Copyright © 2008 Tech Science Press CMES, vol.36, no.1, pp.43-63, 2008

Accurate MLPG Solution of 3D Potential Problems

Giorgio Pini[1], **Annamaria Mazzia**[1] **and Flavio Sartoretto**[2]

Abstract: Meshless methods have been explored in many 2D problems and they have been shown to be as accurate as Finite Element Methods (FEM). Compared to the extensive literature on 2D applications, papers on solving 3D problems by meshless methods are surprisingly few. Indeed, a main drawback of these methods is the requirement for accurate cubature rules. This paper focuses on the so called Meshless Local Petrov Galerkin (MLPG) methods. We show that accurate solutions of 3D potential problems can be attained, provided suitable cubature rules are identified, sparse data structures are efficiently stored, and strategies are devised in order to speed up the computation flow, by avoiding unnecessary integral evaluations. The ensuing MLPG linear systems result to be well conditioned, positive definite ones. Their conditioning does not increase much when the mesh size decreases. We show that cubature errors can lower MLPG convergence speed.

Keyword: Meshless Methods, Poisson Problem, Moving Least Squares, Radial Basis Functions

1 Introduction

Among the methods for solving Partial Differential Equations (PDE), nowadays *meshless methods* attract increasing interest. From an abstract point of view, a meshless method is a Petrov–Galerkin method where, unlike for FEM methods, non–polynomial trial and test basis functions with arbitrarily, i.e. not mesh–guided, overlapping supports, are engaged [Atluri (2004); Babuska, Banerjee, and Osborn (2004); Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)].

When cataloging the huge number of meshless methods proposed, a first dichotomy appears between *true* meshless methods and the *other* ones. The latter methods in principle do not exploit any mesh for discretizing the problem domain, but either for the interpolation of the solution variables or for the integration of the weak forms,

[1] Dipartimento di Metodi e Modelli Matematici per le Scienze Applicate, Università di Padova, Italy, {pini,mazzia}@dmsa.unipd.it

[2] Dipartimento di Informatica Università di Venezia, Italy, sartoret@dsi.unive.it

they exploit local and global meshes, hence requiring time consuming geometric structures to be managed. As a byproduct, *true* meshless methods are more apt to implement adaptivity.

Most research in meshless methods was restricted to solving 2D problems. It is more challenging to solve 3D problems, mainly due to the difficulty in accurately evaluating integrals. Representative 3D works, mainly exploring many distinguished peculiarities of mechanical problems, are [Belytschko, Krysl, and Krongauz (1997); Xiong, Rodrigues, and Martins (2003); Schembri, Crane, and Reddy (2004); Han and Atluri (2004); Li, Shen, Han, and Atluri (2003); Atluri, Liu, and Han (2006a,b)]. ,

The truly meshless local Petrov-Galerkin (MLPG) approach has been developed by S. N. Atluri and co–workers, as a general framework for solving partial differential problems [Atluri and Zhu (1998)]. Under MLPG framework, the PDEs can be solved in their various local symmetric or unsymmetric weak forms, by using a variety of interpolation methods, test functions, integration schemes, and their flexible combinations [Atluri (2004); Atluri, Han, and Rajendran (2004)].

Quoting only recent literature, we point out the noteworthy applications below. MLPG was successfully exploited for the solution of thermo-mechanical, thermoelastic and thermo-piezoelectric problems [Ching and Chen (2006); Sladek, Sladek, Zhang, and Tan (2006); Sladek, Sladek, Zhang, and Solek (2007); Sladek, Sladek, Solek, and Wen (2008)]. In [Gao, Liu, and Liu (2006); Long, Liu, and Li (2008); Sladek, Sladek, Zhang, Solek, and Starek (2007)] the solution of fracture problems is documented, and shell analysis is performed in [Sladek, Sladek, Wen, and Aliabadi (2006); Jarac, Sorić, and Hoster (2007)]. Papers were published on penetration problems [Han, Liu, Rajendran, and Atluri (2006)], nematostatics [Pecher, Elston, and Raynes (2006)], magnetic diffusion [Johnson and Owen (2007)], heat conduction [XueHong, ShengPing, and WenQuan (2007)], composite materials [Dang and Sankar (2008)], incompressible viscous fluid flow [Haji Mohammadi (2008)], topology optimization of structures [Li and Atluri (2008a,b)].

Though tensor product basis functions have been suggested as possible candidates for generating the trial and test spaces in meshless methods [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)], the literature quite exclusively considers Radial Basis Functions (RBF) as the natural candidates.

Unlike for FEM methods, comprehensive error analysis for RBF based MLPG techniques is not available. Interesting theoretical results for uniformly elliptic operators on quite general domains are available [Wendland (1999)], but they apply to MLPG with RBF basis trial functions, which is not implemented in practice, since an RBF basis does not provide a Partition of Unity (PU) [Babuska, Banerjee, and

Osborn (2004)] (see also Section 2.1). In order to have trial basis functions matching this property, one can exploit RBF as weights for the Moving Least Squares (MLS) technique, see e.g. [Atluri, Kim, and Cho (1990)].

Following up to date literature, we implemented MLPG methods exploiting RBF weights. These methods require accurate evaluation of 3D integrals on balls and portions of balls, which is not an easy task to perform. To overcome this problem, meshless boundary integral techniques has been proposed, e.g. in [Zhang, Tanaka, and Endo (2005); Atluri, Han, and Rajendran (2004); Li, Shen, Han, and Atluri (2003)]. However, in order to obtain accurate solutions inside the problem domain, we exploit volume–based integration formulas.

Note that in 3D potential problems, where complicate domain geometries are usually involved, linear FEM with either prismatic or tetrahedral elements is the eliged approach in the vast majority of applications. Higher order elements on 3D meshes are difficult both to implement and to manage. Hence in the sequel MLPG methods are compared with tetrahedral FEM [Gambolati, Pini, and Tucciarelli (1986)], and the acronym "FEM" stands for *linear, tetrahedral FEM*.

In this paper we show that effective MLPG solution, non boundary–integral based, of Poisson 3D problems can be obtained at the price of primarily identifying accurate cubature formulas. Moreover, techniques must be devised in order both to improve the evaluation speed of the stiffness matrix entries and to reduce storage requirements.

2 Meshless schemes

Let us consider the linear 3D Poisson equation on the domain Ω

$$-\nabla^2 u(\mathbf{x}) = f(\mathbf{x}), \tag{1}$$

where f is a given source function. Dirichlet and Neumann boundary conditions are imposed on the domain boundary $\partial\Omega$

$$\begin{aligned} &u = \overline{u} \quad \text{on } \Gamma_u \\ &\frac{\partial u}{\partial \vec{n}} \equiv q = \overline{q} \quad \text{on } \Gamma_q \end{aligned} \tag{2}$$

where $\overline{u}$ and $\overline{q}$ are the prescribed potential and normal flux, respectively, on the Dirichlet boundary Γ_u and on the Neumann boundary Γ_q, being $\partial\Omega = \Gamma = \Gamma_u \cup \Gamma_q$. The outward normal direction to Γ is denoted by $\vec{n}$.

