Journal of Computational and Applied Mathematics 307 (2016) 119–133

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

# Polynomial preserving recovery on boundary

Hailong Guo [a,1], Zhimin Zhang [b,a], Ren Zhao [a], Qingsong Zou [c,*]

[a] Department of Mathematics, Wayne State University, Detroit, MI 48202, United States

[b] Beijing Computational Science Research Center, Beijing 100084, China

[c] School of Mathematics and Computational Science, Sun Yat-sen University, Guangzhou, 510275, PR China

ARTICLE INFO

*Article history:*
Received 19 April 2015
Received in revised form 27 February 2016

*MSC:*
65N50
65N30
65N15

*Keywords:*
Gradient recovery
Superconvergence
Polynomial preserving

ABSTRACT

In this paper, we propose two systematic strategies to recover the gradient on the boundary of a domain. The recovered gradient has comparable superconvergent property on the boundary as that in the interior of the domain. This superconvergence property has been validated by several numerical experiments.



## 1. Introduction

Gradient recovery [1–10] is an effective and widely used post-processing technique in scientific and engineering computation. The main purpose of these techniques is to reconstruct a better numerical gradient from a finite element solution. It can be used for mesh smoothing, a posteriori error estimate [3,7,8,10,5], and adaptive finite element method even with anisotropic meshes [11–14]. More recently, the gradient recovery technique was applied to improve eigenvalue approximation as well [15–18].

The Superconvergent Patch Recovery (SPR) and Polynomial Preserving Recovery (PPR) are two popular methods which have been adopted by commercial software such as ANSYS, Abaqus, COMSOL Multiphysics, Diffpack, LS-DYNA, etc. The SPR is proposed by Zienkiewicz–Zhu in 1992 [9]. It recovers the gradient at a vertex by local least-squares fitting to the finite element gradient in an associated patch. The PPR is proposed by Zhang and Naga in 2005 [6,3]. It recovers the gradient at a vertex by local least-squares fitting to the finite element solution in an associated patch and then taking gradient of the least-squares fitted polynomial.

The PPR often forms a higher-order approximation of the gradient on a patch of mesh elements around each mesh vertex. For regular meshes, the convergence rate of the recovered gradient is $O(h^{p+1})$—the same as for the solution itself [19, p. 471] [20, p. 1061]. However, the accuracy of PPR near boundaries is not as good as that in the interior of the domain. It might even be worse than without recovery [19, p. 471] [20, p. 1061]. Some special treatments are needed to improve the accuracy of PPR on the boundary.

* Correspondence to: School of Mathematics and Computational Science, Sun Yat-sen University, 135, West Xingang Road, Guangzhou, 510275, PR China. Tel.: +86 13533649132.

*E-mail addresses:* hlguo@math.ucsb.edu (H. Guo), zzhang@math.wayne.edu (Z. Zhang), renzhao@wayne.edu (R. Zhao), mcszqs@mail.sysu.edu.cn (Q. Zou).

1 Current Address: Department of Mathematics, University of California, Santa Barbara, CA, 93106, United States.

http://dx.doi.org/10.1016/j.cam.2016.03.003
0377-0427/

In this paper, we present two boundary recovery strategies. Our first strategy to recover the gradient at a boundary vertex is as follows. First, by using the standard PPR local least-squares fitting procedure for interior vertex, we construct a polynomial for each selected interior vertices close to the target boundary vertex. Then we take the average of all quantities evaluating the gradient of the obtained polynomials at the target boundary vertex as the recovered gradient. The second recovery strategy is as below: We construct a relatively large element patch by merging all the element patches of some selected interior vertices near the target point. Then we select all mesh nodes in the above patch as sampling points to fit a polynomial in least-squares sense and define the recovered gradient by the gradient of the constructed polynomial at the target point.

The basic idea behind our two strategies is: the classic PPR method cannot achieve a good approximation on boundary comparable to that in the interior of the domain since the classic selected boundary patch does not contain sufficient information. Therefore, we should replace the boundary patch by the interior patches which has more information than the boundary patch and which has a certain symmetric property. Both the above proposed methods use more information than the classic PPR methods.

