Available online at www.sciencedirect.com
ScienceDirect
Mathematics and Computers in Simulation 79 (2008) 763–813
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Review

Meshless methods: A review and computer implementation aspects

Vinh Phu Nguyen [a], Timon Rabczuk [b], Stéphane Bordas [c,*], Marc Duflot [d]

[a] *Ecole Nationale d'Ingénieur de Saint Etienne (ENISE), Laboratoire de Tribologie et Dynamique des Systèmes (LTDS), France*
[b] *University of Canterbury, Department of Mechanical Engineering, 4800 Private Bag, Christchurch, New Zealand*
[c] *University of Glasgow, Civil Engineering, Rankine Building, Glasgow G12 8LT, UK*
[d] *CENAERO, Rue des Frères Wright 29, 6041 Gosselies, BELGIUM*

Received 30 April 2007; received in revised form 17 September 2007; accepted 8 January 2008
Available online 17 January 2008

Abstract

The aim of this manuscript is to give a practical overview of meshless methods (for solid mechanics) based on global weak forms through a simple and well-structured MATLAB code, to illustrate our discourse. The source code is available for download on our website and should help students and researchers get started with some of the basic meshless methods; it includes intrinsic and extrinsic enrichment, point collocation methods, several boundary condition enforcement schemes and corresponding test cases. Several one and two-dimensional examples in elastostatics are given including weak and strong discontinuities and testing different ways of enforcing essential boundary conditions.

Keywords: Meshless methods; Intrinsic enrichment; Extrinsic discontinuities; Computer implementation; MATLAB

Contents

* Corresponding author. Tel.: +44 1413304075.
E-mail addresses: stephane.bordas@alumni.northwestern.edu (S. Bordas), marc.duflot@cenaero.be (M. Duflot).
URL: http://www.civil.gla.ac.uk/~bordas (S. Bordas).

0378-4754/$32.00
doi:10.1016/j.matcom.2008.01.003

1. Introduction

The finite element method has been used with great success in many fields with both academic and industrial applications. It is however not without limitations. Due to mesh-based interpolation, distorted or low quality meshes lead to higher errors, necessitate remeshing, a time and human labour consuming task, which is not guaranteed to be feasible in finite time for complex three-dimensional geometries.

Additionally, due to the underlying structure of the classical mesh-based methods, they are not well suited to treat problems with discontinuities that do not align with element edges. One strategy for dealing with moving discontinuities in mesh-based methods is remeshing or discontinuous enrichment. However, remeshing is costly, still difficult in three dimensions and requires projection of quantities between successive meshes and consequential degradation of accuracy. An alternative to remeshing in a finite element context is the extended finite element method (XFEM) [6,79,24,45,23,22] enriches the approximation space so that weak and strong discontinuities can be captured.

Meshless methods (MMs) were born with the objective of eliminating part of the difficulties associated with reliance on a mesh to construct the approximation. In MMs, the approximation is built from nodes only. One of the first meshless methods is the smooth particle hydrodynamics (SPH) method by Lucy [77] and Gingold and Monaghan [54]. It was born to solve problems in astrophysics and, later on, in fluid dynamics [20,81,80]. Libersky et al. [71] were the first to employ SPH in solid mechanics (impact). Since the original SPH version suffered from spurious instabilities and inconsistencies [97,9,101], many improvements were incorporated into SPH [12,88,20,21,61,62,35,36,100,105]. While SPH and their corrected versions were based on a strong form, other methods were developed in the 1990s, based on a weak form. Major applications of these methods are in solid mechanics. The element-free Galerkin (EFG) method [14] was developed in 1994 and was one of the first meshless methods based on a global weak form. The

reproducing kernel particle method (RKPM) [73] was developed 1 year later. Though the final equations are very similar to the equations of the EFG method, RKPM has its origin in wavelets. In contrast to RKPM and the EFG method, that use a so-called intrinsic basis, other methods were developed that use an extrinsic basis and the partition of unity concept. This extrinsic basis was initially used to increase the approximation order similar to a p-refinement as, e.g. in the hp-cloud method [40,72]. Melenk and Babuška [78] pointed out the similarities of meshless and finite element methods and developed the so-called partition of unity finite element method (PUFEM). The method is very similar to the hp-cloud method. Generally, PUFEM shape functions are based on Lagrange polynomials, while the general form of the hp-cloud method also includes the MLS-approximation. Strouboulis et al. [91] pointed out in their generalized finite element method (GFEM) that different partition of unities can be used for the usual approximation and the so-called enrichment. In the XFEM [6,79,94], the extrinsic enrichment was modified such that it can handle strong discontinuities without remeshing. Moreover, XFEM is based on a local PU concept.

Another class of meshless methods are methods that are based on local weak forms. The most popular method is the meshless local Petrov–Galerkin (MLPG) method [2–4]. The main difference of the MLPG method to methods such as EFG or RKPM is that local weak forms are generated on overlapping subdomains rather than using global weak forms. The integration of the weak form is then carried out in these local subdomains. Atluri [1] introduced the notion "truly" meshless since no construction of a background mesh is needed for integration purposes. Another well known method that was mainly applied in fluid mechanics is the moving point method [83,82,75].

Some major advantages of MMs are (i) *h*-adaptivity is simpler to incorporate in MMs than in mesh-based methods, (ii) problems with moving discontinuities such as crack propagation, shear bands and phase transformation can be treated with ease, (iii) large deformation can be handled more robustly, [30,29], (iv) higher-order continuous shape functions, (v) non-local interpolation character and (vi) no mesh alignment sensitivity. Beside these advantages, MMs are not without disadvantages. The MMs shape functions are rational functions which requires high-order integration scheme to be correctly computed. The treatment of essential boundary conditions is not as straightforward as in mesh-based methods since the MMs shape functions are not interpolants. They do not satisfy the Kronecker delta property. In general, the computational cost of MMs is higher than one of FEM.

To avoid some difficulties inherent in MMs, MMs were coupled successfully to finite element methods, see, e.g. [18,56,47,58,48,49]. Meanwhile, hybrid methods are available that exploit the advantages of meshfree methods and finite elements [55,74,60,106,107], e.g. the shape functions fulfill the Kronecker delta property while simultaneously exploiting the smoothness and higher-order continuity of meshfree shape functions.

The purpose of this manuscript is to give a practical overview of meshless methods, especially with respect to their computer implementation. Common issues in MMs are approximation, integration of the weak form, imposing essential boundary conditions, how to efficiently compute shape functions and how to incorporate strong and weak discontinuities. In addition, the weighted residual methods such as collocation and Galerkin procedures are also stated with examples. Advanced issues in application of MMs to fracture mechanics, coupling MMs with finite elements are reviewed.

Computer implementation aspects of the EFG and enriched EFG are given in detail through a MATLAB code.[1] In particular, the source code of the program includes intrinsic and extrinsic enrichment for cracks and material interfaces.

The paper is organized as follows. Section 2 gives a detailed description of MMs including their approximations, imposition of essential boundary conditions, numerical integration of the weak form. Some of the typical MMs such as the element-free Galerkin method are introduced. The computer implementation aspects are introduced in Section 3. Section 4 presents some numerical examples on linear elasticity.

2. Meshless methods

2.1. Basic approximations

Meshless approximations for a scalar function u in terms of the material (Lagrangian) coordinates can be written as

$$u(\mathbf{x}, t) = \sum_{I \in \mathcal{S}} \Phi_I(\mathbf{x}) u_I(t) \tag{1}$$

[1] Which is available at http://www.civil.gla.ac.uk/~bordas/codes/efgMatlab/EFGMatlabCode.rar.

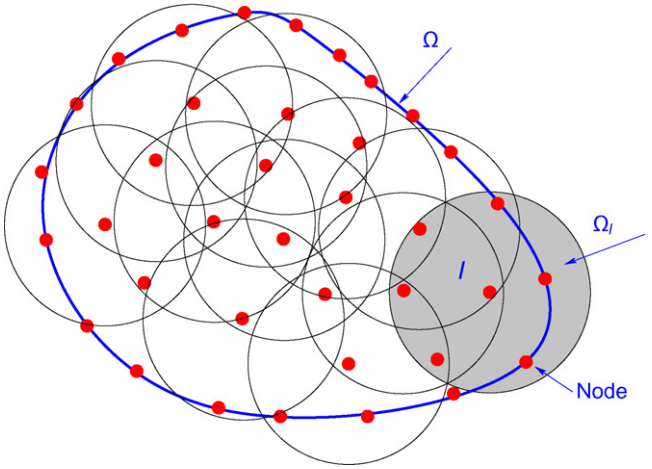

Fig. 1. Discretization using meshless methods: nodes, domains of influence (circular shape).

where Φ_I: $\Omega \to \mathbb{R}$ are the shape functions and the u_I's are the nodal values at particle I located at position $\mathbf{x}_I$ and $\mathcal{S}$ is the set of nodes I for which $\Phi_I(\mathbf{x}) \neq 0$. Note, that the above form is identical to an FEM approximation. However, in contrast to FEM, the shape functions in Eq. (1) are only approximants and not interpolants, since $u_I \neq u(\mathbf{x}_I)$. Therefore special techniques are needed to treat displacement boundary conditions, that will be discussed in a subsequent section.

2.2. Kernel (weight) function

The shape functions Φ_I are obtained from the kernel functions, often called window or weighting functions, which are denoted by w_I: $\Omega \to \mathbb{R}$. The kernel functions have compact support. The support size is defined by the so called dilatation parameter or smoothing length. It is critical to solution accuracy, stability and plays the role of the element size in the finite element method (Fig. 1).

The final characteristics of weight functions is its functional forms. The weight function should be continuous and positive in its support. For all the meshless methods that we will review in this paper, the continuity of the shape function will be determined solely by the continuity of the kernel function, for details see, e.g. [57]. For example, if the kernel function is C^2, then the corresponding shape function is also C^2.

Some commonly used weight functions are

- the cubic spline weight function:

$$w(r) = \begin{cases} \frac{2}{3} - 4r^2 + 4r^3, & r \leq \frac{1}{2} \\ \frac{4}{3} - 4r + 4r^2 - \frac{4}{3}r^3, & \frac{1}{2} < r \leq 1 \\ 0, & r > 1 \end{cases} \tag{2}$$

- the quartic spline weight function:

$$w(r) = \begin{cases} 1 - 6r^2 + 8r^3 - 3r^4, & r \leq 1 \\ 0, & r > 1 \end{cases} \tag{3}$$

with

$$r = \frac{||\mathbf{x}_I - \mathbf{x}||}{d_I} \tag{4}$$

where d_I is the support size of node I.

In two dimensions, circular and rectangular supports are usual.

- Circular support:

$$w(\mathbf{x}-\mathbf{x}_I) = w\left(\frac{||\mathbf{x}_I-\mathbf{x}||}{d_I}\right) \tag{5}$$

- Rectangular support:

$$w(\mathbf{x}-\mathbf{x}_I) = w\left(\frac{|x_I-x|}{d_I^x}\right) w\left(\frac{|y_I-y|}{d_I^y}\right) \tag{6}$$

The derivatives of the weight functions can be computed using the chain rule. For example, for circular supports, we have

$$w_k(r) = w_r(r)r_k = w_r\frac{x_k-x_{Ik}}{rd_I^2} \tag{7}$$

2.3. Completeness

Completeness, often referred to as reproducibility, in Galerkin methods plays the same role as consistency in finite difference methods. Completeness means the ability of an approximation to reproduce a polynomial of a certain order. An approximation is called zero-order complete if it reproduces constant functions exactly. It is called linear (first order) complete if it reproduces linear functions exactly, and so on for higher orders of completeness.

2.4. Partition of unity

A partition of unity (PU) is a paradigm where a domain is divided into overlapping subdomains Ω_I, each of which is associated with a function $\Phi_I(\mathbf{x})$ which is nonzero only in Ω_I and has the following property:

$$\sum_{I=1}^{N}\Phi_I(\mathbf{x}) = 1 \quad \text{in } \Omega \tag{8}$$

Let us recall Eq. (1). There are basically two ways to increase the order of completeness of that approximation. The first opportunity is to increase the completeness of the shape function intrinsically, i.e. by increasing the order of completeness of the shape functions directly. Alternatively, the order of completeness may be increased by modifying Eq. (1) using the partition of unity (PU) concept. In this case, a low-order approximation space (low-order shape functions Φ_I) is enriched with additional functions, which increases the order of completeness. These two concepts will be explained subsequently.

2.5. Intrinsic meshless methods

2.5.1. Smooth particle hydrodynamics

One of the oldest MMs is the smoothed particle hydrodynamics (SPH) [77]:

$$u^h(\mathbf{x}) = \int_\Omega w(\mathbf{x}-\mathbf{y}, h)u(\mathbf{y})\,\mathrm{d}\Omega_{\mathbf{y}} \tag{9}$$

While the continuous form of SPH is second-order complete, it can easily be shown that the discrete SPH form:

$$u^h(\mathbf{x}) = \sum_{I}^{N} w(\mathbf{x}-\mathbf{x}_I)u_I\,\Delta V_I \tag{10}$$

cannot even reproduce constant fields, and hence is not a partition of unity. In Eq. (10), ΔV_I is some measure of the domain surrounding node I.

2.5.2. Reproducing kernel particle method (RKPM)

The reproducing kernel particle method (RKPM) [73] is an improvement of the continuous SPH approximation. In order to increase the order of completeness of the approximation, a correction function $C(\mathbf{x}, \mathbf{y})$ is introduced into the approximation:

$$u^h(\mathbf{x}) = \int_{\Omega_\mathbf{y}} C(\mathbf{x}, \mathbf{y}) w(\mathbf{x} - \mathbf{y}) u(\mathbf{y}) \, \mathrm{d}\Omega_\mathbf{y} \tag{11}$$

where $K(\mathbf{x}, \mathbf{y}) = C(\mathbf{x}, \mathbf{y}) w(\mathbf{x} - \mathbf{y})$ with $C(\mathbf{x}, \mathbf{y})$ is defined such that the approximation is n th order consistent $\mathbf{p}$:

$$u(\mathbf{x}) = \mathbf{p}^\mathrm{T}(\mathbf{x})\mathbf{a} \tag{12}$$

$$\mathbf{p}(\mathbf{x})u(\mathbf{x}) = \mathbf{p}(\mathbf{x})\mathbf{p}^\mathrm{T}(\mathbf{x})\mathbf{a} \tag{13}$$

$$\int_{\Omega_\mathbf{y}} \mathbf{p}(\mathbf{y}) w(\mathbf{x} - \mathbf{y}) u(\mathbf{y}) \, \mathrm{d}\Omega_\mathbf{y} = \int_{\Omega_\mathbf{y}} \mathbf{p}(\mathbf{y}) \mathbf{p}^\mathrm{T}(\mathbf{y}) w(\mathbf{x} - \mathbf{y}) \, \mathrm{d}\Omega_\mathbf{y} \mathbf{a} \tag{14}$$

This is a system of equations for $\mathbf{a}$, which can then be substituted into the approximation $u^h(\mathbf{x}) = \mathbf{p}^\mathrm{T}(\mathbf{x})\mathbf{a}$, it yields:

$$u^h(\mathbf{x}) = \mathbf{p}^\mathrm{T}(\mathbf{x}) \left[\int_{\Omega_\mathbf{y}} \mathbf{p}(\mathbf{y}) \mathbf{p}^\mathrm{T}(\mathbf{y}) w(\mathbf{x} - \mathbf{p}) \, \mathrm{d}\Omega_\mathbf{y} \right]^{-1} \int_{\Omega_\mathbf{y}} \mathbf{p}(\mathbf{y}) w(\mathbf{x} - \mathbf{y}) u(\mathbf{y}) \, \mathrm{d}\Omega_\mathbf{y} \tag{15}$$

with the correction function:

$$C(\mathbf{x}, \mathbf{y}) = \mathbf{p}^\mathrm{T}(\mathbf{x}) \left[\int_{\Omega_\mathbf{y}} \mathbf{p}(\mathbf{y}) \mathbf{p}^\mathrm{T}(\mathbf{y}) w(\mathbf{x} - \mathbf{y}) \, \mathrm{d}\Omega_\mathbf{y} \right]^{-1} \mathbf{p}(\mathbf{y}) = \mathbf{p}^\mathrm{T}(\mathbf{x}) [\mathbf{M}(\mathbf{x})]^{-1} \mathbf{p}(\mathbf{y}) \tag{16}$$

To evaluate this continuous expression, numerical integration must be employed. This step leads from the reproducing kernel method to its discrete version, the reproducing kernel particle method [73]:

$$\begin{aligned} u^h(\mathbf{x}) &= \int_{\Omega_\mathbf{y}} C(\mathbf{x}, \mathbf{y}) w(\mathbf{x} - \mathbf{y}) u(\mathbf{y}) \mathrm{d}\Omega_\mathbf{y} = \sum_{I=1}^{N} C(\mathbf{x}, \mathbf{x}_I) w(\mathbf{x} - \mathbf{x}_I) u_I \, \Delta V_I \\ &= \mathbf{p}^\mathrm{T}(\mathbf{x}) [\mathbf{M}(\mathbf{x})]^{-1} \sum_{I=1}^{N} \mathbf{p}(\mathbf{x}_I) w(\mathbf{x} - \mathbf{x}_I) u_I \, \Delta V_I \end{aligned} \tag{17}$$

The moment matrix $\mathbf{M}(\mathbf{x})$ is also computed by numerical integration:

$$\mathbf{M}(\mathbf{x}) = \int_{\Omega_\mathbf{y}} \mathbf{p}(\mathbf{y}) \mathbf{p}^\mathrm{T}(\mathbf{y}) w(\mathbf{x} - \mathbf{y}) \mathrm{d}\Omega_\mathbf{y} = \sum_{I=1}^{N} \mathbf{p}(\mathbf{x}_I) \mathbf{p}^\mathrm{T}(\mathbf{x}_I) w(\mathbf{x} - \mathbf{x}_I) \, \Delta V_I \tag{18}$$

An interesting remark is observed if we choose $\Delta V_I = 1$: the RKPM and MLS are the same (see next section).

2.5.3. Moving least squares (MLS) approximation

This method was introduced by Shepard [90] in the late 1960s for constructing smooth approximations to fit a specified cloud of points. It was then extended in [65] for general surface generation problems. The most famous application of MLS approximation is probably within the element-free Galerkin (EFG) method, [17,16,7,19].

The approximation $u^h \colon \Omega \to \mathbb{R}$ of the function $u \colon \Omega \to \mathbb{R}$ is posed as a polynomial of order m but with *non-constant* coefficients. The local approximation around a point $\bar{x} \in \Omega$, evaluated at a point $x \in \Omega$ is given by

$$u_L^h(x, \bar{x}) = \mathbf{p}^\mathrm{T}(x)\mathbf{a}(\bar{x}) \tag{19}$$

where $\mathbf{p}(x)$ is a complete polynomial of order m:

$$\mathbf{p}^\mathrm{T}(x) = [1 \quad x \quad x^2, \ldots, x^m] \tag{20}$$

and $\mathbf{a}(x)$ contains non constant coefficients that depend on $\mathbf{x}$ (hence the name "moving"):

$$\mathbf{a}^{\mathrm{T}}(x) = [a_0(x) \quad a_1(x) \quad a_2(x), \ldots a_m(x)] \tag{21}$$

The unknown parameters $a_j(x)$ are determined at any point x, by minimizing a functional $\mathcal{J}(x)$ defined by a weighted[2] average over all nodes $I \in \{1, \ldots, n\}$ where the parameters u_I are specified, of the difference between the local approximation $u_L^h(x_I, x)$ and the value u_I, at node I, of the function u to be approximated:

$$\mathcal{J}(x) = \sum_{I=1}^{n} w(x - x_I)[u_L^h(x_I, x) - u_I]^2 = \sum_{I=1}^{n} w(x - x_I)[\mathbf{p}^{\mathrm{T}}(x_I)\mathbf{a}(x) - u_I]^2 \tag{22}$$

where n is the number of nodes in the neighborhood of x where the weight function $w(x - x_I) \neq 0$.

An extremum of $\mathcal{J}$ in Eq. (22) with respect to the coefficients $\mathbf{a}(x)$ can be obtained by setting the derivative of $\mathcal{J}$ with respect to $\mathbf{a}(x)$ equal to zero. The following equations result:

$$\begin{aligned}
&\sum_{I=1}^{n} w(x - x_I) 2 p_1(x_I)[\mathbf{p}^{\mathrm{T}}(x_I)\mathbf{a}(x) - u_I] = 0 \\
&\sum_{I=1}^{n} w(x - x_I) 2 p_2(x_I)[\mathbf{p}^{\mathrm{T}}(x_I)\mathbf{a}(x) - u_I] = 0 \\
&\vdots \\
&\sum_{I=1}^{n} w(x - x_I) 2 p_m(x_I)[\mathbf{p}^{\mathrm{T}}(x_I)\mathbf{a}(x) - u_I] = 0
\end{aligned} \tag{23}$$

After rearrangements, the above becomes:

$$\sum_{I=1}^{n} w(x - x_I)\mathbf{p}(x_I)\mathbf{p}^{\mathrm{T}}(x_I)\mathbf{a}(x) = \sum_{I=1}^{n} w(x - x_I)\mathbf{p}(x_I)u_I \tag{24}$$

Or more compact as

$$\mathbf{A}(x)\mathbf{a}(x) = \mathbf{B}(x)\mathbf{u} \tag{25}$$

where

$$\mathbf{A}(x) = \sum_{I=1}^{n} w(x - x_I)\mathbf{p}(x_I)\mathbf{p}^{\mathrm{T}}(x_I) \tag{26}$$

and

$$\mathbf{B}(x) = [w(x - x_1)\mathbf{p}(x_1) \quad w(x - x_2)\mathbf{p}(x_2) \ \ldots \ w(x - x_n)\mathbf{p}(x_n)] \tag{27}$$

Solving for $\mathbf{a}(x)$ from Eq. (25) and substituting it into Eq. (19), the MLS approximants can be defined as

$$u^h(x) = \mathbf{p}^{\mathrm{T}}(x)[\mathbf{A}(x)]^{-1}\mathbf{B}(x)\mathbf{u} \tag{28}$$

Recalling the form of the approximation defined in Eq. (1):

$$u^h(x) = \sum_{I=1}^{N} \Phi_I(x)u_I = \mathbf{\Phi}^{\mathrm{T}}(x)\mathbf{u} \tag{29}$$

we can immediately write the MLS shape functions as

$$\mathbf{\Phi}^{\mathrm{T}}(x) = \mathbf{p}^{\mathrm{T}}(x)[\mathbf{A}(x)]^{-1}\mathbf{B}(x) \tag{30}$$

[2] Weight function w defined in Section 2.2.

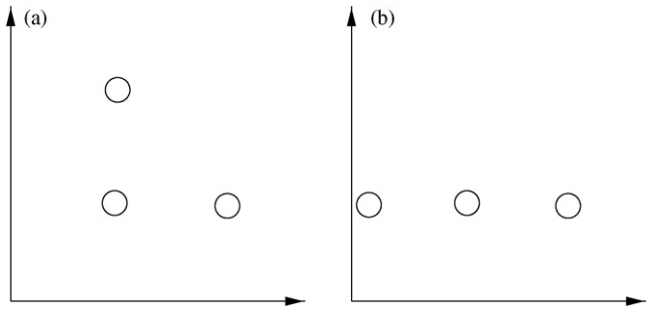

Fig. 2. (a) Particle arrangement for a regular moment matrix for a linear complete MLS basis. (b) Particle arrangement for a singular moment matrix for a linear complete MLS basis.

or, for the shape function Φ_I associated with node I at a point x:

$$\Phi_I(x) = \mathbf{p}^{\mathrm{T}}(x)[\mathbf{A}(x)]^{-1} w(x - x_I)\mathbf{p}(x_I) \tag{31}$$

The matrix $\mathbf{A}(x)$ is often called moment matrix, it is of size $m \times m$. This matrix must be inverted wherever the MLS shape functions are to be evaluated. Obviously, this fact is one drawback of MLS-based MMs because of computational cost and the possibility that this moment matrix may be singular.

Consider a linear basis in one dimension, the moment matrix then becomes:

$$\mathbf{A}(x) = w(x - x_1)\begin{bmatrix} 1 & x_1 \\ x_1 & x_1^2 \end{bmatrix} + w(x - x_2)\begin{bmatrix} 1 & x_2 \\ x_2 & x_2^2 \end{bmatrix} + \cdots + w(x - x_n)\begin{bmatrix} 1 & x_n \\ x_n & x_n^2 \end{bmatrix} \tag{32}$$

It is clear from this equation that if $n = 1$, i.e. point x is covered by only one nodal support while the basis is linear ($m = 2$), then the matrix is singular and cannot be inverted. Therefore, a necessary condition for the moment matrix to be invertible is that $n \geq m$. Note also that if $n = m$, the nodes have to be arranged in different coordinate directions, otherwise the matrix will be singular as well, see Fig. 2.

