Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 347 (2019) 21–58
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A simple and effective gradient recovery scheme and *a posteriori* error estimator for the Virtual Element Method (VEM)

Heng Chi[a], Lourenço Beirão da Veiga[b], Glaucio H. Paulino[a,*]

[a] *School of Civil and Environmental Engineering, Georgia Institute of Technology, 790 Atlantic Drive, Atlanta, GA, 30332, USA*
[b] *Dipartimento di Matematica e Applicazioni, Università di MilanoBicocca, Via Cozzi 53, I-20153, Milano, Italy*

Received 13 March 2018; received in revised form 5 August 2018; accepted 10 August 2018
Available online 14 September 2018

Abstract

This paper introduces a general recovery-based *a posteriori* error estimation framework for the Virtual Element Method (VEM) of arbitrary order on general polygonal/polyhedral meshes. The framework consists of a gradient recovery scheme and *a posteriori* error estimator based on the recovered displacement gradient. A skeletal error, which accurately mimics the behavior of the L^2 error of the displacement gradient by only sampling the displacement gradient on the mesh skeleton, is introduced. Through numerical studies on various polygonal/polyhedral meshes, we demonstrate that the proposed gradient recovery scheme can produce considerably more accurate displacement gradient than the original VEM solutions, and that the *a posteriori* error estimator is able to accurately capture both local and global errors without the knowledge of exact solutions. We also demonstrate the use of the VEM skeletal error estimators to guide adaptive mesh refinement.

Keywords: Virtual element method (VEM); Gradient recovery; Error estimation; Higher order; Element skeleton; Mesh skeleton

1. Introduction

The virtual element method (VEM) is a recent generalization of the finite element method (FEM) that is capable of efficiently handling general polygonal/polyhedral meshes [1]. This feature makes VEM a suitable framework for mesh adaptations (e.g. adaptive refinement and coarsening). To realize this potential, we propose a general recovery-based *a posteriori* error estimation for VEM of arbitrary order on general polygonal/polyhedral meshes, and demonstrate the idea in the context of linear elasticity. More specifically, we first introduce an efficient patch-based gradient recovery scheme for VEM, which reconstructs a more accurate displacement gradient field by least square fitting the displacement degrees of freedom (DOFs) over each selected patch of elements in the mesh. Based on the recovered gradient, we further introduce a simple yet effective recovery-based *a posteriori* error estimator. *To avoid explicit constructions of the VEM basis functions, this error estimator evaluates the displacement error on the skeleton of*

* Corresponding author.
E-mail addresses: hchi6@gatech.edu (H. Chi), lourenco.beirao@unimib.it (L. Beirão da Veiga), paulino@gatech.edu (G.H. Paulino).

https://doi.org/10.1016/j.cma.2018.08.014

the mesh, which mimics the behavior of the L^2 norm of the displacement gradient error. We conduct thorough numerical studies to assess performance of the proposed gradient recovery scheme and error estimator. Through numerical examples, the proposed gradient recovery scheme and error estimator are shown to be accurate on various polygonal/polyhedral meshes and with different types of displacement solutions.

The remainder of this paper is organized as follows. Section 2 provides motivations for the paper and summarizes related work in both VEM and FEM literature. Section 3 reviews the VEM framework for 2D and 3D linear elasticity problems. In Section 4, we introduce the gradient recovery scheme for lower- and higher-order VEM, and *a posteriori* error estimator based on the recovered displacement gradient. In Section 5, some theoretical estimates of the recovered displacement gradient, which provide insights into the proposed gradient recovery scheme, are provided. Several numerical assessments are presented in Section 6 to demonstrate the accuracy of the gradient recovery scheme and *a posteriori* error estimator on various polygonal/polyhedral meshes and with different displacement solution types. Section 7 addresses the use of the VEM skeletal error estimators to guide adaptive mesh refinement. Section 8 contains several concluding remarks and future research directions. An Appendix complements the paper, which presents a Super-convergent Patch Recovery (SPR) type scheme.

2. Motivation and related work

Interest in numerical methods that can handle polygonal/polyhedral meshes has been growing in the fields of mathematics and engineering, see [2–14] for a minimal sample of references. Among them, VEM is an emerging method, first introduced in [1], as a generalization of FEM in the family of Galerkin methods. In the VEM, the basis functions of the local space are defined implicitly through a suite set of partial differential equations (PDEs). Unlike FEM, this set of PDEs is never solved throughout the approximation. Instead, we apply integration by parts to compute suitable projections of the basis functions on to polynomials. [1,15]. Those projections are then used in the VEM approximation to ensure its consistency together with a suitable stabilization term, which is needed in order to avoid the appearance of hourglass modes. As a result, only numerical quadratures for polynomials (and not for more complex functions) are required in the VEM. These unique features allow the VEM to handle general polygonal/polyhedral meshes (including non-convex ones [16,17]) and to construct various types of elements, including $H(\text{div})$ and $H(\text{curl})$ conforming elements [18]. The VEM has undergone substantial developments and has been successfully applied to a wide range of problems. For conciseness, we only focus on the literature of VEM for structural mechanics here. In structural mechanics, the VEM has been introduced for linear elasticity problems [19–21], small deformation nonlinear elastic and inelastic problems [22–25], finite deformation elasticity [16,26] and elasto-plasticity problems [27], plate bending problems [28–31] and contact problems [32].

Among various features of the VEM, the flexibility in dealing with general polygonal/polyhedral meshes makes it appealing for adaptivity. For example, by introducing hanging nodes, adaptive mesh refinement strategies can be made more efficient with polygonal/polyhedral elements because it only requires local modifications to the mesh [2]. On the other hand, the shape generality of polygonal/polyhedral elements (especially the concave ones) enables easier element agglomeration schemes for mesh coarsening [33,34]. To realize the full potential of VEM in adaptivity, development of *a posteriori* error estimator is essential — this is the focus of our work.

A posteriori error estimation is a classic topic in FEM with a vast literature, and it is typically categorized into many types: here we consider the recovery-based error estimation type, see e.g. [35–42]. For more details about the error estimation in FEM, the interested readers are referred to references [41,43] and the references within. Recovery-based error estimators (although supported by a less extended theoretical background with respect to other methods) are often preferable in practical applications because of its simple structure, easy implementation, and effectiveness in predicting errors. The recovery-based error estimators require an additional post processing procedure to obtain recovered solutions (typically of gradient type), which are more accurate than the original ones. Among various recovery techniques in the literature, the most notable one is the super-convergent patch recovery (SPR) developed by Zienkiewicz and Zhu [35,36,44]. The SPR has been adopted in [45] to develop error estimators for polygonal and polyhedral FEM. Other techniques also include the recovery by equilibrium in patches (REP) [40,46] and the polynomial preserving recovery (PPR) [38,47–49]. Comparing to FEM, developing error estimations in the VEM framework is a more involved task because the basis functions of the local VEM space are unknown in the interior of elements. Nevertheless, there exist some error estimators in the literature for C^0 and C^1 VEMs [50–53], but all of them are of residual type.

In this work, we outline a general recovery-based *a posteriori* error estimation framework for H^1 conforming VEM of arbitrary order on general polygonal/polyhedral meshes. For the kth order VEM, a polynomial of order $k+1$ is obtained by least square fit of the displacement DOFs over each patch. The recovered displacement gradients on the sample points (i.e. vertices and edge nodes) in that patch are then taken as the gradient of that polynomial evaluated at those points. Based on the recovered displacement gradient, an error estimator is obtained through a skeletal error, which evaluates the displacement error on the skeleton of the mesh. In the proposed framework, because only displacement DOFs are used in the fitting process and the errors are selectively evaluated on the mesh skeleton, then we avoid the difficulty of considering the explicit values of the VEM basis functions in the interior of elements. Through numerical examples, the proposed error estimation scheme is shown to be accurate for lower- and higher-order VEMs on various polygonal/polyhedral meshes and with different types of displacement solutions (e.g. smooth displacement fields, displacements with sharp gradients, and ones containing singularities). For linear VEM, the accuracy and effectiveness of the proposed error estimation framework are further demonstrated by comparing it with a SPR-type error estimation as outlined in the Appendix.

3. VEM for linear elasticity

In this section, the VEM framework for linear elasticity problem is reviewed. Consider a solid in its stress-free state that occupies a domain $\Omega \subset \mathbb{R}^d$ with d being the dimension. The solid is subjected to a prescribed displacement field, $\mathbf{u}^0$, on one portion of the solid boundary $\Gamma^{\mathbf{u}}$ and a traction $\mathbf{t}$ on the other portion of the solid boundary $\Gamma^{\mathbf{t}}$, such that $\Gamma^{\mathbf{u}} \cup \Gamma^{\mathbf{t}} = \partial\Omega$ and $\Gamma^{\mathbf{u}} \cap \Gamma^{\mathbf{t}} = \emptyset$. In addition, a body force $\mathbf{f}$ is applied in the interior of Ω.

The governing equations of linear elasticity are:

$$\begin{aligned} \operatorname{div}(\mathbf{C}\boldsymbol{\epsilon}(\mathbf{u})) + \mathbf{f} &= \mathbf{0} \text{ in } \Omega \\ \mathbf{u} &= \mathbf{u}^0 \text{ on } \Gamma^{\mathbf{u}} \\ \mathbf{C}\boldsymbol{\epsilon}(\mathbf{u}) \cdot \mathbf{n} &= \mathbf{t} \text{ on } \Gamma^{\mathbf{t}}, \end{aligned} \tag{1}$$

where $\mathbf{C}$ is the elasticity modulus tensor, which possess major and minor symmetries, i.e. $C_{ijkl} = C_{klij} = C_{jikl} = C_{ijlk}$. Additionally, $\boldsymbol{\epsilon}(\mathbf{u})$ is the linearized strain tensor:

$$\boldsymbol{\epsilon}(\mathbf{u}) = \frac{1}{2}\left[(\nabla\mathbf{u})^T + \nabla\mathbf{u}\right], \tag{2}$$

where ∇ stands for the gradient operator. The above governing equations for linear elasticity can be recast in variational form, which consists of finding $\mathbf{u}$ among the space of kinematically admissible displacements such that

$$\int_\Omega \boldsymbol{\epsilon}(\mathbf{v}) : [\mathbf{C}\boldsymbol{\epsilon}(\mathbf{u})]\, \mathrm{d}\mathbf{x} = \int_\Omega \mathbf{f}\cdot\mathbf{v}\mathrm{d}\mathbf{x} + \int_{\Gamma^{\mathbf{t}}} \mathbf{t}\cdot\mathbf{v}\mathrm{d}s \quad \forall \mathbf{v} \in \mathcal{K}^0, \tag{3}$$

where the space $\mathcal{K}^0$ denotes the kinematically admissible displacements that vanish on $\Gamma^{\mathbf{u}}$.

3.1. Virtual elements on arbitrary meshes

In this subsection, we review the definition of the local spaces of virtual elements in 2D and 3D [19,54,55]. As a declaration of notation, in the following presentation we use F and P to denote polygons (an element in 2D or a face in 3D) and polyhedrons, respectively. Moreover, we use e to represent a generic edge in the mesh. Whenever we have definitions that are independent of dimensions, we use E as a generic element (i.e. generic polytope).

3.1.1. 2D virtual elements

Before stating the definition of the local virtual space $\mathcal{V}_k(F)$ for 2D virtual elements, we first introduce a preliminary virtual space denoted by $\widetilde{\mathcal{V}}_k(F)$ as follows

$$\widetilde{\mathcal{V}}_k(F) \doteq \left\{ v \in H^1(F) : v|_e \in \mathcal{P}_k(e)\, \forall e \in \partial F,\ \Delta v \in \mathcal{P}_k(F) \right\}, \tag{4}$$

where Δ stands for the Laplacian operator and $\mathcal{P}_k(\cdot)$ is the space of polynomial functions whose orders are less than or equal to k. By definition, the space $\widetilde{\mathcal{V}}_k(F)$ contains functions whose Laplacians in the interior of F and boundary

values are both kth order polynomials. It is immediate that $\mathcal{P}_k(F) \subseteq \widetilde{\mathcal{V}}_k(F)$. Furthermore, for any v in $\widetilde{\mathcal{V}}_k(F)$, by knowing the following three sets of information:

- the values of v at vertices $\mathbf{x}_v$: $v(\mathbf{x}_v)$, $\quad \forall \mathbf{x}_v \in F$; (5)
- the values of v at $\mathbf{x}_e^\ell$: $v(\mathbf{x}_e^\ell)$, $\quad \forall e \in \partial F, \ell = 1, \ldots, k-1$; (6)
- the moments of v up to order $k-2$: $\displaystyle\int_F v p_{k-2} \mathrm{d}\mathbf{x} \quad \forall p_{k-2} \in \mathcal{P}_{k-2}(F)$, (7)

where $\mathbf{x}_e^\ell$ is the ℓth internal integration point of the Gauss–Lobatto rule of order $2k-1$ on edge e, one can compute a projection $\Pi_F^\nabla v \in \mathcal{P}_k(F)$ such that

$$\begin{cases} \displaystyle\int_F \nabla(\Pi_F^\nabla v) \cdot \nabla p_k \mathrm{d}\mathbf{x} = \int_F \nabla v \cdot \nabla p_k \mathrm{d}\mathbf{x} \quad \forall p_k \in \mathcal{P}_k(F) \\ \displaystyle\sum_{\mathbf{x}_v \in F} \Pi_F^\nabla v(\mathbf{x}_v) = \sum_{\mathbf{x}_v \in E} v(\mathbf{x}_v) \qquad \text{if } k = 1 \\ \displaystyle\int_F \Pi_F^\nabla v \mathrm{d}\mathbf{x} = \int_F v \mathrm{d}\mathbf{x} \qquad \text{if } k \geq 2 \end{cases}. \tag{8}$$

In fact, the computability of the projection $\Pi_F^\nabla v$ when (5)–(7) are given becomes more apparent by applying an integration by parts to the first condition of Eq. (8).

Having introduced the preliminary virtual space $\widetilde{\mathcal{V}}_k(F)$, we formally define the 2D local virtual space $\mathcal{V}_k(F)$ as

$$V_k(F) \doteq \left\{ v \in \widetilde{\mathcal{V}}_k(F) : \int_F v q \mathrm{d}\mathbf{x} = \int_F (\Pi_F^\nabla v) q \mathrm{d}\mathbf{x} \quad \forall q \in (\mathcal{P}_k / \mathcal{P}_{k-2}(F)) \right\}, \tag{9}$$

where $(\mathcal{P}_k / \mathcal{P}_{k-2}(F))$ stands for the polynomial space in $\mathcal{P}_k(F)$ that are L^2 orthogonal to $\mathcal{P}_{k-2}(F)$. By definition, we have $\mathcal{P}_k(F) \subseteq \mathcal{V}_k(F) \subseteq \widetilde{\mathcal{V}}_k(F)$ and, moreover, the sets of information of v in (5)–(7) constitute a complete set of DOFs for $\mathcal{V}_k(F)$.

Remark 3.1. Using the given set of DOFs of v in (5)–(7) and, according to (9), we can exactly compute $\int_F v p_k \mathrm{d}\mathbf{x}$ for any $p_k \in \mathcal{P}_k(F)$. As will be discussed later, this is an important property of the local virtual space $\mathcal{V}_k(F)$ that allows us to compute a L^2 projection of $\mathcal{V}_k(F)$ onto $\mathcal{P}_k(F)$ [55].

3.1.2. 3D virtual elements

For the 3D case, we restrict our attentions to polyhedrons with planar faces. Again, as a preliminary space, $\widetilde{\mathcal{V}}_k(P)$ is introduced as

$$\widetilde{\mathcal{V}}_k(P) \doteq \left\{ v \in H^1(P) : v|_F \in \mathcal{V}_k(F) \ \forall F \in \partial P, \ \ \Delta v \in \mathcal{P}_k(P) \right\}. \tag{10}$$

In the above definition, we specify that the functions in $\widetilde{\mathcal{V}}_k(P)$ belong to the 2D virtual space $\mathcal{V}_k(F)$ on each face F of polyhedron P. For any v in $\widetilde{\mathcal{V}}_k(P)$, the obvious 3D counterpart of the projection $\Pi_P^\nabla v$ defined in (8) of v onto $\mathcal{P}_k(P)$ can be uniquely determined and computed provided that we have the following information on v (see [54]):

- the values of v at vertices $\mathbf{x}_v$: $v(\mathbf{x}_v)$, $\quad \forall \mathbf{x}_v \in P$; (11)
- the values of v at $\mathbf{x}_e^\ell$: $v(\mathbf{x}_e^\ell)$, $\quad \forall e \in \partial P, \ell = 1, \ldots, k-1$; (12)
- the moments of v up to order $k-2$ on face F: $\displaystyle\int_F v p_{k-2} \mathrm{d}s \quad \forall p_{k-2} \in \mathcal{P}_{k-2}(F)$ and $\forall F \in \partial P$; (13)
- the moments of v up to order $k-2$ in P: $\displaystyle\int_P v p_{k-2} \mathrm{d}\mathbf{x} \quad \forall p_{k-2} \in \mathcal{P}_{k-2}(P)$. (14)

As compared to the 2D case, an additional set of information of v is needed in 3D, i.e. (13), which includes the moments of v up to order $k-2$ on every face of the element. In particular, by examining the first condition of (8), also in the 3D counterpart we can apply integration by parts to the term on its right hand side, which yields

$$\int_P \nabla v \cdot \nabla p_k \mathrm{d}\mathbf{x} = \int_{\partial P} v \, (\nabla p_k \cdot \mathbf{n}) \mathrm{d}s - \int_P v \, \Delta p_k \mathrm{d}\mathbf{x}. \tag{15}$$

The terms in the above relation can be exactly computed. On one hand, the second term on the right hand side of the above relation is known due to (14). On the other hand, because v belongs to $\mathcal{V}_k(F)$ on each face $F \in P$, then the first term on the right hand side of the above relation is also computable by combining (13) and (9) (also see Remark 3.1).

With $\widetilde{\mathcal{V}}_k(P)$, we are ready to define the virtual space $\mathcal{V}_k(P)$ for 3D virtual elements as

$$\mathcal{V}_k(P) \doteq \left\{ v \in \widetilde{\mathcal{V}}_k(P) : \int_P vq\,\mathrm{d}\mathbf{x} = \int_P (\Pi_P^\nabla v) q\,\mathrm{d}\mathbf{x} \quad \forall q \in (\mathcal{P}_k/\mathcal{P}_{k-2}(P)) \right\}. \tag{16}$$

Again, it can be shown that (11)–(14) constitute a complete set of DOFs for $\mathcal{V}_k(P)$ [54].

Remark 3.2. For a given element E in 2D or 3D, although the projection operator Π_E^∇ is defined as from $\widetilde{\mathcal{V}}_k(E)$ onto $\mathcal{P}_k(E)$, we can also define the projection Π_E^∇ from $\mathcal{V}_k(E)$ onto $\mathcal{P}_k(E)$ using the same set of definitions (8). For any v in $\mathcal{V}_k(E)$, this projection is computable using only the DOFs of v, i.e. (5)–(7) in 2D or (11)–(14) in 3D.

3.1.3. Two projection operators

In VEM approximations, two L^2 projections are utilized, which respectively project a given function $v \in \mathcal{V}_k(E)$ and its gradient onto polynomial functions. The first projection, denoted as $\Pi_k^0 : \mathcal{V}_k(E) \to \mathcal{P}_k(E)$, is defined for any given v in $\mathcal{V}_k(E)$ as

$$\int_E \Pi_k^0 v\, p_k\,\mathrm{d}\mathbf{x} = \int_E v\, p_k\,\mathrm{d}\mathbf{x}\, \forall p_k \in \mathcal{P}_k(E). \tag{17}$$

Because any $p_k \in \mathcal{P}_k(E)$ can always be decomposed as $p_k = p_{k-2} + q_k$ with $p_{k-2} \in \mathcal{P}_{k-2}(E)$ and $q_k \in (\mathcal{P}_k/\mathcal{P}_{k-2}(E))$, then we can express the right hand side of above expression as

$$\int_E v\, p_k\,\mathrm{d}\mathbf{x} = \int_E v(p_{k-2} + q_k)\,\mathrm{d}\mathbf{x} = \int_E v p_{k-2}\,\mathrm{d}\mathbf{x} + \int_E (\Pi_E^\nabla v) q_k\,\mathrm{d}\mathbf{x}. \tag{18}$$

This shows that the term $\int_E v\, p_k\,\mathrm{d}\mathbf{x}$ can be exactly computed using only the DOFs of v in both 2D and 3D and, thus, the same holds for the projection $\Pi_k^0 v$.

Remark 3.3. For linear and quadratic virtual elements, i.e. $k = 1$ and $k = 2$, it is shown in [55] that the projection $\Pi_k^0 v$ coincides with $\Pi_k^\nabla v$ for any $v \in \mathcal{V}_1(E)$. For virtual elements of order $k \geq 3$, however, projections $\Pi_k^0 v$ and $\Pi_E^\nabla v$ are different.

The second projection, which is utilized to project ∇v, is denoted as $\Pi_{k-1}^0 : [L^2(E)]^d \to [\mathcal{P}_{k-1}(E)]^d$ for a kth order virtual elements. For a given v in $\mathcal{V}_k(E)$, the projection $\Pi_{k-1}^0 \nabla v$ is defined as

$$\int_E \Pi_{k-1}^0 \nabla v \cdot \mathbf{p}_{k-1}\,\mathrm{d}\mathbf{x} = \int_E \nabla v \cdot \mathbf{p}_{k-1}\,\mathrm{d}\mathbf{x} \quad \forall \mathbf{p}_{k-1} \in [\mathcal{P}_{k-1}(E)]^d. \tag{19}$$

Applying integration by part to the right hand side of the above expression gives

$$\int_E \nabla v \cdot \mathbf{p}_{k-1}\,\mathrm{d}\mathbf{x} = \int_{\partial E} v\, \mathbf{p}_{k-1} \cdot \mathbf{n}\,\mathrm{d}s - \int_E v \mathrm{div} \mathbf{p}_{k-1}\,\mathrm{d}\mathbf{x}, \tag{20}$$

which can is also computable in both 2D and 3D using the DOFs of v. More specifically, the second term on the right hand side of the above expression is immediately given by DOFs (7) and (14). Moreover, in the 2D case, the first term on the right hand side of (20) can be exactly integrated using the Gauss–Lobatto rule of order $2k-1$ on each edge using the sets of DOFs (5) and (6). For the 3D case, on the other hand, by decomposing $\mathbf{p}_{k-1}$ as $\mathbf{p}_{k-1} = \mathbf{p}_{k-2} + \mathbf{q}_{k-1}$ with $\mathbf{p}_{k-2} \in [\mathcal{P}_{k-2}(E)]^3$ and $\mathbf{q}_{k-1} \in [\mathcal{P}_{k-1}/\mathcal{P}_{k-2}(E)]^3$, we can express the first term on the right hand side of (20) as

$$\int_{\partial P} v\, \mathbf{p}_{k-1} \cdot \mathbf{n}\mathrm{d}s = \sum_{F \in \partial P} \int_F v(\mathbf{p}_{k-2} + \mathbf{q}_{k-1}) \cdot \mathbf{n}\mathrm{d}s = \sum_{F \in \partial P} \left[\int_F v \mathbf{p}_{k-2} \cdot \mathbf{n}\mathrm{d}s + \int_F \Pi_P^\nabla v \mathbf{q}_{k-1} \cdot \mathbf{n}\mathrm{d}s \right], \tag{21}$$

which is computable using only the DOFs of v.

