An Introduction to Moving Least Squares Meshfree Methods

Piotr Breitkopf — Alain Rassineux — Pierre Villon

Laboratoire de Mécanique Roberval, UMR UTC-CNRS
Université de Technologie de Compiègne
BP 20529, F-60205 Compiègne cedex
e-mail : firstname.lastname@utc.fr

ABSTRACT. *We deal here with some fundamental aspects of a category of meshfree methods based on Moving Least Squares (MLS) approximation and interpolation. These include EFG, RKPM and Diffuse Elements. In this introductory text, we discuss different formulations of the MLS from the point of view of numerical precision and stability. We talk about the issues of both "diffuse" and "full" derivation and we give proof of convergence of both approaches. We propose different algorithms for the computation of MLS based shape functions and we give their explicit forms in 1D, 2D and 3D. The topics of weight functions, the interpolation property with or without singular weights, the domain decomposition and the numerical integration are also discussed. We formulate the integration constraint, necessary for a method to satisfy the linear patch test. Finally, we develop a custom integration scheme, which satisfies this integration constraint.*

RÉSUMÉ. *Nous abordons ici les notions à la base d'une catégorie des méthodes sans maillage, basées sur l'approximation et l'interpolation par moindres carrés mobiles : EFG, RKPM, Eléments Diffus. Nous introduisons les différentes formulations de l'approximation du point de vue de la précision numérique et de la stabilité. Nous analysons la dérivée « diffuse » et « complète » et nous apportons la preuve de convergence des deux approches. Nous proposons différents algorithmes pour le calcul des fonctions de forme et nous donnons leur forme explicite en 1D, 2D et en 3D. Nous discutons les fonctions de pondération, la propriété d'interpolation avec et sans poids singuliers, la décomposition des domaines et l'intégration numérique. Nous formulons la contrainte d'intégration, nécessaire pour le passage du patch test. Enfin, nous développons un schéma d'intégration spécifique qui vérifie la contrainte d'intégration.*

KEYWORDS : *Meshfree Methods, Meshless Methods, Moving Least Squares, Diffuse Elements.*
MOTS-CLÉS : *méthodes sans maillage, moindres carrés mobiles, éléments diffus.*

REEF – 11/2002. Meshfree Computational Mechanics, pages 825 to 867

1. Introduction

The meshfree techniques provide a promising alternative to solving Partial Differential equations (PDE) with finite elements. The main feature of meshfree methods is the absence of an explicit mesh. The Smooth Particle Hydrodynamics (SPH, Lucy, 1977) can be seen as one of the first meshfree approaches. In this introduction we do not pretend to provide an exhaustive survey of the domain. The comprehensive works (Belytchko *et al.*, 1996 and Babuska *et al.,* 2002) may be consulted for reference. Here, we make a quick historical review.

Two main families of methods can be distinguished. The first group involves collocation methods: the Generalized Finite Difference Method (GFDM, Liszka and Orkisz, 1980), Particle in Cell (Sulsky and Schreyer, 1993), the Finite Point Method (Onate and Idesohn, 1998), The Double Grid Collocation (Breitkopf *et al.,* 2000) and the Least Squares Collocation Method (Zhang *et al.*, 2001). Galerkin-like methods were introduced by Diffuse Element Method, (DEM, Nayroles *et al.*, 1992) followed by the Element Free Galerkin Method (EFG, Belytschko *et al.,* 1994) and the Reproducing Kernel Particle Method (RKPM, Liu *et al.,* 1996). More recently appeared variational SPH (Bonet and Lok, 1999), the Meshless Local Petrov-Galerkin (MLPG, Lin and Atluri, 2000), the Method of Finite Spheres (De and Bathe, 2000). The methods of Extended Finite Element Method (XFEM, Sukumar et al., 2000) and the Partition of Unity Method (PUFEM, Babuska and Melenk, 1997) are not presented here.

In this paper, we focus on meshfree methods using the Moving Least Squares (MLS) techniques. The origins of MLS approximation can be found in independent works in several fields. In the domain of geostatistics, we find the early concept of weighted moving approximation in the work of Krige which gave rise to the term of kriging introduced later by Matheron (Krige, 1966, Matheron, 1963). In the field of non-parametric estimation in statistics, the work (Cleveland, 1979) is based on similar principles. In the field of smoothing data, we note the development of methods of approximation without solving a global system (Shepard, 1968, Mac Lain, 1974, Barnhill, 1977, Gordon *et al.*, 1978). The term of Moving Least Squares was introduced by (Lancaster and Salkauskas, 1981).

The fundamental idea behind MLS meshfree concepts aims at a better control of shape functions smoothness and continuity as opposed to the finite elements. This is obtained through the use of the weight functions. The weight functions are associated with a node and their values decrease with the distance. They allow to control the locality and the continuity of the approximation. MLS equivalent of the shape functions is derived from a minimization of a weighted least squares criterion. The difference between weights and shape functions is that the shape functions satisfy the consistency conditions necessary for the numerical solution of the PDEs. We call this process the “shape functions factory”.

In the Galerkin approach, several problems have to be solved. First, the weak form requires the numerical integrations on the boundary and inside the domain. Several authors propose different strategies. The “truly meshless” techniques (Lin and Atluri, 2000, De and Bathe, 2000) can be opposed to domain decomposition techniques (Nayroles et al., 1992, Belytschko *et al.*, 1994). An intermediary method is based on nodal integration (Beissel and Belytschko, 1996, Bonet and Lok, 1999, Chen *et al.,* 2002). The essential boundary conditions can be taken into account by nodal interpolation (Nayroles *et al.,* 1992), with Lagrange multipliers (Belytschko *et al.*, 1994) or by several modified variational principles (Babuska *et al.,* 2002).

The paper is organized as follows. Paragraph 2 details different formulations and implementation issues involved for obtaining a robust MLS approximation. The theoretical and numerical convergence is considered. In paragraph 3 we describe the shape function factory. We also give the explicit form of the shape functions in 1D, 2D and 3D. In paragraph 4, we establish the interpolation property in a general case and we give implementation details. The next paragraph discusses different strategies for the choice of domains of influence. Paragraph 6 is devoted to the integration scheme with respect of integration constraints.

2. Moving Least Squares approximation

We first introduce the Moving Least Squares (MLS) approximation following the approach (Lancaster and Salkauskas, 1986) which may be interpreted (Nayroles *et al*., 1992) as a generalization of the finite elements. An alternative method (Liszka and Orkisz, 1980), based on a local Taylor expansion reveals numerous advantages and can also be used.

We look for a local approximation of a function u_{ex} at a point *x*, based on the nodal values u_i of the function u_{ex} at a limited number of points x_i close to *x*. The unknown function u_{ex} is approximated in the vicinity of *x* by

$$u_{ex}(x) \approx u_{app}(x) = \mathbf{p}^T(x)\,\mathbf{a}(x) \tag{1}$$

The most often-used are polynomial basis functions

$$\mathbf{p}^T(x) = \begin{bmatrix} 1 & x & \cdots & x^n \end{bmatrix} \tag{2}$$

although the use of other functions, for instance trigonometric functions, has also been investigated (Belytschko *et al.,* 1994a, Savignat, 2000).

Coefficients a_i of the approximation are related to the nodal values u_i by minimizing a norm of the weighted difference between the estimated values at nodes and the nodal values u_i .

$$J_x(\mathbf{a})=\frac{1}{2}\sum_i w_i(x_i,x)\left(\mathbf{p}^T(x_i)\mathbf{a}-u_i\right)^2 \tag{3}$$

The contribution of each nodal value to the approximation is influenced by a weighting function $w(x_i,x)$ such that $w(x_i,.)>0$ inside the domain of influence of the node i and $w(x_i,.)=0$ otherwise, providing a local character to the approximation. We discuss the issues relative to the construction and to the choice of different weighting functions in section 2.2 of this paper.

Generally, MLS formulation does not interpolate data, therefore the relation

$$u_{app}(x_i)=u_{ex}(x_i) \tag{4}$$

is not verified. The interpolation property (4) is commonly obtained with the weighting functions which take infinite value at the node

$$x \to x_i \quad \Rightarrow \quad w(x_i,x)\to\infty \tag{5}$$

In this case the influence of other nodes vanishes, the approximation becomes interpolating and (4) is satisfied. In section 4 of this paper, we discuss thoroughly this issue and we present a method in order to obtain non-singular interpolating weight functions. Another way of enforcing interpolation in the context of RKPM was recently proposed (Chen *et al.*, 2002). We remark, that contrarily to the finite element interpolation, the MLS interpolation property is not sufficient for the enforcement of the essential boundary conditions. In a finite element context, the influence of the internal nodes vanishes at the boundary and the interpolation depends only on the boundary nodal values. Because of the construction of the MLS approximation itself, this property is not preserved. Thus, special treatment is needed and one of the techniques used to enforce essential boundary conditions is illustrated in paragraph 6.

2.1. *The "Diffuse" and "full" derivatives*

The derivatives of $u_{app}(x)$ may be approximated in two ways. The first form is a "full derivative", denoted by $\dfrac{du_{app}}{dx}$ and is obtained by usual derivation of both $\mathbf{p}(x)$ and $\mathbf{a}(x)$ in (1):

$$\frac{du_{app}}{dx}=\frac{d\mathbf{p}^T}{dx}(x)\mathbf{a}(x)+\mathbf{p}^T(x)\frac{d\mathbf{a}}{dx} \tag{6}$$

The second form is obtained by considering that coefficients a are constant what leads to the "diffuse derivative" denoted by $\frac{\delta}{\delta x}$

$$\frac{\delta u_{app}}{\delta x}(x)=\frac{d\mathbf{p}^T}{dx}(x)\mathbf{a}(x) \tag{7}$$

The former approach is used in the Element Free Galerkin method (Belytschko *et al.,* 1994) and the latter one is analogous to the derivatives obtained by the GFDM method (Liszka and Orkisz 1980) where second order diffuse derivatives are employed. The first order diffuse derivative was reintroduced by (Nayroles *et al.,* 1992) along with the Diffuse Element method. Both derivatives converge to the exact ones when the discretization size tends to zero.

The two derivatives are equivalent in the three following cases :

– evaluation point *x* is located at a node and the interpolating condition (4) is verified (see section 4),

– weights $w(x_i, x)$ are constant over a vicinity of x: in this case, coefficients $\mathbf{a}$ are constant, the term $\frac{d\mathbf{a}}{dx}$ vanishes and $\frac{\delta u_{app}}{\delta x}(x)\equiv\frac{du_{app}}{dx}(x)$,

– u may be expressed as a linear combination of basis functions p_i: coefficients $\mathbf{a}$ are constant in this case too.

The function converges to the first terms of its Taylor expansion when the discretization size tends to zero. Therefore, for an arbitrary function, the equivalence between the two derivatives is obtained in the limit.

The "diffuse derivative" may be intuitively interpreted as an approximation to the derivative of the function u_{ex}, while the "full derivative" is the derivative of the approximated function u_{app}. Both types of derivatives present drawbacks and advantages and the choice depends on the application. In section 2.5 we develop the interpretation of the diffuse derivative in the terms of Taylor series expansion of function *u(x)* and we demonstrate convergence properties.

2.2. *Weighting functions*

Fundamental properties related to MLS approximation, such as locality and continuity mainly depend on an appropriate choice of the weighting functions w_i. In order to limit the number of nodes used for the local evaluation, the support of the approximation must be bounded. As a consequence, the bandwidth of the resulting global linear system is also reduced. The weight function vanishes at a finite distance from $\mathbf{x}_i$, called radius of influence and denoted as r (for details see below section 2.2.3). The area around $\mathbf{x}_i$ is called domain of influence of node $\mathbf{x}_i$. Function w_i has a maximum (usually unit) value at node $\mathbf{x}_i$, remains positive and decreases continuously over the domain of influence. The choice among weight functions satisfying the above requirements depends on the application at hand. In particular, this choice is influenced by the required degree of continuity of the approximation. Whenever a "full" derivative approach is used, differentiable weights must be chosen.

