 CMES, vol.102, no.6, pp.475-497, 2014

MLPG Refinement Techniques for 2D and 3D Diffusion Problems

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

Abstract: Meshless Local Petrov Galerkin (MLPG) methods are pure meshless techniques for solving Partial Differential Equations. One of pure meshless methods main applications is for implementing Adaptive Discretization Techniques. In this paper, we describe our fresh node–wise refinement technique, based upon estimations of the "local" Total Variation of the approximating function. We numerically analyze the accuracy and efficiency of our MLPG–based refinement. Solutions to test Poisson problems are approximated, which undergo large variations inside small portions of the domain. We show that 2D problems can be accurately solved. The gain in accuracy with respect to uniform discretizations is shown to be appreciable. By extending our procedure to 3D problems, we prove by experiments that good improvements in efficiency can be obtained.

Keywords: Meshless Methods, MLPG, Refinement, Diffusion.

1 Introduction

Meshless methods supply nowadays a good alternative over Finite Element (FE) methods in many real–world problems, e.g. when meshing is a time–consuming and cumbersome task [Gerace, Erhart, Kassab, and Divo (2013)].

Meshless Petrov Galerkin (MLPG) methods [Atluri (2004)] are true meshless, widely used methods.

In this paper we report about our numerical experiments on devising and implementing MLPG–based, refinement strategies for solving diffusion problems.

Adaptive discretization strategies for classical Finite Element (FE) methods were proposed since this method was in its infancy, e.g. by Sewell (1972). Among the devised strategies, the well explored Zienkiewicz-Zhu Superconvergent Patch

[1] Dipartimento ICEA Università di Padova Via Trieste 63, 35121 Padova, Italy.
E-mail: {annamaria.mazzia,giorgio.pini}@unipd.it
[2] DAIS, Università Ca' Foscari Venezia Via Torino 155, 10173 Venezia, Italy.
E-mail: flavio.sartoretto@unive.it

Recovery (ZZ-SPR) [Zienkiewicz and Zhu (1992)] works for high order FEM and structural problems (relatively small number of nodes in meshes) [Babuska, Strouboulis, and Upadhyay (1997)]. In our work, we deal with non–structural problems, requiring a large number of discretization nodes. ZZ-SPR is not apt in such context.

Raising the degree p of FE elements, called p-refinement, is an adaptive strategy used in structural mechanics. Recently, fracture treatment with XFEM were proposed via enrichment of the trial basis [Gordeliy and Peirce (2013)]. Such strategy is not apt to MLPG methods, were trial functions are usually non–poynomial ones [Atluri (2004)].

The most common adaptive refinement strategy called h-refinement is performed by reducing the average nodal spacing in areas of high gradient.

Combination of the two strategies, the so–called $h-p$–methods, were proposed for FE methods [Schwab (1998)]. Dealing with meshless methods, we restrict to h–refinements.

Micchelli (1986) demonstrated that multiquadric surface interpolation is always solvable, for distinct data sets. Although any grid may be used, experience shows that different node distributions produce different results. Therefore, a given global error can be obtained with different number of nodes and positions, as pointed out by Roque, Madeirab, and Ferreira (2014).

Babuska, Strouboulis, and Upadhyay (1997) introduced the notions of local error and pollution error. They point out that local estimations do not account for errors. That is why one labels the values used to trigger refinement/coarsening of meshes as error "indicators", rather than "estimators". Babuska, Strouboulis, and Upadhyay (1997) state that benchmarks are not sufficient to devise better error indicators: local measures do not take into consideration pollution (i.e. global, near–field) error. On the other hand numerical experiments are the only key to attain meshless methods improvement, since no comprehensive convergence theory is available for them. On the other hand, Babuska himself, corroborated by his co–authors, states in [Babuska, Banerjee, and Osborn (2005)] that non–standard numerical methods for the solution of differential equations are often based on heuristic ideas, and verified by numerical experiments. Along this way, in this paper we provide efficiency estimations of our refinement indicators, adapted after Babuska, Strouboulis, and Upadhyay (1997).

Verfürth (2013) suggests that the main goal of adaptive methods is to "place more grid-points where the solution is less regular". Regularity is an analytical property which is hardly well represented by numerical methods. Since we deal with regular solutions, we guess that "less regular" should be changed into "more twist-

ing". Working on this idea, we use the Total Variation, an indicator for refining a discretization.

This paper is organized as follows. Section 2 summarizes main concepts about MLPG techniques. Section 3 describes our refinement procedure and attached issues. Section 4 gives detailed information about suitable test problems. Section 5 describes and anlyzes distinguished numerical results. Section 6 draws our main conclusions.

2 Meshless techniques

2.1 Weak formulation

Let us consider the dimensionless steady–state diffusion equation on the domain Ω

$$-\nabla \cdot \nabla u(\underline{x}) = f(\underline{x}), \tag{1}$$

where f is a given source function, $\underline{x}$ being any point in Ω.

Dirichlet and Neumann boundary conditions are imposed on the domain boundary $\partial\Omega = \Gamma$:

$$\begin{aligned} &u = \bar{u} \qquad \text{on} \quad \Gamma_u, \\ &\nabla u \cdot \underline{n} \equiv q = \bar{q} \quad \text{on} \quad \Gamma_q, \end{aligned} \tag{2}$$

being $\Gamma_q \cup \Gamma_u = \Gamma$, $\Gamma_q \cap \Gamma_u = \emptyset$; the vector $\underline{n}$ is the outward unit normal to Γ.

In order to approximate the solution of our problem, a set of discretization nodes $\underline{x}_i$, $i = 1, \ldots, N$, must be given. A set of trial functions ϕ_i, and a set of test functions τ_i are enrolled, each one being "centered" on node $\underline{x}_i$.

The approximation has the form

$$\tilde{u}(\underline{x}) = \sum_{i=1}^{N} \hat{u}_i \, \phi_i(\underline{x}). \tag{3}$$

Let us set apart for a moment the problem of imposing boundary conditions. The MLPG formulation relies upon restricting to a suitable piece of subdomain $\Omega_i \subset \Omega$, whose boundary is $\Gamma_i = \partial\Omega_i$, the standard, weak approximation of the weighted eq. (1). One obtains the set of equations

$$\int_{\Omega_i} (\nabla \tilde{u}) \cdot (\nabla \tau_i) \, d\Omega - \int_{\Gamma_i^{(u)}} (\nabla \tilde{u} \cdot \underline{n}) \, \tau_i \, d\Gamma = \int_{\Omega_i} f \, \tau_i \, d\Omega + \int_{\Gamma_i^{(q)}} \bar{q} \, \tau_i \, d\Gamma, \quad i = 1, \ldots, N.$$

These equations are called Local Weak Forms (LWF). The piece of subdomain boundary $\Gamma_i^{(u)} = \Gamma_i \cap \Gamma_u$ is the intersection of the i-th local integration domain

boundary with *Dirichlet* boundary. Analogously, $\Gamma_i^{(q)} = \Gamma_i \cap \Gamma_q$ is the intersection of the i-th local integration domain boundary with *Neumann* boundary. Integrals on $\Gamma_i \backslash (\Gamma_i^{(u)} \cup \Gamma_i^{(q)})$, the portion of Γ_i lying inside Ω give null contribution, since for simplicity, without diminishing the range of applicability of our methods, we assume $\tau_i = 0$ on Γ_i.