3.2. Virtual element approximation for linear elasticity

Let us consider a discretization denoted by Ω_h of the domain Ω into non-overlapping polygons or polyhedrons, where h stands for the average element size. We also assume that the boundary of the mesh, denoted by Γ_h, is compatible with the applied displacement and traction boundary conditions, namely, both $\Gamma_h^{\mathbf{u}}$ and $\Gamma_h^{\mathbf{t}}$ consist of unions of edges and faces of the mesh. In 3D, we also use F to denote a generic face in the mesh. For a kth order discretization, the global displacement space $\mathcal{K}_{h,k}$ is a conforming finite dimensional space defined as

$$\mathcal{K}_{h,k} \doteq \left\{ \mathbf{v}_h \in \mathcal{K} : \mathbf{v}_h|_E \in [\mathcal{V}_k(E)]^d,\ \forall E \in \Omega_h \right\}, \tag{22}$$

where $\mathcal{V}_k(E)$ is a kth order local virtual space defined in the preceding subsections. In each element E, each component of the local displacement $\mathbf{v}$ ($\mathbf{v} = [v_x, v_y]^T$ in 2D or $\mathbf{v} = [v_x, v_y, v_z]^T$ in 3D) has the set of DOFs specified in (5)–(7) and (11)–(14) respectively for 2D and 3D cases. In the following discussions, we introduce $\boldsymbol{\Pi}_k^0 : [\mathcal{V}_k(E)]^d \to [\mathcal{P}_k(E)]^d$ as the action of Π_k^0 on each component of the vector field, i.e., $\boldsymbol{\Pi}_k^0\mathbf{v} = [\Pi_k^0 v_x, \Pi_k^0 v_y]^T$ in 2D and $\boldsymbol{\Pi}_k^0\mathbf{v} = [\Pi_k^0 v_x, \Pi_k^0 v_y, \Pi_k^0 v_z]^T$ in 3D. Similarly, we also introduce the projection $\boldsymbol{\Pi}_{k-1}^0 : [L^2(E)]^{d\times d} \to [\mathcal{P}_{k-1}(E)]^{d\times d}$ for second order tensors as

$$\boldsymbol{\Pi}_{k-1}^0 \nabla\mathbf{v} = \begin{bmatrix} (\Pi_{k-1}^0 \nabla v_x)^T \\ (\Pi_{k-1}^0 \nabla v_y)^T \end{bmatrix} \text{ in 2D or } \boldsymbol{\Pi}_{k-1}^0 \nabla\mathbf{v} = \begin{bmatrix} (\Pi_{k-1}^0 \nabla v_x)^T \\ (\Pi_{k-1}^0 \nabla v_y)^T \\ (\Pi_{k-1}^0 \nabla v_z)^T \end{bmatrix} \text{ in 3D.} \tag{23}$$

When applied to the strain tensor $\boldsymbol{\epsilon}(\mathbf{v})$, the projection $\boldsymbol{\Pi}_{k-1}^0\boldsymbol{\epsilon}(\mathbf{v})$ stands for

$$\boldsymbol{\Pi}_{k-1}^0\boldsymbol{\epsilon}(\mathbf{v}) = \frac{1}{2}\left[(\boldsymbol{\Pi}_{k-1}^0\nabla\mathbf{v})^T + \boldsymbol{\Pi}_{k-1}^0\nabla\mathbf{v}\right] \tag{24}$$

Our next step is to introduce the VEM approximation to the continuous problem (3), see for instance [19,20,23]. To that end, for any element E, we need to first approximate the following local bilinear form on the left hand side of (3):

$$a^E(\mathbf{u}_h, \mathbf{v}_h) = \int_E \boldsymbol{\epsilon}(\mathbf{u}_h) : [\mathbf{C}\boldsymbol{\epsilon}(\mathbf{v}_h)]\, \mathrm{d}\mathbf{x}. \tag{25}$$

The VEM approximation $a_h^E(\mathbf{u}_h, \mathbf{v}_h)$ of $a^E(\mathbf{u}_h, \mathbf{v}_h)$ is composed of the following two terms:

$$a_h^E(\mathbf{u}_h, \mathbf{v}_h) = \int_E \boldsymbol{\Pi}_{k-1}^0\boldsymbol{\epsilon}(\mathbf{v}_h) : \mathbf{C} : \boldsymbol{\Pi}_{k-1}^0\boldsymbol{\epsilon}(\mathbf{u}_h)\mathrm{d}\mathbf{x} + \alpha^E S^E(\mathbf{u}_h - \boldsymbol{\Pi}_k^0\mathbf{u}_h, \mathbf{v}_h - \boldsymbol{\Pi}_k^0\mathbf{v}_h), \tag{26}$$

where the first integral on the right hand side is evaluated (exactly) using a numerical integration of order $2k-2$; $S^E(\cdot,\cdot)$ is a bilinear form that is computationally inexpensive to compute and satisfies the coercivity condition; and α^E is a scaling parameter that scales $S^E(\cdot,\cdot)$ to the same order of magnitude as $a^E(\cdot,\cdot)$. Typical choices of $S^E(\cdot,\cdot)$ and α^E are:

$$S^E(\mathbf{u}_h, \mathbf{v}_h) = h_E^{d-2}\sum_{i=1}^{N_E} \Xi_i(\mathbf{u}_h)\cdot\Xi_i(\mathbf{v}_h) \text{ and } \alpha^E = \mathrm{trace}\mathbf{C} = C_{ijij} \text{ (in indicial notation)} \tag{27}$$

where $h_E \doteq |E|^{1/d}$ represents the size of element E, $\Xi_i(\mathbf{v})$ stands for the ith DOF of $\mathbf{v}$ in E, and N_E stands for the total number of such local DOFs in E. In the VEM literature, the first and second terms of $a_h^E(\mathbf{u}_h, \mathbf{v}_h)$ are respectively known as the consistency and stability terms, and they are respectively responsible for the satisfactions of consistency and stability conditions, which are the two key conditions to ensure the convergence of the VEM approximation [1,19].

On the other hand, we approximate the loading term on the right hand side of (3) as

$$\langle \mathbf{f}, \mathbf{v}_h\rangle_h + \langle \mathbf{t}, \mathbf{v}_h\rangle_h = \sum_F \fint_F \mathbf{f}\cdot\boldsymbol{\Pi}_k^0\mathbf{v}_h\mathrm{d}\mathbf{x} + \sum_{e\in\Gamma_h^t}\oint_e \mathbf{t}\cdot\mathbf{v}_h\mathrm{d}\mathbf{x} \qquad \text{in 2D,} \tag{28}$$

and

$$\langle \mathbf{f}, \mathbf{v}_h\rangle_h + \langle \mathbf{t}, \mathbf{v}_h\rangle_h = \sum_P \fint_P \mathbf{f}\cdot\boldsymbol{\Pi}_k^0\mathbf{v}_h\mathrm{d}\mathbf{x} + \sum_{F\in\Gamma_h^t}\oint_F \mathbf{t}\cdot\boldsymbol{\Pi}_k^0\mathbf{v}_h\mathrm{d}\mathbf{x} \qquad \text{in 3D,} \tag{29}$$

where $\iint$ and $\oint$ denote any numerical integrations that are exact for polynomials of order $2k-2$ and $2k-1$, respectively. We note that, in the 2D case, $\mathbf{v}_h$ is by definition known as a polynomial function of order k on each edge. In addition, in the special case of $k = 1$, instead of using a integration rule that is exact of order 0 for $\iint$ (i.e. is able to integrate any constant function exactly), we use a one-point integration rule with the quadrature point and weight being the centroid and either the area or volume of each element, respectively. This integration rule is exact when the integrand is a linear function. For the higher order VEM in this paper, we utilize the commonly used triangulation scheme with the above stated order of accuracy (more sophisticated choices, for example [56,57], could be taken).

We are now ready to state the final form of the VEM approximation for linear elasticity problems, which consists of finding $\mathbf{u}_h \in \mathcal{K}_h$ such that

$$\sum_E \left[\int_E \boldsymbol{\Pi}^0_{k-1}\boldsymbol{\epsilon}(\mathbf{v}_h) : \mathbf{C} : \boldsymbol{\Pi}^0_{k-1}\boldsymbol{\epsilon}(\mathbf{u}_h)\mathrm{d}\mathbf{x} + \alpha^E S^E(\mathbf{u}_h - \boldsymbol{\Pi}^0_k\mathbf{u}_h, \mathbf{v}_h - \boldsymbol{\Pi}^0_k\mathbf{v}_h) \right] = \langle \mathbf{f}, \mathbf{v}_h \rangle_h + \langle \mathbf{t}, \mathbf{v}_h \rangle_h, \quad \forall \mathbf{v}_h \in \mathcal{K}^0_h, \tag{30}$$

where $\mathcal{K}^0_h$ is a subspace of $\mathcal{K}_h$ with functions that vanish on $\Gamma^{\mathbf{u}}_h$.

4. A recovery-based *a posteriori* error estimation for VEM

This section introduces a gradient recovery scheme that reconstructs a more accurate displacement gradient based on the VEM solutions. Making use of the reconstructed displacement gradient, this section further propose an *a posteriori* error estimator for the VEM, which estimates the error of the displacement gradient. According to the VEM philosophy, the gradient recovery scheme presented here will focus on reconstructing displacement gradients on the skeleton (i.e. union of edges) of the mesh. We also employ a H^1-type skeletal norm for the displacement gradient error estimators. For a given discretization Ω_h and a function $\mathbf{v}$, the H^1-type skeletal norm, denoted by $\varepsilon_{\mathbf{v},s}$, is defined as

$$\varepsilon_{\mathbf{v},s} = \left[\sum_{F\in\Omega_h} h_F \sum_{e\in\partial F} \int_e (\nabla\mathbf{v}\cdot\boldsymbol{\tau}_e)\cdot(\nabla\mathbf{v}\cdot\boldsymbol{\tau}_e)\mathrm{d}e \right]^{\frac{1}{2}} \quad \text{in 2D,} \tag{31}$$

and

$$\varepsilon_{\mathbf{v},s} = \left[\sum_{P\in\Omega_h} h_P \sum_{F\in\partial P} h_F \sum_{e\in\partial F} \int_e (\nabla\mathbf{v}\cdot\boldsymbol{\tau}_e)\cdot(\nabla\mathbf{v}\cdot\boldsymbol{\tau}_e)\mathrm{d}e \right]^{\frac{1}{2}} \quad \text{in 3D,} \tag{32}$$

where h_P and h_F are the diameter of polyhedron P and polygon F; and $\boldsymbol{\tau}_e$ denotes the unit tangential vector of edge e. We remark that, unlike the regular H^1 semi-norm that integrates over each element in the mesh, the H^1-type skeletal norms defined in (31) and (32) only sample function gradients along the skeleton of the mesh. These skeletal norms mimic the regular H^1 semi-norms in the following sense: they take H^1 semi-norms on the skeleton of the mesh (so that the norm for a constant function vanishes), and those differences are then scaled in order to achieve the same behavior (with respect to element contractions/expansions) as the regular H^1 semi-norm. We also remark that, similar mesh-dependent norms are considered in the Mimetic Finite Difference (MFD) literature [58,59].

4.1. A gradient recovery scheme for the VEM

We present a gradient recovery scheme for VEM of arbitrary order. This subsection is outlined as follows. We will first describe in detail the displacement gradient reconstructions for linear and quadratic virtual elements, respectively. Afterwards, a general gradient recovery framework will be outlined for VEM of arbitrary order k. Note that, in the present section, we assume to have a patch of elements associated to each vertex of the mesh; the selection of such patches will be discussed later.

Before proceeding, we first introduce the notation utilized in this gradient recovery scheme. For the ith node in the mesh expressed by $\mathbf{x}_i = [x_i, y_i]^T$ in 2D (and $\mathbf{x}_i = [x_i, y_i, z_i]^T$ in 3D), we denote by ω_i a patch of elements associated with it. We will describe the criteria for how to choose ω_i at the end of this subsection. For now, let us assume that

a patch ω is given. We denote by h_ω a characteristic size of the patch ω, which is taken as the maximum distance between the nodes in the patch, and by N_ω^E the total number of elements it contains. For the 2D cases, we use $\Xi_i^v(v)$, $\Xi_i^e(v)$ and $\Xi_i^I(v)$ to denote the ith vertex, edge and internal DOFs of a given function v in the patch ω, respectively, and N_ω^v, N_ω^e and N_ω^I to denote the total numbers of vertex, edge and internal DOFs in this patch, respectively. For the 3D case, we additionally denote by $\Xi_i^F(v)$ and N_ω^F the ith (internal) face DOFs and the total number of (internal) face DOFs in the patch ω, respectively. For instance, if a given patch consists of 2D linear virtual elements, the vertex DOFs of v are its values at the vertices of this patch, and there are no edge and internal DOFs. Alternatively, if a given patch consists of 2D quadratic virtual elements, in addition to the vertex DOFs (as in linear virtual elements), we have one edge DOF at the mid-node on every edge in this patch, and each element in the patch has one internal DOF, which is the zero order moment of v over this element.

Moreover, for a given patch ω, we denote by $\mathcal{P}_k(\omega)$ the set of polynomial functions of degree less than or equal to k with dimension $n_{\mathcal{P}_k}$. For a given order k, we have $n_{\mathcal{P}_k} = (k+1)(k+2)/2$ in 2D and $n_{\mathcal{P}_k} = (k+1)(k+2)(k+3)/6$ in 3D. If we introduce a multi-index $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$, we can define the set of basis functions of $\mathcal{P}_k(\omega)$ as

$$m_{\boldsymbol{\alpha}}^\omega(\mathbf{x}) = \left(\frac{x - x_\omega}{h_\omega}\right)^{\alpha_1}\left(\frac{y - y_\omega}{h_\omega}\right)^{\alpha_2}\left(\frac{z - z_\omega}{h_\omega}\right)^{\alpha_3}, \quad |\boldsymbol{\alpha}| \leq k, \tag{33}$$

where $|\boldsymbol{\alpha}| = \alpha_1 + \alpha_2 + \alpha_3$ and $\mathbf{x}_\omega = [x_\omega, y_\omega, z_\omega]^T$ is the centroid of ω (alternatively, $\mathbf{x}_\omega$ can also be any point in ω, e.g., the mean of all nodes in ω). We note that while the above notation is introduced for the 3D case, it also applies to the 2D case with α_3 being zero. In the following discussions, we will reshape the multi-index $\boldsymbol{\alpha}$, $|\boldsymbol{\alpha}| \leq k$ into a scalar index α, $\alpha = 1, \ldots, n_{\mathcal{P}_k}$ and use $m_\alpha^\omega(\mathbf{x})$ to denote the αth basis functions for $\mathcal{P}_k(\omega)$.

4.1.1. Displacement gradient recovery for linear elements

We address the displacement gradient recovery of linear virtual elements in both 2D and 3D cases. Since the gradient recoveries in 2D and 3D follow similar concept and procedures, we will thoroughly describe the recovery in 2D and then comment on suitable modifications in the 3D case. For both cases, we attempt to reconstruct a displacement gradient at every vertex of the mesh, and, for each vertex, the reconstruction is performed in its associated patches. Once the nodal gradients of all the vertices are obtained, a continuous displacement gradient on mesh skeleton, denoted by $G_h\mathbf{u}_h$, can then be obtained by interpolating those nodal displacement gradients on the mesh skeleton.

For a given patch ω_i (associated with $\mathbf{x}_i$) in 2D, the basic idea of the gradient recovery scheme is to seek a quadratic vector field, denoted by $\mathbf{p}^{\omega_i}(\mathbf{x}) = [p_x^{\omega_i}, p_y^{\omega_i}]^T \in [\mathcal{P}_2(\omega_i)]^2$, such that its x and y components satisfy

$$p_x^{\omega_i} = \underset{\xi \in \mathcal{P}_2(\omega_i)}{\operatorname{argmin}} \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,x})\right]^2 \text{ and } p_y^{\omega_i} = \underset{\xi \in \mathcal{P}_2(\omega_i)}{\operatorname{argmin}} \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,y})\right]^2, \tag{34}$$

respectively, where $\mathbf{u}_h = [u_{h,x}, u_{h,y}]^T$ is the VEM displacement solution.

We can express the quadratic functions $p_x^{\omega_i}$ and $p_y^{\omega_i}$ as linear combinations of the basis functions of $\mathcal{P}_2(\omega_i)$, $m_\alpha^{\omega_i}$, $\alpha = 1, \ldots, 6$, as

$$p_x^{\omega_i}(\mathbf{x}) = \sum_{\alpha=1}^{6} m_\alpha^{\omega_i}(\mathbf{x}) q_\alpha^{\omega_i,x} \text{ and } p_y^{\omega_i} = \sum_{\alpha=1}^{6} m_\alpha^{\omega_i}(\mathbf{x}) q_\alpha^{\omega_i,y}, \tag{35}$$

where $q_\alpha^{\omega_i,x}$ and $q_\alpha^{\omega_i,y}$ are the coefficients. By further considering the DOFs of $p_x^{\omega_i}$ and $p_y^{\omega_i}$, we arrive at the following matrix expressions:

$$\mathbf{P}\mathbf{q}^{\omega_i,x} = \left[\Xi_1^v(p_x^{\omega_i}) \quad \Xi_2^v(p_x^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^v}^v(p_x^{\omega_i})\right]^T \quad \text{and} \quad \mathbf{P}\mathbf{q}^{\omega_i,y} = \left[\Xi_1^v(p_y^{\omega_i}) \quad \Xi_2^v(p_y^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^v}^v(p_y^{\omega_i})\right]^T, \tag{36}$$

where $\mathbf{P} \in \mathbb{R}^{N^v_{\omega_i} \times 6}$ and $\mathbf{q}^{\omega_i,x}, \mathbf{q}^{\omega_i,y} \in \mathbb{R}^{6\times 1}$ are of the forms

$$\begin{aligned}\mathbf{P} &= \begin{bmatrix} \Xi^v_1(m^{\omega_i}_1) & \Xi^v_1(m^{\omega_i}_2) & \Xi^v_1(m^{\omega_i}_3) & \Xi^v_1(m^{\omega_i}_4) & \Xi^v_1(m^{\omega_i}_5) & \Xi^v_1(m^{\omega_i}_6) \\ \Xi^v_2(m^{\omega_i}_1) & \Xi^v_2(m^{\omega_i}_2) & \Xi^v_2(m^{\omega_i}_3) & \Xi^v_2(m^{\omega_i}_4) & \Xi^v_2(m^{\omega_i}_5) & \Xi^v_2(m^{\omega_i}_6) \\ & & \vdots & & & \\ \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_1) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_2) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_3) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_4) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_5) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_6) \end{bmatrix} \\ &= \begin{bmatrix} 1 & \Xi^v_1(\frac{x-x_{\omega_i}}{h_{\omega_i}}) & \Xi^v_1(\frac{y-y_{\omega_i}}{h_{\omega_i}}) & \Xi^v_1((\frac{x-x_{\omega_i}}{h_{\omega_i}})(\frac{y-y_{\omega_i}}{h_{\omega_i}})) & \Xi^v_1((\frac{x-x_{\omega_i}}{h_{\omega_i}})^2) & \Xi^v_1((\frac{y-y_{\omega_i}}{h_{\omega_i}})^2) \\ 1 & \Xi^v_2(\frac{x-x_{\omega_i}}{h_{\omega_i}}) & \Xi^v_2(\frac{y-y_{\omega_i}}{h_{\omega_i}}) & \Xi^v_2((\frac{x-x_{\omega_i}}{h_{\omega_i}})(\frac{y-y_{\omega_i}}{h_{\omega_i}})) & \Xi^v_2((\frac{x-x_{\omega_i}}{h_{\omega_i}})^2) & \Xi^v_2((\frac{y-y_{\omega_i}}{h_{\omega_i}})^2) \\ & & \vdots & & & \\ 1 & \Xi^v_{N^v_{\omega_i}}(\frac{x-x_{\omega_i}}{h_{\omega_i}}) & \Xi^v_{N^v_{\omega_i}}(\frac{y-y_{\omega_i}}{h_{\omega_i}}) & \Xi^v_{N^v_{\omega_i}}((\frac{x-x_{\omega_i}}{h_{\omega_i}})(\frac{y-y_{\omega_i}}{h_{\omega_i}})) & \Xi^v_{N^v_{\omega_i}}((\frac{x-x_{\omega_i}}{h_{\omega_i}})^2) & \Xi^v_{N^v_{\omega_i}}((\frac{y-y_{\omega_i}}{h_{\omega_i}})^2) \end{bmatrix}, \end{aligned} \tag{37}$$

and

$$\mathbf{q}^{\omega_i,x} = \begin{bmatrix} q^{\omega_i,x}_1 & q^{\omega_i,x}_2 & \cdots & q^{\omega_i,x}_6 \end{bmatrix}^T \text{ and } \mathbf{q}^{\omega_i,y} = \begin{bmatrix} q^{\omega_i,y}_1 & q^{\omega_i,y}_2 & \cdots & q^{\omega_i,y}_6 \end{bmatrix}^T. \tag{38}$$

With the introduction of above matrices, we can equivalently rewrite (34) as seeking $\mathbf{q}^{\omega_i,x}$ and $\mathbf{q}^{\omega_i,y} \in \mathbb{R}^{6\times 1}$ such that

$$\mathbf{q}^{\omega_i,x} = \underset{\mathbf{a}\in\mathbb{R}^{6\times 1}}{\operatorname{argmin}} \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,x}\right]^T \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,x}\right] \text{ and } \mathbf{q}^{\omega_i,y} = \underset{\mathbf{a}\in\mathbb{R}^{6\times 1}}{\operatorname{argmin}} \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,y}\right]^T \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,y}\right], \tag{39}$$

where $\mathbf{b}^{\omega_i,x}$ and $\mathbf{b}^{\omega_i,y}$ are vectors that collect the DOFs of VEM solution $\mathbf{u}_h$ in ω_i, namely, $\mathbf{b}^{\omega_i,x} = [\Xi^v_1(u_{h,x}), \ldots, \Xi^v_{N^v_{\omega_i}}(u_{h,x})]^T$ and $\mathbf{b}^{\omega_i,y} = [\Xi^v_1(u_{h,y}), \ldots, \Xi^v_{N^v_{\omega_i}}(u_{h,y})]^T$. Examining the optimality conditions of the above minimization problems, we obtain the following expressions for $\mathbf{q}^{\omega_i,x}$ and $\mathbf{q}^{\omega_i,y}$:

$$\mathbf{q}^{\omega_i,x} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{b}^{\omega_i,x} \text{ and } \mathbf{q}^{\omega_i,y} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{b}^{\omega_i,y}. \tag{40}$$

Once the vectors of coefficients $\mathbf{q}^{\omega_i,x}$ and $\mathbf{q}^{\omega_i,y}$ are computed, we obtain the polynomial functions $p^{\omega_i}_x(\mathbf{x})$ and $p^{\omega_i}_y(\mathbf{x})$, and the value of the reconstructed displacement gradient $G_h\mathbf{u}_h(\mathbf{x}_i)$ at $\mathbf{x}_i$ is taken as

$$G_h\mathbf{u}_h(\mathbf{x}_i) = \nabla\mathbf{p}^{\omega_i}(\mathbf{x}_i) = \begin{bmatrix} \frac{\partial p^{\omega_i}_x}{\partial x}(\mathbf{x}_i) & \frac{\partial p^{\omega_i}_x}{\partial y}(\mathbf{x}_i) \\ \frac{\partial p^{\omega_i}_y}{\partial x}(\mathbf{x}_i) & \frac{\partial p^{\omega_i}_y}{\partial y}(\mathbf{x}_i) \end{bmatrix}. \tag{41}$$

In 3D, the recovery scheme follows a procedure similar to the one outlined above except for two modifications. First, in order to account for the expanding dimension of $\mathcal{P}_2(\omega_i)$ in 3D, the matrix $\mathbf{P}$ defined in (37) is modified as

$$\mathbf{P} = \begin{bmatrix} \Xi^v_1(m^{\omega_i}_1) & \Xi^v_1(m^{\omega_i}_2) & \Xi^v_1(m^{\omega_i}_3) & \cdots & \Xi^v_1(m^{\omega_i}_{10}) \\ \Xi^v_2(m^{\omega_i}_1) & \Xi^v_2(m^{\omega_i}_2) & \Xi^v_2(m^{\omega_i}_3) & \cdots & \Xi^v_2(m^{\omega_i}_{10}) \\ & & \ddots & & \\ \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_1) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_2) & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_3) & \cdots & \Xi^v_{N^v_{\omega_i}}(m^{\omega_i}_{10}) \end{bmatrix}. \tag{42}$$

Moreover, the recovery scheme is expanded to include the z component of $\mathbf{u}_h = [u_{h,x}, u_{h,y}, u_{h,z}]^T$ in 3D. With the modified matrix $\mathbf{P}$, the displacement gradient recovery scheme in 3D consists of seeking $\mathbf{q}^{\omega_i,x}$, $\mathbf{q}^{\omega_i,y}$ and $\mathbf{q}^{\omega_i,z} \in \mathbb{R}^{10\times 1}$ such that

$$\begin{aligned} \mathbf{q}^{\omega_i,x} &= \underset{\mathbf{a}\in\mathbb{R}^{10\times 1}}{\operatorname{argmin}} \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,x}\right]^T \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,x}\right] \\ \mathbf{q}^{\omega_i,y} &= \underset{\mathbf{a}\in\mathbb{R}^{10\times 1}}{\operatorname{argmin}} \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,y}\right]^T \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,y}\right] \\ \mathbf{q}^{\omega_i,z} &= \underset{\mathbf{a}\in\mathbb{R}^{10\times 1}}{\operatorname{argmin}} \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,z}\right]^T \left[\mathbf{Pa} - \mathbf{b}^{\omega_i,z}\right], \end{aligned} \tag{43}$$

where, similarly to $\mathbf{b}^{\omega_i,x}$ and $\mathbf{b}^{\omega_i,y}$, $\mathbf{b}^{\omega_i,z}$ is a vector collecting the DOFs of the z component of $\mathbf{u}_h$ as $\mathbf{b}^{\omega_i,z} = [\Xi^v_1(u_{h,z}), \ldots, \Xi^v_{N^v_{\omega_i}}(u_{h,z})]^T$. Once the coefficient vectors $\mathbf{q}^{\omega_i,x}$, $\mathbf{q}^{\omega_i,y}$ and $\mathbf{q}^{\omega_i,z}$ are obtained, the values of the

reconstructed displacement gradient field $G_h\mathbf{u}_h$ at node $\mathbf{x}_i$ are naturally given by

$$G_h\mathbf{u}_h(\mathbf{x}_i) = \nabla\mathbf{p}^{\omega_i}(\mathbf{x}_i) = \begin{bmatrix} \dfrac{\partial p_x^{\omega_i}}{\partial x}(\mathbf{x}_i) & \dfrac{\partial p_x^{\omega_i}}{\partial y}(\mathbf{x}_i) & \dfrac{\partial p_x^{\omega_i}}{\partial z}(\mathbf{x}_i) \\ \dfrac{\partial p_y^{\omega_i}}{\partial x}(\mathbf{x}_i) & \dfrac{\partial p_y^{\omega_i}}{\partial y}(\mathbf{x}_i) & \dfrac{\partial p_y^{\omega_i}}{\partial z}(\mathbf{x}_i) \\ \dfrac{\partial p_z^{\omega_i}}{\partial x}(\mathbf{x}_i) & \dfrac{\partial p_z^{\omega_i}}{\partial y}(\mathbf{x}_i) & \dfrac{\partial p_z^{\omega_i}}{\partial z}(\mathbf{x}_i) \end{bmatrix}. \tag{44}$$

4.1.2. Displacement gradient recovery for 2D quadratic elements

We present a gradient reconstruction scheme for quadratic elements in 2D, in which we attempt to reconstruct the displacement gradient at every vertex of the mesh, as well as the mid-side node of every edge. In particular, the displacement gradient for every vertex is reconstructed within the associated patch, and the one for each mid-side node is obtained by averaging the reconstructed gradients within both patches associated with the two vertices of its edge.