Our two methods are numerically tested and compared with standard implementation in COMSOL Multiphysics. The numerical results in $L_2$ norm validate that both our methods lead to superconvergent recovered gradient up to boundary. The numerical errors in $L_\infty$ norm show improved accuracy over the classical PPR method near boundary.

The rest of the paper is organized as follows. In Section 2, we present a terse introduction to polynomial preserving recovery. In Section 3, we introduce two gradient recovery strategies of PPR on boundary and give some illustrative examples. Section 4 contains some numerical examples to verify robustness of our recovery strategies. Finally, conclusions are drawn in Section 5.

## 2. Preliminaries

In this section, we will give a brief introduction to the polynomial preserving recovery method. For the sake of clarity, only $C^0$ finite element methods will be considered.

Let $\Omega$ be a bounded polygonal domain with Lipschitz boundary $\partial\Omega$ in $\mathbb{R}^2$. Throughout this article, the standard notation for Sobolev spaces and their associate norms are adopted as in [21,22]. For a subdomain $\mathcal{A}$ of $\Omega$, let $W_p^k(\mathcal{A})$ denote the Sobolev space with norm $\|\cdot\|_{W_p^k(\mathcal{A})}$ and seminorm $|\cdot|_{W_p^k(\mathcal{A})}$. When $p = 2$, we denote simply $H^k(\mathcal{A}) = W_2^k(\mathcal{A})$ and the subscript $p$ is omitted.

For any $0 < h < \frac{1}{2}$, let $\mathcal{T}_h$ be a shape regular triangulation of $\bar{\Omega}$ with mesh size at most $h$, i.e.

$$\bar{\Omega} = \bigcup_{K\in\mathcal{T}_h} K,$$

where $K$ is a triangle. For any $r \in \mathbb{N}$, define the continuous finite element space $S_h$ of order $r$ as

$$S_h = \{v \in C(\bar{\Omega}) : v|_K \in \mathbb{P}_r(K),\ \forall K \in \mathcal{T}_h\} \subset H^1(\Omega),$$

where $\mathbb{P}_r$ denote the space of polynomials defined on $k$ with degree less than or equal to $r$. Denote the finite element solution in $S_h$ by $u_h$, and the set of mesh nodes and interior mesh nodes by $\mathcal{N}_h$ and $\mathring{\mathcal{N}}_h$, respectively. Given a vertex $z \in \mathcal{N}_h$, let $\mathcal{L}(z, n)$ denote the union of mesh elements in the first $n$ layers around $z$, i.e.,

$$\mathcal{L}(z,n) = \begin{cases} z, & \text{if } n = 0,\\ \bigcup\{\tau : \tau \in \mathcal{T}_h,\ \tau \cap \mathcal{L}(z,0) \neq \phi\}, & \text{if } n = 1,\\ \bigcup\{\tau : \tau \in \mathcal{T}_h,\ \tau \cap \mathcal{L}(z,n-1) \text{ is a } (d-1)\text{-simplex}\}, & \text{if } n \geq 2. \end{cases} \tag{2.1}$$

An element patch $\mathcal{K}_z$ around an interior vertex $z$ is defined based on $\mathcal{L}(z, n)$, which contains $n_z$ nodes. For details on construction of $\mathcal{K}_z$, readers are referred to [6,23]. We select all mesh nodes $z_j \in \mathcal{N}_h,\ j = 1, 2, \ldots, n_z$ in this element patch $\mathcal{K}_z$ as sampling points, and fit a polynomial of degree $r + 1$ in the least squares sense, i.e., we seek for $p_z \in \mathbb{P}_{r+1}(\mathcal{K}_z)$ such that

$$\sum_{j=1}^{n_z}(p_z - u_h)^2(z_j) = \min_{q\in\mathbb{P}_{r+1}} \sum_{j=1}^{n_z}(q - u_h)^2(z_j).$$