It is well known that solving Poisson problem, disregarding for a while Dirichlet boundary conditions, is equivalent to finding that function $u \in \mathscr{S}$, $\mathscr{S}$ being a suitable *trial* space, which satisfies a Petrov–Galerkin weak formulation. More

precisely, for *every test function* $v \in \mathscr{T}$, where $\mathscr{T}$ is a suitable test space, one must impose

$$\int_{\Gamma_u} qv \, \mathrm{d}\Gamma + \int_{\Gamma_q} \overline{q} v \, \mathrm{d}\Gamma + \int_{\Omega} \left(\sum_{k=1}^{3} \frac{\partial u}{\partial x_k} \frac{\partial v}{\partial x_k} - f v \right) \mathrm{d}\Omega = 0, \tag{3}$$

being $\mathbf{x} = (x_1, x_2, x_3)$ any point in Ω. The final Ritz–Petrov–Galerkin approach relies upon restricting conditions (3) to suitable finite–dimensional trial and test spaces $\mathscr{B} = \mathrm{span}\{\phi_1, \ldots, \phi_{N_B}\} \subset \mathscr{S}$, $\mathscr{U} = \mathrm{span}\{\psi_1, \ldots, \psi_{N_T}\} \subset \mathscr{T}$. The approximated solution of Poisson problem is obtained by assuming $u = \sum_i u_i \phi_i$ and solving for the unknowns u_i the $N_B \times N_T$ linear system obtained by writing the eqs (3) for each $v = \psi_j$, $j = 1, \ldots, N_T$. In order to deal with a system with as many equations as unknowns, in the sequel we assume $N_B = N_T$. For numerical treatment, the problem domain must be discretized by a set of N nodes. To each mesh node we associate one basis function and one trial function, hence $N_B = N_T = N$ will be assumed in the sequel.

Using *compact supported* trial and test functions in the weak formulation (3) amounts to writing a set of so called Local Symmetric Weak Forms (LSWF), one for each basis test function. The i-th LSWF may be written as

$$\int_{\Gamma_u^{(i)}} q \, \psi_i \, \mathrm{d}\Gamma + \int_{\Gamma_q^{(i)}} \overline{q} \, \psi_i \, \mathrm{d}\Gamma + \int_{\Omega^{(i)}} \left(\sum_{k=1}^{3} \frac{\partial u}{\partial x_k} \frac{\partial \psi_i}{\partial x_k} - f \, \psi_i \right) \mathrm{d}\Omega = 0. \tag{4}$$

The form is *symmetric* in the sense that both the trial and the test functions have equal order differentiability requirements. We have $\Omega^{(i)} = \mathrm{supp}(\psi_i)$, while $\Gamma_u^{(i)} = \partial\Omega^{(i)} \cap \Gamma_u$ is the intersection of our local integration domain boundary with *Dirichlet* boundary pieces. Analogously, $\Gamma_q^{(i)} = \partial\Omega^{(i)} \cap \Gamma_q$ is the intersection of our local integration domain boundary with *Neumann* boundary pieces. Eventually, $\Gamma_I^{(i)} = \partial\Omega^{(i)} \backslash (\Gamma_u^{(i)} \cup \Gamma_q^{(i)})$, is the portion of $\partial\Omega^{(i)}$ lying inside Γ. We assume that *continuous* test functions are used, hence integrals involving $\Gamma_I^{(i)}$ do not appear, being $\psi_i(\partial\Omega^{(i)}) = 0$.

Solving the discrete variational problem allows one to compute the coefficients $\hat{u}_i$, s.t. the final approximated solution is

$$\tilde{u}(\mathbf{x}) = \sum_{i=1}^{N} \hat{u}_i \, \phi_i(\mathbf{x}).$$

The final MLPG linear system of equations is

$$K\hat{\mathbf{u}} = \mathbf{f} \tag{5}$$

being $\hat{\mathbf{u}}$ the vector of fictitious nodal values, K the stiffness matrix; $\mathbf{f}$ is the known vector.

Note that, unlike in FEM, MLPG trial basis is *not* a cardinal one, i.e. $\phi_j(\mathbf{x}_i) \neq \delta_{ij}$ hold true, hence $\tilde{u}(\mathbf{x}_i) \neq \hat{u}_i$ holds, too. For this reason, the $\hat{u}_j$ values are called *fictitious nodal values*. A *recover step* is to be performed in order to compute the *actual* nodal values, $\tilde{u}_i = \tilde{u}(\mathbf{x}_i)$, $i = 1,\ldots,N$.

Imposing boundary conditions (BC) in MLPG schemes is slightly less straightforward than in FEM methods, due to the facts just recalled. Let us focus on Dirichlet BC. One can *approximately* impose BC on each Dirichlet node, $\mathbf{x}_j$, by simply setting $\hat{u}_j = u_j$. However, to *exactly* impose Dirichlet BC, one must compute the values $\phi_i(\mathbf{x}_k)$, for each $\mathbf{x}_k$ which is a Dirichlet boundary node. An *exact Dirichlet value* $\bar{u}_k$ can be set on the corresponding boundary point, $\mathbf{x}_k$, by replacing the k–th MLPG linear system equation with the equation $\tilde{u}(\mathbf{x}_k) = \sum_{i=1}^{N} \hat{u}_i \phi_i(\mathbf{x}_k) = \bar{u}_k$. This latter procedure, called in the sequel *exact Dirichlet*, is usually more time consuming than the former *inexact* one. In the sequel, whenever not otherwise stated, *exact* Dirichlet BC are imposed.

2.1 Trial and test spaces

Trying to extend meshless variational approaches from 1–dimensional to d–dimensional problems, one can exploit tensor–product functions as the test and trial functions, as suggested e.g. in [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)]. We follow a different approach, which is ubiquitous in literature, i.e. we start from Radial Basis Functions (RBF); they have the general form [Powell (1992)]

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \, w(\left\|\mathbf{x}-\mathbf{x}_j\right\|).$$

Let us assume a so called *generating RBF*, $g(r)$, together with a set of points, $S = \{\mathbf{x}_j, \quad j = 1,\ldots,N\}$, are given. The space spanned by the functions $\phi_j(\mathbf{x}) = g(\left\|\mathbf{x}-\mathbf{x}_j\right\|)$, $i = 1,\ldots,N$, will be called the *RBF space generated by g* (and S). The RBF spaces proposed in literature do not provide PU property [Babuska, Banerjee, and Osborn (2004)], also called *zero* order consistency, i.e. $\sum_{j=1}^{N} \phi_j(\mathbf{x}) = 1$; moreover, constant functions *do not belong* to these finite dimensional spaces. Note that such a drawback *does not* prevent convergence of meshless Galerkin methods, as proved in [Wendland (1999)], but is likely to produce very low accuracy in approximating polynomials at any finite dimension. To overcome this problem, many authors suggest that the basis trial functions, ϕ_j, are obtained by exploiting a suitable technique which ensures that PU is fulfilled. Following many

authors, we obtain this goal by exploiting the Moving Least Square (MLS) technique with a set of RBF *weights*, hence obtaining approximations based upon the so called MLS *shape functions* [Atluri and Zhu (2002); Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Lancaster and Salkauskas (1981)]. They become our trial basis functions. In this paper, we show results exploiting quadratic MLS, i.e. the weighted minimization is performed on quadratic polynomials [Lancaster and Salkauskas (1981)], and the ensuing shape functions, beside PU, also achieve *first* order consistency, i.e. $\sum_{j=1}^{N} x_{j,k} \cdot \phi_j(x_1,x_2,x_3) = x_k$, $k = 1,2,3$. Note that $x_{j,k}$ is the k-th component of the j-th node, $\mathbf{x}_j = (x_{j,1},x_{j,2},x_{j,3})$.