Let us first look at the gradient recovery on the vertices. Similar to the linear case, for a given vertex $\mathbf{x}_i$ and its associated patch ω_i, we aim to seek a cubic vector field, denoted by $\mathbf{p}^{\omega_i} = [p_x^{\omega_i}, p_y^{\omega_i}]^T \in [\mathcal{P}_3(\omega_i)]^2$, such that its components satisfy

$$\begin{aligned} p_x^{\omega_i} &= \underset{\xi\in\mathcal{P}_3(\omega_i)}{\operatorname{argmin}} \left\{ \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,x})\right]^2 + \sum_{j=1}^{N_{\omega_i}^e} \left[\Xi_j^e(\xi) - \Xi_j^e(u_{h,x})\right]^2 + \sum_{j=1}^{N_{\omega_i}^I} \left[\Xi_j^I(\xi) - \Xi_j^I(u_{h,x})\right]^2 \right\} \\ p_y^{\omega_i} &= \underset{\xi\in\mathcal{P}_3(\omega_i)}{\operatorname{argmin}} \left\{ \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,y})\right]^2 + \sum_{j=1}^{N_{\omega_i}^e} \left[\Xi_j^e(\xi) - \Xi_j^e(u_{h,y})\right]^2 + \sum_{j=1}^{N_{\omega_i}^I} \left[\Xi_j^I(\xi) - \Xi_j^I(u_{h,y})\right]^2 \right\}, \end{aligned} \tag{45}$$

where we recall that the operator $\Xi_j^I(\cdot) = 1/|E_j| \int_{E_j}(\cdot)\mathrm{d}\mathbf{x}$ represents the internal DOF of the jth element in the patch.

Expanding $p_x^{\omega_i}(\mathbf{x})$ and $p_y^{\omega_i}(\mathbf{x})$ in terms of basis functions $m_\alpha^{\omega_i}(\mathbf{x})$ of $\mathcal{P}_3(\omega_i)$, $\alpha = 1, \ldots, 10$, as

$$p_x^{\omega_i}(\mathbf{x}) = \sum_{\alpha=1}^{10} m_\alpha^{\omega_i}(\mathbf{x}) q_\alpha^{\omega_i,x} \quad \text{and} \quad p_y^{\omega_i}(\mathbf{x}) = \sum_{\alpha=1}^{10} m_\alpha^{\omega_i}(\mathbf{x}) q_\alpha^{\omega_i,y}, \tag{46}$$

we obtain the following relations in matrix forms as

$$\mathbf{P}\mathbf{q}^{\omega_i,x} = \left[\Xi_1^v(p_x^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^v}^v(p_x^{\omega_i}) \quad \Xi_1^e(p_x^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^e}^e(p_x^{\omega_i}) \quad \Xi_1^I(p_x^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^I}^I(p_x^{\omega_i}) \right]^T, \tag{47}$$

and

$$\mathbf{P}\mathbf{q}^{\omega_i,y} = \left[\Xi_1^v(p_y^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^v}^v(p_y^{\omega_i}) \quad \Xi_1^e(p_y^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^e}^e(p_y^{\omega_i}) \quad \Xi_1^I(p_y^{\omega_i}) \quad \cdots \quad \Xi_{N_{\omega_i}^I}^I(p_y^{\omega_i}) \right]^T, \tag{48}$$

where matrices $\mathbf{P} \in \mathbb{R}^{(N_{\omega_i}^v + N_{\omega_i}^e + N_{\omega_i}^I)\times 10}$ and $\mathbf{q}^{\omega_i,x}, \mathbf{q}^{\omega_i,y} \in \mathbb{R}^{10\times 1}$ are given by

$$\mathbf{P} = \begin{bmatrix} \Xi_1^v(m_1^{\omega_i}) & \Xi_1^v(m_2^{\omega_i}) & \Xi_1^v(m_3^{\omega_i}) & \cdots & \Xi_1^v(m_{10}^{\omega_i}) \\ & & \ddots & & \\ \Xi_{N_{\omega_i}^v}^v(m_1^{\omega_i}) & \Xi_{N_{\omega_i}^v}^v(m_2^{\omega_i}) & \Xi_{N_{\omega_i}^v}^v(m_3^{\omega_i}) & \cdots & \Xi_{N_{\omega_i}^v}^v(m_{10}^{\omega_i}) \\ \Xi_1^e(m_1^{\omega_i}) & \Xi_1^e(m_2^{\omega_i}) & \Xi_1^e(m_3^{\omega_i}) & \cdots & \Xi_1^e(m_{10}^{\omega_i}) \\ & & \ddots & & \\ \Xi_{N_{\omega_i}^e}^e(m_1^{\omega_i}) & \Xi_{N_{\omega_i}^e}^e(m_2^{\omega_i}) & \Xi_{N_{\omega_i}^e}^e(m_3^{\omega_i}) & \cdots & \Xi_{N_{\omega_i}^e}^e(m_{10}^{\omega_i}) \\ \Xi_1^I(m_1^{\omega_i}) & \Xi_1^I(m_2^{\omega_i}) & \Xi_1^I(m_3^{\omega_i}) & \cdots & \Xi_1^I(m_{10}^{\omega_i}) \\ & & \ddots & & \\ \Xi_{N_{\omega_i}^I}^I(m_1^{\omega_i}) & \Xi_{N_{\omega_i}^I}^I(m_2^{\omega_i}) & \Xi_{N_{\omega_i}^I}^I(m_3^{\omega_i}) & \cdots & \Xi_{N_{\omega_i}^I}^I(m_{10}^{\omega_i}) \end{bmatrix}, \tag{49}$$

and

$$\mathbf{q}^{\omega_i,x} = \begin{bmatrix} q_1^{\omega_i,x} & q_2^{\omega_i,x} & \cdots & q_{10}^{\omega_i,x} \end{bmatrix}^T \text{ and } \mathbf{q}^{\omega_i,y} = \begin{bmatrix} q_1^{\omega_i,y} & q_2^{\omega_i,y} & \cdots & q_{10}^{\omega_i,y} \end{bmatrix}^T. \tag{50}$$

We can then rewrite Eq. (45) as finding $\mathbf{q}^{\omega_i,x}$ and $\mathbf{q}^{\omega_i,y} \in \mathbb{R}^{10\times 1}$ such that

$$\mathbf{q}^{\omega_i,x} = \underset{\mathbf{a}\in\mathbb{R}^{10\times 1}}{\operatorname{argmin}} \left[\mathbf{P}\mathbf{a} - \mathbf{b}^{\omega_i,x}\right]^T \left[\mathbf{P}\mathbf{a} - \mathbf{b}^{\omega_i,x}\right] \quad \text{and} \quad \mathbf{q}^{\omega_i,y} = \underset{\mathbf{a}\in\mathbb{R}^{10\times 1}}{\operatorname{argmin}} \left[\mathbf{P}\mathbf{a} - \mathbf{b}^{\omega_i,y}\right]^T \left[\mathbf{P}\mathbf{a} - \mathbf{b}^{\omega_i,y}\right], \tag{51}$$

where $\mathbf{b}^{\omega_i,x}$ and $\mathbf{b}^{\omega_i,y}$ are vectors consisting of both edge and internal DOFs of $u_{h,x}$ and $u_{h,y}$ in the patch, i.e.

$$\mathbf{b}^{\omega_i,x} = \left[\Xi_1^v(u_{h,x}) \quad \cdots \quad \Xi_{N_{\omega_i}^v}^v(u_{h,x}) \quad \Xi_1^e(u_{h,x}) \quad \cdots \quad \Xi_{N_{\omega_i}^e}^e(u_{h,x}) \quad \Xi_1^I(u_{h,x}) \quad \cdots \quad \Xi_{N_{\omega_i}^I}^I(u_{h,x}) \right]^T. \tag{52}$$

and

$$\mathbf{b}^{\omega_i,y} = \left[\Xi_1^v(u_{h,y}) \quad \cdots \quad \Xi_{N_{\omega_i}^v}^v(u_{h,y}) \quad \Xi_1^e(u_{h,y}) \quad \cdots \quad \Xi_{N_{\omega_i}^e}^e(u_{h,y}) \quad \Xi_1^I(u_{h,y}) \quad \cdots \quad \Xi_{N_{\omega_i}^I}^I(u_{h,y}) \right]^T. \tag{53}$$

The above equations yield

$$\mathbf{q}^{\omega_i,x} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{b}^{\omega_i,x} \text{ and } \mathbf{q}^{\omega_i,y} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{b}^{\omega_i,y}. \tag{54}$$

Having obtained coefficient vectors $\mathbf{q}^{\omega_i,x}$ and $\mathbf{q}^{\omega_i,y}$, we can reconstruct the displacement gradient $G_h\mathbf{u}_h$ at vertex $\mathbf{x}_i$ using Eq. (41).

Let us then look at the reconstruction on the mid-edge nodes. If we denote by $\widehat{\mathbf{x}}_{ij}$ the middle node of the edge connecting vertices $\mathbf{x}_i$ and $\mathbf{x}_j$, i.e. $\widehat{\mathbf{x}}_{ij} = 1/2(\mathbf{x}_i + \mathbf{x}_j)$, we define the recovered values of the displacement gradient at $\widehat{\mathbf{x}}_{ij}$ as the average value of the ones obtained from both patches associated to the two end vertices, namely,

$$G_h\mathbf{u}_h(\widehat{\mathbf{x}}_{ij}) = \frac{1}{2}\left[\nabla\mathbf{p}^{\omega_i}(\widehat{\mathbf{x}}_{ij}) + \nabla\mathbf{p}^{\omega_j}(\widehat{\mathbf{x}}_{ij})\right] = \frac{1}{2}\begin{bmatrix} \dfrac{\partial p_x^{\omega_i}}{\partial x}(\widehat{\mathbf{x}}_{ij}) + \dfrac{\partial p_x^{\omega_j}}{\partial x}(\widehat{\mathbf{x}}_{ij}) & \dfrac{\partial p_x^{\omega_i}}{\partial y}(\widehat{\mathbf{x}}_{ij}) + \dfrac{\partial p_x^{\omega_j}}{\partial y}(\widehat{\mathbf{x}}_{ij}) \\ \dfrac{\partial p_y^{\omega_i}}{\partial x}(\widehat{\mathbf{x}}_{ij}) + \dfrac{\partial p_y^{\omega_j}}{\partial x}(\widehat{\mathbf{x}}_{ij}) & \dfrac{\partial p_y^{\omega_i}}{\partial y}(\widehat{\mathbf{x}}_{ij}) + \dfrac{\partial p_y^{\omega_j}}{\partial y}(\widehat{\mathbf{x}}_{ij}) \end{bmatrix}, \tag{55}$$

where we recall that $\mathbf{p}^{\omega_i}$ and $\mathbf{p}^{\omega_j}$ are the reconstructed cubic vectorial functions for patch ω_i and ω_j, respectively. Note that the above construction is different than taking the average of the reconstructed gradient at the two vertices (a choice that would lead to a loss of accuracy).

Remark 4.1. By comparing with the definition (34) for linear elements, the one (45) for quadratic elements also considers internal DOFs of $\mathbf{u}_h$ for each patch. This requires evaluating not only the edge DOFs of the basis functions $m_\alpha^{\omega_i}(\mathbf{x})$ but also their internal DOFs when forming the matrix $\mathbf{P}$ defined in (49). While the edge DOFs of $m_\alpha^{\omega_i}(\mathbf{x})$ are their values at the selected points of edges, the evaluation of their internal DOFs (the moments of $m_\alpha^{\omega_i}$) requires the use of at least a 3rd-order numerical integration rule, which imposes addition computational expense on the simulation (recall that only a 2nd order integration is needed to exactly compute the stiffness matrix). As we will demonstrate in the numerical examples, it is possible to neglect the consideration of internal DOFs in definition (45) without sacrificing too much accuracy in reconstructions. Doing this, however, leads to a more efficient gradient recovery scheme, as it avoids the integration of basis functions $m_\alpha^{\omega_i}(\mathbf{x})$ to obtain their internal DOFs as we form the matrix $\mathbf{P}$.

4.1.3. A displacement gradient recovery for virtual elements of arbitrary order

Although not numerically evaluated in this paper, this subsection outlines an extension of the displacement gradient recovery scheme for virtual elements of arbitrary order k in both 2D and 3D. Similarly to before, this subsection first describes the reconstructions at the vertices, and then states the reconstructions at the edge nodes of the mesh.

For the vertex $\mathbf{x}_i$ with its associated patch ω_i, the first step of the gradient recovery scheme for kth order virtual elements is to reconstruct a polynomial function $\mathbf{p}^{\omega_i} \in [\mathcal{P}_{k+1}(\omega_i)]^d$ via a least square fitting of all the DOFs of $\mathbf{u}_h$ in ω_i. More specifically, in the 2D case, the polynomial function $\mathbf{p}^{\omega_i} = [p_x^{\omega_i}, p_y^{\omega_i}]^T$ is obtained following a similar

definition as for quadratic virtual elements. In the component form, we define

$$\begin{aligned} p_x^{\omega_i} &= \underset{\xi\in\mathcal{P}_{k+1}(\omega_i)}{\operatorname{argmin}} \left\{ \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,x})\right]^2 + \sum_{j=1}^{N_{\omega_i}^e} \left[\Xi_j^e(\xi) - \Xi_j^e(u_{h,x})\right]^2 + \sum_{j=1}^{N_{\omega_i}^I} \left[\Xi_j^I(\xi) - \Xi_j^I(u_{h,x})\right]^2 \right\} \\ p_y^{\omega_i} &= \underset{\xi\in\mathcal{P}_{k+1}(\omega_i)}{\operatorname{argmin}} \left\{ \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,y})\right]^2 + \sum_{j=1}^{N_{\omega_i}^e} \left[\Xi_j^e(\xi) - \Xi_j^e(u_{h,y})\right]^2 + \sum_{j=1}^{N_{\omega_i}^I} \left[\Xi_j^I(\xi) - \Xi_j^I(u_{h,y})\right]^2 \right\}. \end{aligned} \tag{56}$$

For the 3D case, because of the presence of face DOFs, the above component-wise definition of $\mathbf{p}^{\omega_i} = [p_x^{\omega_i}, p_y^{\omega_i}, p_z^{\omega_i}]^T$ is expanded to also include those face DOFs. In particular, we seek

$$\begin{aligned} p_x^{\omega_i} = \underset{\xi\in\mathcal{P}_{k+1}(\omega_i)}{\operatorname{argmin}} \Bigg\{ & \sum_{j=1}^{N_{\omega_i}^v} \left[\Xi_j^v(\xi) - \Xi_j^v(u_{h,x})\right]^2 + \sum_{j=1}^{N_{\omega_i}^e} \left[\Xi_j^e(\xi) - \Xi_j^e(u_{h,x})\right]^2 + \sum_{j=1}^{N_{\omega_i}^I} \left[\Xi_j^I(\xi) - \Xi_j^I(u_{h,x})\right]^2 \\ & + \sum_{j=1}^{N_{\omega_i}^F} \left[\Xi_j^F(\xi) - \Xi_j^F(u_{h,x})\right]^2 \Bigg\} \end{aligned} \tag{57}$$

(and the analogous one for the other two components $p_y^{\omega_i}$, $p_z^{\omega_i}$) where we recall that the face DOF operator $\Xi_i^F(\cdot)$, $i = 1, \ldots, N_{\omega_i}^F$, takes the moments of its argument up to order $k-2$ on the faces in patch ω_i. In practice, the polynomial function $\mathbf{p}^{\omega_i}$ is calculated following identical procedures described in the preceding subsections for linear and quadratic virtual elements, namely, using the matrix $\mathbf{P}$, which contains the edge, internal (and face, in 3D) DOFs of the basis functions $m_\alpha^{\omega_i}$ of $\mathcal{P}_{k+1}(\omega_i)$, and vectors $\mathbf{b}^{\omega_i,x}$, $\mathbf{b}^{\omega_i,y}$ and $\mathbf{b}^{\omega_i,z}$, which consist of those DOFs of the x, y and z components of $\mathbf{u}_h$, respectively. For the sake of brevity, those procedures are not repeated here.

Once we obtain the polynomial function $\mathbf{p}^{\omega_i}(\mathbf{x})$ for each patch ω_i, the recovered displacement gradient at vertex $\mathbf{x}^i$ is computed by evaluating $\mathbf{p}^{\omega_i}$ at this vertex, which is identical to the linear and quadratic cases, i.e.

$$G_h\mathbf{u}_h(\mathbf{x}_i) = (\nabla\mathbf{p}^{\omega_i})(\mathbf{x}_i). \tag{58}$$

For higher-order virtual elements, i.e. $k \geq 2$, the gradient recovery scheme also seeks to reconstruct $G_h\mathbf{u}_h$ at $k-1$ internal points on each edge, which coincides with the internal integration points of the Gauss–Lobatto rule of order $k+1$ on that edge. Assuming that $\widehat{\mathbf{x}}_{ij}^\ell$, $\ell = 1, \ldots, k-1$, is the ℓth internal point of the edge that connects vertices $\mathbf{x}_i$ and $\mathbf{x}_j$, we generalize a suitable definition of the values of $G_h\mathbf{u}_h$ at this point from (55) as follows [47]

$$G_h\mathbf{u}_h(\mathbf{x}_i) = \alpha_{ij}^\ell(\nabla\mathbf{p}^{\omega_i})(\widehat{\mathbf{x}}_{ij}^\ell) + (1-\alpha_{ij}^\ell)(\nabla\mathbf{p}^{\omega_j})(\widehat{\mathbf{x}}_{ij}^\ell), \tag{59}$$

where α_{ij}^ℓ is the ratio of the distance between $\widehat{\mathbf{x}}_{ij}^\ell$ and $\mathbf{x}_i$, and the length of the edge, namely, $\alpha_{ij}^\ell = \|\widehat{\mathbf{x}}_{ij}^\ell - \mathbf{x}_i\|/\|\mathbf{x}_i - \mathbf{x}_j\|$.

Remark 4.2. The displacement gradient recovery scheme is tailored for the *a posteriori* error estimator using the H^1 type semi-norms (31)–(32) defined on the skeleton of the mesh. To that end, we only reconstruct the displacement gradient on edges of the mesh, that leads to a computationally cheaper procedure. Nevertheless, by following the same lines, our recovery scheme could be easily extended in order to yield also the internal degree of freedom values of the reconstructed gradient.

Remark 4.3. According to Remark 4.1, it is possible to neglect the internal and facial DOFs of $\mathbf{u}_h$ in the recovery process. As suggested by the numerical examples for quadratic virtual elements, this may lead to almost identically reconstructed displacement gradients, and render the recovery scheme more efficient. For virtual elements of order $k \geq 3$, however, more investigation is needed.

4.1.4. Reconstruction of the displacement gradient $G_h\mathbf{u}_h$

In this subsection, we present the procedures to reconstruct a continuous displacement gradient on the mesh skeleton using the recovered values of $G_h\mathbf{u}_h$ at $k+1$ equally-spaced points on every edge in the mesh skeleton.

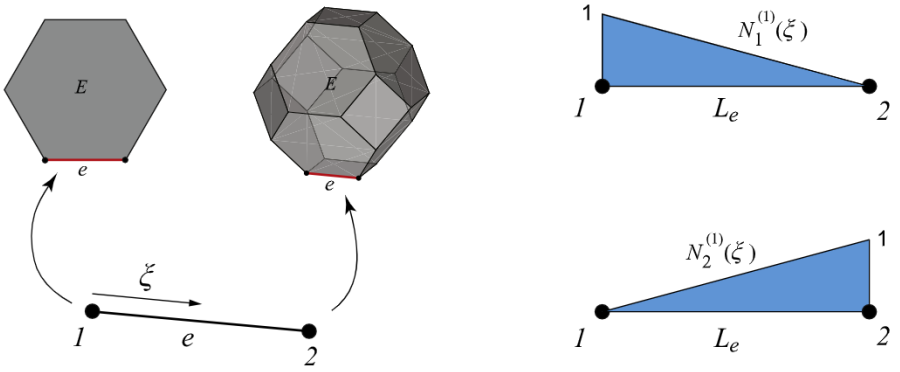

Fig. 1. An illustration of the displacement gradient reconstruction for linear virtual elements in 2D and 3D. The displacement gradient on each edge e is interpolated from the reconstructed nodal values using 1D linear shape functions.

In the remaining, for a given generic edge e in the skeleton, we denote $\mathbf{G}_e^\ell$ by the recovered values of the displacement gradient on the ℓth point on this edge.

The basic idea of the reconstruction is to interpolate the recovered values of $G_h\mathbf{u}_h$ at the $k+1$ edge nodes using C^0 continuous shape functions on each edge. More specifically, for a generic edge e, we interpolate $G_h\mathbf{u}_h$ as:

$$G_h\mathbf{u}_h(\xi) = \sum_{\ell=1}^{k+1} N_\ell^{(k)}(\xi)\mathbf{G}_e^\ell, \tag{60}$$

where $N_\ell^{(k)}(\xi)$ is the 1D Lagrangian shape function (of order k) associated with the ℓ edge node on e, with ξ varying from 0 to L_e, in which L_e is the length of edge e.

In Fig. 1, we provide an illustration of the displacement gradient reconstruction for linear virtual elements, together with the adopted 1D shape functions on a generic edge e.

As a side note, in addition to reconstructing a displacement gradient only on the mesh skeleton, one can also choose to reconstruct a displacement gradient over the entire Ω using suitable basis functions. For linear virtual elements in 2D, for instance, a continuous displacement gradient over Ω can be reconstructed by interpolating the values at vertices obtained from the recovery procedure using nodal vector-valued basis functions of degree 1. Alternatively, one can use both the recovered displacement[1] and displacement gradient on the vertices and perform a C^1 reconstruction of the displacement field using C^1 basis functions [28,60]. The reconstructed displacement gradient is then taken as the gradient of the reconstructed displacement. The advantage of this alternative approach over the former one is that, in addition to giving a continuous displacement gradient, the alternative approach also provides a displacement with higher continuity (i.e. C^1) and potentially higher accuracy over the original displacement solutions (which is only C^0 continuous). Aside from the above comments, however, we stick to recovering a C^0 displacement gradient on the mesh skeleton in the remainder of the paper.

4.2. Suitable choice of patches

In the gradient recovery scheme, the selection of patches is a crucial component. In the sequel, we describe the criteria of how to select the patches. For a vertex $\mathbf{x}_i$, we choose its associated patch in the same fashion as in the finite element literature. As illustrated in Fig. 2(a) and Fig. 3(a) for 2D and 3D cases respectively, the patch ω_i is defined to be the union of all the elements that connect to $\mathbf{x}_i$. In 3D, this patch definition also applies to the boundary facial vertices, as shown in Fig. 3(b).

Similar to the finite element literature, special attention needs to be paid when selecting boundary patches as they are likely to contain insufficient DOFs to uniquely determine the polynomial functions in the recovery process (in this case, the matrix $\mathbf{P}^T\mathbf{P}$ associated with that patch is singular). For instance, if chosen based on the criterion stated above,

[1] Instead of evaluating the gradient of reconstructed polynomial function at vertices as in (41), a recovered displacement is obtained by evaluating the reconstructed polynomial at the vertices.