It can be seen graphically that the MLS shape functions are indeed a partition of unity in Fig. 3 in 1D. Consider an interval $0 \leq x \leq 4$ divided into four (4) equal domains. The weight and shape functions of all five (5) nodes are plotted in Fig. 3. The functions associated with the centre node are represented by a *heavy* line. In this example, the weight function is the quartic spline, the size of all five (5) domains of influence is 2.5. Remark that the MLS shape

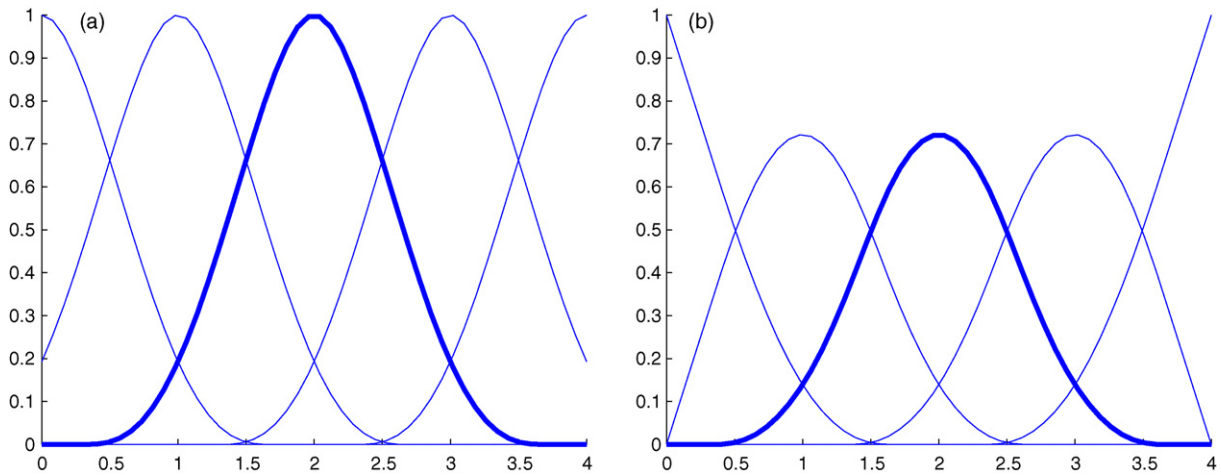

Fig. 3. Weight and shape function of the central node: (a) weight function and (b) shape function.

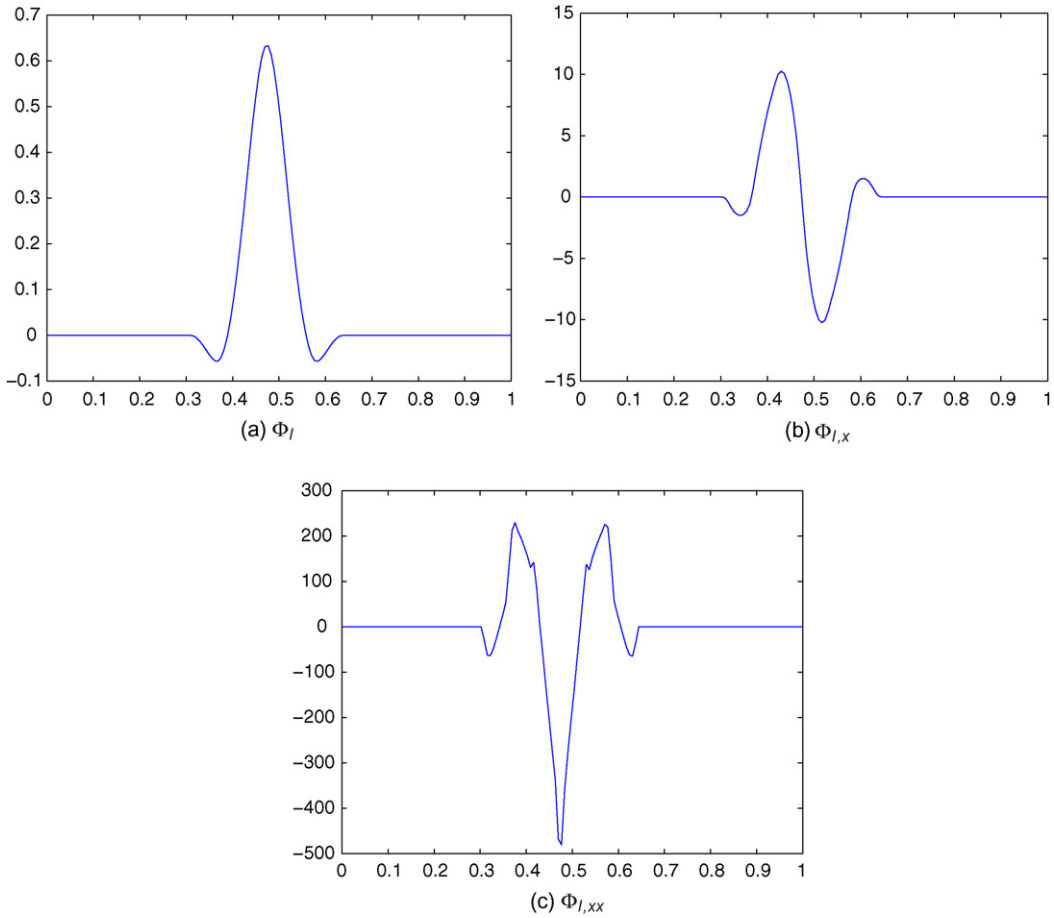

Fig. 4. MLS shape functions and derivatives with quadratic basis.

functions do not satisfy the Kronecker delta property.[3] The derivatives of the MLS shape functions are given in Fig. 4. To get smooth graphs, we computed these derivatives at 150 sampling points on the interval $0 \leq x \leq 1$. An important property of the first derivatives can be observed from this figure: the first derivative of node I vanishes at this node. This makes MLS-collocation–MMs unstable.

For two and three dimensions, x becomes vector $\mathbf{x}$ and the basis $\mathbf{p}(\mathbf{x})$ is given by (only for two dimensions):

- Linear basis:

$$\mathbf{p}^{\mathrm{T}}(\mathbf{x}) = [1 \quad x \quad y] \tag{33}$$

- Quadratic basis:

$$\mathbf{p}^{\mathrm{T}}(\mathbf{x}) = [1 \quad x \quad y \quad x^2 \quad y^2 \quad xy] \tag{34}$$

[3] The shape function associated with a node is not exactly equal to one at this node (in the present case, it is about 0.7, for the centre node), and this shape function is not exactly zero at the other nodes in the domain.

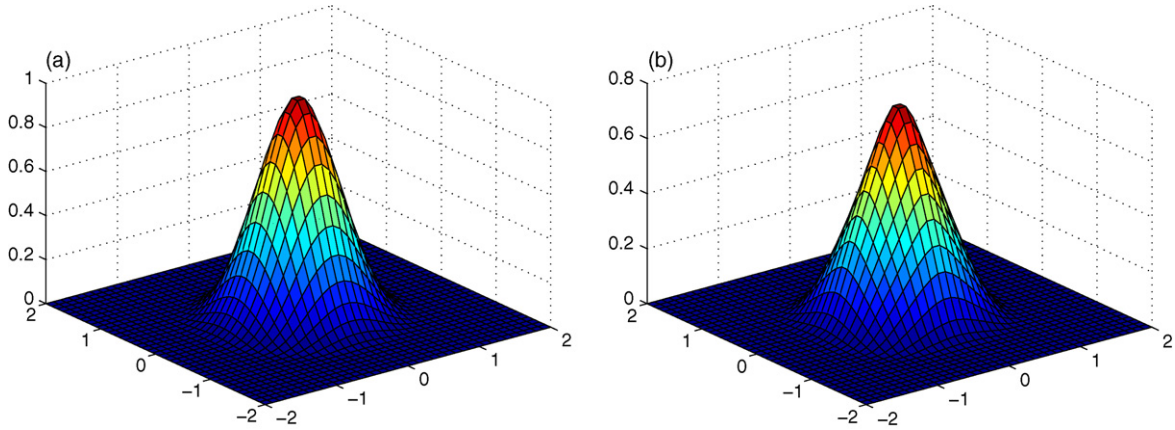

Fig. 5. Weight and MLS shape function: (a) weight function and (b) shape function.

If $\mathbf{p}(\mathbf{x})$ is chosen to be a zeroth order basis, i.e. $\mathbf{p}(\mathbf{x}) = 1$, then the resulting MLS shape function is given by

$$\Phi_I^0(\mathbf{x}) = \frac{w(\mathbf{x} - \mathbf{x}_I)}{\sum_I^n w(\mathbf{x} - \mathbf{x}_I)} \tag{35}$$

which is known as the Shepard function, the lowest order form of MLS shape functions. Note that the basis $\mathbf{p}$ is often shifted by $(\mathbf{x} - \mathbf{x}_J)/d_I$ since the shifting improves the conditioning of the moment matrix. A two-dimensional graphical representation of the quartic spline with the corresponding linear complete MLS shape function is shown in Fig. 5. The first spatial derivatives of the shape functions in the x- and y- directions are depicted in Fig. 6.

2.6. Extrinsic meshless methods

2.6.1. The partition of unity finite element method

In [78], a method called partition of unity finite element method (PUFEM) was developed. The approximation in the PUFEM is given by

$$u^h(\mathbf{x}) = \sum_{I=1}^{N} \phi_I^0(\mathbf{x}) \sum_{j=1}^{l} p_j(\mathbf{x}) v_{jI} = \sum_{I}^{N} \phi_I^0(\mathbf{x}) \mathbf{p}^{\mathrm{T}}(\mathbf{x}) \mathbf{v}_I \tag{36}$$

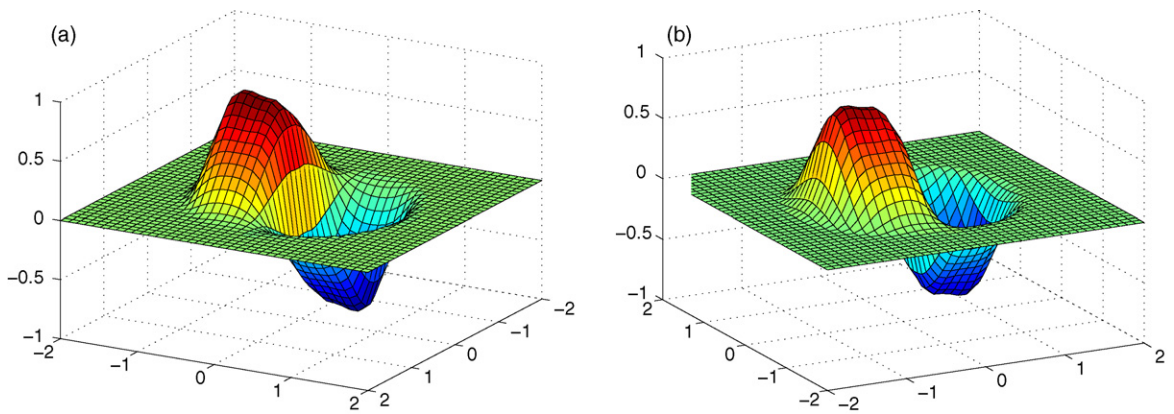

Fig. 6. MLS shape function first derivatives: (a) Φ_x and (b) Φ_y.

where $\phi^0(x)$ are usually shape functions based on Lagrange polynomials. The coefficients v_{jI} are nodal unknowns. The attractive property of the approximation is that it is the number of terms x^k which dictates the order of completeness of the approximation. Another useful property of this approximation is that, special enhancement functions, usually a known feature of the sought solution, are easily incorporated into the approximation through this extrinsic basis.

By examining Eq. (19), we see that in MLS approximations, the basis $\mathbf{p}(\mathbf{x})$, and hence the order of consistency, cannot be varied from node to node without introducing a discontinuity in the approximation. This means that *p*-adaptivity is not naturally obtained by intrinsic enrichment. Regions with different order of consistency may be obtained, but need to be blended together, to acertain continuity between the regions.

2.6.2. hp-clouds

The approximation in the *hp* clouds method [40] writes, at any point $\mathbf{x} \in \Omega$:

$$u^h(\mathbf{x}) = \sum_I^N \phi_I(\mathbf{x}) \left(u_I + \sum_j^l p_j(\mathbf{x}) v_{jI} \right) \tag{37}$$

where the p_j form the so-called *extrinsic basis* since it contains both high-order monomials and enhancement functions as well. *Enhancement functions* or *enrichment functions* are usually introduced into the approximation space to capture special properties such as discontinuities, singularities, boundary layers, or other relevant features of a solution.

Different partitions of unity can be used for the standard and enhanced/enriched parts of the approximation [91]:

$$u^h(\mathbf{x}) = \sum_I^N \phi_I^k(\mathbf{x}) u_I + \sum_I^M \phi_I^m(\mathbf{x}) \sum_j^l p_j(\mathbf{x}) v_{jI} \tag{38}$$

where $\phi_I^k(\mathbf{x})$ and $\phi_I^m(\mathbf{x})$ are meshless shape functions of order k and m, respectively.

2.6.3. A simple example with extrinsic global enrichment

To show the power of enrichment, consider a one-dimensional problem featuring large localized gradients, as shown below:

$$u_{,xx}(x) + b(x) = 0, \quad x \in [0, 1]; \qquad u(0) = 0, \qquad u(1) = 1 \tag{39}$$

with

$$b(x) = \begin{cases} \{2\alpha^2 - 4[\alpha^2(x - 0.5)]^2\} \exp\{-[\alpha(x - 0.5)]^2\}, & x \in [0.42, 0.58] \\ 0, & \text{otherwise} \end{cases} \tag{40}$$

The exact solution of this problem is

$$u(x) = x + \exp\{-[\alpha(x - 0.5)]^2\} x \in [0, 1] \tag{41}$$

At first, the standard EFG method with linear-complete shape functions, a discretization of 30 evenly-spaced particles (29 intervals) and four Gauss points in each interval is employed. The numerical displacement is compared to the exact displacement in Fig. 7(a). It is worth noting that the numerical solution cannot capture the local character of the exact solution (around $x = 0.5$). In order to obtain acceptable results, the discretization must be heavily refined in the neighbourhood of the large gradients, otherwise, spurious oscillations appear (Fig. 7(b)).

In order to capture the local character, the exact solution can be incorporated into the meshless approximation. For such a simple one-dimensional problem, one can choose *global* enrichment:

$$u^h(x) = \sum_I \Phi_I(x) u_I + b\Psi(x) \quad \text{with } \Psi(x) = \exp\{-[\alpha(x - 0.5)]^2\} \tag{42}$$

The result obtained with this approximation (30 uniform nodes and 4 Gauss points for each of 29 subcells) is given in Fig. 7(c) with excellent agreement between numerical and exact solution.

The *global* enrichment strategy has the advantage that only one additional unknown is added for each special function to be added. It has the drawbacks that (1) the enrichment function must have local character, i.e. have a

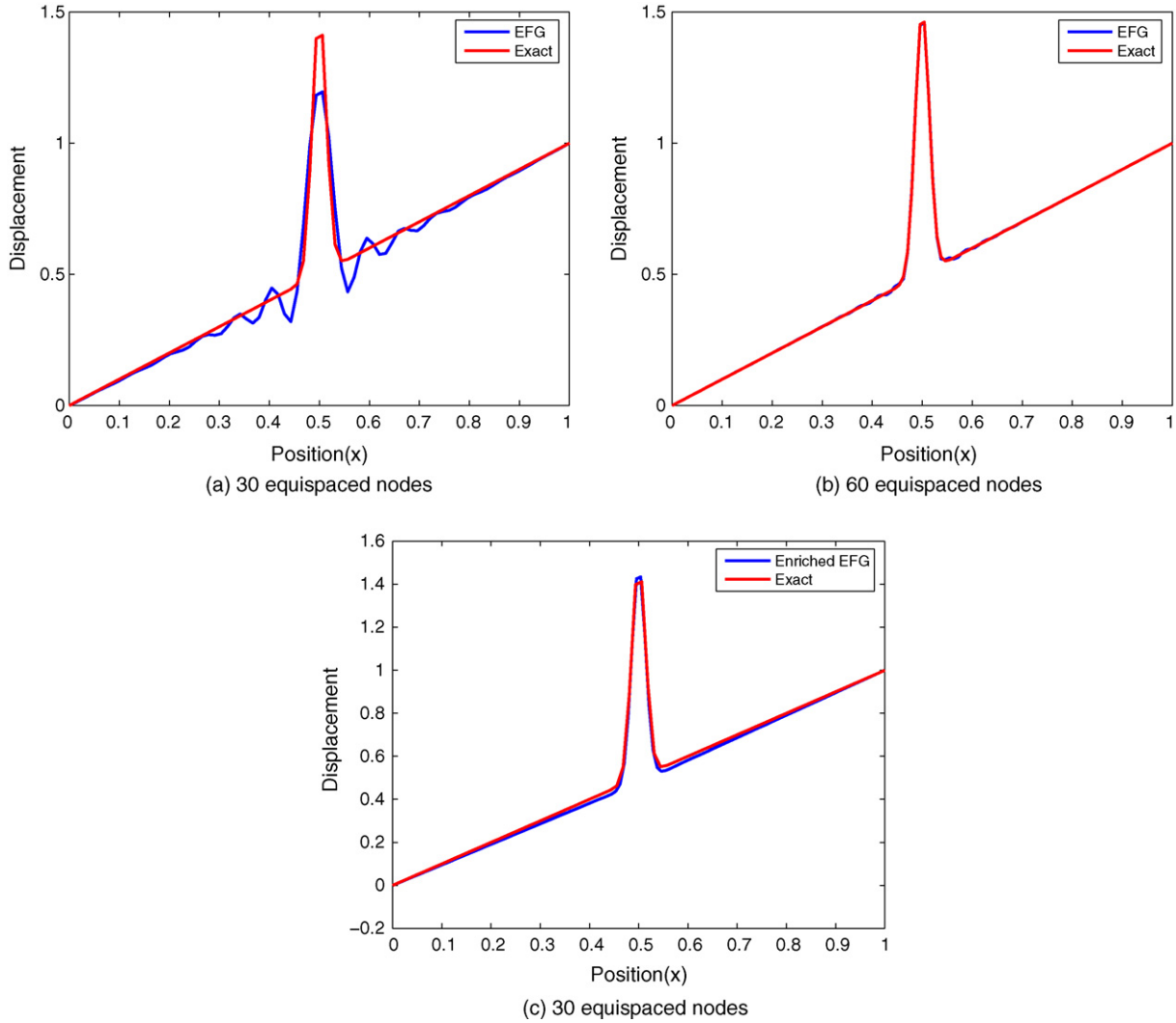

Fig. 7. One-dimensional problem with localized solution: comparison between EFG and enriched EFG solutions.

compact support "small" relative to the domain size, to ensure that the left hand side matrix remains banded; (2) the discrete equations are modified, which complicated the implementation into existing codes. The local (extrinsic) PU enriched formulation is given by

$$u^h(x) = \sum_{I \in \mathcal{S}} \Phi_I(x) u_I + \sum_{J \in \mathcal{S}^c} \Phi_J(x) \Psi(x) a_J \tag{43}$$

where $\mathcal{S}^c$ is the set of nodes whose supports contain the point $x = 0.5$. The displacements obtained with global enrichment, PU-enrichment and the exact solution are plotted in Fig. 8(a). In all computations, the cubic spline with circular support and radius $\bar{r} = s\,\Delta x$ with $s = 2.5$, Δx is the nodal spacing, is employed.

It is obvious that, the number of enriched nodes changes when the size of nodal supports varies. Precisely, when s increases, the number of enriched nodes increases, hence increase the number of problem unknowns. Therefore, choosing a proper value for the support size is necessary in both computational cost and accuracy. Fig. 8(b) shows the results obtained with various support sizes.

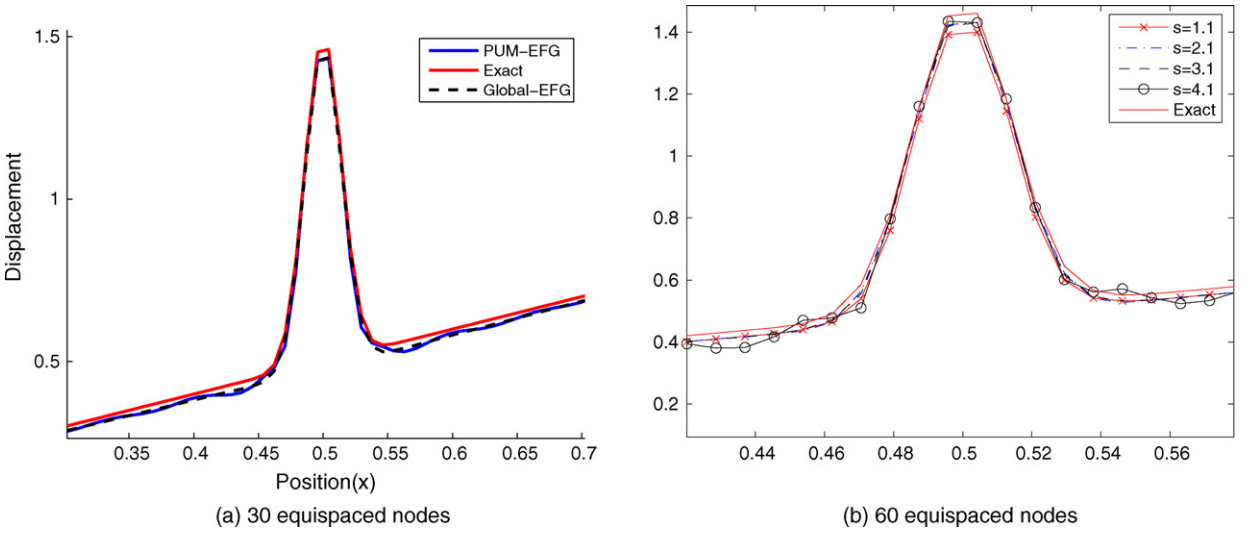

Fig. 8. Comparison of enrichment strategies and effects of nodal support size.

2.7. Weighted residual methods

Considering a partial differential equation on a domain Ω with boundary Γ, defined by the differential operator $\mathcal{L}$: $u \mapsto \mathcal{L}u$ and the linear form f: $\Omega \to \mathbb{R}$:

$$\mathcal{L}u(\mathbf{x}) = f(\mathbf{x}) \quad \text{in } \Omega \tag{44a}$$

$$u = \bar{u} \quad \text{on } \Gamma \tag{44b}$$

One of the most general techniques to solve such an equation numerically is the weighted residual method. In this method, the unknown field u is approximated by trial functions $\mathbf{\Phi}$ and nodal parameters $\mathbf{u}$ in the form $u \approx u^h = \mathbf{\Phi}^{\mathrm{T}}\mathbf{u}$. Replacing u with u^h in the PDE gives:

$$\forall \boldsymbol{x} \in \Omega, \quad \varepsilon^h(\boldsymbol{x}) = \mathcal{L}u^h(\mathbf{x}) - f(\mathbf{x}) \tag{45}$$

where ε^h is the residual error, which is non-zero, since an approximation function, living in a function space of finite size, cannot fulfill the original equation exactly everywhere in Ω.

A set of test functions $\mathbf{\Psi}$ are chosen and the system of equations is determined by setting ε^h orthogonal[4] to this set of test functions:

$$\int_\Omega \mathbf{\Psi}\varepsilon^h \,\mathrm{d}\Omega = 0 \quad \text{or} \quad \int_\Omega \mathbf{\Psi}(\mathcal{L}u^h(\mathbf{x}) - f(\mathbf{x}))\,\mathrm{d}\Omega = 0 \tag{46}$$

$$\int_\Omega \mathbf{\Psi}\left[\mathcal{L}\left(\sum_{I=1}^{N}\Phi_I(x)u_I - f(\mathbf{x})\right)\right]\,\mathrm{d}\Omega = 0 \tag{47}$$

In the above equations, it was implicitly assumed that integrals are capable of being evaluated. This places certain restrictions on the families to which functions Ψ and Φ must belong. In general, if n th order derivatives occur in the operator $\mathcal{L}$, then the trial and test functions must be C_{n-1} ($n-1$ continuous derivatives). Usually, integration by parts is applied in Eq. (47) to lower the order of derivation, decreasing the order of continuity required for the test and trial spaces. The form of the partial differential equation is called the *weak form* associated with the *strong form* given in Eq. (44).

[4] In the sense of the inner product $\langle u, v\rangle = \int_\Omega uv\,\mathrm{d}\Omega$.

In order to obtain the discrete equations, the unknown function $u(\mathbf{x})$ and the test function $\boldsymbol{\Psi}$ are approximated by

$$u^h(\mathbf{x}) = \sum_{I=1}^{N} \Phi_I(\mathbf{x}) u_I \quad \text{and} \quad \boldsymbol{\Psi}(\mathbf{x}) = \sum_{I=1}^{N} \Psi_I(\mathbf{x}) \delta u_I \tag{48}$$

where δu_I are arbitrary coefficients, and u_I are unknowns of the problem.

The choice of the functions $\Psi_I(\mathbf{x})$ leading to different methods such as collocation and Galerkin methods which are described in the next section.

2.7.1. Collocation method

Assume the x_I to denote the set of points in the computational domain, in the collocation method, the test functions $\boldsymbol{\Psi}$ are chosen to be Dirac delta distributions $\delta(x - x_I)$. Because of the *sifting property* of the Dirac delta distributions, the weak form, Eq. (47), reduces to the strong form, evaluated at all the nodes in the domain. The discrete equation can be written as

$$\mathcal{L}u^h(\mathbf{x}_I) = f(\mathbf{x}_I), \quad I \in \Omega - \Gamma \tag{49a}$$

$$u(\mathbf{x}_I) = \bar{u}(\mathbf{x}_I), \quad I \in \Gamma \tag{49b}$$

The above is a set of algebraic equations whose unknowns are u_I.

