

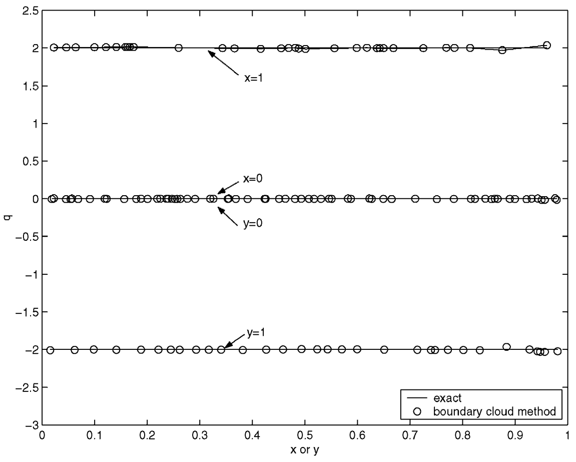

Fig. 20. Comparison of results of BCM and exact solution for a Dirichlet problem with quadratic exact solution for u.

The convergence of the BCM is shown in Fig. 21 and the convergence rate is found to be 1.41 for both regular and truncated Hermite-type interpolations. For the Neumann problem, the boundary conditions are

$$\frac{\partial u}{\partial n} = (3x^2 - 3y^2)[\cos(\widehat{ny}) - \cos(\widehat{nx})] + 6xy[\cos(\widehat{nx}) + \cos(\widehat{ny})] \quad \text{on all the boundary,} \tag{126}$$

$$u = -x^3 - y^3 + 3x^2y + 3xy^2 \quad \text{at one point on the boundary.} \tag{127}$$

The exact solution for u is

$$u = -x^3 - y^3 + 3x^2y + 3xy^2 \quad \text{on all the boundary.} \tag{128}$$

The convergence of the BCM is shown in Fig. 22. The convergence rate is found to be 2.57 for regular Hermite-type interpolation and 2.62 for truncated Hermite-type interpolation.

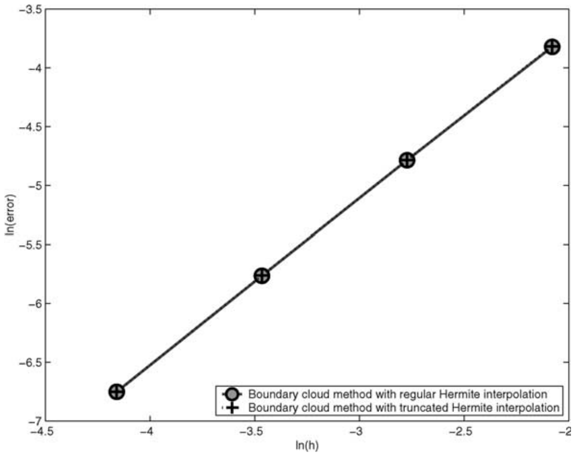

Fig. 21. Convergence of BCM for a Dirichlet problem with cubic exact solution for u on a rotated domain.

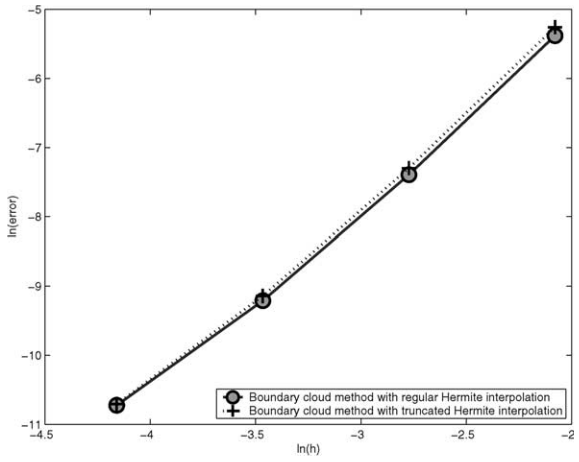

Fig. 22. Convergence of BCM for a Neumann problem with cubic exact solution for u on a rotated domain.