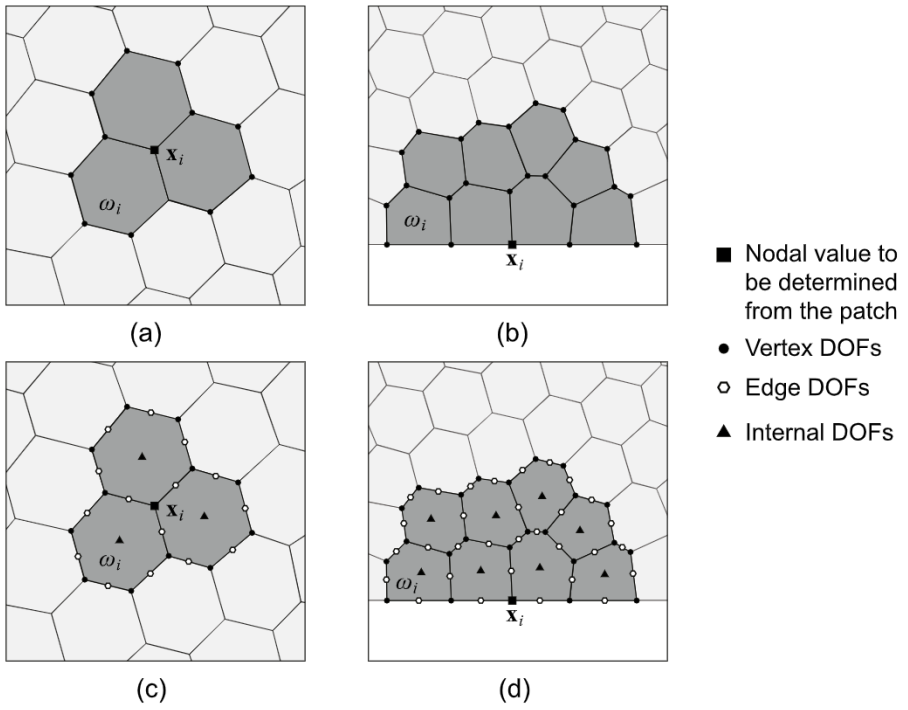

Fig. 2. Examples of (a) internal and (b) boundary patches in 2D for linear virtual elements. Examples of (c) internal and (d) boundary patches in 2D for quadratic virtual elements.

the patches associated with boundary vertices in 2D (cf. Fig. 2(b)) and boundary edge vertices in 3D (cf. Fig. 3(c)) will typically contain only two elements. Those patches are most likely to contain insufficient DOFs, especially for higher order elements. In addition, for general polygonal and polyhedral discretization, similar scenarios may occur even for patches that belong to internal vertices. Therefore, to ensure the robustness of the recovery scheme, we propose to monitor N_ω^E, which is the number of elements the patch contains. If N_ω^E is less than three, we propose to enlarge the vertex patch by including another layer of elements. An enlarged patch is defined to be the union of patches associated with all the vertices that the original patch contains. This criterion will automatically include the boundary patches (i.e. the patches associated with boundary vertices in 2D (cf. Fig. 2(b)) and boundary edge vertices in 3D (cf. Fig. 3(c))) that were discussed previously.[2]

4.3. Error evaluation and a posteriori error estimation for the VEM

We discuss approaches to evaluate the errors of both the original and recovered displacement using the H^1-type skeletal norm defined in (31) and (32). On top of the discussions on error evaluations, we further present *a posteriori* error estimation based on the recovered displacement gradient $G_h\mathbf{u}_h$, which also makes use of the H^1-type skeletal norm defined in (31) and (32).

4.3.1. Error evaluation

According to (31) and (32), the original displacement error from VEM simulations using the H^1-type skeletal norm is given by

$$\varepsilon_{\mathbf{u},s} = \left[\sum_{F\in\Omega_h} h_F \sum_{e\in\partial F}\int_e (\nabla\mathbf{u}_h\cdot\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\cdot(\nabla\mathbf{u}_h\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\mathrm{d}e\right]^{\frac{1}{2}} \quad \text{in 2D,} \tag{61}$$

[2] An even safer procedure to select whenever to enlarge the vertex patches would be a direct check on the non-singularity of the matrix $\mathbf{P}^T\mathbf{P}$. Nevertheless, the procedure proposed here is cheaper and turns out to be robust in all the numerical experiments.

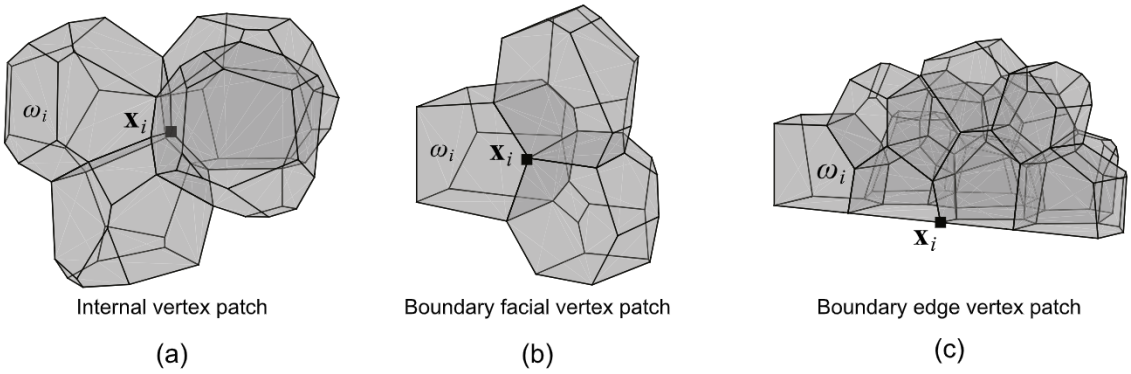

Fig. 3. Examples of patches in 3D a polyhedral mesh: (a) an internal patch, (b) a patch on a face of the boundary, and (c) a patch on a edge of the boundary.

and

$$\boxed{\varepsilon_{\mathbf{u},s} = \left[\sum_{P\in\Omega_h} h_P \sum_{F\in\partial P} h_F \sum_{e\in\partial F} \int_e (\nabla\mathbf{u}_h\cdot\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\cdot(\nabla\mathbf{u}_h\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\mathrm{d}e\right]^{\frac{1}{2}} \quad \text{in 3D,}} \tag{62}$$

where $\mathbf{u}$ is the exact displacement. For the VEM of order k, the displacement solution $\mathbf{u}_h$ possesses kth order variation on a generic edge e, which can be interpolated from the edge DOFs of $\mathbf{u}_h$ on the $k+1$ equally spaced points of this edge (including the two end points). For this generic edge e, let us denote $\mathbf{U}_e^\ell$ as ℓth DOF of $\mathbf{u}_h$ on e. We can interpolate $\mathbf{u}_h$ and $\nabla\mathbf{u}_h\cdot\boldsymbol{\tau}_e$ along this edge as

$$\mathbf{u}_h(\xi) = \sum_{\ell=1}^{k+1} N_\ell^{(k)}(\xi)\mathbf{U}_e^\ell \text{ and } (\nabla\mathbf{u}_h\cdot\boldsymbol{\tau}_e)(\xi) = \sum_{\ell=1}^{k+1} \frac{\mathrm{d}\,N_\ell^{(k)}(\xi)}{\mathrm{d}\,\xi}\mathbf{U}_e^\ell, \tag{63}$$

where $N_\ell^{(k)}(\xi)$ is the 1D shape function (of order k) associated with $\mathbf{U}_e^\ell$ on e, with ξ varying from 0 to L_e, and L_e is the length of edge e. In practice, based on the above parametrization, the original displacement error is evaluated using a Gauss–Lobatto rule of a suitable order on each edge.

For the error evaluation of the recovered fields, the reconstruction $G_h\mathbf{u}_h$ in (60) is used, replacing $\nabla\mathbf{u}_h$ in the original displacement error (61) – (62). In practice, similar to the original displacement error discussed above, the skeletal error of the recovered displacement gradient is evaluated as

$$\boxed{\varepsilon_{\widetilde{\mathbf{u}},s} = \left[\sum_{F\in\Omega_h} h_F \sum_{e\in\partial F} \int_e (G_h\mathbf{u}_h\cdot\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\cdot(G_h\mathbf{u}_h\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\mathrm{d}e\right]^{\frac{1}{2}} \quad \text{in 2D,}} \tag{64}$$

and

$$\boxed{\varepsilon_{\widetilde{\mathbf{u}},s} = \left[\sum_{P\in\Omega_h} h_P \sum_{F\in\partial P} h_F \sum_{e\in\partial F} \int_e (G_h\mathbf{u}_h\cdot\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\cdot(G_h\mathbf{u}_h\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\mathrm{d}e\right]^{\frac{1}{2}} \quad \text{in 3D,}} \tag{65}$$

where we interpolate $G_h\mathbf{u}_h$ along edge e as

$$(G_h\mathbf{u}_h)(\xi) = \sum_{\ell=1}^{k+1} N_\ell^{(k)}(\xi)\mathbf{G}_e^\ell \tag{66}$$

with $N_\ell^{(k)}(\xi)$ being the 1D Lagrangian shape function (of order k) associated with the ℓ edge node on e, ξ varying from 0 to L_e; and $\mathbf{G}_e^\ell$ being the recovered values of $G_h\mathbf{u}_h$ on the ℓth point on edge e. In practice, the above integrals can be evaluated using a Gauss–Lobatto rule of suitable order on each edge.

4.3.2. A recovery-based a posteriori error estimator

In the proceeding subsection, the error evaluations make use of the exact displacement gradients $\nabla\mathbf{u}$, which are typically unknown in practice. Thus, this subsection proposes a recovery-based *a posteriori* error estimator, which is based on the reconstructed displacement gradient, to estimate the original skeletal errors (61)–(62) without knowing $\nabla\mathbf{u}$. The main idea behind this estimator is to replace the exact displacement gradient with the reconstructed one in the original error evaluations (61)–(62). Provided that the reconstructed displacement gradient is more accurate than the original ones, as expected, doing so should yield a reasonable estimation of the original error [41,36].

The error estimator, denoted by $\widetilde{\varepsilon}_{\mathbf{u},s}$, makes use of the reconstructed displacement gradient $G_h\mathbf{u}_h$ on the mesh skeleton. Over a generic element E, we can express the estimated error using as

$$\boxed{\widetilde{\varepsilon}_{\mathbf{u},s}|_E = \left\{ h_F \sum_{e\in\partial F} \int_e (G_h\mathbf{u}_h\cdot\boldsymbol{\tau}_e - \nabla\mathbf{u}_h\cdot\boldsymbol{\tau}_e)\cdot(G_h\mathbf{u}_h\boldsymbol{\tau}_e - \nabla\mathbf{u}_h\cdot\boldsymbol{\tau}_e)\mathrm{d}e \right\}^{\frac{1}{2}} \quad \text{in 2D,}} \tag{67}$$

and

$$\boxed{\widetilde{\varepsilon}_{\mathbf{u},s}|_E = \left\{ h_P \sum_{F\in\partial P} h_F \sum_{e\in\partial F} \int_e (G_h\mathbf{u}_h\cdot\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\cdot(G_h\mathbf{u}_h\boldsymbol{\tau}_e - \nabla\mathbf{u}\cdot\boldsymbol{\tau}_e)\mathrm{d}e \right\}^{\frac{1}{2}} \quad \text{in 3D,}} \tag{68}$$

where the $G_h\mathbf{u}_u$ and $\nabla\mathbf{u}\cdot\boldsymbol{\tau}_e$ are interpolated using (63) and (66) on every edge e of element E, respectively. In practice, the above integrals can be evaluated exactly using Gauss–Lobatto rules of at least order $2k$ on edge e. We note that the above local error estimations are useful in the adaptive refinement and coarsening analysis to pinpoint which regions to refine and which regions to coarsen [44]. Once the estimated error is known for each element, the estimated global error is computed by summing the local errors from every elements as:

$$\widetilde{\varepsilon}_{\mathbf{u},s} = \left[\sum_{E\in\Omega_h} \left(\widetilde{\varepsilon}_{\mathbf{u},s}|_E\right)^2 \right]^{\frac{1}{2}}. \tag{69}$$

In the subsequent numerical studies, we will verify the skeletal errors by comparing them with the standard error measures. To that end, we also consider the more standard L^2 errors of both the original and recovered displacement gradients given as

$$\varepsilon_{\mathbf{u}} = \left[\sum_{E\in\Omega_h} \int_E (\nabla\mathbf{u} - \nabla\mathbf{u}_h)\cdot(\nabla\mathbf{u} - \nabla\mathbf{u}_h)\mathrm{d}\mathbf{x} \right]^{\frac{1}{2}}, \tag{70}$$

and

$$\varepsilon_{\widetilde{\mathbf{u}}} = \left[\sum_{E\in\Omega_h} \int_E (\nabla\mathbf{u} - G_h\mathbf{u}_h)\cdot(\nabla\mathbf{u} - G_h\mathbf{u}_h)\mathrm{d}\mathbf{x} \right]^{\frac{1}{2}}, \tag{71}$$

respectively. In the L^2 errors above, both the original displacement $\mathbf{u}_h$ and recovered displacement gradients $G_h\mathbf{u}_h$ are interpolated using the barycentric coordinates — we use the mean value shape functions in 2D [61] and Wachspress shape functions in 3D [62] for the linear VEM, and the serendipity shape functions constructed from mean value shape functions based on [63] for the quadratic VEM. The Wachspress shape functions are only applicable to convex elements, whereas the mean value shape functions, as well as the serendipity shape functions constructed from them, are applicable to both convex and non-convex elements [3]. Regarding numerical integration, all the integrals are evaluated using a fifth order quadrature rule.

To conclude this section, we summarize in Table 1 the notation of all the error measures we have presented. The subsequent numerical examples will follow those notations.

5. Some theoretical estimates

In the present section we develop some theoretical estimates for the post-processed error $\varepsilon_{\widetilde{\mathbf{u}},s}$ in (64). Note that the present derivations should not be intended as a proof, as they hinge on some reasonable but not demonstrated assumption. Instead, the purpose of this section is to give some theoretical backbone to our construction.

Table 1
Summary of the notations of the errors used in the numerical studies.

	Skeletal norm	L^2 norm
Err. of the original disp. gradient	$\varepsilon_{\mathbf{u},s}$ (by Eqs. (61)–(62))	$\varepsilon_{\mathbf{u}}$ (by Eq. (70))
Err. of the recovered disp. gradient	$\varepsilon_{\widetilde{\mathbf{u}},s}$ (by Eqs. (64)–(65))	$\varepsilon_{\widetilde{\mathbf{u}}}$ (by Eqs. (71))
Estimated err. of the original disp. gradient	$\widetilde{\varepsilon}_{\mathbf{u},s}$ (by Eqs. (67)–(68))	n/a

In the following we will use the standard notation for L^p spaces and H^k Sobolev spaces. In order to keep the derivations simple, we will restrict our attention to the case of $k = 1$ in 2D. The generalization to 3D follows almost identically while higher order cases can be tackled following the same lines. In the following we assume that the (family of) meshes are shape regular, in the sense [1] that every element is star shaped with respect to a ball of uniformly comparable radius and every edge has length that is uniformly comparable to the diameter of the parent elements. Under such assumptions one can derive that (see for instance [64]) the original error (61) satisfies

$$\varepsilon_{\mathbf{u},s} \leq Ch\|\mathbf{u}\|_{H^2(\Omega)}, \tag{72}$$

where here and in the sequel C will denote a generic constant, possibly different at each occurrence, independent of the mesh size h.

For any sufficiently regular tensor valued function $\mathbf{v}$, let us now define the operator $|||\cdot|||$ as

$$|||\mathbf{v}|||^2 = \sum_{F\in\Omega_h} |||\mathbf{v}|||_F^2 \quad \text{where} \quad |||\mathbf{v}|||_F^2 = h_F \sum_{e\in\partial F} \int_e \|\mathbf{v}\cdot\boldsymbol{\tau}_e\|^2 \mathrm{d}e.$$

Note that, by definition, the error in the skeleton norm is given by:

$$\varepsilon_{\widetilde{\mathbf{u}},s} = |||\nabla\mathbf{u} - G_h\mathbf{u}_h|||. \tag{73}$$

We recall that, given any element $F \in \Omega_h$, each of its vertices $\mathbf{x}_i$ (with i running in some integer set $\mathcal{I}$) is associated to a patch of elements ω_i (see Section 4.2). Moreover, we here denote by $\omega_F = \cup_{i\in\mathcal{I}}\omega_i$ the union of such patches. Note that the operator G_h depends only on the pointwise values of $\mathbf{u}_h$ at the vertices of Ω_h, and thus it can be obviously extended to any function space for which pointwise values make sense, e.g. $[H^2(\Omega)]^2$. Given any $\mathbf{w} \in [H^2(\Omega)]^2$ and any vertex $\mathbf{x}_i$ of the mesh, we denote by $\mathbf{p}_{\mathbf{w}}^{\omega_i} \in [\mathcal{P}_2(\omega_i)]^2$ the polynomial built by our least square procedure with input values given by evaluating $\mathbf{w}$ at the vertices of ω_i. The VEM function $G_h\mathbf{w}$ at $\mathbf{x}_i$ is then computed, as usual, as $G_h\mathbf{w}(\mathbf{x}_i) = \nabla\mathbf{p}_{\mathbf{w}}^{\omega_i}(\mathbf{x}_i)$. We moreover observe that our post-processing construction is $\mathcal{P}_2$ preserving, in the sense that for all $F \in \Omega_h$ it holds

$$\nabla\mathbf{p}|_F = G_h\mathbf{p}|_F \qquad \forall\mathbf{p} \in [\mathcal{P}_2(\omega_F)]^2. \tag{74}$$

We start by showing a stability result for the operator G_h. In order to do so, we need the following reasonable assumption (see Remark 5.1):

(A1) There exists a constant $C^\star$ such that, for all $\mathbf{w} \in [H^2(\Omega)]^2$, $F \in \Omega_h$ and all vertices $\mathbf{x}_i$ of F, it holds

$$\|\mathbf{p}_{\mathbf{w}}^{\omega_i}\|_{L^\infty(F)} \leq C^\star \max_{\mathbf{x}_j \text{ vertex in } \omega^i} \|\mathbf{w}(\mathbf{x}_j)\|.$$

Let us now take any $\mathbf{w} \in [H^2(\Omega)]^2$. Given the definition of $|||\cdot|||_F$, first by a Holder inequality and then recalling that $G_h\mathbf{w}$ is linear on edges, we have

$$|||G_h\mathbf{w}|||_F^2 \leq C\, h_F^2 \|G_h\mathbf{w}\|_{L^\infty(\partial F)}^2 \leq C\, h_F^2 \max_{\mathbf{x}_i \text{ vertex of } F} \|G_h\mathbf{w}(\mathbf{x}_i)\|^2 = C\, h_F^2 \max_{\mathbf{x}_i \text{ vertex of } F} \|\nabla\mathbf{p}_{\mathbf{w}}^{\omega_i}(\mathbf{x}_i)\|^2,$$

where the last identity follows from the definition of G_h. From the above bound, first by a trivial inequality, then with a standard inverse estimate for polynomials (on star shaped domains), we get

$$|||G_h\mathbf{w}|||_F^2 \leq C\, h_F^2 \max_{\mathbf{x}_i \text{ vertex of } F} \|\nabla\mathbf{p}_{\mathbf{w}}^{\omega_i}\|_{L^\infty(F)}^2 \leq C \max_{\mathbf{x}_i \text{ vertex of } F} \|\mathbf{p}_{\mathbf{w}}^{\omega_i}\|_{L^\infty(F)}^2.$$

We now conclude our stability estimate for G_h by applying assumption (**A1**)

$$|||G_h\mathbf{w}|||_F^2 \leq C \max_{\mathbf{x}_i \text{ vertex of } F} \left(\max_{\mathbf{x}_j \text{ vertex in } \omega^i} \|\mathbf{w}(\mathbf{x}_j)\|^2 \right) = C \max_{\mathbf{x}_j \text{ vertex in } \omega_F} \|\mathbf{w}(\mathbf{x}_j)\|^2 \tag{75}$$

Let now a generic $F \in \Omega_h$. Using the $\mathcal{P}_2$ preserving property (74), adding/subtracting terms and the triangle inequality easily yield

$$|||\nabla \mathbf{u} - G_h \mathbf{u}_h|||_F \leq |||\nabla(\mathbf{u} - \mathbf{p})|||_F + |||G_h(\mathbf{p} - \mathbf{u})|||_F + |||G_h(\mathbf{u} - \mathbf{u}_h)|||_F =: T_1 + T_2 + T_3, \tag{76}$$

for any polynomial $\mathbf{p} \in [\mathcal{P}_2(\omega_F)]^2$, with obvious meaning of the three terms T_1, T_2, T_3. The first term is bounded by a standard scaled trace inequality and approximation results for polynomials on star-shaped domains, yielding

$$T_1 \leq C\|\nabla(\mathbf{u} - \mathbf{p})\|_{L^2(F)} + h_F|\nabla(\mathbf{u} - \mathbf{p})|_{H^1(F)} \leq Ch_F^2|\mathbf{u}|_{H^3(F)}.$$

The second term is bounded using (75) and again standard approximation results for polynomials

$$T_2 \leq \max_{\mathbf{x}_j \text{ vertex in } \omega_F} \|\mathbf{p}(\mathbf{x}_j) - \mathbf{u}(\mathbf{x}_j)\| \leq Ch_{\omega_F}^2|\mathbf{u}|_{H^3(\omega_F)},$$

while again (75) yields

$$T_3 \leq C \max_{\mathbf{x}_j \text{ vertex in } \omega_F} \|\mathbf{u}(\mathbf{x}_j) - \mathbf{u}_h(\mathbf{x}_j)\|^2.$$

Now, by definition of $|||\cdot|||$ and bounding all diameters with h, it is easy to check that Eq. (76) and the bounds above for T_1, T_2, T_3 finally give

$$\varepsilon_{\widetilde{\mathbf{u}},s} = |||\nabla \mathbf{u} - G_h \mathbf{u}_h||| \leq Ch^2|\mathbf{u}|_{H^3(\Omega)} + C\Big(\sum_{F\in\Omega_h} \max_{\mathbf{x}_j \text{ vertex in } \omega_F} \|\mathbf{u}(\mathbf{x}_j) - \mathbf{u}_h(\mathbf{x}_j)\|^2\Big)^{1/2}. \tag{77}$$

The above bound is to be compared with (72). It shows that the accuracy of the post-processed gradient $G_h\mathbf{u}_h$ depends directly on the values of $\mathbf{u} - \mathbf{u}_h$ at the *vertices* of the mesh. Therefore if such values are particularly accurate (as is typically the case), the post-processed gradient will strongly benefit. In the extremely positive case that $\mathbf{u}_h = \mathbf{u}_I$ (that is the interpolant of $\mathbf{u}$) the second term vanishes and one obtains an $O(h^2)$ convergence rate. The same happens in the presence of some super-convergence phenomena of $\mathbf{u}_h$ to $\mathbf{u}$ at vertices. In many situations $G_h\mathbf{u}_h$ will be more accurate than $\nabla\mathbf{u}_h$ but still of the same order in h.

One can get a better understanding of the last term by assuming that all elements in the mesh have comparable diameter (which is just to make the argument simple). Indeed, in such case the number of elements in the mesh behaves as h^{-2} and thus we get immediately

$$\sum_{F\in\Omega_h} \max_{\mathbf{x}_j \text{ vertex in } \omega_F} \|\mathbf{u}(\mathbf{x}_j) - \mathbf{u}_h(\mathbf{x}_j)\|^2 \leq Ch^{-2} \max_{\mathbf{x}_j \text{ vertex in } \Omega_h} \|\mathbf{u}(\mathbf{x}_j) - \mathbf{u}_h(\mathbf{x}_j)\|^2.$$

Therefore the last term in (77) (considering also the square root) behaves as

$$h^{-1} \max_{\mathbf{x}_j \text{ vertex in } \Omega_h} \|\mathbf{u}(\mathbf{x}_j) - \mathbf{u}_h(\mathbf{x}_j)\|,$$

which (by standard approximation estimates for $(\mathbf{u} - \mathbf{u}_h)$ in the L^∞ norm) is guaranteed to be $O(h)$ for regular enough $\mathbf{u}$, but could be far better in the presence of a super convergence property at vertices.

Remark 5.1. Assumption (A1) represents the (uniform in the mesh family) stability of the polynomial least-square procedure, and depends on the position of the vertices in each patch ω_i. Roughly, it is sufficient that the set of vertices in each patch ω_i uniquely determines a least square $\mathcal{P}_2$ interpolant (that is, any $\mathcal{P}_2$ polynomial that takes value zero at all such points must vanish) and "stays far from" degenerate cases in which such condition is lost. Proving such assumption would be possible focusing on given element patterns.

6. Numerical examples

In this section, both 2D and 3D numerical studies are presented to show the accuracy and effectiveness of the recovered displacement gradient and the (both local and global) recovery-based *a posteriori* error estimators. Various types of displacement solutions are considered here, including ones with both small and large (but finite) gradients, and additional one with singularity. Throughout, we set the Young's modulus and Poisson's ratio of the solids to be $E = 10$ and $\nu = 0.35$, respectively, and consider plane strain condition for the 2D examples. Consistent units are adopted throughout the manuscript.

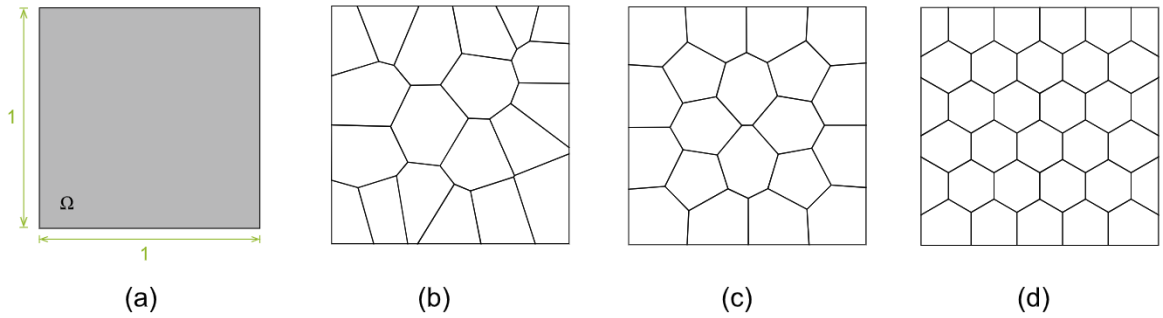

Fig. 4. (a) Problem setup and dimensions of the unit square. (b) An example of the random Voronoi mesh. (c) An example of the CVT mesh. (d) An example of the structured hexagonal mesh.

