Comput Mech (2014) 53:1087–1103
DOI 10.1007/s00466-013-0950-8

ORIGINAL PAPER

Extended particle difference method for weak and strong discontinuity problems: part I. Derivation of the extended particle derivative approximation for the representation of weak and strong discontinuities

Young-Cheol Yoon · Jeong-Hoon Song

Received: 8 August 2013 / Accepted: 12 November 2013 / Published online: 26 November 2013

Abstract In this paper, the extended particle derivative approximation (EPDA) scheme is developed to solve weak and strong discontinuity problems. In this approximation scheme, the Taylor polynomial is extended with enrichment functions, i.e. the step function, the wedge function, and the scissors function, based on the moving least squares procedure in terms of nodal discretization. Throughout numerical examples, we demonstrate that the EPDA scheme reproduces weak and strong discontinuities in a singular solution quite well, and effectively copes with the difficulties in computing the derivatives of the singular solution. The governing partial differential equations, including the interface conditions, are directly discretized in terms of the EPDA scheme, and the total system of equations is derived from the formulation of difference equations which is constructed at the nodes and points representing the problem domain and the interfacial boundary, respectively.

Keywords Extended particle difference method (EPDM) · Extended particle derivative approximation (EPDA) · Interfacial singularity · Strong formulation · Second-order accuracy

Y.-C. Yoon
Department of Civil Engineering, Myongji College,
356-1 Hongeun-Dong, Seodaemoon-Ku,
Seoul 120-776, South Korea

J.-H. Song (✉)
Department of Civil and Environmental Engineering,
University of South Carolina, Columbia, SC 29208, USA
e-mail: jhsong@cec.sc.edu

1 Introduction

Singularities, such as discontinuities, in partial differential equation (PDE) solutions give rise to difficulties in developing new numerical schemes. Discontinuities such as material interfaces and layered singular sources generate singularities that involve jumps in the solution and its derivative fields; these frequently appear as interface conditions in governing equations for the given PDE. The existence of an interface condition can spoil the systematic construction of a numerical scheme. For example, weak form based numerical methods such as the finite element method (FEM), the finite volume method (FVM), and the conventional meshfree method are associated with serious difficulties in integrating the weak form and in constructing the interpolation function due to an arbitrarily placed interfacial boundary. In other words, a mesh or grid structure for interpolation and numerical quadrature can represent a major obstacle when attempting to solve the PDE. Unless a mesh or grid aligned with the interface geometry is used, it is difficult to avoid the aforementioned difficulties. Furthermore, when the interface has topology change physics with respect to time, the computational difficulties can become severe. For these reasons, the development of a numerical method which is independent of a structured mesh or grid has drawn much attention. For instance, the extended finite element method (XFEM) [1–3], developed in the framework of FEM, has been a very popular numerical method for solving discontinuity problems despite the fact that it remains inconvenient as regards the numerical integration of a cell cut by a discontinuity line. In addition, strong formulations that directly discretize a PDE such as the finite difference method (FDM) have been studied for their applicability to these types of cumbersome discontinuity problems.

Developing a simplex jump modeling technique based on strong formulation is challenging. Many FDM user groups

have suffered from deriving difference schemes in the vicinity of an interface that induces a singularity in the solution fields. The immersed boundary method (IBM) [4] was developed in the context of FDM, considering a singular source as a Dirac's delta function and directly discretizing the delta function using a linear hat function. The immersed interface method (IIM), an improved version of the IBM, was proposed by Leveque and Li [5]. In this method [5], the interface condition is strategically immersed in the difference equation. The immersion process manipulates the truncation error to obtain the geometrical information for interface modeling, but this greatly complicates the difference equation. Thus, it is difficult to say that the IIM is completely free from grid dependency, as it involves a great deal of computation for interface modeling. On the other hand, the meshfree method or the particle method can offer important insight into how to overcome the grid dependency. The merits of the meshfree method are enhanced when it is combined with the strong formulation rather than the weak formulation. Specifically, the moving least squares method used in constructing meshfree interpolation may be directly applied to derive the derivative approximation, as in this study; it provides an idea regarding how to take into account the singularity in solution fields in the framework of the meshfree method. At the same time, the simplicity of the high order derivative calculation should be ensured for the direct discretization of governing equations, as the strong formulation naturally requires high order derivatives. However, the computation of the high order derivatives of the moving least squares approximation is not easy; moreover, the full differentiation of the approximation is quite burdensome. Thus, computational efficiency has been a major concern in this class of methods. In this scenario, the diffuse derivative technique may be of primary interest.

Two approaches are common when computing the derivatives of meshfree approximation. One was introduced in the work of Nayroles et al. [6] based on the diffuse element method (DEM), and the other was introduced by Krongauz and Belytschko [7,8] as in the element free Galerkin method (EFGM) [9] based on moving least squares approximation. Although the reproducing Kernel particle method (RKPM) by Liu et al. [10] provided meshfree approximation based on a more solid mathematical foundation, the final discrete form of the approximation completely matches that of the EFGM. In fact, the RKPM approximation is obtained by correcting the kernel approximation of the smoothed particle hydrodynamics (SPH) [11] to enforce the reproducing property of the polynomials up to the order of consistency. Note that other meshfree methods can be classified into one of the aforementioned approaches according to the methodology of the derivative calculation. The former adopts the 'diffuse' derivative technique, in which the polynomial basis part, instead of the total approximation, is only differentiated for the derivative calculation despite the fact that it cannot be supported on a sufficient mathematical basis. On the other hand, the latter method employs the 'full' derivative of the moving least squares approximation, in which the total approximation including the polynomial basis is differentiated. Compared to the diffuse derivative, the full derivative considerably increases the amount of computation. However, when the diffuse derivative is incorporated into the weak formulation, it decreases the accuracy because it does not yield a mathematically exact derivative of the approximation. Krongauz and Belytschko [7,8] discussed the lack of integrability of the test function in the Petrov–Galerkin formulation, which uses different forms for the test function and the trial function. Nevertheless, the diffuse derivative approximation is consistent in the sense that it is still based on the Taylor series such that it can approximate functions up to the order of consistency. When combining it with the strong formulation, one can take advantage of fast computation when discretizing PDEs. Thus, Kim and Kim [12] and Lee and Yoon [13] presented a meshfree point collocation method combined with the diffuse derivative approximation, reporting the robustness and accuracy of the method for solving fluid and solid problems, respectively. Oñate et al. [14,15] and Aluru [16] proposed a collocation method with moving least squares approximation with fixed kernel functions; in their methods, the approximation and derivatives can be evaluated at the node only due to the fixity of the kernel function. Huerta et al. [17] developed a pseudo divergence-free field for the diffuse derivative technique in the framework of weak formulation for an incompressible fluid flow problem. Li and Liu [18,19] developed a hierarchical basis for the meshfree method, in which a reproducing kernel hierarchical partition of unity is proposed in association with a class of basic wavelet functions based on a solid mathematical foundation.

Discontinuous shape functions for the particle method were initially presented as the visibility criterion [20,21], which simply discards the nodes on the other side of a discontinuity line and cuts the support of the weight function. Then, EFGM approximation with a discontinuous derivative was developed for the material discontinuity problem based on wedge function enrichment [22]; the EFGM has also been developed for discontinuity problems [23–28]. Similarly, EFG approximation using the visibility criterion combined with Lagrange multipliers was applied to the material interface problem by Cordes and Moran [29]. Considering that these methods basically involve the weak formulation, compared to the strong formulation, they require a considerable amount of computation. The meshfree point collocation methods employing the full derivative technique, as in Luo and Häussler-Combe [30], may be slightly better than the weak formulation, but they are still computationally inefficient due to the burden from the higher order derivative computation [16,30]. However, this study utilizes the idea of the extrinsic interfacial formula proposed by Kim et al. [31]

for the interface condition on the basis of the strong formulation; the extrinsic interfacial formula enhances the advantages of fast derivative computations for singular functions near the interface. Note that extrinsic enrichment differs from intrinsic enrichment that extends the polynomial basis as in EFGM [21,22,32]. In fact, extrinsic enrichment is a superior means of the embedment of the discontinuity function into the approximation as compared to intrinsic enrichment.

In this study, we propose an extended particle derivative approximation (EPDA) schem to solve weak and strong discontinuity problems. This derivative approximation utilizes an extrinsic approach in the sense that the discontinuity strengths emerge from the shape function; these strengths, meanwhile, need to be found when solving the total system. Although the zeroth order shape function of the EPDA is analogous to that of conventional particle methods, the derivative approximation is wholly different. The EPDA is extrinsically equipped with discontinuous functions that are well designed to capture the characteristics of singular behaviors in PDE solutions; it offers not only excellent efficiency in derivative computations but also convenience in modeling interfacial geometry characteristics that provoke singularities. The singularities are naturally considered in the construction process of the derivative approximation. The EPDA is derived based on the Taylor polynomial expanded by the moving least squares method that effectively associates the derivative approximation with the nodal solution and discontinuity strengths based on the discretization using nodes and interfacial points. In addition, numerical implementation for constructing the discrete form of the EPDA will be explained in detail. In the second part of this study, the EPDA is implemented in the construction of difference equations for the given governing equations; the direct discretization yields a total system of equations for weak and strong discontinuity problems, such as heat transfer problems in composite materials and elasticity problems with material discontinuity.

2 Particle derivative approximation

In this section, we first derive the particle derivative approximation (PDA) scheme based on the Taylor series; the multi-index notation is used for convenience. Let $\mathbf{x} = (x_1, \ldots, x_n)$ be the n-tuple of real numbers, i.e. the n-dimensional real vector, and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ be the n-tuple of non-negative integers, i.e. the n-dimensional non-negative integer vector. The αth power of $\mathbf{x}$ is defined by

$$\mathbf{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n} \tag{1}$$

Then, the αth partial derivative operator is given by $D_{\mathbf{x}}^{\boldsymbol{\alpha}} := \partial_{x_1}^{\alpha_1} \cdots \partial_{x_n}^{\alpha_n}$. The αth derivative of a function $f(\mathbf{x})$ with respect to $\mathbf{x}$ is written as

$$D_{\mathbf{x}}^{\boldsymbol{\alpha}} f(\mathbf{x}) = \frac{\partial^{|\boldsymbol{\alpha}|} f(\mathbf{x})}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_n^{\alpha_n}} \tag{2}$$

where $|\boldsymbol{\alpha}|$ is the sum of all components of $\boldsymbol{\alpha}$; i.e. $|\boldsymbol{\alpha}| = \sum_{i=1}^{n} \alpha_i$. For a given differentiable function $u(\mathbf{x}) \in C^m(\bar{\Omega})$, the Taylor series is expressed as

$$u(\mathbf{x}) = \sum_{|\boldsymbol{\alpha}| \le m} \frac{(\mathbf{x}-\mathbf{y})^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!} D_{\mathbf{x}}^{\boldsymbol{\alpha}} u(\mathbf{y}) + H.O.T. \tag{3}$$

where the series is expanded about the local center $\mathbf{y}$. The mth order Taylor polynomial for approximating a continuous function is then written by neglecting the high order terms as follows

$$u^{cont}(\mathbf{x};\mathbf{y}) = \sum_{|\boldsymbol{\alpha}| \le m} \frac{(\mathbf{x}-\mathbf{y})^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!} D_{\mathbf{x}}^{\boldsymbol{\alpha}} u(\mathbf{y}) = \mathbf{p}_m^T(\mathbf{x};\mathbf{y})\,\mathbf{a}(\mathbf{y}) \tag{4}$$

where $\boldsymbol{\alpha}!$ is the factorial of the n-tuple $\boldsymbol{\alpha}$; i.e. $\boldsymbol{\alpha}! = \prod_{i=1}^{n} \alpha_i!$. As shown in Eq. (4), it is evident that the Taylor polynomial is decomposed into the polynomial vector and the derivative coefficient vector, and then the polynomial vector takes the form

$$\mathbf{p}_m^T(\mathbf{x};\mathbf{y}) = \left(\frac{(\mathbf{x}-\mathbf{y})^{\boldsymbol{\alpha}_1}}{\boldsymbol{\alpha}_1!}, \ldots, \frac{(\mathbf{x}-\mathbf{y})^{\boldsymbol{\alpha}_L}}{\boldsymbol{\alpha}_L!} \right) \tag{5}$$

where $\boldsymbol{\alpha}_L = (0, \ldots, m)$; m is placed at the $\boldsymbol{\alpha}_K$th slot in lexicographic order, and the length of the polynomial vector is $L = \frac{(n+m)!}{n!m!}$. Here, m denotes the order of the highest derivative as well as the order of consistency of the Taylor polynomial. From Eq. (4), the derivative coefficient vector is written as

$$\mathbf{a}(\mathbf{y}) = \begin{pmatrix} D_{\mathbf{x}}^{\boldsymbol{\alpha}_1} u(\mathbf{y}) \\ \vdots \\ D_{\mathbf{x}}^{\boldsymbol{\alpha}_L} u(\mathbf{y}) \end{pmatrix} \tag{6}$$

which includes all of the derivative approximations for $u(\mathbf{y})$ up to the $\boldsymbol{\alpha}_L$th order.

