International Journal of Analytical, Experimental and Finite Element Analysis (IJAEFEA), Issue. 1, Vol. 6, March 2019
e-ISSN: 2394-5141, p-ISSN: 2394-5133, pp 30 - 39
DOI: 10.26706/IJAEFEA.1.6.20190301

A VEM–based mesh–adaptive strategy for potential problems

Annamaria Mazzia [1]
annamaria.mazzia@unipd.it

Flavio Sartoretto [2]
flavio.sartoretto@unive.it

[1] Dipartimento di Ingegneria Civile,
Edile e Ambientale,
Università di Padova,
via F. Marzolo 9,
35131 Padova, Italy

[2] DAIS
Università Ca' Foscari Venezia,
Via Torino 155,
30172 Mestre VE, Italy

Abstract

The Virtual Element Method (VEM) is an evolution of the mimetic finite difference method which overcomes many limitations affecting classic Finite Element Methods (FEM). VEM for 2D problems allows for exploiting meshes consisting of any polygonal elements. No limitations on their internal angles are needed. Hanging nodes are easily treated. Notably, VEM is well apt to mesh–adaptive algorithms. In this paper we detail an implementation of mesh–adaptive VEM for potential problems. We suggest a fresh, promising approach. We show on suitable test problems that a gain in efficiency can be obtained, respect to uniform, fine discretizations.

Index terms - VEM, MESH ADAPTIVITY, POISSON PROBLEM.

1 Introduction

In the last two decades, the numerical treatment of partial differential equations (PDEs) has been focused on treating meshes with arbitrarily-shaped polygonal/polyhedral (polytopal, for short) elements. A non-exhaustive list of such methods include the Mimetic Finite Difference method [14, 15, 19, 42, 44–47] the Polygonal Finite Element Method [48, 51, 52], the polygonal Discontinuous Galerkin Finite Element Methods [4, 8, 26, 27] the Hybridizable Discontinuous Galerkin and Hybrid High-Order Methods [33, 35], the Gradient Discretization method [34, 38], the Finite Volume Method [37], and the BEM-based FEM [50].

An alternative approach that proved to be successful is the Virtual Element method (VEM), originally proposed in [10] for the numerical treatment of second-order elliptic problems [29, 30], and readily extended to linear and nonlinear elasticity [11, 16], plate bending problems [25], Cahn-Hilliard equation [2], Stokes equations [1], Darcy-Brinkam equation [55], discrete topology optimization problems [3], fracture networks problems [22], eigenvalue problems [40, 54]. The mixed virtual element formulation was proposed in [12, 24]. The nonconforming formulations for second-order elliptic problems are analyzed in [7], and later extended to general advection-reaction-diffusion problems, Stokes equation, the biharmonic problems, the eigenvalue problem, and the Schrodinger equation in [5, 23, 31]. The p- and hp versions of the VEM were developed in [13, 21] and efficient multigrid methods for the resulting linear system of equations were investigated in [6]. A posteriori error estimates can be found in [18, 28]. It is also worth mentioning that a peculiar feature of VEM is designing approximation spaces characterized by high continuity properties; For details see cf. [17] and the works on high-order partial differential equations as the biharmonic equations mentioned above.

VEM is notably apt for mesh–adaptive methods. Unlike using conforming FEM, using VEM one can manage many arbitrary types of polygonal elements, and hanging nodes, hence mesh refinement is straightforward: elements with consecutive co-planar edges are allowed. Locally adapted meshes do not require any expensive local mesh post-processing: no complex procedures for obtaining a conforming, refined mesh [49] are required.

On the other hand, identifying a suitable a posteriori error estimator, and an attached criterion for identifying the elements to be refined, is a crucial task [18, 28], like in FEM [9, 32, 41, 49].

We use an adaptive algorithm for elliptic problems consisting of the classic steps: solve, estimate, mark, refine [36]. In this context, given a polygonal subdivision of the problem domain, one solves the VEM problem, estimates the error using our a posteriori error bound, marks a subset of elements for refinement, and refines marked elements.

We restricted to linear VEM since we design the analysis of hydraulic–like problems. While for mechanical–like problems few, high order VEM elements are usually enrolled, hydraulic–like problems typically involve a large number of low–order elements.

An a posteriori error estimator is completely worked out after [28]. Its behavior in our test problems was analyzed by extensive numerical computations.

This paper is organized as follows. Section 2 recalls our model problem, Section 3 depicts the proposed refinement procedure. Section 4 sketches our test problems. Section 5 shows and discusses our numerical results. Section 6 summarizes our conclusions.

2 The problem

Let us consider the Poisson model problem

$$\begin{cases} -\nabla \cdot (\nabla u) = f, & \text{in } \Omega \\ u = g, & \text{on } \partial\Omega_d, \\ (\nabla u) \cdot \vec{n} = q, & \text{on } \partial\Omega_n, \end{cases} \tag{1}$$

$\Omega \subset \mathbb{R}^2$ being a polygonal domain whose boundary is $\partial\Omega = \partial\Omega_d \cup \partial\Omega_n$, $\partial\Omega_d \cap \partial\Omega_n = \emptyset$. Here $\partial\Omega_d$ is the Dirichlet boundary portion, while $\partial\Omega_n$ bears Neumann boundary conditions. Moreover, f is a given source function $\in L^2(\Omega)$, $\vec{n}$ is the outward unit normal to the boundary $\partial\Omega$; $g \in H^{1/2}(\partial\Omega_d)$ is the Dirichlet function, while the flux function is $q \in L^2(\partial\Omega_n)$.

In the sequel, $(\cdot, \cdot)$ is the standard scalar product in $L^2(\Omega)$, and $\underline{x} = (x, y)$ is a point in $\mathbb{R}^2$.

Let $v \in H^1(\Omega)$. By multiplying each side of the differential equation in (1), and by Green's second theorem, we can rewrite our differential problem into the variational formulation

$$\begin{cases} \text{Find } u \in V := H^1(\Omega), \text{ such that} \\ u = g, \text{ on } \partial\Omega_d, \\ a(u, v) = \mathcal{L}(v), \quad \forall v \in V_{\partial\Omega_d} := H^1_{\partial\Omega_d}(\Omega), \end{cases} \tag{2}$$

where

$$a(u, v) = \iint_\Omega \nabla u \cdot \nabla v d\Omega,$$

$$\mathcal{L}(v) = \iint_\Omega f v d\Omega + \int_{\partial\Omega_n} q v ds. \tag{3}$$

We enrol the low-order, linear VEM. Our implementation is standard, based upon the projection operator Π^∇, which is associated to the bilinear form $a(\cdot, \cdot)$ in eq. (3). The local stiffness matrix is decomposed into the sum of a consistency matrix, and a stability matrix. The consistency matrix can be computed, while the stability matrix is not computable. The latter is approximated by introducing a local symmetric positive definite, element–wise bilinear form $S^E(\cdot, \cdot)$. This form is introduced in order to scale the element–wise discretization of $a(\cdot, \cdot)$ on the kernel of Π^∇_E.