The next two examples are Dirichlet and Neumann potential problems on the circular domain shown in Fig. 14(c). For the Dirichlet problem, the boundary condition is

$$u = r\cos(\theta) = x \quad \text{on all the boundary.} \tag{129}$$

The exact solution for q is

$$\frac{\partial u}{\partial n} = \cos(\theta) \quad \text{on all the boundary.} \tag{130}$$

The convergence of the BCM is shown in Fig. 23 and the convergence rate is found to be 3.61 for regular Hermite-type interpolation. For the Neumann problem, the boundary conditions are

$$\frac{\partial u}{\partial n} = \cos(\theta) \quad \text{on all the boundary,} \tag{131}$$

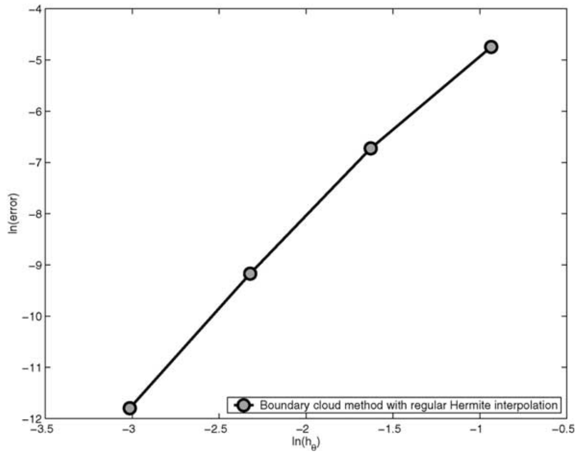

Fig. 23. Convergence of BCM for a Dirichlet problem on a circular domain.

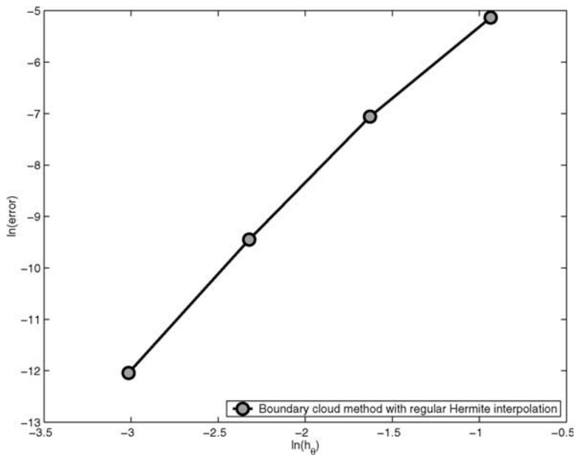

Fig. 24. Convergence of BCM for a Neumann problem on a circular domain.

$$u = x \quad \text{at two points on the boundary.} \tag{132}$$

The exact solution for u is

$$u = x \quad \text{on all the boundary.} \tag{133}$$

The convergence of the BCM is shown in Fig. 24. The convergence rate is found to be 3.55 for regular Hermite-type interpolation.

The next two examples are Dirichlet and Neumann potential problems on the ellipse domain shown in Fig. 14(d). The boundary conditions and the exact solutions for this example are identical to those provided for the circular domain examples. The convergence of the BCM is found to be 2.74 (Fig. 25) for the Dirichlet problem and 2.61 (Fig. 26) for the Neumann problem.

Finally, the first example is solved by using the true meshless approach introduced in Section 8. In this case, for each source point the size of Γ_1 (refer Fig. 13) is taken as the nodal volume of the source point and

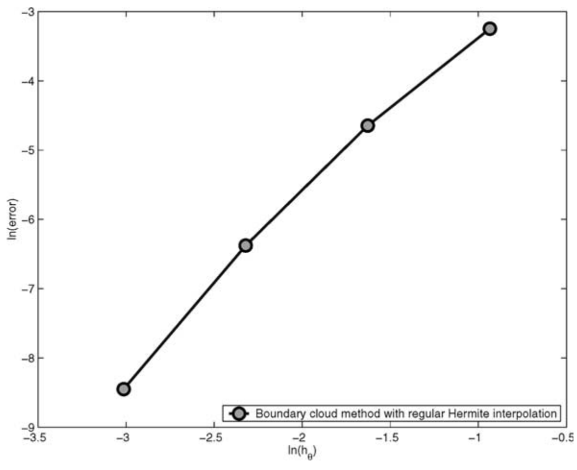

Fig. 25. Convergence of BCM for a Dirichlet problem on an ellipse domain.