One of key ideas of the PDA is to determine $\mathbf{a}(\mathbf{y})$ by incorporating the moving least squares method and the nodal solution without involving the actual differentiation of the PDA. In order to find $\mathbf{a}(\mathbf{y})$, a discrete form of the weighted residual functional, which relates the Taylor polynomial to the nodal solution, is defined as

$$J = \sum_{I=1}^{N} w\left(\frac{\mathbf{x}_I - \mathbf{y}}{\rho_{\mathbf{y}}} \right) \left(\mathbf{p}_m^T(\mathbf{x}_I;\mathbf{y})\,\mathbf{a}(\mathbf{y}) - u_I \right)^2 \tag{7}$$

where u_I is the nodal solution and N is the number of nodes included in the support of the weight function w, of which the center is $\mathbf{y}$. The support size is determined by $\rho_{\mathbf{y}}$ which practically indicates the radius of the weight function. This was termed the dilation function in Kim and Kim [12]. Because the support size determines how many nodes are

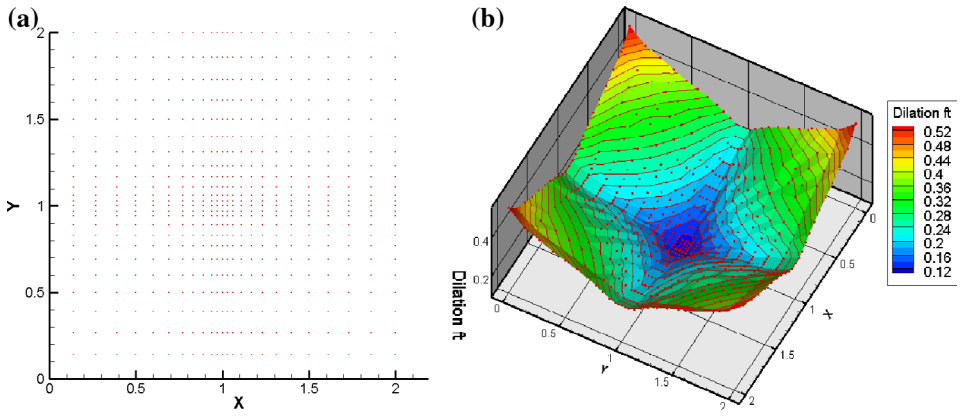

Fig. 1 An irregular node arrangement and the support size (or dilation function) **a** an irregular node distribution with a concentration around the center of the square domain, **b** surface plot of the support size (or dilation function) for the irregular node arrangement

included in the PDA, the number of nodes is designated to fall within a suitable range such that the resolution of the PDA is retained regardless of the position of the local center and node density; as a result, a variable dilation function is implemented in this study. An algorithm presented by Kim and Kim [33] is adopted here to determine the size. The upper and lower bounds of the number of included nodes are associated with the computation efficiency and the invertibility of the moment matrix, respectively. Figure 1a, b show an example of an irregular node distribution in which the nodes tend to be concentrated around the center of the square domain. From the plot of the support size (or dilation function) for this node arrangement as shown in Fig. 1b, it is clearly seen that the support size varies with respect to the position and node density; it becomes smallest at the center to reflect the high node density or to maintain the number of nodes included in the support, while it is largest at the corner of the square domain. On the other hand, conventional particle methods such as EFGM [9] or RKPM [10] determine the size using a constant radius probe regardless of the node density. This may badly affect the resolution of the approximation due to the irregular number of nodes participating in the approximation.

Note that $u_I = u(\mathbf{x}_I)$ in the context of this study; specifically, u_I denotes the nodal solution at $\mathbf{x}_I$. In conventional weak form based meshfree method such as EFGM [9], u_I does not match $u(\mathbf{x}_I)$ exactly; there exists a slight but non-negligible difference between u_I and $u(\mathbf{x}_I)$. The difference mostly comes from the constraint to enforce the essential boundary condition. Therefore, u_I is often called a nodal parameter instead of a nodal solution, and the approximation formula derived needs to be used to compute the nodal solution.

Another salient feature of the PDA scheme is that any function with a conical shape can be used for the weight function. It is worth noting that no differentiability for the weight function is required in the PDA formulation. As long as the function is non-negative and continuous, smoothness is not required. On the other hand, most particle methods demand

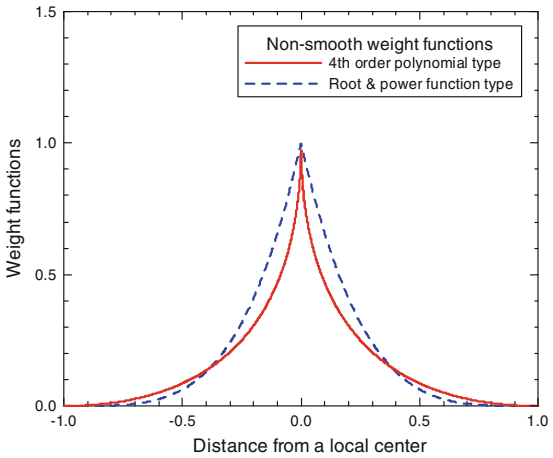

Fig. 2 Non-smooth weight functions

the differentiability of the weight function because the derivative of approximation includes the derivative of the weight function. As shown below, non-differentiable functions with a sharp peak and discontinuous derivatives are used in this study:

$$w\left(\frac{\mathbf{x}-\mathbf{y}}{\rho_{\mathbf{y}}}\right) = \left(1 - \left\|\frac{\mathbf{x}-\mathbf{y}}{\rho_{\mathbf{y}}}\right\|\right)^4 \tag{8}$$

$$w\left(\frac{\mathbf{x}-\mathbf{y}}{\rho_{\mathbf{y}}}\right) = \left(1 - \left\|\frac{\mathbf{x}-\mathbf{y}}{\rho_{\mathbf{y}}}\right\|^{\frac{1}{2}}\right)^2 \tag{9}$$

In fact, non-smooth weight functions were used previously by Kim and Kim [12] and by Lee and Yoon [13] in the framework of the meshfree point collocation method. Figure 2 illustrates an example for non-smooth weight functions. Although smooth weight functions have often been used in the weakly formulated particle method, it was recognized that non-smooth functions are preferable to smooth ones in the strongly formulated particle method through numerical experiments; therefore, the use of smooth weight functions is very feasible in this study.

As in the conventional moving least squares method, the weighted residual functional J of Eq. (7) is minimized by a stationary condition, i.e. $\frac{\partial J}{\partial \mathbf{a}} = 0$, yielding the following normal equation:

$$\mathbf{M}(\mathbf{y})\,\mathbf{a}(\mathbf{y}) = \mathbf{B}(\mathbf{y}) \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix} \tag{10}$$

In the above equation, the moment matrix and the B matrix, respectively, are written as

$$\mathbf{M}(\mathbf{y}) = \sum_{I=1}^{N} \left(\mathbf{p}_m(\mathbf{x}_I;\mathbf{y}) w\left(\frac{\mathbf{x}_I - \mathbf{y}}{\rho_\mathbf{y}}\right) \mathbf{p}_m^T(\mathbf{x}_I;\mathbf{y}) \right) \tag{11}$$

$$\mathbf{B}(\mathbf{y}) = \left(w\left(\frac{\mathbf{x}_1 - \mathbf{y}}{\rho_\mathbf{y}}\right) \mathbf{p}_m(\mathbf{x}_1;\mathbf{y}), \ldots, w \times \left(\frac{\mathbf{x}_N - \mathbf{y}}{\rho_\mathbf{y}}\right) \mathbf{p}_m(\mathbf{x}_N;\mathbf{y}) \right) \tag{12}$$

where the dimensions of $\mathbf{M}(\mathbf{y})$ and $\mathbf{B}(\mathbf{y})$ are correspondingly $L\left(=\frac{(n+m)!}{n!m!}\right) \times L$ and $N \times L$. From Eq. (10), the coefficient vector is obtained by

$$\mathbf{a}(\mathbf{y}) = \mathbf{M}^{-1}(\mathbf{y})\,\mathbf{B}(\mathbf{y}) \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix} \tag{13}$$

and substituting $\mathbf{x}$ for $\mathbf{y}$ yields the final form of the PDA as follows

$$\begin{pmatrix} D_\mathbf{x}^{\alpha_1} u(\mathbf{x}) \\ \vdots \\ D_\mathbf{x}^{\alpha_L} u(\mathbf{x}) \end{pmatrix} = \mathbf{M}^{-1}(\mathbf{x})\,\mathbf{B}(\mathbf{x}) \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix} \tag{14}$$

Here, it is important to note that the PDA includes all of the derivative approximations up to the order of consistency. The substitution of $\mathbf{x}$ for $\mathbf{y}$ implies that the PDA is constructed on the local center such that the coefficients of the Taylor polynomial best fit the derivatives of the original function. The PDA is always used at the position where it is constructed regardless of whether the position is a node or not; non-nodal point is used for the setup of the interfacial equation, while node is used for the discretization of other governing equations. To secure the optimal level of accuracy, it is not used at any position other than where it is derived.

Considering the meshfree point collocation method by Lee and Yoon [13], the derivative computation by the PDA can be viewed equivalently. That is to say, the derivative of the Taylor polynomial can be defined by differentiation with respect to $\mathbf{x}$ and substitution of $\mathbf{x} = \mathbf{y}$ as follows

$$D_\mathbf{x}^\alpha u(\mathbf{y}) \approx \left(D_\mathbf{x}^\alpha u^{cont}(\mathbf{x};\mathbf{y}) \right)\Big|_{\mathbf{x}=\mathbf{y}} \tag{15}$$

where the differentiation precedes the substitution, as in the derivation process of the Taylor series. The derivative coefficient $D_\mathbf{x}^\alpha u(\mathbf{y})$ serves as a derivative approximation of $u(\mathbf{x})$ at $\mathbf{y}$.

For convenience, the PDA of Eq. (14) can be transformed into another form involving nodal shape functions as follows

$$\begin{pmatrix} D_\mathbf{x}^{\alpha_1} u(\mathbf{x}) \\ \vdots \\ D_\mathbf{x}^{\alpha_L} u(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \Phi_1^{[\alpha_1]}(\mathbf{x}), \ldots, \Phi_N^{[\alpha_1]}(\mathbf{x}) \\ \vdots \quad \ddots \quad \vdots \\ \Phi_1^{[\alpha_L]}(\mathbf{x}), \ldots, \Phi_N^{[\alpha_L]}(\mathbf{x}) \end{pmatrix} \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix} \tag{16}$$

where $\Phi_I^{[\alpha_K]}(\mathbf{x})$ denotes the $\boldsymbol{\alpha}_K$th derivative of the nodal shape function for node I; this is referred to as the generalized regular shape function. The nodal shape function is expressed as

$$\Phi_I^{[\alpha]}(\mathbf{x}) = \boldsymbol{\alpha}! \mathbf{e}_\alpha^T \mathbf{M}^{-1}(\mathbf{x})\,\mathbf{p}_m(\mathbf{x}_I;\mathbf{x})\, w\left(\frac{\mathbf{x}_I - \mathbf{x}}{\rho_\mathbf{x}}\right) \tag{17}$$

where $\mathbf{e}_\alpha^T = (0, \ldots, 1, \ldots 0)$, in which 1 is placed at the $\boldsymbol{\alpha}$th slot of $\mathbf{e}_\alpha^T$ in lexicographic order. Then, a scalar product of the row vector of a generalized regular shape function matrix and the nodal solution vector generates the following expression:

$$D_\mathbf{x}^\alpha u(\mathbf{x}) = \sum_{I=1}^{N} \Phi_I^{[\alpha]}(\mathbf{x})\, u_I \tag{18}$$

Note that $\Phi_I^{[\alpha]}(\mathbf{x})$ does not directly indicate $D_\mathbf{x}^\alpha \Phi_I^{[\mathbf{0}]}(\mathbf{x})$. However, it is a very good approximation of $D_\mathbf{x}^\alpha \Phi_I^{[\mathbf{0}]}(\mathbf{x})$ in the sense of satisfying the consistency condition (or reproducing property) of the Taylor polynomial. In other words, $D_\mathbf{x}^{\alpha_1} u(\mathbf{x}), \ldots, D_\mathbf{x}^{\alpha_L} u(\mathbf{x})$ can reproduce any function made via the combination of a polynomial basis up to the order of consistency. It should be stressed that the construction of the PDA requires no differentiation of the shape function; instead, some matrix operations involving moment matrix inversion are necessary. As a result, the computation speed for the derivative calculation is dramatically accelerated. The concept of the PDA shares a common strategy with the diffuse derivative of DEM [6]. It can also be compared with the diffuse derivative of the meshfree approximation by Lee and Yoon [13], which involves the moving process that takes advantage of the fast diffuse derivative calculation.