For the details, see [20].

Recall that the number of Degrees Of Freedom (DOF) for linear VEM equals the number of vertices in the mesh.

3 Refinement procedure

Our refinement procedure starts from an initial partition $\mathcal{P}_1$, of the problem domain, which is assumed to be a convex polygon itself. The partition $\mathcal{P}_1$ is made by a "small" number of convex polygons, which can be a mixture of triangles, quadrilaterals, pentagons, etc. Note that when refining a non–convex polygonal element, our refinement procedure can add some nodes which lie outside the polygon, hence non–convex polygonal elements cannot be enrolled.

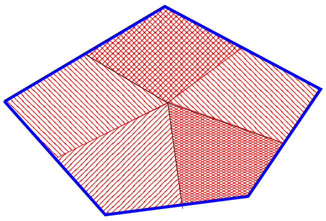

Figure 1: Sample polygon, refinement strategy.

Let $\mathcal{P}_\ell = \mathcal{T}_\ell$ be a partition of the polygonal domain Ω into non–overlapping polygonal elements E, computed by our refinement procedure. It consists of N_E elements, being h_E the diameter of a given element E, and let N_v be the total number of vertexes in our partition.

The approximated numerical solution $\tilde{u}_i$, $i = 1, \ldots, N_v$, to Poisson problem (1) is computed on each vertex of our partition.

Our refinement procedure relies upon identifying those elements which must be refined in our total N_E elements, and then partitioning each convex N_s–sides polygonal element into N_s smaller quadrilaterals, as sketched in Figure 1 for a pentagon. Each side midpoint is connected with the center of the polygon. Thanks to the high robustness of VEM, possible hanging nodes are left as such.

Concerning the identification of the elements which must be refined, for each given element E in a given discretization $\mathcal{T}_\ell$ we compute

$$\eta_E = h_E^2 \, ||f_h||^2_{(0,E)} + \\ S^{(E)} \left((\Pi - I)u_h, (\Pi - I)u_h\right) + \\ \sum_{s \subset \partial E} h_s \|J_s\|^2_{0,s}.$$

where h_E is the diameter of element E, f_h is our discretization of the source function in the model problem (1); Π is a shorthand for Π_E^∇ above, and u_h is our VEM approximated solution. Moreover, h_s is the side length, J_s is the jump across side s. The estimator η_E was adapted to Poisson problem after Theorem 13 in [28].

Following Dörfler criterion [36], we performed the steps detailed in [39]. For any given mesh $\mathcal{P}_\ell = \mathcal{T}_\ell$, we detect the minimal set $\mathcal{M} \subset \mathcal{T}_\ell$ such that

$$\theta \sum_{E \in \mathcal{T}_\ell} \eta_E^2 \le \sum_{E \in \mathcal{M}} \eta_E^2,$$

for a given $0 < \theta < 1$, We mark for refinement only those elements in $\mathcal{M}$, counting let us say N_M elements.

Our refinement strategy, sketched in Figure 1, splits any triangle into three quadrilaterals, and any given n–side polygon, $n > 3$, into n quadrilaterals. On this ground, we compute an expected number of elements N_G in the refined mesh, as

$$N_G = N_E + 3\, N_M.$$

If N_G is larger than a prescribed value $N_{max}^{(E)}$, we assume that a "too fine" refinement is required, hence the refinement is not performed. Our procedure is stopped.

Otherwise, a refined mesh $\mathcal{P}_{\ell+1}$ is built, and the refinement process can be started again.

4 Test problems

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

First, we consider the classical Gaussian function, centered on a given point $Q_0 = (x_0, y_0)$, i.e.

$$u(x, y) = \exp(-c\left((x - x_0)^2 + (y - y_0)^2\right)). \tag{4}$$

The parameter c is a large positive value that generates a high "hump" around Q_0. In the sequel, we set $c = 200$.

Let us assume that we numerically solve the Poisson problem (1) in $\Omega = [0, 1]^2$, $\partial\Omega_d = \partial\Omega$, having set the Dirichlet boundary conditions such that its solution is the function (4). The setting $Q_0 = (1/2, 1/2)$, the center of our domain, corresponds to the 2D problem called PG in the sequel, where "G" stands for "Gaussian–based" test problem.

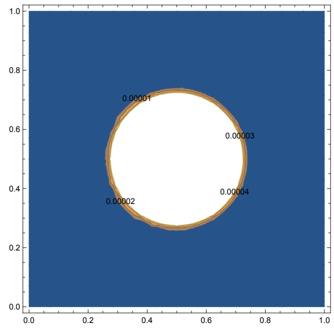

Figure 2: Contour regions for the solution of the test problem PG.

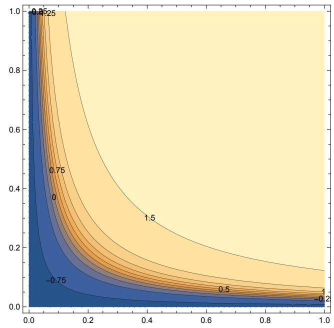

Figure 3: Contour regions for the solution of the test problem PA.

Any adaptive procedure is likely to be effective when finer discretizations adopt a large number of discretization nodes near the point Q_0 where a large variation in u occurs. On the other hand, "far away" from Q_0 the u values are small, and u does not display large variations, so the nodes can be distributed quite coarsely with no appreciable loss of accuracy.

As a further test problem we consider, as in [43]

$$u(x, y) = \tan^{-1}(1000\, x^2\, y^2 - 1). \tag{5}$$

This function displays a "hill" rising from the bottom left side of $[0, 1]^2$. Figure 3 shows the contour levels of the surface.

We numerically solve the Poisson problem (1) in $\Omega = [0, 1]^2$, $\partial\Omega_d = \partial\Omega$, having set the Dirichlet boundary conditions such that its solution is the function (5).

The ensuing differential problem is labeled test problem PA, where "A" is the mnemonic for the "Arctan–based" test solution.

5 Numerical results

We now compare the accuracy one can obtain using linear VEM by exploiting our adaptive refined meshes, respect to using uniformly refined meshes.

The results documented in the sequel were obtained by running our Matlab code on a Dell Inspiron 5749 PC with one Intel i5-5200U CPU @ 2.20GHz (2 cores, 4 threads). The PC works under Linux 3.16.0-4, and is equipped with a 8 GB RAM.

Let us assume that we perform successive, either adaptive or uniform, refinements of initial conforming meshes, made by either triangles, or squares, or n–side polygons, $n \geq 4$. The polygon mesh were obtained by Polymesher software [53]. Figure 4 shows our initial meshes.

From now on, the term "triangle mesh" denotes an initial mesh made by triangular elements, or one of its refinements. Analogously, the terms "square mesh", and "polygon mesh" refer to initial meshes made by either squares or polygons, respectively, and their refinements.