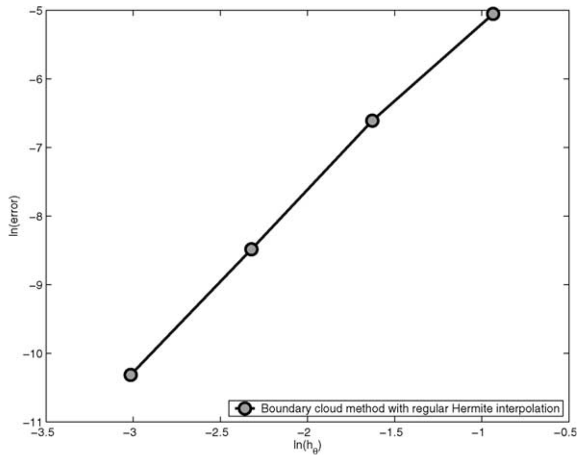

Fig. 26. Convergence of BCM for a Neumann problem on an ellipse domain.

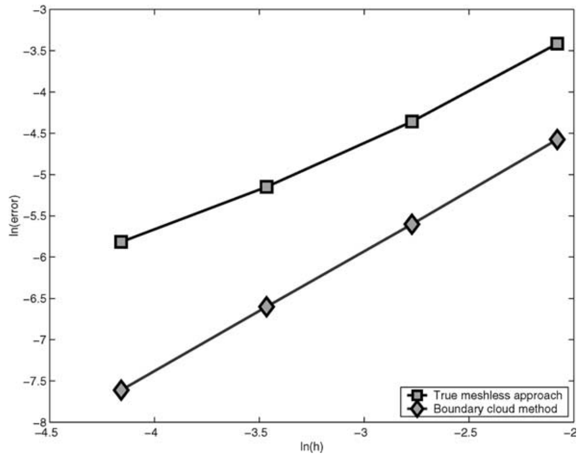

Fig. 27. Comparison of convergence of true meshless approach and BCM for a Dirichlet problem with quadratic exact solution for u.

the size of Γ_2 is the portion of the boundary covered by a 0.335×0.335 square region which is centered at the source point. The convergence rate of the BCM and the true meshless approach is shown in Fig. 27 employing a uniform distribution of 8, 16, 32 and 64 points per edge. The convergence rate of the BCM is found to be 1.46. The convergence rate of the true meshless approach is found to be 1.15.

10. Conclusion

A BCM combining scattered points and boundary integral equations is presented in this paper. One of the difficulties encountered in combining moving least-squares or other least-squares based approaches with boundary integral equations is the development of interpolation functions using Cartesian coordinates. In the boundary node method, the interpolation functions are constructed by employing cyclic or curvilinear coordinates. In this paper, we introduce the construction of Hermite-type interpolation functions for use with boundary integral equations. A key advantage with Hermite-type interpolation functions is the use of Cartesian coordinates. We have also introduced the construction of truncated Hermite-type interpolation functions, which reduces the computational cost of the BCM.

Numerical results are presented for several potential problems comparing the classical BEM and the BCM. The BCM exhibits superior convergence behavior. However, it is more expensive compared to the BEM because of the cost involved in constructing meshless interpolation functions. We also observed that both regular Hermite-type and truncated Hermite-type interpolation functions exhibit similar convergence behavior. We have also attempted to introduce a true meshless approach using boundary integral formulations. Preliminary results obtained with the true meshless approach are promising but several issues still need to be resolved.

Acknowledgements

This work was supported by an NSF CAREER award to N.R. Aluru and by a grant from DARPA under agreement number F30602-98-2-0178. This support is gratefully acknowledged.

Appendix A. Definition of $\overline{\mathrm{N}}$ and $\overline{\mathrm{S}}$ in truncated Hermite-type interpolation

Given a star point t, the shape functions $\overline{\mathbf{N}}$ and $\overline{\mathbf{S}}$, defined in Eqs. (52) and (53), in a truncated Hermite-type approach, are given by

$$\overline{\mathbf{N}}(x,y) = \mathbf{p}(x,y)\overline{\mathbf{C}}_t^{-1}\overline{\mathbf{B}}_t, \tag{A.1}$$

$$\overline{\mathbf{S}}(x,y) = \frac{\partial \mathbf{p}^{\mathrm{T}}(x,y)}{\partial n}\overline{\mathbf{C}}_t^{-1}\overline{\mathbf{A}}_t. \tag{A.2}$$

