 CMES, vol.68, no.1, pp.95-112, 2010

Meshless Solution of Potential Problems by Combining Radial Basis Functions and Tensor Product ones

Annamaria Mazzia[1] **and Flavio Sartoretto**[2]

Abstract: Meshless methods for the solution of Partial Differential Equations receive nowadays increasing attention. Many meshless strategies have been proposed. The majority of meshless variational methods one can find in the literature, use Radial Basis Functions (RBF) as generators of suitable trial and test spaces. One of the main problems encountered when exploiting RBF is performing numerical integrations over circles (when 2D problems are attacked, spheres for 3D ones). We exploit Tensor Product Functions (TPF) as the test function space. This strategy allows one to consider rectangular integration domains, which are much easier to manage. This paper numerically analyzes the effectiveness in solving potential problems of various settings for trial and test functions. Finally, the accuracy of our best choice method is analyzed, when using both uniform and pseudo–random meshes.

Keywords: Meshless Methods, Poisson Problem, Mixed Boundary Conditions, Moving Least Squares, Radial Basis Functions, Tensor Product Functions.

1 Introduction

Overcoming Finite Element (FE) methods is one of the keywords for improving both accuracy and efficiency in the numerical solution of (some types of) Partial Differential Equations (PDE). Nowadays *meshless methods* provide increasing attractive techniques in this direction [Atluri (2004); Babuska, Banerjee, and Osborn (2004); Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Fries and Matthies (2004); Nguyen, Rabczuk, Bordas, and Duflot (2008)].

We focus our attention on the so–called *true* Meshless methods, which do not exploit any mesh for discretizing the problem domain. As a byproduct, *true* meshless methods are more apt to implement adaptivity. Among these methods, the truly

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

[2] Dipartimento di Informatica Università di Venezia, Italy, flavio.sartoretto@unive.it

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, a PDE can be solved in its 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)]. In the literature one finds many implementations relying upon Radial Basis Functions (RBF), see e.g. [Atluri (2004)], and the references herein. A smaller number of papers exploiting Tensor Product Functions (TPF) is also available [Ni, Ho, Yang, and Ni (2004); Duflot and Nguyen-Dang (2002); Sun, Wang, and Miao (2008); Liu (2009); Sterk and Trobec (2008)].

The advantages of meshless methods are peculiarly dominant when 3D problems are considered [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); Mazzia, Pini, and Sartoretto (2008)]. On the other hand, tuning and analysis of meshless methods is likely to be started using test 2D problems. When attacking 2D problems, RBF–based meshless techniques require the evaluation of integrals on circular domains, which is cumbersome [Mazzia and Pini (2010)].

In this paper we analyze the accuracy of TPF and RBF based MLPG techniques, in the solution of potential problems. Our approach drives one to deal with rectangular integration domains, in place of circles, hence allowing easier and more accurate quadrature to be performed, thus curing one of the main drawbacks of pure RBF–based meshless methods, i.e. accurate and efficient evaluation of the numerical integrals.

We numerically analyze the accuracy and convergence of meshless methods which combine RBF and TPF functions. We use both uniform and pseudo–random, fine meshes. We show that accurate results can be attained by devising appropriate strategies and performing suitable parameter tuning.

2 Meshless schemes

Let us consider the linear 2D Poisson equation on the domain Ω

$$-\nabla^2 u = f, \tag{1}$$

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

$$u = \overline{u} \quad \text{on } \Gamma_u, \quad \frac{\partial u}{\partial \underline{n}} \equiv q = \overline{q} \quad \text{on } \Gamma_q \tag{2}$$

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

Solving Poisson problem using a weighted residual method, is equivalent to finding that function $u \in \mathscr{S}$, $\mathscr{S}$ being a suitable *trial* space, such that for *every test function* $\psi \in \mathscr{T}$, where $\mathscr{T}$ is a suitable test space, one has

$$\int_\Omega (\nabla^2 u + f)\,\psi \,\mathrm{d}\Omega = 0, \tag{3}$$

where u is the trial function and ψ is a test function.

The final Ritz–Petrov–Galerkin approach relies upon restricting conditions (3) to suitable finite–dimensional trial and test spaces

$$\mathscr{B} = \mathrm{span}\{\phi_1, \ldots, \phi_{N_B}\} \subset \mathscr{S}, \quad \mathscr{U} = \mathrm{span}\{\psi_1, \ldots, \psi_{N_T}\} \subset \mathscr{T}.$$

For numerical treatment, the problem domain is discretized by a set of N nodes. To each mesh node, P, 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 (ϕ) and test (ψ) 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. Using the divergence theorem and imposing Neumann boundary conditions, the LSWF pertaining to the i-th node can be written

$$\int_{\Gamma_I^{(i)}} q\,\psi_i \,\mathrm{d}\Gamma + \int_{\Gamma_u^{(i)}} q\,\psi_i \,\mathrm{d}\Gamma + \int_{\Gamma_q^{(i)}} \overline{q}\,\psi_i \,\mathrm{d}\Gamma -$$
$$\int_{\Omega^{(i)}} \left(\frac{\partial u}{\partial x}\frac{\partial \psi_i}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial \psi_i}{\partial y} + f\,\psi_i \right) \mathrm{d}\Omega = 0$$