The collocation method has two major advantages, namely (i) efficiency in constructing the final system of equations since no integration is required and (ii) shape functions are only evaluated at nodes rather than at integration points as in other methods. The price to pay is that, one must evaluate high-order derivatives of MMs shape functions which is quite burdensome. In addition, two other drawbacks are difficulties in imposing natural boundary conditions and non-symmetric stiffness matrix.

To better illustrate the method, consider the problem of a string on an elastic foundation with the governing equations:

$$-a\frac{\mathrm{d}^2 u}{\mathrm{d}x^2}(x) + cu(x) + f = 0, \quad 0 < x < 1; \qquad u(0) = u(1) = 0 \tag{50}$$

with specific parameters for the solution are chosen $a = 0.01, c = 1$ and $f = -1$. The domain is divided into an equally spaced set of nodes located at x_J, $J = 1, \ldots, N$ where the boundary points are nodes x_1 and x_N. By imposing the equations given in Eq. (50) at the N nodes, we obtain the following equations:

$$-a\frac{\mathrm{d}^2}{\mathrm{d}x^2}\left(\sum_{I=1}^{N} \Phi_I(x_J) u_I\right) + c\left(\sum_{I=1}^{N} \Phi_I(x_J) u_I\right) + f = 0, \quad J = 2, \ldots, N-1 \tag{51a}$$

$$\sum_{I=1}^{N} \Phi_I(x_1) u_I = 0, \qquad \sum_{I=1}^{N} \Phi_I(x_N) u_I = 0 \tag{51b}$$

The Eq. (51a) is rewritten in the familiar form:

$$\left(-a\sum_{I=1}^{N} \Phi_{I,xx}(x_J) + c\sum_{I=1}^{N} \Phi_I(x_J)\right) u_I + f = 0, \quad J = 2, \ldots, N-1 \tag{52}$$

which is of the familiar form $\mathbf{Ku} = \mathbf{f}$ where the assembly procedure is performed by looping on separate sets of nodes (herein, there are interior and essential boundary nodes).

It is worth noting that, using the point collocation method, one must deal with high-order derivatives (here second order).[5] Hence the meshless shape functions must have at least continuous second-order derivatives, which is the case if the kernel (weight) function is C_2 continuous. The numerical solution obtained with the point collocation method is given in Fig. 9.

[5] The second derivatives of MLS shape functions are given in Section 3.2.

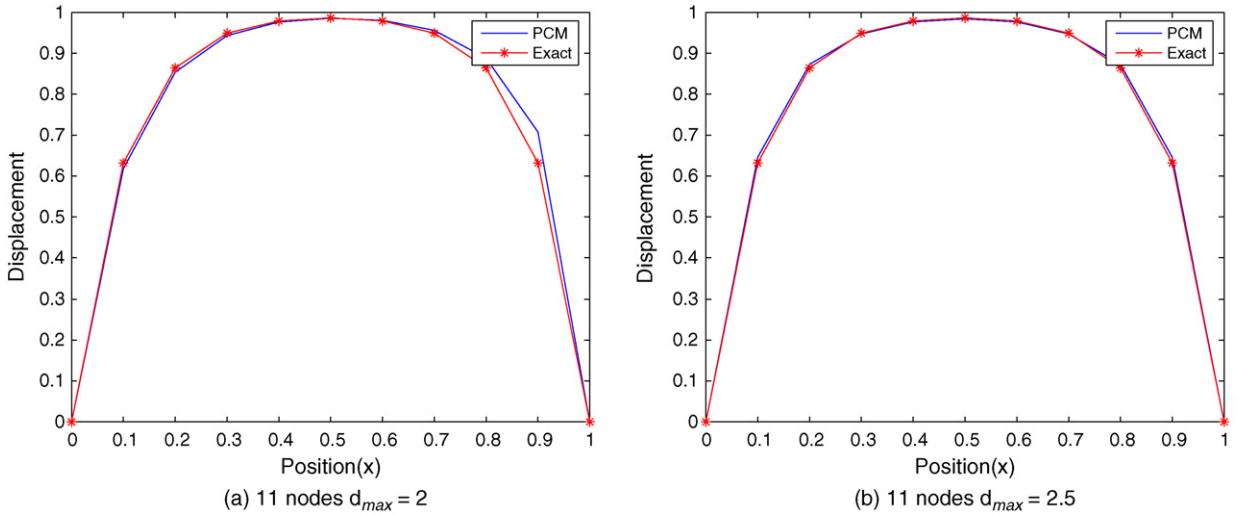

Fig. 9. String on elastic foundation: point collocation solutions.

2.7.2. Galerkin methods

The trial and test functions in Galerkin methods are given by

$$u^h(\mathbf{x}) = \sum_{I=1}^{N} \Phi_I(\mathbf{x}) u_I, \qquad \delta u^h(\mathbf{x}) = \sum_{I=1}^{N} \Psi_I(\mathbf{x}) \delta u_I \tag{53}$$

If different shape functions are used for the approximation of the test and trial functions, a Petrov–Galerkin method is obtained, otherwise we have a Bubnov–Galerkin method.[6] We will assume now that $\Psi_I = \Phi_I$ though all derivations apply also for a Petrov–Galerkin method.

As an example, the problem of a string on an (using the divergence theorem – integration by parts – in Eq. (50)) elastic foundation is solved again, but now with a Galerkin-based meshless methods. The weak form of this problem is

$$a \int_0^1 v_x u_x \, \mathrm{d}x + c \int_0^1 v u \, \mathrm{d}x + f \int_0^1 v \, \mathrm{d}x = 0 \tag{54}$$

where v is the test function. The discrete equations are obtained by substituting the approximations of u and v into the above:

$$\left(a \int_0^1 \Phi_{I,x} \Phi_{J,x} \, \mathrm{d}x + c \int_0^1 \Phi_I \Phi_J \, \mathrm{d}x \right) u_J + f \int_0^1 \Phi_I \, \mathrm{d}x = 0 \tag{55}$$

The above has the familiar matrix form $\mathbf{Ku} = \mathbf{f}$ where

$$\mathbf{K}_{IJ} = \int_0^1 \left(a \Phi_{I,x} \Phi_{J,x} + c \Phi_I \Phi_J \right) \mathrm{d}x, \qquad \mathbf{f}_I = -f \int_0^1 \Phi_I \, \mathrm{d}x \tag{56}$$

The exact solution of this problem is given by

$$u(x) = 1 - \cosh(mx) - (1 - \cosh(m)) \frac{\sinh(mx)}{\sinh(m)}, \quad m = \left(\frac{c}{a} \right)^{1/2} \tag{57}$$

The numerical solutions obtained with the element free Galerkin method are given in Fig. 10.

[6] Often called Galerkin method.

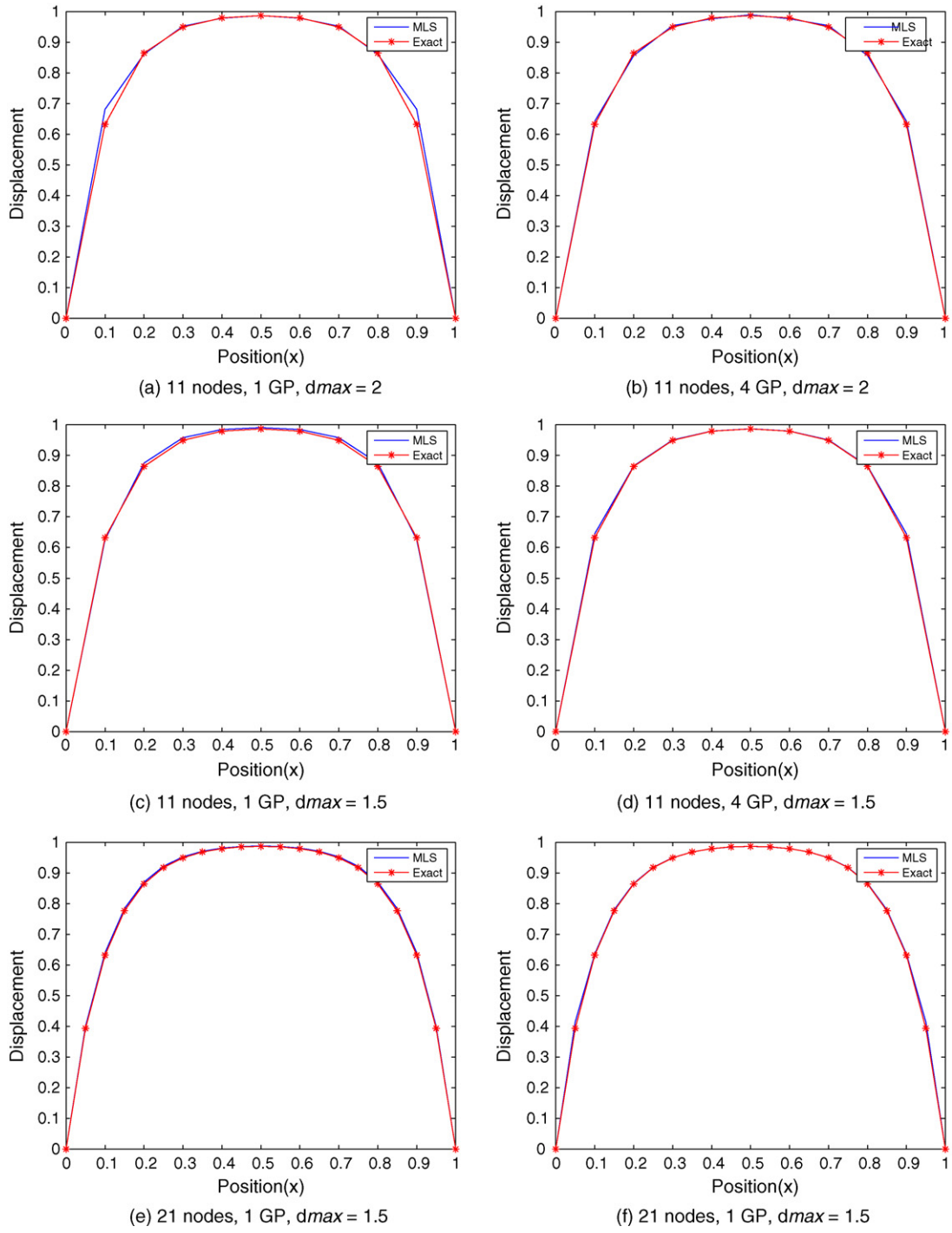

Fig. 10. String on elastic foundation: EFG solutions.

2.8. Discrete equations for elastostatics

Consider a domain Ω, bounded by Γ. The boundary is partitioned into two sets: Γ_u and Γ_t. Displacements are prescribed on Γ_u whereas tractions are prescribed on Γ_t. The weak form of linear elastostatics problems is to find **u** in

the trial space,[7] such that for all test functions $\delta\mathbf{u}$ in the test space[8]:

$$\int_{\Omega} \boldsymbol{\varepsilon}(\mathbf{u}) : \mathbf{C} : \boldsymbol{\varepsilon}(\delta\mathbf{u})\,\mathrm{d}\Omega = \int_{\Gamma} \bar{\mathbf{t}} \cdot \delta\mathbf{u}\,\mathrm{d}\Gamma + \int_{\Omega} \mathbf{b} \cdot \mathbf{v}\,\mathrm{d}\Omega \tag{58}$$

Substitution of approximations for $\mathbf{u}$ and $\delta\mathbf{u}$ into the above gives the discrete equations:

$$\mathbf{K}\mathbf{u} = \mathbf{f} \tag{59}$$

with

$$\mathbf{K}_{IJ} = \int_{\Omega} \mathbf{B}_I^{\mathrm{T}} \mathbf{C} \mathbf{B}_J\,\mathrm{d}\Omega, \qquad \mathbf{f}_I = \int_{\Gamma_t} \Phi_I \bar{\boldsymbol{t}}\mathrm{d}\Gamma + \int_{\Omega} \Phi_I \mathbf{b}\,\mathrm{d}\Omega \tag{60}$$

In two dimensions, the $\mathbf{B}$ matrix is given by

$$\mathbf{B}_I = \begin{bmatrix} \Phi_{I,x} & 0 \\ 0 & \Phi_{I,y} \\ \Phi_{I,y} & \Phi_{I,x} \end{bmatrix} \tag{61}$$

Note that we have omitted Dirichlet boundary conditions in our formulations. The incorporation of Dirichlet boundary conditions will be discussed in the next section. Note also that if an extrinsic basis is used the nodal vector $\mathbf{u}$ will contain additional unknowns, see Section 2.6. Different methods can now be constructed by using different shape functions. If we choose Dirac delta functions for the test function, we have a collocation method. Otherwise we obtain a Galerkin method.

2.8.1. Integration

The major disadvantage of MMs using Galerkin method is the numerical integration of the weak form. This is due to the non-polynomial (rational) form of most meshless shape functions (MLS for instance). So, exact integration is difficult to impossible for most meshfree methods. The most frequently used techniques include:

Direct nodal integration. The integrals are evaluated only at the nodes that also serve as integration points:

$$\int_{\Omega} f(\mathbf{X})\,\mathrm{d}\Omega = \sum_{J \in \mathcal{S}} f(\mathbf{X}_J) V_J \tag{62}$$

The quadrature weights V_J are usually volume associated with the node. The volume is obtained from a Voronoi diagram that is constructed at the beginning of the computation. This approach is more efficient than using full integration. However, nodal integration leads to instabilities due to rank deficiency similar to reduced integrated finite elements. We would also like to remark that nodal integrated meshless methods are very similar to meshless collocation methods [13,5,10].

Stabilized nodal integration. Chen et al. [31] proposed the stabilized confirming nodal integration using strain smoothing. They recognized that the vanishing derivatives of the meshfree shape functions at the particles cause of the instabilities. In their strain smoothing procedure, the nodal strains are computed as the divergence of a spatial average of the strain field. The strain smoothing avoids evaluating derivatives of the shape functions at the nodes and hence eliminates defective modes. An excellent overview of different methods to stabilize nodal integration is given by Puso et al. [85]. Recently, the smoothed Finite Element Method (SFEM) was introduced, by coupling this stabilized conforming nodal integration to finite elements, resulting in a higher stress accuracy, insensitivity to volumetric locking, superconvergence, at the cost of stability (in some instances). The interested reader is referred to the review paper [109] and the contributions in [108,110,111,112].

[7] Contains C^0 functions.

[8] Contains C^0 functions but vanishes on Γ_u.

Stress point integration. Adding additional stress points to the nodes is another possibility to avoid instabilities due to rank deficiency:

$$\int_{\Omega} f(\mathbf{X})\,\mathrm{d}\Omega = \sum_{J\in\mathcal{S}^N} f(\mathbf{X}_J)V_J^N + \sum_{J\in\mathcal{S}^S} f(\mathbf{X}_J)V_J^S \tag{63}$$

where the superimposed N denote nodes and the superimposed S denote stress points. Note that all kinematic values are obtained via the nodes and only stresses are evaluated at these stress points. This concept of stress points was first introduced in an SPH setting in one dimension by Dyka and Ingel [46] and later on extended into higher-order dimensions by Randles and Libersky [89] and Belytschko et al. [9]. Note that there is a subtle difference between the stress point integration of Randles and Libersky [89] and Belytschko et al. [9]. While Randles and Libersky [89] evaluate stresses only at the stress point, Belytschko et al. [9] evaluate stresses also at the nodes. A slightly different approach was proposed by Cueto-Felgueroso et al. [34]. For large deformations, rules have to be found to move the stress points.

Support-based integration. In the method of finite spheres, the integration is performed on every intersections of overlapping supports. A truly meshfree method for integrating the weak form over overlapping supports, related to the supports of the meshfree approximation was developed independently by Duflot and Nguyen-Dang [43](called moving least square quadrature) and Carpinteri et al. [27](called partition of unity quadrature). This integration technique is improved in Carpinteri et al. [28] and Zhang et al. [103] to take cracks into account.

Background mesh or cell structure. The domain is divided into integration cells over which Gaussian quadrature is performed:

$$\int_{\Omega} f(\mathbf{X})\,\mathrm{d}\Omega = \sum_{J} f(\boldsymbol{\xi}_J) w_J \det J^{\boldsymbol{\xi}}(\boldsymbol{\xi}) \tag{64}$$

where $\boldsymbol{\xi}$ are local coordinates and $\det J^{\boldsymbol{\xi}}(\boldsymbol{\xi})$ is the determinant of the Jacobian, i.e. the mapping from the parent into the physical domain. If a background mesh is present, nodes and the vertices of the integration usually coincide (as in conventional FEM meshes, Fig. 11). When cell structures are utilized, a regular array of domains is created, independently of the particle position [38].

MMs which are based on local weak forms such as MLPG adopt integration over the shape function supports or intersection of supports. Interested readers should refer to [2–4,70] and references therein for details.

Methods based on nodal and stress point integration are frequently employed in dynamics and where large deformations are expected. We will consider only methods that employ Gauss quadrature and utilize a background mesh. These methods are more accurate and they are ideally applicable to small and moderate deformation.

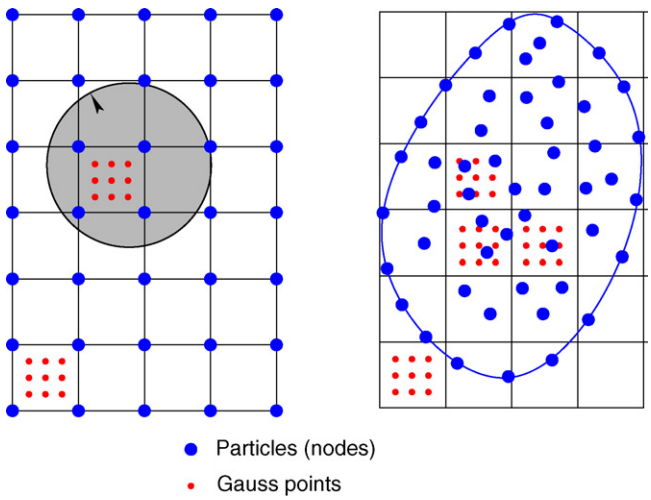

Fig. 11. Integration in Galerkin-based MMs: background mesh (left) and background structure cells (right).

2.8.2. Essential boundary conditions

Due to the lack of the Kronecker delta property of MMs shape functions, the essential boundary conditions cannot be imposed as easily as in FEM. Several techniques have been proposed, namely (i) methods based on the modification of the weak form and (ii) methods using modified shape functions. This section gives a brief description of these methods, for more details, one should refer to [57].

Methods based on the modification of the weak form includes the Lagrange multiplier method, the penalty method and Nitsche's method. In order to understand these methods, the so-called variational principle should be first presented.

A variational principle specifies a scalar quantity, named functional Π, which is defined by an integral form:

$$\Pi = \int_{\Omega} F(\mathbf{u}, \mathbf{u}_x, \ldots)\,\mathrm{d}\Omega + \int_{\Gamma} E(\mathbf{u}, \mathbf{u}_x, \ldots)\,\mathrm{d}\Gamma \tag{65}$$

where $\mathbf{u}$ is the unknown function, F and E are differential operators. The solution to the continuum problem is a function $\mathbf{u}$ which makes Π stationary with any arbitrary variations $\delta\mathbf{u}$:

$$\delta\Pi = 0 \quad \text{with any } \delta\mathbf{u} \tag{66}$$

2.8.2.1. Lagrange multipliers. Let us consider a general problem of making a functional Π stationary with constraints:

$$\mathbf{C}(\mathbf{u}) = 0 \quad \text{on } \Gamma \tag{67}$$

To satisfy the above constraint, we build the following functional:

$$\bar{\Pi}(\mathbf{u}, \boldsymbol{\lambda}) = \Pi(\mathbf{u}) + \int_{\Gamma} \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{C}(\mathbf{u})\,\mathrm{d}\Gamma \tag{68}$$

The variation of this new functional is given by

$$\delta\bar{\Pi} = \delta\Pi + \int_{\Gamma} \delta\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{C}(\mathbf{u})\,\mathrm{d}\Gamma + \int_{\Gamma} \boldsymbol{\lambda}^{\mathrm{T}}\delta\mathbf{C}(\mathbf{u})\,\mathrm{d}\Gamma \tag{69}$$

In order to derive the discrete equations, the Lagrange multipliers must be approximated (l is the number of shape functions required to approximate the multipliers on the boundary, e.g. If two-noded finite elements are used, $l = 2$):

$$\boldsymbol{\lambda}(\boldsymbol{x}) = \sum_{I=1}^{l} N_I^{\mathrm{L}}(\mathbf{x})\lambda_I \tag{70}$$

There are several choices for the approximation space for the Lagrange multipliers, i.e. choices of $N_I^{\mathrm{L}}(\mathbf{x})$, namely, (i) finite element interpolation on the boundary Γ, (ii) meshless approximations on this boundary and (iii) the point collocation method which uses the Dirac delta function:

$$N_I^{\mathrm{L}}(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_I^{\mathrm{L}}) \tag{71}$$

where $\mathbf{x}_I^{\mathrm{L}}$ is a set of points locating along the boundary Γ. Using this method, the system of equations of elastostatics is given by

$$\begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{G}^{\mathrm{T}} & 0 \end{bmatrix} \begin{Bmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{f} \\ \mathbf{q} \end{Bmatrix} \tag{72}$$

with

$$\mathbf{G}_{IK} = -\int_{\Gamma_u} \Phi_I \mathbf{N}_K\,\mathrm{d}\Gamma = -\begin{bmatrix} \Phi_I(\mathbf{x}_K) & 0 \\ 0 & \Phi_I(\mathbf{x}_K) \end{bmatrix} \tag{73}$$

$$\mathbf{q}_K = -\int_{\Gamma_u} \mathbf{N}_K \bar{\mathbf{u}}\,\mathrm{d}\Gamma = -\bar{\mathbf{u}}(\mathbf{x}_K) \tag{74}$$

It is obvious that one drawback of the Lagrange multiplier method is the introduction of additional unknowns to the problem. In addition, from Eq. (72), there are now zero terms on the diagonal of the matrix which makes the matrix no longer positive definite.

2.8.2.2. Penalty function. We have the functional for the problem given in the preceding sections:

$$\bar{\Pi}(\mathbf{u},\alpha) = \Pi(\mathbf{u}) + \frac{\alpha}{2}\int_{\Gamma} \mathbf{C}(\mathbf{u})^{\mathrm{T}}\mathbf{C}(\mathbf{u})\,\mathrm{d}\Gamma \tag{75}$$

Applying the penalty method to elastostatics, we obtain the following weak form:

$$\int_{\Omega} \boldsymbol{\varepsilon}^{\mathrm{T}}(\mathbf{u}) : \mathbf{C} : \boldsymbol{\varepsilon}(\mathbf{v})\,\mathrm{d}\Omega = \int_{\Gamma} \bar{\mathbf{t}}\cdot\mathbf{v}\,\mathrm{d}\Gamma + \int_{\Omega} \mathbf{b}\cdot\mathbf{v}\,\mathrm{d}\Omega + \alpha\int_{\Gamma_u} \mathbf{u}\cdot\mathbf{v}\,\mathrm{d}\Gamma - \alpha\int_{\Gamma_u} \bar{\mathbf{u}}\cdot\mathbf{v}\,\mathrm{d}\Gamma \tag{76}$$

which gives the equation $\mathbf{K}\mathbf{u} = \mathbf{f}$, where

$$\mathbf{K}_{IJ} = \int_{\Omega} \mathbf{B}_I^{\mathrm{T}}\mathbf{C}\mathbf{B}_J\,\mathrm{d}\Omega - \alpha\int_{\Gamma_u} \Phi_I\Phi_J\,\mathrm{d}\Gamma \tag{77}$$

$$\mathbf{f}_I = \int_{\Gamma_t} \Phi_I\bar{\mathbf{t}}\,\mathrm{d}\Gamma + \int_{\Omega} \Phi_I\mathbf{b}\,\mathrm{d}\Omega - \alpha\int_{\Gamma_u} \Phi_I\bar{\mathbf{u}}\,\mathrm{d}\Gamma \tag{78}$$

The main advantage of the penalty method compared with the Lagrange multiplier approach is that no additional unknowns are required. However, the conditioning of the matrix much depends on the choice of the penalty parameter. What is more, in the penalty method, the constraints are only satisfied *approximately*.

Recently, the augmented Lagrangian method has been proposed by Ventura [98] to handle essential boundary conditions in meshfree methods. This method has been shown to be stable and effective, particularly in contact problems where it has replaced the penalty and Lagrangian multipliers methods.

2.9. Discontinuities

There are mainly four approaches to model discontinuities in meshless methods, namely (i) modification of the weight function such as the visibility method, the diffraction method and the transparency method [11,63,84], (ii) modification of the intrinsic basis [50] to incorporate special functions, (iii) methods based on an extrinsic MLS enrichment [50] and (iv) methods based on the extrinsic PUM enrichment [99,113–118]. More recently, the augmented Lagrangian method has been used to model strong discontinuity (crack problems) in Carpinteri [25] and weak discontinuity (material discontinuity) in Carpinteri [26].