We use MLS approach in order to interpolate u. The functions $\phi_i(\underline{x})$ are the MLS "shape" functions [Lancaster and Salkauskas (1981)]. We exploited either quadratic (identified in the sequel by $B = 2$) or cubic polynomial ($B = 3$) basis [Mazzia and Sartoretto (2010)]. Recall that MLPG unknowns are the coefficients $\hat{u}_i$ of the linear combination (3). Now, MLS shape functions $\phi_i(\underline{x})$, are not interpolating, i.e. they do not possess the Kronecker delta property $\phi_i(\underline{x}_j) = \delta_{ij}$ [Fries and Matthies (2004)]. In order to obtain the $\tilde{u}(\underline{x}_i)$ approximation values, a recovery step is necessary, e.g. by applying an MLS reconstruction.

The boundary conditions on a Dirichlet node $\underline{x}_i$ are set by replacing the j-th LWF in (2.1) with $\tilde{u}_i = u(\underline{x}_i)$. On the other hand, when $\underline{x}_i$ is a Neumann boundary node, the i-th equation in system (2.1) is obtained by imposing the corresponding Neumann conditions via $\tilde{u}_i$ values.

There are many Meshless Petrov–Galerkin (MLPG) methods [Atluri (2004); Fries and Matthies (2004)], each one is identified by a peculiar pair of trial and test function spaces. Following our previous works (Mazzia, Pini, and Sartoretto, 2012; Mazzia and Sartoretto, 2010), we consider MLPG methods where the trial functions are MLS shape functions generated by suitable Radial Basis Functions (RBF), while the test functions are Tensor Product Functions (TPF) (see the sequel).

2.2 Finite dimensional spaces

Each MLS weight function associated to a discretization node is obtained by using a single 1D, "generator" function. Assume $g_1(t)$ is a given, 1D differentiable, compact supported, generator function. A RBF associated to node $\underline{x}_i$ is devised by suitably setting a support radius r_i and considering the ensuing function $g_1\left(\frac{\|\underline{x}-\underline{x}_i\|_2}{r_i}\right)$. We performed many numerical experiments by using several types of 1D generator functions. We considered cubic and quartic spline, and exponential functions, like in [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)]. Our numerical results suggested that the best effectiveness is obtained by the Gaussian generator after [Lu, Belytschko, and Gu (1994)], i.e.

$$g_1(t) = \begin{cases} \dfrac{\exp(-(\sigma t)^2) - \exp(-\sigma^2)}{1 - \exp(-\sigma^2)} & 0 \le t \le 1 \\ 0 & t \ge 1, \end{cases} \tag{4}$$

where σ is a parameter controlling the function shape. In the sequel, we assume $\sigma = 4$.

Concerning test functions, we exploited polynomial generators after [Liu (2009)]

$$g_2(t) = \begin{cases} 1-t^2, & 0 \le t \le 1, \\ 0, & t \ge 1. \end{cases} \tag{5}$$

To each node, $\underline{x}_i$, we associate the TPF

$$\tau_i(x,y) = g_2(|x-x_i|/\eta_i^{(x)}) \cdot g_2(|y-y_i|/\eta_i^{(y)}), \tag{6}$$

by suitably choosing the $\eta_i^{(*)}$ factors. For simplicity, we assume $\eta_i^{(x)} = \eta_i^{(y)} = \bar{\eta}_i$, hence the support of $\tau_i(\underline{x})$ is a square centered at $\underline{x}_i$, whose "radius" (half side–length) is η_i.

2.3 Radiuses identification

A crucial step for obtaining an accurate MLPG scheme is the identification of the trial basis radiuses r_i, and the test basis ones ρ_i, $i = 1, \ldots, N$ [Mazzia and Sartoretto (2010)].

For each node we sort in ascending order its distances from its $n^{(N)}$, neighbors, $n^{(N)}$ integer value must be identified.

Assume an irregular discretization I is given. For each node $\underline{x}_i$ in I we compute the $n^{(N)} = 7$ closest nodes to $\underline{x}_i$, and we sort their distances in ascending order $d_i^{(1)} \le d_i^{(2)} \le \ldots \le d_i^{(7)}$.

(a) If $d_k = d, k = 1, \ldots, 4$, we assume that the node distribution around the i-th node is "practically uniform". We set $r_i = \beta d$, $\rho_i = \alpha d$, α, β being parameters to be tuned. See the sequel for details.

(b) Otherwise we set

$$r_i = \beta d_i^{(7)}, \quad \rho_i = \alpha d_i^{(7)}. \tag{7}$$

The parameters α and β were identified by numerical experiments: They fall in not too large ranges [Mazzia and Sartoretto (2010)]. Typical intervals are

$$0.5 \le \alpha \le 1.5, \quad 1.0 \le \beta \le 5.0, \quad \alpha < \beta. \tag{8}$$

Throughout this paper we assume $\alpha = 1$. This setting proved suitable for all test problems shown in the sequel.

Concerning 3D problems, we perform step (a) on six points, in place of four, i.e. "local uniformity" corresponds to $d_i^{(k)} = d$ for $k = 1, \ldots, 6$. Note that there are six faces in a square box centered on any given point $\underline{x}_i$. Else, we perform step (b) assuming $n^{(N)} = 9$. This last parameter was identified by extensive numerical experiments.

3 Discretizations

3.1 Introduction

Assume for simplicity that our domain is either the $[0,1]^2$ square, for 2D problems, or the $[0,1]^3$ cube for 3D problems.

When dealing with 2D problems, let us start with a uniform discretization obtained by evenly dividing the x- and y-side into $n_x = n_y = 4$ parts. Let us call this uniform discretization U_1, the level $\ell = 1$ discretization. The interval spacing is $h_1 = 1/4$. Each finer, uniform discretization level is obtained by halving each subinterval.

3D uniform discretizations are obtained in a similar manner, by setting $n_x = n_y = n_z = 4$ at the initial level $\ell = 1$, $h_1 = 1/4$. Each finer level is obtained by halving each subinterval.

Our refinement strategy builds irregular discretizations, whose effectivity for MLPG computations is to be tested. Let I be a given one, consisting of N nodes, $\underline{x}_1, \ldots, \underline{x}_N$. When any irregular discretization is considered, the following two measures are worth considering. They are the fill distance $h_{I,\Omega}$, and the separation distance, q_I, i.e. [Fasshauer (2007)]

$$h_{I,\Omega} = \sup_{\underline{x} \in \Omega} \min_{1 \le i \le N} \|\underline{x} - \underline{x}_i\|, \quad q_I = \frac{1}{2} \min_{k \neq i} \|\underline{x}_k - \underline{x}_i\|, \quad 1 \le k, i \le N.$$

3.2 Refinement strategy

Assume an N–node irregular discretization I was computed.