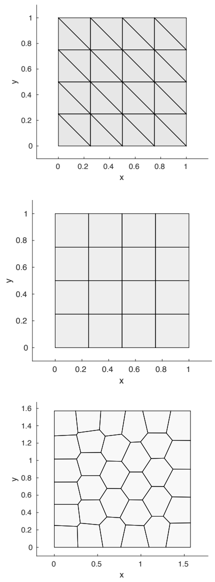

Figure 4: Top, center, and bottom frame shows the initial meshes made by triangles, squares, and n–side polygons, $n = 4, 5, 6$, respectively.

-	triangles		squares		polygons	
ℓ	N_E	N_v	N_E	N_v	N_E	N_v
1	32	25	16	25	32	66
2	128	81	64	81	128	257
3	512	289	256	289	512	1022
4	2048	1089	1024	1089	2048	4088
5	8192	4225	4096	4225	8192	16318

Table 1: Number of elements and vertices in our uniformly refined meshes.

-	PG		
ℓ	triangles	squares	polygons
1	9.33E-01	1.06E+00	1.07E+00
2	6.17E-01	7.92E-01	6.17E-01
3	4.91E-01	4.52E-01	3.41E-01
4	2.76E-01	2.53E-01	1.76E-01
5	1.41E-01	1.27E-01	8.99E-02
-	PA		
ℓ	triangles	squares	polygons
1	8.88E-01	1.11E+00	1.09E+00
2	8.09E-01	8.28E-01	7.15E-01
3	5.82E-01	6.24E-01	4.67E-01
4	3.03E-01	3.16E-01	2.31E-01
5	1.57E-01	1.66E-01	1.20E-01

Table 2: H_1–errors raised by VEM when approximating problems PG and PA by uniformly refined meshes.

vs h						
	PG			PA		
ℓ	triang.	squares	polyg.	triang.	squares	polyg.
1	n.c.	n.c.	n.c.	n.c.	n.c.	n.c.
2	0.60	0.43	0.80	0.13	0.42	0.62
3	0.33	0.81	0.93	0.48	0.41	0.67
4	0.83	0.84	0.94	0.94	0.98	1.00
5	0.97	0.99	0.93	0.95	0.93	0.91
vs DOF						
	PG			PA		
ℓ	triang.	squares	polyg.	triang.	squares	polyg.
1	n.c.	n.c.	n.c.	n.c.	n.c.	n.c.
2	-0.35	-0.25	-0.40	-0.08	-0.25	-0.31
3	-0.18	-0.44	-0.43	-0.26	-0.22	-0.31
4	-0.43	-0.44	-0.48	-0.49	-0.51	-0.51
5	-0.49	-0.51	-0.49	-0.48	-0.47	-0.48

Table 3: Estimated H_1–convergence order $p_{\ell,\ell-1}$ vs the mesh diameter h (upper Table), and $q_{\ell,\ell-1}$ vs DOF number (lower Table), raised by VEM when approximating problems PG and PA by uniformly refined meshes. The acronym "n.c." means "not computable": level $\ell = 0$ does not exist.

We performed uniform refinements until level $\ell = 5$. Level $\ell = 6$ gives a "too large" number of elements, i.e. $N_E \gg N_{max}^{(E)}$. Here and in the sequel we assume $N_{max}^{(E)} = 8 \times 10^3$.

For each element E in a given mesh, let $u(\underline{c}_E)$ the exact solution on its center $\underline{c}_E = (c_1^{(E)}, c_2^{(E)})$ of E. Let $\tilde{u}^{(\underline{c}_E)}$ be the corresponding approximate, numerical solution on the center, obtained by projection of the approximate solution values computed on the vertices of E. Analogously, let ∇u be the gradient vector of the exact solution, then

$$\nabla \tilde{u}(\underline{c}_E) = \nabla_m \Pi(\tilde{u}),$$

is the gradient of the numerical solution on the center.

For each given mesh $\mathcal{T}$, featuring N_E elements and N_v vertices, we consider the following error measure:

$$e_{H_1} = \frac{\left(\iint_\Omega \left\| \nabla u(\underline{c}_E) - \nabla \tilde{u}^{(\underline{c}_E)} \right\|^2 dxdy\right)^{1/2}}{\left(\iint_\Omega \left\| \nabla u(\underline{c}_E) \right\|^2 dxdy\right)^{1/2}},$$

Table 1 reports the number of elements in the corresponding uniformly refined meshes. Table 2 shows the corresponding H_1–errors raised by VEM, when attacking either problem PG, or problem PA.

For shortness, let us assume that e is the e_{H_1} error, and h is the mesh diameter. Let us also assume that the following asymptotic convergence relation holds

$$e^{(\ell)} = C\left(h^{(\ell)}\right)^p,$$

for a given p, and a constant C not depending on the refinement level ℓ.

If $D^{(\ell)}$ is the corresponding number of DOF, one has [28]

$$D^{(\ell)} \simeq \frac{1}{h^2}, \quad h \simeq \frac{1}{\sqrt{D^{(\ell)}}},$$
$$e^{(\ell)} = C\left(\frac{1}{\sqrt{D^{(\ell)}}}\right)^p = C\left(D^{(\ell)}\right)^{-p/2}.$$

Hence by defining $q = -p/2$,

$$p_{j,k} = \frac{\log\left(e^{(j)}/e^{(k)}\right)}{\log\left(h^{(j)}/h^{(k)}\right)}, \tag{6}$$

one has the asymptotic relations

$$q_{j,k} = \frac{\log\left(e^{(j)}/e^{(k)}\right)}{\log\left(D^{(j)}/D^{(k)}\right)} \to q \leftarrow -p_{j,k}/2, \tag{7}$$

when $j > k$, $j,k \to +\infty$.

Table 3 shows either the corresponding H_1–convergence order estimation p, or the q estimation. Our code implements linear VEM technique. One can see that p estimations approach 1, when the refinement level increases, as expected. On the other hand, q estimations approach -1/2, confirming that our code displays linear convergence order.

Figure 4 shows our initial triangular mesh, the square one, and the polygon one.

Let us assume now that we exploit our adaptive refinement procedure, by setting $\theta = 0.3$.

The top frame in Figure 5 shows the triangle mesh obtained by ℓ =10 adaptive refinements, when attacking problem PG. The bottom frame shows the ℓ =20 refinement. Note that the mesh was refined exactly where the solution undergoes large variations, as one can see by comparing the frames in Figure 5 with the contour regions shown in Figure 2, which are reported for in the background for easy comparison.

The top frame in Figure 6 shows our refined square mesh at level ℓ =10. In the background, a sketch of the contour regions for problem PG is shown. The bottom frame shows the ℓ =20 adaptively refined mesh, obtained when solving Problem PG. Like when an initial triangle mesh is exploited, the square mesh was refined exactly where the solution undergoes large variations, as one can see by comparing the refined meshes with the contour regions in Figure 2, reported in the background of Figure 6.