When implementing MLPG, in principle one can avoid explicitly computing the MLS shape functions [Atluri (2004)], but, in order to perform the *recover* step, i.e. computing the nodal solution values, one must compute the values $V = \{\phi_j(\mathbf{x}_i), i,j = 1,\ldots,N\}$. This is the reason why *recovering* requires further computations compared to merely solving the MLPG linear system, which provides the fictitious values only. Note that the values $\phi_i(\mathbf{x}_k)$ for each $\mathbf{x}_k$ on the Dirichlet boundary, which form a subset in V, allow for imposing the exact Dirichlet BC, as pointed out in [Atluri, Kim, and Cho (1990)]; previous papers stated that only *inexact* Dirichlet BC can be imposed in MLPG methods based upon MLS.

It is important to note that the support of each trial basis function ϕ_j, which is an MLS shape function, is equal to the spherical support of the corresponding RBF weight function w_j, whose radius is denoted in the sequel by r_j.

In order to specify our MLPG procedure, we need to identify suitable *test* functions. Many choices were proposed [Atluri (2004)]; following [Atluri and Zhu (2002); Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Lu, Belytschko, and Gu (1994)], we use RBF functions with the same analytical expression as that used for the MLS weights. This approach is called MLPG1 in [Atluri (2004)], and hereafter is denoted. Note that trial functions differ from the test ones, hence *unsymmetric* stiffness matrices are generated. Our computational experience on 2D problems [Ferronato, Mazzia, Pini, and Gambolati (2007a,b)] suggests that the support of the test functions can be smaller than the trial ones. In the sequel we denote by ρ_i the support radius of the *test* RBF function based upon node $\mathbf{x}_i$. For more details, see for instance [Ferronato, Mazzia, Pini, and Gambolati (2007a)].

2.2 RBF generators

As the MLS weights and test functions, we exploited either Gaussian or spline 3D RBF. In both cases the support of the function is a ball with radius η_i.

Gaussian RBF have the analytical form [Lu, Belytschko, and Gu (1994)]

$$w_i(\mathbf{x}) = \begin{cases} \dfrac{\exp\left[-\left(\frac{d_i}{c_i}\right)^{2k}\right] - \exp\left[-\left(\frac{\eta_i}{c_i}\right)^{2k}\right]}{1-\exp\left[-\left(\frac{\eta_i}{c_i}\right)^{2k}\right]}, & 0 \le d_i \le \eta_i \\ 0, & d_i \ge \eta_i \end{cases} \tag{6}$$

where $d_i = \|\mathbf{x}-\mathbf{x}_i\|$ is the distance between $\mathbf{x}$ and $\mathbf{x}_i$; As usual in literature, we set $k = 1$. The parameter c_i controls the shape of the function. When a uniform grid with mesh size h is enrolled, following [Lu, Belytschko, and Gu (1994)] we set $c_i = c = h$.

We also considered quartic spline functions after [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)]

$$w_i(\mathbf{x}) = \begin{cases} 1-6\left(\frac{d_i}{\eta_i}\right)^2+8\left(\frac{d_i}{\eta_i}\right)^3-3\left(\frac{d_i}{\eta_i}\right)^4, & 0 \le d_i \le \eta_i \\ 0, & d_i \ge \eta_i \end{cases} \tag{7}$$

Our results published in [Ferronato, Mazzia, Pini, and Gambolati (2007a,b)] show that Gauss RBF weights are preferable over splines. In the sequel, the Gauss RBF generating function is exploited, unless not specified otherwise.

3 Implementation issues

3.1 Cubature rules

In order to obtain an effective MLPG1 solution, an accurate and efficient evaluation of the integrals in the LSWF (4) must be performed. Since we decided to exploit RBF–based trial and test spaces, the integration sub-domains in our LSWF, $\Omega^{(i)} = \mathrm{supp}(\psi_i) \cap \Omega$, result to be either (a) spheres inside Ω, when interior nodes far from the boundary are considered; or (b) portions of spheres when $\Omega^{(i)}$ overflows the problem domain Ω. We focus on cubature rules developed for sub-domains of type (a) only. Integration over regions of type (b) is implemented by the same procedure as in case (a), by neglecting the quadrature points lying outside Ω.

The simplest idea is to exploit Gauss-Legendre's product rule with integration nodes chosen along x-, y-, and z-directions. Unfortunately, this approach is not as accurate as one should expect, even confining to 2D integrals. Indeed, when approximating the integral of $z(x,y) = 1$ on the unit circle, in order to achieve an as small as 2×10^{-4} magnitude absolute error, one must enroll a 20×20 Gauss-Legendre formula [De and Bathe (2001)].

We tested the feasibility of cubature rules in producing accurate MLPG solutions, by performing the so called *Patch Test* [Atluri (2004)]. It consists in checking that the MLPG solution is *exact* when Poisson problem on a square is solved on only one internal node, all the other nodes being Dirichlet boundary points. We consider both *linear* and *quadratic* Patch Tests, which means that the exact solution to be reproduced is either a linear, or a quadratic one, respectively.

Using Gauss formulas in a polar coordinate frame, the 2D Patch Test is fulfilled, provided suitable nodes are enrolled [Mazzia, Ferronato, Pini, and Gambolati (2007)]. The 3D Patch Test is fulfilled only when a large number of nodes is available.

Modern scientific literature is surprisingly poor of integration rules for balls; this is a major point why meshless methods based upon compact–supported RBF are said to be troublesome. We focused on well-known Stroud's cubature rules [(Stroud, 1971, pag. 291,292)], more precisely Rule $S_3 : 7-4$, with degree 7, featuring 64 nodes; Rule $S_3 : 14-1$, degree 14, with 288 nodes (subroutine `SPH14`); degree 15, 512 nodes (subroutine `SPH15`, after [(Stroud, 1971, pag. 352)]). The 7–point Stroud formula matches the linear Patch Test, but not the quadratic one. The 14–point Stroud formula does not comply neither the linear Patch Test, nor the quadratic one, while the 15–point formula matches both.

We found that the 15–degree rule provide accurate integral values at affordable computational costs. In the sequel, all 3D integrations are performed using this rule.