2.9.1. Modification of weight function

The visibility method [8,15] was the first method to incorporate strong discontinuities into meshless methods. In the visibility method, the crack boundary is considered to be opaque. Nodes that are on the opposite side of the crack are excluded in the approximation of the displacement field. Difficulties arise for particles close to the crack tip since undesired interior discontinuities occur, see Fig. 12. Non-convex boundaries cannot be treated by the visibility criterion correctly either.

The diffraction method [84] is an improvement of the visibility method. It removes the *undesired* interior discontinuities as shown in Fig. 13. The diffraction method is also suitable for non-convex crack boundaries. The method is motivated by the way light diffracts around a sharp corner but the equations used in constructing the domain of influence and the weight function bear almost no relationship to the equation of diffraction. The method is only applicable to radial basis kernel functions with a single parameter. The idea of the diffraction method is to treat the crack as opaque but to evaluate the length of the ray by a path which passes around the corner of the discontinuity, see Fig. 13. It should be noted that the shape function of the diffraction method is quite complex with several areas of rapidly varying derivatives that complicates quadrature of the discrete Galerkin form. Moreover, the extension of the diffraction method into three dimensions is complex.

The transparency method was developed as an alternative to the diffraction method by Organ et al. [84]. The transparency method is easier to extend into three dimensions than the diffraction method. In the transparency method, the crack is made transparent near the crack tip. An additional requirement is usually imposed for particles close to the crack. Since the angle between the crack and the ray from the node to the crack tip is small, a sharp gradient in the weight function across the line ahead of the crack is introduced. In order to reduce this effect, Organ et al. [84] imposed that all nodes have a minimum distance from the crack surface.

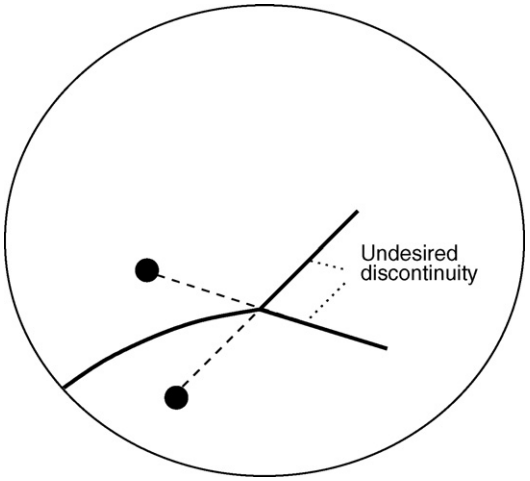

Fig. 12. Undesired introduced discontinuities by the visibility method.

2.9.2. Modification of the intrinsic basis

In methods that use an intrinsic basis such as the EFGM, the intrinsic basis can be modified according to the crack kinematics [50]. In LEFM, generally the asymptotic near-tip displacement field of the Westergaard solution is introduced into the basis $\mathbf{p}$:

$$\mathbf{p}^{\mathrm{T}}(\mathbf{X}) = \left[1, X, Y, \sqrt{r}\,\sin(\theta/2), \sqrt{r}\,\cos(\theta/2), \sqrt{r}\,\sin(\theta/2)\,\sin(\theta), \sqrt{r}\,\cos(\theta/2)\,\sin(\theta)\right] \tag{79}$$

where r is the radial distance to the crack tip and θ the angle to the crack. One drawback of intrinsic enrichment is that it has to be used in the entire domain. Otherwise, undesired discontinuities are introduced. To reduce computational cost, a blending domain is often introduced where the higher-order basis is decreased to a basis of lower-order (in our case linear complete basis) continuously.

[44] suggested an alternative intrinsic enrichment by enriched kernel functions:

$$\begin{aligned} w_{\mathrm{c}}(\mathbf{X}) &= \alpha\sqrt{r}\,\cos\frac{\theta}{2} w_4(\mathbf{X}) \\ w_{\mathrm{p}}(\mathbf{X}) &= \alpha\sqrt{r}\left(1+\sin\frac{\theta}{2}\right) w_4(\mathbf{X}) \\ w_{\mathrm{p}}(\mathbf{X}) &= \alpha\sqrt{r}\left(1-\sin\frac{\theta}{2}\right) w_4(\mathbf{X}) \end{aligned} \tag{80}$$

where $w_4(\mathbf{X})$ is the quartic spline and the factor α controls the amplitude of the enriched kernel function compared with the amplitude of the regular nodes. The value of α is usually set to 1. The indices c, m and p stand for cos, minus sin and plus sin, respectively. An advantage of this method is that no blending domain needs to be introduced.

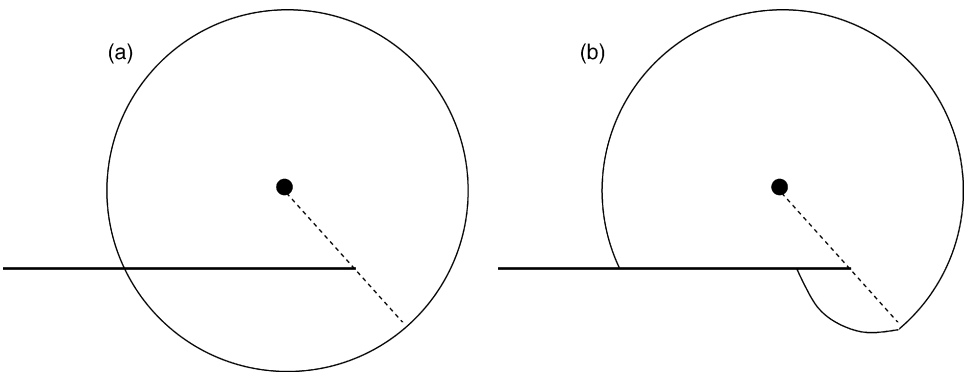

Fig. 13. (a) Scheme of the visibility method and (b) scheme of the diffraction/transparency method.

2.9.3. Methods based on an extrinsic MLS enrichment

Another possibility to model cracks in meshless methods is to introduce the analytical solution extrinsically [50]:

$$\mathbf{u}^h(\mathbf{X},t)=\sum_{J\in S}\mathbf{p}(\mathbf{X}_J)^{\mathrm{T}}\mathbf{a}(\mathbf{X},t)+\sum_{K=1}^{n_c}\left(k_{\mathrm{I}}^K\mathbf{Q}_{\mathrm{I}}^K(\mathbf{X}_{\mathrm{I}})+k_{\mathrm{I}}^K\mathbf{Q}_{\mathrm{II}}^K(\mathbf{X}_{\mathrm{I}})\right) \tag{81}$$

where n_c is the number of cracks in the model, $\mathbf{u}^h$ is the approximation of $\mathbf{u}$, $\mathbf{p}$ is the usual polynomial basis and k_{I} and k_{II} are additional degrees of freedom associated with mode-I fracture and mode-II fracture. The functions Q_{I}^i and Q_{II}^i, $i=1,2$ describe the near-tip displacement field and are given by

$$Q_{\mathrm{I}}^1(\mathbf{X})=\frac{1}{2G}\sqrt{\frac{r}{2\pi}}\cos(0.5\theta)(\kappa-1+2\sin^2(0.5\theta)) \tag{82}$$

$$Q_{\mathrm{I}}^2(\mathbf{X})=\frac{1}{2G}\sqrt{\frac{r}{2\pi}}\sin(0.5\theta)(\kappa+1-2\cos^2(0.5\theta)) \tag{83}$$

$$Q_{\mathrm{II}}^1(\mathbf{X})=\frac{1}{2G}\sqrt{\frac{r}{2\pi}}\sin(0.5\theta)(\kappa+1+2\cos^2(0.5\theta)) \tag{84}$$

$$Q_{\mathrm{II}}^2(\mathbf{X})=-\frac{1}{2G}\sqrt{\frac{r}{2\pi}}\cos(0.5\theta)(\kappa-1-2\sin^2(0.5\theta)) \tag{85}$$

where G is the shear modulus and κ is the Kolosov constant defined as $\kappa=3-4\nu$ for plane strain and $\kappa=(3-\nu)/(1+\nu)$ for plane stress conditions where ν is the Poisson's ratio.

One advantage of the MLS extrinsic enrichment is that the stress intensity factors can be directly obtained without considering the J-integral. Therefore, the enrichment has to be introduced globally, which comes with additional computational cost.

2.9.4. Methods based on an extrinsic PUM

Motivated by the XFEM [79], an extrinsic PU enrichment for meshless methods was presented in [99]:

$$\mathbf{u}^h(\mathbf{x})=\sum_{I\in\mathcal{S}}\Phi_I(\mathbf{x})\mathbf{u}_I+\sum_{J\in\mathcal{S}^c}\Phi_J(\mathbf{x})H(\mathbf{x})\mathbf{a}_J+\sum_{K\in\mathcal{S}^f}\Phi_K(\mathbf{x})\sum_{\alpha=1}^{4}\mathbf{b}_K^{\alpha}B_\alpha(\mathbf{x}) \tag{86}$$

where Φ_I are MLS shape functions. The Heaviside function and the branch functions are given by

$$H(\mathbf{x})=\begin{cases}+1 & \text{if } (\mathbf{x}-\mathbf{x}^*)\cdot\mathbf{n}\geq 0\\ -1 & \text{otherwise}\end{cases} \tag{87}$$

where $\mathbf{x}^*$ is the projection of point $\mathbf{x}$ on the crack:

$$\mathbf{B}(r,\theta)\equiv[B_1,B_2,B_3,B_4]=\left[(r,\theta)\sqrt{r}\sin\frac{\theta}{2},\sqrt{r}\cos\frac{\theta}{2},\sqrt{r}\sin\frac{\theta}{2}\cos\theta,\sqrt{r}\cos\frac{\theta}{2}\cos\theta\right] \tag{88}$$

where r and θ are polar coordinates in the local crack front coordinate system. A two-dimensional plot of the branch functions is shown in Fig. 14 The set $\mathcal{S}^c$ includes the nodes whose support contains point $\mathbf{x}$ and is cut by the crack, see Fig. 15 whereas the set $\mathcal{S}^f$ are nodes whose support contains point $\mathbf{x}$ and the crack tip $\mathbf{x}_{\mathrm{tip}}$, see Fig. 16.

Using the Galerkin procedure as described in previous sections, the usual discrete equations are obtained with only one difference in the $\mathbf{B}$ [9] matrix which is now larger:

$$\mathbf{B}=[\mathbf{B}^{\mathrm{std}}|\mathbf{B}^{\mathrm{enr}}] \tag{89}$$

[9] With the assumption that nodes on essential and natural boundaries are not enriched. For more details, refer to [93].

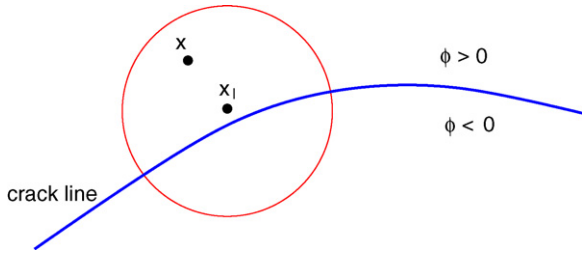

Fig. 14. Two-dimensional plot of branch functions. It is clear that the first function is discontinuous through crack face.

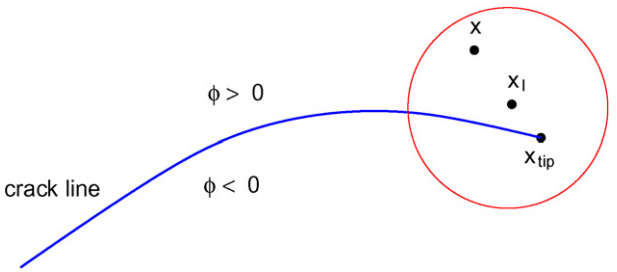

Fig. 15. The elements of set $\mathcal{N}^c$ are nodes whose support contains point $\mathbf{x}$ and cut by the crack.

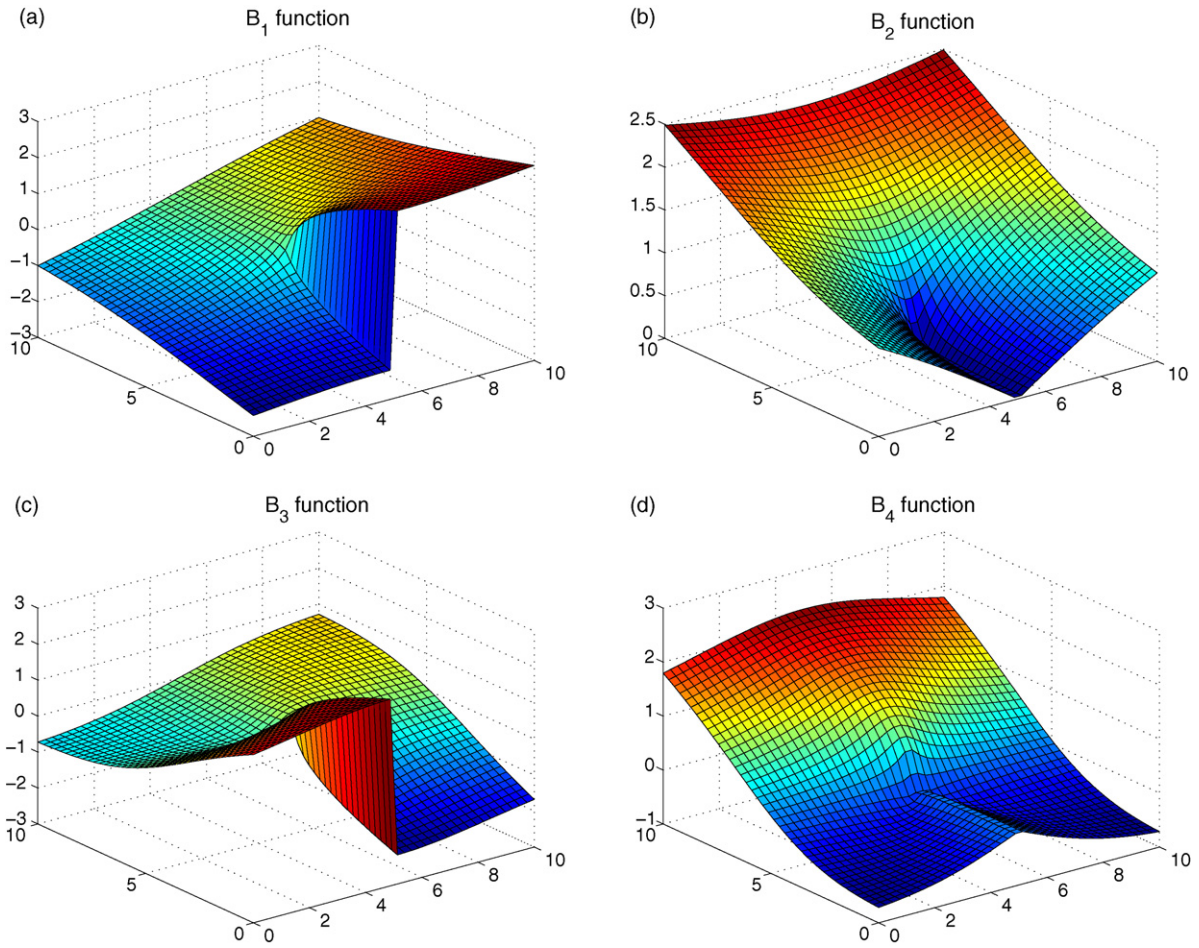

Fig. 16. The elements of set $\mathcal{N}^f$ are nodes whose support contains point $\mathbf{x}$ and the crack tip $\mathbf{x}_{\text{tip}}$.

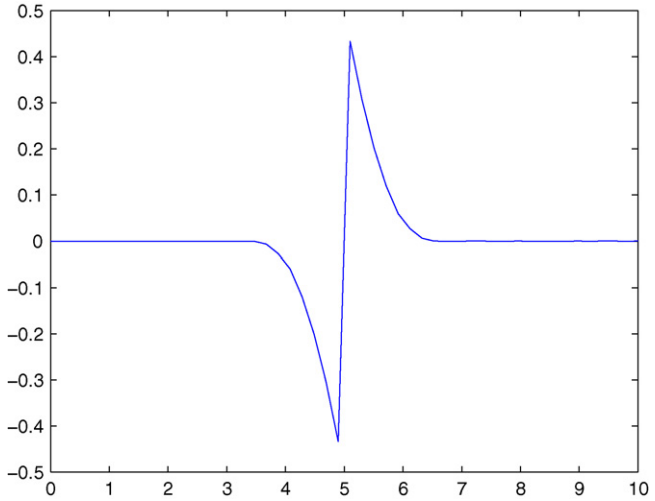

Fig. 17. Enrichment function Ψ with discontinuous derivative.

where $\mathbf{B}^{\text{std}}$ is the standard $\mathbf{B}$ and $\mathbf{B}^{enr}$ is the enriched $\mathbf{B}$ matrix:

$$\mathbf{B}_{\text{I}}^{\text{enr}} = \begin{bmatrix} (\Phi_I)_{,x}\Psi_I + \Phi_I(\Psi_I)_{,x} & 0 \\ 0 & (\Phi_I)_{,y}\Psi_I + \Phi_I(\Psi_I)_{,y} \\ (\Phi_I)_{,y}\Psi_I + \Phi_I(\Psi_I)_{,y} & (\Phi_I)_{,x}\Psi_I + \Phi_I(\Psi_I)_{,x} \end{bmatrix} \tag{90}$$

where $\Phi_I(\mathbf{x})$ can be either the Heaviside function $H(\mathbf{x})$, or the branch functions $B_\alpha(\mathbf{x})$. This enriched EFG can be implemented within an available EFG code with little modification.

2.9.5. Discontinuous derivatives

For PDEs with discontinuous coefficients, the solutions usually have discontinuous derivatives along the discontinuity. While it is trivial to treat discontinuous derivatives such as material interfaces in FEM by meshing the domain such that the element edges are aligned with the interface, it is not so simple in MMs. There are different approaches to treat discontinuous derivatives such as the Lagrange multiplier method, the global enrichment approach [13], the local or PUM-enrichment strategy [92]. In the global enrichment method, a special function Ψ whose derivative is discontinuous through the line of discontinuity (material interface for instance) is added into the approximation space:

$$u^h(x) = \sum_I \Phi_I(x)u_I + b\Psi(x - x_a) \tag{91}$$

where $\Psi(x)$ is the enrichment function and b is additional unknown of the problem, Ψ has the form (see Fig. 17 to see its discontinuous derivative):

$$\Psi(x) = \langle x - x_a\rangle - \sum_I \phi_I(x)\langle x_I - x_a\rangle \tag{92}$$

with

$$\langle x\rangle = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases} \tag{93}$$

As an example, consider the following problem:

$$(E(x)u_{,x})_{,x} + x = 0, \quad 0 \leq x \leq 10; \qquad u(0) = u(10) = 0 \tag{94a}$$

$$E(x) = \begin{cases} 1 & 0 \leq x < 5 \\ 0.5 & 5 \leq x \leq 10 \end{cases} \tag{94b}$$

The weak form of this problem is given by

$$-\int_0^1 v_{,x}(x)E(x)u_{,x}(x)\,\mathrm{d}x + \int_0^1 v(x)x\,\mathrm{d}x = 0 \tag{95}$$

Trial and test functions are constructed by Eq. (91), it results in (using the arbitrariness of δu_i and δb):

$$\begin{aligned}&\int_0^1 \Phi_{I,x}(x)E(x)\Phi_{J,x}(x)u_J + \int_0^1 \Phi_{I,x}(x)E(x)\Psi_{,x}(x)b\,\mathrm{d}x - \int_0^1 \Phi_I(x)x\,\mathrm{d}x = 0,\\ &\int_0^1 \Psi_{,x}(x)E(x)\Phi_{J,x}(x)u_J\,\mathrm{d}x + \int_0^1 \Psi_{,x}(x)\Psi_{,x}(x)b\,\mathrm{d}x - \int_0^1 \Psi(x)x\,\mathrm{d}x = 0\end{aligned} \tag{96}$$

In matrix form:

$$\begin{bmatrix}\mathbf{K} & \mathbf{B}\\ \mathbf{B}^{\mathrm{T}} & g_1\end{bmatrix}\begin{Bmatrix}\mathbf{u}\\ b\end{Bmatrix} = \begin{Bmatrix}\mathbf{f}\\ g\end{Bmatrix} \tag{97}$$

with

$$\mathbf{K}_{IJ} = \int_0^1 \Phi_{I,x}(x)E(x)\Phi_{J,x}(x)\,\mathrm{d}x, \qquad \mathbf{B}_{I1} = \int_0^1 \Phi_{I,x}(x)E(x)\Psi_{,x}(x)\,\mathrm{d}x \tag{98}$$

$$\mathbf{f}_I = \int_0^1 \Phi_I(x)x\,\mathrm{d}x, \qquad g = \int_0^1 \Psi(x)x\,\mathrm{d}x, \qquad g_1 = \int_0^1 \Psi_{,x}(x)\Psi_{,x}(x)\,\mathrm{d}x \tag{99}$$

The results obtained with this enrichment are plotted in Fig. 18. It is clear that without enrichment, the discontinuity in the derivative of the unknown function cannot be captured.

It is clear that with this global enrichment method, one must choose smartly the enrichment function, namely this function must have local character (discontinuous derivative through a material interface, for instance) and zero elsewhere. The choice of this function is not trivial in two dimensions, especially for complex interfaces. For these cases, the local PUM-enrichment strategy works best. To model a discontinuity in the derivative (or weak discontinuity), the following approximation is used:

$$\mathbf{u}^h(\mathbf{x}) = \sum_{I\in\mathcal{S}} \Phi_I(\mathbf{x})\mathbf{u}_I + \sum_{J\in\mathcal{S}^c} \Phi_J(\mathbf{x})|f(\mathbf{x})|\mathbf{a}_J \tag{100}$$

where f is the signed distance to the discontinuity line and $\mathcal{S}^c$ is the set of nodes whose support is cut by this line.

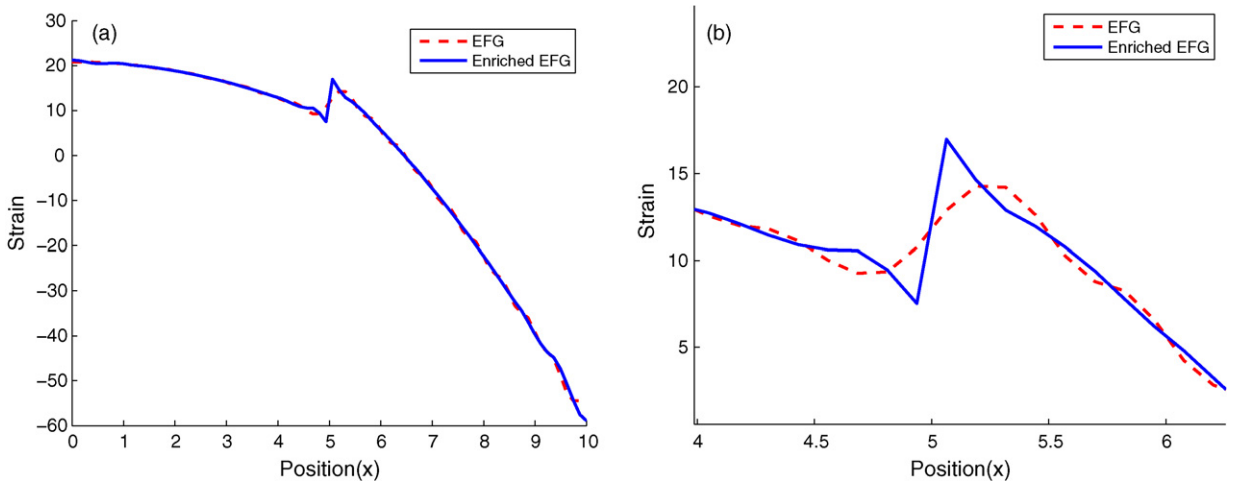

Fig. 18. Strain computed with and without enrichment.

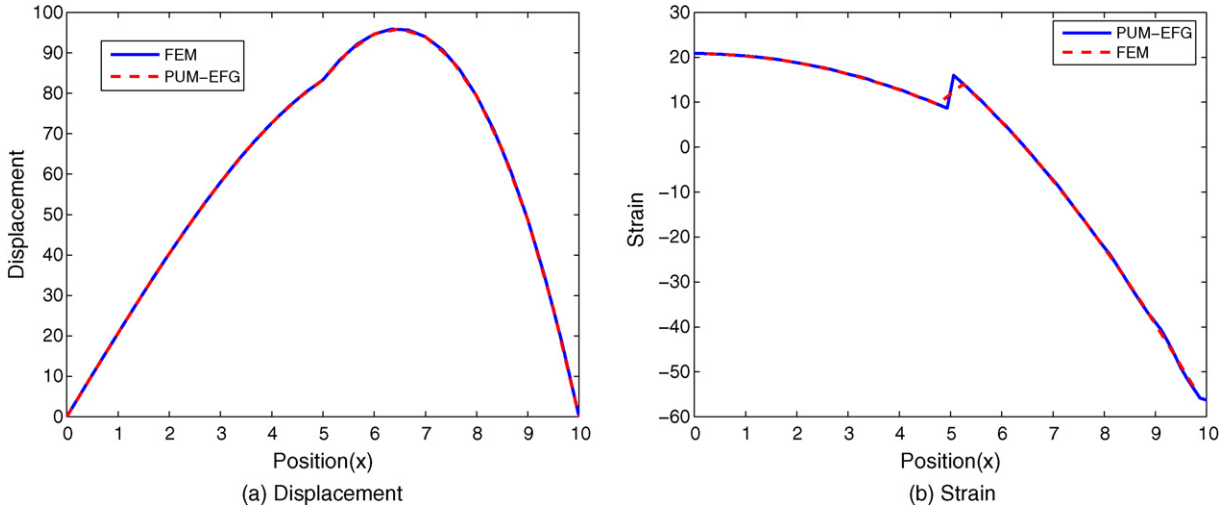

Fig. 19. Numerical solution with the PUM enrichment strategy.