The recovered gradient at $z$ is then defined as

$$(G_h u_h)(z) := \nabla p_z(z). \tag{2.2}$$

If $r = 1$, all mesh nodes are vertices and $G_h u_h$ is completely defined. However, $\mathcal{N}_h$ may contain edge nodes or interior nodes for higher order elements. If $z$ is an edge node which lies in an edge between two vertices $z_1$ and $z_2$, we define

$$(G_h u_h)(z) = \beta\nabla p_{z_1}(z) + (1-\beta)\nabla p_{z_2}(z) \tag{2.3}$$

where $\beta$ is determined by the ratio of distances of $z$ to $z_1$ and $z_2$. If $z$ is an interior node which lies in a triangle formed by three vertices $z_1$, $z_2$, and $z_3$, we define

$$(G_h u_h)(z) = \sum_{j=1}^{3} \beta_j \nabla p_{z_j}(z), \tag{2.4}$$

where $\beta_j$ is the barycentric coordinate of $z$. With all nodal values of $G_h u_h$ determined, the gradient recovery operator: $G_h : S_h \to S_h^d$ is then well defined.

It was proved in [6] that the least squares fitting procedure has a unique solution under certain geometric conditions. As for linear element, we need at least six nodes to fit a quadratic polynomial and those sampling points should not be on a conic curve.

In addition, the gradient recovery operator $G_h$ has the following properties [23,6]:

1. $G_h$ is a bounded operator in the sense that there exists a constant $C$, independent of $h$, such that
$$\|G_h v\|_{L_2(\Omega)} \leq C|v|_{H^1(\Omega)}, \quad \forall v \in S_h.$$
2. For any nodal point $z$, if $p \in P_{r+1}(\mathcal{K}_z)$,
$$(G_h p)(z) = \nabla p(z).$$

Furthermore, the following superconvergence results hold [6].

**Theorem 2.1.** *Let $\mathcal{T}_h$ be an arbitrary mesh. Then, $G_h$ preserves polynomials of degrees up to $r+1$ in $\Omega$. Furthermore, if the nodes involved in PPR at a mesh vertex $z \in \mathcal{N}_h$ are symmetrically distributed around $z$, and if $r$ is even, then $G_h$ preserves polynomials of degree up to $r+2$ at $z$.*

**Theorem 2.2.** *Let $z$ be a mesh node and $\mathcal{K}_z$ be the corresponding patch. If $u \in W_\infty^{r+2}(\mathcal{K}_z)$, then*

$$\|\nabla u - G_h u\|_{L_\infty(\mathcal{K}_z)} \leq Ch^{r+1}|u|_{W_\infty^{r+2}(\omega_z)},$$

*where $\omega_z$ is a larger element patch which contains $\mathcal{K}_z$.*

## 3. PPR on boundary

If not handled properly, gradient recovery techniques may deteriorate near boundary [6,23]. High performance near/on boundary is one of the key characteristics of a good gradient recovery technique. In this section, we present two systematic strategies to construct robust PPR operator up to boundary. Both strategies have comparable accuracy near boundary $\partial\Omega$ as in the interior of $\Omega$. Only linear element is considered here. Extension to higher-order elements can be done by combining ideas in this work with PPR for higher-order cases. In the sequel, we denote $z$ as a mesh vertex on boundary, i.e., $z \in \mathcal{N}_h \cap \partial\Omega$.

### 3.1. Strategy 1

Simple averaging of the recovered gradient from PPR under uniform triangular mesh of the regular pattern produces ultra-convergence (two orders higher) gradient recovery for quadratic element at element edge centers [6]. In light of this fact, our first strategy is to treat $z \in \mathcal{N}_h \cap \partial\Omega$ similarly as an edge center in quadratic element.

For any boundary vertex $z$, define

$$\mathcal{K}_z = \mathcal{L}(z, n_0), \tag{3.1}$$

where $n_0$ is the smallest integer such that $\mathcal{L}(z, n_0)$ contains at least one interior vertex.