2.2.1. *Window functions*

The weight functions are constructed from the reference window functions w_{ref}. When using diffuse derivative, we do not differentiate the weights and the choice of the basic hat function

$$w_{ref} = \begin{cases} 1 - s, s < 1 \\ 0, s \geq 1 \end{cases} \tag{8}$$

is straightforward. The choice of w_{ref} is also driven by performance purposes. For this reason, spline functions are preferred rather then exponentials or trigonometric functions. A C^1 piecewise cubic spline coefficients may for instance be computed to ensure the following conditions

$$w_{ref}(0) = 1_{,}\ w_{ref,x}(0) = w_{ref}(1) = w_{ref,x}(1) = 0$$

giving the expression

$$w_{ref}(s) = \begin{cases} 1 - 3s^2 + 2s^3, 0 \leq s < 1 \\ 0, s \geq 1 \end{cases} \tag{9}$$

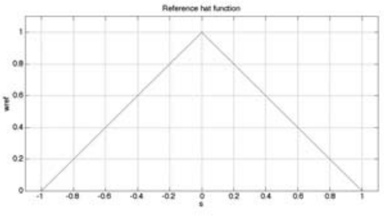


Figure 1

Hat reference window function

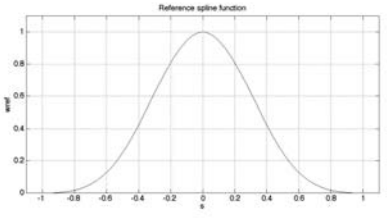


Figure 2

Spline reference window function

Figure 1 and Figure 2 show the examples of 1D reference window functions given respectively by the formulae (8) and (9).

2.2.2. *Weight functions*

Weight functions are obtained from the reference window functions by substituting the relative distance between the evaluation point and the node.

$$w(x) = w_{ref}\left(\frac{dist(x_i, x)}{r}\right) \tag{10}$$

2D and 3D weight functions can be obtained directly from w_{ref} by the use of an appropriate norm to compute the relative distance $dist(\mathbf{x}_i, \mathbf{x})/r$. Another way consists in using a tensor product of one-dimensional weights

$$\begin{gathered} w_{2D}(x,y) = w_{ref}\left(\frac{|x_i - x|}{r_x}\right) w_{ref}\left(\frac{|y_i - y|}{r_y}\right) \\ w_{3D}(x,y,z) = w_{ref}\left(\frac{|x_i - x|}{r_x}\right) w_{ref}\left(\frac{|y_i - y|}{r_y}\right) w_{ref}\left(\frac{|z_i - z|}{r_z}\right) \end{gathered} \tag{11}$$

2.2.3. *Domains of influence*

We call domain of influence of node *i*, the adherence of the set $\Omega_i = \{x / w(x_i, x) > 0\}$. Two different strategies are possible for establishing the “radius of influence” r appearing in the equation (10):

– at each evaluation point we take into account *k* closest nodes – this method is referred to as the *R(x)* strategy;

– the domains of influence are arbitrarily fixed by assigning a radius of influence to each node – this method is referred to as the r_i strategy.

Figure 3, Figure 4 and Figure 5 show different forms of domains of influence of the central node using various definitions of the radius of influence r. In all three cases $n_{\mathbf{p}}$ =3 and the radius of influence is chosen in such a way that at least 4 closest neighbors are selected. $n_{\mathbf{p}}$ is the number of terms of the polynomial basis vector $\mathbf{p}$. A regular 2D grid is used in first two figures. The Figure 3 represents $R(x)$ strategy. In this case, the domain of influence is the union of 4^{th} order Voronoi cells connected with the central node. Figure 4 shows the r_i strategy combined with L^2 norm which results in a circular domain**.** A randomly perturbed grid is used in Figure 5, where the L^∞ norm is employed in order to get a square domain of influence.

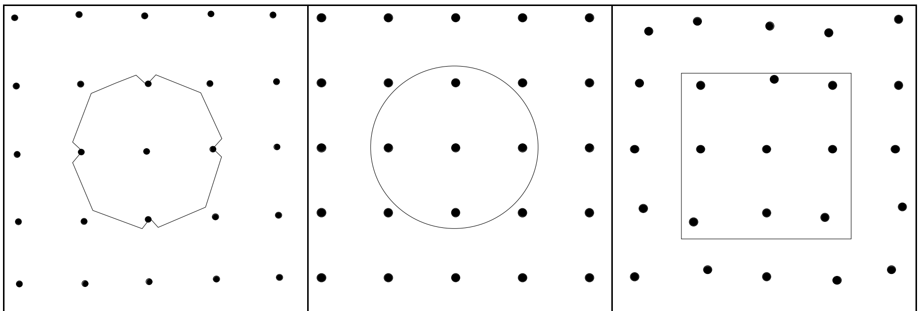


Figure 3
$R(x)$ strategy

Figure 4
r_i strategy, L^2 norm

Figure 5
r_i strategy, L^∞ norm

Different forms of domains of influence of the central node using the various definitions of the radius of influence r on a regular 2D (Figure 3, Figure 4) grid and on a randomly perturbed grid (Figure 5) with $n_\mathbf{p}$ *=3 and at least 4 closest neighbors*

The existence of the approximation requires a number of nodes at least equal to $n_\mathbf{p}$ at each evaluation point. When $n = n_\mathbf{p}$, MLS degenerates to polynomial Lagrange interpolation and the weights have no longer effect. So, in order to guarantee the continuity, the size of the domains of influence must be adjusted. In a general case, at least $n_\mathbf{p} + \mathrm{dim}$ nodes are recommended at each point of the domain, where dim is the space dimension.

2.3. *Centered Moving Least Squares*

Let us introduce a polynomial basis $\mathbf{q}$ centered at the evaluation point x. For a node x_i we have

$$\mathbf{q}^T(x_i - x) = \left\lfloor 1 \quad (x_i - x) \quad \cdots \quad \frac{(x_i - x)^k}{k!} \right\rfloor \tag{12}$$

For $k = 2$, the new basis $\mathbf{q}$ is related to the basis $\mathbf{p}$ by the following relationship

$$\mathbf{Qp}(x_i) = \mathbf{q}(x_i - x), \quad \mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ -x & 1 & 0 \\ \frac{1}{2}x^2 & -x & \frac{1}{2} \end{bmatrix} \tag{13}$$

and inversely, the basis $\mathbf{p}$ is related to the basis $\mathbf{q}$, centered at x_i by

$$\mathbf{p}(x_i) = \mathbf{Q}^{-1}\mathbf{q}(x_i - x), \quad \mathbf{Q}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ x & 1 & 0 \\ x^2 & 2x & 2 \end{bmatrix} \tag{14}$$

The matrix **Q** is nonsingular as it corresponds to a basis change in a polynomial vector space. The nodal approximation (1) becomes

$$u_{app}(x_i) = \mathbf{q}^T(x_i - x)\mathbf{Q}^{-T}(x)\mathbf{a}(x) = \mathbf{q}^T(x_i - x)\alpha(x) \tag{15}$$

where

$$\alpha(x) = \mathbf{Q}^{-T}(x)\mathbf{a}(x) \tag{16}$$

The insertion (16) into criterion (3) leads to a modified criterion which depends on vector α

$$\mathrm{J}_x(\alpha) = \frac{1}{2}\sum_j w(x_j,x)\left(\mathbf{q}^T(x_j - x)\alpha - u_j\right)^2 \tag{17}$$

The minimization of (17) yields

$$\sum_j \left(\mathbf{q}^T(x_j - x)\alpha - u_j\right) w(x_j,x)\,\mathbf{q}(x_j - x) = \mathbf{0} \tag{18}$$

thus the coefficients α are obtained from

$$\alpha(x) = \mathbf{A}(x)^{-1}\mathbf{B}(x)\mathbf{u} \tag{19}$$

where **A** and **B** matrices are given by the following formulae:

$$\begin{aligned}\mathbf{A}(x) &= \sum_i w_i \mathbf{q}(x_i - x)\mathbf{q}^T(x_i - x)\\ \mathbf{B}(x) &= \begin{bmatrix}\cdots & w_i\mathbf{q}(x_i - x) & \cdots\end{bmatrix}\end{aligned} \tag{20}$$

The algorithms based on the centered approach exhibit better conditioning properties then those using the global coordinates. In the centered approach, the condition number of the matrix $\mathbf{A}$ does not depend on the absolute position of the set of nodes.

By computing the consecutive terms of the matrix-vector product (16), we find that coefficients α are the diffuse derivatives of the approximation as introduced in (7)

$$\begin{aligned}\alpha_0 &= a_0 + a_1 x + a_2 x^2 = \mathbf{pa} = u_{app}(x)\\ \alpha_1 &= a_1 + 2a_2 x = \frac{d\mathbf{p}}{dx}\mathbf{a} = \frac{\delta u_{app}}{\delta x}(x)\\ \alpha_2 &= 2a_2 = \frac{d^2\mathbf{p}}{dx^2}\mathbf{a} = \frac{\delta^2 u_{app}}{\delta x^2}(x)\end{aligned} \tag{21}$$

2.4. *Dimensionless Moving Least Squares*

The centered approach gives better conditioned matrices $\mathbf{A}$ than formulations expressed in a global coordinate system. However, when the characteristic size of the nodal pattern decreases, near-singular matrices are obtained. The conditioning can be further improved by introducing local dimensionless coordinates $\xi_i = \dfrac{x_i - x}{h}$. Scaling factor h chosen in such a way that $0 \le \xi \le 1$, for instance $h = \max(dist(\mathbf{x}_i, \mathbf{x}))$. We write

$$\mathbf{D}(h)\mathbf{p}(x_i - x) = \mathbf{p}(\xi_i) \tag{22}$$

where

$$\mathbf{D} = \begin{bmatrix} 1 & & & 0 \\ & \frac{1}{h} & & \\ & & \ddots & \\ 0 & & & \frac{1}{h^k} \end{bmatrix} \tag{23}$$

Cost function (17) is expressed in the dimensionless coordinate system as

$$\mathrm{J}_x(\beta)=\frac{1}{2}\sum_i w(x_i,x)\left(\mathbf{q}^T(\xi)\,\beta-u_i\right)^2 \tag{24}$$

The relationship between α and β coefficients is given by diagonal matrix **D** introduced in (23)

$$\alpha=\mathbf{D}(h)\beta \tag{25}$$

We show in section 2.6 why this formulation should be preferred in practical programming.

2.5. *Convergence of the MLS approximation*

In this section we show that the diffuse derivatives (7) correspond to an approximation of a Taylor series expansion. Let us consider that function $u_{ex}(.)$ is $k+1$ times continuously differentiable. It can be proved (Villon 1991) that the vector of coefficients α converges to the vector of the “full derivatives” (6) generalized for an arbitrary order of derivation *k*

$$\mathbf{U}_{ex}(x)=\left[u_{ex}(x),\frac{du_{ex}}{dx}(x),\ldots,\frac{d^k u_{ex}}{dx^k}(x)\right]^T \tag{26}$$

The Taylor expansion of $u_{ex}(x)$ in the vicinity of point *x* gives

$$u(x_i)=\mathbf{q}^T(x_i-x)\mathbf{U}_{ex}(x)+\varepsilon_i \tag{27}$$

where ε is the truncation error. We substitute (27) into (17) and we get a criterion

$$\mathrm{J}_x(\alpha)=\frac{1}{2}\sum_j w(x_j,x)\left(\mathbf{q}^T(x_j-x)(\mathbf{U}_{ex}-\alpha)+\varepsilon_j\right)^2 \tag{28}$$

We perform the minimization of $J_x(\alpha)$ and we introduce the dimensionless coordinates (22), the associated polynomial basis (23) and the matrix **A** (20) which gives

$$\mathbf{A}(\xi)\mathbf{D}^{-1}(\mathbf{U}_{ex}-\alpha)=\sum_i w_i\varepsilon_i\mathbf{q}(\xi_i) \tag{29}$$

We note, that for a given nodal pattern, the dimensionless coordinates ξ do not depend on the pattern size *r*, thus $\mathbf{A}(\xi)$ and $\mathbf{q}(\xi_i)$ are constant too. Using the interpretation (21) of the subsequent coefficients α, the equation (26) and the fact that $\varepsilon_i=\xi_i^{k+1}\dfrac{u_{ex}^{(k+1)}(x,x_i)}{(k+1)!}$, we conclude that the error of the *l*'th "diffuse derivative" is bounded by

$$\left|\alpha_l(x)-\frac{d^l u_{ex}}{dx^l}\right|<\frac{r(x)^{k-l+1}}{(l+1)!}\left\|\frac{d^{k+1}u_{ex}}{dx^{k+1}}\right\|K(x) \tag{30}$$