In terms of interpolation, one may argue that the PDA is not an interpolant because it does not interpolate a solution between nodes. Although the PDA is capable of generating the function value at an arbitrary position based on known nodal values, it does not use a predefined interpolant such as a finite element (FE) shape function. Wherever interpolation is necessary, the PDA should be constructed at the position of interest. However, there may be no objection to the statement that the PDA is a derivative approximation, as it can generate a derivative function everywhere in a numerical model.

The key feature of the PDA is that the Taylor series is approximated by the moving least squares method based on a

complete node-wise scheme. The PDA relates the nodal solution u_I values to the derivatives of the solution via the generalized regular shape function. In most conventional meshfree methods such as EFGM, the derivative approximation is obtained from a mathematically exact differentiation of the approximation. This usually provokes the cumbersome differentiation of the inverse of the moment matrix as well as the weight function. Furthermore, the high order derivative computation for the strong formulation may considerably deteriorate the computational efficiency. However, the PDA successfully retains its level of efficiency regardless of the derivative order and maintains its accuracy up to the consistency order; as the derivative order increases, the accuracy naturally decreases according to the consistency order of the Taylor polynomial, but the computational effort does not increase at all because all derivative approximations from zeroth to the highest order are obtained at the same time.

3 Extended particle derivative approximation for discontinuity modeling

3.1 Finding the projection point and normal vector

In this paper, we consider the PDE solution has weak and strong discontinuities along the interface; not only the solution itself but also the tangential and normal derivatives of the solution have jumps. Therefore, it is crucial to develop well-designed jump functions which are able to capture the singular behavior. Furthermore, these jump functions should be suitable for the framework of strong form particle methods using complete node-wise computation.

Let us consider a projection map to Γ:

$$\mathbf{y}_\Gamma = proj_\Gamma \mathbf{y} \in \Gamma, \quad \mathbf{y} \in \Omega^S \backslash \Gamma \tag{19}$$

where Ω^S denotes a singular domain that requires discontinuity modeling. The above maps $\mathbf{y}$ to the closest point $\mathbf{y}_\Gamma$ on the interface; i.e. $\mathbf{y}_\Gamma := \arg\left(\inf_{\mathbf{x}\in\Gamma} \|\mathbf{x}-\mathbf{y}\|\right)$. The normal vector map is then defined as

$$\mathbf{n}_\Gamma(\mathbf{y}) = \frac{\mathbf{y}-\mathbf{y}_\Gamma}{\|\mathbf{y}-\mathbf{y}_\Gamma\|}, \quad \mathbf{y} \in \Omega^S \backslash \Gamma \tag{20}$$

and it yields a unit normal vector to Γ for $\mathbf{y}$. In addition, a unit tangential vector is defined by rotating the unit normal vector 90° counterclockwise, and the following relation holds

$$\mathbf{t}_\Gamma(\mathbf{y}) \cdot \mathbf{n}_\Gamma(\mathbf{y}) = 0 \tag{21}$$

where $\mathbf{t}_\Gamma(\mathbf{y})$ is a unit tangential vector. At this point, the tangent hyperplane function is defined as shown below; for details refer to Kim et al. [26]

$$H_y(\mathbf{x}) = \mathbf{n}_\Gamma(\mathbf{y}) \cdot (\mathbf{x}-\mathbf{y}_\Gamma) \tag{22}$$

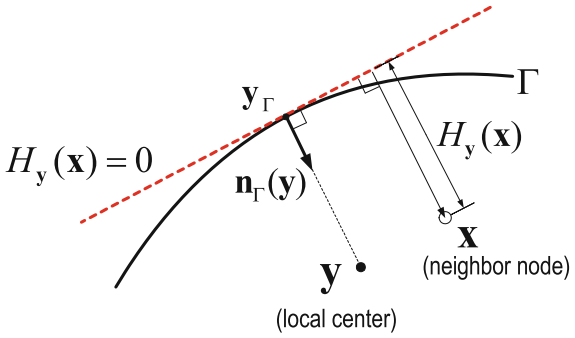

Fig. 3 Concept of the tangent hyperplane function

The function is readily determined using $\mathbf{y}_\Gamma$ and $\mathbf{n}_\Gamma(\mathbf{y})$. Defining the tangent hyperplane function is based on the projection point of the field point $\mathbf{y}$ and the normal vector $\mathbf{n}_\Gamma(\mathbf{y})$. The definitions of the projection point and normal vector of this study are very natural from a mathematical viewpoint; accordingly, they can easily be applied to a non-smooth interface [32]. Once the tangent hyperplane function is defined for the local center $\mathbf{y}$, it is only a function of $\mathbf{x}$, not $\mathbf{y}$. Figure 3 illustrates the concept of the tangent hyperplane function. The tangent hyperplane function can have a sign of plus or minus; in fact, it denotes the distance from $\mathbf{x}$ to the tangential line ($H_y(\mathbf{x}) = 0$) passing through $\mathbf{y}_\Gamma$. It will replace the true interface during the construction of the derivative approximation because troublesome high order differentiation as regards jump functions is effectively exempted due to its use. Given that all of the nodes in the support (or the domain of influence) use the same normal vector, considerable computational effort can be saved.

In many numerical methods, the interface is modeled by a series of line segments, but this type of interface modeling complicates the definition of the normal vector at times. For example, defining a normal vector is highly problematic at the junction of two segments or at the end point of an interface. However, the definition of a normal vector used in this study can be successfully applied in a consistent manner in both cases. As a result, discontinuous functions can be seamlessly constructed along the interface, and the sharpness of the approximation is effectively preserved because the normal vector can be naturally determined with no awkwardness at any position.

The numerical representation of the interface is fixed to a mesh or grid structure in many numerical schemes. This fixity can cause cumbersome numerical difficulties in developing new numerical schemes, especially when simulating free or moving boundary problems involving consecutive geometrical changes. However, the use of a tangent hyperplane function can effectively ease the difficulty while preserving the advantages of the particle method coming from the node-wise computation. On the other hand, when the FEM is used

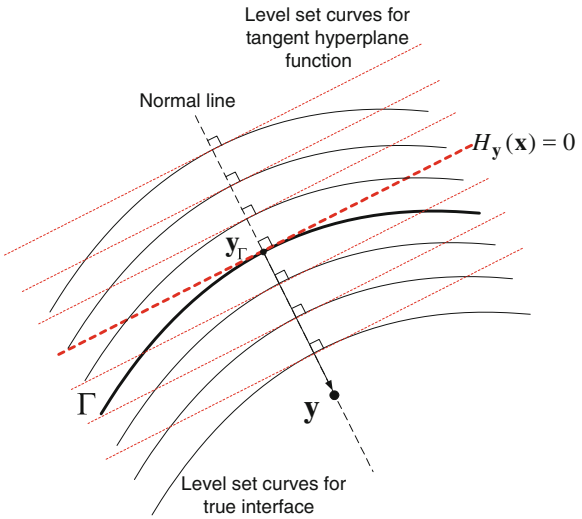

Fig. 4 Two types of level set curves for the interface modeling

with level set curves, the normal vector should be defined at the projection point for every Gauss quadrature point in the enriched elements [34,35]; inevitably, this becomes very troublesome and time consuming. To clarify the difference between an approach based on the FEM and that of this study, two types of level set curves are drawn for the true interface shown in Fig. 4. The straight lines parallel to the tangential hyperplane function indicate the level set curves for the derivative approximation of this study. It should be noted that the gradients of the two types of level set curves exactly match along the normal line passing through the projection point. Therefore, the full derivative and diffuse derivative are equivalent along the normal line. This feature is very important because all computations regarding the interfacial singularity occur along this normal line in the proposed formulation.

3.2 Step function, wedge function, and scissors function for discontinuity modeling

The solution and its derivatives for PDEs with an interfacial singularity exhibit discontinuities across the interface. Here, three types of jump functions are presented for interfacial discontinuity modeling. First, for jump modeling in the solution field, a step function is introduced, as shown below.

$$b_s(\mathbf{x};\mathbf{y}) = sign\left(\mathbf{n}_\Gamma(\mathbf{y})\cdot(\mathbf{x}-\mathbf{y}_\Gamma)\right) = sign\left(H_y(\mathbf{x})\right) \quad (23)$$

Figure 5a presents a schematic drawing of the step function. When constructing the EPDA, the tangent hyperplane function serves as an interface for all nodes included in the domain of influence. Once $\mathbf{y}_\Gamma$ and $\mathbf{n}_\Gamma(\mathbf{y})$ are fixed, $b_s(\mathbf{x};\mathbf{y})$ no longer depends on $\mathbf{y}$ but becomes a function of $\mathbf{x}$ instead. Interestingly, differentiation of $\mathbf{n}_\Gamma(\mathbf{y})$ or $\mathbf{y}_\Gamma$ is not necessary during the construction of the EPDA because this differentiation always occurs with respect to $\mathbf{x}$. Moreover, the step function is a constant function. Therefore, the complicated derivative calculations can be effectively avoided; in fact, they are all zero.

Secondly, for normal derivative jump modeling, a wedge function is employed as given below.

$$b_n(\mathbf{x};\mathbf{y}) = \left\| H_y(\mathbf{x}) \right\| \quad (24)$$

Figure 5b shows a plot of the wedge function. Note that the wedge function can be obtained by integrating the step function, while the step function is given by taking the sign value of the tangent hyperplane function. Actually, $b_n(\mathbf{x};\mathbf{y})$ is the distance function, indicating the minimum distance from $\mathbf{x}$ to the tangent hyperplane. Since $\mathbf{y}_\Gamma$ and $\mathbf{n}_\Gamma(\mathbf{y})$ are not functions of $\mathbf{x}$ in the construction stage, the derivative calculation of the wedge function is also very simple not associated with the differentiation of $\mathbf{y}_\Gamma$ and $\mathbf{n}_\Gamma(\mathbf{y})$.

Thirdly, the design of a scissors function demands a tactical approach. The scissors function is devised to capture the tangential derivative jump in the solution, as follows:

$$b_t(\mathbf{x};\mathbf{y}) = sign\left(H_y(\mathbf{x})\right)\left(\mathbf{t}(\mathbf{y})\cdot(\mathbf{x}-\mathbf{y}_\Gamma)\right) \quad (25)$$

Here, the unit tangential vector has the property of $\mathbf{n}_\Gamma(\mathbf{y})\cdot\mathbf{t}_\Gamma(\mathbf{y}) = 0$. An illustration of the scissors function is presented in Fig. 5c, showing that the scissors function exhibits a jump across the interface. Although the normal derivative of the scissors function is zero, the function itself and its tangential derivative feature a jump across the tangent hyperplane. This type of combination of step, wedge, and scissors functions with the PDA can approximate various types of weak and strong discontinuities; this enriched means of derivative approximation is termed the EPDA.