Let us focus on 2D problems. In order to refine the discretization, we consider on each node $\underline{x}_i$ the "local" Total Variation (TV) of the solution u, defined as

$$\|u\|_{TV,i} = \|\nabla u\|_{1,i} = \int\int_{\Omega_i} (|u_x| + |u_y|)\, d\Omega,$$

where u_x and u_y are the partial derivatives of the solution. The advantages of this "variation measure" are clearly pointed out by Strang (2007): The variational methods which use the TV norm "allow for discontinuities but disfavor oscillations".

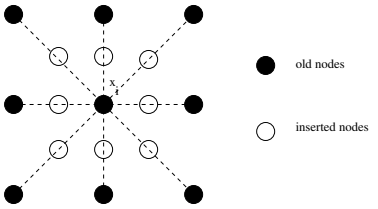


Figure 1: Node insertion 2D strategy, when a "locally–uniform" discretization is detected around node $\underline{x}_i$.

In order to approximate the TV on node $\underline{x}_i$ for our numerical solution $\tilde{u}$, we exploit the one point Gauss integration scheme, i.e. we set

$$\|\tilde{u}\|_{TV,i} = (|\tilde{u}_x(\underline{x}_i)| + |\tilde{u}_y(\underline{x}_i)|)\,|\Omega_i|,$$

being $|\Omega_i|$, the area of the i-th integration subdomain. One can see that, due to our previous assumptions on the domains of trial and test functions, Ω_i is the support of the i-th test function, i.e. the square centered at $\underline{x}_i$, whose radius is ρ_i. The partial derivatives u_x and u_y are approximated by differentiating the MLS solution $\tilde{u}(x)$.

Let

$$\mu = \max_{i=1,\ldots,N} \|\tilde{u}\|_{TV,i}.$$

Assume a threshold parameter γ is given, so that for each node $\underline{x}_i$, we refine our discretization around that node if and only if

$$\|\tilde{u}\|_{TV,i} > \gamma\mu.$$

Our refinement procedure around node $\underline{x}_i$ consists of adding one node in the middle of each line joining $\underline{x}_i$ with its closest $n^{(C)}$ nodes. The value $n^{(C)}$ must be guessed on the ground of geometrical considerations. We set $n^{(C)} = 8$ for 2D problems, hence on an uniform grid all "diagonal" nodes are added (see Figure 1). When an inserted node overlaps an old one, insertion is skipped.

Once all nodes in the discretization were processed, we say a new "discretization level" is reached.

Our refinement procedure is repeated until a maximum number N_{max} of nodes is reached, or a maximum number of "discretization levels" ℓ_{max} is computed.

Note that under the previous assumptions, when $\gamma = 0$ and an initial uniform distribution is enrolled, uniform refinements are obtained.

When 3D problems are attacked, the straightforward generalization of our 2D "local" TV applies, where subdomain volumes, in place of areas, are involved.

Our refinement strategy is accordingly updated by setting $n^{(C)} = 24$, in order to insert all "diagonal" nodes in a cube, when a (local) uniform discretization is detected.

3.3 Implementation issues

We implemented our algorithms into FORTRAN 77 codes, compiled via XLF v9.1. They were run on a machine operating under Ubuntu, featuring an Intel Core i7-2600K, 3.40 GHz (quadricore) processor, and 2x4GB, 1333 MHz, RAM.

No involved data structures were necessary in order to deal with our discretizations, since they are mere clouds of points.

Our Fortran codes exploit CSR sparse matrix storage representation, in order to deal with the large number of nodes needed by volume discretizations in non–structural, e.g. flow, heat, problems.

In order to sort the distances of each node from its neighbours, we exploited the Fortran subroutine SORT2 after NAPACK library.

4 Test problems

In order to check our adaptive strategies, we assign the forcing function f and we impose Dirichlet boundary conditions in eq. (1), so that its "test" solution is a function u undergoing large variations on a small portion of the domain.

4.1 2D problems

First, we consider the classical Gaussian function, centered at a given point $P_0 = (x_0, y_0)$, i.e.

$$u(x,y) = \exp(-c\,\left((x-x_0)^2 + (y-y_0)^2\right).$$

The parameter c is a large positive value which generates a high "bump" around P_0. In the sequel, $c = 200$ is set. Any adaptive procedure is like to be effective when finer discretizations enrol a large numer of dicretization nodes nearby P_0, where a large variation in u occurs. On the other hand, "far" from P_0 the u values are small, and u does not display large variations, hence nodes can be quite coarsely distributed without appreciable loss in estimation error. The setting $P_0 = (1/2, 1/2)$, the centroid of our domain, corresponds to the problem called in the sequel $\mathscr{P}_{GC}$.

As a further test problem we consider, after [Kee, Liu, Zhang, and Lu (2008)]

$$u(x,y) = \tan^{-1}(1000\,x^2\,y^2 - 1).$$

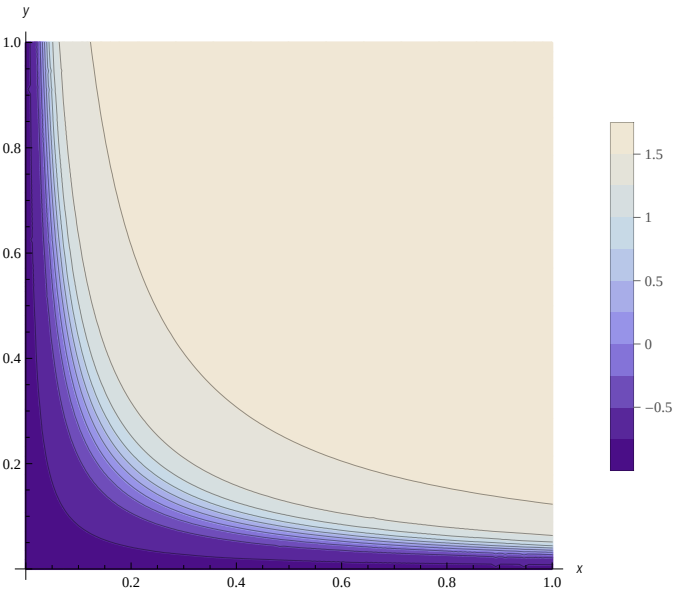


Figure 2: Contour regions for the solution of test problem $\mathscr{P}_H$.

This function displays a "hill" growing into the domain from the left and bottom sides of $[0,1]^2$. Figure 2 shows the contour levels of the surface. In the sequel, this test problem is called $\mathscr{P}_H$.

4.2 3D problems

By straightfordwardly extending our 2D test problems, we consider Gaussian functions centered at a given point $P_0 = (x_0, y_0, z_0)$

$$u(x,y,z) = \exp(-c\left((x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2\right).$$

When attacking 3D problems, $c = 200$ is set.

Three settings are exploited, corresponding to Gaussian "centers" on suitable domain points, either internal or on the boundary of our domain.

- The setting $P_0 = (1/2, 1/2, 1/2)$, will be again called test problem $\mathscr{P}_{GC}$.
- Setting $P_0 = (1/2, 1/2, 0)$, we speak of test problem $\mathscr{P}_{GXY}$.
- The setting $P_0 = (1/2, 0, 0)$, is called test problem $\mathscr{P}_{GX}$.