The problem of a one-dimensional bar with two materials is now solved again with this so-called local enrichment strategy. In the computation, 21 equally spaced particles are used, in each of the 20 intervals, 3 Gauss points are employed. The size of the circular nodal support is $2\,\Delta x$ where Δx is the nodal spacing. The solution is given in Fig. 19 and a comparison with the solution obtained by 21 linear finite elements is also given.

2.10. Error estimation and adaptivity

Due to the absence of a mesh, *h*-adaptivity is easier to incorporate in MMs than in mesh-based methods. Also *p*-adaptivity and *r*-adaptivity is conceptually easier to implement in a meshfree framework. To drive the adaptivity, a local error estimator – or, at least, an indicator – is necessary. The most significant works on error estimation in the frame of MMs are as follows. Duarte and Oden, in [39], present an *a posteriori* error estimator and use it in an *hp*-adaptive method. [64] developed an *a posteriori* approximation error in order to adaptively refine corrected derivative in meshfree methods. [32] suggest a residual-based error estimator based on the difference between a recovered stress field and a raw EFG field, like in the well-known ZZ error estimator in the FEM [104]. This estimator is used in an adaptive method for static cracks in [33] and for propagating cracks in [69]. This estimator is also found in [67,68]. Other estimators and adaptive methods are proposed in [59,66,87,86,102,52,51,53]. Global strict bounds on the energy are obtained by a dual meshfree method in [42]. An excellent overview on adaptive Galerkin meshfree methods is given in [76].

The very recent work of Duflot and Bordas [119–121] on error measures for extended finite element methods may be a good starting point for further developments of error estimators in the context of meshfree methods with intrinsic (see Section 2.9.2) or extrinsic (see Section 2.9.4) enrichment.

3. Computer implementation aspects

There are considerable differences between the finite element methods and meshless methods, which leads to different computer implementation of MMs compared to FEM. We could cite (i) computation of shape functions and their derivatives, (ii) assembly procedure, (iii) imposing essential boundary conditions and (iv) post-processing step. This section gives details on how to write an EFG code. In addition, the PUM-enriched EFG is also presented. The Matlab language is chosen.

3.1. General meshless procedure

1. Node generation including node coordinates and associated weight functions. At each node, one must specify (i) the shape of the domain of influence (for example, circular), (ii) size of this support (radius for circular support) and (iii) the functional form (for instance the quartic spline function).

2. Insert integration points (coordinates and weights) in the domain.
3. Insert integration points along traction and essential boundaries.
4. Integrate on the domain. For each Gauss point $\mathbf{x}_g$:
   - Find nodes within the support of $\mathbf{x}_g$.
   - For each of these nodes, compute, the weight function, shape function and shape function derivatives.
   - Compute **B** matrix.
   - Compute and assemble **K** matrix.
5. Integrate on the boundaries. Integrate forces along the traction boundary to form the nodal force vector **f** and also on the essential boundary to impose essential boundary conditions.
6. Solve the resulting system of equations (obtain the fictitious displacement field, if the approximation does not have the Kronecker delta property).
7. Reconstruct the true nodal displacement from the fictitious displacements.

3.2. Efficient shape function computation

The computation of the MLS shape functions as well as its derivatives involves the inverse of the moment matrix which becomes burdensome in two and three dimensions. An efficient approach, presented in [11,41] is reproduced here for completeness.

In order to avoid the direct computation of the inverse of the moment matrix, the MLS shape function is usually written in the form:

$$\Phi_I(\mathbf{x}) = \mathbf{c}^{\mathrm{T}}(\mathbf{x}) w_I(\mathbf{x}) \mathbf{p}(\mathbf{x}_I), \quad \text{where } \mathbf{A}(\mathbf{x})\mathbf{c}(\mathbf{x}) = \mathbf{p}(\mathbf{x}) \tag{101}$$

with

$$\mathbf{A}(\mathbf{x}) = \sum_{I=1}^{n} w_I(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) \mathbf{p}^{\mathrm{T}}(\mathbf{x}_I) \tag{102}$$

To efficiently compute $\mathbf{c}(\mathbf{x})$, the LU factorization of **A** is performed together with backward substitution:

$$\mathbf{LUc}(\mathbf{x}) = \mathbf{p}(\mathbf{x}), \qquad \mathbf{Uc}(\mathbf{x}) = \mathbf{L}^{-1}\mathbf{p}(\mathbf{x}), \qquad \mathbf{c}(\mathbf{x}) = \mathbf{U}^{-1}\mathbf{L}^{-1}\mathbf{p}(\mathbf{x}) \tag{103}$$

The first derivatives of the shape functions are given by

$$\Phi_{I,k}(\mathbf{x}) = \mathbf{c}_k^{\mathrm{T}}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) w_I(\mathbf{x}) + \mathbf{c}^{\mathrm{T}}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) w_{I,k}(\mathbf{x}) \tag{104}$$

with

$$\begin{aligned} \mathbf{c}_k(\mathbf{x}) &= \mathbf{A}_k^{-1}(\mathbf{x})\mathbf{p}(\mathbf{x}) + \mathbf{A}^{-1}(\mathbf{x})\mathbf{p}_k(\mathbf{x}) = -\mathbf{A}^{-1}(\mathbf{x})\mathbf{A}_k(\mathbf{x})\mathbf{A}^{-1}(\mathbf{x})\mathbf{p}(\mathbf{x}) + \mathbf{A}^{-1}(\mathbf{x})\mathbf{p}_k(\mathbf{x}) \\ &= \mathbf{A}^{-1}(\mathbf{x})[-\mathbf{A}_k(\mathbf{x})\mathbf{c}(\mathbf{x}) + \mathbf{p}_k(\mathbf{x})] = \mathbf{A}^{-1}(\mathbf{x})\mathbf{b}_k \end{aligned} \tag{105}$$

and

$$\mathbf{A}_k(\mathbf{x}) = \sum_{i=1}^{n} w_{i,k}(\mathbf{x}) \mathbf{p}(\mathbf{x}_i) \mathbf{p}^{\mathrm{T}}(\mathbf{x}_i) \tag{106}$$

The second derivatives (usually necessary in point collocation method, in plate and shell modeling or for stabilization schemes based on finite increment calculus [122]) are computed in the same manner:

$$\Phi_{I,kl}(\mathbf{x}) = \mathbf{c}_{kl}^{\mathrm{T}}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) w_I(\mathbf{x}) + \mathbf{c}_k^{\mathrm{T}}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) w_{I,l}(\mathbf{x}) + \mathbf{c}_l^{\mathrm{T}}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) w_{I,k}(\mathbf{x}) + \mathbf{c}^{\mathrm{T}}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) w_{I,kl}(\mathbf{x}) \tag{107}$$

with

$$\mathbf{c}_{kl}(\mathbf{x}) = \mathbf{A}^{-1}(\mathbf{x}) \left(\mathbf{p}_{kl}(\mathbf{x}) - \mathbf{A}_l(\mathbf{x})\mathbf{c}_k(\mathbf{x}) - \mathbf{A}_k(\mathbf{x})\mathbf{c}_l(\mathbf{x}) - \mathbf{A}_{kl}(\mathbf{x})\mathbf{c}(\mathbf{x}) \right) \tag{108}$$

where

$$\mathbf{A}_{kl}(\mathbf{x}) = \sum_{I=1}^{n} w_{I,kl}(\mathbf{x}) \mathbf{p}(\mathbf{x}_I) \mathbf{p}^{\mathrm{T}}(\mathbf{x}_I) \tag{109}$$

Listing 1. Stiffness matrix assembly.

```
function [phi,dphidx,dphidy] =
MLS_ShapeFunction(pt,index,node,di,form)

% Compute the MLS shape function at point pt for all nodes
% within the support of this point pt.
% Basis used is linear basis pT = [1 x y]

% compute the moment matrix A

A    = zeros(3,3) ; dAdx = zeros(3,3) ; dAdy = zeros(3,3) ;
for m = 1 : size(index,2)
    xi = [node(index(m),1) node(index(m),2)] ;
    [wi,dwidx,dwidy] = circle_spline(pt,xi,di,form);
    pTp = [1 xi(1,1) xi(1,2)]'*[1 xi(1,1) xi(1,2)] ;
    A    = A    + wi*pTp ;
    dAdx = dAdx + dwidx*pTp ;
    dAdy = dAdy + dwidy*pTp ;
    % store weight function and derivatives at node I
    w(m)    = wi ;
    dwdx(m) = dwidx ;
    dwdy(m) = dwidy ;
end

p  = [1; pt(1,1); pt(1,2)];

% ---------------------------------
%   compute  matrix c(x),c,k(x)
% ---------------------------------

% Using LU factorization for A
[L,U,PERM] = lu(A) ;

for i = 1 : 3
    if i == 1          % backward substitution for c(x)
        C = PERM*p;
    elseif i == 2      % backward substitution for c,x(x)
        C = PERM*([0 1 0]' - dAdx*c(1:3,1));
    elseif i == 3      % backward substitution for c,y(x)
        C = PERM*([0 0 1]' - dAdy*c(1:3,1));
    end

    D1 = C(1);
    D2 = C(2) - L(2,1)*D1;
    D3 = C(3) - L(3,1)*D1 - L(3,2)*D2 ;

    c(3,i) = D3/U(3,3) ;
    c(2,i) = (D2 - U(2,3)*c(3,i))/(U(2,2));
    c(1,i) = (D1 - U(1,2)*c(2,i) - U(1,3)*c(3,i))/(U(1,1));
end

  for m = 1 : size(index,2)
      xi = [node(index(m),1) node(index(m),2)] ;
      piT = [1 xi(1,1) xi(1,2)]';
      phi(m) = c(:,1)'* piT*w(m) ;
      dphidx(m) = c(:,2)'*piT*w(m) + c(:,1)'*piT*dwdx(m) ;
      dphidy(m) = c(:,3)'*piT*w(m) + c(:,1)'*piT*dwdy(m);
  end
```

3.3. Gauss point generation

Assume that the integration is performed with background integration cells. In two dimensions, each integration cell is a four node quadrilateral element with shape functions N_I and nodal coordinates $\mathbf{x}_0$ ($I = 1, \ldots, 4$). For each

Gauss point ($\boldsymbol{\xi}_{\text{gp}}$, w_{gp}) of a given cell, the isoparametric mapping is used to get its global coordinates $\mathbf{x}_{\text{gp}}$:

$$\mathbf{x}_{\text{gp}} = \sum_{I=1}^{4} N_I(\boldsymbol{\xi}_{\text{gp}})\mathbf{x}_I \tag{110}$$

and its global weight is given by

$$w = w_{\text{gp}} \times det\mathbf{J} \tag{111}$$

where $\mathbf{J}$ is the Jacobian of the physical-parent transformation.

3.4. Assembly procedure

The assembly procedure in MMs is performed on the domain of influence of the point under consideration (often a Gauss point). If we store the nodal unknowns $\mathbf{u}_I$ as follows:

$$\mathbf{u}^{\text{T}} = [u_1 \quad v_1 \quad u_2 \quad v_2 \ldots u_n \quad v_n] \tag{112}$$

where n is the number of nodes. Then a node I will contribute to the $(2I-1)$ th row and the $(2I)$ th column. If we denote the variable *index* containing the number of nodes within the support of a given Gauss point, then the assembly procedure at this Gauss point is given in the following listing.

Listing 2. Stiffness matrix assembly.

```
B  = zeros(3,2*size(index,2)) ;
en = zeros(1,2*size(index,2));
[phi,dphidx,dphidy] = MLS_ShapeFunction(pt,index,node,di,form);
for m = 1 : size(index,2) % loop on nodes within support
    B(1:3,2*m-1:2*m) = [dphidx(m) 0;
                        0 dphidy(m);
                        dphidy(m) dphidx(m)];
    en(2*m-1) = 2*index(m)-1;
    en(2*m  ) = 2*index(m)  ;
end
K(en,en) = K(en,en) + B'*C*B * W(igp)*J(igp) ;
```

In collocation methods, assembly of the stiffness matrix is done row by row, i.e. degree of freedom by degree of freedom.

3.5. Integration on the essential boundaries

3.5.1. Point collocation method

Recall the formulas for matrix $\mathbf{G}$ and vector $\mathbf{q}$:

$$\mathbf{G}_{IK} = -\Phi_I(\mathbf{x}_K)\mathbf{S} \tag{113}$$

$$\mathbf{q}_K = -\mathbf{S}\bar{\mathbf{u}}(\mathbf{x}_K) \tag{114}$$

where $\mathbf{S}$ is a diagonal matrix of size 2×2 in two dimensions, and $S_{ii} = 1$ if the displacement is imposed on x_i and $S_{jj} = 0$ otherwise. $\mathbf{x}_K$ are collocation points.

Assume that, along the essential boundary Γ_u, m collocation points are used. Then we have $m \times 2$ constraint equations (in two dimensions). Hence, the dimension of $\mathbf{G}$ is $2n \times 2m$ with n the number of nodes in the domain.

Listing 3. Point collocation for imposing essential BCs.

```
q  = zeros(1,2*m); G  = zeros(2*n,2*m); ind = 0 ;
for i = 1 : m     % loop on collocation points
    ind = ind + 1 ;
    pt = node(disp_nodes(i),:); % x_K
    % compute prescribed displacement, u_bar
    ux_bar = ... ;
    uy_bar = ... ;
    % q vector
    q(2*ind-1) = -ux_bar ;
    q(2*ind)   = -uy_bar ;
    % S matrix
    S = [1 0; 0 1]; % both directions
    % G matrix
    [index] = define_support(node,pt,di);
    [phi,dphidx,dphidy] =
    MLS_ShapeFunction(pt,index,node,di,form);
    for j = 1 : size(index,2)
        row1 = 2*index(j)-1 ;
        row2 = 2*index(j)   ;
        Gj   = -phi(j) * S ;
        G(row1:row2,2*ind-1:2*ind) = ...
        G(row1:row2,2*ind-1:2*ind) + Gj;
    end
end
```

3.5.2. *Finite element interpolation for Lagrange multiplier*

For ease of reading, the equations are recalled:

$$\mathbf{G}_{IK} = -\int_{\Gamma_u} \Phi_I N_K \mathbf{S}\,\mathrm{d}\Gamma, \qquad \mathbf{q}_K = -\int_{\Gamma_u} N_K \mathbf{S}\bar{\mathbf{u}}\,\mathrm{d}\Gamma \tag{115}$$

Let us discretize the essential boundary with $(m-1)$ two-noded finite elements. For each element, n_{gp} Gauss points are used. The shape functions for a two-node element are given by (l_e is the length of the element)

$$N_1(x) = 1 - \frac{x}{l_e}, \qquad N_2(x) = 1 - N_1(x) \tag{116}$$

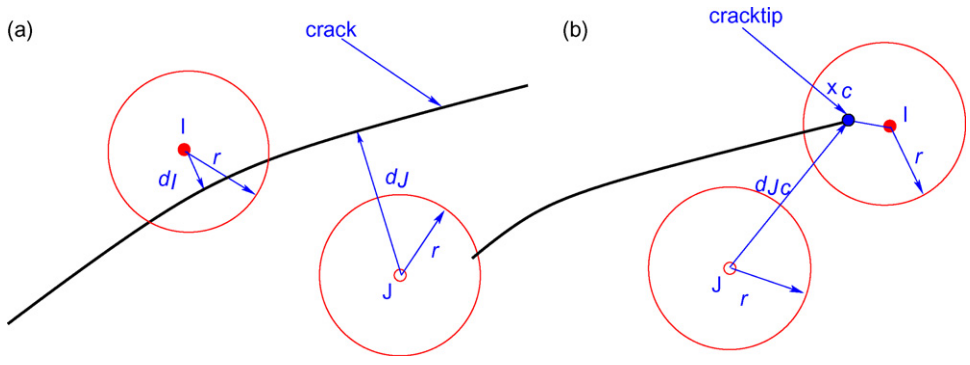

Fig. 20. Selection of enriched particles (filled particles): (a) discontinuous enriched particles; (b) near tip enriched particles.

Listing 4. Finite element Lagrange multipliers.

```
m1 = 0 ;
for i = 1 : (m - 1) % loop on 1D elements
    ctr = [disp_nodes(i) disp_nodes(i+1)];
  m1 = m1 + 1 ;
  m2 = m1 + 1 ;
  le = norm(disp_nodes(m1) - disp_nodes(m2));
  for q = 1 : n_GP  % loop on GPs
      pt = Q1(q,:); % local GP
      wt = W1(q);   % weight
      [N,dNdxi] = lagrange_basis('L2',pt);
      J0 = dNdxi'*node(sctr,:);
      detJ = norm(J0) ;
      pt = N' * node(sctr,:); % global GP
      % compute prescribed displacement, u_bar
      ux_bar = ... ;
      uy_bar = ... ;
      % S matrix
      S = [1 0; 0 1]; % both directions
      % 1D element's shape functions N_K
      N1 = 1 - pt(1,2)/le ; N2 = 1 - N1 ;
      % qk vector
      qk(2*m1-1) = qk(2*m1-1) - wt * detJ * N1 * ux_bar ;
      qk(2*m1)   = qk(2*m1)   - wt * detJ * N1 * uy_bar ;
      qk(2*m2-1) = qk(2*m2-1) - wt * detJ * N2 * ux_bar ;
      qk(2*m2)   = qk(2*m2)   - wt * detJ * N2 * uy_bar ;

      % G matrix
      [index] = define_support(node,pt,di);
      [phi,dphidx,dphidy] =
      MLS_ShapeFunction(pt,index,node,di,form);
      for j = 1 : size(index,2)
          r1 = 2*index(j)-1 ;
          r2 = 2*index(j)   ;
          G1   = - wt * detJ * phi(j) * N1 * S;
          G2   = - wt * detJ * phi(j) * N2 * S;
          G(r1:r2,2*m1-1:2*m1) = G(r1:r2,2*m1-1:2*m1) + G1;
          G(r1:r2,2*m2-1:2*m2) = G(r1:r2,2*m2-1:2*m2) + G2;
      end
  end
end
```

3.6. Enriched EFG

In comparison to the EFG, the enriched EFG has the following differences:

- Detection of non-enriched and enriched particles.
- Treatment of enriched (additional) degrees of freedom.
- Computation of stiffness matrices.

The selection of enriched particles with circular support is illustrated in Fig. 20. It suffices to compute the signed distances from particles to the crack line and the distances from particles to the crack tip and compare these distances to the radius of the domains of influence. This procedure, implemented in Matlab, is given in the following listing.

Listing 5. Selection of enriched particles.

```
x0  = xCr(1,1); y0 = xCr(1,2); x1  = xCr(2,1); y1 = xCr(2,2);
for i = 1 : numnode
    x = node(i,1);
    y = node(i,2);
    l   = sqrt((x1-x0)*(x1-x0)+(y1-y0)*(y1-y0)) ;
    phi = (y0-y1)*x + (x1-x0)*y + (x0*y1-x1*y0);
    ls(i,1) = phi/l;              % normal LS
    ls(i,2) = ([x y]-xTip)*t';    % tangent LS
    dt(i)   = norm([x y]-xTip); % distances from node I to tip
end

set1 = find(abs(ls(:,1)) - di' < 0); set2 = find(ls(:,2) < 0);
set3 = find(dt - di < 0);          % tip nodes
set4 = intersect(set1,set2);       % split nodes

% some nodes belong to both sets, remove tip
split_nodes = setdiff(set4,set3); tip_nodes   = set3;
```

Due to the presence of additional degrees of freedom (dofs) the assembly procedure has to be revised. We use fictitious nodes to handle these additional dofs. At a *H*-enriched node (discontinuous enrichment), we add one phantom node and, at a tip enriched node, we add four phantom nodes. The numbering of these fictitious nodes start from the total number of true nodes plus one. For example, if there are five nodes numbered from one to five where the third node and fifth is enriched with the Heaviside function and the fourth node is a near tip enriched one, then, we have $5 + 2 \times 1 + 1 \times 4 = 11$ nodes. Then, at the third node, we add a phantom node numbered 6, at the fourth node, we add four fantom nodes numbered 7, 8, 9, 10 and at the fifth node, a phantom node numbered 11 is added. An array named *pos* is built to contain the number of these fantom nodes. It is an array of dimension numnode $\times$ 1 where numnode is the number of true nodes. For this example, pos is $pos = [0 \quad 0 \quad 6 \quad 7 \quad 11]$.

Listing 6. Selection of enriched particles (or nodes).

```
pos = zeros(numnode,1); nsnode = 0 ; ntnode = 0 ;
for i = 1 : numnode
    [snode] = ismember(i,split_nodes);
    [tnode] = ismember(i,tip_nodes);
    if (snode ~= 0)
        pos(i) = (numnode + nsnode*1 + ntnode*4) + 1 ;
        nsnode = nsnode + 1 ;
    elseif (tnode ~= 0)
        pos(i) = (numnode + nsnode*1 + ntnode*4) + 1 ;
        ntnode = ntnode + 1 ;
    end
end
```

In two dimensions, at a certain node numbered i there are always two unknowns associated with equation numbers $2i - 1$ and $2i$ in the global matrix. If this node is a discontinuous-enriched node, then it has two additional unknowns associated with equation numbers at $2 \times \text{pos}(i) - 1$ and $2 \times \text{pos}(i)$ in the global matrix. If it is a near tip enriched node, then it has eight additional unknowns with equation numbers $(2 \times \text{pos}(i) - 1, 2 \times \text{pos}(i))$,$(2 \times (\text{pos}(i) + 1) - 1, 2 \times (\text{pos}(i) + 1))$,$(2 \times (\text{pos}(i) + 2) - 1, 2 \times (\text{pos}(i) + 2))$ and $(2 \times (\text{pos}(i) + 3) - 1, 2 \times (\text{pos}(i) + 3))$, where each pair corresponds to each added phantom node. Listing 7 gives the implementation in Matlab of this assembly procedure.

Listing 7. Assembly procedure.

```
function [sctrB] = assembly(index,split_nodes,tip_nodes,...
                            pos)

[snode,s_loc] = ismember(index,split_nodes);
[tnode,t_loc] = ismember(index,tip_nodes);

% number of H and tip enriched nodes
% in index
num_split_node = size(find(snode ~= 0),2);
num_tip_node   = size(find(tnode ~= 0),2);

% Scatter of standard part of B matrix
le       = size(index,2);
sctrStdB = zeros(2*le,1);
sctrStdB(1:2:2*le) = index.*2-1 ; % x displacement
sctrStdB(2:2:2*le) = index.*2   ; % y displacement

% No enriched nodes in index

 if (num_split_node == 0) && (num_tip_node == 0)
     sctrB = sctrStdB ;
 % There is at least one enriched node in index
 else
     sn = num_split_node;
     tn = num_tip_node  ;
     sctrEnrB = zeros(2*(sn*1+tn*4),1);
     cnt = 0 ;
     for k = 1 : le
         nodeI = index(k) ; % I-th node in index
         if (snode(k) ~= 0)
             cnt = cnt + 1 ;
             sctrEnrB(2*cnt - 1) = 2 * pos(nodeI) - 1;
             sctrEnrB(2*cnt    ) = 2 * pos(nodeI)    ;
         elseif (tnode(k) ~= 0)
             cnt = cnt + 1 ;
             sctrEnrB(2*cnt - 1) = 2 * pos(nodeI) - 1;
             sctrEnrB(2*cnt    ) = 2 * pos(nodeI)    ;

             cnt = cnt + 1 ;
             sctrEnrB(2*cnt - 1) = 2 * (pos(nodeI)+1) - 1;
             sctrEnrB(2*cnt    ) = 2 * (pos(nodeI)+1)    ;

             cnt = cnt + 1 ;
             sctrEnrB(2*cnt - 1) = 2 * (pos(nodeI)+2) - 1;
             sctrEnrB(2*cnt    ) = 2 * (pos(nodeI)+2)    ;

             cnt = cnt + 1 ;
             sctrEnrB(2*cnt - 1) = 2 * (pos(nodeI)+3) - 1;
             sctrEnrB(2*cnt    ) = 2 * (pos(nodeI)+3)    ;
         end
     end
     sctrB = [ sctrStdB;sctrEnrB ];
 end
```

The **B** matrix at a Gauss point gp is composed of two parts: the standard and the enriched part, where the standard part is always computed and the enriched part is only computed if in the nodes whose supports cover gp, there exist enriched nodes. This is implemented in Matlab and given in the following listing.