This form is *symmetric* in the sense that both the trial and the test functions have equal order of differentiability requirements. We have $\Omega^{(i)} = \mathrm{supp}(\psi_i)$, $\Gamma_u^{(i)} = \Omega^{(i)} \cap \Gamma_u$ is the intersection of our local integration domain with *Dirichlet* boundary pieces. Analogously, $\Gamma_q^{(i)} = \Omega^{(i)} \cap \Gamma_q$ is the intersection of our local integration domain with *Neumann* boundary pieces. Eventually, $\Gamma_I^{(i)} = \partial\Omega^{(i)} \backslash (\Gamma_u^{(i)} \cup \Gamma_q^{(i)})$, is the portion of our local domain boundary, $\partial\Omega^{(i)}$, lying inside Ω. Recall that, being $\Omega^{(i)} = \mathrm{supp}(\psi_i)$, ψ_i vanishes on $\Gamma_I^{(i)}$.

Since each LSWF is prescribed over node $\underline{z}_i = (x_i, y_i)$, $i = 1, \ldots, N$, we obtain as many equations as the number N of nodes in the global domain. The ensuing linear

system of equations is $K\underline{\hat{u}} = \underline{f}$ where

$$K_{ij} = \int_{\Omega^{(i)}} \left(\frac{\partial \Phi_j}{\partial x}\frac{\partial \psi_i}{\partial x} + \frac{\partial \Phi_j}{\partial y}\frac{\partial \psi_i}{\partial y} \right) \mathrm{d}\Omega - \int_{\Gamma_u^{(i)} \cup \Gamma_I^{(i)}} \left(\frac{\partial \Phi_j}{\partial x} n_x + \frac{\partial \Phi_j}{\partial y} n_y \right) \psi_i \, \mathrm{d}\Gamma, \quad (4)$$

$$i, j = 1, \ldots, n$$

$$f_i = \int_{\Gamma_q^{(i)}} \overline{q} \psi_i \, \mathrm{d}\Gamma + \int_{\Omega^{(i)}} f \, \psi_i \, \mathrm{d}\Omega, \qquad i = 1, \ldots, n \quad (5)$$

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

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

Incidentally note that, unlike in FE, our trial basis is *not* a cardinal one, i.e. $\phi_j(\underline{z}_i) \neq \delta_{ij}$ hold true, hence $\tilde{u}(\underline{z}_i) \neq \hat{u}_i$ 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, $\tilde{u}_i = \tilde{u}(\underline{z}_i)$, $i = 1, \ldots, N$.

When $\underline{z}_k = (x_k, y_k)$ is a Dirichlet node, the k-th equation in the linear system is replaced by $\sum_{j=1}^{N} \Phi_j(\underline{z}_k)\hat{u}_j = \underline{u}_k$, thus exactly enforcing Dirichlet conditions over Dirichlet nodes [Liu (2009); Fries and Matthies (2004)].

3 Trial and test spaces

In order to identify a suitable trial space, we exploit the Moving Least Square (MLS) technique with a set of suitable *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.

For simplicity, let us confine to 2D problems. Let $\underline{z} = (x, y)$ an arbitrary point in the plane. Assume a set of mesh nodes $\underline{z}_i = (x_i, y_i)$, $\quad i = 1, \ldots, N$, is given. Let us consider the *linear* MLS technique. It is an approximation method which after a given set of functions, S, computes a new basis set, S', which "better" approximates polynomials. Even if $1 \notin V = \mathrm{span}\{S\}$, one has $1 \in V' = \mathrm{span}\{S'\}$. Assume that we need to approximate the function $u = u(x, y)$. Let $\underline{p} = \underline{p}(x, y) = (1, x, y)^T$, i.e. the degree of $\underline{p}$ is $d = 1$, and the number of basis functions is $m = 3$; Let us define the $N \times m$ matrix

$$P = \begin{pmatrix} \underline{p}(\underline{z}_1)^T \\ \ldots \\ \underline{p}(\underline{z}_N)^T \end{pmatrix},$$

together with the $N \times N$ matrix

$$W(\underline{z}) = \mathrm{diag}(w_i(\underline{z})), \quad w_i(\underline{z}) \geq 0, \quad i = 1, \ldots, N.$$

The functions $w_i(\underline{z})$ are called the MLS *weights*.

Let $\underline{u} = (u(\underline{z}_1), \ldots, u(\underline{z}_N))^T$, $A(x,y) = P^T W P$. The linear MLS technique minimizes $\underline{a} = \underline{a}(\underline{z}) = (a_j(\underline{z}))^T$, $j = 1,2,3$, i.e. on all linear polynomials in $\underline{z} = (x,y)$, the functional

$$J(\underline{z}) = \sum_{I=1}^{N} (\underline{p}^T(\underline{z}_I) \cdot \underline{a}(\underline{z}) - u(\underline{z}_I))^2 W(\underline{z}) = (P\,\underline{a}(\underline{z}) - \underline{u})^T W(\underline{z}) (P\,\underline{a}(\underline{z}) - \underline{u}) \geq 0.$$

By defining

$$\Phi(\underline{z}) = W(\underline{z})\, P(\underline{z})\, A^{-1}(\underline{z})\, \underline{p}(\underline{z}), \tag{6}$$

the minimization of the functional gives the approximation [Lancaster and Salkauskas (1981)]

$$\hat{u}(\underline{z}) = \Phi^T(\underline{z})\, \underline{u} = \sum_{i=1}^{N} \phi_i(\underline{z})\, u(\underline{z}_i).$$

Let $\Phi = (\phi_1, \ldots, \phi_N)$; the functions ϕ_i are called the MLS *shape* functions [Atluri (2004)].