5 Numerical results

Being N the number of discretization nodes, we define the numerical error

$$e_{u,N} = \frac{\max_{i=1}^{N} |u_i - \tilde{u}_i|}{\max_{i=1}^{N} |u_i|} = \frac{\|u - \tilde{u}\|_1}{\|u\|_1},$$

where u_i is the exact value on node $\underline{x}_i$, while $\tilde{u}_i$ is the corresponding approximated value.

We are interested in analyzing the behavior of the errors when the number of nodes in the discretization increases, using our refinement procedure.

Recall that our strategy increases the number of nodes trying to node–wise identify the higher TV. We must check if the TV is an effective indicator of the error.

5.1 2D Numerical Results

5.1.1 Refinement parameters

By numerical experiments, we found that a convenient interval is $1/1000 \leq \gamma \leq 1/10$. Smaller γ values lead to uniform discretizations. Larger values lead to coarse discretizations which do not allow one to compute enough accurate approximations.

When 2D problems were attacked, we set $N_{max} = 6,000$, $\ell_{max} = 7$.

5.1.2 Discretizations

Table 1 shows fill and separation distances of our 2D finest discretizations, when $\mathscr{P}_{GC}$ and $\mathscr{P}_H$ test problems were solved. By inspecting the first line of Table 1 one can see that the 5th uniform discretization level corresponds to $n_x = n_y = 64$ subdivisions on each axis. One has $N_5 = (n_x+1)^2 = 4225$ nodes. The node distance on each axis is $h_5 = 1/64 \simeq 1.56\text{E-}2$. Note that at the 5th uniform discretization, we get $q_5 = 7.81\text{E-}3 = h_5/2$.

When solving problem $\mathscr{P}_H$ with our refinement technique, we stop at the same level and get the same q_5 value. When solving problem $\mathscr{P}_{GC}$, we stop at $\ell = 6$, hence displaying $q_6 = q_5/2$. It seems that the Gaussian solution allows for a more dense discretization around its center, without requiring too much nodes elsewhere in the domain. On the other hand, test problem $\mathscr{P}_H$, where the solution undergoes high variations on a "front" (instead of a single point), requires insertion of more nodes. The level $\ell = 6$ cannot be completed: more than N_{max} nodes should be enrolled.

Figure 3 shows the nodes in the finest discretization, when the test problem $\mathscr{P}_H$ is attacked via our refinement procedure. One can see that the discretization nodes

Table 1: 2D tests. Number of nodes, N_ℓ, fill distance, $h_{\ell,[0,1]^2}$, and separation distance, q_ℓ, of our final ℓ-th irregular discretization. Note: the label "*any*" means that any value is admitted; $\gamma = 0$ denotes uniform discretization. Parameter values are $\alpha = 1$, $\beta = 4$.

test	B	γ	ℓ	N_ℓ	$h_{\ell,[0,1]^2}$	q_ℓ
any	any	0	5	4225	1.10E-2	7.81E-3
$\mathscr{P}_H$	2	0.01	5	2401	6.25E-2	7.81E-3
	2	0.001	5	3282	4.42E-2	7.81E-3
	3	0.01	5	2069	6.25E-2	7.81E-3
	3	0.001	5	3260	4.42E-2	7.81E-3
$\mathscr{P}_{GC}$	2	0.01	6	3205	4.42E-2	3.91E-3
	2	0.001	6	5467	3.13E-2	3.91E-3
	3	0.01	6	2822	4.42E-2	3.91E-3
	3	0.001	6	3981	4.42E-2	3.91E-3

condense nearby the $x = 0$ and $y = 0$ sides. They well match the zone where our test solution undergoes the largest variation, as one can argue by comparing with the solution contour regions displayed in Figure 2.

When test problem $\mathscr{P}_{GC}$ is attacked, one could see that the nodes in the finer discretizations condense around the centroid of $[0,1]^2$ (not shown, for brevity).

5.1.3 Convergence history

In the sequel, the Figures reporting our numerical results start from considering the $\ell = 2$ level, enrolling $N_2 = 81$ nodes. Level $\ell = 1$, counting $N_1 = 25$ nodes, proved too a coarse discretization for producing useful results.

Figure 4 shows errors and maximum TV, recorded when test problem $\mathscr{P}_{GC}$ is attacked. We set $\alpha = 1$, together with the optimal value, identified by experiments, $\beta = 5$. Either uniform discretizations or our refinement strategy with $\gamma = 0.1, 0.01, 0.001$, are considered in each frame.

Frames (a) and (c) display our results obtained by setting $B = 2$, i.e. using a quadratic polynomial basis in the MLS approximation. Frames (b) and (d) report our results when $B = 3$ was set, meaning a cubic polynomial MLS basis.

By inspecting frames (a) and (b) in Figure 4 one can see that when $\gamma = 0.01, 0.001$, the error decreases as the number of nodes increases. Moreover, when either $\gamma = 0.01$ or $\gamma = 0.001$ the convergence of our refinement strategy outperforms the uniform discretization. When $\gamma = 0.1$ the TV is not a good error indicator. Compar-

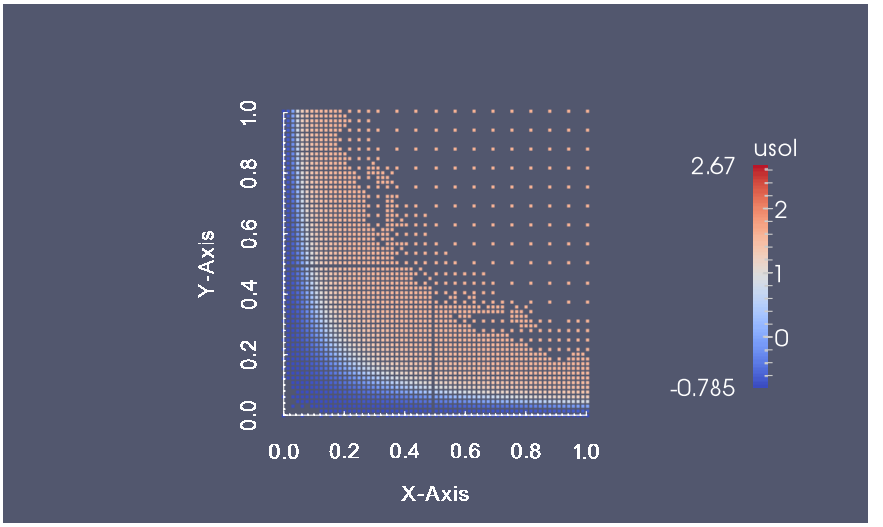


Figure 3: Test problem $\mathscr{P}_H$. Nodes in the finest, refined 2D discretization, $\beta = 4$, $\gamma = 0.01$.

ing frames (a) and (c), one can see that while the TV is either decreasing or mildly increasing when finer discretizations are enrolled. The corresponding error behaviors in frame (a) well match the TV ones in frame (c), except in the case $\gamma = 0.1$. Recall that estimating the TV one must rely upon numerical approximations of partial derivatives, which is more error prone that evaluating the plain function [Libre, Emdadi, Kansa, Rahimian, and Shekarchi (2008); Babuska, Strouboulis, and Upadhyay (1997)]. One can argue that better estimations of the partial derivatives should give more accurate TV (and hence error) estimations, i.e. a better adaptive strategy.