Listing 8. Stiffness matrix computation.

```
function B = Bmatrix(pt,index,node,di,form,...
                     snode,tnode,xCr,xTip,alpha)

[phi,dphidx,dphidy] = MLS_ShapeFunction(pt,index,node,di,form);

le    = length(index);
StdB = zeros(3,2*le);
StdB(1,1:2:2*le)  = dphidx ;
StdB(2,2:2:2*le)  = dphidy ;
StdB(3,1:2:2*le)  = dphidy ;
StdB(3,2:2:2*le)  = dphidx ;

% number of H and tip enriched nodes
% in index

num_split_node = size(find(snode ~= 0),2);
num_tip_node   = size(find(tnode ~= 0),2);

QT = [cos(alpha) sin(alpha); -sin(alpha) cos(alpha)];

% no enriched nodes, B is simply standard B matrix
if (num_split_node == 0) && (num_tip_node == 0)
    B = StdB;
% there are enriched nodes, compute enriched part, Benr
else
    Benr = [];
    for m = 1 : le
        % a split enriched node
        if (snode(m) ~= 0)
            % compute Heaviside and derivatives
            dist = signed_distance(xCr,pt);
            [H,dHdx,dHdy] = heaviside(dist);

            BI_enr = [dphidx(m)*H 0 ;
                0 dphidy(m)*H;
                dphidy(m)*H dphidx(m)*H];
            Benr   = [Benr BI_enr];
            clear BI_enr ;
            % a tip enriched node
        elseif (tnode(m) ~= 0)
            % compute branch functions
            xp = QT*(pt-xTip)';
            [theta,r] = cart2pol(xp(1),xp(2));
            [Br,dBdx,dBdy] = branch(r,theta,alpha);

            aa = dphidx(m)*Br(1) + phi(m)*dBdx(1) ;
            bb = dphidy(m)*Br(1) + phi(m)*dBdy(1);
            B1_enr = [aa 0 ; 0 bb ; bb aa];
```

```
            aa = dphidx(m)*Br(2) + phi(m)*dBdx(2) ;
            bb = dphidy(m)*Br(2) + phi(m)*dBdy(2);
            B2_enr = [aa 0 ; 0 bb ; bb aa];

            aa = dphidx(m)*Br(3) + phi(m)*dBdx(3) ;
            bb = dphidy(m)*Br(3) + phi(m)*dBdy(3);
            B3_enr = [aa 0 ; 0 bb ; bb aa];

            aa = dphidx(m)*Br(4) + phi(m)*dBdx(4) ;
            bb = dphidy(m)*Br(4) + phi(m)*dBdy(4);
            B4_enr = [aa 0 ; 0 bb ; bb aa];

            BI_enr = [B1_enr B2_enr B3_enr B4_enr];
            clear B1_enr; clear B2_enr;
            clear B3_enr; clear B4_enr;

            Benr = [Benr BI_enr];
            clear BI_enr ;
        end
    end   % end of loop on nodes in neighbour of Gp
    % Total B matrix
    B = [StdB Benr];
    clear StdB; clear Benr;

        end       % end of check enriched nodes
```

4. Numerical examples

In this section, numerical examples in linear elasticity are presented with the purpose to verify the Matlab code.

4.1. The Timoshenko beam

Consider a beam of dimensions $L \times D$, subjected to a parabolic traction at the free end as shown in Fig. 21. The beam is considered to be of unit depth and is in plane stress state. This problem was solved previously by Dolbow and Belytschko [37].

The parabolic traction is given by

$$t_y(y) = -\frac{P}{2I}\left(\frac{D^2}{4} - y^2\right) \tag{117}$$

where $I = D^3/12$ is the moment of inertia (second moment of area). The exact displacement solution for this problem is

$$u_x(x, y) = -\frac{Py}{6EI}\left[(6L - 3x)x + (2 + \nu)\left(y^2 - \frac{D^2}{4}\right)\right],$$
$$u_y(x, y) = \frac{P}{6EI}\left[3\nu y^2(L - x) + (4 + 5\nu)\frac{D^2 x}{4} + (3L - x)x^2\right] \tag{118}$$

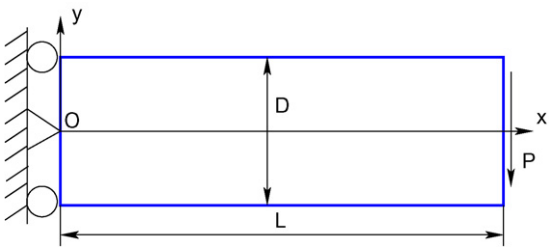

Fig. 21. The Timoshenko beam.

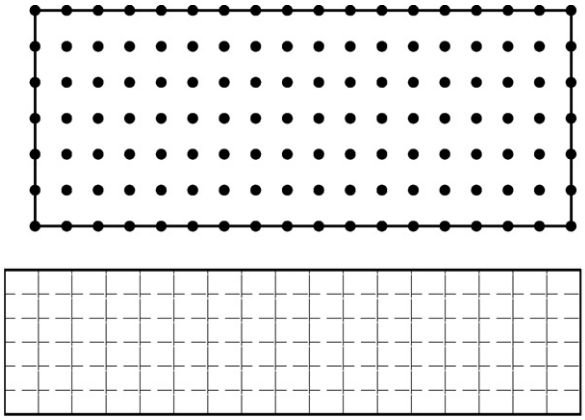

Fig. 22. 18 × 7 regular nodes distribution for the Timoshenko beam.

and the exact stresses are

$$\sigma_x(x, y) = -\frac{P(L-x)y}{I}, \qquad \sigma_{xy}(x, y) = -\frac{P}{2I}\left(\frac{D^2}{4} - y^2\right), \qquad \sigma_y(x, y) = 0 \tag{119}$$

In the computations, material properties are taken as $E = 3.0 \times 10^7$, $\nu = 0.3$ and the beam dimensions are $D = 12$ and $L = 48$. The shear force is $P = 1000$. The regular node distribution together with the background mesh used for numerical integration of the weak form are shown in Fig. 22. In each integration cell, 4×4 Gauss quadrature is used. A linear basis and cubic spline weight function are used in the MLS approximation. Circular nodal support of radius 3.5 times the nodal spacing is employed.

We check the error in the energy and displacement norm. The energy norm is given by

$$e_{\text{energy}} = \left[\frac{1}{2}\int_{\Omega} \left(\boldsymbol{\varepsilon}_{\text{num}} - \boldsymbol{\varepsilon}_{\text{exact}}^{\text{T}}\right) : \mathbf{D} : (\boldsymbol{\varepsilon}_{\text{num}} - \boldsymbol{\varepsilon}_{\text{exact}})\, \mathrm{d}\Omega\right]^{1/2} \tag{120}$$

and the displacement norm is given by

$$e_{\text{displacement}} = \sqrt{\int_{\Omega} [(\mathbf{u}_{\text{num}} - \mathbf{u}_{\text{exact}}) \cdot (\mathbf{u}_{\text{num}} - \mathbf{u}_{\text{exact}})\, \mathrm{d}\Omega]} \tag{121}$$

where $\boldsymbol{\varepsilon}_{\text{num}}$ and $\boldsymbol{\varepsilon}_{\text{exact}}$ are the numerical strain vector and exact strain vector, respectively. The same notation applies to the displacement vector $\mathbf{u}_{\text{num}}$ and $\mathbf{u}_{\text{exact}}$. The calculation has been done with the same Gauss quadrature as given above and the node distributions are 17×5, 33×9, 65×17 and 133×34.

In Table 1, the vertical displacement at point $(L, 0)$ calculated by EFG is compared with the exact solution. This table shows excellent agreement between EFG and the analytical solution.

The stresses at the center of the beam ($x = L/2$, $y \in [-D/2, D/2]$) are computed and compared with the exact solution. Eleven Gauss points along the vertical line $x = L/2$ are used in the computation. Fig. 23 shows very good agreement between the EFG result and the exact stresses. In addition, the distribution of the normal stress in the beam is plotted in Fig. 24. It is of particular interest that very smooth stresses were obtained without any additional treatment as is necessary in FEM (stress extrapolation or stress recovery).

Table 1
Comparison of vertical displacement at end of beam

Nodes	u_y exact	u_y EFG	Error (%)
7×5	−0.0089	−0.0083	−6.74
11×5	−0.0089	−0.0087	−2.24
15×9	−0.0089	−0.0088	−1.12
20×9	−0.0089	−0.0088	−1.12

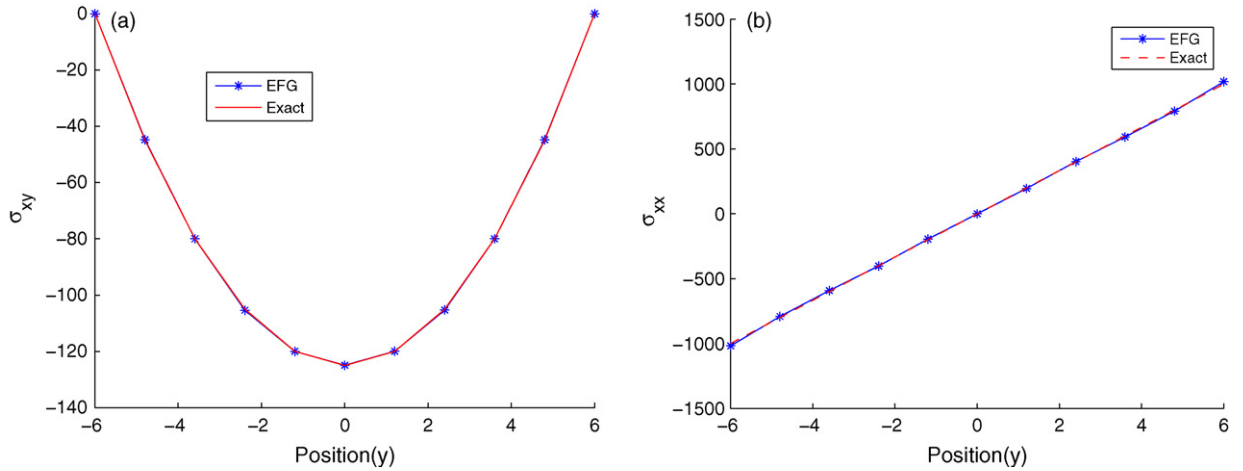

Fig. 23. Stresses comparison: (a) shear stress; (b) normal stress.

In Fig. 25, we show the error in the energy and in the displacement norm. h is the horizontal spacing between the nodes and e is the error. The convergence rate in the energy norm is 1.43.

4.2. Plate with hole

Consider an infinite plate with a centered circular hole under unidirectional tension along the x-direction. The plate dimension is taken to be $L \times L$ and the circle of radius a (Fig. 26).

The exact stress in the plate is given by

$$\sigma_x(r,\theta) = 1 - \frac{a^2}{r^2}\left(\frac{3}{2}\cos 2\theta + \cos 4\theta\right) + \frac{3}{2}\frac{a^4}{r^4}\cos 4\theta \tag{122a}$$

$$\sigma_y(r,\theta) = -\frac{a^2}{r^2}\left(\frac{1}{2}\cos 2\theta - \cos 4\theta\right) - \frac{3}{2}\frac{a^4}{r^4}\cos 4\theta \tag{122b}$$

$$\sigma_{xy}(r,\theta) = -\frac{a^2}{r^2}\left(\frac{1}{2}\sin 2\theta + \sin 4\theta\right) + \frac{3}{2}\frac{a^4}{r^4}\sin 4\theta \tag{122c}$$

where r, θ are the usual polar coordinates centered at the center of the hole.

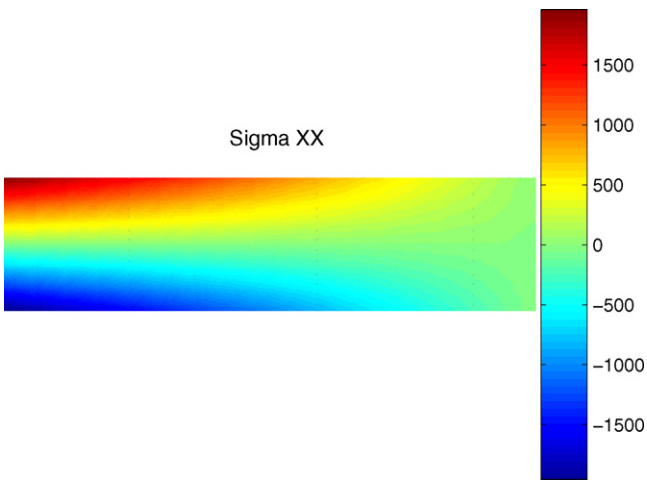

Fig. 24. Distribution of normal stress in the beam.

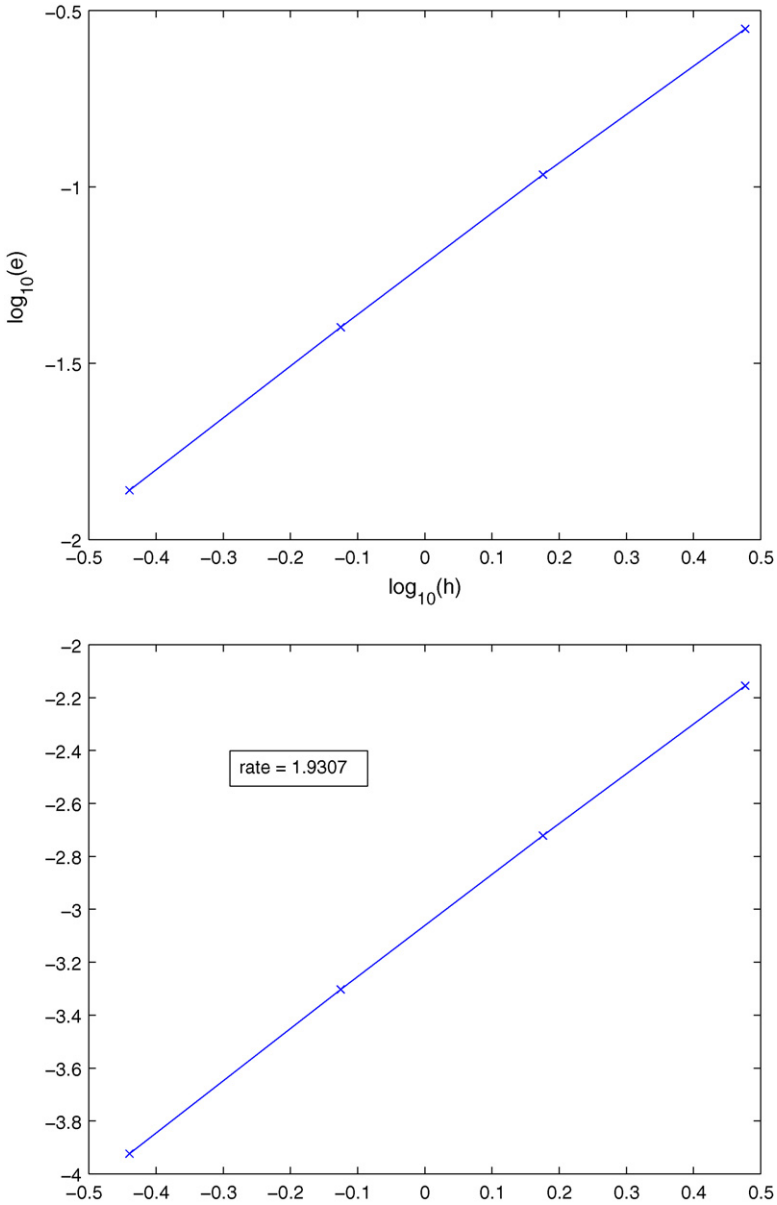

Fig. 25. Convergence in the energy (top) and displacement (bottom) norm. The optimal rates are 1.0 and 2.0, respectively.

The boundary conditions include (i) essential boundary conditions on the bottom $u_y(x, y = 0)$ and left edges $(u_x(x = 0, x = L, y) = 0)$; (ii) natural boundary conditions on the right and top edges on which traction $\bar{\mathbf{t}}$ computed from the exact stress given in Eq. (122) are applied. More precisely:

$$\bar{t}_i = \sigma_{ij} n_j \tag{123}$$

On the right edge, $\mathbf{n} = (1, 0)$, hence $\bar{\mathbf{t}}^{\mathrm{T}} = (\sigma_x, \sigma_{xy})$. Similarly, on the top edge, the imposed traction is $\bar{\mathbf{t}}^{\mathrm{T}} = (\sigma_{xy}, \sigma_y)$.

In the computation, the material properties are taken as a Young modulus of 10^3, a Poisson's ratio equal to 0.3, and the geometry is such that $L = 10$ and $a = 1$. The background mesh constructed for the numerical integration is given in Fig. 27(a). Another possibility is the structured cells as shown in Fig. 27(b). However, the background mesh is chosen for simplicity. For each background cell, a 4×4 Gauss quadrature is employed (Fig. 27).

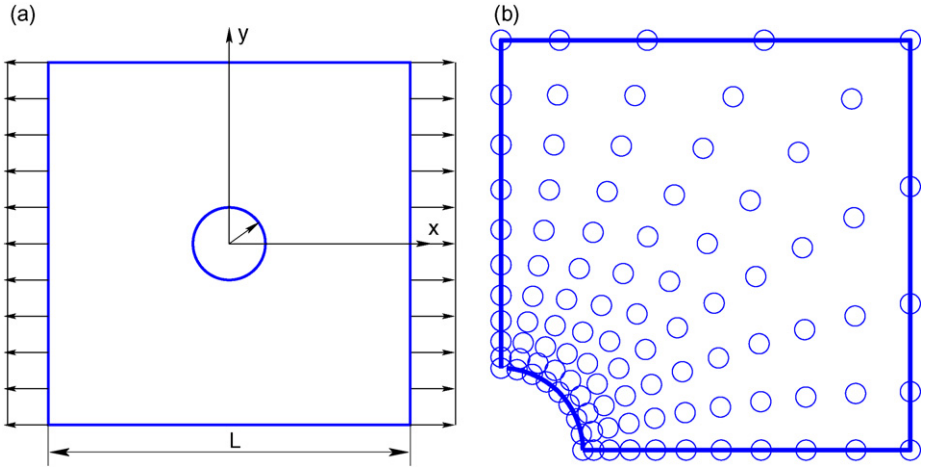

Fig. 26. Plate with a hole: (a) whole domain; (b) 1/4 model with irregular nodes.

The cubic spline function serves as a weight function. The domain of influence for all nodes is a circle with varying radius. They are chosen such that the support is small for nodes near the hole and bigger for nodes near the edges.

The essential boundary conditions are imposed with the boundary point collocation method where collocation points are coincident with nodes along the bottom and left edges.

The stress σ_x computed at nodes are plotted and compared to the exact solution (Fig. 28). With a coarse discretization of 99 nodes, quite accurate results are obtained.

4.3. *Infinite plate with a center crack*

Consider an infinite plate containing a straight crack of length $2a$ and loaded by a remote uniform stress field σ. Along ABCD the closed form solution in terms of polar coordinates in a reference frame (r, θ) centered at the crack tip is

$$u_x(r,\theta) = \frac{2(1+\upsilon)}{\sqrt{2\pi}}\frac{K_I}{E}\sqrt{r}\cos\frac{\theta}{2}\left(2-2\upsilon-\cos^2\frac{\theta}{2}\right),$$
$$u_y(r,\theta) = \frac{2(1+\upsilon)}{\sqrt{2\pi}}\frac{K_I}{E}\sqrt{r}\sin\frac{\theta}{2}\left(2-2\upsilon-\cos^2\frac{\theta}{2}\right) \tag{124}$$

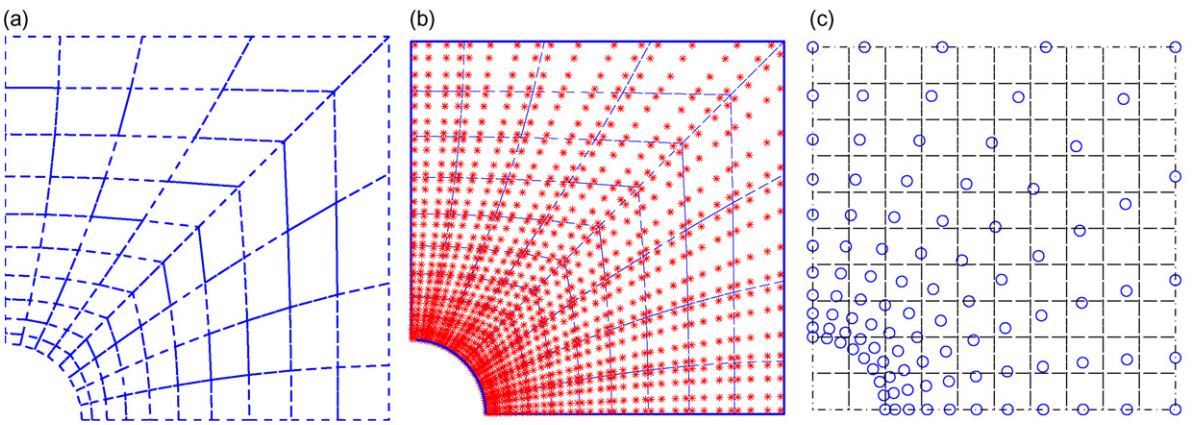

Fig. 27. Gauss points: (a) background mesh; (b) structured cells.

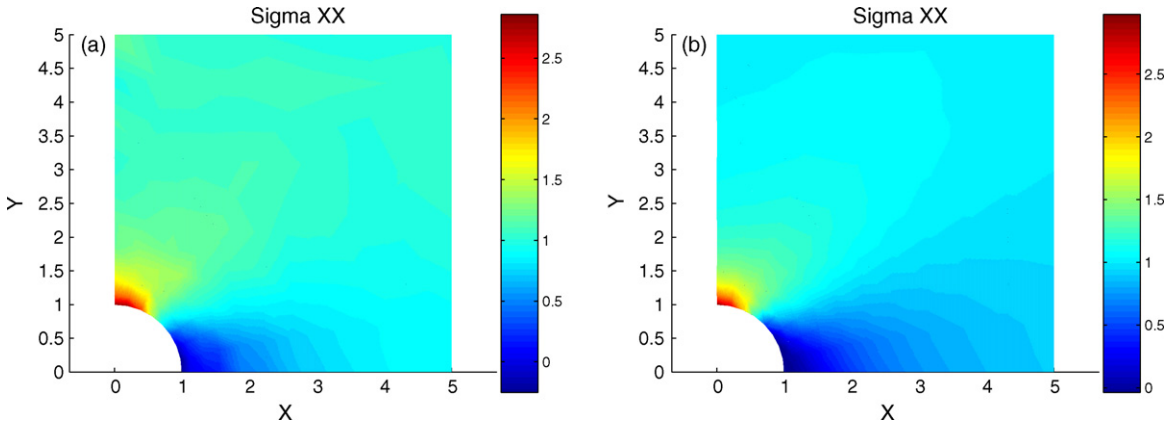

Fig. 28. Stress plot: (a) EFG σ_x; (b) exact σ_x.

where $K_I = \sigma\sqrt{\pi a}$ is the stress intensity factor, υ is Poisson's ratio and E is Young's modulus. ABCD is a square of $10\,\text{mm} \times 10\ \text{mm}$, $a = 100\ \text{mm}$; $E = 10^7\,\text{N/mm}^2$, $\upsilon = 0.3$ and $\sigma = 10^4\,\text{N/mm}^2$.

The geometry of the computational domain ABCD is shown in Fig. 29. Displacement of nodes on the bottom, right and top edges are prescribed by Eq. (124).

The crack is modeled by different techniques described in Section 2.9. The deformed configuration is plotted in Fig. 30 for the enriched partition of unity method (see Section 2.9.4) where the level set method is used to represent the geometry of the crack.

Numerical integration is performed on a background mesh of 20×20 rectangular elements. On each element, a 6×6 Gauss quadrature is adopted.

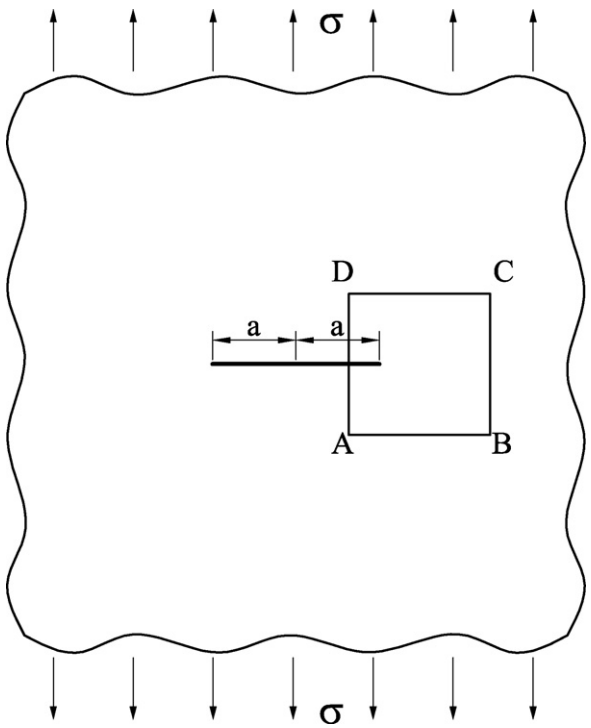

Fig. 29. Infinite cracked plate under remote tension: geometry and loads.