3.2 Speeding up computations

When the radii of the test functions, ρ_i, are either as large as, or larger, than the domain diameter, D, a dense MLPG1 linear system matrix K is obtained. The computational cost rapidly increases with the number of nodes, N, and storage requirements become prohibitively large. In order to obtain a *sparse*, manageable, MLPG1 linear system, we set $\rho_i \ll D$, $i = 1,\ldots,N$. The ensuing matrix K has in general an arbitrary pattern; hence K was stored in CSR (Compressed Sparse Row) format.

When performing MLPG1 computations, a main time consuming task consists in evaluating the stiffness coefficients, i.e. integrating $\nabla\phi_i \cdot \nabla\psi_j$ product functions. Since the radii of both the trial and test function supports are usually far smaller than the domain diameter, one has $\nabla\phi_i \cdot \nabla\psi_j = 0$ each time $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|$ is large, compared to r_i and ρ_j. Hence, in order to efficiently evaluate the integrals in eq. (4), one is likely to first identify for each node, $\mathbf{x}_i$, both those indexes j s.t. $\nabla\phi_j(\mathbf{x}_i) \neq 0$, and those indexes k s.t. $\nabla\psi_k(\mathbf{x}_i) \neq 0$. In principle, all grid nodes must be considered, but in order to attain a reasonable efficiency, a search value s is set, s.t. when

$|\mathbf{x}_i - \mathbf{x}_m| > s$, both $\nabla\phi_m(\mathbf{x}_i) \simeq 0$, and $\nabla\psi_m(\mathbf{x}_i) \simeq 0$, $m = 1,\ldots,N$, $j \neq m$ can be assumed. When performing cubature rules, we considered only the points inside the sphere centered at $\mathbf{x}_i$ with radius $R = \min\{s, 2(\rho_i + r_i)\}$. Such a strategy allows for a dramatic reduction of the computational cost.

3.3 Assessment of radii

Our MLPG1 method need the crucial assessment of trial and test function radii. There is no exhaustive theory for performing this issue, hence parameter tuning was performed by analyzing MLPG1 error on a set of test solutions, which we now list.

3.3.1 Test solutions

We browsed the not so large literature on solving 3D potential problems by non–standard (i.e. non–FEM) methods. After [Zhang, Tanaka, and Endo (2004)], we consider the harmonic polynomial

$$u(x,y,z) = x^3 + y^3 + z^3 - 3yx^2 - 3xz^2 - 3zy^2. \tag{8}$$

In the sequel, this test solution is labeled (uP3).

After [Wang, Zhong, and Zhang (2006)], two trigonometric functions are considered.

- The function (uT1) which equally varies along the three coordinate directions

$$u(x,y,z) = \sin x + \sin y + \sin z + \sin(3x) + \sin(3y) + \sin(3z). \tag{9}$$

- The function (uT2) which changes more rapidly in the z- and y-directions.

$$u(x,y,z) = \sin x + \sin y + \sin z + \sin(5y) + \sin(10z). \tag{10}$$

Moreover, we consider the *composed cosine–polynomial* solution (uCP) after [Mazzia, Pini, Putti, and Sartoretto (2003)]

$$u = \cos\left(3\pi\left(\frac{x^3+y^3+z^3}{3} - \frac{x^2+y^2+z^2}{2}\right)\right). \tag{11}$$

3.3.2 Heuristics.

To allow parameter assessment by a reasonably small test bed, we cannot afford neither to work with an irregular domain, nor exploiting non–uniform meshes. We

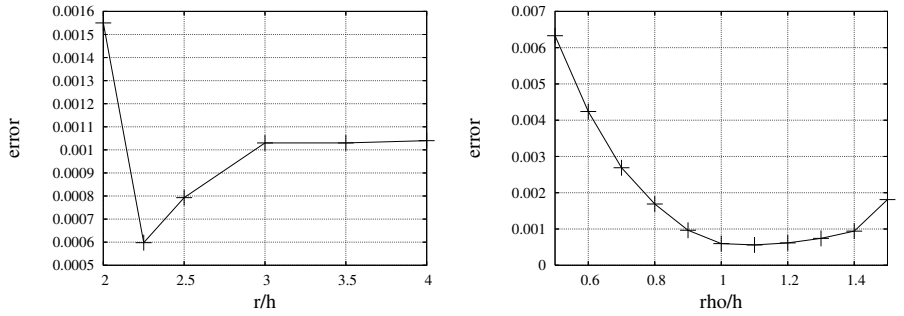

Figure 1: MLPG1 error (e) behavior vs the ratio r/h, when $\rho = h$ was set (left frame). The right frame shows the MLPG1 error behavior vs the ratio ρ/h, when $r = 2.25\,h$. The exact solution is (uCP), $h = 1/8$ was set.

choose a simple problem domain, i. e. the cube $C = [0,1]^3$. A uniform discretization grid is set, featuring the same mesh size $h = h_x = h_y = h_z$ on each coordinate direction. Irrespective of the grid point $\mathbf{x}_i$, we set unique r and ρ values, which depend upon the grid spacing [Nie, Atluri, and Zuo (2006)].

Let $\tilde{u}$ be our numerical approximation function of the exact problem solution, u. After [Atluri (2004)] we exploit the following error estimator

$$e = \frac{\sqrt{\sum_{i=1}^{N}(u_i - \tilde{u}_i)^2}}{\|\underline{u}\|_2}, \tag{12}$$

which is consistent with the L_2 norm of the true error, where $\underline{u} = (u_1, \ldots, u_N)$, $u_i = u(\mathbf{x}_i)$ is the exact solution nodal value.

The generating RBF is Gauss exponential after eq. (6), where $c_i = h$ is set.

The basis trial function support radius, r, should be chosen large enough so that the regularity of K is ensured [Nie, Atluri, and Zuo (2006)]. On the other hand, r should be small enough to maintain the local character of the MLS approximation. A guessed interval is $2\,h \leq r \leq 4\,h$, which is so large that K is non singular.

The test function support radius, ρ, must be small, in order to reduce the computational cost of integrations in our LSWF. On the other hand ρ must be so large that the support union of all test functions encompasses the whole domain. The guess $\rho = h$ matches these requirements.

Assume $\rho = h$ is set, and (uCP) solution is computed. The left frame in Figure 1 shows the behavior of the MLPG1 error as a function of the ratio r/h. One can see

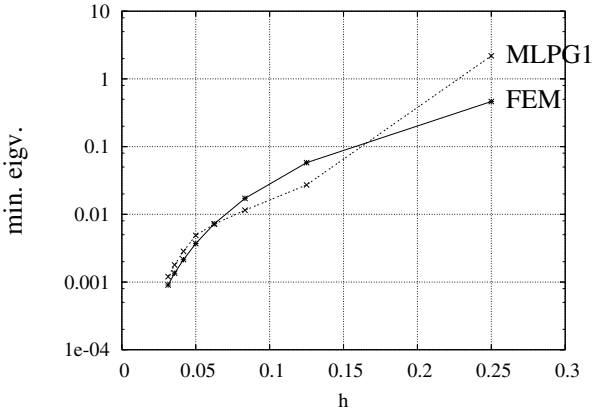

Figure 2: Minimum eigenvalues of FEM and MLPG1 linear operators vs the mesh spacing, h.

that a minimum error is attained for $r = 2.25\,h$. This result was obtained for all our test solutions.