Note that, after eq. (6), one has $\phi_i(\underline{z}) = w_i(\underline{z})\, (P A^{-1}\, \underline{p})_i$, hence the support of each *shape* function, ϕ_i, is enclosed into the support of its related *weight* function, w_i.

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 $V = \{\phi_j(\underline{z}_i),$ $i, j = 1, \ldots, N\}$. Hence *recovering* requires further computations compared to merely solving the linear system, which provides the fictitious values only.

Usually, RBF [Buhmann (2003)] are used as the MLS weights. In this paper, we exploit both TPF, see e.g. [Duflot and Nguyen-Dang (2002); Ni, Ho, Yang, and Ni (2004); Sun, Wang, and Miao (2008)], and more usual RBF, as the weights. Concerning TPF, assume $g^{(\eta)}(t)$ is a given, differentiable function, η being a parameter which controls the amplitude of its support. When dealing with RBF, we shall denote the "radius" as precisely the radius of its circular domain. When dealing with TPF functions, we denote the x–"radius" as half the length of the x–side, and analogously for the y–"radius". In this paper, we shall use equal x– and y– values, hence we shall speak of "the" TPF radius (see the sequel for further details). To each node, $\underline{z}_i = (x_i, y_i)$, a TPF is associated by

$$w_i(x,y) = g^{(\eta_i^{(x)})}(|x - x_i|) \cdot g^{(\eta_i^{(y)})}(|y - y_i|), \tag{7}$$

by suitably choosing the two x- and y- factors (see the sequel for details).

In order to fully specify our meshless procedure, we need to further identify suitable *test* functions. Results in our previous papers [Mazzia, Pini, and Sartoretto (2008); Mazzia and Pini (2010)], rely upon RBF weights for MLS and RBF of the same type, as the test functions. In this paper we analyze the advantages of using TPF as the test function space generators.

3.1 Generating functions

MLS weights and test spaces were obtained by using three types of "generating" functions.

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

$$S(t,\eta)=\begin{cases}1-6\left(\frac{t}{\eta}\right)^2+8\left(\frac{t}{\eta}\right)^3-3\left(\frac{t}{\eta}\right)^4, & 0\le t\le\eta,\\ 0, & t\ge\eta.\end{cases} \tag{8}$$

We used Gaussian generators [Lu, Belytschko, and Gu (1994)]

$$G(t,\eta)=\begin{cases}\dfrac{\exp\left[-\left(\frac{t}{c}\right)^{2k}\right]-\exp\left[-\left(\frac{\eta}{c}\right)^{2k}\right]}{1-\exp\left[-\left(\frac{\eta}{c}\right)^{2k}\right]}, & 0\le t\le\eta,\\ 0, & t\ge\eta.\end{cases} \tag{9}$$

As usual in literature, we set $k=1$. The parameter c controls the shape of the function. Following [Atluri (2004)] we set $c=\eta/4$.

Finally, we exploited [Liu (2009)]

$$F(t,\eta)=\begin{cases}1-(t/\eta)^2, & 0\le t\le\eta,\\ 0, & t\ge\eta.\end{cases} \tag{10}$$

For any mesh node $\underline{z}_i=(x_i,y_i)$ a TPF is obtained after eq. (7) by setting

$$g^{(\eta_i^{(x)})}(|x-x_i|)=\gamma(|x-x_i|,\eta_i^{(x)}),\quad g^{(\eta_i^{(y)})}(|y-y_i|)=\gamma(|y-y_i|,\eta_i^{(y)}),$$

where either $\gamma=S(t,\eta)$, or $\gamma=G(t,\eta)$, or $\gamma=F(t,\eta)$. The parameter $\eta_i^{(x)}$ is the support x–radius, while $\eta_i^{(y)}$ is the support y radius. In other terms, the support of $w_i(x,y)$ is a rectangle centered at $\underline{z}_i=(x_i,y_i)$, whose x–sides are $2\,\eta_i^{(x)}$ long, while the y–sides are $2\,\eta_i^{(y)}$ wide. Note that one could set $\eta_i^{(x)}\neq\eta_i^{(y)}$, hence obtaining

rectangular domains. For simplicity, in the sequel we assume $\eta_i^{(x)} = \eta_i^{(y)}$, $i = 1,\ldots,N$, thus square function supports are obtained.

On the other hand, a RBF associated to any node $\underline{z}_i$ is identified by suitably setting a domain radius r_i and considering

$$\gamma(\sqrt{(x-x_i)^2+(y-y_i)^2}), r_i),$$

where either $\gamma = S(t,\eta)$, or $\gamma = G(t,\eta)$, or $\gamma = F(t,\eta)$.

4 Test problems

Let us consider Poisson equation

$$-\nabla^2 u(x,y) = f(x,y), \quad (x,y) \in [0,1]^2, \tag{11}$$

and two test solutions.

- One is drawn after [Wagner and Liu (2001)],

  $$u(x,y) = [\cosh(\pi y) - \coth\pi \sinh(\pi y)] \sin(\pi x). \tag{12}$$

  It solves equation (11) with $f(x,y) = 0$ (Laplace equation). We consider three problems, whose solution is (12), by setting three types of boundary conditions.