Let $z_0, z_1, \ldots, z_{n_z}$ be all the interior vertices in $\mathcal{K}_z$. Then our recovered gradient at $z$ is defined as

$$(G_h u_h)(z) = \frac{1}{n_z + 1} \sum_{j=0}^{n_z} \nabla p_{z_j}(z), \tag{3.2}$$

where $p_{z_j}$ is the polynomial that fits $u_h$ at the interior vertex $z_j$ in $\mathcal{K}_{z_j}$, a well defined element patch according to [6].

To describe how to construct $\mathcal{K}_z$, consider a typical Delaunay unstructured mesh on rectangle $[0, 2] \times [0, 1]$ which is obtained using **Triangle** [24], see Fig. 3.1. Boundary vertices can be grouped into those connecting with one interior vertex, two interior vertices, three interior vertices, and so on. It is worth to mention that the first group usually contains only corner vertices. Fig. 3.1 depicts three types of boundary vertices and their corresponding patches.

(1) The left lower corner $z$ is contained in two elements that share the same interior vertex $z_0$. According to definition, $\mathcal{K}_z$ is the element patch which consists of two triangles. We then define $(G_h u_h)(z) = \nabla p_{z_0}(z)$.

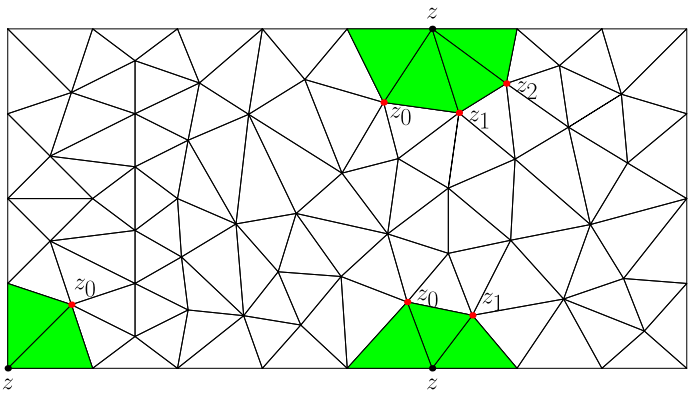


**Fig. 3.1.** Examples for patch used in Strategy 1.

(2) The bottom $z$ is contained in three elements that have two interior vertices $z_0$ and $z_1$. According to definition $\mathcal{K}_z$ is the element patch which consists of three triangles. We then define $(G_h u_h)(z) = \frac{1}{2}(\nabla p_{z_0}(z) + \nabla p_{z_1}(z))$.
(3) The upper $z$ is contained in four elements that have three interior vertices $z_0$, $z_1$, and $z_2$. According to definition, $\mathcal{K}_z$ is the element patch which consists of four triangles. The recovered gradient at $z$ is then defined as $(G_h u_h)(z) = \frac{1}{3}(\nabla p_{z_0}(z) + \nabla p_{z_1}(z) + \nabla p_{z_2}(z))$.

### 3.2. Strategy 2

Here we treat $z$ just like an interior vertex. However, the definition of $\mathcal{K}_z$ is more delicate and deserves special consideration. $\mathcal{K}_z$ is constructed in two steps. In the first step, we define a temporary patch $\tilde{\mathcal{K}}_z$ as $\mathcal{K}_z$ in (3.1). After constructing the temporary patch $\tilde{\mathcal{K}}_z$, we define

$$\mathcal{K}_z = \left( \bigcup_{\tilde{z} \in \tilde{\mathcal{K}}_z \cap \mathring{\mathcal{N}}_h} \mathcal{K}_{\tilde{z}} \right) \bigcup \left( \bigcup_{\tilde{z} \in \tilde{\mathcal{K}}_z \cap \mathcal{N}_h \cap \partial\Omega} \mathcal{L}(\tilde{z}, 1) \right), \tag{3.3}$$