3.3 Derivative calculation of the step function, wedge function, and scissors function

Derivative computations of the jump functions are essential when deriving the EPDA. Although differentiation is exempted in the PDA based on the fact that the coefficients of the Taylor polynomial substitute for the derivative functions of the original function, differentiation for the EPDA is nonetheless required due to the existence of the jump functions. Let us first consider the derivative of the step function. The step function has constant values. Thus, all derivatives higher than the first-order are zero reading:

$$D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_s(\mathbf{x};\mathbf{y}) = 0, \quad \mathbf{y}\in\Omega^S\backslash\Gamma, \quad |\boldsymbol{\alpha}|\geq 1 \quad (26)$$

The derivative computation of the wedge function is straightforward. Since $b_n(\mathbf{x};\mathbf{y})$ is a linear distance function, its first-order derivatives are constant. For $|\boldsymbol{\alpha}| = 1$, the first-order derivatives are readily given by

$$D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_n(\mathbf{x};\mathbf{y}) = sign\left(H_y(\mathbf{x})\right)\left(\mathbf{n}_\Gamma(\mathbf{y})\right)^{\boldsymbol{\alpha}}, \ \mathbf{x}\in\Omega^S\backslash\Gamma, |\boldsymbol{\alpha}| = 1 \quad (27)$$

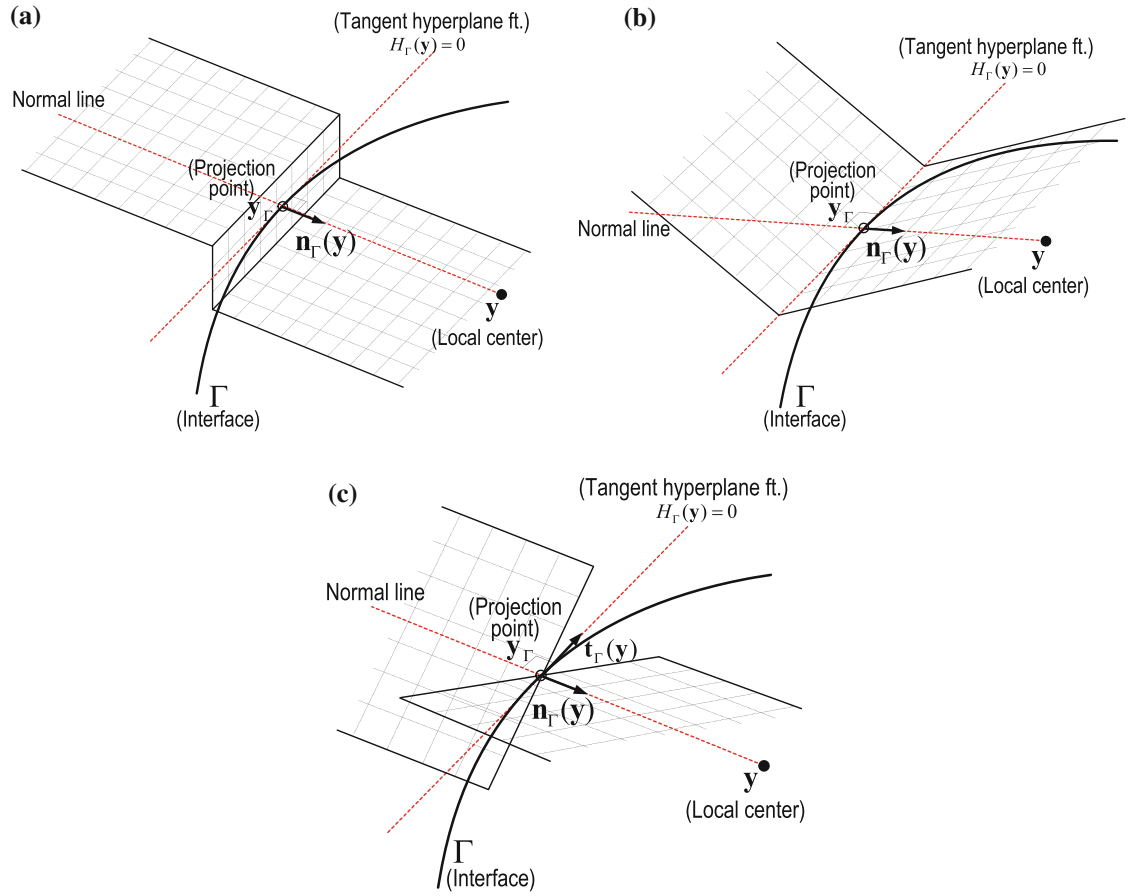

Fig. 5 Schematic drawings of jump functions: a normal line and a tangent hyperplane function are depicted with unit normal and tangent vectors, **a** step function, **b** wedge function, and **c** scissors function

where $sign\left(H_y(\mathbf{x})\right)$ is $+1$ or -1. As a result, the components of the unit normal vector corresponding to the derivative directions appear in the first derivative. For instance, $(\mathbf{n}_\Gamma(\mathbf{y}))^{(1,0)} = n_1$ and $(\mathbf{n}_\Gamma(\mathbf{y}))^{(0,1)} = n_2$ from the definition of the multi-index notation. Also, computing the derivative of jump functions does not involve cumbersome problems, as the $\mathbf{y}$'s of $b_s(\mathbf{x};\mathbf{y})$, $b_n(\mathbf{x};\mathbf{y})$, and $b_t(\mathbf{x};\mathbf{y})$ are immediately replaced by $\mathbf{x}$ as soon as the differentiation is completed; recall the process presented in Eq. (14).

The first-order derivative of the scissors function is similar to that of the wedge function except that the derivative of the tangential direction is non-zero rather than the normal direction. Thus, the first-order derivative ($|\boldsymbol{\alpha}| = 1$) is written as follows

$$D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_t(\mathbf{x};\mathbf{y}) = sign\left(H_y(\mathbf{x})\right)(\mathbf{t}(\mathbf{y}))^{\boldsymbol{\alpha}}, \quad \mathbf{x}\in\Omega^S\backslash\Gamma, \quad |\boldsymbol{\alpha}|=1 \tag{28}$$

Here, note that derivatives higher than the first order are all zero because the jump functions such as step, wedge, and scissors functions are basically constant or linear functions. The second-order derivatives of the discontinuous function appearing during the discretization of the PDEs naturally disappear, and the strong formulation becomes easy to handle. Furthermore, the first-order derivatives of the step and scissors functions are singular on Γ but non-singular elsewhere; in the proposed formulation, no differentiation is required on Γ. Accordingly, no singularity or computational awkwardness arises in the numerical schemes under construction.

As illustrated in Fig. 5, the following relationships can be deduced for the jump functions:

$$D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_{s,n}(\mathbf{x},\mathbf{y}) = D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_{s,n}(\mathbf{y}+\tau\mathbf{t}_\Gamma(\mathbf{y}),\mathbf{y}) = 0, \quad |\boldsymbol{\alpha}| = 1 \tag{29}$$

$$D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_{s,t}(\mathbf{x},\mathbf{y}) = D^{\boldsymbol{\alpha}}_{\mathbf{x}} b_{s,t}(\mathbf{y}+\tau\mathbf{n}_\Gamma(\mathbf{y}),\mathbf{y}) = 0, \quad |\boldsymbol{\alpha}| = 1 \tag{30}$$

where $b_{s,n}(\mathbf{y}+\tau\mathbf{t}_\Gamma(\mathbf{y}),\mathbf{y})$ are functions defined along the lines of latitude perpendicular to the normal line passing through the projection point, while $b_{s,t}(\mathbf{y}+\tau\mathbf{n}_\Gamma(\mathbf{y}),\mathbf{y})$ are functions defined along the lines of longitude parallel to the normal line. Recall that the tangential hyperplane and level set curve are equivalent along the normal line such that no discrepancy exists in the derivative calculation. The actual

derivative computation of the EPDA mostly occurs on the projection point. Therefore, the derivative approximation in this study is very efficient and accurate in this scenario.

Particle methods would be expected not to employ an element or grid structure for any purpose, such as the integration of the weak form. Moreover, they are expected to be free from the limitation of the topological modeling of the interface. To realize this, the EPDA is designed, which contains the geometric information of the interface. The geometric information is immersed in the EPDA during the process of finding the projection point. Hence, the derivative approximation, which is easy to construct, free from a mesh or grid, and relevant to the interfacial singularity, is effectively developed.

3.4 Construction of the extended particle derivative approximation

This section presents how to construct the EPDA for the discretization of a PDE with discontinuous coefficients or a singular source term across an interface. Due to the discontinuity or singularity, the solution and/or its derivatives may yield discontinuities across the interface. The key idea here is to incorporate the jump conditions given a priori in the derivative approximation construction.

Consider a body Ω consisting of two different material coefficients, i.e. an inclusion and a matrix (as in Fig. 6). A composite material containing an inclusion with a different material coefficient can be such a case. The boundary between the two materials forms the interface Γ. As shown in Fig. 6, the singular domain Ω^S is defined as

$$\Omega^S = \{\mathbf{x} \in \Omega | dist(\mathbf{x}, \Gamma) < \rho_{\mathbf{x}}\} \tag{31}$$

where $dist\ (\mathbf{x}, \Gamma)$ denotes the distance from $\mathbf{x}$ to Γ and $\Omega = \overline{\Omega} \backslash \partial\Omega$. Recall that the dilation function $\rho_{\mathbf{x}}$ is the radius of the domain of influence. The dilation function affects the partition of the body for a numerical simulation. When the domain of influence of $\mathbf{x}$ touches the interface, $\mathbf{x}$ is considered

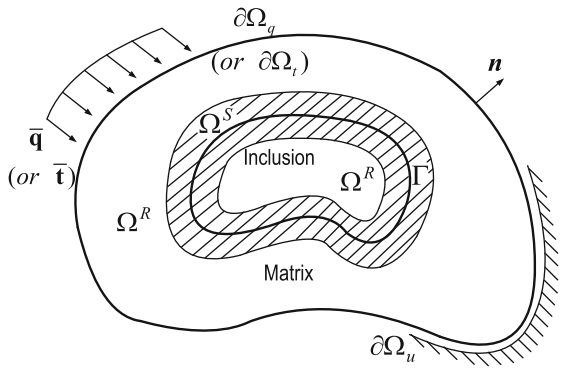

Fig. 6 A body with an interface, consisting of the singular domain Ω^S and the regular domain Ω^R bounded by $\partial\Omega\ (= \partial\Omega_u \cup \partial\Omega_t)$

to belong to the singular domain. On the other hand, the regular domain is denoted as

$$\Omega^R = \Omega \backslash \Omega^S \tag{32}$$

yielding $\Omega^R \cup \Omega^S = \Omega$ and $\Omega^R \cap \Omega^S = 0$.

In a regular domain involving no discontinuities or singularities, $u^{cont}(\mathbf{x}; \mathbf{y})$ is readily adopted as a regular approximation function as follows

$$u^R(\mathbf{x}; \mathbf{y}) = u^{cont}(\mathbf{x}; \mathbf{y}), \quad \mathbf{y} \in \Omega^R \tag{33}$$

The relevant derivative approximation for the process described above, as discussed in the previous section, will be necessary here as well to yield the difference equations. On the other hand, in the singular domain, a singular approximation is defined by combining the continuous and discontinuous approximation functions as follows

$$u^S(\mathbf{x}; \mathbf{y}) = u^{cont}(\mathbf{x}; \mathbf{y}) + u^{disc}(\mathbf{x}; \mathbf{y}), \quad \mathbf{y} \in \Omega^S \backslash \Gamma \tag{34}$$

where the discontinuous part approximates the discontinuous portion in the solution or its derivatives, while the continuous part accounts for the smooth part of the solution. The term 'regular' implies that the singular term may be effectively eliminated from the given singular function.