Note that the accuracy with the finest uniform discretization is lower than when using our adaptive strategy, when setting either $\gamma = 0.01$, or $\gamma = 0.001$.

Let us continue analyzing Figure 4, which refers to test problem $\mathscr{P}_{GC}$. Analogous results as before can be drawn by inspecting frame (b) for errors, and comparing it with frame (d) for TV values. Recall that they report our results obtained by setting $B = 3$, i.e. cubic polynomials are exploited in the MLS approximation.

Let us now consider Figure 5, which reports our results obtained when attacking test problem $\mathscr{P}_H$. Again, frames (a) and (c) refer to setting $B = 2$ (quadratic MLS basis), while frames (b) and (d) refer to setting $B = 3$ (cubic MLS basis).

By inspecting frames (a) and (c) in Figure 5, one can see that the accuracy obtained using uniform discretizations is quite the same as the corresponding "level"

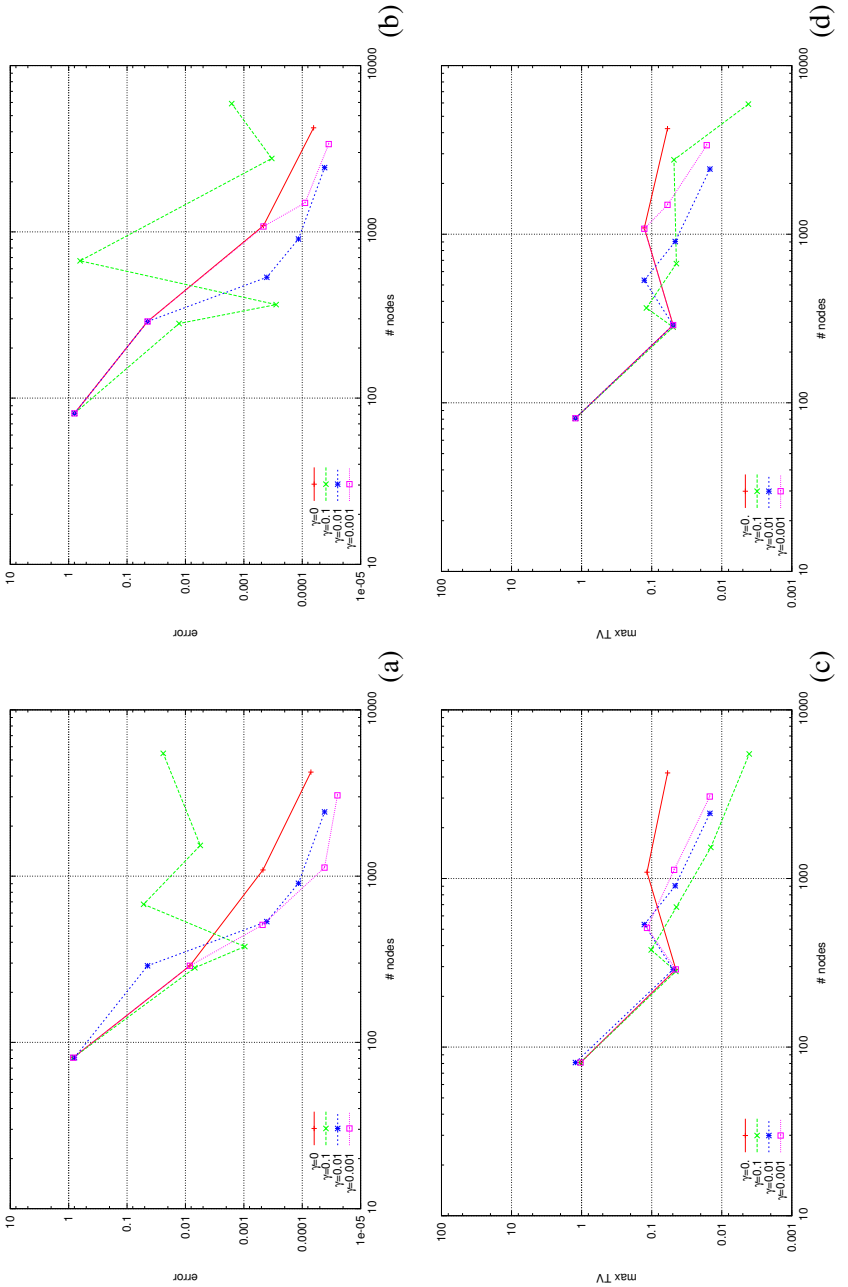


Figure 4: Errors and max TV when attacking test problem $\mathscr{P}_{GC}$. We set $\beta = 5$. Frames (a) and (c) report results obtained by quadratic, ($B = 2$) MLS basis. Frames (b) and (d) pertain to cubic ($B = 3$) MLS basis.