where $\mathcal{K}_{\tilde{z}}$ is defined in Ref. [6] for $\tilde{z} \in \mathring{\mathcal{N}}_h$. Note that we distinguish between interior vertices and boundary vertices in the temporary patch $\tilde{\mathcal{K}}_z$. For a boundary vertex $z'$, only triangles having $z'$ as a vertex is added to $\mathcal{K}_z$; but for an interior vertex $z''$, its own patch $\mathcal{K}_{z''}$ is adding to $\mathcal{K}_z$. Let $p_z \in \mathbb{P}_2(\mathcal{K}_z)$ be the polynomial that best fits $u_h$ at the mesh nodes in $K_z$ in discrete least squares sense, i.e.,

$$p_z = \arg \min_{p \in \mathbb{P}_2(\mathcal{K}_z)} \sum_{\tilde{z} \in \mathcal{N}_h \cap \mathcal{K}_z} |(u_h - p)(\tilde{z})|^2. \tag{3.4}$$

Then define the gradient recovery operator at vertex $z$ as

$$(G_h u_h)(z) = \nabla p_z(z).$$

To demonstrate the process of constructing $\mathcal{K}_z$ in Strategy 2, we use the same Delaunay mesh as in Strategy 1. All three types of boundary vertices are described in previous subsection. Note that we construct $\mathcal{K}_z$ in two steps. Firstly, we construct $\tilde{\mathcal{K}}_z$ which is shown in Fig. 3.1. Then $\mathcal{K}_z$ can be constructed which is illustrated in Fig. 3.2. Take the boundary vertex connecting with 3 interior vertices as an example; see the solid dot point on the top edge in Fig. 3.2. In the first step, we construct $\tilde{\mathcal{K}}_z$ which consists of four triangles having $z$ as a vertex; see Fig. 3.1 for details. $z_0, z_1$ and $z_2$ are all interior vertices in $\tilde{\mathcal{K}}_z$. According to (3.3), $\mathcal{K}_z$ contains $\mathcal{K}_{z_0}$, $\mathcal{K}_{z_1}$ and $\mathcal{K}_{z_2}$. The union of $\mathcal{K}_{z_0}$, $\mathcal{K}_{z_1}$ and $\mathcal{K}_{z_2}$ is all green triangles near the top edge in Fig. 3.2. For other boundary vertices in $\tilde{\mathcal{K}}_z$, we only add triangles containing them into $\mathcal{K}_z$, i.e. the two red triangles near the top edge. Thus $\mathcal{K}_z$ is the element patch consisting of sixteen triangles.

**Remark 3.1.** Definition of $\mathcal{K}_z$ in (3.3) always guarantees the existence and uniqueness of $p_z$. The construction procedure is systematic and works for arbitrary mesh.

**Remark 3.2.** Comparing with the boundary recovery methods proposed in [6], the points involved in our procedure are more symmetric. Hence, this strategy is more stable and robust.

Before ending this subsection, we consider a special situation. For mesh generated by engineering procedure such as Delaunay mesh generator, any vertex connects with at least one interior vertex, i.e. $\mathcal{L}(z, 1) \cap \mathring{\mathcal{N}}_h \neq \emptyset$; see Figs. 3.1 or 3.2. But it may occur that $\mathcal{L}(z, 1) \cap \mathring{\mathcal{N}}_h = \emptyset$, such as regular and chevron pattern of uniform mesh. Even in this case, both our