In this study, three types of discontinuities need to be dealt with: the solution jump, the normal derivative jump, and the tangential derivative jump. Therefore, all jump functions developed are necessary. The discontinuous part is made by a linear combination of these three jump functions as following

$$u^{disc}(\mathbf{x}; \mathbf{y}) = \beta_s b_s(\mathbf{x}; \mathbf{y}) + \beta_n b_n(\mathbf{x}; \mathbf{y}) + \beta_t b_t(\mathbf{x}; \mathbf{y}) \tag{35}$$

where β_s, β_n, and β_t represent the discontinuity strength or jump magnitude corresponding to each jump function. The discontinuity strengths are closely related to the interface conditions of the governing equations. They can be manifested by taking the directional jump of the singular approximation function. First, the jumps of the singular approximation and its derivatives across the interface is expressed as

$$\left[u^S(\mathbf{x}; \mathbf{y})\right]_\Gamma = \beta_s \left[b_s(\mathbf{x}; \mathbf{y})\right]_\Gamma = 2\beta_s \tag{36}$$

where $[\ ]_\Gamma$ denotes the directional difference on both sides of Γ; i.e. $[f]_\Gamma = f^+ - f^-$, in which the superscript $+$ or $-$ denotes the limit value toward the interface. Note that the continuous part disappears because it has no jump. Also, the wedge function and the scissors function have no jump across the interface (more precisely, along the normal line passing through the projection point). As shown in Fig. 5c, the scissors function is absolutely continuous along the normal line. Next, taking the normal derivative jump of the singular approximation yields

$$\left[\frac{\partial u^S(\mathbf{x}; \mathbf{y})}{\partial \mathbf{n}_\Gamma(\mathbf{y})}\right]_\Gamma = \beta_n \left[\frac{\partial b_n(\mathbf{x}; \mathbf{y})}{\partial \mathbf{n}_\Gamma(\mathbf{y})}\right]_\Gamma = 2\beta_n \tag{37}$$

Similarly, the tangential derivative jump can be obtained by

$$\left[\frac{\partial u^S(\mathbf{x};\mathbf{y})}{\partial \mathbf{t}_\Gamma(\mathbf{y})}\right]_\Gamma = \beta_t \left[\frac{\partial b_t(\mathbf{x};\mathbf{y})}{\partial \mathbf{t}_\Gamma(\mathbf{y})}\right]_\Gamma = 2\beta_t \tag{38}$$

The discontinuous part can then be rewritten using Eqs. (36)–(38) as

$$\begin{aligned} u^{disc}(\mathbf{x};\mathbf{y}) = &\frac{1}{2}\left[u^s(\mathbf{x};\mathbf{y})\right]_\Gamma b_s(\mathbf{x};\mathbf{y}) \\ &+\frac{1}{2}\left[\frac{\partial u^s(\mathbf{x};\mathbf{y})}{\partial \mathbf{n}_\Gamma(\mathbf{y})}\right]_\Gamma b_n(\mathbf{x};\mathbf{y}) \\ &+\frac{1}{2}\left[\frac{\partial u^s(\mathbf{x};\mathbf{y})}{\partial \mathbf{t}_\Gamma(\mathbf{y})}\right]_\Gamma b_t(\mathbf{x};\mathbf{y}) \end{aligned} \tag{39}$$

where $\left[u^s(\mathbf{x};\mathbf{y})\right]_\Gamma$, $\left[\frac{\partial u^s(\mathbf{x};\mathbf{y})}{\partial \mathbf{n}_\Gamma(\mathbf{y})}\right]_\Gamma$, and $\left[\frac{\partial u^s(\mathbf{x};\mathbf{y})}{\partial \mathbf{t}_\Gamma(\mathbf{y})}\right]_\Gamma$ need to be determined from the governing equations. In most cases, the jump magnitudes are explicitly given from the interface conditions of the governing equations. If not, they can be readily obtained from the relevant interface condition. Here, it is important to note that unlike the normal derivative jump, the tangential derivative jump can be expressed in terms of the solution jump as shown below

$$\left[\frac{\partial u}{\partial \mathbf{t}}\right]_\Gamma = \frac{\partial [u]_\Gamma}{\partial \mathbf{t}} \tag{40}$$

Thus, Eq. (39) can be rewritten as

$$\begin{aligned} u^{disc}(\mathbf{x};\mathbf{y}) = &\frac{1}{2}[u]_{\mathbf{y}_\Gamma} b_s(\mathbf{x};\mathbf{y}) + \frac{1}{2}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{y}_\Gamma} \\ &\times b_n(\mathbf{x};\mathbf{y}) + \frac{1}{2}\frac{\partial [u]_{\mathbf{y}_\Gamma}}{\partial \mathbf{t}} b_t(\mathbf{x};\mathbf{y}) \end{aligned} \tag{41}$$

where $[u]_{\mathbf{y}_\Gamma}$, $\frac{1}{2}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{y}_\Gamma}$, and $\frac{1}{2}\frac{\partial [u]_{\mathbf{y}_\Gamma}}{\partial \mathbf{t}}$ replace the jumps of the singular approximation, which are given directly by the governing equations or which should be determined by numerically solving the PDE. The derivatives of the discontinuous part are expressed as

$$\begin{aligned} \begin{pmatrix} D_{\mathbf{x}}^{\alpha_1} u^{disc}(\mathbf{x};\mathbf{y}) \\ \vdots \\ D_{\mathbf{x}}^{\alpha_L} u^{disc}(\mathbf{x};\mathbf{y}) \end{pmatrix} = &\frac{1}{2}[u]_{\mathbf{y}_\Gamma} \begin{pmatrix} D_{\mathbf{x}}^{\alpha_1} b_s(\mathbf{x};\mathbf{y}) \\ \vdots \\ D_{\mathbf{x}}^{\alpha_L} b_s(\mathbf{x};\mathbf{y}) \end{pmatrix} \\ &+\frac{1}{2}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{y}_\Gamma} \begin{pmatrix} D_{\mathbf{x}}^{\alpha_1} b_n(\mathbf{x};\mathbf{y}) \\ \vdots \\ D_{\mathbf{x}}^{\alpha_L} b_n(\mathbf{x};\mathbf{y}) \end{pmatrix} \\ &+\frac{1}{2}\frac{\partial [u]_{\mathbf{y}_\Gamma}}{\partial \mathbf{t}} \begin{pmatrix} D_{\mathbf{x}}^{\alpha_1} b_t(\mathbf{x};\mathbf{y}) \\ \vdots \\ D_{\mathbf{x}}^{\alpha_L} b_t(\mathbf{x};\mathbf{y}) \end{pmatrix} \end{aligned} \tag{42}$$

where $D_{\mathbf{x}}^{\alpha_K} b_s(\mathbf{x};\mathbf{x})$, $D_{\mathbf{x}}^{\alpha_K} b_n(\mathbf{x};\mathbf{x})$, and $D_{\mathbf{x}}^{\alpha_K} b_t(\mathbf{x};\mathbf{x})$ can easily be calculated because $b_s(\mathbf{x};\mathbf{x})$, $b_n(\mathbf{x};\mathbf{x})$, and $b_t(\mathbf{x};\mathbf{x})$ were constructed before this step.

The singular approximation is rewritten as

$$\begin{aligned} u^S(\mathbf{x};\mathbf{y}) = &\mathbf{p}_m^T(\mathbf{x};\mathbf{y})\,\mathbf{a}_S(\mathbf{y}) + \frac{1}{2}[u]_{\mathbf{y}_\Gamma} b_s(\mathbf{x};\mathbf{y}) \\ &+\frac{1}{2}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{y}_\Gamma} b_n(\mathbf{x};\mathbf{y}) + \frac{1}{2}\frac{\partial [u]_{\mathbf{y}_\Gamma}}{\partial \mathbf{t}} b_t(\mathbf{x};\mathbf{y}) \end{aligned} \tag{43}$$

where $\mathbf{a}_S(\mathbf{y})$ is the only unknown factor if the jump magnitudes are provided or assumed to be determined. As a result, the continuous part is approximated by the Taylor polynomial up to the order of consistency, and the singular part is modeled by a linear combination of the jump functions. Technically, the accuracy of the EPDA relies on the resolution of the jump extraction. The resolution of extraction is also dependent of the regularity of the jump function. Although the jump functions in this paper take the simplest form for computational efficiency, if necessary, other jump function forms with elevated regularity can readily be implemented.

At this point, the weighted residual functional for the singular approximation can be set as

$$J = \sum_{I=1}^{N} w\left(\frac{\mathbf{x}_I - \mathbf{y}}{\rho_{\mathbf{y}}}\right)\left(\mathbf{p}_m^T(\mathbf{x}_I;\mathbf{y})\,\mathbf{a}_S(\mathbf{y}) - u_I^R\right)^2 \tag{44}$$

where the Taylor polynomial serves to approximate the nonsingular part of the solution. The regular part of the nodal solution is expressed as

$$\begin{aligned} u_I^R = u_I &- \frac{1}{2}[u]_{\mathbf{y}_\Gamma} b_s(\mathbf{x}_I;\mathbf{y}) - \frac{1}{2}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{y}_\Gamma} b_n(\mathbf{x}_I;\mathbf{y}) \\ &-\frac{1}{2}\frac{\partial [u]_{\mathbf{y}_\Gamma}}{\partial \mathbf{t}} b_t(\mathbf{x}_I;\mathbf{y}) \end{aligned} \tag{45}$$

where u_I is the nodal solution for the original PDE and $b_s(\mathbf{x}_I;\mathbf{y})$, $b_n(\mathbf{x}_I;\mathbf{y})$, and $b_t(\mathbf{x}_I;\mathbf{y})$ are the nodal values of the jump functions. The resolution of the extraction of the jumps at each node level governs the resolution of u_I^R. Note that the moving least squares method successfully associates the nodal solution of the PDE with the derivative coefficients of the Taylor polynomial.

The procedure for determining $\mathbf{a}_S(\mathbf{y})$ is identical to that of the PDA except that the u_I^R values replace the u_I values. Employing the stationary condition of the weighted residual functional yields

$$\mathbf{a}_S(\mathbf{y}) = \mathbf{M}^{-1}(\mathbf{y})\,\mathbf{B}(\mathbf{y}) \cdot \begin{pmatrix} u_1^R \\ \vdots \\ u_N^R \end{pmatrix} \tag{46}$$

where the constitutions of $\mathbf{M}(\mathbf{y})$ and $\mathbf{B}(\mathbf{y})$are identical to the case of the PDA. Then, substituting $\mathbf{x}$ for $\mathbf{y}$ in Eq. (46) generates the following derivative approximation:

$$\mathbf{a}_S(\mathbf{x}) = \begin{pmatrix} D_{\mathbf{x}}^{\boldsymbol{\alpha}_1} u^{cont}(\mathbf{x};\mathbf{x}) \\ \vdots \\ D_{\mathbf{x}}^{\boldsymbol{\alpha}_L} u^{cont}(\mathbf{x};\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \Phi_1^{[\boldsymbol{\alpha}_1]}(\mathbf{x}), \ldots, \Phi_N^{[\boldsymbol{\alpha}_1]}(\mathbf{x}) \\ \vdots \quad \ddots \quad \vdots \\ \Phi_1^{[\boldsymbol{\alpha}_L]}(\mathbf{x}), \ldots, \Phi_N^{[\boldsymbol{\alpha}_L]}(\mathbf{x}) \end{pmatrix} \cdot \begin{pmatrix} u_1^R \\ \vdots \\ u_N^R \end{pmatrix} \tag{47}$$

Therefore, the singular derivative approximation is obtained as follows

$$\begin{pmatrix} D_{\mathbf{x}}^{\boldsymbol{\alpha}_1} u^S(\mathbf{x};\mathbf{x}) \\ \vdots \\ D_{\mathbf{x}}^{\boldsymbol{\alpha}_L} u^S(\mathbf{x};\mathbf{x}) \end{pmatrix} = \begin{pmatrix} D_{\mathbf{x}}^{\boldsymbol{\alpha}_1} u^{cont}(\mathbf{x};\mathbf{x}) \\ \vdots \\ D_{\mathbf{x}}^{\boldsymbol{\alpha}_L} u^{cont}(\mathbf{x};\mathbf{x}) \end{pmatrix} + \begin{pmatrix} D_{\mathbf{x}}^{\boldsymbol{\alpha}_1} u^{disc}(\mathbf{x};\mathbf{x}) \\ \vdots \\ D_{\mathbf{x}}^{\boldsymbol{\alpha}_L} u^{disc}(\mathbf{x};\mathbf{x}) \end{pmatrix}, \quad \mathbf{x} \in \Omega^S \backslash \Gamma \tag{48}$$

Finally, the EPDA takes the following form:

$$\begin{aligned} D_{\mathbf{x}}^{\boldsymbol{\alpha}} u^S(\mathbf{x};\mathbf{x}) = & \sum_{I=1}^{N} \Phi_I^{[\boldsymbol{\alpha}]}(\mathbf{x})\, u_I \\ & + \frac{1}{2}[u]_{\mathbf{x}_\Gamma} \left(D_{\mathbf{x}}^{\boldsymbol{\alpha}} b_s(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\boldsymbol{\alpha}]}(\mathbf{x})\, b_s(\mathbf{x}_I;\mathbf{x}) \right) \\ & + \frac{1}{2}\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{x}_\Gamma} \left(D_{\mathbf{x}}^{\boldsymbol{\alpha}} b_n(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\boldsymbol{\alpha}]}(\mathbf{x})\, b_n(\mathbf{x}_I;\mathbf{x}) \right) \\ & + \frac{1}{2}\frac{\partial [u]_{\mathbf{x}_\Gamma}}{\partial \mathbf{t}} \left(D_{\mathbf{x}}^{\boldsymbol{\alpha}} b_t(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\boldsymbol{\alpha}]}(\mathbf{x})\, b_t(\mathbf{x}_I;\mathbf{x}) \right) \end{aligned} \tag{49}$$

In this equation, except for the u_I values, the remaining terms on the right hand side are wholly computable if the jump magnitudes are properly provided. The u_I values are obtained by numerically solving the PDE. Because the Taylor polynomial based on the moving least squares method approximates the continuous (or non-singular) function only, the mathematical clarity of the EPDA is not spoiled. Furthermore, this approach somewhat resembles those of Benzley [36] and Gifford and Hilton [37], which were proposed in the frame of the FEM for an elastic crack analysis. They employed the moving least squares method to derive the cracked displacement field that models the near-tip field by means of interpolation with stress intensity factors.