6.1. Numerical investigation in a 2D unit square

In the 2D numerical studies, we test the proposed scheme on a unit square domain $\Omega = (0, 1)^2$, as depicted in Fig. 4(a). We assume two exact displacement fields. The first displacement field $\mathbf{u} = [u_x, u_y]^T$ is given by

$$u_x = \sin(x)e^y \text{ and } u_y = y^2 - 2x, \tag{78}$$

which is referred to as the "smooth solution" due to its smoothness and regularity. The second displacement field $\mathbf{u}$ takes the form of [50]

$$u_x = 16x(1-x)y(1-y)\text{atan}(\frac{25x - 100y + 50}{4}) \text{ and } u_y = 2x^2 + 4y. \tag{79}$$

We refer to this displacement field as the "steep solution" as its x component has a sharp (but finite) gradient along the line $x - 4y + 2 = 0$. In the numerical studies, the assumed exact displacements $\mathbf{u}$ are applied on the entire boundary of the unit square $\partial\Omega$. We consider various types of polygonal discretizations: random Voronoi, centroid Voronoi Tessellation (CVT), and structured hexagonal meshes, as shown in Figs. 4(b)–(d), respectively. We test the both linear and quadratic VEMs.

6.1.1. Linear VEM ($k = 1$)

Let us first verify the accuracy of the skeletal error by comparing it with the standard error measures. To that end, we plot in Figs. 5(a)–(f) the convergence of the skeletal errors of both the original and reconstructed displacement gradients (solid lines) for the smooth and steep solutions. In the plots, those skeletal errors are compared with the standard L^2 error of the original and reconstructed displacement gradients (dashed lines). It is immediate from the comparisons that, for both the smooth and steep solutions, the proposed skeletal norm is able to capture the correct convergence behavior of the L^2 error of both the original and reconstructed displacement gradients. Thus, for the remainder of this study, only the skeletal errors will be utilized.

We then study the accuracy of the recovered displacement gradient $G_h\mathbf{u}_h$. To do so, we depict in Figs. 6(a)–(f) the comparisons between the (global) errors of $G_h\mathbf{u}_h$ and those of the original displacement gradient $\nabla\mathbf{u}_h$ for the smooth and steep solutions. In case of the smooth solution, we observe that $G_h\mathbf{u}_h$ is far more accurate than $\nabla\mathbf{u}_h$ for all families of meshes considered. Only for the structured hexagonal meshes, $G_h\mathbf{u}_h$ exhibits superconvergent behavior — the convergence rate being roughly 1.5 as compared to 1 for those of the original displacement gradient. On the other hand, in case of the steep solution, $G_h\mathbf{u}_h$ exhibits superconvergent behaviors for all families of meshes — its errors start to converge at a higher convergence rate (i.e. rate of 2) with respect to the original displacement gradient (i.e. rate of 1) as the meshes are refined. Finally, we also perform a SPR-type recovery scheme (see the detailed description in Appendix) and plot the errors of the reconstructed $G_h\mathbf{u}_h$ in Figs. 6(a)–(f). By comparing the accuracy of $G_h\mathbf{u}_h$ obtained from the proposed recovery scheme and the SPR-type recovery scheme, we conclude that the proposed recovery scheme seems to yield more accurate reconstructions than the SPR-type recovery schemes.

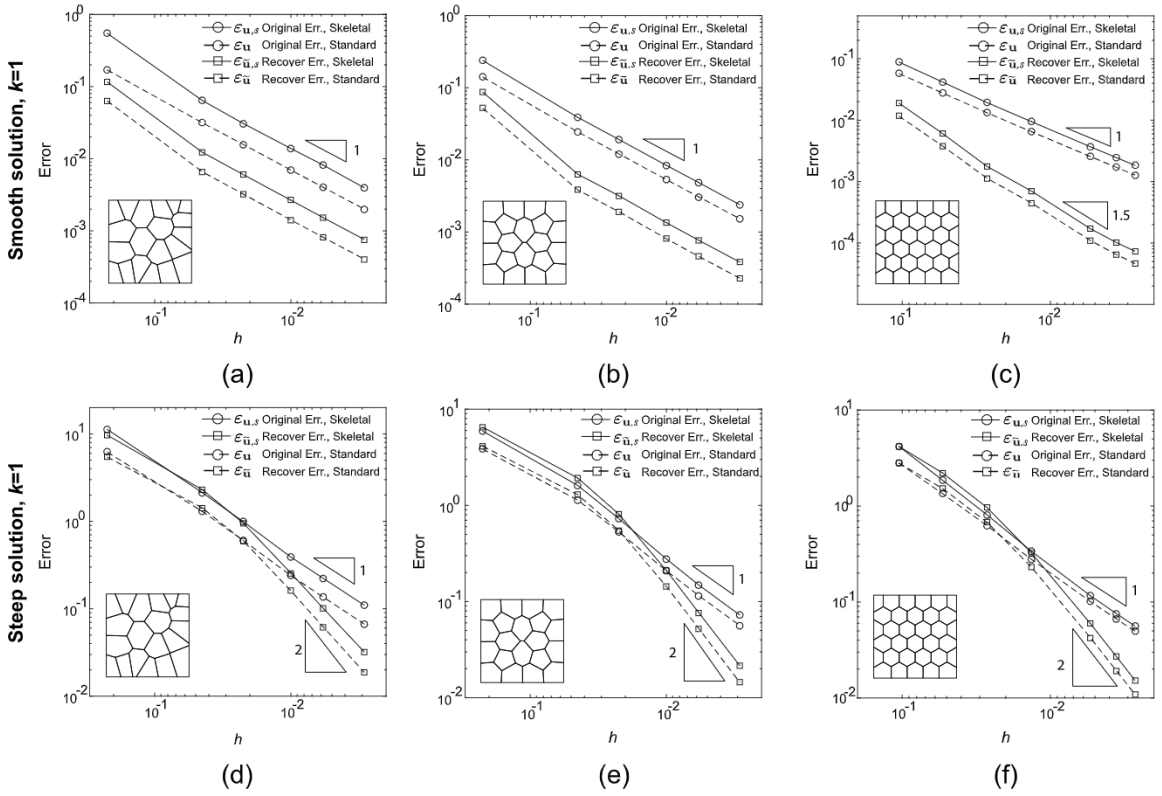

Fig. 5. Comparing the convergence behaviors between the skeletal errors ($\varepsilon_{\mathbf{u},s}$ and $\varepsilon_{\tilde{\mathbf{u}},s}$) and the standard ones ($\varepsilon_{\mathbf{u}}$ and $\varepsilon_{\tilde{\mathbf{u}}}$) for 2D linear VEM. Smooth solution: (a) random Voronoi meshes, (b) CVT meshes, and (c) structured hexagonal meshes. Steep solution: (d) random Voronoi meshes, (e) CVT meshes, and (f) structured hexagonal meshes.

Lastly, the accuracy of both global and local error estimators is studied. In Figs. 7(a)–(f), we plot the convergence of the global error estimator computed from $G_h\mathbf{u}_h$ for the smooth and steep solutions on all families of meshes. For the local error estimators, we show in Figs. 8–10 fringe plots of the element-wise error estimators and exact errors for random Voronoi, CVT and structured hexagonal meshes, respectively. The comparisons demonstrate that the error estimator introduced in this work is effective for various polygonal discretizations, and is capable of accurately estimating both the global and local displacement errors, especially for the smooth solution case. For the steep solution, although the error estimators are less accurate on coarse meshes, they quickly converge to the exact errors as the meshes are refined.

6.1.2. Quadratic VEM ($k = 2$)

For quadratic VEM, a similar set of numerical tests are conducted for both smooth and steep solutions on the three families of meshes shown in Fig. 4(b)–(d). To begin with, the use of skeletal errors is verified by comparing them with the standard L^2 errors of the displacement gradients. Figs. 11(a)–(f) plot the skeletal errors of both original and recovered displacement gradients and compare their convergence to the corresponding L^2 errors. From the comparisons, we observe that, for both steep and steep solutions, the skeletal errors of both the original and recovered displacement gradients agree well (in terms of rate and trend of convergence) with the standard L^2 errors. This observation provides us confidence in using the skeletal error in the following tests.

Furthermore, we plot in Figs. 12(a)–(f) errors of the recovered displacement gradients obtained with and without the internal DOFs (see Remark 4.1) as functions of average mesh size h. The errors of the original displacement gradient are also included in the plot as references. We can see from the figures that neglecting the internal DOFs in the recovery scheme almost does not affect the accuracy of recovered displacement gradient. Thus, for the sake of computational efficiency, we conclude that it is favorable to neglect the internal DOFs in the recovery scheme.

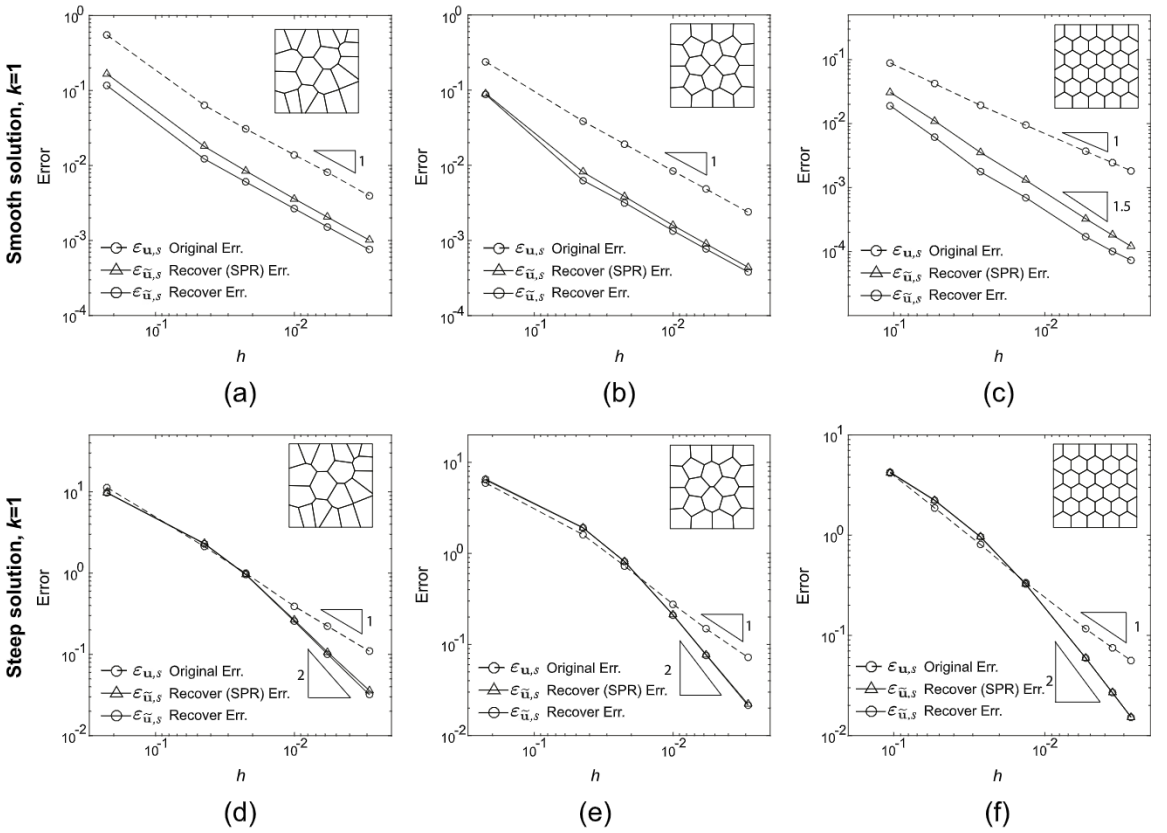

Fig. 6. Comparing the accuracy between the original errors ($\varepsilon_{\mathbf{u},s}$) and the recovered ones ($\varepsilon_{\widetilde{\mathbf{u}},s}$) for 2D linear VEM. The recovered errors include ones obtained from the proposed scheme and a SPR-type scheme introduced in Appendix. Smooth solution: (a) random Voronoi meshes, (b) CVT meshes, and (c) structured hexagonal meshes. Steep solution: (d) random Voronoi meshes, (e) CVT meshes, and (f) structured hexagonal meshes.

Additionally, similar to the observations in linear VEM, the recovered displacement gradients are more accurate than the original ones and, in case of the steep solution, they exhibit super-convergent behaviors — the rate of convergence is roughly 3.

Finally, we investigate the accuracy of both global and local error estimators by comparing them with the exact errors in cases both of the smooth and steep solutions. For the global error estimator, we depict in Figs. 13(a)–(f) the convergence of both the estimated and exact errors as the meshes are refined and, for the local error estimator, Figs. 14–16 show the fringe plots of the element-wise distributions of the estimated and exact errors for random Voronoi, CVT and hexagonal meshes, respectively. It is apparent from both comparisons that the global and local error estimators can effectively and accurately predict the exact errors and, as the meshes are refined, the predictions become more and more accurate. Moreover, aligned with previous conclusions, the recovered displacement gradients obtained without considering the internal DOFs yield almost identical estimated errors to the ones computed including the internal DOFs, which confirms again that we can neglect the internal DOFs in the recovery procedure (see Remark 4.1).

6.2. Numerical investigation in a 2D "L"-shaped domain

In this numerical example, we consider an L-shaped domain, whose dimensions are given in Fig. 17(a). The domain is clamped on its top edge and subjected to a constant shear $\tau = 1$ pointing downward on its right edge. In this subsection, we will consider three families of polygonal meshes, i.e. random Voronoi, CVT and concave octagonal meshes, as shown in Figs. 17(b)–(d), respectively. We will then assess the performance of the proposed gradient recover scheme and the global and local error estimators under uniform mesh refinement. In Section 7, we will restrict

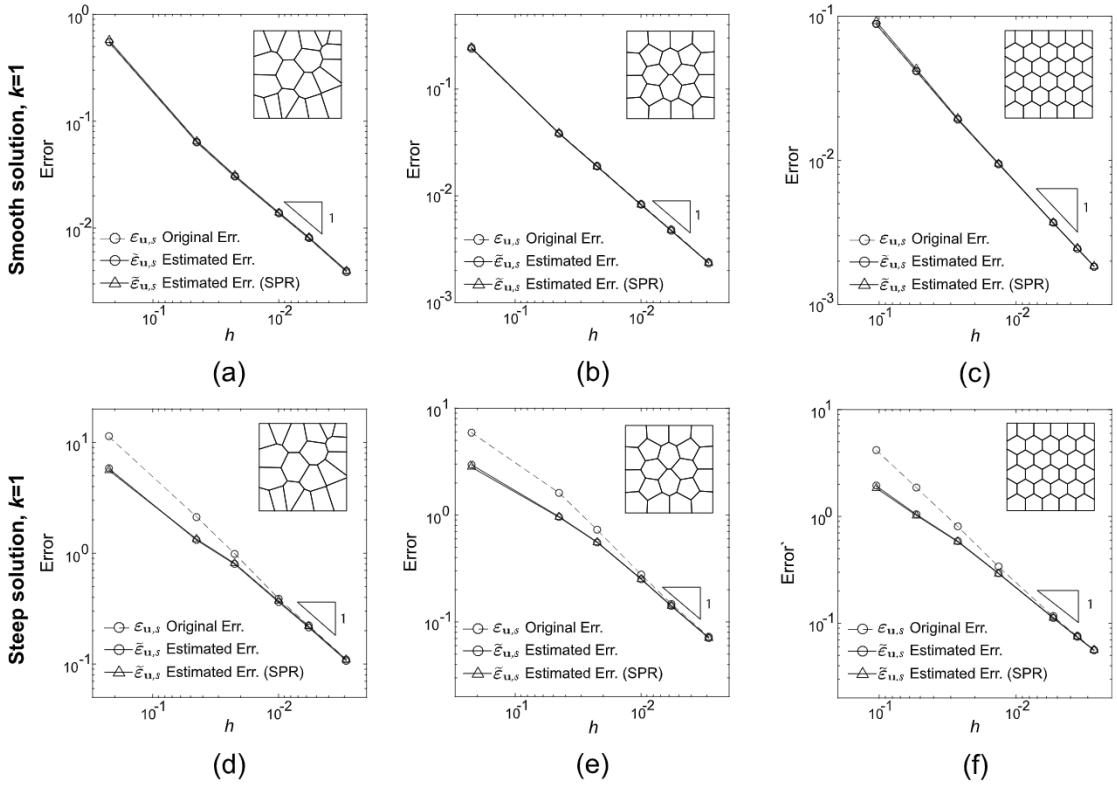

Fig. 7. Comparing the accuracy between the original errors ($\varepsilon_{\mathbf{u},s}$) and the estimated ones ($\widetilde{\varepsilon}_{\mathbf{u},s}$) for 2D linear VEM. The estimated errors include ones obtained from the proposed scheme and a SPR-type scheme introduced in Appendix. Smooth solution: (a) random Voronoi meshes, (b) CVT meshes, and (c) structured hexagonal meshes. Steep solution: (d) random Voronoi meshes, (e) CVT meshes, and (f) structured hexagonal meshes.

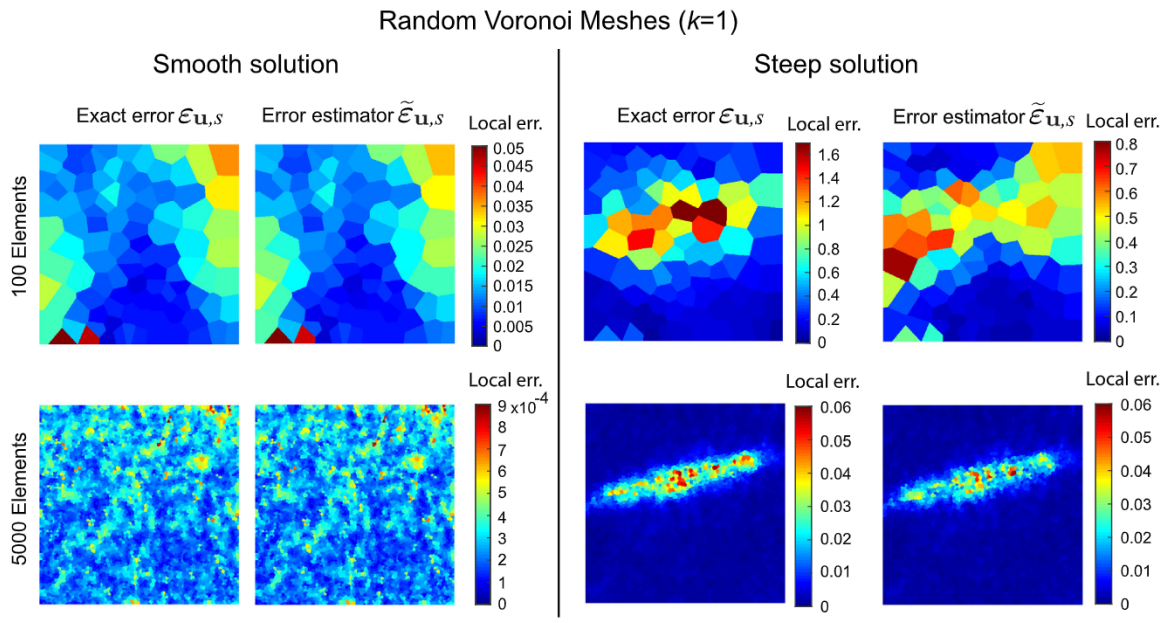

Fig. 8. Fringe plots of the element-level exact errors and estimated ones for 2D linear VEM on random Voronoi meshes.

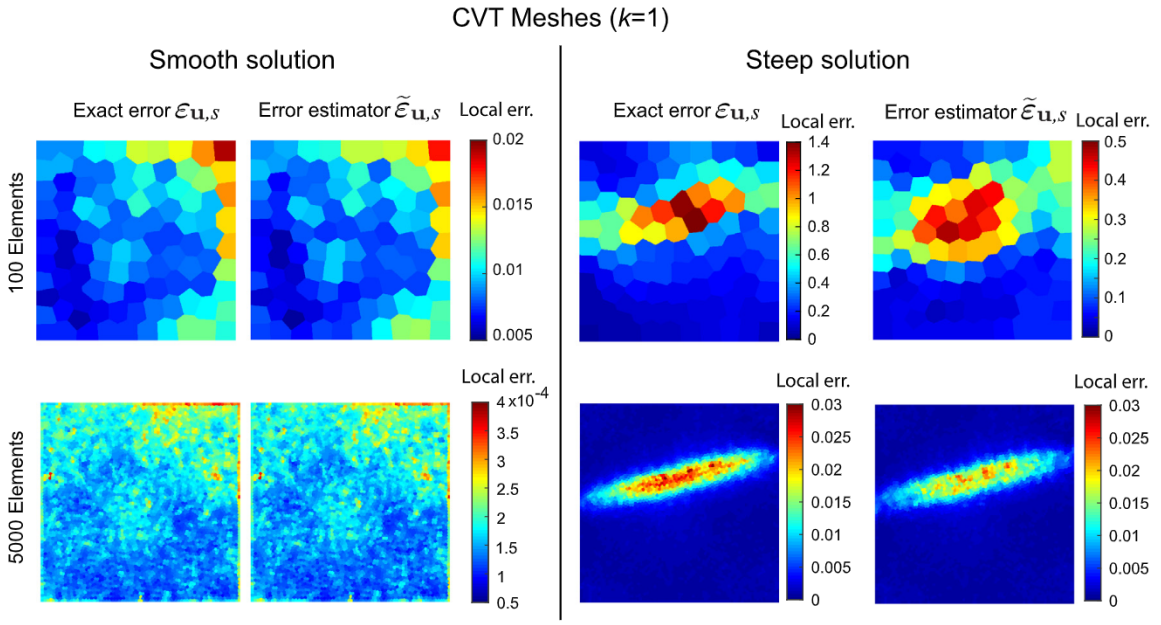

Fig. 9. Fringes plots of the element-level exact errors and estimated ones for 2D linear VEM on CVT meshes.

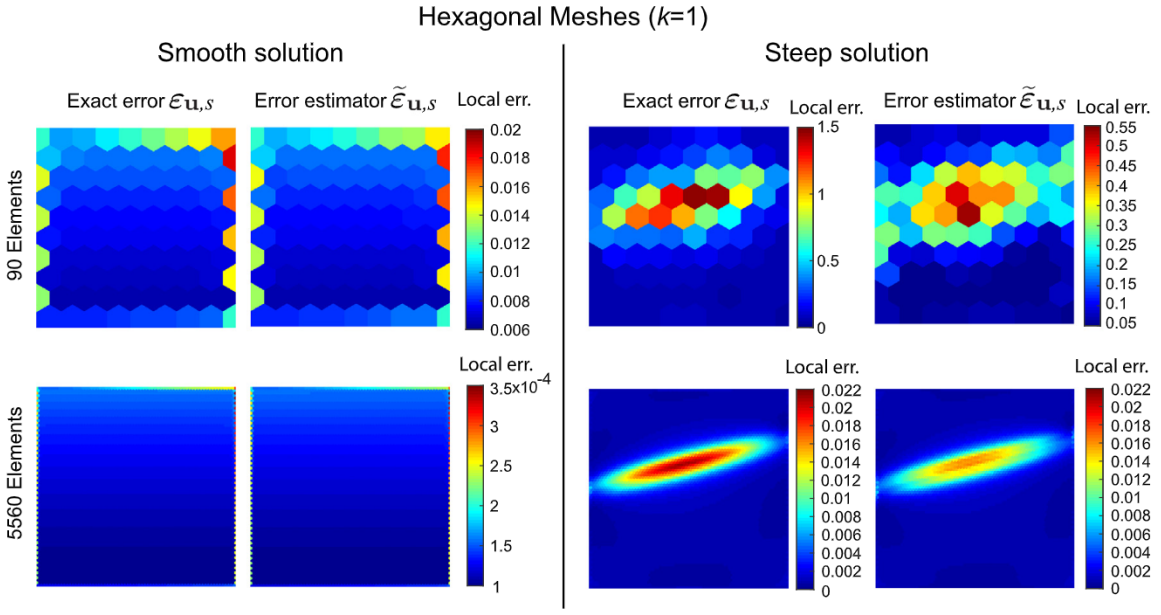

Fig. 10. Fringes plots of the element-level exact errors and estimated ones for 2D linear VEM on hexagonal meshes.

our attention to a CVT mesh and evaluate the effectiveness of the proposed error estimator under various adaptive mesh refinement strategies. For this boundary value problem, singularity exists in the displacement and stress solutions at the reentrant corner of the domain. Thus, one goal of this example is to showcase that the proposed error estimator is able to capture this singularity in the solution and can effectively drive the adaptive mesh refinement.