The top frame in Figure 7 shows our adaptively refined mesh at level ℓ =10, obtained by the refining our initial polygon mesh. The

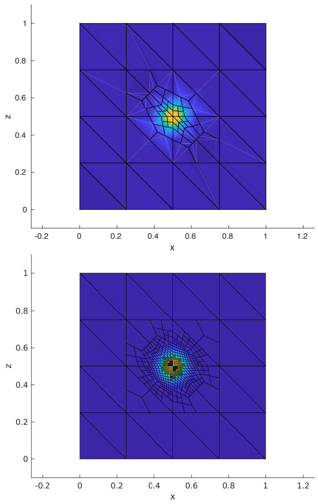

Figure 5: Problem PG, triangular mesh at level $\ell = 10$ (top frame) and at $\ell = 20$ (bottom frame).

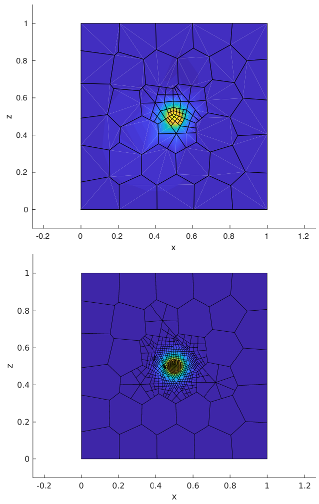

Figure 7: Problem PG, polygonal mesh at level $\ell = 10$ (top frame) and at $\ell = 20$ (bottom frame).

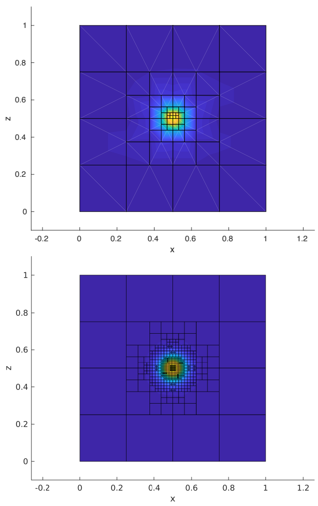

Figure 6: Problem PG, square mesh at level $\ell = 10$ (top frame) and at $\ell = 20$ (bottom frame).

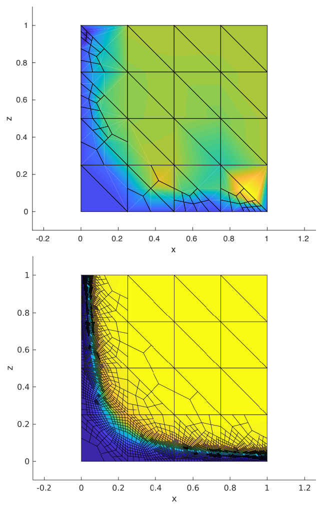

Figure 8: Problem PA, triangle mesh at level $\ell = 10$ (top frame) and at $\ell = 30$ (bottom frame).

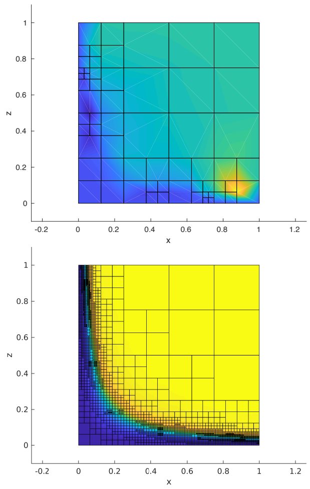

Figure 9: Problem PA, square mesh at level $\ell = 10$ (top frame) and at $\ell = 30$ (bottom frame).

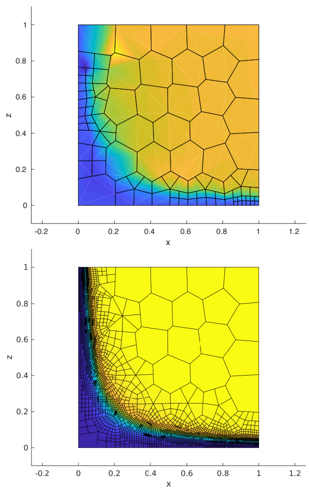

Figure 10: Problem PA, hexagon mesh at level $\ell = 10$ (top frame) and at $\ell = 30$ (bottom frame).

problem	elements	ℓ	Uniform refinements e_{H1}	N_E	N_V
PG	triangles	5	1.41E-01	8192	4225
PG	squares	5	1.27E-01	4096	4225
PG	polygons	5	8.99E-02	8192	16318
PA	triangles	5	1.57E-01	8192	4225
PA	squares	5	1.66E-01	4096	4225
PA	polygons	5	1.20E-01	8192	16318
problem	**elements**	**ℓ**	**Adaptive refinements e_{H1}**	**N_E**	**N_V**
PG	triangles	20	1.07E-01	663	756
PG	squares	20	1.16E-01	634	746
PG	polygons	21	8.36E-01	1080	1231
PA	triangles	25	1.38E-01	1349	1611
PA	squares	24	1.53E-01	904	1101
PA	polygons	27	1.11E-01	2721	3167

Table 4: Best accuracy with uniform refinements compared with adaptive refinements.

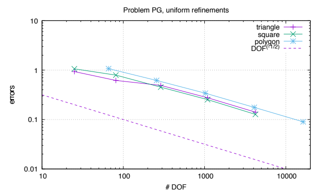

Figure 11: Problem PG, errors raised by uniform refinements. The values of e_{H_1} are shown vs the number of DOF, together with the DOF$^{-1/2}$ line.

bottom frame reports the ℓ =20 refined mesh. Analogously, comforting conclusions can be drawn concerning the adaptive refinements, as for the preceding triangular and square meshes.

Note that only when square elements are exploited, the refined meshes consist of the same type of polygons (squares) as in the initial mesh. Using our peculiar terminology we can say that only "square meshes" are made exclusively by squares.

Let us now focus on problem PA. Figures 8, 9, 10, shows our refinements for triangular, square, and polygonal meshes, respectively. In the background, contour regions for the solution of problem PA are sketched.

Analogous conclusions concerning the refinement regions as for problem PG can be drawn, by comparing the given frames with the contour regions in the background, which are also given in Figure 3. Refinements are performed "only inside domain regions where refinements are positively needed".

Figure 11 shows the behaviors of our error measures, when problem PG is solved by uniformly refined, meshes. One can observe that triangle and square meshes allow for attaining quite the same accuracy, while polygon meshes allow for attaining a slighter larger accuracy. Comparing our convergence lines with DOF$^{-1/2}$ we can confirm that linear convergence speed ($q = -1/2$) is attained.

Table 2 reports the H_1–errors raised when exploiting uniform discretizations.

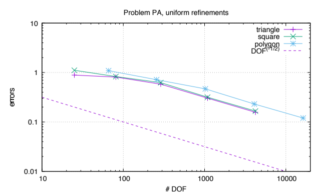

Figure 12: Analogous to Figure 11, concerning Problem PA.