In the above formula, k is the degree of the polynomial basis, r is the characteristic size of the nodal pattern. The term $\left\|\dfrac{d^{k+1}u_{ex}}{dx^{k+1}}\right\|$ depends on the regularity of the function $u_{ex}(x)$ and the term $K(x)$ is related to the local topological "quality" of the pattern. When a fixed pattern of points is used and the radius r decreases, the order of convergence of the approximation of the k-th derivative is $0(k-l)$. The value of the term $K(x)$ is related to the conditioning of the system $\mathbf{A}(\xi)^{-1}\mathbf{B}(\xi)$ and depends on the local nodal pattern taken into account in the approximation at point x. The following figures illustrate different cases:

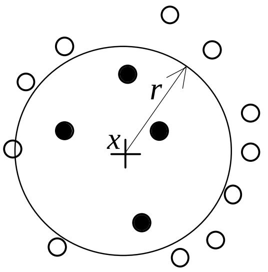


Figure 6

Well conditioned pattern with linear basis
$\mathbf{p}=[1 \quad x \quad y]$

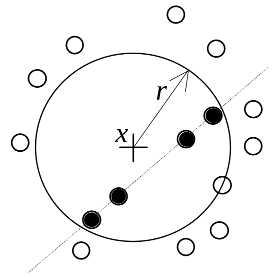


Figure 7

Pathological pattern with linear basis
$\mathbf{p}=[1 \quad x \quad y]$

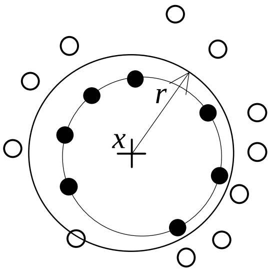


Figure 8

Pathological pattern with quadratic basis
$\mathbf{p}=\left[1 \quad x \quad y \quad \frac{x^2}{2} \quad xy \quad \frac{y^2}{2}\right]$

In these examples, we chose $w(x_i, x) = w_{ref}\left(\|x_i - x\ \|/r\right)$ where w_{ref} is a bell shaped window function, $\|.\|$ is the Euclidian norm L2 and *r*(x) is the distance from *x* to the $n_{\mathbf{p}} + 2$ neighbor node of *x*. The nodes selected at the point *x* are indicated by full dots. Figure 6 presents a well-conditioned case with a linear basis for a random distribution of nodes. Figure 7 and Figure 8 show particular cases where matrix A becomes singular, respectively with collinear points with a linear basis or with co circular points with a quadratic basis. These pathological situations are the limit cases in which the approximation cannot be performed. The patterns close to these singular ones may lead to ill-conditioned matrices and therefore spoil the convergence. We remark that these results do not depend on the choice of the weighting function and that they are similar to those obtained by (Syczewski and Tribillo, 1981) for a finite difference scheme on irregular mesh.

Expression (30) shows that when a linear polynomial basis **p** is used, the convergence of the function approximation is quadratic and the convergence of the diffuse derivative is linear. The advantage of MLS is a better control of the continuity properties provided by an appropriate choice of weight reference function w_{ref}. When a quadratic base **p** is used, a cubic convergence of the function together with a quadratic convergence of the first derivative and a linear convergence of the second derivative is obtained. This last property is mandatory when implementing a second order finite difference scheme commonly used in computational mechanics. However, the number of nodes involved in the computation and consequently the bandwidth of the resulting global system depends on the number of terms in **p** and augments with its degree. Moreover, in order to preserve the local support of the approximation, it is interesting to keep the degree of **p** reduced. Thus, a linear **p** will be frequently used in the Galerkin-like formulations while a quadratic **p** is necessary in GFDM.

2.6. *Stability of the numerical scheme*

Figure 9 and Figure 10 illustrate the behavior of the condition number of the matrix $\mathbf{A}$ for the three different formulations of the MLS criterion: $J_x(\mathbf{a})$, $J_x(\alpha)$ and $J_x(\beta)$ as introduced in the expressions (3), (17) and (24). The polynomial basis chosen for this example is linear. The evaluation points are located on a regular grid covering the 1D domain (-5,5) discretized with equidistant nodes perturbed randomly by 30%. Three closest nodes are taken into account at each evaluation point. The Schwartz window reference function is used. The condition number is defined as the ratio of the largest to the smallest eigenvalue of the matrix **A.** This value determines the precision of the obtained MLS approximation coefficients. Large values of the condition number indicate that the matrix is nearly singular.

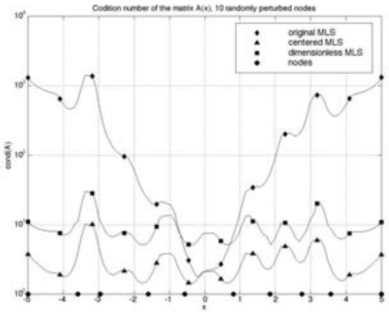


Figure 9
Distribution of the condition number of the matrix A over a 1D domain with 10 randomly perturbed nodes. $J_x(\mathbf{a})$, $J_x(\alpha)$ and $J_x(\beta)$ formulations

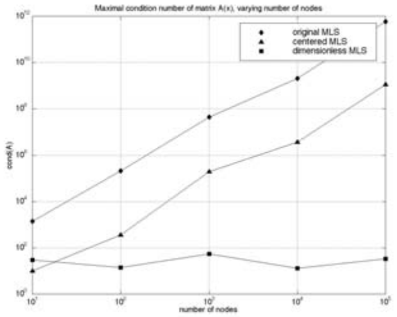


Figure 10
Maximal condition number of the matrix A over a 1D domain with increasing density of randomly perturbed nodes. $J_x(\mathbf{a})$, $J_x(\alpha)$ and $J_x(\beta)$ formulations

Figure 9 shows the variation of the condition number over the domain for the three criteria and a ten-node discretization. $J_x(\mathbf{a})$ and $J_x(\alpha)$ give similar conditioning in the vicinity of the origin of the coordinate system chosen here at the center of the domain. $J_x(\beta)$ is slightly worse then $J_x(\mathbf{a})$ and $J_x(\alpha)$, but its condition number is always lower then 100 and is largely acceptable. When the distance from the origin increases, we observe an important degradation of $J_x(\mathbf{a})$ performance while $J_x(\alpha)$ and $J_x(\beta)$ give roughly bounded and constant conditioning. Random nodal positions result in irregular oscillation of the curves.

Figure 10 illustrates the case of a progressively refined set of nodes. Three neighboring nodes are again taken at each evaluation point and consequently, the size of the domains of influence decreases. We analyze the maximal value of the condition number at each evaluation point. We observe that $J_x(\alpha)$ is always better than $J_x(\mathbf{a})$ and differs by 2 orders of magnitude. However, both formulations gradually degenerate when the number of nodes increases. The dimensionless formulation $J_x(\beta)$, while slightly worse than $J_x(\alpha)$ for very low numbers of nodes, is always well conditioned, independently from the nodal density. As in the previous figure, the slightly nonlinear character of lines is due to the random nodal positions.

Similar behavior is observed in 2D and 3D as well as for higher order polynomial basis. The use of the dimensionless formulation $J_x(\beta)$ is therefore mandatory when performing convergence studies and is strongly recommended in practical programming.

3. MLS shape functions

In the vocabulary of the finite element method, we may identify the coefficients α as the MLS equivalent of the shape functions.

In scope of the interpretation (21) we may rewrite relation (19) as

$$\begin{Bmatrix} u_{app}(x) \\ \dfrac{\delta u_{app}}{\delta x}(x) \\ \vdots \\ \dfrac{\delta u_{app}^{k}}{\delta x^{k}}(x) \end{Bmatrix} = \mathbf{A}^{-1}\mathbf{B}\{u_i\} \tag{31}$$

or, alternatively with the usual finite element notation

$$\begin{aligned} u_{app}(\mathbf{x}) &= \sum_i N_i(\mathbf{x}) u_i \\ \frac{\delta u_{app}^{l}}{\delta x^{l}}(\mathbf{x}) &= \sum_i \frac{\delta^{l} N_i(\mathbf{x})}{\delta x^{l}} u_i \end{aligned} \tag{32}$$

All the shape functions N_i along with their diffuse derivatives are then expressed in a compact form as subsequent columns of a matrix resulting from product $\mathbf{A}^{-1}\mathbf{B}$

$$\begin{bmatrix} \mathbf{N}^{T} \\ \dfrac{\delta \mathbf{N}^{T}}{\delta x} \\ \dfrac{\delta \mathbf{N}^{T}}{\delta y} \\ \vdots \end{bmatrix} = \mathbf{A}^{-1}\mathbf{B} \tag{33}$$

The computational cost involved depends primarily on the inversion of matrix $\mathbf{A}$ which does not require to be performed explicitly. $\mathbf{LU}$ decomposition can be used instead (Belytchko *et al.,* 1996). The computation of the diffuse derivatives of the shape functions can be performed with no significant extra cost.

3.1. *Consistency conditions*

The consistency conditions, necessary for quadratic convergence of $u_{app}(x)=\sum_i N_i(x)u_i$ can be expressed as

$$\begin{aligned} \sum_i N_i &= 1 \\ \sum_i N_i\left(x_i - x\right) &= 0 \end{aligned} \tag{34}$$

The cubic convergence of u_{app} also requires

$$\sum_i N_i\left(x_i - x\right)^2 = 0 \tag{35}$$

The three consistency constraints (34), (35) can be presented compactly in a single matrix condition

$$\mathbf{PN} = \mathbf{e}^1 \tag{36}$$

with

$$\mathbf{e}^1 = \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix}, \quad \mathbf{e}^2 = \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}, \quad etc\ldots \tag{37}$$

and

$$\mathbf{P} = \begin{bmatrix} 1 & \cdots & 1 \\ x_1 - x & \cdots & x_n - x \\ \frac{1}{2}\left(x_1 - x\right)^2 & \cdots & \frac{1}{2}\left(x_n - x\right)^2 \end{bmatrix} \tag{38}$$

Linear convergence of $\dfrac{du_{app}}{dx}(x)=\sum_i N_i'(x)u_i$ requires

$$\sum_i N_i' = 0$$
$$\sum_i N_i'\left(x_i - x\right) = 1 \tag{39}$$

and for quadratic convergence we also have to satisfy

$$\sum_i N_i'\left(x_i - x\right)^2 = 0 \tag{40}$$

Thus, by analogy to (36) and using (37), (38) we obtain

$$\mathbf{PN}' = \mathbf{e}^2 \tag{41}$$

These properties, well known in finite elements under the name of consistency conditions, are necessary for the convergence of a variational formulation based on first derivatives, such as Finite Elements or Diffuse Elements.

In the collocation formulations based on second derivatives, the linear convergence of the terms $\frac{d^2 u_{app}}{dx^2}\left(x\right) = \sum_i N_i''\left(x\right)u_i$ appearing in the equilibrium equations, requires :

$$\sum_i N_i'' = 0$$
$$\sum_i N_i''\left(x_i - x\right) = 0 \tag{42}$$
$$\sum_i N_i'' \frac{\left(x_i - x\right)^2}{2} = 1$$

or

$$\mathbf{PN}'' = \mathbf{e}^3 \tag{43}$$

Conditions (36), (41) and (43) are automatically satisfied when deriving shape functions based on a complete quadratic polynomial basis by MLS or by GFDM. In the next section, we define a way to construct the shape functions directly from required properties.

3.2. *Consistency based approach for shape functions determination*

We now introduce an alternative technique of shape function construction explicitly based on the desired consistency conditions. This approach presents several advantages both from the computational and from the formal point of view. The obtained algorithm is efficient and some supplementary conditions, different from consistency, can also be introduced.

Let us consider an evaluation point x and a set of associated nodes $\{x_1, \ldots x_n\}$ "close to" the point x. We note

$$\mathbf{W} = \begin{bmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_n \end{bmatrix} \tag{44}$$

We introduce the objective function

$$J(\mathbf{N}) = \frac{1}{2}\mathbf{N}^T \mathbf{W}^{-1} \mathbf{N} \tag{45}$$

and we look for functions **N** which are solutions of $Min(J(\mathbf{N}))$ subjected to the first order consistency constraints (36).