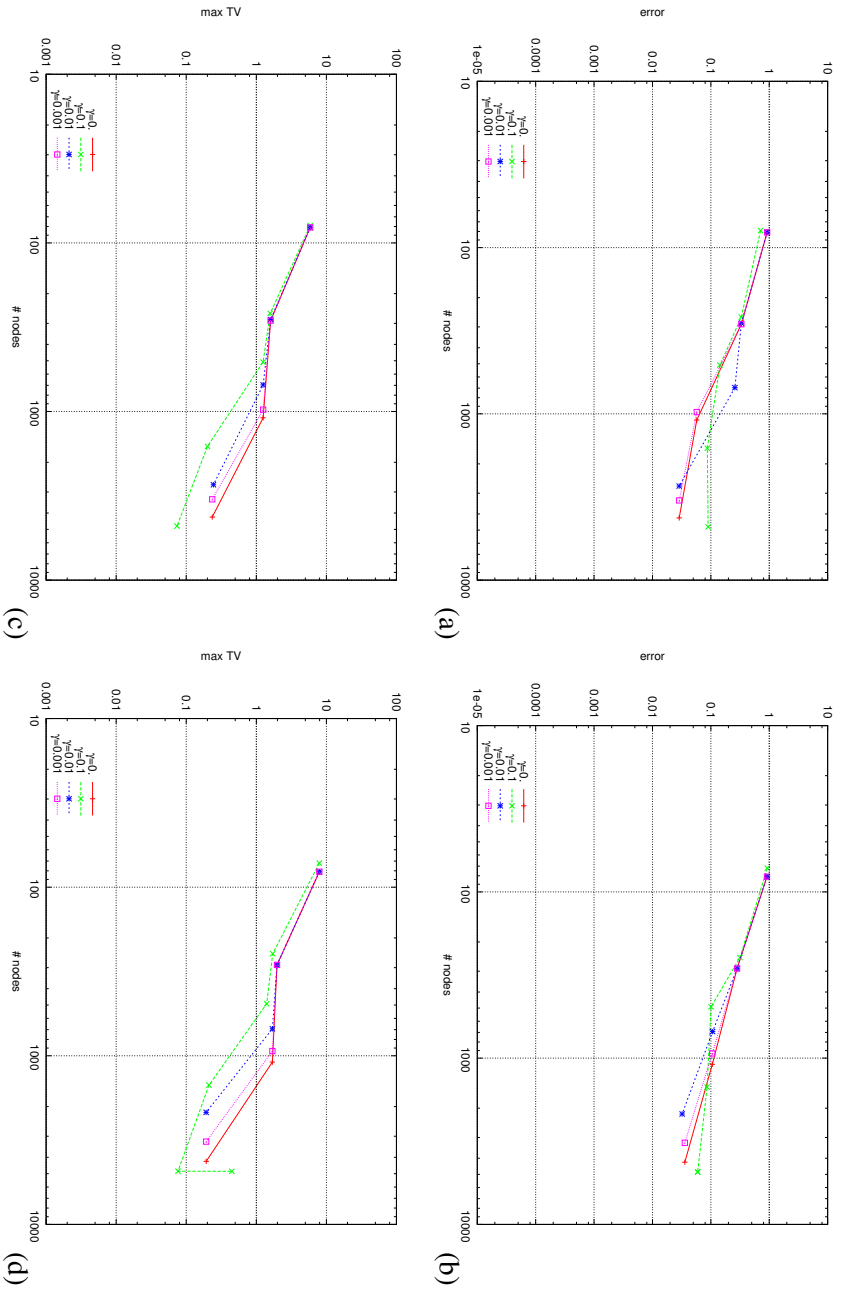


Figure 5: Analogous to the previous Figure. Test problem $\mathscr{P}_H$.

of adaptive discretization, when either $\gamma = 0.01$ or $\gamma = 0.001$ was set. The setting $\gamma = 0.1$ gives less accurate results.

By comparing frames (a), (c) and (b), (d), one can see that the TV is an apt error indicator, except when $\gamma = 0.1$ is set.

We tested our 2D adaptive strategy on many other test problems. We obtained error and TV behaviors which resemble either one of the previous ones.

5.1.4 Effectivity

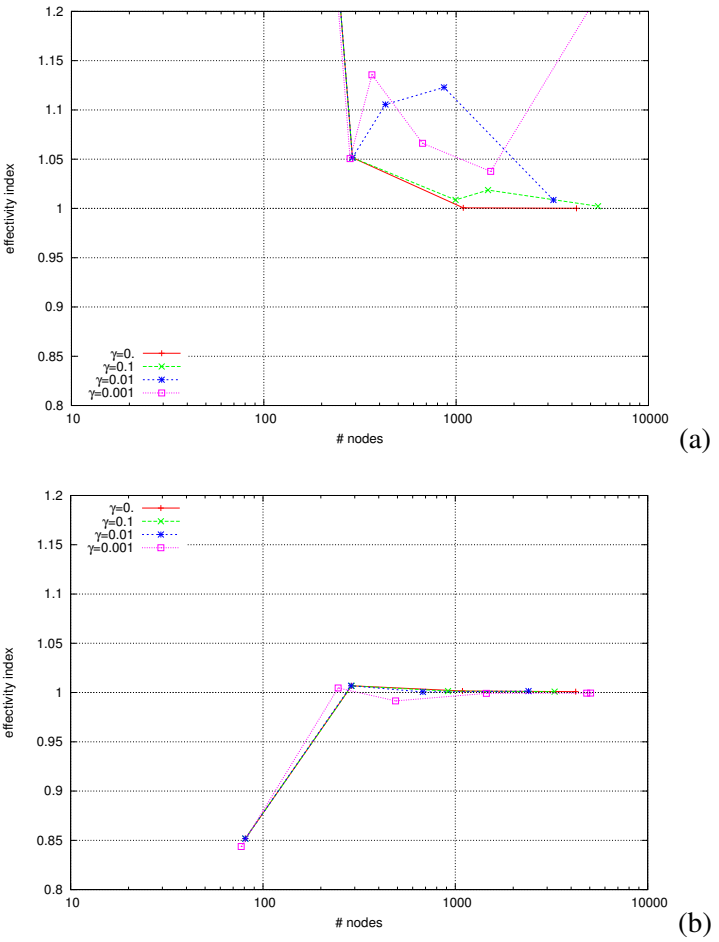


Figure 6: Effectivity index behavior, $\beta = 5$, $B = 2$. Frame (a): test problem $\mathscr{P}_{GC}$. Frame (b): test problem $\mathscr{P}_H$.

Let us define the *effectivity index* at discretization level ℓ, adapted after [Babuska, Strouboulis, and Upadhyay (1997)]

$$\kappa_\ell = \frac{\|\tilde{u}\|_{1,\ell}}{\|u\|_{1,\ell}}.$$

The idea is that the more κ_ℓ is close to one, the better our TV indicator should match the exact one. Babuska, Strouboulis, and Upadhyay (1997) showed that this indicator can be effective when Finite Element techniques are exploited. We wondered wether this is true also for MLPG techniques.

Note that in [Babuska, Strouboulis, and Upadhyay (1997)] the efficiency index is element–wise, while here it is node–wise. True meshless methods deal with "nodes", not with "elements".

Figure 6 displays the effectivity index behavior when either the $\mathscr{P}_{GC}$, frame (a), or the $\mathscr{P}_H$, frame (b), are attacked, by a quadratic polynomial MLS basis ($B = 2$). Analogous behaviors could be shown for cubic basis (not shown, for brevity).

Test problem $\mathscr{P}_{GC}$: by inspecting frame (a) one can see that when uniform discretizations are exploited, the index is monotonically decreasing to 1. Quite erratic behaviors are displayed when our adaptive procedure is enrolled. Recall that when test problem $\mathscr{P}_{GC}$ is attacked, the error displays a monotonic decrease when either uniform or adaptive refinements are computed, except when $\gamma = 0.1$ is set. We can say that our efficiency index is not a good indicator of the effectiveness of our MLPG procedure.

Test problem $\mathscr{P}_H$: by inspecting frame (b) one can see that the effectivity index monotonically converges to 1 irrespective of the discretization exploited. We say that in this case the efficiency index is a relevant indicator of the effectiveness of our procedures.

By numerical experiments (not shown here) we found that this index is not a useful value when the MLPG technique is exploited. In the sequel we do not report about it anymore.

5.2 3D Numerical Results

5.2.1 Discretizations

When 3D problems were solved $N_{max} = 50,000$, $\ell_{max} = 6$ were set.

By extensive numerical experiments, we found that $0.01 \leq \gamma \leq 0.005$ is an apt interval for the refining parameter.