  - Dirichlet boundary conditions on the whole boundary, problem labelled L_D (Laplace equation, Dirichlet boundary conditions).
  - Dirichlet boundary conditions on the horizontal sides, Neumann boundary conditions on the vertical ones (L_M, Laplace equation, Mixed conditions).
  - Dirichlet boundary conditions on the vertical sides, Neumann boundary conditions on the horizontal ones (L_{M_1}, Laplace equation, alternative Mixed conditions).

- Another test function is proposed in [Atluri and Zhu (1998)],

  $$u(x,y) = -(5/6)(x^3+y^3) + 3x^2y + 3xy^2. \tag{13}$$

  It solves Poisson equation (11) where $f(x,y) = x+y$. Assume Dirichlet boundary conditions on the whole boundary are set. Let us label this problem P_D, standing for Poisson equation with Dirichlet boundary conditions.

Let $u(x,y)$ be the exact solution of our problem, while $\tilde{u}(x,y)$ is the approximated solution estimated by our procedure.

Assume we have a mesh, $\mathscr{M}$, set up for error estimation, whose nodes are $\underline{z}_i$, $i = 1,\ldots,Q$. Our error estimator, drawn after [Atluri, Han, and Rajendran (2004)] is

$$e_{\mathscr{M}} = \left(\frac{\sum_{i=1}^{Q} (u(\underline{z}_i) - \tilde{u}(\underline{z}_i))^2}{\sum_{i=1}^{Q} u^2(\underline{z}_i)} \right)^{1/2}, \tag{14}$$

being $\underline{u} = (u(\underline{z}_1),\ldots,u(\underline{z}_Q))$, the vector of the exact nodal values.

5 Numerical results

5.1 Uniform grids

In order to analyze the main features of our Meshless procedure, we start focusing on *uniform grids* on $[0,1]^2$, called $G_1,\ldots,G_4$. Their diameters (uniform distance between nodes) are $h_1 = 0.125$, $h_{i+1} = h_i/2$, $i = 1,2,3$, respectively.

In the sequel, as our standard error measure we use $e_{\mathscr{M}}$, where $\mathscr{M}$ is a uniform grid with 68×68 nodes, i.e. a slightly finer grid than G_4.

Based upon numerical tuning (see the sequel), we set a pair of values, one for the *trial* function support *radius*, r, another for the *test* function support radius, ρ. All trial functions share the same support radius, and the same holds for all test functions, a valid choice since we exploit uniform grids.

Usually [Atluri (2004)] RBF are exploited as both the MLS weights, and the test functions. Our main idea is to exploit TPF as the generators of the test space. Since our algorithm performs integration based upon the test function domain, we obtain rectangular integration domains, in place of circular ones. Such an approach simplifies greatly the integrations. Numerical quadratures are performed by simple Gauss product rules, enrolling N_G quadrature points both in the x– and y–directions. We extensively investigated the importance of N_G. The values $N_G = 4,\ldots,12,18,20,32,64$ were tested.

Let us consider a TPF–TPF spline based method, i.e. TPF based upon splines of type (8) were exploited both as MLS weights, and test functions. Quasi–optimal r and ρ radiuses where determined by numerical experiments. Fig. 1 shows the error raised when attacking problem P_D. Note that convergence becomes poor when finer and finer grids are exploited. Analogous results we observed when TPF based upon Gaussian functions of type (9) are used (not shown for brevity). We guess that exploiting TPF as MLS weights produce poor interpolating spaces, hence poor accuracy.

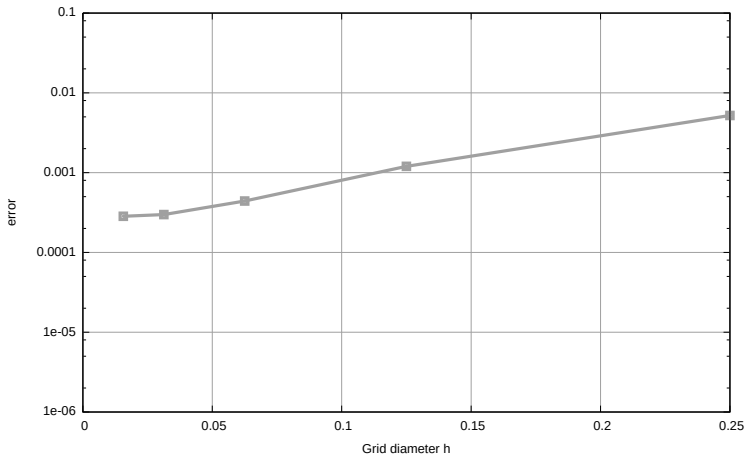

Figure 1: Errors vs grid diameter, h. P_D problem is solved with TPF–TPF method, using spline generators for both trial and test functions. Parameter values $\rho/h = 1.4$, $r/h = 2.8$ are set.

On the ground of these results, we decided to exploit RBF as the MLS weights, while the test function space is based upon TPF. In the sequel, we refer to this technique as "RBF–TPF". Recall that since integration is performed by our algorithm on the domains of the test function space, using TPF test functions produces square integration domains, which is more efficient than using RBF circular domains.

We found that effective generators for our TPF test functions are those in eq. (10). This choice is exploited for time–dependent problems in [Sterk and Trobec (2008)], being recommended in [Liu (2009)]. Using either spline or Gaussian generators, we obtained less accurate results.