The associated Lagrangian is

$$L(\mathbf{N},\lambda) = \frac{1}{2}\mathbf{N}^T \mathbf{W}^{-1} \mathbf{N} + \lambda^T \left(\mathbf{PN} - \mathbf{e}^1\right) \tag{46}$$

and the optimality conditions are

$$\begin{aligned} &\mathbf{PN} - \mathbf{e}^1 = 0 \\ &\mathbf{N}^T \mathbf{W}^{-1} + \lambda^T \mathbf{P} = 0 \end{aligned} \tag{47}$$

leading to the following linear system

$$\begin{bmatrix} \mathbf{W}^{-1} & \mathbf{P}^T \\ \mathbf{P} & 0 \end{bmatrix} \begin{Bmatrix} \mathbf{N} \\ \lambda \end{Bmatrix} = \begin{Bmatrix} 0 \\ \mathbf{e}^1 \end{Bmatrix} \tag{48}$$

The solution of (48) is given by $\mathbf{N} = -\mathbf{WP}^T\lambda$, so $\mathbf{P}\left(-\mathbf{WP}^T\lambda\right) = \mathbf{e}^1$ and

$$\lambda = -\left(\mathbf{PWP}^T\right)^{-1}\mathbf{e}^1 \tag{49}$$

and finally

$$\mathbf{N}^T = \mathbf{e}^{1^T}\left(\mathbf{PWP}^T\right)^{-1}\mathbf{PW} \tag{50}$$

where we recognize the matrices (20)

$$\begin{aligned} \mathbf{A} &= \mathbf{PWP}^T \\ \mathbf{B} &= \mathbf{PW} \end{aligned} \tag{51}$$

Functions **N** in the expression (50) correspond obviously to the Moving Least Squares shape functions.

Moreover, the first derivatives of the shape functions $\mathbf{N}'$ are solution of $Min\left(J\left(\mathbf{N}'\right)\right)$ under the constraint (41). The second derivative, $\mathbf{N}''$ is solution of $Min\left(J\left(\mathbf{N}''\right)\right)$ under the constraint (43), leading to the following expressions of the first and to the second diffuse derivatives of the shape functions

$$\begin{aligned} \mathbf{N}'^T &= \mathbf{e}^{2^T}\left(\mathbf{PWP}^T\right)^{-1}\mathbf{PW} \\ \mathbf{N}''^T &= \mathbf{e}^{3^T}\left(\mathbf{PWP}^T\right)^{-1}\mathbf{PW} \end{aligned} \tag{52}$$

We establish in this way that

$$\begin{aligned} N_i'(x) &\equiv \frac{\delta N_i}{\delta x}(x) \\ N_i''(x) &\equiv \frac{\delta^2 N_i}{\delta x^2}(x) \end{aligned} \tag{53}$$

Finally, we find again formula (33)

$$\begin{bmatrix} N_1(x) & \cdots & N_n(x) \\ \frac{\delta N_1}{\delta x}(x) & \cdots & \frac{\delta N_n}{\delta x}(x) \\ \frac{\delta^2 N_1}{\delta x^2} N_1(x) & \cdots & \frac{\delta^2 N_n}{\delta x^2}(x) \end{bmatrix} = \mathbf{A}(x)^{-1}\mathbf{B} \tag{54}$$

It is important to note that

– the subsequent derivatives of the functions N are obtained here as a result of the minimization of criterion (45) subjected to different consistency constraints (36), (41) and (43),

– we have shown, that these “consistency based” derivatives correspond to the diffuse derivatives. It implies that the diffuse derivatives are sufficient for the convergence of the solution of PDEs.

Other constraints that consistency may be applied. Therefore, this alternative presentation of the shape functions based on explicit constraints provides a powerful way to handle varied optimization constraints. This includes equality constraints such as incompressibility (Huerta *et al.,* 2002) and inequality constraint such as plastic admissibility (Breitkopf *et al.,* 2001). The $J(\mathbf{N})$ formulation was also used for the development of an efficient, well-conditioned algorithm for the shape functions evaluation, without an explicit inversion of the matrix **A** (Breitkopf *et al.,* 2000).

3.3. *Explicit form of the shape functions*

A further insight into the MLS methodology can be given by developing an explicit shape functions formulation. When the linear consistency constraints are required, the task leads to an inversion of a 2*2, 3*3 and 4*4 matrices respectively for the 1D, 2D and 3D cases.

3.3.1. *Shape functions and their derivatives in 1D*

The domains of influence of nodes are chosen in order to provide a 3-node connectivity, respectively x_1, x_2, x_3, at each evaluation point x. The weights are denoted as w_1, w_2, w_3.

Linear consistency constraints (36) are obtained with the matrix **P**

$$\mathbf{P} = \begin{bmatrix} 1 & 1 & 1 \\ x_1 - x & x_2 - x & x_3 - x \end{bmatrix} \tag{55}$$

Then we have the diagonal weight matrix

$$\mathbf{W} = \begin{bmatrix} w_1 & 0 & 0 \\ 0 & w_2 & 0 \\ 0 & 0 & w_3 \end{bmatrix} \tag{56}$$

and performing the necessary algebra we get the following explicit expressions of the three shape functions

$$
\begin{aligned}
N_1 &= \frac{w_1 w_2 (x - x_2)(x_1 - x_2) + w_1 w_3 (x - x_3)(x_1 - x_3)}{w_1 w_2 (x_1 - x_2)^2 + w_3 w_2 (x_3 - x_2)^2 + w_1 w_3 (x_1 - x_3)^2} \\
N_2 &= \frac{w_2 w_1 (x - x_1)(x_2 - x_1) + w_2 w_3 (x - x_3)(x_2 - x_3)}{w_1 w_2 (x_1 - x_2)^2 + w_3 w_2 (x_3 - x_2)^2 + w_1 w_3 (x_1 - x_3)^2} \\
N_3 &= \frac{w_3 w_1 (x - x_1)(x_3 - x_1) + w_3 w_2 (x - x_2)(x_3 - x_2)}{w_1 w_2 (x_1 - x_2)^2 + w_3 w_2 (x_3 - x_2)^2 + w_1 w_3 (x_1 - x_3)^2}
\end{aligned}
\tag{57}
$$

The recursive approach for an arbitrary number n of nodes in 1D gives a general expression of the shape functions in the form

$$
\begin{aligned}
N_i &= \frac{\sum\limits_{j \neq i} w_i w_j (x - x_j)(x_i - x_j)}{d} \\
d &= \sum_{i=1,n-1} \sum_{j=i+1,n} w_i w_j (x_i - x_j)^2
\end{aligned}
\tag{58}
$$

and the “full” x derivative is given by

$$
\begin{aligned}
\frac{dN_i}{dx} &= \frac{1}{d} \sum_{j \neq i} w_i w_j (x_i - x_j) + \frac{1}{d} \sum_{j \neq i} (w_{i,x} w_j + w_i w_{j,x})(x - x_j)(x_i - x_j) - N_i \\
d' &= \sum_{i=1,n-1} \sum_{j=i+1,n} (w_{i,x} w_j + w_i w_{j,x})(x_i - x_j)^2
\end{aligned}
\tag{59}
$$

where the first term corresponds to the diffuse derivative $w_{i,x} = 0$

$$
\frac{\delta N_i}{\delta x} = \frac{1}{d} \sum_{j \neq i} w_i w_j (x_i - x_j) \tag{60}
$$

One may also note that when $w_i = 0$ for $i > 2$ or when the number of nodes is equal to 2, the approximation degenerates as expected to the well known finite element shape functions:

$$
\begin{aligned}
N_1 &= \frac{x - x_2}{x_1 - x_2}, \quad \frac{dN_1}{dx} = \frac{\delta N_1}{\delta x} = \frac{1}{x_1 - x_2} \\
N_2 &= \frac{x_1 - x}{x_1 - x_2}, \quad \frac{dN_2}{dx} = \frac{\delta N_2}{\delta x} = \frac{-1}{x_1 - x_2}
\end{aligned}
\tag{61}
$$

3.3.2. *Shape functions and their derivatives in 2D*

When taking a 4 node neighborhood $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, \mathbf{x_4}$ at each evaluation point $\mathbf{x}$ in 2D with corresponding weights w_1, w_2, w_3, w_4 together with linear consistency constraints we obtain the first shape function

$$
\begin{aligned}
N_1 = \frac{1}{d}\big(\ & w_1 w_2 w_3 (-x_2 y + x_3 y + x y_2 - x_3 y_2 - x y_3 + x_2 y_3)(-x_2 y_1 + x_3 y_1 + x_1 y_2 - x_3 y_2 - x_1 y_3 \\
& w_1 w_2 w_4 (-x_2 y + x_4 y + x y_2 - x_4 y_2 - x y_4 + x_2 y_4)(-x_2 y_1 + x_4 y_1 + x_1 y_2 - x_4 y_2 - x_1 y_4 + x_2 \\
& w_1 w_3 w_4 (-x_3 y + x_4 y + x y_3 - x_4 y_3 - x y_4 + x_3 y_4)(-x_3 y_1 + x_4 y_1 + x_1 y_3 - x_4 y_3 - x_1 y_4 + x_3
\end{aligned}
\tag{62}
$$

The other shape functions are given by similar expressions, which can be written under the general form

$$
\begin{aligned}
N_i &= \frac{w_i}{d} \sum_{j \neq i} \sum_{k > j, k \neq i} w_j w_k \Theta_{2D}(\mathbf{x}, \mathbf{x_j}, \mathbf{x_k}) \Theta_{2D}(\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k}) \\
d &= \sum_{i=1,n-2} \sum_{j=i+1,n-1} \sum_{k=j,n} w_i w_j w_k \left(\Theta_{2D}(\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k})\right)^2
\end{aligned}
\tag{63}
$$

where

$$
\Theta_{2D}(\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k}) = -x_j y_i + x_k y_i + x_i y_j - x_k y_j - x_i y_k + x_j y
\tag{64}
$$

and the diffuse derivatives are

$$
\begin{aligned}
\frac{\delta N_i}{\delta x} &= \frac{w_i}{d} \sum_{j \neq i} \sum_{k > j, k \neq i} w_j w_k (y_j - y_k) \Theta_{2D}(\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k}) \\
\frac{\delta N_i}{\delta y} &= \frac{w_i}{d} \sum_{j \neq i} \sum_{k > j, k \neq i} w_j w_k (x_k - x_j) \Theta_{2D}(\mathbf{x_i}, \mathbf{x_j}, \mathbf{x_k})
\end{aligned}
\tag{65}
$$

Full derivatives can also be easily obtained by differentiating expression (63) or by formal differentiation of the computer code.