The EPDA generates discontinuity strengths corresponding to the type of jump operation at the projection point of $\mathbf{x}$. For example, taking the jump of the singular approximation at $\mathbf{x}_\Gamma$ yields the following:

$$\left[u^S(\mathbf{x};\mathbf{x})\right]_{\mathbf{x}_\Gamma} = \frac{1}{2}[u]_{\mathbf{x}_\Gamma}[b_s(\mathbf{x};\mathbf{x})]_{\mathbf{x}_\Gamma} = [u]_{\mathbf{x}_\Gamma} \tag{50}$$

In addition, $\left[\frac{\partial u^S(\mathbf{x};\mathbf{x})}{\partial \mathbf{n}_\Gamma(\mathbf{x})}\right]_{\mathbf{x}_\Gamma} = \left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{x}_\Gamma}$ and $\frac{\partial [u^S(\mathbf{x};\mathbf{x})]_{\mathbf{x}_\Gamma}}{\partial \mathbf{t}_\Gamma(\mathbf{x})} = \frac{\partial [u]_{\mathbf{x}_\Gamma}}{\partial \mathbf{t}}$ are obtained in a similar manner. In general, the discontinuity strengths of the EPDA are directly given from the interface conditions of the governing equations. However, when the interface conditions are not explicitly given from the governing equations, the discontinuity strengths become unknown and additional difference equations must be established to identify them. This leads to an increase in the size of the system overall but still yields an even-determined system in which the number of equations and the number of unknowns are identical.

To clarify the concept of the EPDA, the composition of the singular approximation is illustrated here. The one dimensional singular function shown below is tested by the EPDA:

$$u(x) = -\frac{1}{10}x^2 + x + 2sign(x) - |x| \tag{51}$$

As shown in Fig. 7a–d, the above function consists of the continuous part $(-\frac{1}{10}x^2 + x)$, the step part $(2sign(x))$, and the wedge part $(-|x|)$. Figure 8a–d shows the reproduced functions of the original function using the EPDA; the reproduced functions were computed using the exact nodal solution and discontinuity strengths obtained by Eq. (51). For the test, a numerical model with 40 equally spaced nodes was used. The continuous part, step part, and wedge part are clearly and sharply reproduced using the EPDA. Because the original function is essentially a quadratic function except for the step and wedge parts, it can be completely reproduced by the EPDA based on the quadratic Taylor polynomial, the step function, and the linear wedge function. When the regularity or reproducibility of the EPDA is insufficient for a given singular function; the regularity of the jump function should be elevated. It is an interesting feature of the EPDA that the regularity of the jump function is controllable and the consistency of the Taylor polynomial is extended without difficulty.

One of the main concerns in this study is how accurately the EPDA can reproduce a singular function. Kim et al. [32] numerically verified the performance of enriched meshfree approximation by means of a reproducing test. The test checked if the given function can be reproduced by plugging the exact values for the nodal solution and jump magnitude taken from the given function into the enriched meshfree approximation. Error estimation was done by measuring the difference between the reproduced and the exact values. As performed here, the EPDA can reproduce the polynomial space up to the order of consistency as well as the discontinuous function up to the regularity of the jump functions. When the given function is continuous, the step, wedge, and scissors parts automatically become zero and the original function is reproduced by the continuous part of the EPDA.

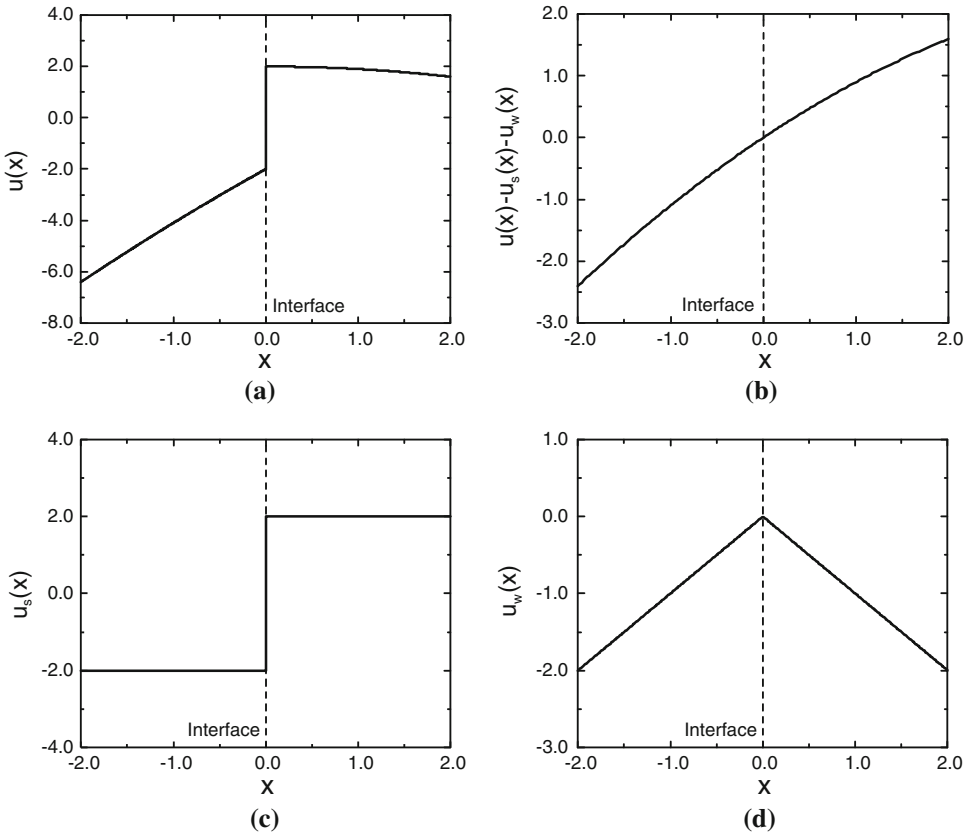

Fig. 7 Decomposition of a singular function with the weak and strong discontinuities into the continuous, step and wedge parts; i.e. (a) = (b) + (c) + (d), with $u_s\,(x)$ and $u_w\,(x)$ indicating the step part and the wedge part, respectively: **a** $u\,(x)$, **b** $u\,(x) - u_s\,(x) - u_w\,(x)$, **c** $u_s\,(x)$ and **d** $u_w\,(x)$

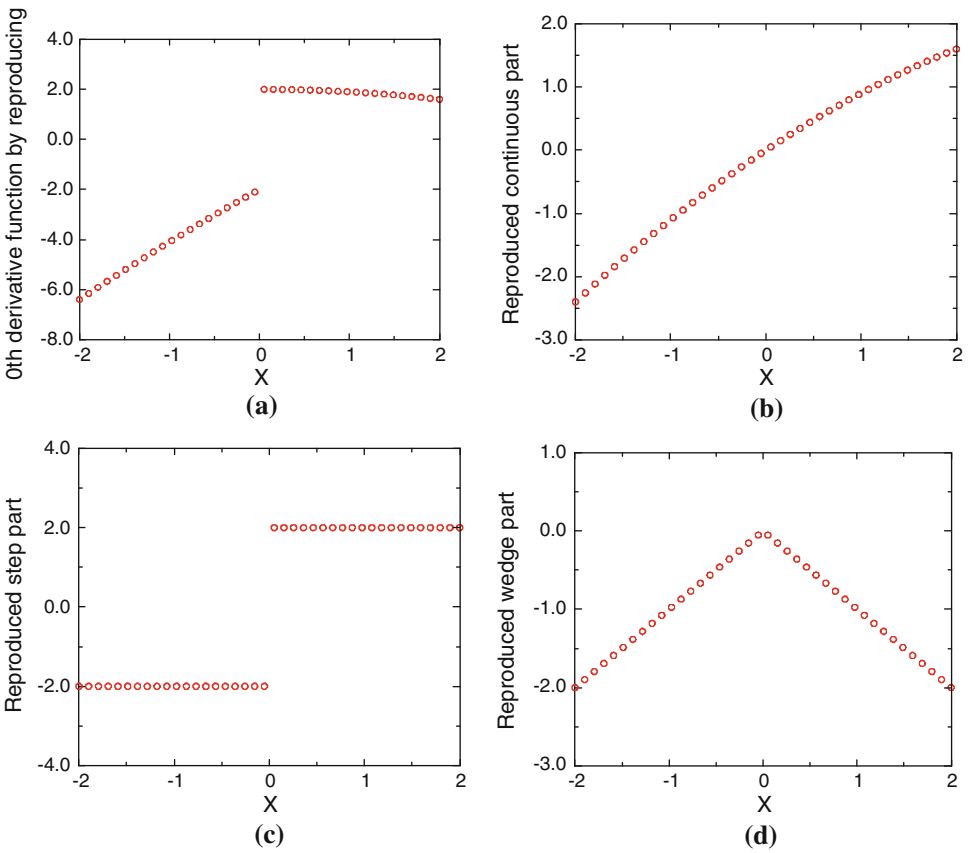

Fig. 8 Reproduced functions for the original singular function; the continuous, step, and wedge parts are separately reproduced; i.e. (a) = (b) + (c) + (d) (in the reproducing test, a 40 node model is used). **a** $\sum_{I=1}^{N} \Phi_I^{[\alpha]}\,(x)\left(u_I - \frac{1}{2}\,[u]_{x_\Gamma}\,b_s\,(x_I;x) - \frac{1}{2}\left[\frac{du}{dx}\right]_{x_\Gamma} b_n\,(x_I;x)\right) + \frac{1}{2}\,[u]_{x_\Gamma}\,b_s\,(x;x) + \frac{1}{2}\left[\frac{du}{dx}\right]_{\Gamma} b_n\,(x;x)$, **b** $\sum_{I=1}^{N} \Phi_I^{[\alpha]}\,(x)\left(u_I - \frac{1}{2}\,[u]_{x_\Gamma}\,b_s\,(x_I;x) - \frac{1}{2}\left[\frac{du}{dx}\right]_{x_\Gamma} b_n\,(x_I;x)\right)$, **c** $\frac{1}{2}\,[u]_{x_\Gamma}\,b_s\,(x;x)$ and **d** $\frac{1}{2}\left[\frac{du}{dx}\right]_{x_\Gamma} b_n\,(x;x)$

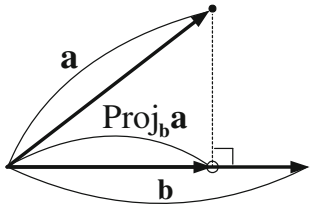

Fig. 9 The concept of a projection of one vector onto another vector

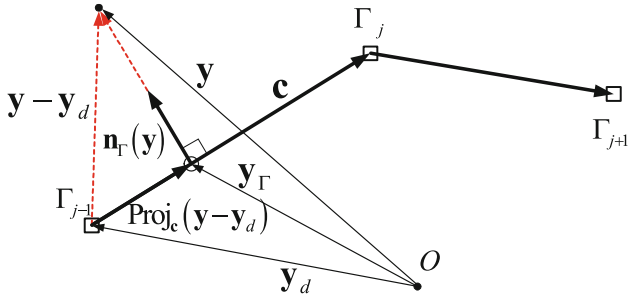

Fig. 10 A schematic drawing of normal vector determination based on vector operations

3.5 Discrete form of the extended particle derivative approximation

When solving weak and strong discontinuity problems, both the interior domain and the boundaries are discretized by nodes; moreover, the interface needs to be discretized for adequate definitions of jump functions. The interface, which is a curve in general, is discretized by a series of line segments. The interface conditions are embedded in the EPDA in the form of jump magnitudes. The position vector of the projection point of a local center (or sampling point) and the normal vector are essential to the construction of the EPDA. Given that the jump magnitudes are defined only at the segmental point and that the projection point can fall anywhere on the segments, the jump magnitudes need to be interpolated.