On the ground of our results above, in the sequel we shall consider only methods whose test space is made by TPF functions based upon eq. (10).

Moreover, we only exploit RBF trial functions obtained using either spline or Gaussian weights. Let us compare the effectiveness of these two settings.

Fig. 2 shows the error behavior vs r/h, when spline generators are exploited, $\rho/h = 1$ is constant. One can see that the error behavior is quite erratic, hence the setting of r value is questionable. On the other hand, as r/h enlarges, the number of non–zero entries in the final linear system matrix increases (see Fig. 3), since larger interpolation domain radiuses implies a larger number of nodal contacts activated. In order to minimize the error, and maintain small the number of non–zero entries

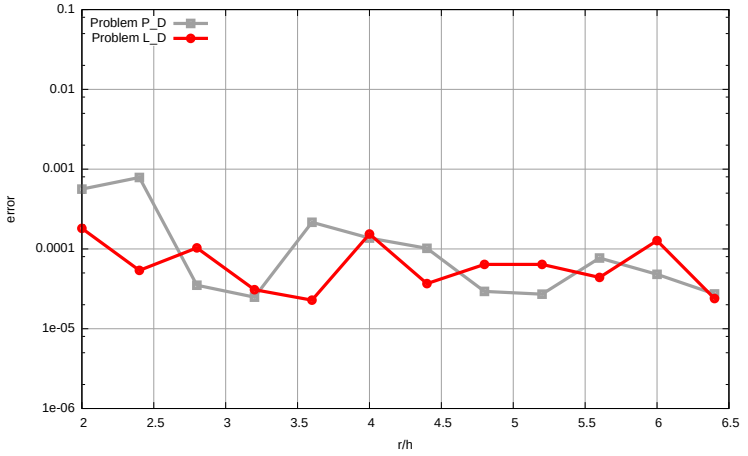

Figure 2: Errors vs r/h, when attacking P_D and L_D problems using spline weights for RBF trial functions. The domain is discretized by grid G_3, $\rho/h = 1$ is set.

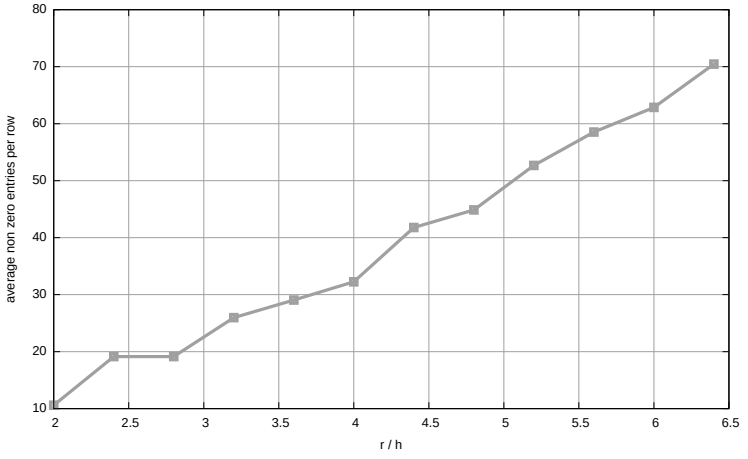

Figure 3: Fill-in vs r/h, when $\rho/h = 1$. Spline weights for RBF trial functions. The domain is discretized by grid G_3.

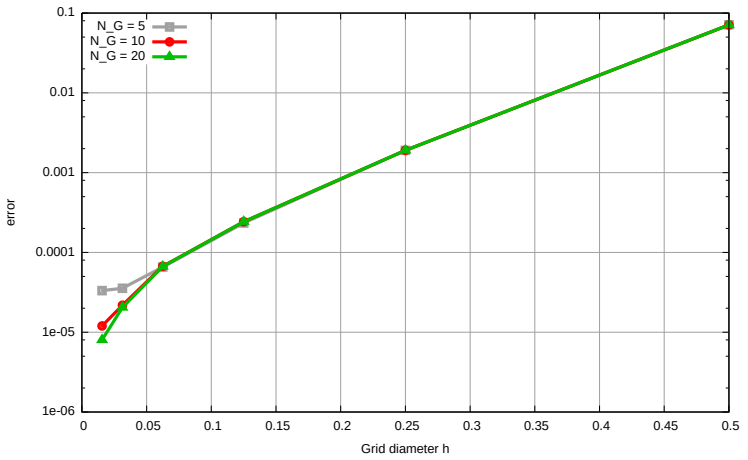

Figure 4: Error behavior vs grid diameter, when solving problem P_D. RBF trial functions are obtained by spline weights, $\rho/h = 1$, $r/h = 2.8$.

in the final system, we experimented with $r/h = 2.8$. This setting was confirmed also by our results on convergence, which are shown in the sequel.

Let us analyze the degree of dependence of our meshless technique upon the number of quadrature nodes, i.e. the accuracy of numerical integrations. Fig. 4 shows that using $N_G = 5$ the error on the finest grid is appreciably higher than with $N_G \geq 10$. By numerical experiments we found that this dependence on the number of integration nodes is not relevant when using Gaussian generators (not shown for brevity).