Once $r = 2.25\,h$ is set, we study the assessment of ρ. The right frame in Figure 1 shows the behavior of the MLPG1 error as a function of the ratio ρ/h. One can see that $\rho/h = 1$, i.e. $\rho = h$ is a reasonable choice for attaining a small error. This result makes us more confident that our $\rho = h$ guess, set before assessing r, is a well–founded choice. This same conclusion was drawn when approximating all our test solutions.

In the sequel, $r = 2.25\,h$, $\rho = h$, are set.

4 Numerical results

4.1 Discrete operator conditioning

Figure 2 shows the minimum eigenvalue behavior of both FEM and MLPG1 operator, when h changes. Note that MLPG1 operator, like FEM, is a positive definite one. Our numerical experiments show that the maximum eigenvalue of MLPG1 operators in all our test problems, is close to 1 when the mesh spacing is "small", like for FEM operators. Hence the conditioning of both FEM and MLPG1 operators is quite the same when the asymptotic regime is reached.

The MLPG1 linear system in 2D problems is usually solved via direct methods, like [Li, Demmel, Gilbert, and Grigori (2007); MUMPS (2007); PARADISO (2007)]. These methods are efficient and accurate for 2D problems, but too storage demand-

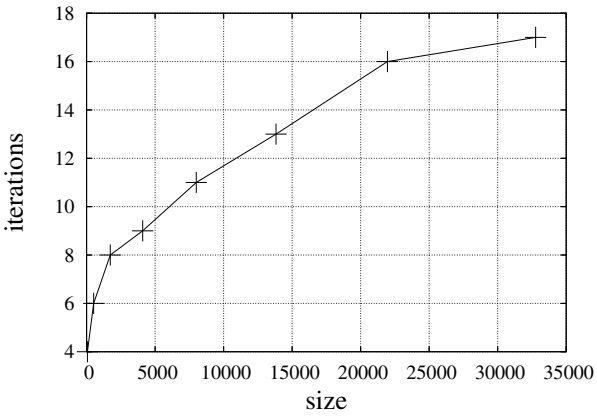

Figure 3: Number of iterations spent to solve the MLPG1 linear system, vs its size, when the (uCP) solution is computed.

ing when 3D real–life problems are attacked. Preconditioned iterative methods are the best choice for large 3D problems. We efficiently solved our 3D problems via preconditioned Bi-CGSTAB [van der Vorst (1992)]. Preconditioning was performed via incomplete Crout factorization [Kershaw (1978)]. The iterations for solving the MLPG1 linear system $Kx = b$, were stopped when the relative residual is smaller than a suitable tolerance, more precisely $\|b - Kx_m\|/\|b\| \leq 10^{-15}$. Figure 3 shows that the number of iterations for solving the MLPG1 linear system is likely to be as small as $O(N^{1/3})$. This result confirms the good conditioning of our MLPG1 operator. Other MLPG techniques do not provide such numerically well–behaved projected operators. As an example, we implemented the MLPG5 method [Atluri (2004)], which exploits Heaviside step functions as the *test* functions. When solving our Poisson test problems with MLPG5, we cannot compute the solution of the ensuing linear system by ILUT [Saad (1994)] preconditioned Bi-CGSTAB, since no convergence is attained.

4.2 MLPG1 vs FEM error behavior

Figure 4 shows our error results when solving Poisson problems on the unit cube, imposing Dirichlet boundary conditions. MLPG1 solution was computed using the uniform grids $\mathcal{G}_h$, featuring grid spacings $h = 1/(4q)$, $q = 1, 2, \ldots, 8$. For comparison, linear FEM solutions of our test problems were computed, too. Our FEM implementation [Gambolati, Pini, and Tucciarelli (1986)] exploited the tetrahedral meshes which are obtained by dividing into three pieces each cube in the

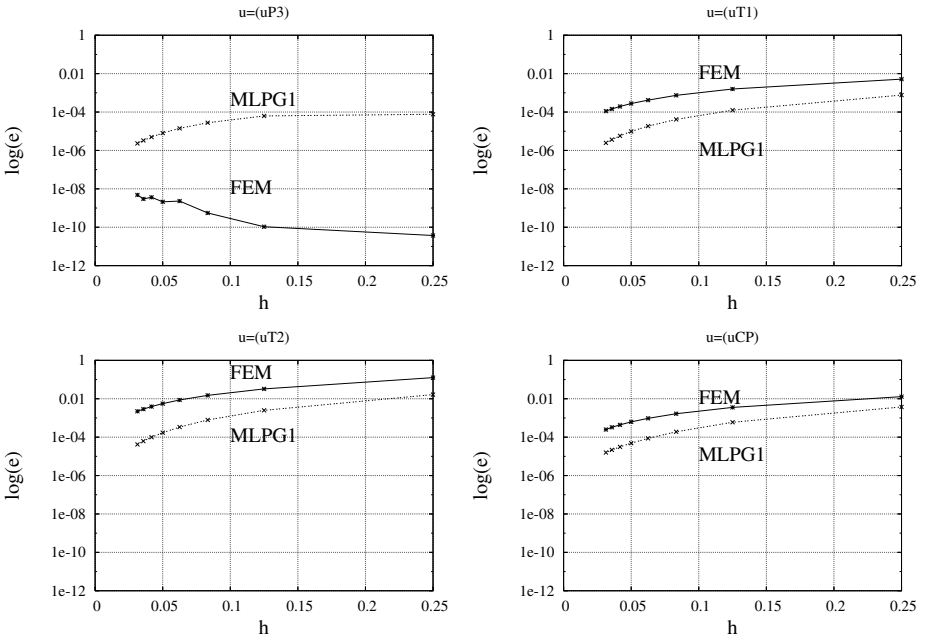

Figure 4: FEM and MLPG1 error behaviors vs the mesh spacing h. Going from top to bottom, the exact solutions are (uP3), (uT1), (uT2), and (uCP).

corresponding $\mathscr{G}_h$ mesh. Both MLPG1 and FEM errors were estimated using formula (12).

Figure 4 shows that FEM error decreases with h, except but the case (uP3). In the latter case, even when the *coarsest mesh* is exploited, FEM solution is as accurate as the linear system solver. Refining the mesh seemingly enlarges the error.

MLPG1 errors are smaller than FEM ones, except in the (uP3) case, where the superiority of linear FEM in approximating low degree polynomial solutions is clearly shown.

We also tested our MLPG1 code when the spline in eq. (7) is enrolled as the RBF generator for both the trial and test spaces. Table 1 shows MLPG error behavior vs the mesh spacing h, when the (uCP) solution is approximated. As one can see, poor accuracy was obtained. Moreover, no useful approximations can be computed when $h < 6.25 \times 10^{-2}$. We conclude that quartic splines given by eq. (7) are not apt for solving 3D potential test problems by our MLPG1 procedure.