ℓ	triangles				squares				polygons			
	N_E	N_V	e_{H_1}	$q_{\ell,10}$	N_E	N_V	e_{H_1}	$q_{\ell,10}$	N_E	N_V	e_{H_1}	$q_{\ell,10}$
1	32	25	9.33E-01	x	16	25	1.06E+00	x	32	66	1.07E+00	x
2	34	29	8.80E-01	x	22	34	9.34E-01	x	37	73	9.80E-01	x
3	36	33	8.80E-01	x	28	43	8.48E-01	x	40	78	1.40E+00	x
4	40	39	1.85E+00	x	31	47	9.84E-01	x	45	84	6.52E-01	x
5	44	43	7.01E-01	x	34	50	7.41E-01	x	50	89	5.07E-01	x
6	50	53	5.54E-01	x	40	59	7.26E-01	x	53	93	4.19E-01	x
7	56	61	5.15E-01	x	49	72	5.38E-01	x	56	96	4.06E-01	x
8	62	67	4.24E-01	x	52	75	4.09E-01	x	65	109	3.89E-01	x
9	68	77	3.89E-01	x	55	79	3.79E-01	x	77	123	3.72E-01	x
10	80	91	3.45E-01	x	58	82	3.61E-01	x	89	137	3.42E-01	x
11	95	112	3.27E-01	-0.26	73	100	3.09E-01	-0.78	107	158	2.65E-01	-1.79
12	110	129	3.14E-01	-0.27	88	117	2.36E-01	-1.20	139	199	2.33E-01	-1.03
13	134	157	2.48E-01	-0.61	112	145	2.20E-01	-0.87	172	243	2.02E-01	-0.92
14	161	193	2.32E-01	-0.53	145	184	2.02E-01	-0.72	208	279	1.99E-01	-0.76
15	203	241	2.10E-01	-0.51	193	243	1.93E-01	-0.58	256	329	1.64E-01	-0.84
16	257	300	1.99E-01	-0.46	250	312	1.70E-01	-0.56	316	392	1.65E-01	-0.69
17	332	391	1.55E-01	-0.55	310	380	1.43E-01	-0.60	391	484	1.45E-01	-0.68
18	416	480	1.31E-01	-0.58	376	445	1.31E-01	-0.60	526	652	1.31E-01	-0.62
19	527	605	1.23E-01	-0.54	496	574	1.39E-01	-0.49	690	833	1.10E-01	-0.63
20	663	756	1.07E-01	-0.55	634	746	1.16E-01	-0.51	864	1023	9.96E-02	-0.61
21	867	980	9.76E-02	-0.53	838	980	1.01E-01	-0.51	1080	1231	8.36E-02	-0.64
22	1137	1293	8.59E-02	-0.52	1063	1214	8.15E-02	-0.55	1398	1590	8.43E-02	-0.57
23	1455	1641	7.47E-02	-0.53	1309	1448	6.86E-02	-0.58	1803	2043	7.11E-02	-0.58
24	1866	2062	6.70E-02	-0.53	1636	1781	5.80E-02	-0.59	2324	2633	6.33E-02	-0.57
25	2334	2553	5.98E-02	-0.53	2278	2496	5.31E-02	-0.56	2960	3314	5.85E-02	-0.55
26	2991	3278	5.34E-02	-0.52	3049	3358	4.89E-02	-0.54	3710	4116	5.13E-02	-0.56
27	3927	4305	5.05E-02	-0.50	3901	4270	4.27E-02	-0.54	4624	5053	4.77E-02	-0.55
28	5079	5571	4.37E-02	-0.50	4876	5219	3.61E-02	-0.55	5775	6303	4.61E-02	-0.52
29	6456	7027	3.89E-02	-0.50	6172	6546	3.14E-02	-0.56	7113	7796	4.06E-02	-0.53

Table 5: Summary of our numerical results on mesh adaptivity, concerning problem PG. We consider the refined meshes obtained by starting with the initial meshes shown in Figure 4. Our initial triangular mesh is identified by the label “triangles”, our initial uniform square mesh by “squares”, our polygonal mesh obtained by Polymesher software using the label “polygons”. For each refinement step ℓ the number of elements N_E and vertices N_V, the H_1–error e_{H_1}, and the estimated convergence order $q_{\ell,10}$ are shown. The “x” symbol labels those $q_{\ell,10}$ values which are not meaningful.

	triangles				squares				polygons			
ℓ	N_E	N_V	e_{H_1}	$q_{\ell,10}$	N_E	N_V	e_{H_1}	$q_{\ell,10}$	N_E	N_V	e_{H_1}	$q_{\ell,10}$
1	32	25	8.88E-01	x	16	25	1.11E+00	x	32	66	1.09E+00	x
2	34	29	9.22E-01	x	19	30	9.33E-01	x	36	72	1.06E+00	x
3	36	33	9.09E-01	x	22	35	1.08E+00	x	44	83	9.28E-01	x
4	40	41	9.00E-01	x	28	45	1.04E+00	x	52	94	9.36E-01	x
5	44	49	9.08E-01	x	34	53	9.69E-01	x	58	102	8.39E-01	x
6	50	59	1.05E+00	x	37	58	9.00E-01	x	65	112	8.51E-01	x
7	56	67	8.93E-01	x	40	63	9.12E-01	x	75	125	7.43E-01	x
8	65	82	8.59E-01	x	49	76	9.34E-01	x	78	130	6.58E-01	x
9	73	93	8.27E-01	x	55	84	1.10E+00	x	87	142	6.39E-01	x
10	84	107	7.94E-01	x	61	92	8.34E-01	x	99	158	5.51E-01	x
11	98	126	8.45E-01	-0.38	67	100	7.22E-01	-1.73	116	179	5.08E-01	-0.65
12	114	148	6.02E-01	-0.85	76	114	6.32E-01	-1.29	141	210	4.94E-01	-0.38
13	131	169	5.94E-01	-0.63	85	126	6.26E-01	-0.91	144	215	5.41E-01	-0.06
14	149	193	5.71E-01	-0.56	94	136	5.92E-01	-0.88	147	219	4.56E-01	-0.58
15	184	240	4.97E-01	-0.58	106	151	4.92E-01	-1.07	156	231	3.80E-01	-0.98
16	229	290	4.51E-01	-0.57	130	185	3.97E-01	-1.06	177	258	3.29E-01	-1.05
17	262	327	4.23E-01	-0.56	163	228	3.37E-01	-1.00	216	304	3.07E-01	-0.89
18	313	390	3.05E-01	-0.74	199	268	2.99E-01	-0.96	277	378	2.74E-01	-0.80
19	343	426	2.87E-01	-0.74	250	323	2.74E-01	-0.89	360	483	2.45E-01	-0.73
20	405	507	2.43E-01	-0.76	325	415	2.44E-01	-0.82	456	590	2.17E-01	-0.71
21	510	634	2.19E-01	-0.72	412	515	2.19E-01	-0.78	576	733	1.93E-01	-0.68
22	641	789	2.01E-01	-0.69	535	659	1.97E-01	-0.73	754	946	1.73E-01	-0.65
23	815	995	1.70E-01	-0.69	703	878	1.78E-01	-0.68	985	1222	1.55E-01	-0.62
24	1052	1283	1.58E-01	-0.65	904	1101	1.53E-01	-0.68	1277	1555	1.42E-01	-0.59
25	1349	1611	1.38E-01	-0.65	1186	1419	1.36E-01	-0.66	1649	1987	1.31E-01	-0.57
26	1751	2067	1.20E-01	-0.64	1576	1866	1.25E-01	-0.63	2145	2540	1.24E-01	-0.54
27	2288	2681	1.12E-01	-0.61	2089	2450	1.41E-01	-0.54	2721	3167	1.11E-01	-0.53
28	2945	3399	1.05E-01	-0.58	2458	2895	1.20E-01	-0.56	3540	4133	9.62E-02	-0.53
29	3767	4351	1.08E-01	-0.54	3259	3764	9.53E-02	-0.58	4530	5206	9.12E-02	-0.51
30	4616	5320	1.03E-01	-0.52	4135	4752	9.10E-02	-0.56	5667	6514	8.31E-02	-0.51
31	5609	6414	1.03E-01	-0.50	5062	5759	8.98E-02	-0.54	7113	8111	7.61E-02	-0.50
32	6515	7451	9.46E-02	-0.50	5965	6763	7.78E-02	-0.55	n.a.	n.a.	n.a.	n.a.
33	7961	9145	8.84E-02	-0.49	7555	8585	7.86E-02	-0.52	n.a.	n.a.	n.a.	n.a.