Fig. 5 shows that when $\rho/h = 1$, the error raised with Gaussian generators monotonically decreases when r/h increases. One can see that the error decreases slowly when $r/h > 4.5$. Taking into consideration that as r/h increases the fill–in in the final system increases in the identical manner shown by Fig. 3, we set as a suitable choice $r/h = 5$. Less erratic error behavior w.r.t. r/h, and very low dependence on the number of quadrature points suggest that Gaussian generators are preferable over splines.

Up to now, we considered only Dirichlet boundary conditions. Concerning mixed boundary conditions, Fig. 6 shows the errors raised when L_M problem is attacked. Errors marked with circles are obtained by integrating on all nodes with N_G quadrature points. Note that when $N_G = 8$ the error is quite large. Errors marked with squares are obtained by performing numerical integrations with $N_G = 2^k$, $k = 2,3,4,5$, on all nodes, except but on Neumann ones, where a more accurate quadrature rule,

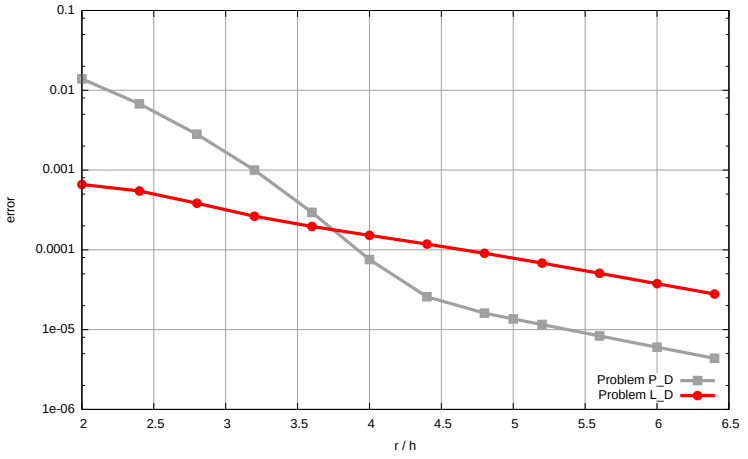

Figure 5: Errors vs r/h, when $\rho/h = 1$. Problems P_D and L_D are solved using RBF trial functions obtained by Gauss weights. Grid G_3 was exploited.

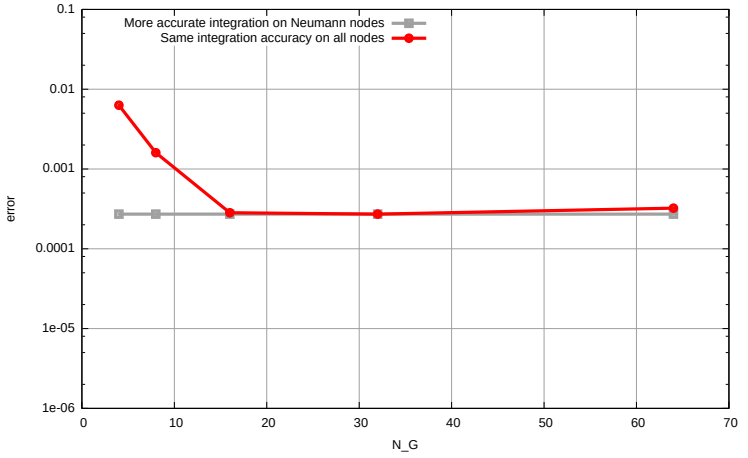

Figure 6: Treatment of mixed boundary conditions. Problem L_M. Circles show errors obtained when integrations on all nodes are performed by $N_G = 2^k$ nodes, $k = 2,3,4,5$. Squares mark errors raised when only on Neumann nodes integrations are performed by more accurate N_G=32 quadrature formula. Grid G_3 was used, $r/h = 5$, $\rho/h = 1$ was set.

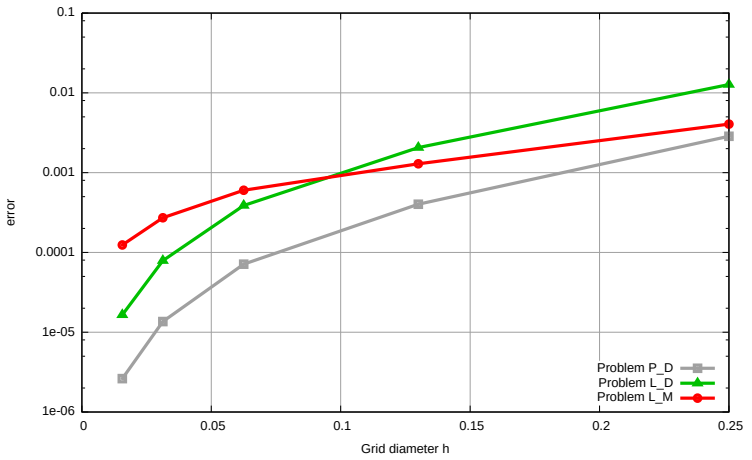

Figure 7: Errors for P_D, L_D, L_M problems. The error curve for problem L_{M_1} overlaps that one for L_M. Parameter values are: $r/h = 5$, $\rho/h = 1$, $N_G = 5$.

using $N_G = 32$ integration points was exploited. Inspecting Fig. 6, one could argue that $N_G = 16$ could be enough, but our extensive numerical experiments suggest that in order to attain convergence with fine grids, $N_G = 32$ is likely to be set. Summarizing, one can obtain accurate results by integrating on non–Neumann nodes with a small number of quadrature points, but Neumann nodes require a quite large number of quadrature points.