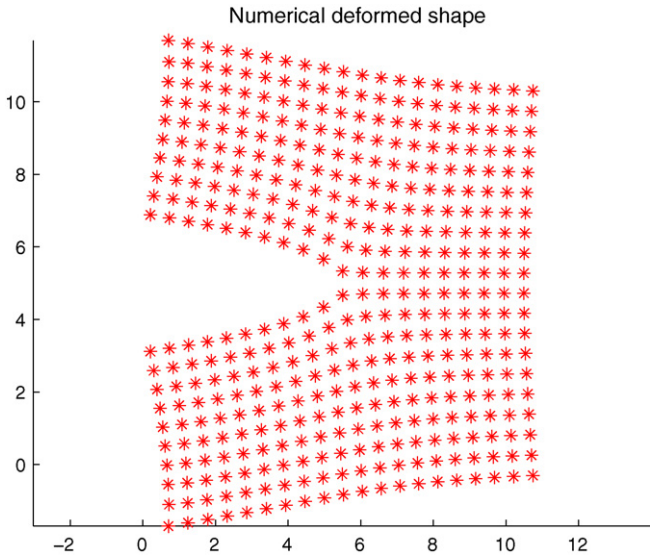

Fig. 30. Computed deformed configuration (scaled).

The error in the energy norm is illustrated in Fig. 31(a) for visibility, diffraction and transparency method. Note that we did not use any crack tip enrichment. The convergence rate of these methods are similar though the absolute error is smaller for the diffraction and the transparency method. Fig. 31(b) shows the normalized SIF K_I. Also for local convergence, the diffraction and transparency methods perform better. There is barely a difference in the results for the transparency and diffraction method.

The error in the energy norm is illustrated in Fig. 32(a) for some enriched methods (with crack tip enrichment), Sections 2.9.2 and 2.9.4. We have also included the results obtained with the visibility criterion in this figure. The methods that include the crack tip enrichment give more accurate results and a much better convergence rate which is expected of course. The most accurate results and the highest convergence rate of 0.94 are obtained with extrinsic MLS enrichment. However, the computational cost is higher since the enrichment is applied in the entire domain. The extrinsic PU enrichment gives a convergence rate of 0.86 and is only slightly less accurate than the extrinsic MLS enrichment. The intrinsic PU enrichment lies in between these two results. The same observation can be made for local convergence. The fact that the SIFs can be directly obtained is a major advantage of the extrinsic MLS enrichment and probably leads to more accurate results with respect to local convergence. Nevertheless, also

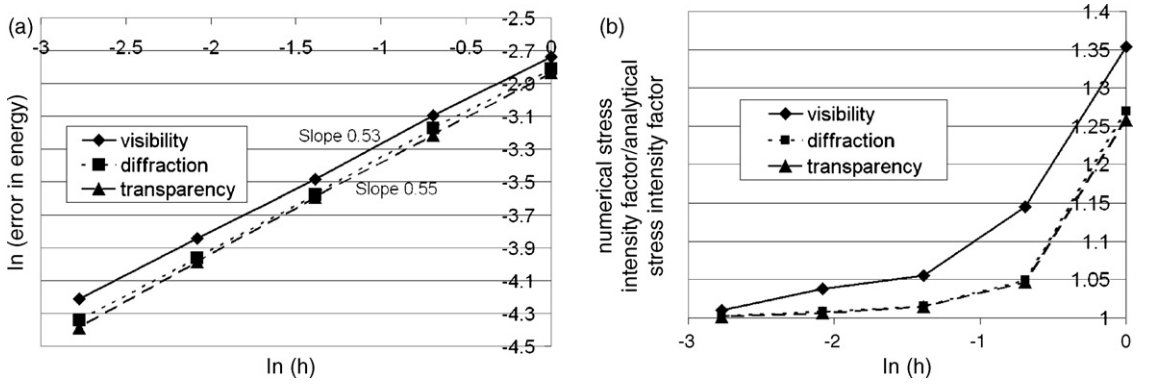

Fig. 31. (a) Error in the energy for the *mode I* problem using the visibility, diffraction and transparency criterion; (b) normalized stress intensity factor vs. h.

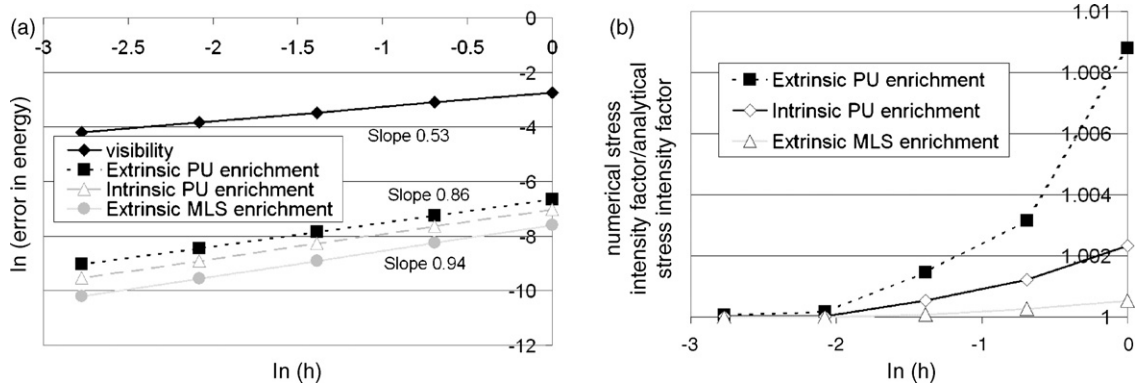

Fig. 32. (a) Error in the energy for the *mode I* problem using the intrinsic and extrinsic PU enrichment, respectively, and the extrinsic MLS enrichment; (b) normalized stress intensity factor vs. h.

the PU enrichment based method give excellent local convergence, see the scale of the y-axis in Figs. 31(b) and 32(b).

4.4. Infinite plate with a center inclusion

Consider an infinite plate with a center inclusion. For the computation, the square domain has dimensions 10×10 with a circular inclusion of radius 1 as shown in Fig. 33. The matrix properties are $E_1 = 1000$, $\nu_1 = 0.3$ whereas material characteristics of the inclusion are taken as $E_2 = 1$, $\nu_2 = 0.3$. The traction along the vertical direction is applied on the top edge while nodes along the bottom edge are constrained along the vertical direction. One more constraint is imposed to avoid rigid body modes.

This problem is solved with both PUM-enriched EFG and the extended finite element method XFEM (this XFEM code is written in Matlab and available from the website http://www.civil.gla.ac.uk/∼bordas). The particle arrangement is uniform (20×20) as well as the finite element mesh (20×20 four noded quadrilateral elements) and are given in Fig. 34. It is emphasized that the particles and the finite element mesh are independent of the shape and position of the inclusion.

For numerical integration of the EFG weak form, a background mesh is built with 6×6 Gauss quadrature for each cell, whereas, for the XFEM weak form, usual Gauss quadrature is adopted for elements which are not cut by the inclusion's boundary, while, for elements that *are* cut by the inclusion boundary, element partitioning is used.

Fig. 35 gives a comparison of the vertical displacement fields computed by both methods.

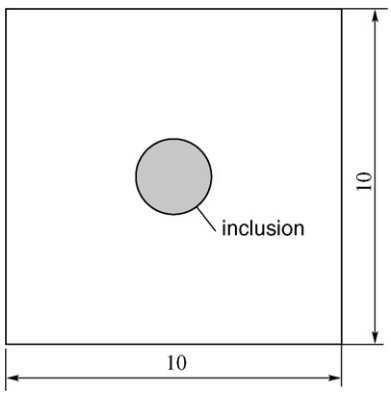

Fig. 33. Infinite plate with circular inclusion.

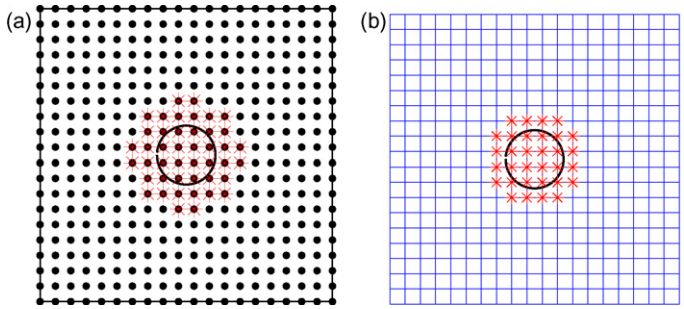

Fig. 34. Domain discretization with enriched nodes: (a) meshless particles; (b) finite element mesh.

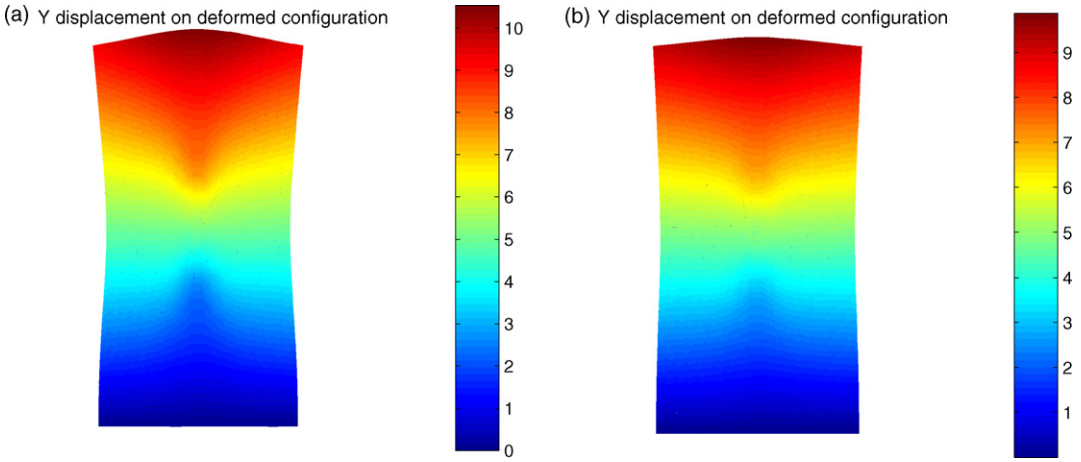

Fig. 35. Vertical displacement field: (a) EFG solution and (b) XFEM solution.

4.5. *Quasi-static crack propagation*

We study the cracked speciment of Fig. 36, also known as *double cantilever beam* (DCB). The dimensions are as follows: length $L = 300$ mm, height $h = 100$, initial crack length $a = 138$ mm, the load $P = 100\,\mathrm{N}$ and mechanical properties $E = 200\,\mathrm{GPa}$ and $\nu = 0,3$, and we assume plane strain conditions.

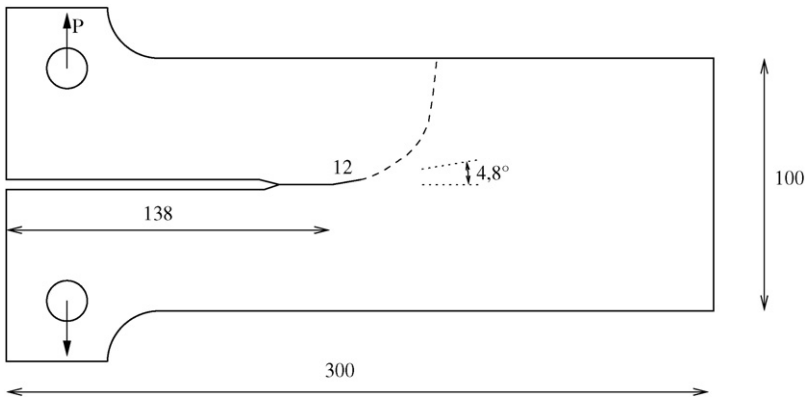

Fig. 36. Geometry of the *double cantilever beam*.

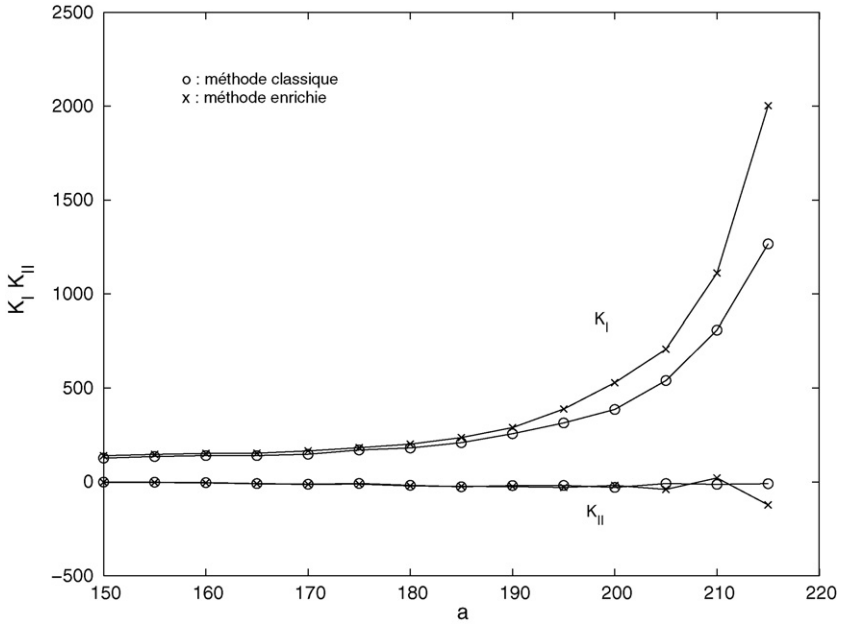

Fig. 37. Stress intensity factors for a crack growth increment $\Delta a = 5$ mm at each step.

A perturbation of the crack direction is introduced at the crack tip: angle 4.8° and length $dx = 12$ mm, which, physically, could be associated with the presence of a defect (void, inclusion) in the vicinity of the crack tip. The crack propagation phenomenon is studied experimentally by Sumi et al. [96] who show the unstable nature of the crack path, which either curves downwards or upwards. The crack path is represented by a dashed line in Fig. 36. Sumi [95] solve this problem by the finite element method and Fleming et al. [50] by meshfree methods (either with refined point distributions in the vicinity of the crack tip or with intrinsic enrichment of the MLS

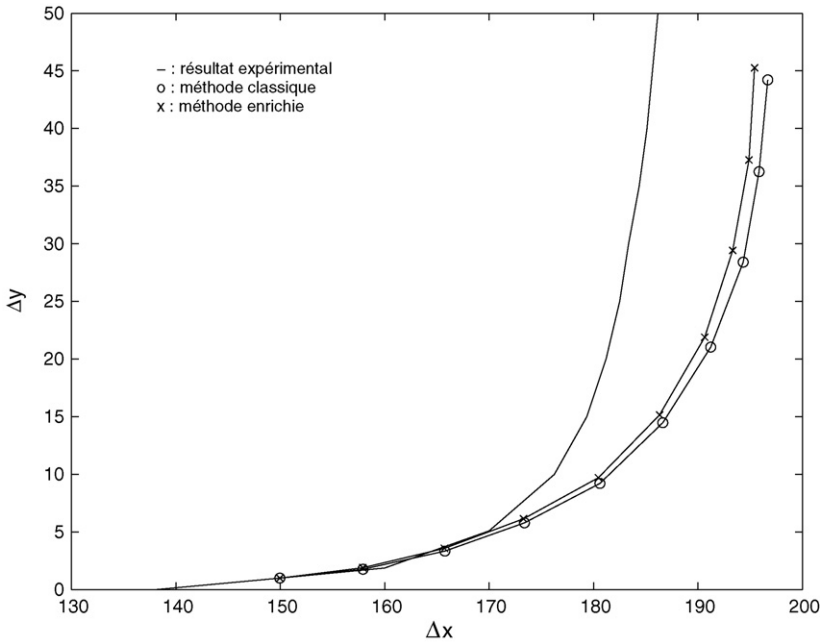

Fig. 38. Crack path ($\Delta a = 8$ mm).

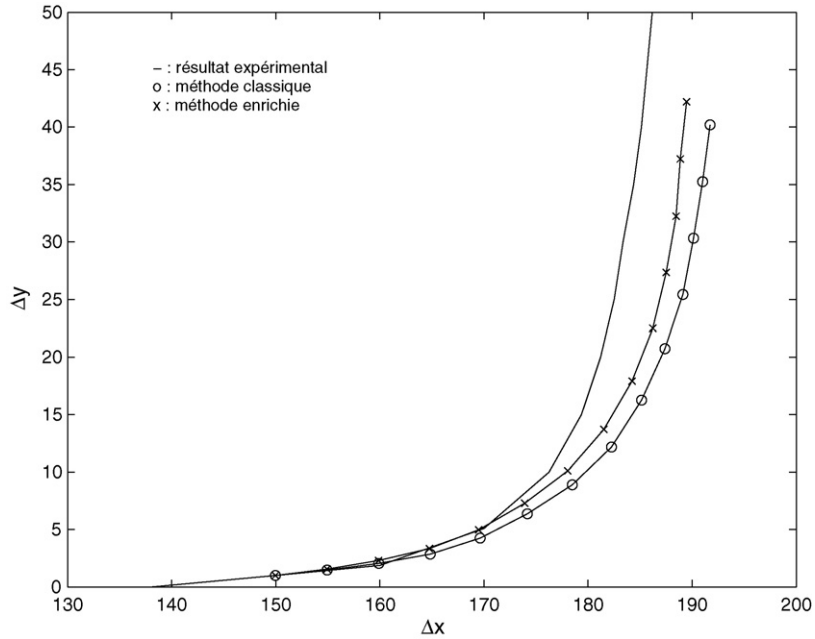

Fig. 39. Crack path ($\Delta a = 5$ mm).

basis). We solve the problem using both a standard EFG method (node distribution 49×17) and an enriched EFG method.

The evolution of the stress intensity factors during propagation are shown in Fig. 37 where we see that the crack is predominantly loaded in mode I. The slightly negative SIF is sufficient to curve the crack path in the direction of the top edge of the beam. We compare the experimental results to our numerical results in Figs. 38–40 for crack propagation increments, at each step of 8, 5 and 3 mm, respectively. We note that enrichment significantly improves the results.

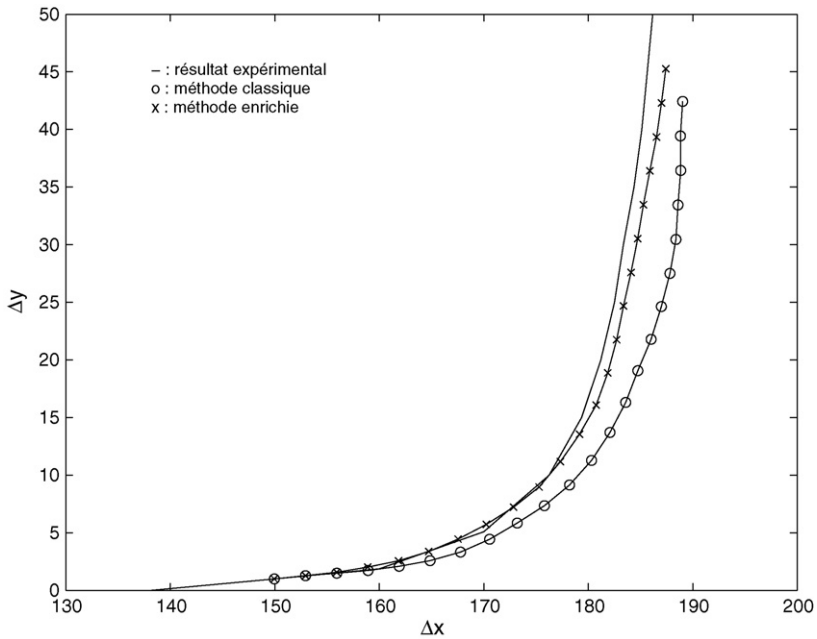

Fig. 40. Crack path ($\Delta a = 3$ mm).

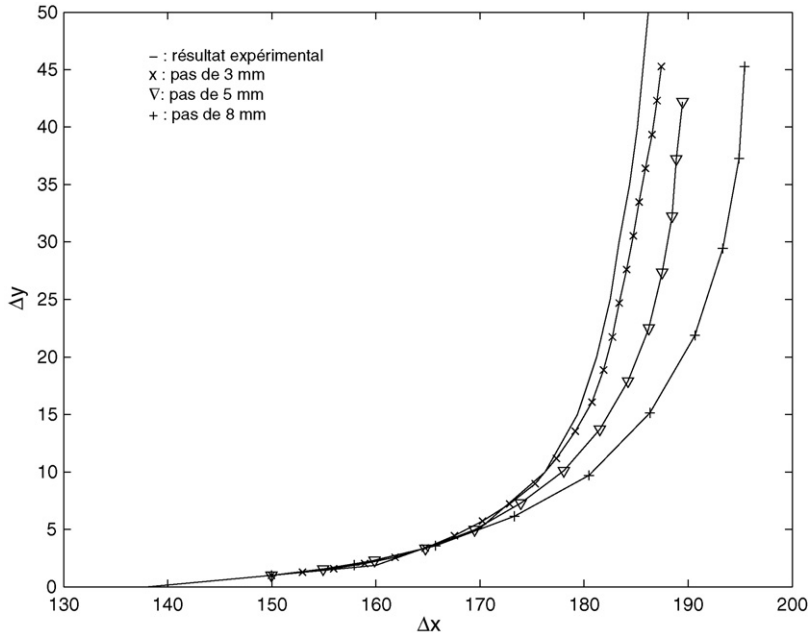

Fig. 41. Crack paths for various crack growth increments.

Fig. 41 shows the different crack paths obtained for three values of the crack growth increment. We note that the latter should not be "too small", otherwise, the nodal supports and J integral domains also become small which decreases the role of near-tip enrichment and poor accuracy in the stress intensity factors calculation, which exhibit spurious oscillations.

5. Conclusions

We have given an overview of meshless methods, that are based on a global weak form, with emphasis on implementation aspects. The meshless methods described in this manuscript are especially well-suited for Solid Mechanics applications and we have applied them to linear elastic material problems.

The meshless methods were classified into two categories, methods that are based on an intrinsic basis and methods based on an extrinsic basis. The latter class of methods were found especially useful for problems where information of the solution can be built into the approximation though it is also possible to include such information into certain intrinsic meshless methods. We have discussed properties, advantages and drawbacks of meshless methods compared to standard finite elements for simple examples. A summary is already given in the introduction of this manuscript.

Since standard meshless methods do not fulfill the so-called Kronecker–Delta property, essential boundary conditions cannot be enforced as easily as in finite element methods. We have summarized different opportunities how to incorporate essential boundary conditions and discussed two of them in more detail, the penalty method and the Lagrange multiplier method. Both methods are also described with respect to their implementation.

Different integration techniques used in meshless methods based on a global weak form were summarized: nodal integration, stress-point integration and background integration. We have decided to use background integration since background integration is best suited to the range of problems we studied here, i.e. small deformations. Background integration is the most accurate way of obtaining the discrete equations.

We have also given an overview of how to incorporate strong discontinuities, i.e. cracks, and weak discontinuities, i.e. material interfaces, into meshless methods. There are basically six ways how to handle cracks in MMs. The earliest methods use the visibility method that consider the crack as opaque. The diffraction and transparency is an extension of the visibility method that removes certain inaccuracies, i.e. undesired discontinuities. Newer approaches are based on the PU concept. In these methods, the information of the solution is incorporated in the approximation. That can be done intrinsically or extrinsically. We have also proposed an extrinsic enrichment based on an MLS technique.

With respect to LEFM, the main advantage of an MLS extrinsic enrichment is that SIFs are obtained directly from the analysis. The drawback of the method is that it uses global enrichment in contrast to local enrichment typically employed in PU based methods.

This paper is addressed to students or researchers who would like to have a quick start with meshfree methods, especially with respect to Solid Mechanics applications and Fracture Mechanics. Therefore, we have illustrated the capabilities of some methods for very simple examples. For the EFG-method, we have given and discussed a Matlab code for several problems discussed in Section 3. This includes the imposition of essential boundary conditions and different types of enrichment. The Matlab code can be downloaded from the following website: http://www.civil.gla.ac.uk/∼bordas/codes/efgMatlab/EFGMatlabCode.rar

As a glimpse into the future, it can be interesting to ask the question of the competitivity of meshfree methods compared to newly emerging techniques such as the extended finite element method. Meshfree techniques benefit from higher order continuity, which is very useful when solving fracture mechanics problems, since it provides a smoother stress distribution around the crack fronts.

Meshfree methods also seem to handle large deformations more naturally as the distortion of the cloud of points appears to have a smaller influence on accuracy (at least for Galerkin meshfree methods) than in finite elements. Adaptivity is also simplified in such problems. Despite these advantages, meshfree methods are more cumbersome to implement and computationally expensive.

Extended finite element methods (and partition of unity FEM techniques in general) appear as bringing together the advantages of the finite element method (simplicity, ease of implementation, robustness and computational efficiency) and some of those of meshfree methods (ability to treat discontinuities and singularities independently of the mesh).

An important drawback of the XFEM is the lack of smoothness of the resulting derivatives, as repeatedly admitted in the literature (e.g. [23,119–121]). What is more, XFEM, as FEM, cannot deal with distorted meshes very well, which would decrease its direct applicability to problems involving high mesh distortion.