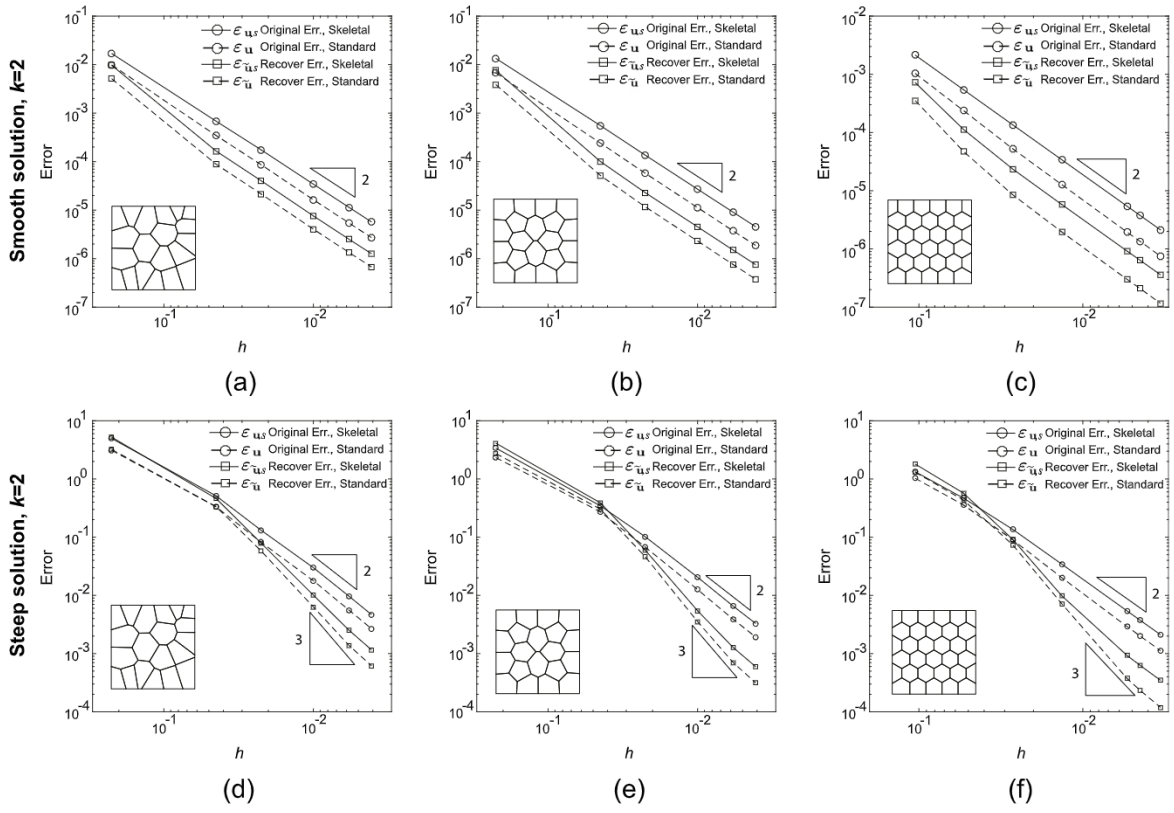

Fig. 11. Comparing the convergence behaviors between the skeletal errors ($\varepsilon_{\mathbf{u},s}$ and $\varepsilon_{\widetilde{\mathbf{u}},s}$) and the more standard ones ($\varepsilon_{\mathbf{u}}$ and $\varepsilon_{\widetilde{\mathbf{u}}}$) for 2D quadratic VEM. Smooth solution: (a) random Voronoi meshes, (b) CVT meshes, and (c) structured hexagonal meshes. Steep solution: (d) random Voronoi meshes, (e) CVT meshes, and (f) structured hexagonal meshes.

We first study the accuracy of the proposed gradient recovery scheme and the error estimators under uniform mesh refinement. Because we do not know the exact solution for this boundary value problem, only the estimated errors when linear ($k = 1$) and quadratic ($k = 2$) virtual elements are considered. For quadratic VEM, the gradient recovery are conducted excluding the internal DOFs. In Fig. 18(a)–(c), we plot the convergence of estimated global errors for both linear and quadratic virtual elements on Random Voronoi, CVT, and octagonal meshes, respectively. We can see that, for all families of meshes, the estimated errors all converge at a degenerated rate of roughly 0.5 irrespective of the order of the virtual element. This behavior agrees with theoretical results in the literature [65], indicating the effectiveness of the proposed error estimators. Moreover, we also depict the distributions of element-wise error estimators for linear and quadratic VEMs in fringe plots 19 and 20, respectively. It is immediate to appreciate that the proposed error estimators are able to capture the singularity of the displacement at the reentrant corner for both linear and quadratic meshes.

6.3. Numerical investigation in a 3D unit cube

In the last example, we present a 3D study on a unit cube $\Omega = (0, 1)^3$, as depicted in Fig. 21. The exact displacement solution $\mathbf{u} = [u_x, u_y, u_z]^T$ is given by

$$u_x = \sin(x)e^{y+2z}, \quad u_y = \cos(z) \quad \text{and} \quad u_z = x^3 - 2y^2, \tag{80}$$

which is applied on the entire boundary of the unit cube $\partial\Omega$. In this study, we consider various types of polyhedral discretizations: truncated octahedral, CVT, distorted hexahedral, and extruded octagonal meshes, as shown in Figs. 21(b)–(e), respectively. Only linear virtual elements are considered here.

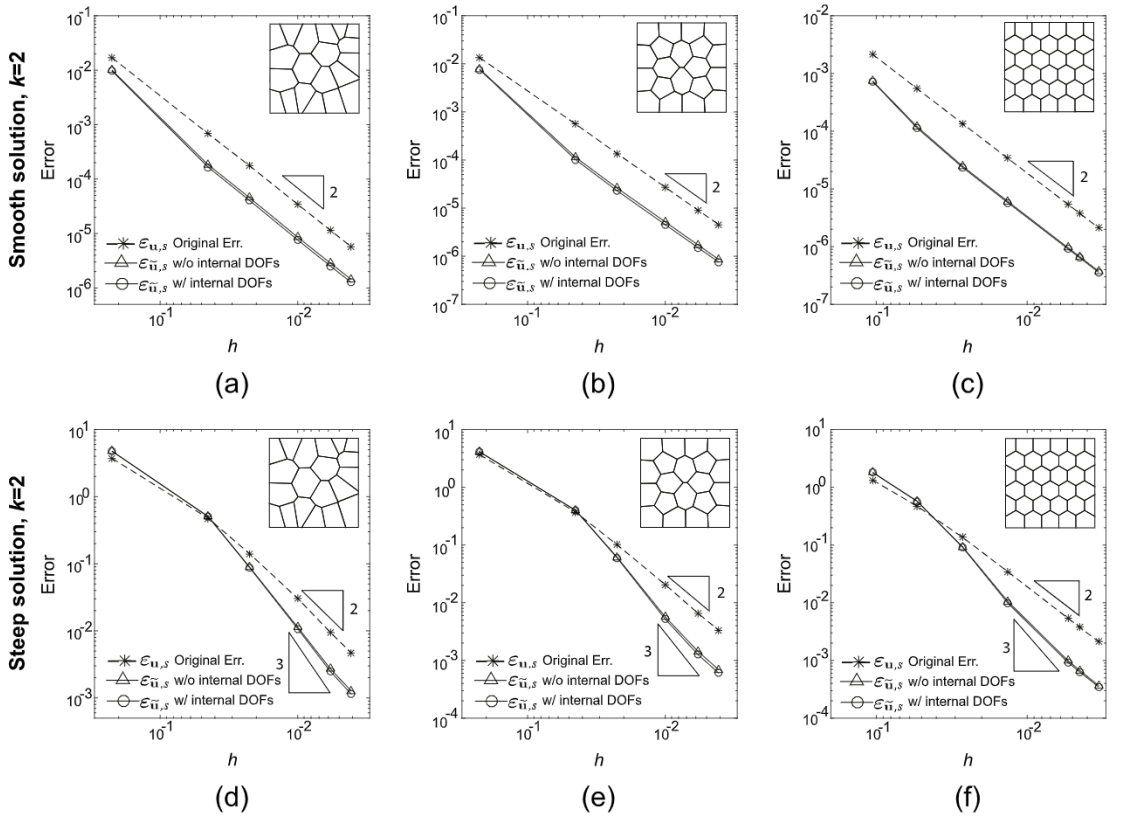

Fig. 12. Comparing the accuracy between the original errors ($\varepsilon_{\mathbf{u},s}$) and the recovered ones ($\varepsilon_{\widetilde{\mathbf{u}},s}$) for 2D quadratic VEM. The recovered errors include ones obtained from the proposed scheme and a SPR-type scheme introduced in Appendix. Smooth solution: (a) random Voronoi meshes, (b) CVT meshes, and (c) structured hexagonal meshes. Steep solution: (d) random Voronoi meshes, (e) CVT meshes, and (f) structured hexagonal meshes.

Similar to the 2D studies, we first verify the use of skeletal errors in 3D by comparing them with the standard L^2 errors of the displacement gradients. Figs. 22(a)–(d) show the comparison of these two types of errors for both original and recovered displacements on truncated octahedral, CVT, distorted hexahedral, and extruded octagonal meshes, respectively. The comparisons indicate that, for these mesh types, the skeletal errors agree well with the standard L^2 errors of the displacement gradients. Moreover, Figs. 22(a)–(d) compare the accuracy between the original and recovered displacement gradients and showcase that, for all types of meshes, the recovered displacement gradients are more accurate than the original ones. It is also interesting to note that, for the extruded octagonal meshes, the recovered displacement gradient exhibits super convergence — the rate of converge is 2. In addition, comparing with the SPR-type scheme introduced in Appendix, the proposed recovery scheme consistently gives more accurate recovered displacement gradients on all types of meshes.

Lastly, we investigate the accuracy of error estimators by comparing them with the exact errors. To that end, Figs. 23(a)–(d) show the convergence of both the estimated and exact global errors under mesh refinements for the truncated octahedral, CVT, distorted hexahedral, and extruded octagonal meshes, respectively. For the local error estimator, on the other hand, Figs. 24–27 show the fringe plots of the element-wise estimated and exact errors for those meshes. From the comparisons, we conclude that the error estimators (using both the proposed recovery scheme and the SPR-type scheme) can accurately predict the exact errors on both global and local levels.

7. On the use of the skeletal error estimators for adaptive mesh refinement

We study the effectiveness of the gradient recovery scheme and error estimators under adaptive mesh refinements. In this section, we restrict our attention to the CVT mesh and consider both linear and quadratic virtual elements. We

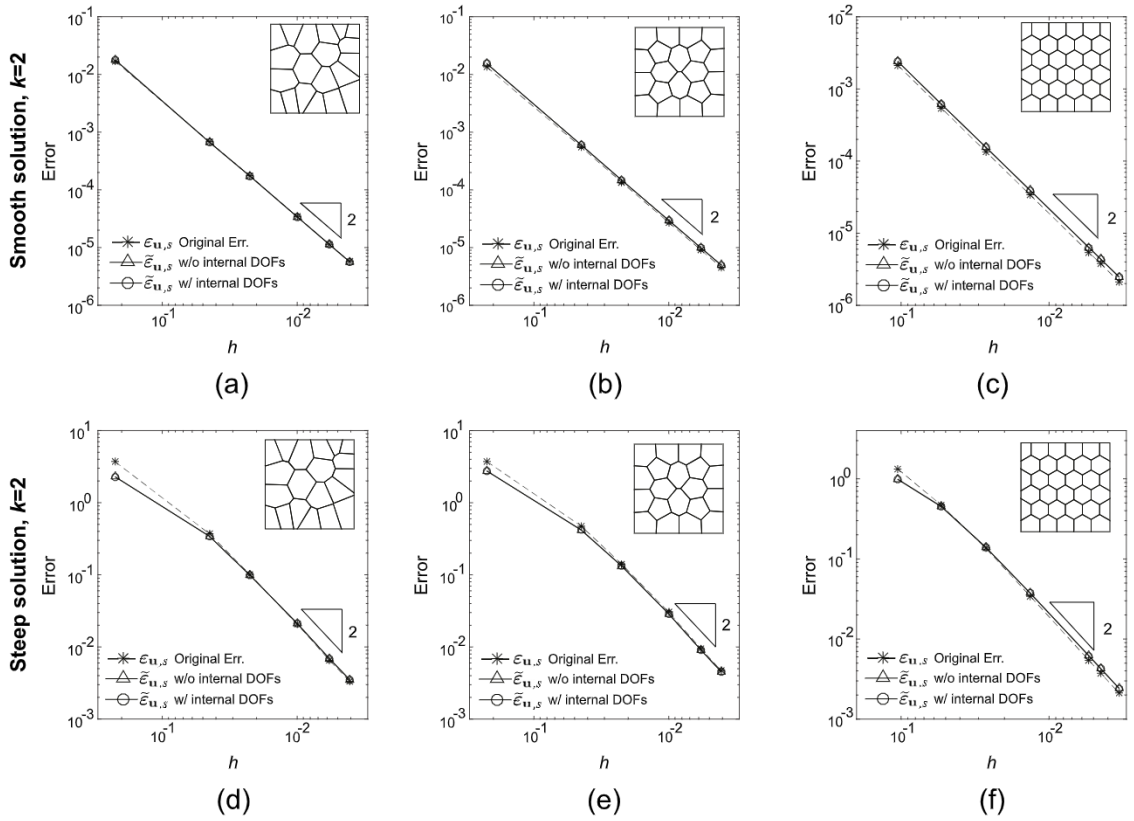

Fig. 13. Comparing the accuracy between the original errors ($\varepsilon_{\mathbf{u},s}$) and the estimated ones ($\widetilde{\varepsilon}_{\mathbf{u},s}$) for 2D quadratic VEM. The estimated errors include ones obtained from the proposed scheme and a SPR-type scheme introduced in Appendix. Smooth solution: (a) random Voronoi meshes, (b) CVT meshes, and (c) structured hexagonal meshes. Steep solution: (d) random Voronoi meshes, (e) CVT meshes, and (f) structured hexagonal meshes.

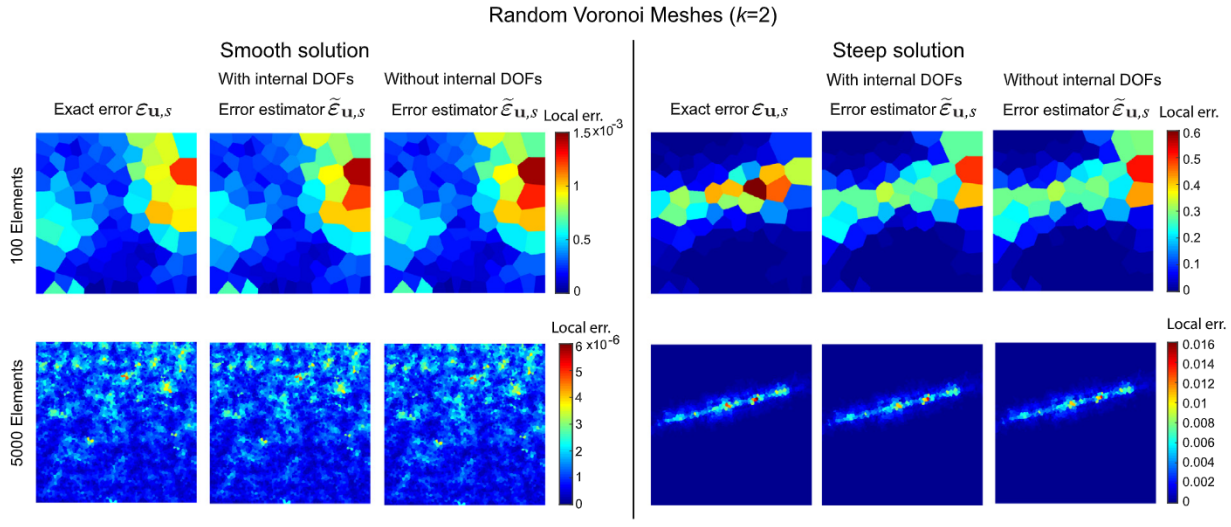

Fig. 14. Fringes plots of the element-level exact errors and estimated ones for 2D quadratic VEM on random Voronoi meshes.

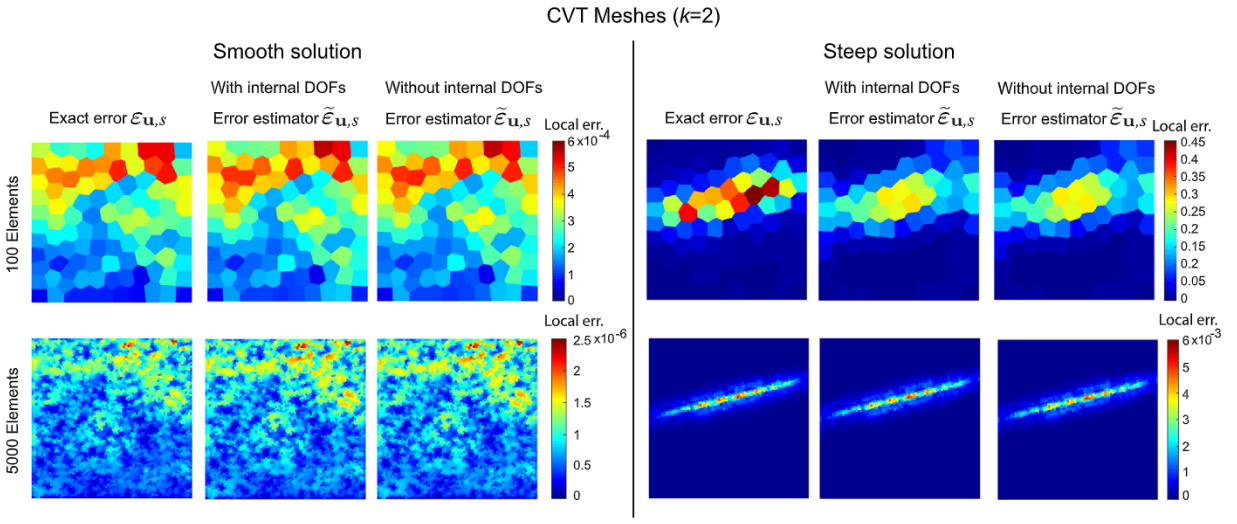

Fig. 15. Fringes plots of the element-level exact errors and estimated ones for 2D linear VEM on CVT meshes.

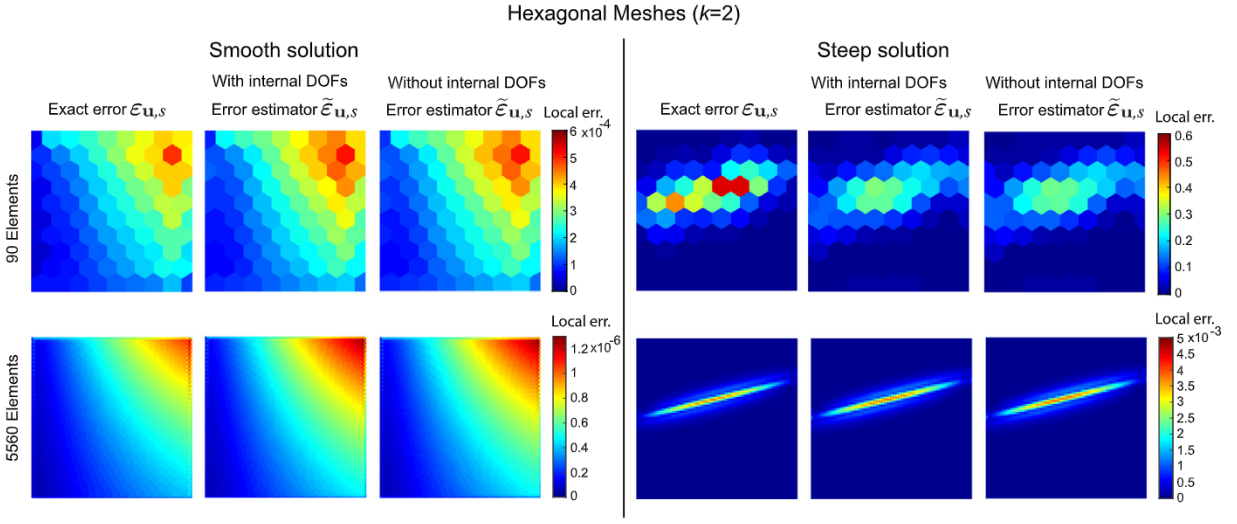

Fig. 16. Fringes plots of the element-level exact errors and estimated ones for 2D linear VEM on hexagonal meshes.

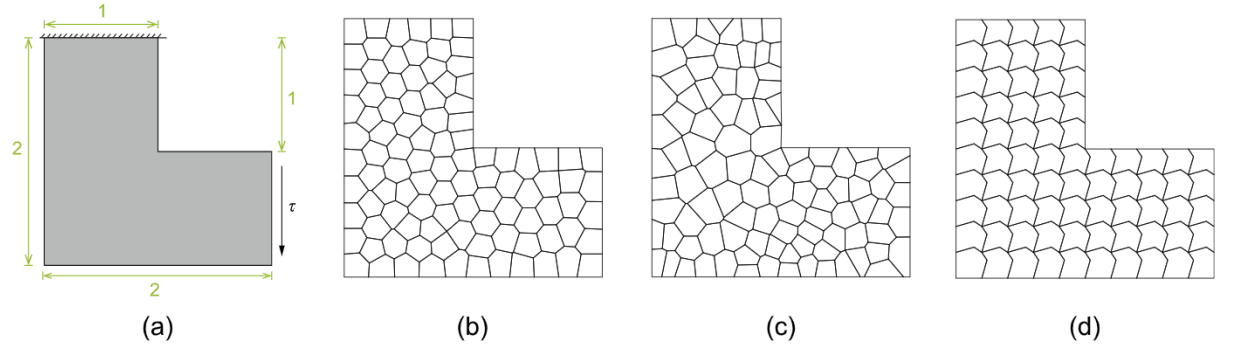

Fig. 17. (a) Dimensions, load and boundary conditions of the L-shape beam example. (b) An example of the random Voronoi mesh. (c) An example of the CVT mesh. (d) An example of the concave octagonal mesh.

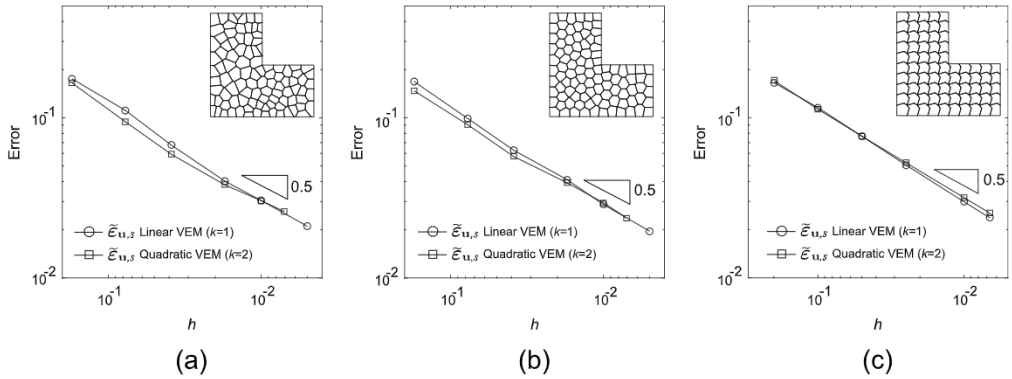

Fig. 18. Convergence of the estimated errors as functions of the average mesh size h for linear and quadratic VEMs on (a) random Voronoi meshes, (b) CVT meshes, and (c) concave octagonal meshes.