Figure 7 shows the error behavior when solving our test problems. One can see that the attained accuracy is satisfactory.

5.2 Pseudo–random meshes

Now let us perturb our grids G_i, $i = 1,2,3,4$, in order to analyze the performance of our method on (pseudo–)random meshes. More precisely, for each grid G_i, nodes on the domain boundary are not moved, in order to avoid modifying the domain. On the other hand, each internal node is either left unchanged, or uniformly random shifted $h_i/2$ either up, or down, or left or right. Fig. 8 shows a cloud of points that were obtained by shifting the nodes in our uniform grid G_4. Using this perturbation strategy, we obtained pseudo–random meshes G_i', $i = 1,2,3,4$.

In the sequel, we focus on our best–choice RBF–TPF algorithm for uniform meshes. Its RBF trial functions come from Gaussian weights, while the test functions are TPF obtained by functions of type (10).

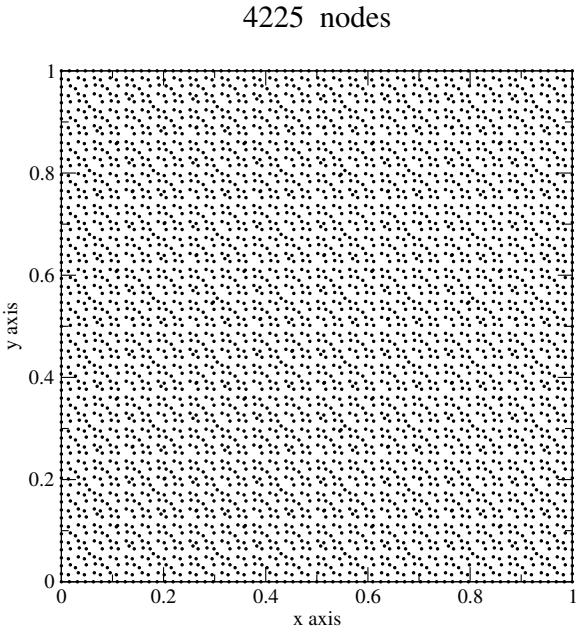

Figure 8: Pseudo–random mesh obtained by perturbing grid G_3.

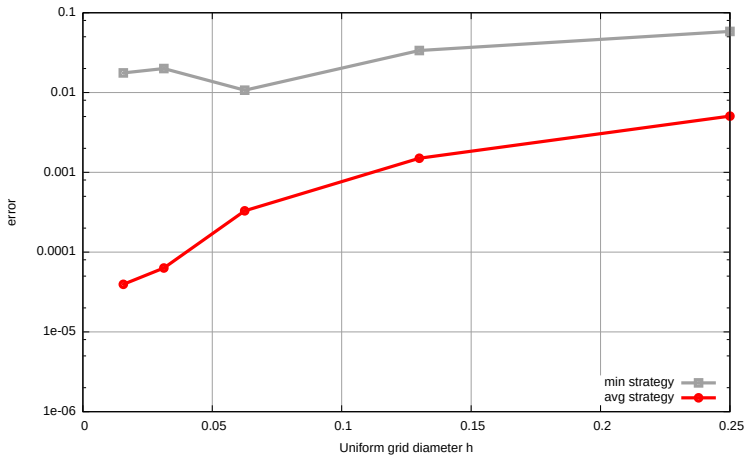

Figure 9: Error on random grids for problem P_D, when either the "min" or the "avg" strategies are exploited. For each random grid, on the x- axis the diameter of the corresponding unperturbed grid is reported.

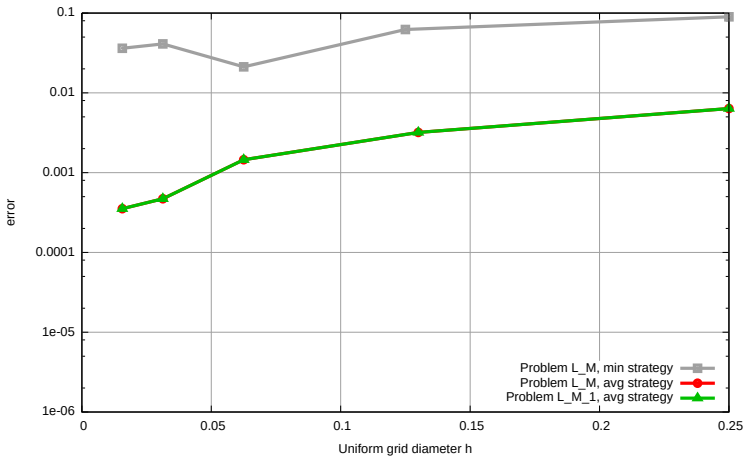

Figure 10: Analogous to the previous Figure, apply to problem L_M. The error for the L_{M_1} problem with "avg" strategy equals that one for L_M problem.