At any point (x,y), the shape functions can be rewritten as

$$\overline{N}_I(x,y) = \mathbf{p}^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\mathbf{p}'(x_I,y_I)w_I(x_t,y_t), \tag{A.3}$$

$$\overline{S}_I(x,y) = \mathbf{p}'^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\mathbf{p}(x_I,y_I)w_I(x_t,y_t), \tag{A.4}$$

where

$$\mathbf{p}^{\mathrm{T}}(x,y) = [\,1 \quad x \quad y\,], \quad m = 3, \tag{A.5}$$

$$\mathbf{p}'^{\mathrm{T}}(x,y) = \left[0 \quad \frac{\partial x}{\partial n} \quad \frac{\partial y}{\partial n}\right] = [\,0 \quad \cos(\widehat{nx}) \quad \cos(\widehat{ny})\,], \quad m = 3. \tag{A.6}$$

In a truncated Hermite-type interpolation, all nodes included in a cloud have the same normal direction (assuming all boundary edges are straight lines). Denoting $\overline{w}_j(x_t,y_t)$ as $\overline{w}_j^t$, the moment matrix $\overline{\mathbf{C}}_t$ can be expressed as

$$\overline{\mathbf{C}}_t = \begin{bmatrix} \sum_{j=1}^{NP}\overline{w}_j^t & \sum_{j=1}^{NP}\overline{w}_j^t x_j & \sum_{j=1}^{NP}\overline{w}_j^t(kx_j+b) \\ \sum_{j=1}^{NP}\overline{w}_j^t x_j & \sum_{j=1}^{NP}\overline{w}_j^t\{x_j^2+k^2\cos^2(\widehat{ny})\} & \sum_{j=1}^{NP}\overline{w}_j^t\{kx_j^2+bx_j-k\cos^2(\widehat{ny})\} \\ \sum_{j=1}^{NP}\overline{w}_j^t(kx_j+b) & \sum_{j=1}^{NP}\overline{w}_j^t\{kx_j^2+bx_j-k\cos^2(\widehat{ny})\} & \sum_{j=1}^{NP}\overline{w}_j^t\{(kx_j+b)^2+\cos^2(\widehat{ny})\} \end{bmatrix}. \tag{A.7}$$

The inverse of $\overline{\mathbf{C}}_t$ is given by

$$\overline{\mathbf{C}}_t^{-1} = \frac{1}{\det[\overline{\mathbf{C}}_t]}\begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}, \tag{A.8}$$

where

$$\begin{aligned} c_{11} = {} & b^2 \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 - b^2 \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t x_j + \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 \sum_{j=1}^{NP} \overline{w}_j^t \cos^2(\widehat{ny}) \\ & + \sum_{j=1}^{NP} \overline{w}_j^t (kx_j + b)^2 \sum_{j=1}^{NP} \overline{w}_j^t k^2 \cos^2(\widehat{ny}) + 2 \sum_{j=1}^{NP} \overline{w}_j^t x_j (kx_j + b) \sum_{j=1}^{NP} \overline{w}_j^t k \cos^2(\widehat{ny}), \end{aligned} \tag{A.9}$$

$$\begin{aligned} c_{12} = c_{21} = {} & kb \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 - kb \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t x_j - \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t \cos^2(\widehat{ny}) \\ & - \sum_{j=1}^{NP} \overline{w}_j^t (kx_j + b) \sum_{j=1}^{NP} \overline{w}_j^t k \cos^2(\widehat{ny}), \end{aligned} \tag{A.10}$$

$$\begin{aligned} c_{13} = c_{31} = {} & -b \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 + b \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t x_j - \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t k \cos^2(\widehat{ny}) \\ & + \sum_{j=1}^{NP} \overline{w}_j^t (kx_j + b) \sum_{j=1}^{NP} \overline{w}_j^t k^2 \cos^2(\widehat{ny}), \end{aligned} \tag{A.11}$$

$$c_{22} = k^2 \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 - k^2 \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t x_j + \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t \cos^2(\widehat{ny}), \tag{A.12}$$

$$c_{23} = c_{32} = -k \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 + k \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t x_j + \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t k \cos^2(\widehat{ny}), \tag{A.13}$$

$$c_{33} = \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t x_j^2 - \sum_{j=1}^{NP} \overline{w}_j^t x_j \sum_{j=1}^{NP} \overline{w}_j^t x_j + \sum_{j=1}^{NP} \overline{w}_j^t \sum_{j=1}^{NP} \overline{w}_j^t k^2 \cos^2(\widehat{ny}). \tag{A.14}$$