Table 1: MLPG error behavior vs the mesh spacing h, when the RBF *quartic spline* given by eq. (7) is exploited. The (uCP) solution is approximated.

h	MLPG error
2.50E-1	4.10E-3
1.25E-1	1.58E-3
8.33E-2	7.00E-4
6.25E-2	4.98E-4

4.3 Convergence order

Assuming that an asymptotic regime is reached, the error, e_h, of a numerical procedure, as a function of the grid spacing h can be roughly assumed to be $e_h = C \cdot h^p$. The constant C does not depend on the geometry of the mesh, when an initial uniform mesh is enrolled and the other meshes are obtained by uniform refinements. Assume that the errors were computed for mesh spacings $h_1 > h_2 > \cdots > h_r$. Convergence order estimates, $p(h_i)$, can obtained by

$$p(h_i) = (\log \frac{e_{h_i}}{e_{h_{i-1}}})/(\log \frac{h_i}{h_{i-1}}), \quad i = 2, \ldots, r.$$

Figure 5 shows the convergence order estimations computed on the ground of our results shown in Figure 4. One can see that, except when the solution (uP3) is considered, FEM convergence order approaches $p = 2$ when h goes to zero, as predicted by theory. In the (uP3) case, FEM error displays a peculiar behavior, discussed in the previous Section, producing negative, worthless, $p(h_i)$ values.

When (uP3), (uT1) and (uT2) solutions are chased, $p > 2$ hold true for MLPG1 convergence order, except only for (uP3), when the h–pair (0.25,0.125) is considered. This is a better convergence speed than linear FEM. A distinguished case is when the (uCP) solution is considered. In this case MLPG1 convergence order is oscillating. We confronted with analogous behaviors when solving poroelastic problems with MLPG1, due to inaccurate integral evaluations [Ferronato, Mazzia, Pini, and Gambolati (2007b)]. We argue that the peculiar MLPG1 convergence behavior on (uCP) is due to an higher variation in the accuracy of the MLPG1 known vector entries, when the number of sampling points changes. In order to substantiate this hypothesis, assume that we need to compute a paradigmatic MLPG1 known vector entry, i.e. an integral $I = \int_\Omega g(\mathbf{x})\, d\mathbf{x}$, where $g = f\bar{w}$, and $\bar{w}$ is Gauss RBF weight based upon a grid point, say the centroid of our test domain $C = [0,1]^3$, f being our source function. Let us consider the uniform grids, featuring grid spacings $h = \bar{h}, 2\bar{h}, 4\bar{h}$,

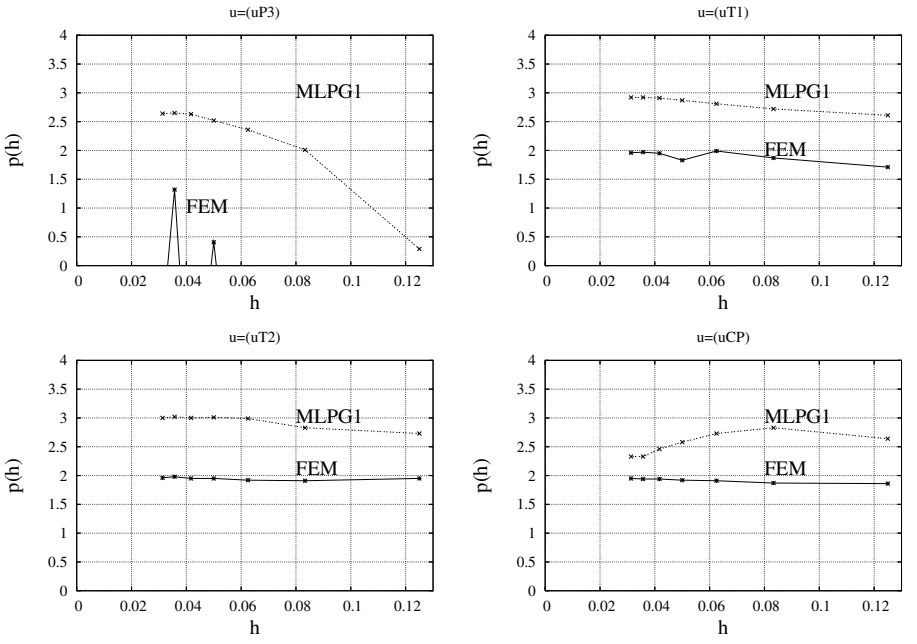

Figure 5: Behaviors of FEM and MLPG1 convergence order estimations, $p(h)$, vs h. Going from top to bottom, the exact solutions are (uP3), (uT1), (uT2), and (uCP).

where $\bar{h}$ is a suitable *base* grid spacing. Assume that the *exact I* value is obtained by running Matlab `triplequad` command on the *finest* grid. On the other hand, let us consider the *naive* cubature rule $I_h = h^3 \sum_{j=1}^{n^3} g(\mathbf{x}_j^{(h)})$, where $n = [1/h]$. Figure 6 plots the relative differences $|1 - I_h/I|$ vs h, when $\bar{h} = 3.8 \times 10^{-3}$, a grid spacing which ensures a reliable sampling of the function g. The approximated solution is either (uT2) or (uCP). One can see that the variation in the difference is much higher in the (uCP) case, compared to the (uT2) one, confirming that the variation of the cubature error produced by changing the number of sampling points is higher when (uCP) solution is chased. This result corroborates our hypothesis that cubature errors are responsible for the oscillating MLPG1 convergence behavior in the (uCP) case.

4.4 Dirichlet BC

Figure 7 shows the MLPG1 error behaviors vs h, when the (uCP) solution is approximated, and either *exact*, or *inexact* Dirichlet BC are imposed. One can see that

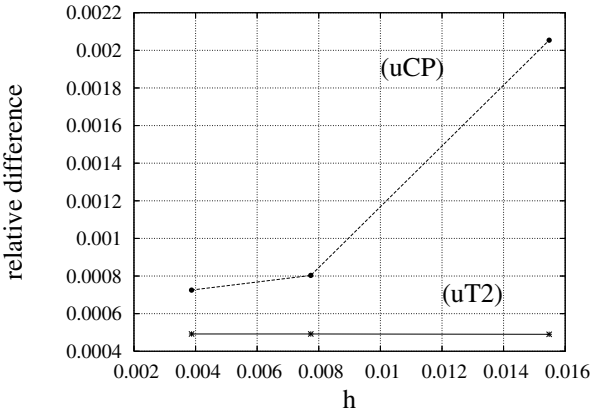

Figure 6: Relative difference on cubature estimations.