3.3.3. *Shape functions and their derivatives in 3D*

For 3D shape functions

$$
\begin{aligned}
N_i &= \frac{w_i}{d} \sum_{j \neq i} \sum_{j<k,k \neq i} \sum_{k<l,l \neq i} w_j w_k w_l \Theta_{3D}\left(\mathbf{x},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right)\Theta_{3D}\left(\mathbf{x_i},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right) \\
d &= \sum_{i=1,n-3} \sum_{i<j<n-2} \sum_{j<k<n-1} \sum_{k<l<n} w_i w_j w_k w_l \left(\Theta_{3D}\left(\mathbf{x_i},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right)\right)^2
\end{aligned}
\tag{66}
$$

with

$$
\begin{aligned}
\Theta_{3D}\left(\mathbf{x_i},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right) = z_i \Theta_{2D}\left(\mathbf{x}_j,\mathbf{x}_k,\mathbf{x}_l\right) + z_j \Theta_{2D}\left(\mathbf{x}_i,\mathbf{x}_k,\mathbf{x}_l\right) + \\
z_k \Theta_{2D}\left(\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_l\right) + z_l \Theta_{2D}\left(\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k\right)
\end{aligned}
\tag{67}
$$

The corresponding diffuse derivatives are

$$
\begin{aligned}
\frac{\delta N_i}{\delta x} &= \frac{w_i}{d} \sum_{j \neq i} \sum_{j<k,k \neq i} \sum_{k<l,l \neq i} \Theta_{2D}\left(\begin{Bmatrix} y_j \\ z_j \end{Bmatrix}, \begin{Bmatrix} y_k \\ z_k \end{Bmatrix}, \begin{Bmatrix} y_l \\ z_l \end{Bmatrix}\right)\Theta_{3D}\left(\mathbf{x_i},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right) \\
\frac{\delta N_i}{\delta y} &= \frac{w_i}{d} \sum_{j \neq i} \sum_{j<k,k \neq i} \sum_{k<l,l \neq i} \Theta_{2D}\left(\begin{Bmatrix} y_j \\ z_j \end{Bmatrix}, \begin{Bmatrix} y_k \\ z_k \end{Bmatrix}, \begin{Bmatrix} y_l \\ z_l \end{Bmatrix}\right)\Theta_{3D}\left(\mathbf{x_i},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right) \\
\frac{\delta N_i}{\delta z} &= \frac{w_i}{d} \sum_{j \neq i} \sum_{j<k,k \neq i} \sum_{k<l,l \neq i} \Theta_{2D}\left(\begin{Bmatrix} y_j \\ x_j \end{Bmatrix}, \begin{Bmatrix} y_k \\ x_k \end{Bmatrix}, \begin{Bmatrix} y_l \\ x_l \end{Bmatrix}\right)\Theta_{3D}\left(\mathbf{x_i},\mathbf{x_j},\mathbf{x_k},\mathbf{x_l}\right)
\end{aligned}
\tag{68}
$$

We observe that the practical use of these explicit forms of the shape functions is limited by the cost of their evaluation. The cost of the numerical algorithm is linear with respect to the number of nodes in the neighborhood of $\mathbf{x}$. However, the number of operations involved in a straightforward evaluation of the latter explicit formula for n nodes and k constraints is proportional to

$$
C_n^k = \frac{n!}{(n-k)!k!}
\tag{69}
$$

resulting in an $0(k)$ complexity. The formal expressions, which can still be applied for linear consistency constraints, become fairly complex when quadratic consistency are required. The cost of an explicit inversion of a 6*6 matrix in 2D or of a 10*10 matrix in a 3D becomes prohibitive.

The knowledge of the explicit forms is however important when considering issues such as continuity, the implementation of boundary conditions or integration strategies in variational formulations.

4. Interpolating MLS

In a general case, the MLS approximation does not interpolate data. In order to enforce the interpolating condition (4) at a node *i*, we separate the *i*th term in the optimality condition (18) for $J_x(\alpha)$

$$\lambda \mathbf{q}^T(x_i - x) + \sum_{j \neq i}\left(\mathbf{q}^T\left(x_j - x\right)\alpha - u_j\right) w_j\left(x_j, x\right)\mathbf{q}^T\left(x_j - x\right) = \mathbf{0} \qquad (70)$$

where

$$\lambda = w_i\left(\mathbf{q}^T(x_i - x)\,\alpha - u_i\right) \qquad (71)$$

In a matrix form, equations (70) and (71) become

$$\begin{bmatrix} \mathbf{A}_i & \mathbf{q}(x_i - x) \\ \mathbf{q}(x_i - x)^T & -\dfrac{1}{w_i} \end{bmatrix} \begin{Bmatrix} \alpha \\ \lambda \end{Bmatrix} = \begin{Bmatrix} \mathbf{b}_i \\ u_i \end{Bmatrix} \qquad (72)$$

with $\mathbf{A}_i = \sum_{j \neq i} \mathbf{q}\left(x_j - x\right)^T w_j \, \mathbf{q}\left(x_j - x\right)$ and $\mathbf{b}_i = \sum_{j \neq i} w_j u_j \, \mathbf{q}\left(x_j - x\right)$. When the weight function is singular at node *i* (5), then system (72) tends uniformly to

$$\begin{bmatrix} \mathbf{A}_i & \mathbf{e}^1 \\ \mathbf{e}^{1T} & 0 \end{bmatrix} \begin{Bmatrix} \alpha \\ \lambda \end{Bmatrix} = \begin{Bmatrix} \mathbf{b}_i \\ u_i \end{Bmatrix} \qquad (73)$$

which is precisely the optimality condition (18) for $J_x(\alpha)$ under the constraint (4) written as

$$u_{app}(x_i) = \mathbf{q}^T(x_i - x_i)\,\alpha = \alpha_0 = u_{ex}(x_i) \qquad (74)$$

where λ appears as the Lagrange multiplier.

It is interesting to examine the properties of the diffuse (6) and full (7) derivatives of the MLS interpolation at node x_i. When derivating (72), we obtain

$$\begin{bmatrix} \mathbf{A}_{i,x} & \mathbf{q}_{i,x} \\ \mathbf{q}_{i,x}^T & \dfrac{w_{i,x}}{w_i^2} \end{bmatrix} \begin{Bmatrix} \alpha \\ \lambda \end{Bmatrix} + \begin{bmatrix} \mathbf{A}_i & \mathbf{q}_i \\ \mathbf{q}_i^T & -\dfrac{1}{w_i} \end{bmatrix} \begin{Bmatrix} \alpha_{,x} \\ \lambda_{,x} \end{Bmatrix} = \begin{Bmatrix} \mathbf{b}_{i,x} \\ 0 \end{Bmatrix} \tag{75}$$

We observe that when $x \longrightarrow x_i$ then $\mathbf{q}(x_i - x) \longrightarrow \mathbf{e}^1$ and $\mathbf{q}_{,x}(x_i - x) \longrightarrow -\mathbf{e}^2$, thus $\mathbf{q}_i^T \dfrac{d\alpha}{dx} \longrightarrow \dfrac{d\alpha_1}{dx} = \dfrac{du}{dx}$ and $\mathbf{q}_{i,x}^T \alpha \longrightarrow -\alpha_2 = -\dfrac{\delta u}{\delta x}$. The last line of the matrix form (75) becomes

$$-\frac{\delta u}{\delta x} + \lambda \frac{w_{i,x}}{w_i^2} + \frac{du}{dx} - \frac{1}{w_i}\lambda_{,x} = 0 \tag{76}$$

For singular weights, when the derivative of the reference weight function w_i is bounded, the detailed analysis of the system (73) shows that the second and fourth terms in (76) disappear for $x \longrightarrow x_i$ (5). We have then

$$\lim_{x \to x_i} \frac{\delta u}{\delta x} = \frac{du}{dx} \tag{77}$$

This property means, that in the interpolating version of the MLS, the diffuse derivative is equal to the full derivative at the node. This feature is important in the meshfree methods based on the nodal integration schemes or in the nodal collocation methods.

4.1. *Singular weights*

The singular weights can be obtained by scaling the original weight functions in order to give a unit value at a node $w_i(x_i) = 1$ and then by applying the following substitution

$$[w(x_i, x)] \rightarrow \left[\frac{w(x_i, x)}{1 - w(x_i, x)} \right] \tag{78}$$

Interpolating shape functions are then obtained by minimization of any of the proposed criteria $J_x(\mathbf{a}),\ J_x(\alpha),\ J_x(\beta),\ J(\mathbf{N})$ with modified weights at any evaluation point $x \neq x_i$.

When an evaluation point is located at a node i, the modified weight function becomes singular. Nevertheless, the shape functions are known at the node without computation and are given by the interpolation condition:

$$N_i\left(x = x_j\right) = \delta_i^j \tag{79}$$

However, when the evaluation point is "close to" but not exactly "at" the node, then the use of singular weights becomes uncomfortable. Several methods may be applied but in practice it is sufficient to limit the growth of the $\tilde{w}_i$ by taking

$$\tilde{w}_i = \frac{w_i}{1 + \varepsilon - w_i} \tag{80}$$

where ε is a "small" value. It is then not necessary to distinguish between the two separate cases (78), (79) and the resulting approximation continuity is limited only by the continuity of the reference weight function.

Such "near singular weights" may result however in an ill conditioning of the algorithm.

4.2. *Interpolation with non-singular weights*

A different strategy based on a Shepard Interpolation can be used in order to obtain an interpolating MLS approximation. We notice first, that the substitution

$$[w] \rightarrow \lambda[w] \tag{81}$$

does not modify the solution of the optimality system for any $\lambda \neq 0$.

Let now introduce the scaled weights $\tilde{w}_j$

$$S(x) = \sum_i w(x_i, x), \quad \tilde{w}_j = \frac{1}{S(x)} w\left(x_j, x\right) \tag{82}$$

In the neighborhood of node $\mathbf{x}_j$ the modified weight function $\tilde{w}_j$ may be then written without singularity

$$\tilde{w}\left(\mathbf{x}_j, \mathbf{x}\right) = \frac{w_j}{w_j + \left(1 - w_j\right) \sum_{i \neq j} \frac{w_i}{1 - w_i}} \tag{83}$$

and $\tilde{w}\left(\mathbf{x}_j, \mathbf{x}_j\right) = 1.$

In the neighborhood of a node $\mathbf{x}_k,\ k \neq j$, expression (83) becomes singular as $w_k \rightarrow 1$. However, the weight function $\tilde{w}_j$ may by now computed from the expression (83) reformulated in the following way

$$\tilde{w}(\mathbf{x}_j, \mathbf{x}) = \frac{w_j(1 - w_k)}{(1 - w_j)\left(w_k + (1 - w_k)\sum_{i \neq k} \frac{w_i}{1 - w_i}\right)} \tag{84}$$

and we see that $\tilde{w}\left(\mathbf{x}_j, \mathbf{x}_k\right) = 0,\ k \neq j$.

These modified weights have the following properties

$$\tilde{w}(x_i, x) \in [0,1], \quad \sum_i \tilde{w}(x_i, x) = 1, \quad \tilde{w}\left(x_i, x_j\right) = \delta_{ij} \tag{85}$$

Figure 11 and Figure 12 illustrate the normalized interpolating weighting functions, derived from the reference weights (8) and (9) for a three-node configuration

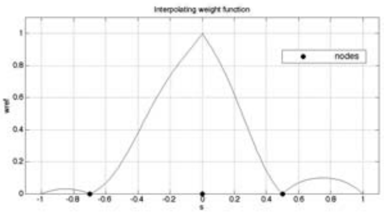


Figure 11
Nonsingular interpolating weights derived from the hat reference weight

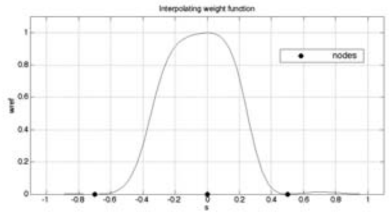


Figure 12
Nonsingular interpolating weights derived from the spline reference weight

The weights obtained by this procedure are not singular, take a unit value at their reference node and vanish at other nodes.

5. Diffuse elements

We have chosen to develop a meshfree method which may be implemented at a minimal cost within a standard finite element software framework. By analogy to the finite elements, we define a "diffuse element" which can be used in the assembly procedure for the global matrix. The diffuse element is identified by a list of nodes with non-zero contributions. From the geometrical point of view, a diffuse element corresponds to the intersections of the domains of influence of connected nodes. MLS shape functions are used instead of their finite element equivalents in order to obtain elementary matrices and vectors through a process of numerical integration. The issue of definition of domains of influence is tightly coupled to that of the precision of numerical integration (Dolbow and Belytschko, 1999). We extend here their approach in order to reduce the number of integration cells.

This approach is not the only way to construct a meshfree method and alternative approaches which do not use a background grid for numerical integration have been proposed. These methods include the "truly meshless" techniques (Lin and Atluri, 2000, De and Bathe, 2000) and nodal integration methods (Beissel and Belytschko, 1996, Bonet and Lok, 1999, Chen *et al.,* 2002).

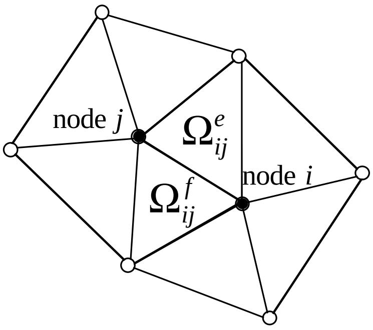
The implementation of the "diffuse element" approach is straightforward. However, several questions which do not arise in the finite element method, have still to be solved. These include primarily the issues of

– domain decomposition,
– numerical integration schemes,
– essential boundary conditions,
– patch test.