To find the projection point of a given point, a vectorial operation involving a projection map is introduced. Figure 9 illustrates the concept of the projection. The first step is to find the closest segmental point $\mathbf{y}_d$ on Γ from $\mathbf{y}$; i.e.

$$\mathbf{y}_d = \arg\left(\inf_{\mathbf{x}\in\hat{\Gamma}} \|\mathbf{y}-\mathbf{x}\|\right) \tag{52}$$

where $\hat{\Gamma}$ denotes the set of interfacial segments, namely, $\hat{\Gamma} = \bigcup_{j=1}^{l} \overline{\Gamma_j \Gamma_{j+1}}$ where l is the total number of segments and Γ_j is the segmental point.

A projection vector, which projects vector $\mathbf{a}$ onto $\mathbf{b}$, is defined as

$$\mathrm{Proj}_{\mathbf{b}}\mathbf{a} = \frac{\mathbf{a}\cdot(\mathbf{b}\otimes\mathbf{b})}{\mathbf{b}\cdot\mathbf{b}} \tag{53}$$

where $\otimes$ denotes the tensor product operator (as in Fig. 9). When $\mathbf{c}$ is the vector of the current segment connected to Γ_{j-1} and Γ_j, of which the starting point is Γ_{j-1}, the projection point $\mathbf{y}_\Gamma$ is determined by

$$\mathbf{y}_\Gamma = \mathbf{y}_d + Proj_{\mathbf{c}}(\mathbf{y}-\mathbf{y}_d) \tag{54}$$

where $\mathbf{y}$, $\mathbf{y}_d$, and $\mathbf{y}_\Gamma$ are the position vectors. The unit normal vector is then defined as

$$\mathbf{n}_\Gamma(\mathbf{y}) = \frac{(\mathbf{y}-\mathbf{y}_d) - Proj_{\mathbf{c}}(\mathbf{y}-\mathbf{y}_d)}{\|(\mathbf{y}-\mathbf{y}_d) - Proj_{\mathbf{c}}(\mathbf{y}-\mathbf{y}_d)\|} \tag{55}$$

Note that all terms on the right hand side of the above equation are computable. Figure 10 illustrates the definitions of the position vectors and the relevant vector operations. This procedure makes the finding of the normal vector highly strategic and is thus suitable for a computer program. Also, this procedure may work perfectly in a three dimensional case without any modification.

After finding the projection point, it should be determined whether $\mathbf{y}_\Gamma$ is on the current segment. This can be easily done by checking the sign of the scholar product $(\mathbf{y}-\mathbf{y}_d)\cdot\mathbf{c}$. For example, when $(\mathbf{y}-\mathbf{y}_d)\cdot\mathbf{c} > 0$ and $\left\|\mathrm{Proj}_{\mathbf{c}}(\mathbf{y}-\mathbf{y}_d)\right\| < \|\mathbf{c}\|$, then the projection point of $\mathbf{y}$ is $\mathbf{y}_\Gamma$. On the other hand, when $(\mathbf{y}-\mathbf{y}_d)\cdot\mathbf{c} < 0$; i.e. when $\mathbf{y}-\mathbf{y}_d$ has a different direction from vector $\mathbf{c}$, then $\mathbf{y}_\Gamma$ falls onto the outside of the current segment.

Let us consider the discrete form of the EPDA. Provided that the jump magnitudes are interpolated by a linear interpolant, the discrete form of the EPDA is written as

$$\begin{aligned} D_{\mathbf{x}}^{\alpha} u^S(\mathbf{x};\mathbf{x}) = &\sum_{I=1}^{N} \Phi_I^{[\alpha]}(\mathbf{x})\, u_I \\ &+\frac{1}{2}\left(\sum_{J=1}^{2} N_J(\mathbf{x}_\Gamma)\,\delta_J\right)\Psi^{[\alpha]}(\mathbf{x}) \\ &+\frac{1}{2}\left(\sum_{J=1}^{2} N_J(\mathbf{x}_\Gamma)\,\omega_J\right)\bar{\Psi}^{[\alpha]}(\mathbf{x}) \\ &+\frac{1}{2}\left(\sum_{J=1}^{2} \frac{\partial N_J(\mathbf{x}_\Gamma)}{\partial \mathbf{t}}\delta_J\right)\bar{\bar{\Psi}}^{[\alpha]}(\mathbf{x}) \end{aligned} \tag{56}$$

where the generalized singular shape functions for jumps in the solution, normal derivative, and tangential derivatives are respectively expressed as

$$\Psi^{[\alpha]}(\mathbf{x}) = D_{\mathbf{x}}^{\alpha} b_s(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\alpha]}(\mathbf{x})\, b_s(\mathbf{x}_I;\mathbf{x}) \tag{57}$$

$$\bar{\Psi}^{[\alpha]}(\mathbf{x}) = D_{\mathbf{x}}^{\alpha} b_n(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\alpha]}(\mathbf{x})\, b_n(\mathbf{x}_I;\mathbf{x}) \tag{58}$$

$$\bar{\bar{\Psi}}^{[\alpha]}(\mathbf{x}) = D_{\mathbf{x}}^{\alpha} b_t(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\alpha]}(\mathbf{x})\, b_t(\mathbf{x}_I;\mathbf{x}) \tag{59}$$

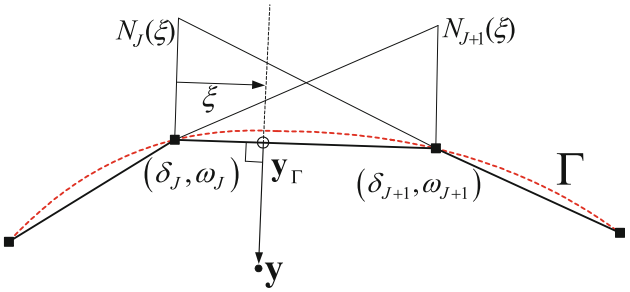

Fig. 11 Discretization of the interface using line segmentation; linear interpolation is used to approximate the jump magnitudes for the solution and normal derivative jumps

It can be seen that the generalized singular shape functions above consist of the full derivatives of the jump functions and the interpolation of the nodal values of the jump functions based on the regular generalized shape function. Because linear interpolation is adopted in the description of jump functions along the segmented interface, Eq. (56) was obtained by applying the following relationships to Eq. (49):

$$[u]_{\mathbf{x}_\Gamma} = \sum_{J=1}^{2} N_J(\xi)\delta_J \tag{60}$$

$$\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{x}_\Gamma} = \sum_{J=1}^{2} N_J(\xi)\omega_J \tag{61}$$

where ξ is a local coordinate defined on the segment of interest; δ_J denotes the jump magnitude of the solution at segmental point J; and ω_J is the jump magnitude of the normal derivative. Additionally, $N_J(\mathbf{x}_\Gamma)$ is the Lagrange linear interpolant. Practically, the use of linear interpolation for the jump magnitudes is recommended for the sake of computational efficiency. It never ruins the resolution of the numerical solution when combined with the second-order Taylor polynomial. Figure 11 depicts the numerical representation of the interface based on linear interpolation.

When $[u]_{\mathbf{x}_\Gamma}$ and $\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{x}_\Gamma}$ are given from the governing equations, δ_J and ω_J can directly be applied to the EPDA without generating an extra difference equation to enforce the interface conditions. In fact, the jump related terms are transferred to the right hand side of the difference equations for the equilibrium equation, serving as forcing terms which may be compared to Dirac's delta function in the IBM formulation [4]. However, when the interface conditions are not explicitly given in terms of $[u]_{\mathbf{x}_\Gamma}$ and $\left[\frac{\partial u}{\partial \mathbf{n}}\right]_{\mathbf{x}_\Gamma}$, an additional difference equation should be formulated from the relevant interfacial relationship.

3.6 Comparison with other enrichment methods

There have been two approaches to enrich the approximation function in particle or meshfree methods. One is intrinsic enrichment [20,32,38] and the other is extrinsic enrichment [2,20,31,39]. The major difference between the two enrichment methods mostly revolves around the target of enrichment. While the intrinsic enrichment approach extends the polynomial basis with enrichment functions, the extrinsic enrichment approach adds the enrichment function not to the polynomial but to the local approximation in an extrinsic manner. Although both enrichment approaches utilize the moving least squares method, they generate quite different forms of approximation. For example, the strength of singularity explicitly appears in the extrinsic formula but not in the intrinsic formula. Furthermore, in the intrinsic enrichment method, a careful selection of the enrichment function is required to preserve the invertibility of the moment matrix; the jump function should be linearly independent of other basis components. However, this is not necessarily required in the extrinsic enrichment method.

Furthermore, intrinsic enrichment can produce an overdetermined system at times due to the disagreement between the number of equations and the number of unknowns. This feature can lead to the deterioration of the level of solution accuracy and degrade the solvability of the total system compared to the extrinsic approach. On the other hand, extrinsic enrichment always produces an even-determined system. In the extrinsic formulation, the physical meaning of the singularity strength, such as that of the jump magnitude, is clear; meanwhile, the singularity strength does not even emerge from the intrinsic formulation. Compared to the intrinsic formulation, the extrinsic approach does not provoke any mathematical ambiguities. As a result, because the EPDA belongs to the extrinsic enrichment family, it can utilize all of these attractive advantages of extrinsic formulation.

The XFEM [1,3,35] is a popular numerical method that considers discontinuity problems in a tactical manner. The computational simplicity mostly results from the fact that discontinuity modeling is independent of a mesh structure. When excluding crack tip enrichment, the XFEM approximation for displacement takes the form of

$$\mathbf{u}^h(\mathbf{x}) = \sum_I \Phi_I^{FE}(\mathbf{x})\mathbf{u}_I + \sum_I \Phi_I^{FE}(\mathbf{x})H(\mathbf{x})\mathbf{d}_I \tag{62}$$

where $\Phi_I^{FE}(\mathbf{x})$ is the standard FE shape function and $H(\mathbf{x})$ is a step function defined along the discontinuity line; in addition, $\mathbf{u}_I$ is the nodal displacement vector and $\mathbf{d}_I$ is an additional nodal unknown related to the step function. The second term on the right hand side of Eq. (62) provides discontinuity modeling that is distinguished from the conventional FE discontinuity modeling method based on mesh splitting [40,41]. It should also be noted that the FE shape function serves as a projection map that projects the distance measured from $\mathbf{x}$ to the discontinuity line onto the nodal unknown $\mathbf{d}_I$. The key idea of discontinuity modeling is that the jump across the discontinuity line is interpolated by $\Phi_I^{FE}(\mathbf{x})\,H(\mathbf{x})$. Although

$\mathbf{d}_I$ does not directly indicate the jump magnitude, the jump magnitude at any position can be determined by interpolating the $\mathbf{d}_I$ values. As a result, the discontinuity modeling of the XFEM relies considerably on the sharpness of the FE shape function.

As an improved version of Eq. (62), Zi and Belytschko [3] proposed the following formula using a shifted nodal sign function:

$$\mathbf{u}^h(\mathbf{x}) = \sum_I \Phi_I^{FE}(\mathbf{x})\mathbf{u}_I + \sum_I \Phi_I^{FE}(\mathbf{x}) H_I(\mathbf{x})\mathbf{d}_I \tag{63}$$

The step function is given by

$$H_I(\mathbf{x}) = sign(\phi(\mathbf{x})) - sign(\phi_I) \tag{64}$$

where $\phi(\mathbf{x})$ denotes the level set function, indicating the signed distance from the crack surface line, i.e. $\phi(\mathbf{x}) = 0$. By doing so, all cracked elements are enriched without the blending of the local partition of unity; i.e. the enrichment region is connected to the standard approximation region without a blending function. As shown in Eq. (62) or (63), the jump enrichment of the XFEM is not intrinsic. However, the jump magnitude does not appear to emerge such that it is difficult to define the jump magnitude explicitly at an arbitrary position.