Substituting Eqs. (A.8)–(A.14) into Eq. (A.3)

$$\begin{aligned} \overline{N}_I(x,y) &= \mathbf{p}^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\mathbf{p}'(x_I,y_I)\overline{w}_I^t \\ &= \frac{1}{\det[\overline{\mathbf{C}}_t]}[1 \quad x \quad y]\begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}\begin{bmatrix} 0 \\ \cos(\widehat{nx}) \\ \cos(\widehat{ny}) \end{bmatrix}\overline{w}_I^t \\ &= \frac{1}{\det[\overline{\mathbf{C}}_t]}[3\,1 \quad x \quad y]\begin{bmatrix} c_{12}\cos(\widehat{nx}) + c_{13}\cos(\widehat{ny}) \\ c_{22}\cos(\widehat{nx}) + c_{23}\cos(\widehat{ny}) \\ c_{32}\cos(\widehat{nx}) + c_{33}\cos(\widehat{ny}) \end{bmatrix}\overline{w}_I^t = 0. \end{aligned} \tag{A.15}$$

Recalling that $\overline{\mathbf{C}}_t^{-1}$ is symmetric, $\overline{S}_I(x,y)$ is given by

$$\overline{S}_I(x,y) = \mathbf{p}'^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-1}\mathbf{p}(x_I,y_I)\overline{w}_I^t = [\mathbf{p}^{\mathrm{T}}(x,y)\overline{\mathbf{C}}_t^{-\mathrm{T}}\mathbf{p}'(x_I,y_I)]^{\mathrm{T}}\overline{w}_I^t = 0. \tag{A.16}$$

Thus, the shape functions $\overline{\mathbf{N}}$ and $\overline{\mathbf{S}}$ are both zero in a truncated Hermite-type interpolation.

Appendix B. Analytical integration of singular integrals

As shown in Eq. (98), products of Green's function and the basis polynomial need to be integrated over a straight segment which contains the source point. Green's function for 2-D potential problems is given by

$$G(P,Q) = \ln r(P,Q), \tag{B.1}$$

where r is the distance from P to Q.

The integrals that need to be computed are

$$I_1 = \int_{\mathrm{d}S} G(P,Q)\,\mathrm{d}S_Q \tag{B.2}$$

$$I_2 = \int_{\mathrm{d}S} G(P,Q)x\,\mathrm{d}S_Q \tag{B.3}$$

$$I_3 = \int_{\mathrm{d}S} G(P,Q)y\,\mathrm{d}S_Q \tag{B.4}$$

As shown in Fig. B.1, the source point is denoted as P_0. When the domain of integration is from P_1 to P_2, the integrals in Eqs. (B.2)–(B.4) are singular. However, the Cauchy principle values [1] of the integrals exist. In this case, numerical integration could introduce large errors. Hence, analytical integration is performed to ensure accuracy.

We will assume that the segment is of the form $y = kx + b$. The coordinates of the points are shown in Fig. B.1. The Cauchy principle values of the integrals are given by

$$\begin{aligned} I_1 &= \int_{\mathrm{d}S} G(P,Q)\,\mathrm{d}S_Q \\ &= \sqrt{1+k^2}\left\{|x_1 - x_0|\left(\ln\sqrt{(x_1-x_0)^2 + (y_1-y_0)^2} - 1\right)\right. \\ &\quad \left. + |x_2 - x_0|\left(\ln\sqrt{(x_2-x_0)^2 + (y_2-y_0)^2} - 1\right)\right\}, \end{aligned} \tag{B.5}$$