The recent inception of the smoothed finite element method (SFEM) [108,110–112,125,126], regained interest in polygonal interpolation [127] and, in particular, the recent discovery of maximum entropy approximations [123,124] may help bringing some more flexibility to the XFEM. This could help to further bridge the gap between FEM and meshfree methods.

The SFEM is quite insensitive to mesh distortion and allows computations to be carried out on arbitrary polygonal meshes. Coupling the SFEM with the idea of quad (oct) -tree refinement proposed in [128] or *enrichment adaptivity* advocated in [119–121] appears promising as suggested by the preliminary results presented in the review paper [109].

Another item for future research concerns the mathematical theory of meshfree methods. A unified theory of the approximation properties and stability of meshfree methods has attracted a lot of recent interest, yet, a potential unified theory appears to be still eluding us. The influence of the shape and size of the domains of influence and point constellation on accuracy and stability in MM based on local/global weak forms or collocation methods alike, remains unclear.

Acknowledgements

The third author would like to thank the input on meshfree methods from Dr Thomas Zimmermann during his stay at the on meshfree methods Laboratory of Structural and Continuum Mechanics, Swiss Institute of Technology, Lausanne, Switzerland, from 2003 to 2006.

References

[1] S.N. Atluri, The Meshless Local Petrov–Galerkin (MLPG) Method, Tech Science Press, 2002.
[2] S.N. Atluri, S. Shen, The meshless local Petrov–Galerkin (MLPG) method: a simple and less-costly alternative to the finite element and boundary element methods, Comput. Model Eng. Sci. 3 (2002) 11–51.
[3] S.N. Atluri, T. Zhu, A new meshless local Petrov–Galerkin (MLPG) approach in computational mechanics, Comput. Mech. 22 (1998) 117–127.
[4] S.N. Atluri, T. Zhu, The meshless local Petrov–Galerkin (MLPG) approach for solving problems in elasto-statics, Comput. Mech. 25 (2000) 169–179.
[5] S. Beissel, T. Belytschko, Nodal integration of the element-free Galerkin method, Comput. Methods Appl. Mech. Eng. 139 (1996) 49–74.
[6] T. Belytschko, T. Black, Elastic crack growth in finite elements with minimal remeshing, Int. J. Numer. Methods Eng. 45 (1999) 601–620.

[7] T. Belytschko, M. Fleming, D. Organ, Y. Krongauz, W.K. Liu, Smoothing and accelerated computations in the element free Galerkin method, J. Comput. Appl. Math. 74 (1996) 111–126.
[8] T. Belytschko, L. Gu, Y.Y. Lu, Fracture and crack growth by element-free Galerkin methods, Model. Simul. Mater. Sci. Eng. 2 (1994) 519–534.
[9] T. Belytschko, Y. Guo, W.K. Liu, S.P. Xiao, A unified stability analysis of meshfree particle methods, Int. J. Numer. Methods Eng. 48 (2000) 1359–1400.
[10] T. Belytschko, Y. Krongauz, J. Dolbow, C. Gerlach, On the completeness of meshfree particle methods, Int. J. Numer. Methods Eng. 43 (1998) 785–819.
[11] T. Belytschko, Y. Krongauz, M. Fleming, D. Organ, W.K. Liu, Smoothing and accelerated computations in the element-free Galerkin method, J. Comput. Appl. Math. 74 (1996) 111–126.
[12] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl, Meshless methods: an overview and recent developments, Comput. Methods Appl. Mech. Eng. 139 (1996) 3–47.
[13] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl, Meshless methods: an overview and recent developments, Comput. Methods Appl. Mech. Eng. 139 (1996) 3–47.
[14] T. Belytschko, Y.Y. Lu, L. Gu, Element-free Galerkin methods, Int. J. Numer. Methods Eng. 37 (1994) 229–256.
[15] T. Belytschko, Y.Y. Lu, L. Gu, Crack propagation by element-free Galerkin methods, Eng. Fract. Mech. 51 (2) (1995) 295–315.
[16] T. Belytschko, Y.Y. Lu, Element-free Galerkin methods for static and dynamic fracture, Int. J. Solids Struct. 32 (1995) 2547–2570.
[17] T. Belytschko, Y.Y. Lu, L. Gu, Element-free Galerkin methods, Int. J. Numer. Methods Eng. 37 (1994) 229–256.
[18] T. Belytschko, D. Organ, Y. Krongauz, A coupled finite element–element-free Galerkin method, Comput. Mech. 17 (3) (1995) 186–195.
[19] T. Belytschko, M. Tabbara, Dynamic fracture using element-free Galerkin methods, Int. J. Numer. Methods Eng. 39 (6) (1996) 923–938.
[20] J. Bonet, S. Kulasegaram, Correction and stabilization of smooth particle hydrodynamics methods with application in metal forming simulations, Int. J. Numer. Methods Eng. 47 (6) (2000) 1189–1214.
[21] J. Bonet, T. Lok, Variational and momentum preservation aspects of smooth particle hydrodynamic formulations., Comput. Methods Appl. Mech. Eng. 180 (1–2) (1999) 97–115.
[22] S. Bordas, J.G. Conley, B. Moran, J. Gray, E. Nichols, A simulation-based design paradigm for complex cast components, Eng. Comput. 23 (1) (2007) 25–37.
[23] S. Bordas, B. Moran, Enriched finite elements and level sets for damage tolerance assessment of complex structures, Eng. Fract. Mech. 73 (2006) 1176–1201.
[24] S. Bordas, V.P. Nguyen, C. Dunant, H. Nguyen-Dang, A. Guidoum, An extended finite element library, Int. J. Numer. Methods Eng. 71 (2008) 703–732.
[25] A. Carpinteri, Post-peak and post-bifurcation analysis of cohesive crack propagation, Eng. Fract. Mech. 32 (1989) 265–278.
[26] A. Carpinteri, A scale-invariant cohesive crack model for quasi-brittle materials, Eng. Fract. Mech. 69 (2002) 207–217.
[27] A. Carpinteri, G. Ferro, G. Ventura, The partition of unity quadrature in meshless methods, Int. J. Numer. Methods Eng. 54 (2002) 987–1006.
[28] A. Carpinteri, G. Ferro, G. Ventura, The partition of unity quadrature in element-free crack modelling, Comput. Struct. 81 (2003) 1783–1794.
[29] J.S. Chen, C. Pan, C.M.O.L. Rogue, H.P. Wang, A Lagrangian reproducing kernel particle method for metal forming analysis, Comput. Mech. 22 (1998) 289–307.
[30] J.S. Chen, C. Pan, C.T. Wu, W.K. Liu, Reproducing kernel particle methods for large deformation analysis of nonlinear structures, Comput. Methods Appl. Mech. Eng. 139 (1996) 195–227.
[31] J.S. Chen, C.T. Wu, S. Yoon, Y. You, A stabilized conforming nodal integration for Galerkin meshfree-methods, Int. J. Numer. Methods Eng. 50 (2001) 435–466.
[32] H.J. Chung, T. Belytschko, An error estimate in the EFG method, Comput. Mech. 21 (1998) 91–100.
[33] H.J. Chung, G.H. Lee, C.K. Choi, Adaptive nodal generation with the element-free Galerkin method, Struct. Eng. Mech. 10 (2000) 635–650.
[34] L. Cueto-Felgueroso, I. Colominas, G. Mosqueira, F. Navarrina, M. Casteleiro, On the Galerkin formulation of smoothed particle hydrodynamics, Int. J. Numer. Methods Eng. 60 (2004) 1475–1512.
[35] G.A. Dilts, Moving least square particle hydrodynamics. I. Consistency and stability, Int. J. Numer. Methods Eng. 44 (2000) 1115–1155.
[36] G.A. Dilts, Moving least square particle hydrodynamics. II. Conservation and boundaries, Int. J. Numer. Methods Eng. (2000) 1503–1524.
[37] J. Dolbow, T. Belytschko, An introduction to programming the meshless element-free Galerkin method, Arch. Comput. Mech. 5 (3) (1998) 207–241.
[38] J. Dolbow, T. Belytschko, Numerical integration of the Galerkin weak form in meshfree methods, Comput. Mech. 23 (1999) 219–230.
[39] C.A. Duarte, J.T. Oden, An hp adaptive method using clouds, Comput. Methods Appl. Mech. Eng. 139 (1996) 237–262.
[40] C.A. Duarte, J.T. Oden, Hp clouds–an hp meshless method, Numerical Methods for Partial Differential Equations 12 (1996) 673–705.
[41] M. Duflot, A meshless method with enriched weight functions for three-dimensional crack propagation, Int. J. Numer. Methods Eng. 65 (2006) 1970–2006.
[42] M. Duflot, H. Nguyen-Dang, Dual analysis by a meshless method, Commun. Numer. Methods Eng. 18 (2002) 621–631.
[43] M. Duflot, H. Nguyen-Dang, A truly meshless method based on a moving least squares quadrature, Commun. Numer. Methods Eng. 18 (2002) 441–449.
[44] M. Duflot, H. Nguyen-Dang, A meshless method with enriched weight functions for fatigue crack growth, Int. J. Numer. Methods Eng. 59 (2004) 1945–1961.
[45] C. Dunant, P. Nguyen, M. Belgasmia, S. Bordas, A. Guidoum, H. Nguyen-Dang, Architecture trade-offs of including a mesher in an object-oriented extended finite element code, Eur. J. Comput. Mech. (16) (2007) 237–258, Special issue on the extended finite element method (XFEM).
[46] C.T. Dyka, R.P. Ingel, An approach for tensile instability in smoothed particle hydrodynamics, Comput. Struct. 57 (1995) 573–580.

[47] S. Fernández-Méndez, P. Díez, A. Huerta, Convergence of finite elements enriched with meshless methods, Numer. Math. 96 (2003) 43–59.
[48] S. Fernández-Méndez, A. Huerta, Imposing essential boundary conditions in mesh-free methods, Comput. Methods Appl. Mech. Eng. 193 (2004) 1257–1275.
[49] S. Fernández-Méndez, A. Huerta, Coupling finite elements and particles for adaptivity: an application to consistently stabilized convection–diffusion, in: M. Griebel, M.A. Schweitzer (Eds.), Meshfree Methods for Partial Differential Equations, Lecture Notes in Computational Science and Engineering, vol. 26, Springer-Verlag, Berlin, 2002, pp. 117–129 (Papers from the International Workshop, Universität Bonn, Germany, September 11–14, 2001).
[50] M. Fleming, Y.A. Chu, B. Moran, T. Belytschko, Enriched element-free Galerkin methods for crack tip fields, Int. J. Numer. Methods Eng. 40 (1997) 1483–1504.
[51] L. Gavete, J.L. Cuesta, A. Ruiz, A procedure for approximation of the error in the EFG method, Int. J. Numer. Methods Eng. 53 (2002) 677–690.
[52] L. Gavete, S. Falcën, A. Ruiz, An error indicator for the element-free Galerkin method, Eur. J. Mech. A/Solids 20 (2001) 327–341.
[53] L. Gavete, M.L. Gavete, B. Alonso, A.J. Martën, A posteriori error approximation in EFG method, Int. J. Numer. Methods Eng. 58 (2003) 2239–2263.
[54] R.A. Gingold, J.J. Monaghan, Smoothed particle hydrodynamics: theory and application to non-spherical stars, Monthly Notices R. Astron. Soc. 181 (1977) 375–389.
[55] S. Hao, W.K. Liu, Moving particle finite element method with superconvergence: nodal integration formulation and applications, Comput. Methods Appl. Mech. Eng. 195 (2006) 6059–6072.
[56] A. Huerta, S. Fernández-Méndez, Enrichment and coupling of the finite element and meshless methods, Int. J. Numer. Methods Eng. 48 (11) (2000) 1615–1636.
[57] A. Huerta, T. Belytschko, S. Fernandez-Mendez, T. Rabczuk, Encyclopedia of Comput. Mech., John Wiley and Sons, 2004 (Chapter Meshfree Methods).
[58] A. Huerta, S. Fernández-Méndez, W.K. Liu, A comparison of two formulations to blend finite elements and mesh-free methods, Comput. Methods Appl. Mech. Eng. 193 (2004) 1105–1117.
[59] U. Hëussler-Combe, C. Korn, An adaptive approach with the element-free Galerkin method, Comput. Methods Appl. Mech. Eng. 162 (1998) 203–222.
[60] S.R. Idelsohn, E. Oñate, F. Del Pin, The particle finite element method: a powerful tool to solve incompressible flows with free surfaces and breaking waves, Int. J. Numer. Methods Eng. 61 (2004) 964–989.
[61] G.R. Johnson, S.R. Beissel, Normalized smoothing functions for sph impact computations, Int. J. Numer. Methods Eng. 39 (1996) 2725–2741.
[62] G.R. Johnson, S.R. Beissel, R.A. Stryk, A generalized particle algorithm for high velocity impact computations, Comput. Mech. 25 (2000) 245–256.
[63] Y. Krongauz, T. Belytschko, EFG approximation with discontinuous derivatives, Int. J. Numer. Methods Eng. 41 (1998) 1215–1233.
[64] Y. Krongauz, T. Belytschko, EFG approximation with discontinuous derivatives, Int. J. Numer. Methods Eng. 41 (7) (1998) 1215–1233.
[65] P. Lancaster, K. Salkauskas, Surfaces generated by moving least squares methods, Math. Comput. 37 (1981) 141–158.
[66] T. Laouar, P. Villon, Adaptive analysis for the diffuse element method, in: S. Idelsohn, E. Oëate, E. Dvorkin (Eds.), Comput. Mech., CIMNE, Barcelona, 1998.
[67] C.K. Lee, C.E. Zhou, On error estimation and adaptive refinement for element free Galerkin method. Part I. Stress recovery and a posteriori error estimation, Comput. Struct. 82 (2004) 413–428.
[68] C.K. Lee, C.E. Zhou, On error estimation and adaptive refinement for element free Galerkin method. Part II. Adaptive refinement, Comput. Struct. 82 (2004) 429–443.
[69] G.H. Lee, H.J. Chung, C.K. Choi, Adaptive crack propagation analysis with the element-free Galerkin method, Int. J. Numer. Methods Eng. 56 (2003) 331–350.
[70] Q. Li, J. Soric, T. Jarak, S.N. Atluri, A locking-free meshless local Petrov–Galerkin formulation for thick and thin plates, J. Comput. Phys. 208 (2005) 116–133.
[71] L.D. Libersky, A.G. Petscheck, T.C. Carney, J.R. Hipp, F.A. Allahdadi, High strain Lagrangian hydrodynamics, J. Comput. Phys. 109 (1993) 67–75.
[72] T.J. Liszka, C.A. Duarte, W.W. Tworzydlo, Hp-meshless cloud method, Comput. Methods Appl. Mech. Eng. 139 (1996) 263–288.
[73] W.K. Liu, S. Jun, Y.F. Zhang, Reproducing kernel particle methods, Int. J. Numer. Methods Eng. 20 (1995) 1081–1106.
[74] W.K. Liu, S. Li, W. Han, H. Lu, J. Cao, Reproducing kernel element method. Part I. Theoretical formulation, Comput. Methods Appl. Mech. Eng. 193 (2004) 933–951.
[75] R. Loehner, C. Sacco, E. Oñate, A finite point method for compressible flow, Int. J. Numer. Methods Eng. 53 (2002) 1765–1779.
[76] H. Lu, J.S. Chen, Adaptive Galerkin particle method, Lect. Notes Comput. Sci. Eng. 26 (2002) 251–267.
[77] L.B. Lucy, A numerical approach to the testing of the fission hypothesis, Astron. J. 82 (1977) 1013–1024.
[78] J.M. Melenk, I. Babuška, The partition of unity finite element method: Basic theory and applications, Comput. Methods Appl. Mech. Eng. 139 (1996) 289–314.
[79] N. Moës, J. Dolbow, T. Belytschko, A finite element method for crack growth without remeshing, Int. J. Numer. Methods Eng. 46 (1999) 131–150.
[80] J.J. Monaghan, Why particle methods work, SIAM J. Sci. Stat. Comput. 3 (3) (1982) 422–433.
[81] J.J. Monaghan, An introduction to SPH, Comput. Phys. Commun. 48 (1) (1988) 89–96.
[82] E. Oñate, S. Idelsohn, A mesh-free finite point method for advective-diffusive transport and fluid flow problems, Comput. Mech. 21 (4/5) (1998) 283–292.

[83] E. Oñate, S. Idelsohn, O.C. Zienkievicz, R.L. Taylor, A finite point method in computational mechanics: applications to convective transport and fluid flow, Int. J. Numer. Methods Eng. 39 (1996) 3839–3866.
[84] D. Organ, M. Fleming, T. Terry, T. Belytschko, Continuous meshless approximations for nonconvex bodies by diffraction and transparency, Comput. Mech. 18 (1996) 225–235.
[85] M.A. Puso, J.S. Chen, E. Zywicz, W. Elmer, Meshfree and finite element nodal integration methods, Int. J. Numer. Methods Eng. 74 (2008) 416–446.
[86] T. Rabczuk, T. Belytschko, An adaptive continuum/discrete crack approach for meshfree particle methods, Latin Am. J. Solids Struct. 1 (2003) 141–166.
[87] T. Rabczuk, T. Belytschko, Adaptivity for structured meshfree particle methods in 2D and 3D, Int. J. Numer. Methods Eng. 63 (11) (2005) 1559–1582.
[88] P.W. Randles, L.D. Libersky, Recent improvements in sph modeling of hypervelocity impact, Int. J. Impact Eng. 20 (1997) 525–532.
[89] P.W. Randles, L.D. Libersky, Normalized sph with stress points, Int. J. Numer. Methods Eng. 48 (2000) 1445–1462.
[90] D. Shepard, A two-dimensional function for irregularly spaced points, in: Proceedings of the 23rd ACM National Conference, 1968, pp. 517–524.
[91] T. Strouboulis, K. Copps, I. Babuška, The generalized finite element method: an example of its implementation and illustration of its performance, Int. J. Numer. Methods Eng. 47 (8) (2000) 1401–1417.
[92] N. Sukumar, D.L. Chopp, N. Moës, T. Belytschko, Modelling holes and inclusions by level sets in the extended finite element method, Comput. Methods Appl. Mech. Eng. 190 (2000) 6183–6200.
[93] N. Sukumar, D.L. Chopp, B. Moran, Extended finite element method and fast marching method for three-dimensional fatigue crack propagation, Eng. Fract. Mech. 70 (2003) 29–48.
[94] N. Sukumar, N. Moës, B. Moran, T. Belytschko, Extended finite element method for three-dimensional crack modelling, Int. J. Numer. Methods Eng. 48 (2000) 1549–1570.
[95] Y. Sumi, Computational crack path prediction, Theor. Appl. Fract. Mech. 4 (1985) 149–156.
[96] Y. Sumi, S. Nemat-Nasser, L.M. Keer, On crack path stability in a finite body, Eng. Fract. Mech. 22 (1985) 759–771.
[97] L.W. Swegle, D.L. Hicks, S.W. Attaway, Smooth particle hydrodynamics stability analysis, J. Comput. Phys. 116 (1995) 123–134.
[98] G. Ventura, An augmented Lagrangian approach to essential boundary conditions in meshless methods, Int. J. Numer. Methods Eng. 53 (2002) 825–842.
[99] G. Ventura, J.X. Xu, T. Belytschko, A vector level set method and new discontinuity approximations for crack growth by EFG, Int. J. Numer. Methods Eng. 54 (2002) 923–944.
[100] L.P. Vila, On particle weighted methods and smooth particle hydrodynamics, Math. Models Methods Appl. Sci. 9 (2) (1999) 161–209.
[101] S.P. Xiao, T. Belytschko, Material stability analysis of particle methods, Adv. Comput. Math. 23 (2005) 171–190.
[102] Y. You, J.S. Chend, H. Lu, Filters, reproducing kernel and adaptive meshfree method, Comput. Mech. 31 (2003) 316–326.
[103] Z.Q. Zhang, J.X. Zhou, X.M. Wang, Y.F. Zhang, L. Zhang, Investigations on reproducing kernel particle method enriched by partition of unity and visibility criterion, Comput. Mech. (2004), http://www.springerlink.com/link.asp?id=pxqfcbkr3ggfaf2b.
[104] O.C. Zienkiewicz, J.Z. Zhu, A simple error estimator and adaptive procedure for practical engineering analysis, Int. J. Numer. Methods Eng. 24 (1987) 337–357.
[105] T. Rabczuk, T. Belytschko, S.P. Xiao, Stable particle methods based on Lagrangian kernels, Comput. Methods Appl. Mech. Eng. 193 (12–14) (2004) 1035–1063.
[106] A. Huerta, T. Belytschko, S. Fernandez-Mendez, T. Rabczuk, Meshfree Methods, in: E. Stein, R. De Borst, T.J.R. Hughes (Eds.), Encyclopedia of Computational Mechanics, Wiley & Sons, 2004.
[107] T. Rabczuk, S.P. Xiao, M. Sauer, Coupling of meshfree methods with finite elements: Basic concepts and test results, Commun. Numer. Methods Eng. 22 (10) (2006) 1031–1065.
[108] H. Nguyen-Xuan, S. Bordas, H. Nguyen-Dang. Smooth finite elements: convergence, accuracy and properties, IJNME 74 (2008) 175–208.
[109] S. Bordas, T. Rabczuk, H. Nguyen-Xuan, S. Natarajan, Review and Recent Developments on the Smoothed Finite Element Method (SFEM) and First Results in the Smoothed eXtended Finite Element Method (SmXFEM), Comput. Struct. (2007), http://www.civil.gla.ac.uk/∼bordas/pdf/.
[110] G.R. Liu, K.Y. Dai, T.T. Nguyen. A smoothed finite element for mechanics problems, CM 39 (2007) 859–877.
[111] G.R. Liu, T.T. Nguyen, K.Y. Dai, K.Y. Lam, Theoretical aspects of the smoothed finite element method (SFEM), IJNME 71 (8) (2007) 902–930.
[112] H. Nguyen-Xuan, S. Bordas, H. Nguyen-Dang, A smoothed finite element method for plate analysis, CMAME 197 (13–16) (2008) 1184–1203.
[113] T. Rabczuk, T. Belytschko, Cracking particles: a simplified meshfree method for arbitrary evolving cracks, Int. J. Numer. Methods Eng. 61 (13) (2004) 2316–2343.
[114] T. Rabczuk, P.M.A. Areias, A meshfree thin shell for arbitrary evolving cracks based on an external enrichment, Comput. Model. Eng. Sci. 16 (2) (2006) 115–130.
[115] T. Rabczuk, G. Zi, A meshfree method based on the local partition of unity for cohesive cracks, Comput. Mech. 39 (6) (2007) 743–760.
[116] T. Rabczuk, T. Belytschko. A three dimensional large deformation meshfree method for arbitrary evolving cracks, Comput. Methods Appl. Mech. Eng. 196 (29–30) (2007), 2777–2799.
[117] G. Zi, T. Rabczuk, W.A. Wall, Extended Meshfree Methods without Branch Enrichment for Cohesive Cracks, Comput. Mech. 40 (2) (2007) 367–382.
[118] T. Rabczuk, P.M.A. Areias, T. Belytschko, A simplified meshfree methods for shear bands with cohesive surfaces, Int. J. Numer. Methods Eng. 69 (5) (2007) 993–1021.

[119] S. Bordas, M. Duflot, Derivative recovery and a posteriori error estimate for extended finite elements, Comput. Methods Appl. Mech. Eng. 196 (35–36) (2007) 3381–3399.
[120] S. Bordas, M. Duflot, P. Le, A simple a posteriori error estimator for the extended finite element method, CNME 24 (2008) 961–971.
[121] M. Duflot, S. Bordas, An extended global recovery procedure for a posteriori error estimation in extended finite element methods, IJNME, in press, http://www.civil.gla.ac.uk/∼bordas/pdf/duflotbordas.pdf.
[122] E. Oñate, Possibilities of finite calculus in computational mechanics, Int. J. Numer. Methods Eng. 60 (1) (2004) 255–281.
[123] N. Sukumar, Construction of Polygonal Interpolants: A Maximum Entropy Approach, IJNME 61 (12) (2004) 2159–2181.
[124] M. Arroyo, M. Ortiz, Local maximum-entropy approximation schemes: a seamless bridge between finite elements and meshfree methods, Int. J. Numer. Methods Eng. 65 (13) (2006) 2167–2202.
[125] H. Nguyen-Xuan, S. Bordas, H. Nguyen-Dang, Selective integration in the smoothed finite element method, Commun. Numer. Methods Eng. (2008), in press.
[126] N. Nguyen, H. Nguyen-Xuan, S. Bordas, H. Nguyen-Dang, A locking free smoothed finite element method for shells with high tolerance to mesh distortion, CMAME (2007), in press (http://www.civil.gla.ac.uk/∼bordas/pdf/nxhbordas4.pdf).
[127] N. Sukumar, E.A. Malsch, Recent Advances in the Construction of Polygonal Finite Element Interpolants, Arch. Computat. Methods Eng. 13 (1) (2006) 129–163.
[128] A. Tabarraei, N. Sukumar, Extended finite element method on polygonal and quadtree meshes, Comput. Methods Appl. Mech. Eng. 197 (5) (2008) 425–435.