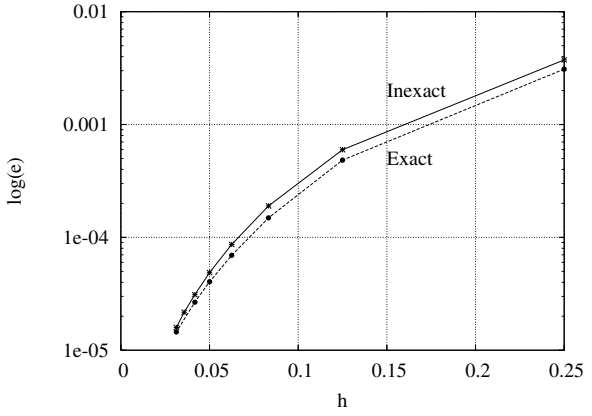

Figure 7: MLPG1 error vs h when either exact or inexact Dirichlet BC are imposed. The (uCP) solution is approximated.

the difference in the errors is negligible, suggesting that the more time-consuming procedure of imposing exact boundary conditions, does not provide useful improvements. The same conclusion was obtained for all our test solutions.

5 Conclusions

We analyzed the MLPG solution of 3D Poisson problems. The following points deserve mention.

- Test and trial spaces were identified, and crucial parameter values were assessed, in order to obtain an effective MLPG1 scheme for solving 3D Poisson problems.
- Accurately performing 3D integrals is a main issue for obtaining accurate MLPG1 solutions. Using product Gauss rules does not provide accurate enough estimations. We identified cubature rules which entail the required accuracy.
- While linear systems arising in 2D problems are usually solved via direct methods, solving those large linear systems arising in 3D applications requires iterative solvers. We efficiently solved our 3D problems via preconditioned Bi-CGSTAB.
- Strategies for reducing the computational cost of the MLPG1 computation flow were devised and tuned, which allow for a dramatic reduction in the computing time. Applying these techniques has an analogous cost-reduction valence as the *assembling* step in FEM procedures.
- The behavior of our MLPG1 error was analyzed and compared with linear FEM. Apart from the case when low degree polynomial solutions are approximated, MLPG1 error reduction is superior than FEM one. MLPG1 convergence speed is anytime $p > 2$, which outperforms $p \simeq 2$ produced by FEM. However, MLPG1 convergence speed is affected by the cubature accuracy in the computation of the MLPG1 known vector.
- Using quartic splines given by eq. (7), as both the MLS weight and the test functions, did not provide useful results.

Acknowledgement: This work was partially funded by the Italian MUR Project (PRIN) "Advanced numerical methods and models for environmental fluid–dynamics and geomechanics". Support by the 2007 FRA granted by Università di Venezia and INDAM–GNCS was also given.

References

Atluri, S. N. (2004): *The Meshless method (MLPG) for domain & BIE discretizations*. Tech Science, 2004, Forsyth GA.

Atluri, S. N.; Han, Z. D.; Rajendran, A. M. (2004): A new implementation of the meshless finite volume method, through the MLPG "mixed" approach. *Computer Modeling in Engineering and Sciences*, vol. 6, no. 6, pp. 491–513.

Atluri, S. N.; Kim, H.-G.; Cho, J. Y. (1990): A critical assessment of the truly meshless local Petrov-Galerkin (MLPG), and local boundary integral equation (LBIE) methods. *Computational Mechanics*, vol. 24, no. 5, pp. 348–372.

Atluri, S. N.; Liu, H. T.; Han, Z. D. (2006): Meshless Local Petrov-Galerkin (MLPG) Mixed Collocation method for elasticity problems. *Computer Modeling in Engineering and Sciences*, vol. 14, no. 3, pp. 141–152.

Atluri, S. N.; Liu, H. T.; Han, Z. D. (2006): Meshless Local Petrov-Galerkin (MLPG) Mixed Finite Difference method for solid mechanics. *Computer Modeling in Engineering and Sciences*, vol. 15, no. 1, pp. 1–16.

Atluri, S. N.; Zhu, T. (1998): A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational Mechanics*, vol. 22, pp. 117–127.

Atluri, S. N.; Zhu, T. (2002): The meshless local Petrov-Galerkin (MLPG) method: A simple & less–costly alternative to the finite element methods. *Computer Modeling in Engineering and Sciences*, vol. 3, no. 1, pp. 11–51.

Babuska, I.; Banerjee, U.; Osborn, J. E. (2004): Generalized finite element methods – main ideas, results and perspective. *International Journal of Computational Methods*, vol. 1, no. 1, pp. 67–103.

Belytschko, T.; Krongauz, Y.; Organ, D.; Fleming, M.; Krysl, P. (1996): Meshless methods: an overview and recent developments. *Comp. Methods App. Mech. Eng.*, vol. 139, pp. 3–47.

Belytschko, T.; Krysl, P.; Krongauz, Y. (1997): A three–dimensional explicit element–free galerkin method. *International Journal for Numerical Methods in Fluids*, vol. 24, pp. 1253–1270.

Ching, H. K.; Chen, J. K. (2006): Thermomechanical analysis of functionally graded composites under laser heating by the MLPG method. *Computer Modeling in Engineering and Sciences*, vol. 13, no. 3, pp. 199–217.

Dang, T. D.; Sankar, B. V. (2008): Meshless Local Petrov–Galerkin micromechanical analysis of periodic composites including shear loadings. *Computer Modeling in Engineering and Sciences*, vol. 26, no. 3, pp. 169–187.

De, S.; Bathe, K. J. (2001): The method of finite spheres with improved numerical integration. *Comp. & Struct.*, vol. 79, pp. 2183–2196.

Ferronato, M.; Mazzia, A.; Pini, G.; Gambolati, G. (2007): Accuracy and performance of meshless local Petrov-Galerkin methods in 2-D elastostatic problems. *JP Journal of Solids and Structures*, vol. 1, pp. 35–59.

Ferronato, M.; Mazzia, A.; Pini, G.; Gambolati, G. (2007): A meshless method for axi-symmetric poroelastic simulations: numerical study. *International Journal for Numerical Methods in Engineering*, vol. 70, no. 11, pp. 1346–1365.

Gambolati, G.; Pini, G.; Tucciarelli, T. (1986): A 3-D finite element conjugate gradient model of subsurface flow with automatic mesh generation. *Adv. Water Resources*, vol. 3, pp. 34–41.

Gao, L.; Liu, K.; Liu, Y. (2006): Applications of MLPG method in dynamic fracture problems. *Computer Modeling in Engineering and Sciences*, vol. 12, no. 3, pp. 181–195.

Haji Mohammadi, M. (2008): Stabilized meshless Local Petrov-Galerkin (MLPG) method for incompressible viscous fluid flows. *Computer Modeling in Engineering and Sciences*, vol. 29, no. 2, pp. 75–94.

Han, Z. D.; Atluri, S. N. (2004): Meshless local Petrov–Galerkin (MLPG) approaches for solving 3D problems in elasto–statics. *Computer Modeling in Engineering and Sciences*, vol. 6, no. 2, pp. 169–188.

Han, Z. D.; Liu, H. T.; Rajendran, A. M.; Atluri, S. N. (2006): The applications of Meshless Local Petrov-Galerkin (MLPG) approaches in high-speed impact, penetration and perforation problems. *Computer Modeling in Engineering and Sciences*, vol. 14, no. 2, pp. 119–128.

Jarac, T.; Sorić, J.; Hoster, J. (2007): Analysis of shell deformation responses by the Meshless Local Petrov-Galerkin (MLPG) approach. *Computer Modeling in Engineering and Sciences*, vol. 18, no. 3, pp. 235–246.