Table 6: Analogous to Table 5, concerning problem PA. The “n.a.” shorthand means “not available”: the iteration was not performed, due to our stopping criterion.

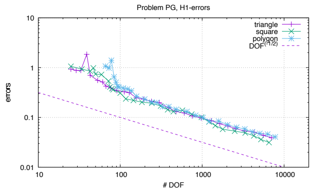

Figure 13: Problem PG, adaptive refinements, H_1-errors vs the DOF number, together with the DOF$^{-1/2}$ line.

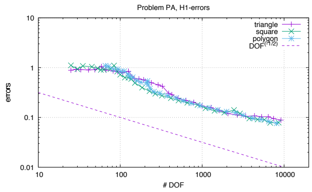

Figure 14: Analogous to Figure 13, concerning Problem PA.

Table 5 summarizes our main numerical results when approximating the solution of problem PG, by VEM, using our adaptive mesh procedure.

Note that our adaptive refinement procedure, with the proposed parameter values, at each level adds few elements, as compared with uniform refinements. Our stop citerion $N_E \gg N_{max}^{(E)}$ is attained by performing a quite larger number of adaptive refinement levels $\ell_A \geq 29$, than for uniform refinements (recall $\ell_U \leq 5$).

Let us roughly assume that the computational cost of our adaptive procedure is proportional to the number DOF, which in linear VEM is equal to N^ℓ, in the adapted mesh at a given refinement level ℓ. Let us consider two mesh levels, one labeled ℓ_U, pertaining to uniform refinements, which counts $N_V^{(\ell_U)}$ vertices. Another, labelled ℓ_A, obtained by adaptive steps, which counts $N_V^{(\ell_A)}$ vertices. Assume that the error $e^{(\ell_A)}$ raised by using the adaptive mesh is smaller that the error $e^{(\ell_U)}$ raised by the unformly refined mesh, i.e. $e^{(\ell_U)} > e^{(\ell_A)}$. Our adaptive procedure can be expected to be computationally efficient if $N_V^{(\ell_U)} \gg N^{(\ell_A)}$. In other words, better accuracy is attained using our adaptive refinement, by expoiting a far smaller number of DOF in the adaptively refined mesh, than in the uniform refinement.

By inspecting Figure 13 one can see the behavior of the H_1–error for Problem "Gauss" (PG), when solved by adaptively refining either a triangle, or square, or Polymesher, initial mesh. The convergence order q approaches -1/2, as graphically confirmed by comparing with the DOF$^{-1/2}$ line, also reported. One can see that when $\ell \geq 10$, say, the error strictly decreases, proportionally to the refinement level ℓ. Each refinement level adds few elements to our meshes, as compared to uniform refinements (see columns 2, 6, 10). One could check that by computing $q_{\ell+1,\ell}$ using formula (7), poor approximations to q are obtained (not shown). In order to display sound approimations, we computed $q_{\ell,10}$ approximations, for $\ell > 10$. They are reported in Table 5, columns 5, 9, 13. Note that, an an example, the $q_{29,10}$ values confirm linear convergence (q =-1/2) both for triangle, and square, and polygonal meshes.

Concerning our approximations to the solution of Problem PA, by examining Table 6 and Figure 14, one can infer the same observations as for PG problem given above.

Le us go back to solving problem PG using an initial mesh consisting of triangles. Let us examine the top frame in Figure 15. One can see that, when comparing uniform and adaptive meshes with quite the same number of DOF, adaptive refinements of our initial triangle mesh allow for higher accuracy than uniform refinements. By inspecting level ℓ =5 in Tables 1, and 2, in order to attain the best accuracy $e_{H_1}^{(U,5)}$=1.41E-1, by uniform refinements of the initial triangle mesh, $N_V^{(U,5)}$=4,225 DOF are required. Comparing with Table 5, one can infer that as few as $N_V^{(A,20)}$=756 $\ll N_E^{(U,5)}$ elements are required in order to achieve $e_{H_1}^{(A,20)}$=1.07E-1 $< e_{H_1}^{(U,5)}$.

Analogous results can be inferred for square and polygon adaptive meshes, by inspecting level ℓ =5 in Tables 1, and 2, together with

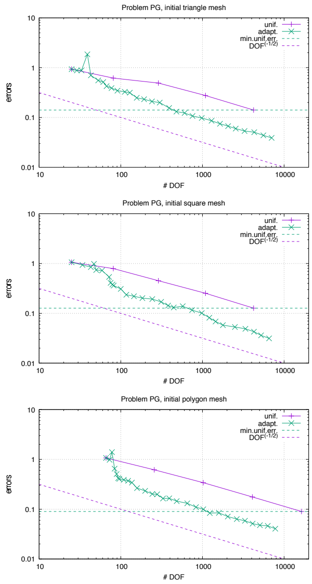

Figure 15: Problem PG, uniform and adaptive refinements. Lines pointing out the minimum errors achievable by uniform refinements are also shown.