On the other hand, the jump magnitude extrinsically appears in the EPDA. When considering the step function only, the EPDA is expressed as follows

$$\begin{aligned} u^S(\mathbf{x};\mathbf{x}) &= \sum_{I=1}^{N} \Phi_I^{[\mathbf{0}]}(\mathbf{x})\, u_I \\ &\quad + \frac{1}{2}[u]_{\mathbf{x}_\Gamma}\left(b_s(\mathbf{x};\mathbf{x}) - \sum_{I=1}^{N} \Phi_I^{[\mathbf{0}]}(\mathbf{x})\, b_s(\mathbf{x}_I;\mathbf{x})\right) \\ &= \sum_{I=1}^{N} \Phi_I^{[\mathbf{0}]}(\mathbf{x})\left(u_I - \frac{1}{2}[u]_{\mathbf{x}_\Gamma} b_s(\mathbf{x}_I;\mathbf{x})\right) \\ &\quad + \frac{1}{2}[u]_{\mathbf{x}_\Gamma} b_s(\mathbf{x};\mathbf{x}) \end{aligned} \tag{65}$$

The above approximation appears quite different from the XFEM approximation. The physical meaning of $\mathbf{d}_I$ in the XFEM is different from that of $\frac{1}{2}[u]_{\mathbf{x}_\Gamma}$ above. The zeroth order generalized regular shape function $\Phi_I^{[\mathbf{0}]}(\mathbf{x})$ interpolates a regular part of the nodal solution; the regular nodal solution means the nodal solution subtracted by the nodal value of the step function, i.e. $u_I - \frac{1}{2}[u]_{\mathbf{x}_\Gamma} b_s(\mathbf{x}_I;\mathbf{x})$. The singular part is treated separately via $\frac{1}{2}[u]_{\mathbf{x}_\Gamma} b_s(\mathbf{x};\mathbf{x})$.

4 Conclusions

The final goal of this study is to develop the extend particle difference method (EPDM) for solving weak and strong discontinuity problems, such as heat transfer problems in composite materials and elasticity problems with material discontinuity. In this study, the EPDA was derived based on the Taylor polynomial expanded by the moving least squares method; the EPDA appropriately approximates the singular solution with weak and strong discontinuities by associating the generalized shape function with the nodal solution and discontinuity strength. The discontinuous functions were designated to capture jumps in the solution, the normal and tangential derivative fields, which are defined on the basis of the concept of the tangent hyperplane function. The use of the discontinuous functions in the framework of the tangent hyperplane function effectively minimizes the actual differentiation process in the derivative formula without sacrificing mathematical robustness up to the order of consistency. The EPDA facilitates a faster derivative computation and preserves the reproducing property for the polynomial basis and the discontinuous functions. Unlike the FEM, which is restricted by the element connectivity, the EPDM can easily handle an interfacial singularity based on a regular node set. Also, the EPDM is equipped with a derivative approximation function, which the FDM does not provide. Furthermore, unlike conventional particle methods such as the EFGM or RKPM, which are based on the weak formulation, the proposed method considerably elevates the computational efficiency through its use of the strong formulation with no numerical integration. Therefore, it can be referred to as a 'truly' meshfree method that completely overcomes mesh or grid dependency. This node-wise discretization plays a salient role, even in interface problems, when used with the EPDA. The complex relationship between the geometry of the interface and the singularity of a given function is effectively dealt with. Thus, the EPDA can be readily applied to solving burdensome problems related to interfacial singularities.

The EPDA can be classified as an extrinsic enrichment method owing to the apparently extracted discontinuity strengths from the shape function. In fact, it takes advantage of the diffuse derivative technique [6], and it sharply captures wedge and jump behaviors without computing the derivative of the weight function or using the derivative of inverted moment matrix. Also, extrinsic jump modeling does not deteriorate the advantages of mesh-independency. The requirement of high order derivatives in the strong formulation is effectively satisfied by the EPDA; the EPDA circumvents the full differentiation of the approximation function. Furthermore, it nicely overcomes the lack of accuracy found in intrinsic enrichment methods. For example, the meshfree point collocation method utilizes intrinsic enrichment to solve weak discontinuity problems [32]; this method is associated with a lack of accuracy resulting from the intrinsically constructed approximation, tracing singular behaviors in a least squares sense, as well as with a lack of efficiency

resulting from the creation of an over-determined system, provoking a mismatch between the number of equations and the number of unknowns. However, the extrinsic approach in this study yields an even-determined discrete system in which the number of unknowns is always equal to the number of equations.

It is well known that strong form based particle methods have valuable advantages when used to solve problems with an internal boundary, especially those with a geometrically complex interface. In contrast, weak form based particle methods cannot achieve this due to their use of numerical integration. The EPDA combined with the strong formulation can form a complete node-wise discretization scheme; this approach accelerates the computation speed not only for calculating the derivatives of the solution but also for building up the discrete system. Meanwhile, additional discretization for the interfacial geometry is necessary. A series of line segments is introduced for the enforcement of the interface condition, and the resolution of this segmentation is required to maintain the resolution level in nodal discretization. As a result, both the nodal solution and discontinuity strength along the segmented interface are obtained by solving the discrete system at the same time. In a reproducing property test of a simple singular function, it was shown that the EPDA accurately reproduces not only the continuous part of the singular function but also the step and wedge parts.

In the second part of this paper, strong formulations are developed for heat transfer problems in composite materials, potential problems with weak and strong discontinuities, and elasticity problems with a material discontinuity. Difference equations are constructed for the given governing equations, resulting in an even-determined discrete system. The robustness and efficiency of the formulations are clearly provided through various numerical experiments. In addition, numerical results show that the developed methods are very promising in treating interfacial singularities of the types that confound other numerical methods.

Acknowledgments The first author gratefully acknowledge the support of Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0006050). The second author also gratefully acknowledge the support of the Office of Naval Research under Grant N00014-13-1-0386.

References

1. Moës N, Dolbow J, Belytschko T (1999) A finite element method for crack growth without remeshing. Int J Numer Methods Eng 46:131–150
2. Ventura G, Xu J, Belytschko T (2002) A vector level set method and new discontinuity approximations for crack growth by EFG. Int J Numer Methods Eng 54:923–944
3. Zi G, Belytschko T (2003) New crack-tip elements for XFEM and applications to cohesive cracks. Int J Numer Methods Eng 57:2221–2240
4. Tu C, Peskin CS (1992) Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods. SIAM J Sci Stat Comput 13:1361–1376
5. LeVeque RJ, Li Z (1994) The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. SIAM J Numer Anal 31:1019–1044
6. Nayroles B, Touzot G, Villon P (1992) Generalizing the finite element method: diffuse approximation and diffuse elements. Comput Mech 10:307–318
7. Krongauz Y, Belytschko T (1997) Consistent pseudo-derivatives in meshless methods. Comput Methods Appl Mech Eng 146:371–386
8. Krongauz Y, Belytschko T (1997) A Petrov–Galerkin diffuse element method (PG DEM) and its comparison to EFG. Comput Mech 19:327–333
9. Belytschko T, Lu YY, Gu L (1994) Element-free Galerkin methods. Int J Numer Methods Eng 37:229–256
10. Liu WK, Jun S, Zhang Y (1995) Reproducing kernel particle methods. Int J Numer Methods Fluids 20:1081–1106
11. Monaghan JJ (1992) Smoothed particle hydrodynamics. Annu Rev Astron Astrophys 30:543–574
12. Kim DW, Kim Y-S (2003) Point collocation methods using the fast moving least square reproducing kernel approximation. Int J Numer Methods Eng 56:1445–1464
13. Lee S-H, Yoon Y-C (2004) Meshfree point collocation method for elasticity and crack problem. Int J Numer Methods Eng 61:22–48
14. Oñate E, Idelsohn S, Zienkiewicz OC, Taylor RL, Sacco C (1996) A stabilized finite point method of analysis of fluid mechanics problems. Comput Methods Appl Mech Eng 139:315–346
15. Oñate E, Perazzo F, Miquel J (2001) A finite point method for elasticity problems. Comput Struct 79:2151–2163
16. Aluru NR (2000) A point collocation method based on reproducing kernel approximations. Int J Numer Methods Eng 47:1083–1121
17. Huerta A, Vidal Y, Villon P (2004) Pseudo-divergence-free element free Galerkin method for incompressible fluid flow. Comput Methods Appl Mech Eng 193:1119–1136
18. Li S, Liu WK (2002) Meshfree and particle methods and their applications. Appl Mech Rev 55:1–34
19. Li S, Liu WK (1999) Reproducing kernel hierarchical partition of unity: part I. Formulation and theory. Int J Numer Methods Eng 45:251–288
20. Fleming M, Chu YA, Moran B, Belytschko T (1997) Enrichment element-free Galerkin methods for crack tip fields. Int J Numer Methods Eng 40:1483–1504
21. Belytschko T, Fleming M (1999) Smoothing, enrichment and contact in the element-free Galerkin method. Comput Struct 71:173–195
22. Krongauz Y, Belytschko T (1998) EFG approximation with discontinuous derivatives. Int J Numer Methods Eng 41:1215–1233
23. Rabczuk T, Belytschko T (2004) Cracking particles: a simplified meshfree method for arbitrary evolving cracks. Int J Numer Methods Eng 61(13):2316–2343
24. Rabczuk T, Zi G (2007) A meshfree method based on the local partition of unity for cohesive cracks. Comput Mech 39(6):743–760
25. Rabczuk T, Belytschko T (2007) A three dimensional large deformation meshfree method for arbitrary evolving cracks. Comput Methods Appl Mech Eng 196(29–30):2777–2799
26. Zi G, Rabczuk T, Wall WA (2007) Extended meshfree methods without branch enrichment for cohesive cracks. Comput Mech 40(2):367–382
27. Rabczuk T, Bordas S, Zi G (2007) A three-dimensional meshfree method for continuous multiplecrack initiation, nucleation and propagation in statics and dynamics. Comput Mech 40(3):473–495

28. Rabczuk T, Zi G, Bordas S, Nguyen-Xuan H (2010) A simple and robust three dimensional cracking-particle method without enrichment. Comput Methods Appl Mech Eng 199(37–40):2437–2455
29. Cordes LW, Moran B (1996) Treatment of material discontinuity in the element-free Galerkin method. Comput Methods Appl Mech Eng 139:75–89
30. Luo Y, Häussler-Combe U (2002) A generalized finite-difference method based on minimizing global residual. Comput Methods Appl Mech Eng 191:1421–1438
31. Kim DW, Yoon Y-C, Liu WK, Belytschko T (2007) Extrinsic meshfree approximation using asymptotic expansion for interfacial discontinuity of derivative. J Comput Phys 221:370–394
32. Kim DW, Liu WK, Yoon Y-C, Belytschko T, Lee S-H (2007) Meshfree point collocation method with intrinsic enrichment for interface problems. Comput Mech 40(6):1037–1052
33. Kim DW, Kim H-K (2004) Point collocation method based on the FMLSRK approximation for electromagnetic field analysis. IEEE Trans Magn 40(2):1029–1032
34. Legay A, Wang H-W, Belytschko T (2005) Strong and weak arbitrary discontinuities in spectral finite elements. Int J Numer Methods Eng 64:991–1008
35. Chessa J, Smolinski P, Belytschko T (2002) The extended finite element method (XFEM) for solidication problems. Int J Numer Methods Eng 53(8):1959–1977
36. Benzley SE (1974) Representation of singularities with isotropic finite elements. Int J Numer Methods Eng 8:537–545
37. Gifford LN Jr, Hilton PD (1978) Stress intensity factors by enriched finite elements. Eng Fract Mech 10:485–496
38. Yoon Y-C, Lee S-H, Belytschko T (2006) Enriched meshfree collocation method with diffuse derivatives for elastic fracture. Comput Math Appl 51(8):1349–1366
39. Yoon Y-C, Kim DW (2010) Extended meshfree point collocation method for electromagnetic problems with layered singularity. IEEE Trans Magn 46(8):2951–2954
40. Swenson DV, Ingraffea A (1988) Modeling mixed-mode dynamic crack propagation using finite elements: theory and applications. Comput Mech 3:381–397
41. Martha L, Wawrzynek P, Ingraffea A (1993) Arbitrary crack propagation using solid modeling. Eng Comput 9:63–82