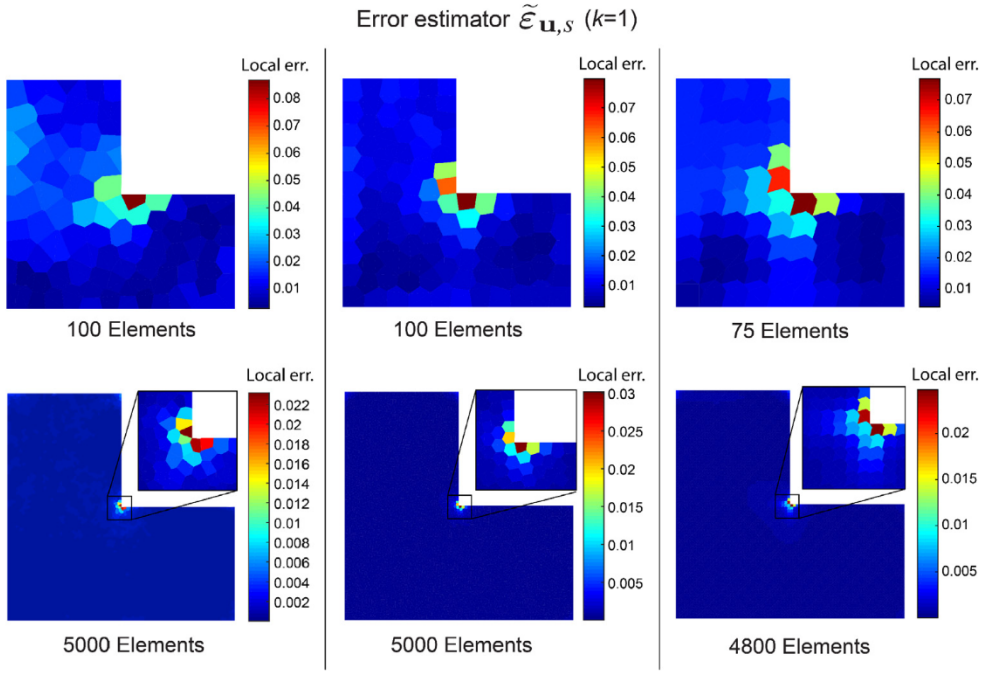

Fig. 19. Fringe plots of element-level estimated errors for linear VEM on random Voronoi meshes (left column), CVT meshes (middle column) and concave octagonal meshes (right column).

consider the L-shaped problem in Section 6.2 and take the initial mesh to be the CVT one shown in Fig. 17(c) with 100 elements. We then progressively refine this CVT mesh following a solve → estimate → mark → refine procedure: once the estimated error for each element is computed based on the VEM solution, we first mark those elements whose estimated errors are above $\theta \max_{E\in\Omega_h}(\widetilde{\varepsilon}_{\mathbf{u},s}|_E)$, where θ is a user-defined threshold set to be 0.2 throughout this example, and then perform refinement on those marked elements. In order to demonstrate the effectiveness of the proposed gradient recovery scheme and error estimators for a wide variety of adaptive refinement strategies, we consider the following three adaptive refinement strategies in the literature:

Strategy 1: In this strategy, each marked element is bisected through its centroid along the eigenvector corresponding to the smaller eigenvalue of the covariance matrix $\mathbf{M}_{\text{cov}}$ [66]. For element E, the covariance

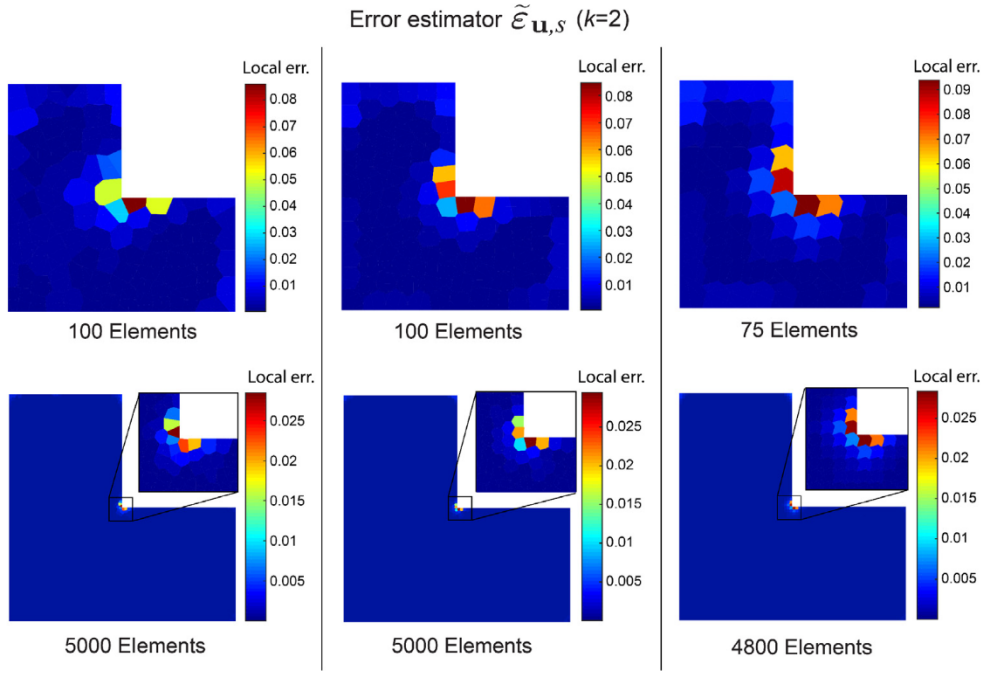

Fig. 20. Fringe plots of element-level estimated errors for quadratic VEM on random Voronoi meshes (left column), CVT meshes (middle column) and concave octagonal meshes (right column).

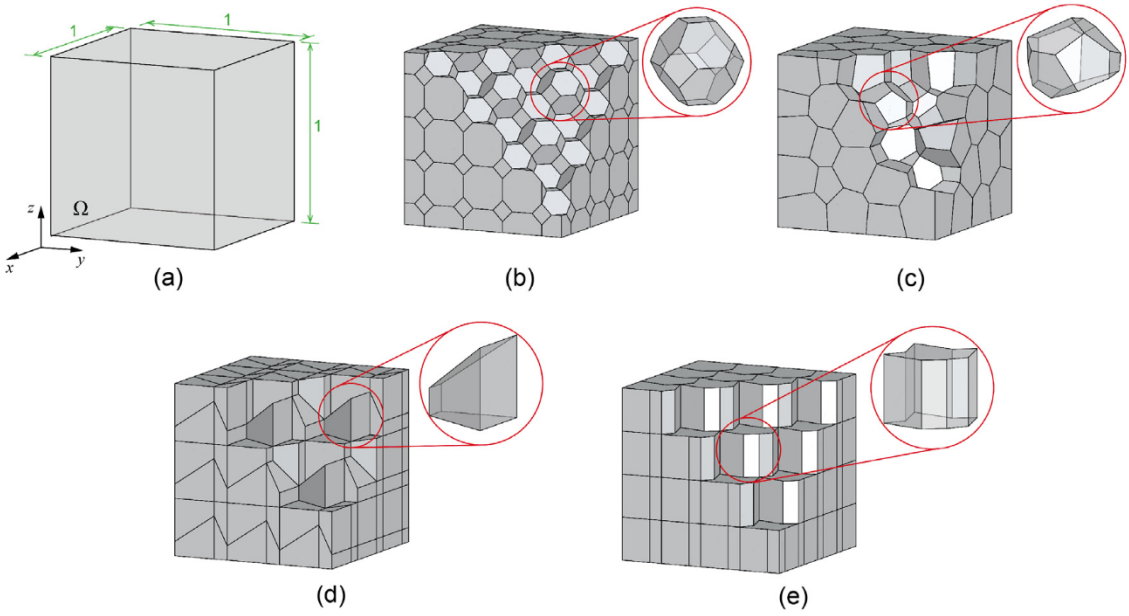

Fig. 21. (a) Problem setup and dimensions of the unit cube. (b) An example of the truncated octahedral meshes. (c) An example of the CVT meshes. (d) An example of the distorted hexahedral meshes. (e) An example of the extruded octagonal meshes.

matrix $\mathbf{M}_{\text{cov}}(E)$ is defined as

$$\mathbf{M}_{\text{cov}}(E) = \frac{1}{|E|}\int_E (\mathbf{x}-\mathbf{x}_c^E)\cdot(\mathbf{x}-\mathbf{x}_c^E)\mathrm{d}\mathbf{x}, \tag{81}$$

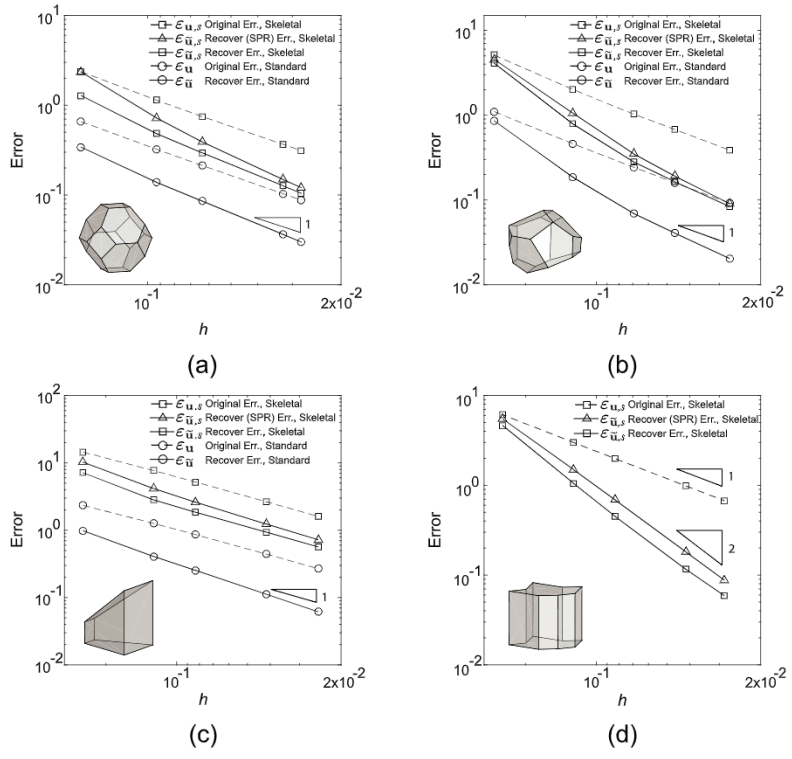

Fig. 22. Comparing the convergence behaviors between the skeletal errors ($\varepsilon_{\mathbf{u},s}$ and $\varepsilon_{\widetilde{\mathbf{u}},s}$) and the standard ones ($\varepsilon_{\mathbf{u}}$ and $\varepsilon_{\widetilde{\mathbf{u}}}$) for 3D linear VEM on: (a) truncated octahedral meshes, (b) CVT meshes, (c) distorted hexahedral meshes, and (d) extruded octagonal meshes. Because the extruded octagonal meshes contain concave elements, the 3D Wachspress shape functions are not applicable [62]. Thus, we do not consider the standard error for the extruded octagonal meshes.

where $\mathbf{x}_c^E$ is the centroid of E. A schematic illustration of this strategy is shown in Fig. 28(b).

Strategy 2: In this strategy, each marked element with n vertices is subdivided into n quadrilateral elements by connecting the midpoint of each edge to its centroid [51,67]. A schematic illustration of this strategy is shown in Fig. 28(c).

Strategy 3: In this strategy, each marked element with n vertices is subdivided into $n+1$ CVT elements using the Lloyd's algorithm. The initial seeds of the Lloyd's algorithm are placed at the centroid this element as well as the midpoints of the lines connecting the centroid and the vertices [68,69]. A schematic illustration of this strategy is shown in Fig. 28(d).

We note that the above adaptive refinement strategies will naturally introduce hanging nodes to the elements adjacent to the ones that are refined. In our implementation of both Strategies 2 and 3, in order to save degrees of freedom, those hanging nodes are not regarded as regular vertices. For instance, in Strategy 2, let us assume that a quadrilateral element with one hanging nodes needs to be refined. If the hanging node is considered as a regular vertex of that element, then the element will be subdivided into five elements. Instead, our implementation does not consider this hanging node as a regular vertex and, thus, will split this element into only four elements.

Figs. 29(a) and (b) show the convergence of the estimated global errors $\widetilde{\varepsilon}_{\mathbf{u},s}$ as functions of the total number of nodes with the above-mentioned adaptive mesh refinement strategies using linear and quadratic virtual elements, respectively. It is observed from the both figures that, unlike the cases of uniform refinement shown in Figs. 18(a)–(c), all the three adaptive mesh refine strategies driven by the proposed error estimator are able to restore the optimal convergence rates of the H^1-type displacement error (i.e. 0.5 for linear and 1 for quadratic VEM) with respect to the total numbers of nodes, demonstrating the effectiveness as well as the flexibility of the proposed gradient recovery scheme and error estimator. Moreover, three representative meshes are selected for each strategy at three similar

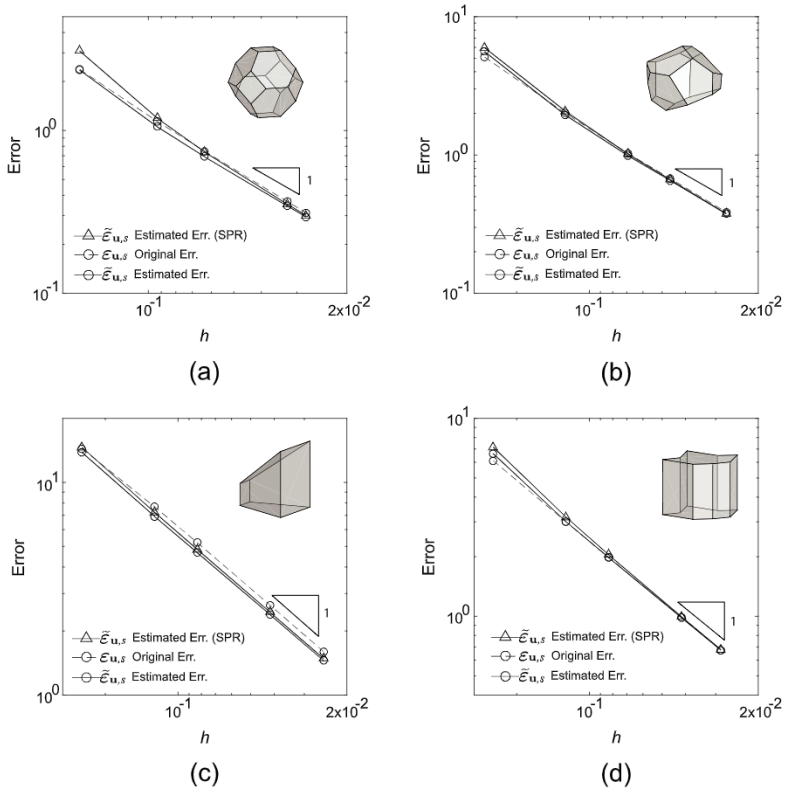

Fig. 23. Comparing the accuracy between the original errors ($\varepsilon_{\mathbf{u},s}$) and the estimated ones ($\widetilde{\varepsilon}_{\mathbf{u},s}$) for 3D linear VEM on: (a) truncated octahedral meshes, (b) CVT meshes, (c) distorted hexahedral meshes, and (d) extruded octagonal meshes. The estimated errors include ones obtained from the proposed scheme and a SPR-type scheme introduced in Appendix.

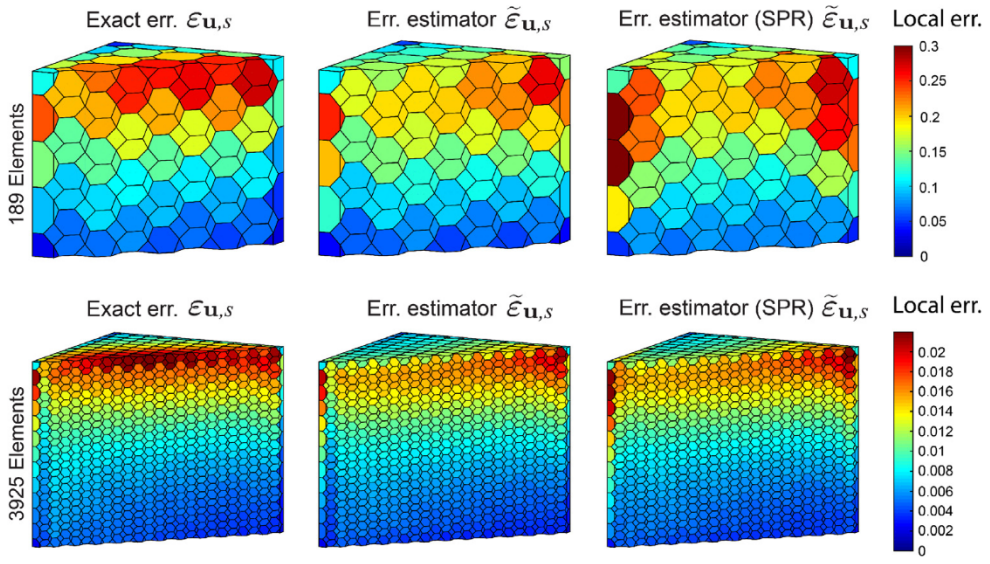

Fig. 24. Fringes plots of the element-level exact errors and estimated ones for 3D linear VEM on truncated octahedral meshes.

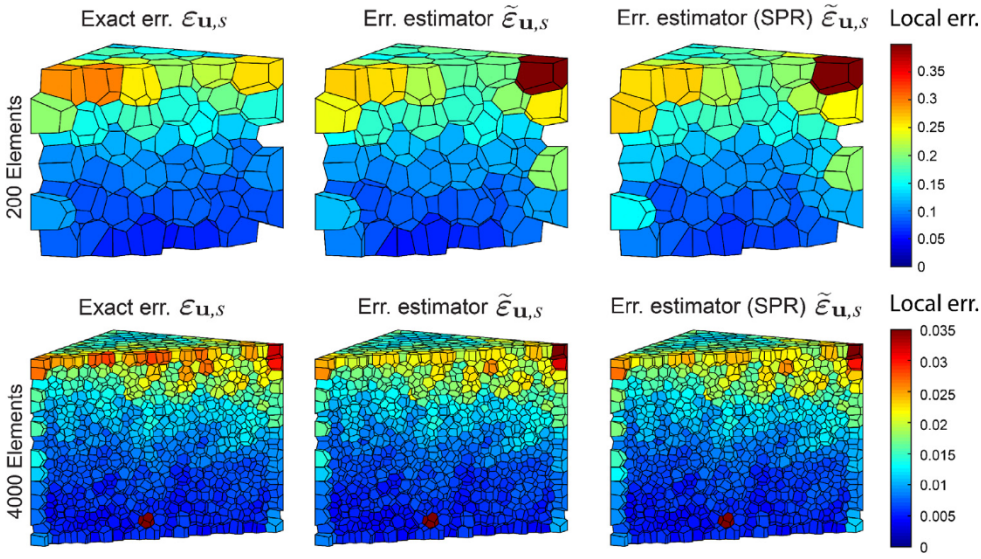

Fig. 25. Fringes plots of the element-level exact errors and estimated ones for 3D linear VEM on CVT meshes.

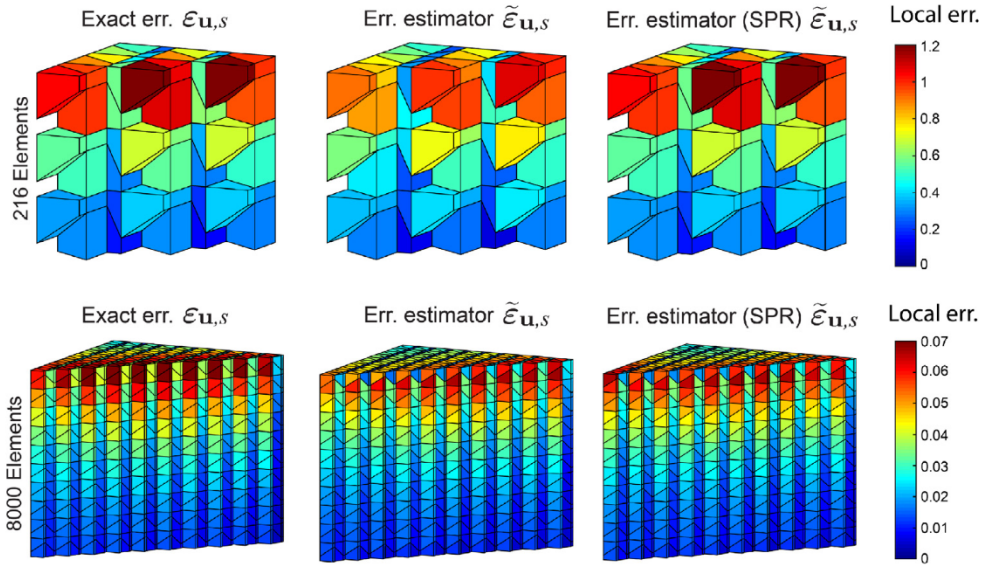

Fig. 26. Fringes plots of the element-level exact errors and estimated ones for 3D linear VEM on distorted hexahedral meshes.

global error levels and are plotted in Figs. 30 and 31 for linear and quadratic VEMs, respectively. We conclude from both figures that all the three adaptive refinement strategies effectively capture the problematic regions of the domain: the reentrant corner and the two ends of the clamped top edge. We also observe that, compared to the refinements with linear virtual elements, the ones with quadratic virtual elements are more localized to those problematic regions because of the higher potential local accuracy in the approximation. Finally, we comment that, for both linear and quadratic VEMs, Strategy 2 seems to produce the most accurate result under similar number of nodes (although this may be problem-dependent), and Strategy 3 is more aggressive than the other ones in the sense that it achieves similar levels of global errors using less numbers of refinements.

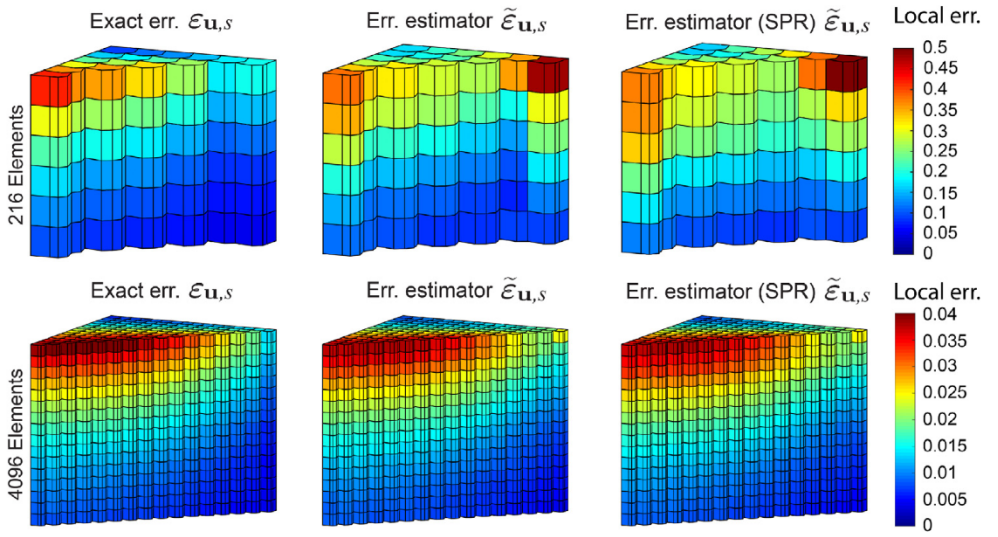

Fig. 27. Fringes plots of the element-level exact errors and estimated ones for 3D linear VEM on extruded octagonal meshes.

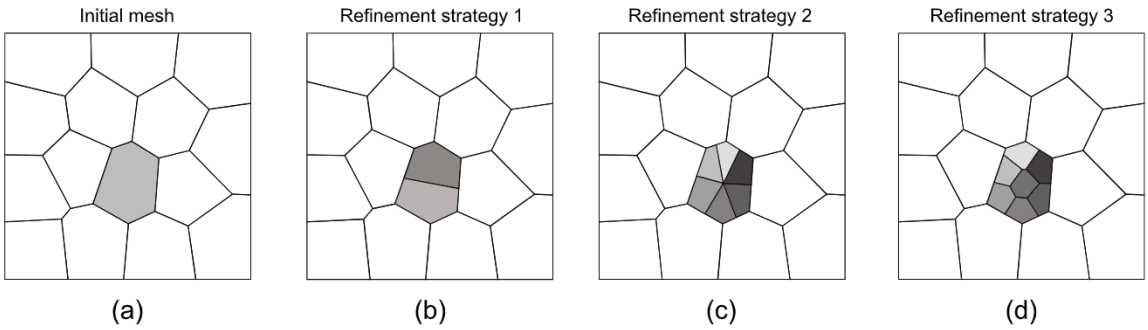

Fig. 28. (a) An illustration of an initial CVT mesh with the gray element being marked for refinement. (b) A schematic illustration of the adaptive refinement strategy 1, where the marked element is bisected into two elements. (c) A schematic illustration of the adaptive refinement strategy 2, where the marked element is subdivided into 6 quadrilateral elements. (d) A schematic illustration of the adaptive refinement strategy 3, where the marked element tessellated into 7 CVT elements.

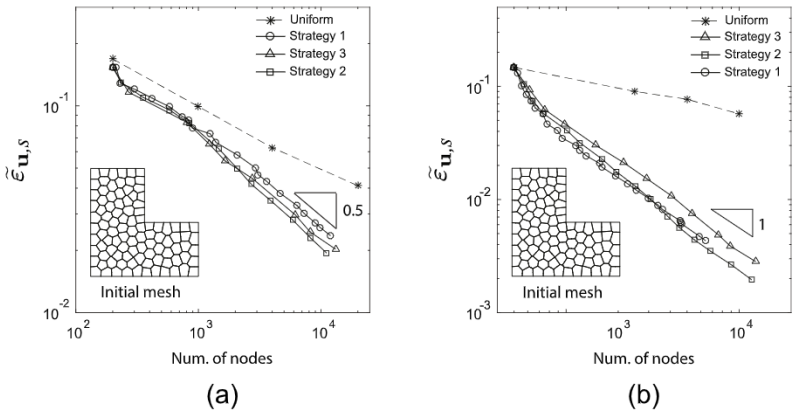

Fig. 29. Convergence of the estimated errors as functions of the total numbers of nodes with the three adaptive mesh refinement strategies for (a) linear VEM, and (b) quadratic VEM.