In the finite element method, the continuum Ω is divided into a finite number (say E) of open disjoint subregions – finite elements $\{\Omega_e, \quad e=1,2,\ldots,E\}$ such that $\mathbf{interior}(\Omega_e)\cap\mathbf{interior}(\Omega_f)=\varnothing$ for $e\neq f$. The finite element interpolation is defined locally in each element. The finite elements are also used as integration cells for numerical evaluation of the global integrals over the domain Ω, generally using the Gauss-Legendre scheme. The contribution *(i,j)* to the global linear system is then assembled from the set of elements sharing the nodes *i* and *j* (Figure 13).

In the diffuse element method, the *(i,j)* term of the global system is integrated over the intersection of domains of influence of nodes *i* and *j* (Figure 14). The evaluation of integrals over Ω is however less obvious as the domains of influence overlap in general in an irregular manner. Moreover, the domains do not respect the boundary. Thereafter, the integration scheme depends on the strategy in which we define the influence domains of the nodes (cf. section 2.2.3.).

R(x) strategy guarantees the existence of the MLS approximation at each evaluation point belonging to Ω. Figure 15 illustrates the domains of influence for a regular grid of nodes with *k*=4. The individual integration cells are then given by the Voronoï diagram of 4-th order polygons defined by the sets of points sharing the same list of 4 closest neighboring nodes. In the following figures, the nodes are denoted by thick dots.



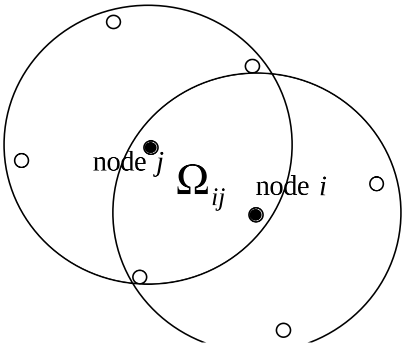


Figure 13
Finite elements

Figure 14
Diffuse elements, r_i strategy

Example of integration cells for the term (i,j) of the global system

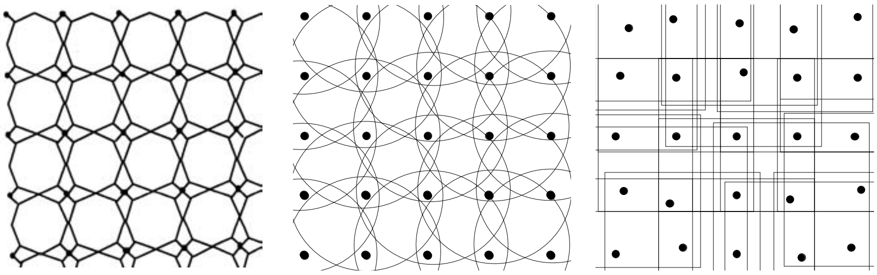


Figure 15
$R(x)$ strategy

Figure 16
r_i strategy in L^2 norm

Figure 17
r_i strategy and, L^{∞} norm

Shapes of integration cells using various definitions of the radius of influence on a regular (Figure 15, Figure 16) and on a randomly perturbed 2D grid (Figure 17)

We can notice, that the shape of the integration domains is complex even if the grid of nodes is regular. Moreover, the generation of high order Voronoï diagrams is much more costly then the generation of a finite element mesh which only requires a first order Voronoï diagram.

The shapes of domains of influence of nodes is greatly simplified when using r_i strategy. However, when the L^2 norm is used for computing distances, the domains of influence are circular and the cost of computing individual contributions over all the intersections may still be prohibitive (Figure 16). In Figure 17 we show the domains of influence obtained using the L^{∞} norm. An integration domain is given in this case by intersection of rectangular domains of influence of the nodes. For this reason, in the actual work we choose the L^{∞} norm.

Another difficulty when using r_i strategy consists in satisfying the two contradictory requirements:

– the domains of influence have to be big enough in order to guarantee the existence of the approximation at each point of Ω ;

– the domains of influence should be as small as possible in order to limit the bandwidth of the resulting global system.

The second requirement governs also the accuracy of the approximation. It may be shown that these conditions are satisfied with the procedure illustrated in Figure 18. First, we build the first order Voronoï diagram and then, for each node we create the domain of influence as a rectangular envelope of Voronoï cells surrounding the cell to which the node belongs. This technique is sufficient for the linear basis $\mathbf{p} = \langle 1 \quad x \quad y \rangle$ as it connects at least 4 neighbors at each evaluation point.

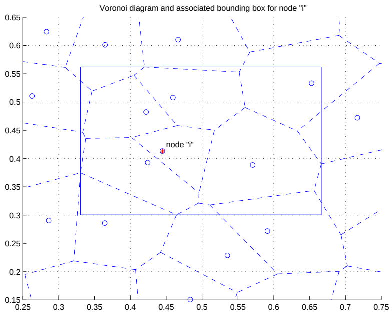


Figure 18
Domain of influence defined as an envelope of surrounding Voronoï cells

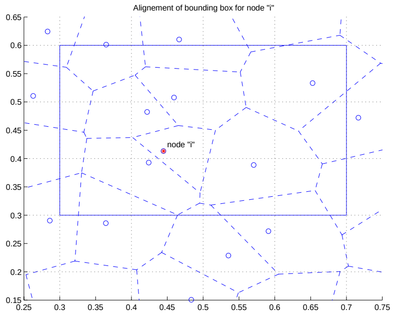


Figure 19
Domain of influence expanded to fit the tessellation grid

We note that node “i” is not centered in the resulting domain of influence. We handle this situation in the following way. First, we introduce a C^1 continuous mappings $\xi(x)$ and $\eta(y)$ from asymmetric domain in 1D to a symmetric one along each axis (Figure 20).

Then, we define the 2D weight function as a tensor product of 1D weight functions $w(x,y) = w_{ref}(\xi) w_{ref}(\eta)$. Figure 21, Figure 22 and Figure 23 show a typical interpolating shape function and its derivatives over an asymmetric domain.

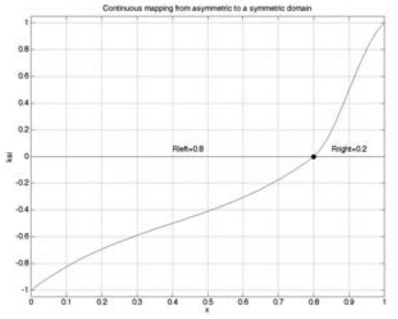


Figure 20
Mapping used for nodes excentered in their domain of influence in 1D

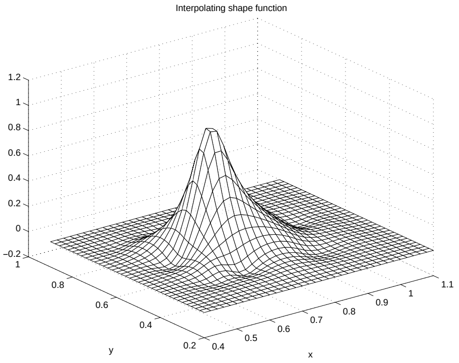


Figure 21
Interpolating shape function over an asymmetric domain

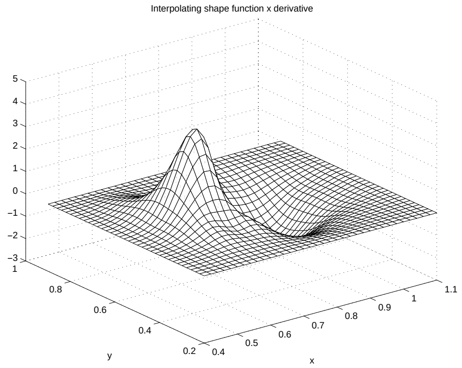


Figure 22
Interpolating shape function x derivative over an asymmetric domain

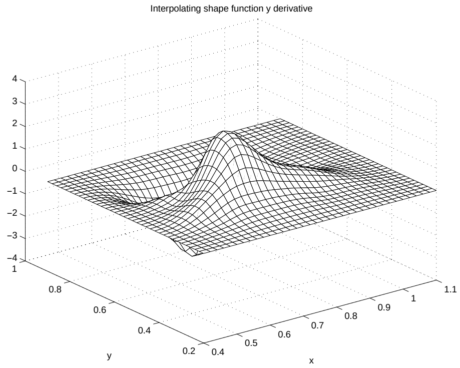


Figure 23
Interpolating shape function y derivative over an asymmetric domain

In order to reduce the number of integration subregions, we define a tessellation of the domain. The tessellation grid may be regular or adjusted locally to the average size of the Voronoï cells. The domains of influence corresponding to Figure 18 are further expanded to match this tessellation grid. As each nodal domain of influence consists of a set of tessels, the intersection between two domains is also given by a set of tessels. The list of connected nodes is constant over a tessel. The Figure 19 illustrates the further simplification of the form of the integration cells.

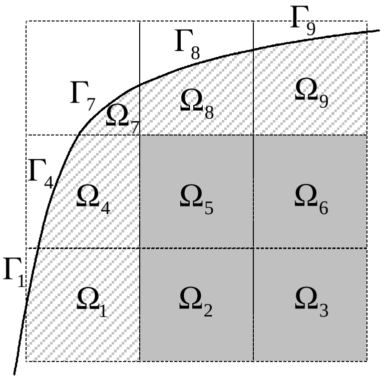


Figure 24
Integration cells: internal (solid) and boundary (hatched)

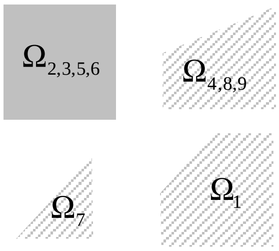


Figure 25
Typical integration cells in 2D

The common choice for numerical integration is the usual Gauss Legendre scheme. The integration points for rectangular domains are directly obtained by a linear mapping from the reference domain. In Figure 24, the solid tessels correspond to internal integration cells and the shaded ones belong to the boundary tessels. The cells intersected by the boundary Γ have more complex forms and must be treated separately. For these cases, an isoparametric mapping can be used in a similar way as in a finite element context. Another solution consists in subdividing the boundary tessels into simpler shapes. The Figure 25 provides the typical integration cells in 2D.

6. Integration scheme

6.1. *Patch test*

In the “patch test”, a linear elasticity problem is solved over a domain with the displacements prescribed along all outside boundaries by a linear function of the coordinates. The resulting strains and stresses in this case are constant. When a numerical method of solving the partial differential equations verifies this condition, we say that it satisfies the “patch test”. This approach to verify the numerical formulation and the code itself is standard in the finite element method. In the following section, we use our tessellation scheme along with the DEM and EFG formulations. Details of the reference test problem can be found in (Lu *et al.*, 1994). In the present work, the boundary conditions are enforced using the modified variational principle (Mukherjee and Mukherjee, 1997).

We define a domain Ω with boundary $\Gamma = \Gamma_N \cup \Gamma_D$. We study the Laplace’s equation

$$
\begin{aligned}
&-\Delta u = f \text{ in } \Omega \\
&u=\bar{u} \text{ on } \Gamma_D \text{ and } \frac{\partial u}{\partial n}=\bar{q} \text{ on } \Gamma_N
\end{aligned}
\tag{86}
$$

where n is the outer normal defined on the boundary.

We use the extended variational formulation analogous to that given by (Lu *et al.*, 1994)

$$
\begin{gathered}
\int_{\Omega} \nabla u \nabla v d\Omega - \int_{\Gamma_D} \frac{\partial(uv)}{\partial n} d\Gamma = \int_{\Omega} f v d\Omega + \int_{\Gamma N} \bar{q} v d\Gamma - \int_{\Gamma_D} \bar{u} \frac{\partial v}{\partial n} d\Gamma \\
\forall v \in \left\{ v \in H^1(\Omega) / \Delta v \in L^2(\Omega) \right\}
\end{gathered}
\tag{87}
$$

The associated discrete system

$$
\mathbf{Ku}=\mathbf{F} \tag{88}
$$

is obtained in the usual way with

$$
\begin{aligned}
K_{ij} &= \int_{\Omega} \nabla N_i \nabla N_j d\Omega - \int_{\Gamma_D} \left(N_i \frac{\partial N_j}{\partial n} + N_j \frac{\partial N_i}{\partial n} \right) d\Gamma \\
F_i &= \int_{\Omega} f(x) N_i(x) d\Omega + \int_{\Gamma_N} \bar{q} N_i d\Gamma - \int_{\Gamma_D} \bar{u} \frac{\partial N_i}{\partial n} d\Gamma
\end{aligned}
\tag{89}
$$

where N_i are the usual MLS shape functions.