Table 2 shows fill and separation distances of our 3D finest discretizations, when our 3D test problems were solved. By inspecting the first line of Table 2 one can

Table 2: 3D tests, $B = 3$. Number of nodes, N_ℓ, fill distance, $h_{\ell,[0,1]^3}$, and separation distance, q_ℓ, of our final ℓ-th irregular discretization. The half grid size, $h_\ell/2$, of our uniform discretizations, is also shown. Note: the label "*any*" means that any value is admitted; $\gamma = 0$ denotes uniform discretization. Parameter values are $\alpha = 1$, $\beta = 4$.

test	γ	ℓ	N_ℓ	$h_{\ell,[0,1]^2}$	q_ℓ
any	0	4	35937	2.71E-2	1.56E-2
$\mathscr{P}_{GC}$	0.01	5	12673	1.25E-1	7.81E-3
	0.005	4	7926	1.08E-1	1.56E-2
$\mathscr{P}_{GXY}$	0.01	5	7270	1.40E-1	7.81E-3
	0.005	5	9594	1.40E-1	7.81E-3
$\mathscr{P}_{GX}$	0.01	6	18526	1.40E-1	3.91E-3
	0.005	5	5168	1.40E-1	7.81E-3

see that the 4th uniform discretization level corresponds to $n_x = n_y = n_z = 32$ subdivisions on each axis; hence the distance of nodes on each axis is $h_4 = 1/32 \simeq$ 3.13E-2. One has $N_4 = (n_x + 1)^3 = 35,937$ nodes. Note that at 4th uniform discretization, we get $q_4 = 1.56\text{E-}2 = h_4/2$.

The finest uniform discretization bears $\ell = 4$, while our refined discretizations go up to $\ell = 6$ (see line 6 in Table 2) for test problem $\mathscr{P}_{GX}$. Indeed each solution of problems $\mathscr{P}_{GXY}$ and $\mathscr{P}_{GX}$ changes around a point on the domain boundary. Less nodes are inserted than for problem $\mathscr{P}_{GC}$, whose solution changes inside (around the centroid of) the domain.

Figure 7 shows the nodes in the finest discretization when test problem $\mathscr{P}_{GX}$ is attacked via our refinement procedure. The "blob" around $P_0 = (1/2, 0, 0)$ is generated by our technique, which inserts nodes around the point of maximum variation in the exact solution. As expected, the nodes are "quite uniformly" distributed "far" from P_0, where the solution is practically constant.

5.2.2 Convergence history

Figure 8 shows the error behavior when $B = 3$, $\alpha = 1$, $\beta = 4$. By setting $B = 2$ we obtained similar results (not shown for brevity).

Errors for $\gamma = 0.0$ are not reported, since they overlap the "unif" case. Such overlap confirms that, as expected, our adaptive strategy produce uniform discretizations when $\gamma = 0$ is set. Implementation errors and/or unfeasibility of our adaptive strategy are restrained by these results.

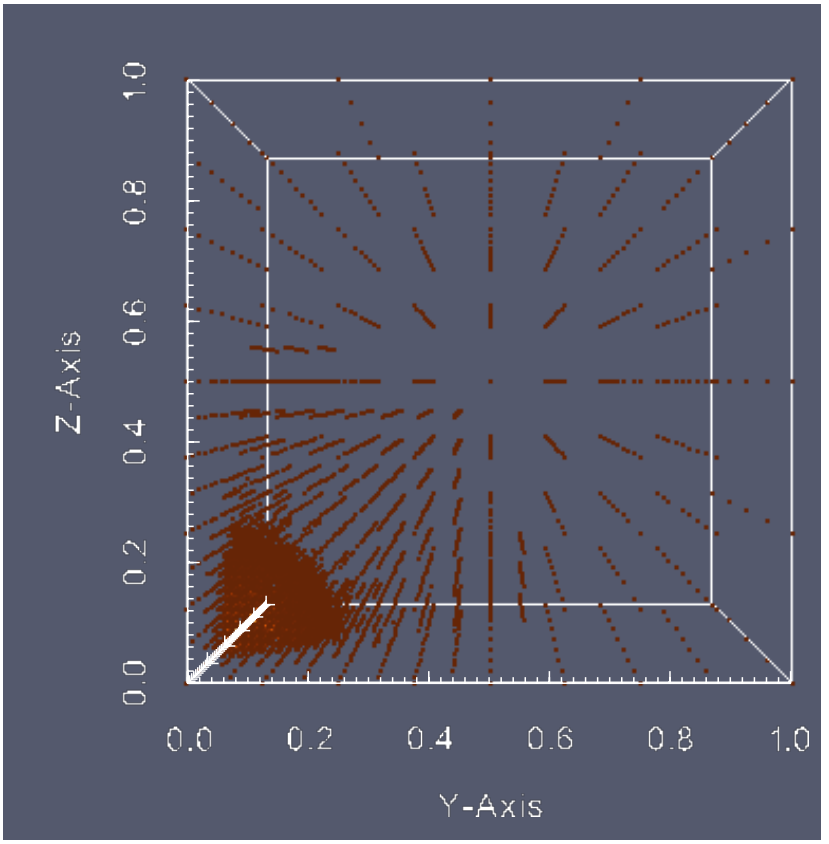
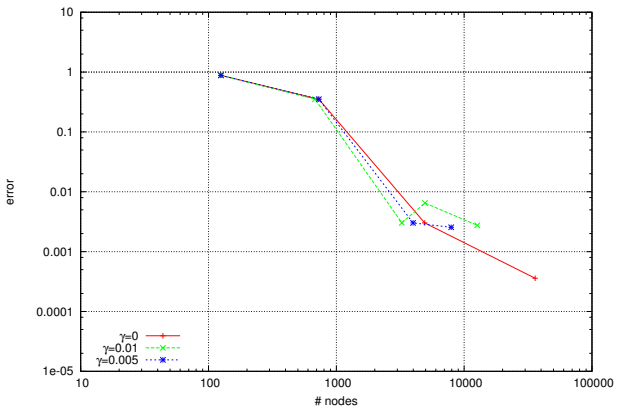


Figure 7: Test problem $\mathscr{P}_{GX}$. Nodes in the finest, refined 3D discretization, $B = 3$, $\beta = 4$, $\gamma = 0.005$.

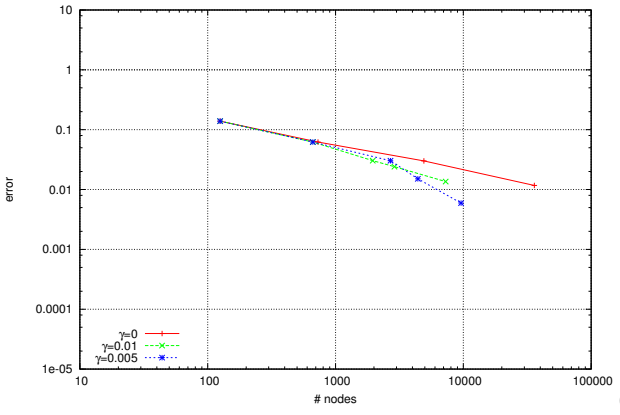
By inspecting Figure 8 one can argue that the accuracy of our adaptive procedure well compares with that one of a brute uniform discretization. The setting $\gamma = 0.005$ proves effective, raising quite the same (or better) accuracy than the uniform discretization, requiring far less nodes than the latter.