Johnson, J. N.; Owen, J. M. (2007): A Meshless Local Petrov-Galerkin method for magnetic diffusion in non-magnetic conductors. *Computer Modeling in Engineering and Sciences*, vol. 22, no. 3, pp. 165–188.

Kershaw, D. S. (1978): The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations. *J. Comp. Phys.*, vol. 26, pp. 43–65.

Lancaster, P.; Salkauskas, K. (1981): Surfaces generated by moving least squares methods. *Math. Comp.*, vol. 37, no. 155, pp. 141–158.

Li, Q.; Shen, S.; Han, Z. D.; Atluri, S. N. (2003): Application of meshless local Petrov-Galerkin (MLPG) to problems with singularities, and material discontinuities, in 3-D elasticity. *Computer Modeling in Engineering and Sciences*, vol. 4, no. 5, pp. 571–585.

Li, S.; Atluri, S. N. (2008): The MLPG mixed collocation method for material orientation and topology optimization of anisotropic solids and structures. *Computer Modeling in Engineering and Sciences*, vol. 30, no. 1, pp. 37–56.

Li, S.; Atluri, S. N. (2008): Topology-optimization of structures based on the MLPG mixed collocation method. *Computer Modeling in Engineering and Sciences*, vol. 26, no. 1, pp. 61–74.

Li, S.; Demmel, J.; Gilbert, J.; Grigori, L. (2007): SuperLU. `http://crd.lbl.gov/~xiaoye/SuperLU`, 2007. Last accessed: December 1, 2007.

Long, S. Y.; Liu, K. Y.; Li, G. Y. (2008): An analysis for the elasto-plastic fracture problems by the Meshless Local Petrov-Galerkin method. *Computer Modeling in Engineering and Sciences*, vol. 28, no. 3, pp. 203–216.

Lu, Y. Y.; Belytschko, T.; Gu, L. (1994): A new implementation of the element free Galerkin method. *Comp. Methods App. Mech. Eng.*, vol. 113, pp. 397–414.

Mazzia, A.; Ferronato, M.; Pini, G.; Gambolati, G. (2007): A comparison of numerical integration rules for the meshless local Petrov–Galerkin method. *Numerical Algorithms*, vol. 45, no. 1–4, pp. 61–74.

Mazzia, A.; Pini, G.; Putti, M.; Sartoretto, F. (2003): Comparison of 3D flow fields arising in mixed and standard unstructured finite elements. In et al., P. S.(Ed): *Computational Science – ICCS 2003*, volume 2657 of *Lecture Notes in Computer Sciences*, pp. 560–567, Berlin. Springer-Verlag.

MUMPS: a MUltifrontal Massively Parallel sparse direct Solver. `http://graal.ens-lyon.fr/MUMPS`, 2007. Last accessed: December 1, 2007.

Nie, Y. F.; Atluri, S. N.; Zuo, C. W. (2006): The optimal radius of the support of radial weights used in moving least squares approximation. *Computer Modeling in Engineering and Sciences*, vol. 12, no. 2, pp. 137–147.

PARADISO solver project. http://www.paradiso-project.org/, 2007. Last accessed: December 1, 2007.

Pecher, R.; Elston, S.; Raynes, P. (2006): Meshfree solution of Q-tensor equations of nematostatics using the MLPG method. *Computer Modeling in Engineering and Sciences*, vol. 13, no. 2, pp. 91–101.

Powell, M. J. D. (1992): The theory of radial basis function approximation in 1990. In Light, W.(Ed): *Advances in Numerical Analysis. Wavelets, Subdivision Algorithms, and Radial Basis Functions*, pp. 105–210. Clarendon Press, Oxford.

Saad, Y. (1994): ILUT: A dual threshold incomplete ILU factorization. *Numer. Lin. Alg. Appl.*, vol. 1, pp. 387–402.

Schembri, P.; Crane, D. L.; Reddy, J. N. (2004): A three–dimensional computational procedure for reproducing meshless methods and the finite element method. *International Journal for Numerical Methods in Engineering*, vol. 61, pp. 896–927.

Sladek, J.; Sladek, V.; Solek, P.; Wen, P. H. (2008): Thermal bending of Reissner–Mindlin plates by the MLPG. *Computer Modeling in Engineering and Sciences*, vol. 28, no. 1, pp. 57–76.

Sladek, J.; Sladek, V.; Wen, P. H.; Aliabadi, M. H. (2006): Meshless Local Petrov-Galerkin (MLPG) method for shear deformable shells analysis. *Computer Modeling in Engineering and Sciences*, vol. 13, no. 2, pp. 103–117.

Sladek, J.; Sladek, V.; Zhang, C.; Solek, P. (2007): Application of the MLPG to thermo-piezoelectricity. *Computer Modeling in Engineering and Sciences*, vol. 22, no. 3, pp. 217–233.

Sladek, J.; Sladek, V.; Zhang, C.; Solek, P.; Starek, L. (2007): Fracture analyses in continuously nonhomogeneous piezoelectric solids by the MLPG. *Computer Modeling in Engineering and Sciences*, vol. 19, no. 3, pp. 247–262.

Sladek, J.; Sladek, V.; Zhang, C.; Tan, C. L. (2006): Meshless Local Petrov-Galerkin method for linear coupled thermoelastic analysis. *Computer Modeling in Engineering and Sciences*, vol. 16, no. 1, pp. 57–68.

Stroud, A. H. (1971): *Approximate Calculation of Multiple Integrals*. Prentice-Hall, Englewood Cliffs, NJ.

van der Vorst, H. A. (1992): Bi-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, vol. 13, pp. 631–644.

Wang, J.; Zhong, W.; Zhang, J. (2006): A general meshsize fourth-order compact difference discretization scheme for 3D Poisson equation. *Applied Mathematics and Computation*, vol. 183, no. 2, pp. 804–812.

Wendland, H. (1999): Meshless Galerkin methods using radial basis functions. *Math. Comp.*, vol. 68, no. 228, pp. 1521–1531.

Xiong, S.; Rodrigues, J. M. C.; Martins, P. A. F. (2003): Numerical simulation of three–dimensional steady–state rolling by the reeproducing kernel particle method. *Engineering Computations*, vol. 20, no. 7, pp. 855–874.

XueHong, W.; ShengPing, S.; WenQuan, T. (2007): Meshless Local Petrov-Galerkin Collocation method for two-dimensional heat conduction problems. *Computer Modeling in Engineering and Sciences*, vol. 22, no. 1, pp. 65–76.

Zhang, J.; Tanaka, M.; Endo, M. (2004): Meshless analysis of potential problems in three dimensions with the hybrid boundary node method. *International Journal for Numerical Methods in Engineering.*, vol. 59, pp. 1147–1166.

Zhang, J.; Tanaka, M.; Endo, M. (2005): The hybrid boundary node method accelerated by fast multipole expansion technique for 3D potential problems. *International Journal for Numerical Methods in Engineering.*, vol. 63, pp. 660–680.