The boundary conditions on both Neumann Γ_N and Dirichlet Γ_D parts of the boundary are associated to a linear field u_{ex}. In order to verify the patch test, we have to check whether the numerical solution procedure restitutes u_{ex} exactly inside Ω.

Figure 26 and Figure 27 illustrate the convergence of the patch test in 1D. Three nodes are used and we test the precision of the patch test versus the precision of the numerical integration. The orders of standard Gauss integration vary from 1 to 12. Different number of integration cells, with interpolating version of MLS shape functions, alternatively with full and with diffuse derivative are used.

The following conclusions may be stated:

- in both cases the patch test is not *a priori* satisfied;

– when the “full” derivative is used, the patch test converges independently of the tessellation density, when refining the numerical integration; the error is however relatively important even for high number of Gauss points; further refinement of the integration scheme leads to numerical errors;

– the diffuse derivative performs poorly, independently of the tessellation density and of the number of Gauss points.

The first two points can be easily explained when considering the explicit expressions of the shape functions established in paragraph 3.3. As opposed to the (simplest) case of finite elements, MLS shape functions do not have a polynomial form. In fact, when the weights *w* are given by spline functions, MLS shape functions are rational and their order is both defined by the number of connected nodes and the order of the polynomial expression *w(*x,x*ᵢ*). Therefore, the integration is not well performed by the classical Gauss-Legendre scheme.

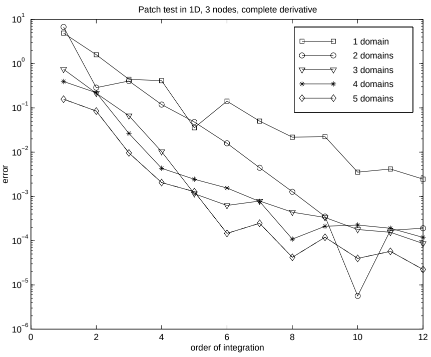


Figure 26
Full derivative

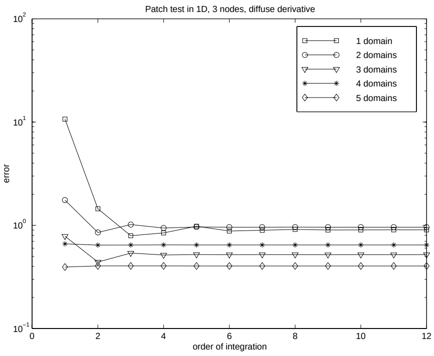
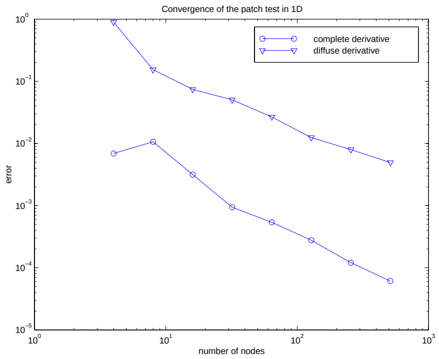


Figure 27
Diffuse derivative

Precision of 3-node patch test in 1D for varying orders of standard Gauss integration and different numbers of integration cells

The convergence of the patch test can be then explained by the convergence of the numerical integration itself. The failure of the diffuse derivative in a variational formulation cannot be easily explained. This behavior is opposite to that observed for strong formulations based on finite differences on irregular grids (Liszka and Orkisz, 1980) where the diffuse derivation performs well.

In Figure 28, we analyze the convergence of the patch test with different numbers of nodes and with full and diffuse derivatives. We observe that both formulations converge at approximately the same rate for high numbers of nodes. However, the obtained precision is not acceptable when considering the computational cost. The precision for the diffuse derivative is still significantly worse.



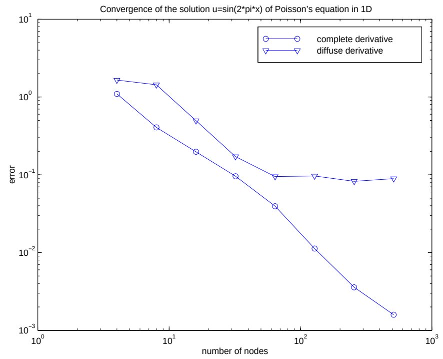


Figure 28
Patch test

Figure 29
Solution of Poisson equation

Convergence in 1D with varying number of nodes, comparison of the full and diffuse derivatives

Figure 29 gives the results for the same domain as that used for the patch test. However, the equation solved and the boundary conditions are chosen in order to give the exact solution *u=sin(2*pi*x)*. The full derivative version performs reasonably well, while the diffuse derivative diverges.

Though, the proposed discretization scheme satisfies the patch test at convergence. Both the number of numerical integration points and number of nodes increase. The full derivative must be used. Similar results are obtained when solving an arbitrary problem. The drawback of the method is that the patch test is not verified exactly for low number of nodes and for low number of integration points. This can be explained by the fact that Gauss Legendre method poorly integrates the shape functions. Therefore, we propose in the following section a custom quadrature scheme for MLS shape functions in order to ensure the properties needed for exact verification of the patch test.

6.2. *Integration constraint*

We focus now on the necessary condition that the formulation has to fulfill in order to satisfy the patch test in the three following cases:

$$\begin{aligned} u_{ex}(\mathbf{x}) &= 1 \\ u_{ex}(\mathbf{x}) &= x \\ u_{ex}(\mathbf{x}) &= y \end{aligned} \tag{90}$$

We take

$$u_{ap}(x) = u_1 N_1(x) + \cdots + u_n N_n(x) \tag{91}$$

and we suppose that the global system (88) has a unique solution. In this case we should obtain $u_i = u_{ex}(x_i)$, due to the linear consistency properties of the MLS approximation. The interpolation property is not necessary in this case. Substituting (91) into (87), we get

$$\int_{\Omega} \nabla N_i \left(\sum_j u_j \nabla N_j \right) d\Omega - \int_{\Gamma_D} \left\{ N_i \left(\sum_j u_j \frac{\partial N_j}{\partial n} \right) + \left(\sum_j u_j N_j \right) \frac{\partial N_i}{\partial n} \right\} d\Gamma = F_i \tag{92}$$

with

$$F_i = \int_{\Gamma_N} N_i \left(\sum_j u_j \frac{\partial N_j}{\partial n} \right) d\Gamma - \int_{\Gamma_D} \left(\sum_j u_j N_j \right) \frac{\partial N_i}{\partial n} d\Gamma \tag{93}$$

When substituting $u = (1 \quad \ldots \quad 1)^T$ and taking into account the consistency properties of the N_i functions (guaranteed by construction) the first of the three conditions (90) is automatically verified.

For a linear field $u_{ex}(x) = x$ we obtain on the RHS of (87)

$$F_i = \int_{\Gamma_N} n_x N_i d\Gamma - \int_{\Gamma_D} x \frac{\partial N_i}{\partial n} d\Gamma \tag{94}$$

and substituting the further consistency conditions

$$\sum_i x_i N_i = x, \quad \sum_i x_i \frac{\partial N}{\partial x} = 1, \quad \sum_i x_i \frac{\partial N}{\partial y} = 0 \tag{95}$$

The condition (94) reduces to

$$\int_{\Omega} \frac{\partial N_i}{\partial x} d\Omega = \int_{\Gamma} n_x N_i d\Gamma \tag{96}$$

that must be satisfied for a linear field $u_{ex}(x) = x$. An analogous analysis for $u_{ex}(\mathbf{x}) = y$ gives

$$\int_{\Omega} \frac{\partial N_i}{\partial y} d\Omega = \int_{\Gamma} n_y N_i d\Gamma \tag{97}$$

and subsequently, for an arbitrary linear field $u_{ex}(\mathbf{x}) = ax + by + c$

$$\int_{\Omega} \nabla \mathrm{N_i d\Omega} = \int_{\Gamma} \mathrm{N_i \mathbf{n} d\Gamma} \tag{98}$$

has to hold. This can be seen as the expression of the Green-Riemann theorem for the discrete integration. In the numerical program, the integrals are calculated in an approximate way, the precision of the patch test depends on the choice of the numerical integration method. We note also, that the interpolation property of the MLS approximation is not necessary for the patch test.

6.3. *Custom integration scheme*

As shown in the explicit expressions (paragraph 3.3), MLS shape functions are not polynomial. Thus, by opposition to the finite element context, the standard Gauss Legendre integration scheme is not suited for the meshfree methods. In the following, we take into account that the domain is splitted into a set of integration subdomains called tiles or “diffuse elements” (paragraph 5). We propose below a specific integration scheme denoted by $\tilde{\int}_{\Omega}$ which satisfies the global conditions

$$\begin{aligned} &\tilde{\int_{\Omega}} \frac{\partial}{\partial x} \{N_i(\mathbf{x})\} d\Omega = \int_{\Gamma} N_i(\mathbf{x}(s)) dy(s) \\ &\tilde{\int_{\Omega}} \frac{\partial}{\partial y} \{N_i(\mathbf{x})\} d\Omega = -\int_{\Gamma} N_i(\mathbf{x}(s)) dx(s) \end{aligned} \tag{99}$$

on the “tile by tile” basis. The integration cells Ω_e are chosen in such a way that $\textbf{interior}(\Omega_i) \cap \textbf{interior}(\Omega_j) = \varnothing,\ i \neq j$ and $\bigcup_e \Omega_e = \Omega$. In the numerical procedure, the boundary integrals $\int_{\Gamma_e} (.) d\Omega$ are computed using a standard Gauss integration. The specific integrals over Ω_e are noted by $\int \ \int_{\Omega_e} d\Omega \tilde{\int} (\cdot)$ and are defined so that

$$\tilde{\int\limits_{\Omega_e}} \frac{\partial N_i}{\partial x} d\Omega = \int\limits_{\Gamma_e} N_i dy, \ \tilde{\int\limits_{\Omega_e}} \frac{\partial N_i}{\partial y} d\Omega = -\int\limits_{\Gamma_e} N_i dx \tag{100}$$

is satisfied over each individual subdomain of integration and for any node i connected to Ω_e. In the summation procedure, the boundary integrals between individual subdomains mutually cancel. Therefore, only non-zero contribution comes from the external boundary and the global condition (99) is satisfied.

The discrete LHS integrals are written as

$$\begin{aligned} \sum_g \omega_g \frac{\partial N_i}{\partial x}(\mathbf{x}_g) &= \int\limits_{\Gamma_e} N_i(\mathbf{x}) dy \\ \sum_g \omega_g \frac{\partial N_i}{\partial y}(\mathbf{x}_g) &= -\int\limits_{\Gamma_e} N_i(\mathbf{x}) dx \end{aligned} \tag{101}$$

where $\mathbf{x}_g$ are the usual Gauss-Legendre integration points and ω_g are the custom integration weights. The above expressions can be presented in a matrix form

$$\mathbf{D}\omega = \mathbf{d} \tag{102}$$

Matrix $\mathbf{D}$ and the column vector $\mathbf{d}$ are obtained directly from (25). The integration scheme is extended in order to integrate polynomials. For this reason, we choose a set of monomials $\mathbf{m} = \{1 \quad x \quad y \quad x^2 \quad xy \quad y^2 \quad \dots\}$ which have to be exactly integrated

$$\sum_g \omega_g m(\mathbf{x}_g) = \int\limits_{\Omega_e} m(\mathbf{x}) d\Omega \tag{103}$$

which may be written in the matrix form

$$\mathbf{G}\omega = \mathbf{g} \tag{104}$$

We note, that the solution of the coupled system (102), (104) is not straightforward due to a poor conditioning. Our experience shows that a practical way consists in minimizing $\|\mathbf{G}\omega - \mathbf{g}\|^2$ under the constraint (102). If the dimension of $\mathbf{m}$ is properly chosen, then the minimum is zero. In this case, the complete system ((102),(104)) is satisfied.