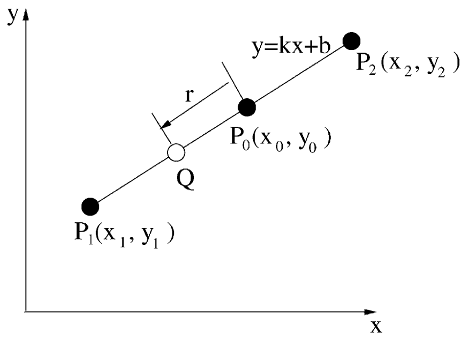

Fig. B.1. Singular integration along a straight segment.

$$\begin{aligned}I_2 &= \int_{\mathrm{d}S} G(P,Q)x\,\mathrm{d}S_Q \\ &= \frac{\sqrt{1+k^2}}{4}\left\{|x_1-x_0|^2\left(\ln\sqrt{(x_1-x_0)^2+(y_1-y_0)^2}-1\right)\right. \\ &\qquad \left. -|x_2-x_0|^2\left(\ln\sqrt{(x_2-x_0)^2+(y_2-y_0)^2}-1\right)\right\} \\ &\quad +\frac{\sqrt{1+k^2}}{2}\left\{|x_1-x_0|x_0\left(\ln\sqrt{(x_1-x_0)^2+(y_1-y_0)^2}-2\right)\right. \\ &\qquad \left. -|x_2-x_0|x_0\left(\ln\sqrt{(x_2-x_0)^2+(y_2-y_0)^2}-2\right)\right\},\end{aligned} \tag{B.6}$$

$$I_3 = \int_{\mathrm{d}S} G(P,Q)y\,\mathrm{d}S_Q = kI_2 + bI_1. \tag{B.7}$$

References

[1] J.H. Kane, Boundary Element Analysis in Engineering Continuum Mechanics, Prentice-Hall, Engelwood Cliffs, NJ, 1994.

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

[3] T.J.R. Hughes, The Finite Element Method, Prentice-Hall, Engelwood Cliffs, NJ, 1987.

[4] P. Lancaster, K. Salkauskas, Surface generated by moving least squares methods, Math. Comput. 37 (1981) 141–158.

[5] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl, Meshless methods: an overview and recent developments, Comput. Meth. Appl. Mech. Engrg. 139 (1996) 3–47.

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

[7] N.R. Aluru, G. Li, Finite cloud method: a true meshless technique based on a fixed reproducing kernel approximation, Int. J. Numer. Meth. Engrg. 50 (10) (2001) 2373–2410.

[8] E. Onate, S. Idelsohn, O.C. Zienkiewicz, R.L. Taylor, C. Sacco, A stabilized finite point method for analysis of fluid mechanics problems, Comput. Meth. Appl. Mech. Engrg. 139 (1996) 315–346.

[9] Y.X. Mukherjee, S. Mukherjee, The boundary node method for potential problems, Int. J. Numer. Meth. Engrg. 40 (1997) 797–815.

[10] M.K. Chati, S. Mukherjee, The boundary node method for three-dimensional problems in potential theory, Int. J. Numer. Meth. Engrg. 47 (2000) 1523–1547.

[11] M.K. Chati, S. Mukherjee, The boundary node method for three-dimensional linear elasticity, Int. J. Numer. Meth. Engrg. 46 (1999) 1163–1184.

[12] L.M. Delves, J.L. Mohamed, Computational Methods for Integral Equations, Cambridge University Press, Cambridge, 1985.

[13] S. Kapur, D.E. Long, High-order Nyström schemes for efficient 3-d capacitance extraction, in: 38th International Conference on Computer-Aided Design, San Jose, CA, USA, 1998, pp. 178–185.

[14] G.H. Golub, C.F. Van Loan, Matrix Computations, John Hopkins University Press, Baltimore, MD, 1989.

[15] G.T.A. Kovas, Micromachined Transducers Sourcebook, McGraw-Hill, New York, 1998.

[16] N.R. Aluru, J. White, An efficient numerical technique for electromechanical simulation of complicated microelectromechanical structures, Sensors Actuators A 58 (1997) 1–11.

[17] T.J. Liszka, C.A.M. Duarte, W.W. Tworzydlo, hp-Meshless cloud method, Comput. Meth. Appl. Mech. Engrg. 139 (1996) 263–288.

[18] S.N. Atluri, J.Y. Cho, H.-G. Kim, Analysis of thin beams, using the meshless local Petrov–Galerkin method, with generalized moving least squares interpolations, Comput. Mech. 24 (1999) 334–347.