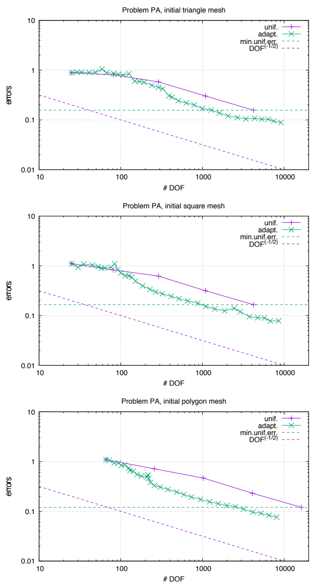

Figure 16: Analogous to Figure 15, concerning Problem PA.

Table 5, and Figure 13.

Table 4 summarizes our results. In all our tests we found that the best attainable accuracy with uniform refinements is attained by exploiting a far smaller number of elements in our adaptively refined meshes.

Summarizing, our adaptive procedure is likely to be really effective when exploiting both triangle meshes, and square ones, as well as polygon ones.

6 Conclusions

The following points are worth mentioning.

- A mesh–adaptive VEM–based procedure was described, implemented, and tested.
- The abstract criterion proposed in [28] for identifying the elements to be refined is worked out for our problem, and the assessment of the involved parameters is performed.
- Our adaptive refinement procedure refines any initial mesh only where the solution undergoes large variations.
- Our adaptive refinement procedure allows for attaining a better accuracy than the best one that can be reached by uniform refinements, by using a far smaller number of DOF.

Acknowledgments

The Authors thank Gianmarco Manzini for his valuable suggestions, which greatly improved their work. Some computations were performed on SCSCF, a multiprocessor cluster system owned by Universitá Ca' Foscari Venezia.

References

[1] P. F. Antonietti, L. Beirão da Veiga, D. Mora, and M. Verani. A stream virtual element formulation of the stokes problem on polygonal meshes. *SIAM Journal on Numerical Analysis*, 52(1):386–404, 2014.

[2] P. F. Antonietti, L. Beirão da Veiga, S. Scacchi, and M. Verani. A C^1 virtual element method for the Cahn-Hilliard equation with polygonal meshes. *SIAM Journal on Numerical Analysis*, 54(1):34–56, 2016.

[3] P. F. Antonietti, M. Bruggi, S. Scacchi, and M. Verani. On the Virtual Element Method for topology optimization on polygonal meshes: A numerical study. *Computers and Mathematics with Applications*, 74(5):1091–1109, 2017.

[4] P. F. Antonietti, S. Giani, and P. Houston. *hp*–Version Composite Discontinuous Galerkin Methods for elliptic problems on complicated domains. *SIAM J. Sci. Comput.*, 35(3):A1417–A1439, 2013.

[5] P. F. Antonietti, G. Manzini, and M. Verani. The fully nonconforming Virtual Element method for biharmonic problems. *Mathematical Models & Methods in Applied Sciences*, 28(2), 2018.

[6] P. F. Antonietti, L. Mascotto, and M. Verani. A multigrid algorithm for the *p*–version of the Virtual Element method. *ESAIM: Mathematical Modelling and Numerical Analysis*, 2018. In press.

[7] B. Ayuso de Dios, K. Lipnikov, and G. Manzini. The nonconforming virtual element method. *ESAIM: Mathematical Modelling and Numerical Analysis*, 50(3):879–904, 2016.

[8] F. Bassi, L. Botti, and A. Colombo. Agglomeration-based physical frame DG discretizations: An attempt to be mesh free. *Math. Models Methods Appl. Sci.*, 24(8):1495–1539, 2014.

[9] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. In A. Iserles, editor, *Acta Numerica*, pages 1–102. Cambridge University Press, Cambridge, 2001.

[10] L. Beirão da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L. D. Marini, and A. Russo. Basic principles of virtual element methods. *Mathematical Models & Methods in Applied Sciences*, 23:119–214, 2013.

[11] L. Beirão da Veiga, F. Brezzi, and L. D. Marini. Virtual elements for linear elasticity problems. *SIAM Journal on Numerical Analysis*, 51(2):794–812, 2013.

[12] L. Beirão da Veiga, F. Brezzi, L. D. Marini, and A. Russo. Mixed virtual element methods for general second order elliptic problems on polygonal meshes. *ESAIM: Mathematical Modelling and Numerical Analysis*, 50(3):727–747, 2016.

[13] L. Beirão da Veiga, A. Chernov, L. Mascotto, and A. Russo. Basic principles of hp virtual elements on quasiuniform meshes. *Mathematical Models & Methods in Applied Sciences*, 26(8):1567–1598, 2016.

[14] L. Beirão da Veiga, K. Lipnikov, and G. Manzini. Arbitrary order nodal mimetic discretizations of elliptic problems on polygonal meshes. *SIAM Journal on Numerical Analysis*, 49(5):1737–1760, 2011.

[15] L. Beirão da Veiga, K. Lipnikov, and G. Manzini. *The Mimetic Finite Difference Method*, volume 11 of *MS&A. Modeling, Simulations and Applications*. Springer, I edition, 2014.

[16] L. Beirão da Veiga, C. Lovadina, and D. Mora. A virtual element method for elastic and inelastic problems on polytope meshes. *Computer Methods in Applied Mechanics and Engineering*, 295:327–346, 2015.

[17] L. Beirão da Veiga and G. Manzini. A virtual element method with arbitrary regularity. *IMA Journal on Numerical Analysis*, 34(2):782–799, 2014. DOI: 10.1093/imanum/drt018, (first published online 2013).

[18] L. Beirão da Veiga and G. Manzini. Residual a posteriori error estimation for the virtual element method for elliptic problems. *ESAIM: Mathematical Modelling and Numerical Analysis*, 49:577–599, 2015.

[19] L. Beirão da Veiga, G. Manzini, and M. Putti. Post-processing of solution and flux for the nodal mimetic finite difference method. *Numerical Methods for PDEs*, 31(1):336–363, 2015.

[20] L. Beirão da Veiga, F. Brezzi, L. D. Marini, and A. Russo. The hitchhiker's guide to the Virtual Element Method. *Mathematical Models and Methods in Applied Sciences*, 24(8):1541–1573, 2014.

[21] L. Beirão da Veiga, A. Chernov, L. Mascotto, and A. Russo. Exponential convergence of the *hp* virtual element method in presence of corner singularities. *Numerische Mathematik*, 138, 2018.

[22] M. F. Benedetto, S. Berrone, S. Pieraccini, and S. Scialò. The virtual element method for discrete fracture network simulations. *Computer Methods in Applied Mechanics and Engineering*, 280(0):135 – 156, 2014.

[23] S. Berrone, A. Borio, and G. Manzini. SUPG stabilization for the nonconforming virtual element method for advection-diffusion-reaction equations. *Computer Methods in Applied Mechanics and Engineering*, 340:500–529, 2018.

[24] F. Brezzi, R. S. Falk, and L. D. Marini. Basic principles of mixed virtual element methods. *ESAIM. Mathematical Modelling and Numerical Analysis*, 48(4):1227–1240, 2014.