Let us consider our Poisson problem with Dirichlet boundary conditions. When we used uniform grids, we set one trial domain radius for all trial functions, depending upon grid diameters, and one for all test functions. Now we must devise new strategies, which assign a peculiar domain radius to each trial and test function. By numerical experiments, we analyzed many strategies. Here, we describe two of them, both seeming sound procedures, one which proved unsuccessful, another which proved effective. The "min" strategy relies upon assigning as the domain radius of a function (either trial or test) associated to node n, the distance of the nearest mesh point to node n. Incidentally, this is the strategy that we exploited for uniform meshes. The "avg" strategy assigns the average distance in the six nearest points to n. Fig. 9 shows the errors raised by either "min" or "avg" strategy. One can see that using "min" strategy, the error is quite large and not monotonically decreasing with h. On the opposite, "avg" strategy produces a satisfactory small, monotonically decreasing error.

We guess that the radiuses produced by "min" strategy usually result to be too small. It is not guaranteed that the union of the supports of the trial functions covers the entire problem domain, and the same holds for the supports of the test functions. Such an eventuality makes convergence either poor or even prevented.

Let us consider our Laplace problem with mixed boundary conditions. Fig. 10 shows the errors raised using a pseudo–random mesh G_i'. On the x–axis, the radius of the corresponding unperturbed grid G_i, is reported. The results for problem L_{M_1}

superimpose those for L_M, suggesting that modifying the position of the Neumann boundary pieces does not affect the accuracy. This check is worth performing, since test functions are TPF based, and TPF are not independent of rotations, wherever RBF are.

By comparing Fig. 9 and Fig. 10 with Fig. 7, one can see that satisfying accuracy is obtained with non uniform meshes, though it is slightly smaller than the one obtained using the unperturbed grids.

The error behavior of Laplace problem with Dirichlet boundary conditions is not displayed, since it is analogous to those already shown.

6 Conclusions

We analyzed the performance of RBF–TPF–based meshless algorithms in the solution of Poisson problems. The following results are worth emphasizing.

- Using TPF–based trial space does not provide a sufficiently accurate interpolating space. Implementing a solution scheme relying upon an RBF–based *trial* space combined with a TPF *test* space, preserves the rectangular shape for integration domains, yet exploiting a better interpolation space.

- Concerning trial functions, our numerical experiments suggest that Gaussian generators provide higher accuracy than splines.

- Our RBF–TPF based meshless algorithm provides an effective solution to sample potential problems, for both Dirichlet and mixed boundary conditions. In the latter case, good accuracy at a reduced computational cost can be attained by using accurate quadrature rules only on Neumann nodes.

- Use of test spaces based upon TPF, in place of RBF, reduces the criticality of numerical quadrature accuracy. When Dirichlet boundary conditions are set, quite the same errors can be obtained when reducing the number of quadrature nodes from 64×64 to 4×4. When mixed boundary conditions are considered, in order to achieve good accuracy, a small number of quadrature nodes is allowed on mesh points, except but on Neumann nodes, where a large number of integration points is required.

Acknowledgement: This work was partially funded by Italian MIUR project (PRIN) “Advanced numerical methods and models for environmental fluid–dynamics and geomechanics”.

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. *CMES: Computer Modeling in Engineering and Sciences*, vol. 6, no. 6, pp. 491–513.

Atluri, S. N.; Liu, H. T.; Han, Z. D. (2006): Meshless Local Petrov-Galerkin (MLPG) Mixed Collocation method for elasticity problems. *CMES: 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. *CMES: 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. *CMES: 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.

Buhmann, M. D. (2003): *Radial Basis Functions: Theory and Implementation*, volume 12 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge.

Duflot, M.; Nguyen-Dang, H. (2002): A truly meshless method based on a moving least squares quadrature. *Commun. Numer. Methods Eng.*, vol. 18, pp. 441–449.

Fries, T.-P.; Matthies, H.-G. (2004): Classfication and overview of meshfree methods. Technical Report 2003-3, Technical University Braunschweig, Brunswick, Germany, 2004.

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

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. *CMES: Computer Modeling in Engineering and Sciences*, vol. 4, no. 5, pp. 571–585.

Liu, G. R. (2009): *Meshfree Methods: Moving Beyond the Finite Element Method*. CRC Press, second edition.

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.; Pini, G. (2010): Product Gauss quadrature rules vs cubature rules in the meshless local Petrov-Galerkin method. *Journal of Complexity*, vol. 26, pp. 82–101.

Mazzia, A.; Pini, G.; Sartoretto, F. (2008): Accurate MLPG solution for 3D potential problems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 36, no. 1, pp. 43–63.

Nguyen, V. P.; Rabczuk, T.; Bordas, S.; Duflot, M. (2008): Meshless methods: A review and computer implementation aspects. *Mathematics and Computers in Simulation*, vol. 79, pp. 763–813.

Ni, G.; Ho, S. L.; Yang, S.; Ni, P. (2004): Meshless local Petrov-Galerkin method and its application to electromagnetic field computations. *International Journal of Applied Electromagnetics and Mechanics*, vol. 19, no. 1, pp. 111–117.

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.

Sterk, M.; Trobec, R. (2008): Meshless solution of a diffusion equation with parameter optimization and error analysis. *Engineering Analysis with Boundary Elements*, vol. 32, pp. 567–577.

Sun, H.; Wang, Y.; Miao, Y. (2008): Meshless numerical method based on tensor product. *Front. Archit. Civ. Eng. China*, vol. 2, no. 2, pp. 166–171.

Wagner, G. J.; Liu, W. K. (2001): Hierarchical enrichment for bridging scales and mesh-free boundary conditions. *Int. J. Numer. Methods Eng.*, vol. 50, pp. 507–524.

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