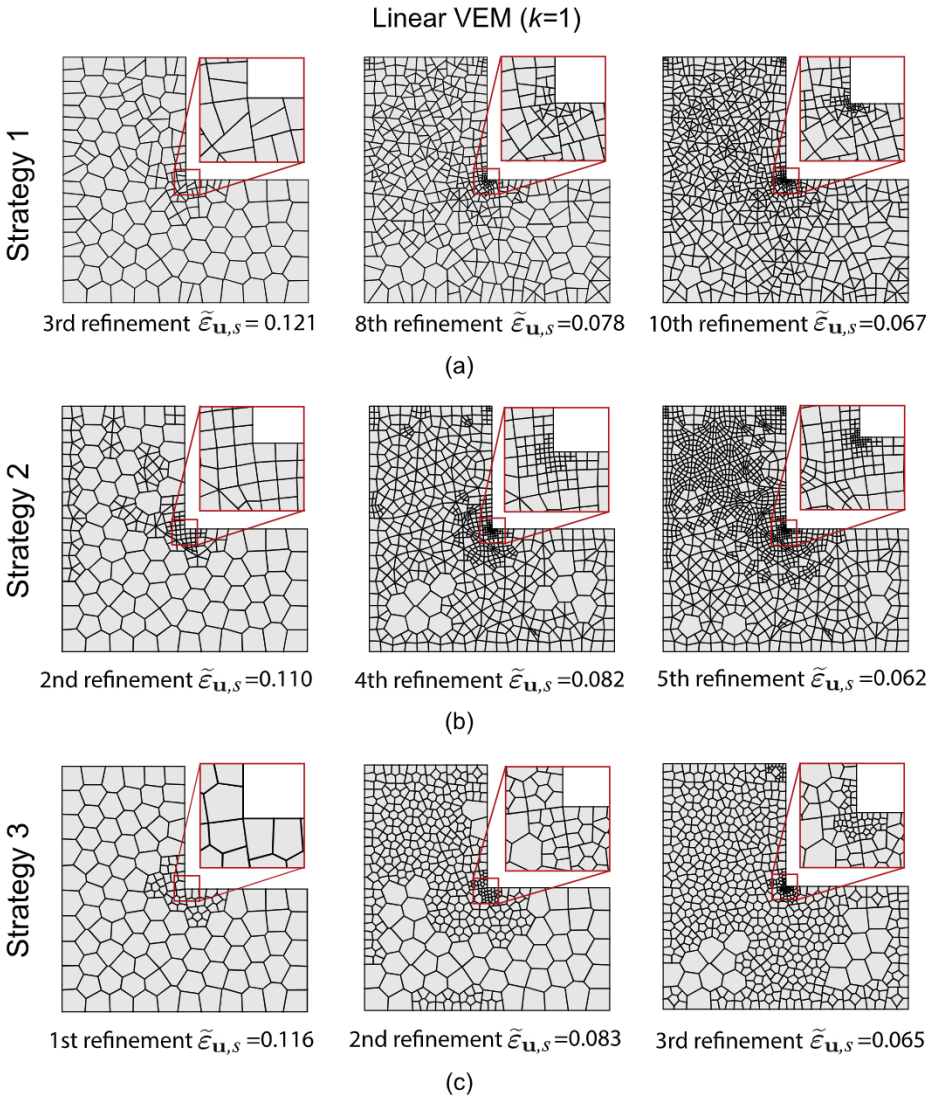

Fig. 30. When linear virtual elements are considered, some representative meshes using (a) adaptive refinement strategy 1, (b) adaptive refinement strategy 2, and (c) adaptive refinement strategy 3. The representative meshes are selected so that the three meshes in each column of this figure have a similar level of estimated global error.

8. Concluding remarks

This paper introduces a general recovery-based *a posteriori* error estimation framework for lower and higher order VEM on polygonal and polyhedral meshes, and demonstrates the theory in the context of linear elasticity. Within the error estimation framework, we first recover a more accurate displacement gradient through least square fitting of the displacement DOFs over each patch in the mesh. Based on the recovered gradient, an error estimator is obtained by evaluating the difference between the recovered and original displacement gradients on the skeleton of mesh. This skeletal error is shown in the numerical studies to capture the standard L^2 norm of the displacement gradient error extremely well. The introduced error estimation is shown to be accurate for both linear and quadratic virtual elements on various polygonal/polyhedral meshes and with various types of displacement solutions (e.g. smooth ones, ones with sharp gradients, and ones containing singularities). For linear VEM, the accuracy and effectiveness of the proposed error estimation framework are further demonstrated by comparing it with a SPR-type error estimation as outlined in the Appendix. For higher-order VEM, the numerical studies also suggest that we can neglect the internal

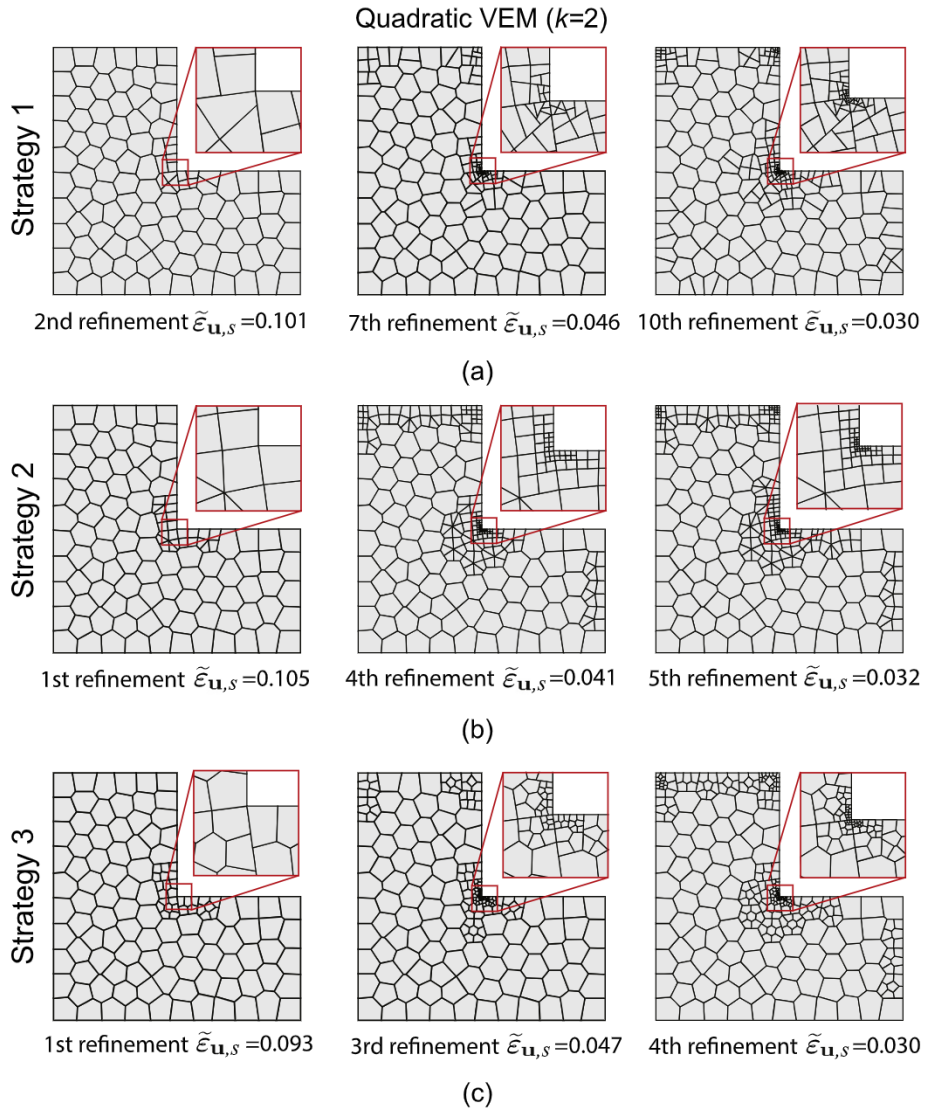

Fig. 31. When quadratic virtual elements are considered, some representative meshes using (a) adaptive refinement strategy 1, (b) adaptive refinement strategy 2, and (c) adaptive refinement strategy 3. The representative meshes are selected so that the three meshes in each column of this figure have a similar level of estimated global error.

displacement DOFs (which are in the form of function moments over the elements) in the gradient recovery procedure without sacrificing accuracy. We note that, although being proposed for VEM and linear elasticity problems, the error estimation framework can also be readily employed for polygonal/polyhedral FEM and nonlinear elasticity problems.

In terms of future work, we remark that the proposed recovery-based *a posteriori* error estimation offers an effective tool for adaptive VEM analysis. Extensions of this research include developing novel and efficient mesh adaption strategies for VEM on polygonal/polyhedral discretization. Another promising area of research consists of extending the ideas presented here to nonlinear fracture mechanics [70] and space-time evolution problems.

Acknowledgments

HC and GHP acknowledge support from the US National Science Foundation (NSF) under grant #1624232 (formerly #1437535). We also acknowledge support from the Raymond Allen Jones Chair at the Georgia Institute

of Technology. The information presented in this paper is the sole opinion of the authors and does not necessarily reflect the views of the sponsoring agencies.

Appendix. A SPR-type recovery scheme of linear VEM

To complement the discussions in Section 5, this appendix presents a SPR-type scheme, which is conceptually similar to the SPR in the FEM literature [35,36,44], for 2D and 3D linear VEM. We will particularly focus on the reconstructing displacement gradient at the vertices of the mesh. A similar scheme is used in [71] to recover the stress at the vertices of the mesh for 2D linear elastic fracture mechanics problems. We note that, unlike the gradient recovery scheme proposed in Section 4, the extension of this SPR-type scheme to higher order VEMs is not straightforward.

Following the notations introduced in Section 4 and for a given patch ω_i, the basic idea of this SPR-type recovery scheme is to seek a linear tensorial field, denoted by $\mathbf{g}^{\omega_i}(\mathbf{x}) \in [\mathcal{P}_1(\omega_i)]^{d\times d}$, such that

$$\mathbf{g}^{\omega_i} = \underset{\boldsymbol{\xi}\in[\mathcal{P}_1(\omega_i)]^{d\times d}}{\operatorname{argmin}} \sum_{j=1}^{N_{\omega_i}^E} \left[\boldsymbol{\xi}(\mathbf{x}_c^{E_j}) - \boldsymbol{\Pi}_0^0(\nabla\mathbf{u}_h)|_{E_j}\right] : \left[\boldsymbol{\xi}(\mathbf{x}_c^{E_j}) - \boldsymbol{\Pi}_0^0(\nabla\mathbf{u}_h)|_{E_j}\right], \tag{82}$$

where $N_{\omega_i}^E$ is the total number of elements in patch ω_i, E_j is the jth element in ω_i with its centroid being $\mathbf{x}_c^{E_j}$, and we recall from (23) that $\boldsymbol{\Pi}_0^0$ is the L^2 projection of the displacement gradient onto constant functions. With $\mathbf{g}^{\omega_i}$ obtained, the value of the reconstructed gradient $G_h\mathbf{u}_h$ at $\mathbf{x}_i$ is given by

$$G_h\mathbf{u}_h(\mathbf{x}_i) = \mathbf{g}^{\omega_i}(\mathbf{x}_i). \tag{83}$$

Having computed $G_h\mathbf{u}_h$ at every vertex of the mesh, the remaining steps of the SPR-type scheme are identical to the ones outlined in Section 4.1.4.

Finally, we remark that the rules for choosing patches in this SPR-type scheme are the same as the ones described in Section 4.2 except for the threshold on the number of elements which decides whether to enlarge the patch. In this SPR-type scheme, a necessary condition for the recovery (82) to have unique solution is $N_{\omega_i}^E \geq 3$ in 2D and $N_{\omega_i}^E \geq 4$ in 3D. Thus, the threshold for the SPR-type scheme is set to be 3 in 2D and 4 in 3D. For any patch that contains less number of elements than the threshold, the patch will be enlarged in the recovery process.

References

[1] L. Beirão da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L.D. Marini, A. Russo, Basic principles of virtual element methods, Math. Models Methods Appl. Sci. 23 (1) (2013) 199–214.
[2] H. Chi, C. Talischi, O. Lopez-Pamies, G.H. Paulino, Polygonal finite elements for finite elasticity, Internat. J. Numer. Methods Engrg. 101 (2015) 305–328.
[3] H. Chi, C. Talischi, O. Lopez-Pamies, G.H. Paulino, A paradigm for higher order polygonal elements in finite elasticity, Compuat. Methods Appl. Mech. Eng. 306 (2016) 216–251.
[4] C. Talischi, G.H. Paulino, A. Pereira, I.F.M. Menezes, PolyTop: A Matlab implementation of a general topology optimization framework using unstructured polygonal finite element meshes, Struct. Multidiscip. Optim. 45 (3) (2012) 329–357.
[5] C. Talischi, G.H. Paulino, A. Pereira, I.F.M. Menezes, PolyMesher: A general-purpose mesh generator for polygonal elements written in Matlab, Struct. Multidiscip. Optim. 45 (3) (2012) 309–328.
[6] C. Talischi, A. Pereira, G.H. Paulino, I.F.M. Menezes, M.S. Carvalho, Polygonal finite elements for incompressible fluid flow, Internat. J. Numer. Methods Fluids 74 (2014) 134–151.
[7] N. Sukumar, E.A. Malsch, Recent advances in the construction of polygonal finite element interpolations, Arch. Comput. Methods Eng. 13 (1) (2006) 129–163.
[8] N. Sukumar, A. Tabarraei, Conforming polygonal finite elements, Internat. J. Numer. Methods Engrg. 61 (12) (2004) 2045–2066.
[9] N. Sukumar, Construction of polygonal interpolants: A maximum entropy approach, Internat. J. Numer. Methods Engrg. 61 (12) (2004) 2159–2181.
[10] D.W. Spring, S.E. Leon, G.H. Paulino, Unstructured polygonal meshes with adaptive refinement for the numerical simulation of dynamic cohesive fracture, Int. J. Fract. 189 (1) (2014) 33–57.
[11] S.O.R. Biabanaki, A.R. Khoei, A polygonal finite element method for modeling arbitrary interfaces in large deformation problems, Comput. Mech. 50 (1) (2012) 19–33.
[12] S. Rjasanow, S. Weißer, Higher order BEM-based FEM on polygonal meshes, SIAM J. Numer. Anal. 50 (5) (2012) 2357–2378.
[13] J.E. Bishop, Simulating the pervasive fracture of materials and structures using randomly close packed Voronoi tessellations, Comput. Mech. 44 (4) (2009) 455–471.
[14] S.O.R. Biabanaki, A.R. Khoei, P. Wriggers, Polygonal finite element methods for contact-impact problems on non-conformal meshes, Comput. Methods Appl. Mech. Engrg. 269 (2014) 198–221.

[15] L. Beirão da Veiga, F. Brezzi, L. Marini, A. Russo, The hitchhiker's guide to the virtual element method, Math. Models Methods Appl. Sci. 24 (08) (2014) 1541–1573.
[16] H. Chi, L. Beirão da Veiga, G. Paulino, Some basic formulations of the virtual element method (VEM) for finite deformations, Comput. Methods Appl. Mech. Engrg. 318 (2017) 148–192.
[17] G.H. Paulino, A.L. Gain, Bridging art and engineering using Escher-based virtual elements, Struct. Multidiscip. Optim. 51 (4) (2015) 867–883.
[18] L. Beirão da Veiga, F. Brezzi, L.D. Marini, A. Russo, $H(\mathrm{div})$ and $H(\mathrm{curl})$-conforming virtual element methods, Numer. Math. 133 (2) (2016) 303–332.
[19] L. Beirão da Veiga, F. Brezzi, L.D. Marini, Virtual elements for linear elasticity problems, SIAM J. Numer. Anal. 51 (2) (2013) 794–812.
[20] A.L. Gain, C. Talischi, G.H. Paulino, On the virtual element method for three-dimensional linear elasticity problems on arbitrary polyhedral meshes, Comput. Methods Appl. Mech. Engrg. 282 (2014) 132–160.
[21] E. Artioli, S. De Miranda, C. Lovadina, L. Patruno, A stress/displacement virtual element method for plane elasticity problems, Comput. Methods Appl. Mech. Engrg. 325 (2017) 155–174.
[22] L. Beirão da Veiga, C. Lovadina, D. Mora, A virtual element method for elastic and inelastic problems on polytope meshes, Comput. Methods Appl. Mech. Engrg. 295 (2015) 327–346.
[23] E. Artioli, L. Beirão da Veiga, C. Lovadina, E. Sacco, Arbitrary order 2D virtual elements for polygonal meshes: Part I, elastic problem, Comput. Mech. 60 (2017) 355–377.
[24] E. Artioli, L. Beirão da Veiga, C. Lovadina, E. Sacco, Arbitrary order 2D virtual elements for polygonal meshes: Part II, inelastic problem, Comput. Mech. 60 (4) (2017) 643–657.
[25] R.L. Taylor, E. Artioli, VEM for inelastic solids, in: E. Oñate, D. Peric, E. de Souza Neto, M. Chiumenti (Eds.), Advances in Computational Plasticity: A Book in Honour of D. Roger J. Owen, Springer, Cham, 2018, pp. 381–394.
[26] P. Wriggers, B. Reddy, W. Rust, B. Hudobivnik, Efficient virtual element formulations for compressible and incompressible finite deformations, Comput. Mech. 60 (2017) 253–268.
[27] P. Wriggers, B. Hudobivnik, A low order virtual element formulation for finite elasto-plastic deformations, Comput. Methods Appl. Mech. Engrg. 327 (2017) 459–477.
[28] F. Brezzi, L.D. Marini, Virtual element methods for plate bending problems, Comput. Methods Appl. Mech. Engrg. 253 (2013) 455–462.
[29] P. Antonietti, G. Manzini, M. Verani, The fully nonconforming virtual element method for biharmonic problems, 2016. arXiv preprint arXiv:1611.08736.
[30] D. Mora, G. Rivera, I. Velásquez, A virtual element method for the vibration problem of Kirchhoff plates, 2017. arXiv preprint arXiv:1703.04187.
[31] J. Zhao, S. Chen, B. Zhang, The nonconforming virtual element method for plate bending problems, Math. Models Methods Appl. Sci. 26 (09) (2016) 1671–1687.
[32] P. Wriggers, W. Rust, B. Reddy, A virtual element method for contact, Comput. Mech. 58 (6) (2016) 1039–1050.
[33] F. Bassi, L. Botti, A. Colombo, Agglomeration-based physical frame dG discretizations: An attempt to be mesh free, Math. Models Methods Appl. Sci. 24 (08) (2014) 1495–1539.
[34] F. Bassi, L. Botti, A. Colombo, D.A. Di Pietro, P. Tesini, On the flexibility of agglomeration based physical space discontinuous galerkin discretizations, J. Comput. Phys. 231 (1) (2012) 45–65.
[35] O.C. Zienkiewicz, J. Zhu, The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique, Internat. J. Numer. Methods Engrg. 33 (7) (1992) 1331–1364.
[36] O.C. Zienkiewicz, J.Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. Part 2: Error estimates and adaptivity, Internat. J. Numer. Methods Engrg. 33 (7) (1992) 1365–1382.
[37] G. Maisano, S. Micheletti, S. Perotto, C. Bottasso, On some new recovery-based a posteriori error estimators, Comput. Methods Appl. Mech. Engrg. 195 (37) (2006) 4794–4815.
[38] A. Naga, Z. Zhang, A posteriori error estimates based on the polynomial preserving recovery, SIAM J. Numer. Anal. 42 (4) (2004) 1780–1800.
[39] M. Ainsworth, A. Craig, A posteriori error estimators in the finite element method, Numer. Math. 60 (1) (1991) 429–463.
[40] B. Boroomand, O. Zienkiewicz, Recovery by equilibrium in patches (REP), Internat. J. Numer. Methods Engrg. 40 (1) (1997) 137–164.
[41] M. Ainsworth, J.T. Oden, A Posteriori Error Estimation in Finite Element Analysis, Vol. 37, John Wiley & Sons, 2011.
[42] T. Blacker, T. Belytschko, Superconvergent patch recovery with equilibrium and conjoint interpolant enhancements, Internat. J. Numer. Methods Engrg. 37 (3) (1994) 517–536.
[43] R. Verfürth, A Posteriori Error Estimation Techniques for Finite Element Methods, OUP Oxford, 2013.
[44] O. Zienkiewicz, J. Zhu, The superconvergent patch recovery (SPR) and adaptive finite element refinement, Comput. Methods Appl. Mech. Engrg. 101 (1–3) (1992) 207–224.
[45] K. Balaji, R. Amirtham, S. Paul, Adaptive Poly-FEM for the analysis of plane elasticity problems, Int. J. Comput. Methods Eng. Sci. Mech. 18 (2–3) (2017) 146–165.
[46] B. Boroomand, O. Zienkiewicz, An improved REP recovery and the effectivity robustness test, Internat. J. Numer. Methods Engrg 40 (17) (1997) 3247–3277.
[47] Z. Zhang, A. Naga, A new finite element gradient recovery method: Superconvergence property, SIAM J. Sci. Comput. 26 (4) (2005) 1192–1213.
[48] A. Naga, Z. Zhang, The polynomial-preserving recovery for higher order finite element methods in 2D and 3D, Discrete Contin. Dyn. Syst. Ser. B 5 (3) (2005) 769.
[49] Z. Zhang, Polynomial preserving gradient recovery and a posteriori estimate for bilinear element on irregular quadrilaterals, Int. J. Numer. Anal. Model 1 (1) (2004) 1–24.

[50] L. Beirão da Veiga, G. Manzini, Residual a posteriori error estimation for the virtual element method for elliptic problems, ESAIM Math. Model. Numer. Anal. 49 (2) (2015) 577–599.
[51] A. Cangiani, E.H. Georgoulis, T. Pryer, O.J. Sutton, A posteriori error estimates for the virtual element method, Numer. Math. 137 (4) (2017) 857–893.
[52] D. Mora, G. Rivera, R. Rodríguez, A posteriori error estimates for a Virtual Element Method for the Steklov eigenvalue problem, Comput. Math. Appl. 74 (2017) 2172–2190.
[53] S. Berrone, A. Borio, A residual a posteriori error estimate for the virtual element method, Math. Models Methods Appl. Sci. 27 1423–1458.
[54] L. Beirão da Veiga, F. Dassi, A. Russo, High-order virtual element method on polyhedral meshes, Comput. Math. Appl. 74 (2017) 1110–1122.
[55] B. Ahmad, A. Alsaedi, F. Brezzi, L.D. Marini, A. Russo, Equivalent projectors for virtual element methods, Comput. Math. Appl. 66 (3) (2013) 376–391.
[56] E.B. Chin, J.B. Lasserre, N. Sukumar, Numerical integration of homogeneous functions on convex and nonconvex polygons and polyhedra, Comput. Mech. 56 (6) (2015) 967–981.
[57] A. Sommariva, M. Vianello, Gauss–Green cubature and moment computation over arbitrary geometries, J. Comput. Appl. Math. 231 (2) (2009) 886–896.
[58] F. Brezzi, A. Buffa, K. Lipnikov, Mimetic finite differences for elliptic problems, Math. Model. Numer. Anal. 43 (2) (2009) 277–295.
[59] L. Beirão da Veiga, K. Lipnikov, G. Manzini, Arbitrary-order nodal mimetic discretizations of elliptic problems on polygonal meshes, SIAM J. Numer. Anal. 49 (5) (2011) 1737–1760.
[60] P. Antonietti, L. Beirão da Veiga, S. Scacchi, M. Verani, A C^1 virtual element method for the Cahn–Hilliard equation with polygonal meshes, SIAM J. Numer. Anal. 54 (1) (2016) 34–56.
[61] M.S. Floater, Mean value coordinates, Comput. Aided Geom. Design 20 (1) (2003) 19–27.
[62] M. Floater, A. Gillette, N. Sukumar, Gradient bounds for Wachspress coordinates on polytopes, SIAM J. Numer. Anal. 52 (1) (2014) 515–532.
[63] A. Rand, A. Gillette, C. Bajaj, Quadratic serendipity finite elements on polygons using generalized barycentric coordinates, Math. Comput. 83 (290) (2014) 2691–2716.
[64] L. Beirão da Veiga, C. Lovadina, A. Russo, Stability analysis for the virtual element method, Math. Models Methods Appl. Sci. 27 (13) (2017) 2557–2594.
[65] G. Strang, G.J. Fix, An Analysis of the Finite Element Method, Vol 212, Prentice-Hall Englewood Cliffs, NJ, 1973.
[66] S. Weißer, Residual error estimate for BEM-based FEM on polygonal meshes, Numer. Math. 118 (4) (2011) 765–788.
[67] A.V. Astaneh, F. Fuentes, J. Mora, L. Demkowicz, High-order polygonal discontinuous Petrov–Galerkin (PolyDPG) methods using ultraweak formulations, Comput. Methods Appl. Mech. Engrg. 332 (2018) 686–711.
[68] E.T. Filipov, J. Chun, G.H. Paulino, J. Song, Polygonal multiresolution topology optimization (PolyMTOP) for structural dynamics, Struct. Multidiscip. Optim. 53 (4) (2016) 673–694.
[69] H. Nguyen-Xuan, A polytree-based adaptive polygonal finite element method for topology optimization, Internat. J. Numer. Methods Engrg. 110 (10) (2017) 972–1000.
[70] K. Park, G.H. Paulino, W. Celes, R. Espinha, Adaptive mesh refinement and coarsening for cohesive zone modeling of dynamic fracture, Internat. J. Numer. Methods Engrg. 92 (1) (2012) 1–35.
[71] V.M. Nguyen-Thanh, X. Zhuang, H. Nguyen-Xuan, T. Rabczuk, P. Wriggers, A Virtual Element Method for 2D linear elastic fracture analysis, Comput. Methods Appl. Mech. Engrg. (2018).