5.2.3 Performance

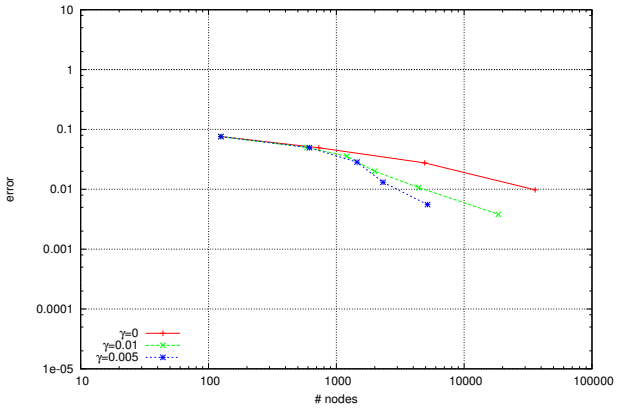
Table 3 shows CPU times spent in order to compute the solution of our test problems. The ratios to the corresponding uniform discretization level are also shown. Recall that in the previous Section we showed that quite the same (or better accuracy) can be obtained by exploiting our adaptive procedure. Respect to uniform discretizations, a far less number of nodes can be used by our adaptive technique,



(a)



(b)



(c)

Figure 8: Errors recorded when our 3D test problems are solved. Parameter setting: $\beta = 4$, $B = 3$. Frame (a) refers to test problem $\mathscr{P}_{GC}$, frame (b) to test problem $\mathscr{P}_{GXY}$, frame (c) to $\mathscr{P}_{GX}$.

Table 3: CPU seconds spent when solving the proposed test problems via either the finest uniform discretization U, or our corresponding refined, irregular, finest discretizations, I_γ, $\gamma = 0.01, 0.005$. Note: $\gamma = 0$ means "uniform discretization": $I_0 = U$ holds true. For each test problem, the ratios are also shown: seconds spent when using U divided by seconds spent using I_γ.

test	γ	ℓ	N_ℓ	CPU (s)	U/I_γ
$\mathscr{P}_{GC}$	0	4	35937	6299.3	1.00
	0.01	5	12673	2533.4	2.49
	0.005	4	7926	1123.9	5.60
$\mathscr{P}_{GXY}$	0	4	35937	6546.1	1.00
	0.01	5	7270	1096.9	5.97
	0.005	5	9594	1900.0	3.45
$\mathscr{P}_{GX}$	0	4	35937	6708.4	1.00
	0.01	6	18526	5097.9	1.32
	0.005	5	5168	759.0	8.84

hence allowing up to 8.84 (Table 3, last line, last column) faster CPU time (and possibly better accuracy, e.g. see Figure 8, frame (c)) than uniform refinements.

6 Conclusions

We analyzed the accuracy and efficiency in solving diffusion problems of a fresh refinement technique for MLPG.

The following points are worth mentioning.

Concerning 2D problems.

- Our adaptive strategy condense nodes around the regions where the approximating function undergoes "high" variation.
- Our adaptive strategy allows for attaining quite the same (or higher) accuracy as uniform discretizations, by using far less nodes.
- The "numerical, local" TV is a sound measure of the numerical error, provided the refine parameter is not "too large".
- Our "efficiency index", adapted after [Babuska, Strouboulis, and Upadhyay (1997)], seems not a good estimator of the effectiveness of our refinement MLPG procedure. Monotone decreasing errors with finer discretization levels can correspond to oscillating efficiency index behaviors.

Concerning 3D problems.

- Like when solving 2D problems, as expected the majority of nodes is inserted where the solution undergoes high variation.
- As in the 2D case, one can obtain quite the same (or higher) accuracy as exploiting uniform discretizations, by using far less nodes.
- Comparable accuracy can be obtained by using either uniform or our adaptive discretizations, the latter being up to more than 8 times faster.

Future work: We plan to extend our refinement technique to DMLPG methods [Mazzia, Pini, and Sartoretto (2012)]. In principle the DMLPG approach allow for approximating any linear functional of the solution. Partial derivatives, and hence the TV, can be directly approximated by DMLPG with expected higher accuracy and/or efficiency.

Acknowledgement: This work was partially funded by INDAM–GNCS. The third Author received partial support from ADIR funding of Università Ca' Foscari Venezia.

References

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

Babuska, I.; Banerjee, U.; Osborn, J. E. (2005): Survey of meshless and generalized finite element methods: A unified approach. Technical report, Office of Naval Research, One Liberty Center, 875 North Randolph Street Suite 1425, Arlington, VA, 22203-1995, 2005.

Babuska, I.; Strouboulis, T.; Upadhyay, C. (1997): A model study of the quality of a posteriori error estimators for finite element solutions of linear elliptic problems, with particular reference to the behavior near the boundary. *International Journal for Numerical Methods in Engineering*, vol. 40, no. 14, pp. 2571–2577.

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.

Fasshauer, G. E. (2007): *Meshfree approximation methods with MATLAB*, volume 6 of *Interdisciplinary Mathematical Sciences*. World Scientific Publishing Co., Singapore. With 1 CD-ROM (Windows, Macintosh and UNIX).

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

Gerace, S.; Erhart, K.; Kassab, A.; Divo, E. (2013): A model-integrated localized collocation meshless method (MIMS). *Computer Assisted Methods in Engineering and Science*, vol. 20, pp. 207–225.

Gordeliy, E.; Peirce, A. (2013): Implicit level set schemes for modeling hydraulic fractures using the XFEM. *Comput. Methods Appl. Mech. Engrg.*

Kee, B. B. T.; Liu, G. R.; Zhang, G. Y.; Lu, C. (2008): A residual based error estimator using radial basis functions. *Finite Elem. Anal. Des.*, vol. 44, no. 9-10, pp. 631–645.

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

Libre, N. A.; Emdadi, A.; Kansa, E. J.; Rahimian, M.; Shekarchi, M. (2008): A stabilized RBF collocation scheme for Neumann type boundary value problems. *Computer Methods in Applied Mechanics and Engineering*, vol. 24, no. 1, pp. 61–80.

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.; Sartoretto, F. (2012): Numerical investigation on direct MLPG for 2d and 3d potential problems. *Computer Modeling in Engineering & Sciences*, vol. 88, pp. 183–210.

Mazzia, A.; Sartoretto, F. (2010): Meshless solution of potential problems by combining radial basis functions and tensor product ones. *Computer Modeling in Engineering & Sciences*, vol. 68, no. 1, pp. 95–112.

Micchelli, C. (1986): Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation*, vol. 2, no. 1, pp. 11–22.

Roque, C.; Madeirab, J.; Ferreira, A. (2014): Node adaptation for global collocation with radial basis functions using direct multisearch for multiobjective optimization. *Engineering Analysis with Boundary Elements*, vol. 39, pp. 5–14.

Schwab, C. (1998): *P- and Hp- Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics*. G.H.Golub and others. Clarendon Press.

Sewell, E. G. (1972): *Automatic Generation of Triangulations for Piecewise Polynomial Approximation*. PhD thesis, Purdue Univ., West Lafayette, Ind., 1972.

Strang, G. (2007): *Computational Science and Engineering*. Wellesley–Cambridge Press, Wellesley, MA.

Verfürth, R. (2013): Adaptive finite element methods. Technical report, Fakultät für Mathematik, Ruhr-Universitat Bochum, 2013.

Zienkiewicz, O.; Zhu, J. (1992): Superconvergent patch recovery and a posteriori error estimates. part 1: the recovery technique. *International Journal for Numerical Methods in Engineering*, vol. 33, no. 7, pp. 1331–1364.