6.4. *Numerical verification of the patch test*

In this section, we present patch test results for the example problems (Figure 30, Figure 31 and Figure 32) given in (Lu *et al.*, 1994) and for an arbitrary domain.

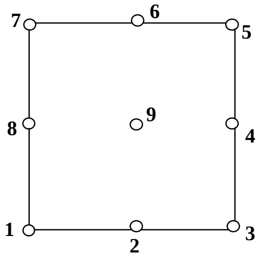


Figure 30
Regular grid of nodes

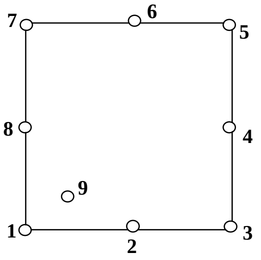


Figure 31
Irregular node 9

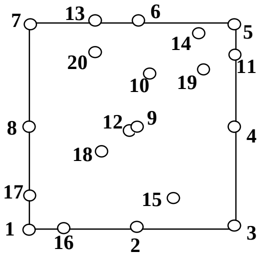


Figure 32
Irregular grid

The performances of classical and custom integration schemes are compared using the above problems alternatively with interpolating and non-interpolating shape functions and with the full and diffuse derivatives. The results obtained are given in the following two tables:

Gauss weights		Full derivative	Diffuse derivative
MLS approximation	Figure 30	7.7716e-16	8.8818e-16
	Figure 31[1]	5.2826e-3	4.3726e-1
	Figure 31[2]	1.5014e-4	1.0933e-1
	Figure 32	5.8584e-2	6.7871e+1
	arbitrary patch	4.0122e-6	3.8446e-3
MLS interpolation	Figure 30	7.7716e-16	0
	Figure 31[1]	6.3801e-4	1.2775e-1
	Figure 31[2]	4.7010e-4	3.0384e-3
	Figure 32	2.2963e-3	3.4729e-1
	arbitrary patch	1.0930e-5	1.5457e-3

Table 1. *Patch test results using L_2 norm with standard Gauss-Legendre integration; In Figure 31[1] node 9 is located at (0.3, 04), in Figure 31[2] node 9 is located at (0.9, 0.9)*

Modified weights		Full derivative	Diffuse derivative
MLS approximation	Figure 30	5.5511e-16	5.0653e-11
	Figure 31^1	2.5434e-14	5.9145e-14
	Figure 31^2	2.1663e-16	3.8993e-15
	Figure 32	1.9350e-17	4.3859e-11
	arbitrary patch	9.6859e-14	2.9635e-15
MLS interpolation	Figure 30	3.9968e-15	4.6629e-14
	Figure 31^1	2.0186e-16	2.3012e-14
	Figure 31^2	4.9825e-15	1.8413e-14
	Figure 32	6.4393e-15	6.8530e-10
	arbitrary patch	3.0192e-14	3.4437e-13

Table 2. *Patch test results using L_2 norm with the custom integration scheme. In Figure 31^1 node 9 is located at (0.3, 04), in Figure 31^2 node 9 is located at (0.9, 0.9)*

As shown in Table 1 and Table 2, in the case of rectangular domain with regular node pattern, the patch test is always satisfied (within the limits of numerical precision); This is true independently of the choice of other parameters and can be easily explained by symmetry reasons. This property is no longer valid in the case of non rectangular domains or for an irregular distribution of nodes.

The following conclusions can be drawn:

– the modified weights pass the patch test “exactly”, for both full and diffuse derivative and for both approximating and interpolating MLS; these results are more accurate than (Dolbow and Belytschko, 1999) obtained in the scope of EFG;

– for the standard Gauss integration, the full derivative is mandatory and the precision of the patch test depends on the density of the Gauss points; the use of the interpolating shape functions improves slightly the results in this case.

However, the computational cost of modified weights is high, as the system (102), (103) has to be solved on each integration domain.

7. Closing remarks

Throughout this paper, we have developed the basis of Moving Least Squares meshfree approximation and interpolation methods. As an illustration, we have described the Diffuse Element Method for solving PDEs. The meshfree methods do not require an explicit mesh. Only a set of data points and a description of the boundary surfaces are needed. At each evaluation point, a list of nearest nodes is used to approximate the value at that point. The finite element shape functions are replaced by their Moving Least Squares equivalents. This approximation procedure is used to obtain a global system of linear equations. The goal is to achieve a better control of the continuity of the solution, an easier handling of evolving boundaries, the possibilities of adding or removing nodes and the treatment of distorted domains without remeshing.

Despite the undeniable success in many applications, the meshfree methods are still in an early phase of development. The practical implementation of such methods encounters several problems which do not appear in the finite element method. A number of alternative methods in order to take into account the essential boundary conditions reveal advantages and shortcomings. The numerical integration issues must also be clarified. The standard finite element patch test is not *a priori* satisfied in the case of meshfree methods. The clouds of points used to discretize the domain require an additional treatment in order to establish nodal connectivity. One of the most important theoretical points is the discrete ellipticity of the standard variational formulation. In the extended variational formulation, discrete inf-sup condition has to be established the in order to prove convergence. We also have to explain why the patch test integration constraint is crucial in the improvement of numerical results.

The above questions are still opened and no definite answers can be given. That is why the topic of meshfree methods gives numerous and exciting opportunities for new ideas and contributions.

8. References

Babuska I., Banerjee U., Osborn J.E., Meshless and generalized finite element methods: a survey of some major results, TICAM Report 02-03, University of Texas at Austin, January 2002.

Babuska I., Melenk J. M., "The Partition of Unity Method", *International Journal for Numerical Methods in Engineering,* 40, 727-758, 1997.

Barnhill, R.E., *Representation and approximation of surfaces in Mathematical Software,* ed. J.R. Rice, Academic Press, New York, 69-120, 1977.

Beissel S., Belytschko T., "Nodal Integration of the Element-Free Galerkin Method", *Computer Methods in Applied Mechanics and Engineering*, 139, 49-74, 1996.

Belytschko T., Krongauz Y., Organ D., Fleming M., and Krysl P., "Meshless Methods: An Overview and Recent Developments", *Computer Methods in Applied Mechanics and Engineering*, 139, 3-47, 1996.

Belytschko T., Lu Y.Y., Gu L., "Element-free Galerkin Methods", *International Journal for Numerical Methods in Engineering,* 37, 229-256, 1994.

Belytschko T., Gu L., Lu. Y. Y., "Fracture and crack growth by element-free Galerkin methods. Modelling and Simulation", *Material Science and Engineering,* 2, 519-534, 1994(a).

Bonet J., Lok T.-S. L., “Variational and momentum preservation aspects of smooth particle hydrodynamics formulation”, *Computer Methods in Applied Mechanics and Engineering,* 180, 97-115, 1999.

Breitkopf P., Rassineux A., Touzot G., Villon P., “Explicit form and efficient computation of MLS shape functions and their derivatives”*, International Journal for Numerical Methods in Engineering,* 48, 451-456, 2000.

Breitkopf P., Rassineux A., Villon P., Saannouni K., Cherouat H., “Meshfree operators for consistent field transfer in large deformation plasticity”, *ECCOMAS-ECCM-2001*, Cracow, Poland, 26-29 June 2001.

Breitkopf P., Touzot G., Villon P., “Double Grid Diffuse Collocation Method”, *Computational Mechanics,* 25, No 2/3, 199-206, 2000.

Chen J-S, Han W., You Y., Meng X., “A Reproducing Kernel Method with Nodal Interpolation Property”, *International Journal for Numerical Methods in Engineering,* in press, 2002.

Chen J. S., Wu C. T., Yoon S., You Y., “Nonlinear Version of Stabilized Conforming Nodal Integration for Galerkin Meshfree Methods”, *International Journal for Numerical Methods in Engineering*, 53, 2587-2615, 2002.

Cleveland, W.S., “Robust Locally Weighted Regression and Smoothing Scatterplots”, *Journal of the American Statistical Association,* December, Vol, 74, No. 368, 829-836, 1979.

De S., Bathe K.J., “The Method of Finite Spheres”*, Computational Mechanics*, 25, 329-345, 2000.

Dolbow J., Belytschko T., “Numerical integration of the Galerkin weak form in meshfree methods”, *Computational Mechanics,* 23(3), 219-230, 1999.

Gordon William J., Wixom James A., “Shepard's method of metric interpolation to bivariate and multivariate interpolation”, *Math. Comp*. 32(141), 253-264, 1978.

Huerta A., Vidal Y., Villon P., “Locking in the Incompressible Limit: Pseudo-Divergence-Free Element Free Galerkin”*, Proceedings of the Fifth World Congress on Computational Mechanics (WCCM V),* July 7-12, 2002, Vienna, Austria.

Krige, D.G. Two-dimensional weighted moving average trend surfaces for ore evaluation. *Journal of the South African Institute of Mining & Metallurgy*, 67, 13-79, 1966.

Lancaster P., Salkauskas K., *Curve and Surface Fitting: an Introduction,* Academic Press, London, Orlando, 1986.

Lancaster P., Salkauskas K., “Surfaces generated by moving least squares methods”, *Math. Comp*. 37, 141-158, 1981.

Lin H., Atluri S.N., “Meshless Local Petrov-Galerkin (MLPG) Method for Convection - Diffusion Problems”*, Computer Modeling in Engineerng & Sciences*, 1(2), 45-60, 2000.

Liszka T., Orkisz J., “The Finite Difference Method at Arbitrary Irregular Grids and its Application in Applied Mechanics”, *Computers and Structures,* 11, 83-95, 1980.

Liu W. K., Chen Y., Jun S., Chen J. S., Belytschko T., Pan C., Uras R. A. Chang C. T., “Overview and Applications of the Reproducing Kernel Particle Methods”, *Archives of Computational Methods in Engineering: State of the art reviews*, 3, 3-80, 1996.

Lu Y.Y., Belytschko T., Gu L., “A New Implementation of the Element Free Galerkin Method”, *Computer Methods in Applied Mechanics and Engineering*, 113, 397-414, 1994.

Lucy L. B., “A numerical approach to the testing of the fission hypothesis”, *Astronomical Journal* 82, 1013-1024, 1977.

Mac Lain DH, “Drawing Contours with Arbitrary Data Points”, *The Computer Journal*, 17(4), 318-324, 1974.

Matheron G., “Principles of Geostatistics”, *Economic Geology*, 58, 1246- 1266, 1963.

Mukherjee Y.X., Mukherjee S., “On boundary conditions in Element-Free Galerkin Method”, *Computational Mechanics*, 11, 1997, 264-270.

Nayroles B., Touzot G., Villon P., “Generalizing the Finite Element Method: Diffuse Approximation and Diffuse Elements”, *Computational Mechanics*, 10, 307-318, 1992.

Oñate E., Idelsohn S.R., “A mesh-free finite point method for advective-diffusive transport and fluid flow problems”, *Computational Mechanics*, 21, 283-292, 1998.

Savignat J-M, Approximation diffuse Hermite et ses applications, Thèse de Doctorat, Ecole des Mines de Paris, October 2000.

Shepard D., “A two-dimensional interpolation function for irregularly spaced data”, *Proc. 23rd National Conference ACM*, 517-524, 1968.

Sukumar N., Moës N., Moran B. Belytschko T., “Extended Finite Element Method for Three-Dimensional Crack Modeling”, *International Journal for Numerical Methods in Engineering*, 48(11), 1549-1570, 2000.

Sulsky D., Schreyer HL, The Particle-In-Cell Method as a Natural Impact Algorithm, Sandia National Laboratories, Contract No. AC-1801, 1993.

Syczewski M., Tribillo R., “Singularities of Sets Used in the Mesh Method”, *Computers and Structures*, 14(5-6), 509-511, 1981.

Villon P, Contribution à l’Optimisation, Thèse de Docteur d’Etat, Université de Technologie de Compiègne, France, 1991.

Wyatt M.J., Davies G., Snell C., “A New Difference Based Finite Element Method”, *Instn. Engineers*, 59(2), 395-409, 1975.

Zhang X., Liu X-H., Song K-Z., Lu M-W., Least-squares collocation meshless method, *International Journal for Numerical Methods in Engineering*, 51, 1089-1100, 2001.