[25] F. Brezzi and L. D. Marini. Virtual element methods for plate bending problems. *Computer Methods in Applied Mechanics and Engineering*, 253:455–462, 2013.

[26] A. Cangiani, Z. Dong, and E. Georgoulis. *hp*-version space-time discontinuous Galerkin methods for parabolic problems on prismatic meshes. *SIAM Journal on Scientific Computing*, 39(4):A1251–A1279, 2017.

[27] A. Cangiani, E. Georgoulis, and P. Houston. *hp*-version discontinuous Galerkin Methods on polygonal and polyhedral meshes. *Math. Models Methods Appl. Sci.*, 24(10):2009–2041, 2014.

[28] A. Cangiani, E. H. Georgoulis, T. Pryer, and O. J. Sutton. A posteriori error estimates for the virtual element method. *Numerische Mathematik*, pages 1–37, 2017.

[29] A. Cangiani, V. Gyrya, G. Manzini, and O. Sutton. Chapter 14: Virtual element methods for elliptic problems on polygonal meshes. In K. Hormann and N. Sukumar, editors, *Generalized Barycentric Coordinates in Computer Graphics and Computational Mechanics*, pages 1–20. CRC Press, Taylor & Francis Group, 2017.

[30] A. Cangiani, G. Manzini, A. Russo, and N. Sukumar. Hourglass stabilization of the virtual element method. *International Journal on Numerical Methods in Engineering*, 102(3-4):404–436, 2015.

[31] A. Cangiani, G. Manzini, and O. Sutton. Conforming and nonconforming virtual element methods for elliptic problems. *IMA Journal on Numerical Analysis*, 37:1317–1354, 2017. (online August 2016).

[32] C. Carstensen, M. Eigel, R. H. W. Hoppe, and C. Löbhard. A review of unified a posteriori finite element error control. *Numer. Math. Theor. Meth. Appl.*, pages 509–558, 2012.

[33] B. Cockburn, D. A. Di Pietro, and A. Ern. Bridging the Hybrid High-order and Hybridizable Discontinuous Galerkin Methods. *ESAIM Math. Model. Numer. Anal.*, 50(3):635–650, 2016.

[34] D. A. Di Pietro, J. Droniou, and G. Manzini. Discontinuous skeletal gradient discretisation methods on polytopal meshes. *Journal of Computational Physics*, 355:397–425, 2018.

[35] D. A. Di Pietro, A. Ern, and S. Lemaire. An arbitrary-order and compact-stencil discretization of diffusion on general meshes based on local reconstruction operators. *Comput. Methods Appl. Math.*, 14(4):461–472, 2014.

[36] W. Dörfler. A convergent adaptive algorithm for poisson's equation. *SIAM J. Numer. Anal.*, 33(3):1106–1124, June 1996.

[37] J. Droniou. Finite volume schemes for diffusion equations: Introduction to and review of modern methods. *Mathematical Models and Methods in Applied Sciences*, 24(08):1575–1619, 2014.

[38] J. Droniou, R. Eymard, T. Gallouet, and R. Herbin. Gradient schemes: a generic framework for the discretisation of linear, nonlinear and nonlocal elliptic and parabolic equations. *Math. Models Methods Appl. Sci.*, 23(13):2395–2432, 2013.

[39] S. Funken, D. Praetorius, and P. Wissgott. Efficient implementation of adaptive P1-FEM in Matlab. *Computational Methods in Applied Mathematics*, 11(4), 2011.

[40] F. Gardini and G. Vacca. Virtual element method for second order elliptic eigenvalue problems. *IMA Journal on Numerical Analysis*, 2017. Preprint, arXiv:1610.03675.

[41] T. Grätsch and K.-J. Bathe. A posteriori error estimation techniques in practical finite element analysis. *Computers and Structures*, pages 235–265, 2005.

[42] V. Gyrya, K. Lipnikov, and G. Manzini. The arbitrary order mixed mimetic finite difference method for the diffusion equation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 50(3):851–877, 2016.

[43] B. B. T. Kee, G. R. Liu, G. Y. Zhang, and C. Lu. A residual based error estimator using radial basis functions. *Finite Elem. Anal. Des.*, 44(9-10):631–645, June 2008.

[44] K. Lipnikov and G. Manzini. A high-order mimetic method on unstructured polyhedral meshes for the diffusion equation. *Journal of Computational Physics*, 272:360–385, 2014.

[45] K. Lipnikov, G. Manzini, J. D. Moulton, and M. Shashkov. The mimetic finite difference method for elliptic and parabolic problems with a staggered discretization of diffusion coefficient. *Journal of Computational Physics*, 305:111 – 126, 2016.

[46] K. Lipnikov, G. Manzini, and M. Shashkov. Mimetic finite difference method. *Journal of Computational Physics*, 257 – Part B:1163–1227, 2014. Review paper.

[47] G. Manzini, K. Lipnikov, J. D. Moulton, and M. Shashkov. Convergence analysis of the mimetic finite difference method for elliptic problems with staggered discretizations of diffusion coefficients. *SIAM Journal on Numerical Analysis*, 55(6):2956–2981, 2017.

[48] G. Manzini, A. Russo, and N. Sukumar. New perspectives on polygonal and polyhedral finite element methods. *Mathematical Models & Methods in Applied Sciences*, 24(8):1621–1663, 2014.

[49] R. H. Nochetto, K. G. Siebert, and A. Veeser. Theory of adaptive finite element methods: An introduction. In R. DeVore and A. Kunoth, editors, *Multiscale, Nonlinear and Adaptive Approximation: Dedicated to Wolfgang Dahmen on the Occasion of his 60th Birthday*, pages 409–542. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[50] S. Rjasanow and S. Weißer. Higher order BEM-based FEM on polygonal meshes. *SIAM J. Numer. Anal.*, 50(5):2357–2378, 2012.

[51] N. Sukumar and A. Tabarraei. Conforming Polygonal Finite Elements. *Internat. J. Numer. Methods Engrg.*, 61(12):2045–2066, 2004.

[52] A. Tabarraei and N. Sukumar. Extended Finite Element Method on polygonal and quadtree meshes. *Comput. Methods Appl. Mech. Engrg.*, 197(5):425–438, 2008.

[53] C. Talischi, G. H. Paulino, A. Pereira, and I. F. M. Menezes. `PolyMesher`: a general-purpose mesh generator for polygonal elements written in Matlab. *Struct. Multidisc. Optim.*, 45(3):309–328, 2012.

[54] O. Čertík, F. Gardini, G. Manzini, and G. Vacca. The virtual element method for eigenvalue problems with potential terms on polytopic meshes. *Applications of Mathematics*, 63(3):333–365, 2018.

[55] G. Vacca. An H^1-conforming virtual element for Darcy and Brinkman equations. *Mathematical Models & Methods in Applied Sciences*, 28(1):159–194, 2018.