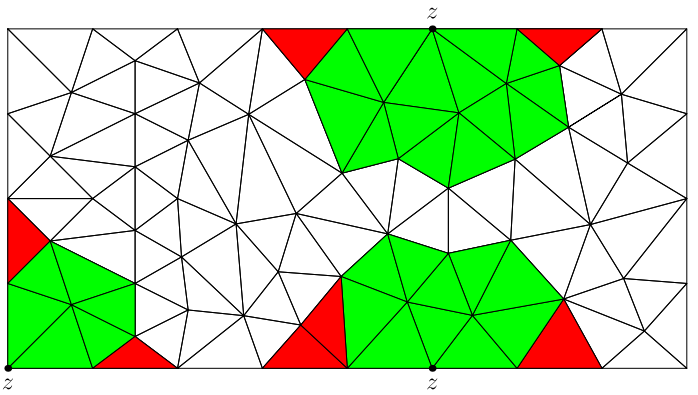


**Fig. 3.2.** Examples for patch used in Strategy 2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

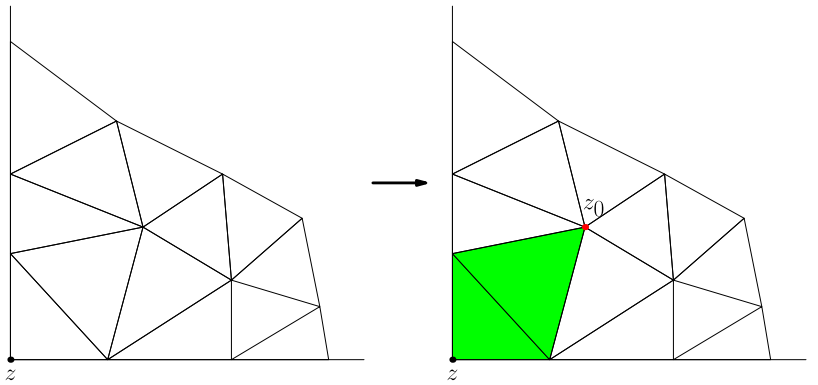


**Fig. 3.3.** Patch of isolated corned vertex in Strategy 1. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

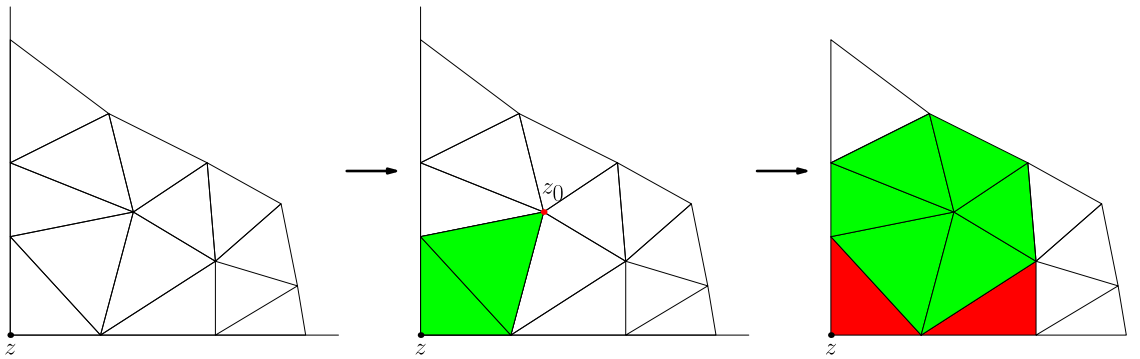


**Fig. 3.4.** Patch of isolated corned vertex in Strategy 2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

strategies can be applied without any change. One typical example is shown in Figs. 3.3 or 3.4. For strategy 1, $\mathcal{K}_z$ should be defined as $\mathcal{L}(z, 2)$ instead of $\mathcal{L}(z, 1)$. In other words, $\mathcal{K}_z$ are two green triangles in Fig. 3.3. Then the recovered gradient at $z$ is defined as $(G_h u_h)(z) = \nabla p_{z_0}(z)$. In order to define $\mathcal{K}_z$ in strategy 2, we first construct $\tilde{\mathcal{K}}_z$ containing one interior vertex $z_0$; see the second sub-figure of Fig. 3.4. According to (3.3), $\mathcal{K}_z$ contains $K_{z_0}$, i.e. all green triangles in the third sub-figure of Fig. 3.4. Similarly, all triangles containing $z'$ are added to $\mathcal{K}_z$ for each boundary vertex $z'$ in $\tilde{\mathcal{K}}_z$.

### 3.3. Some illustrations

In this subsection, we use three examples of uniform mesh to demonstrate superconvergence and robustness of our two gradient recovery strategies on boundary. Let $G_h^1$ and $G_h^2$ denote boundary recovery operator defined by Strategy 1 and Strategy 2, respectively.

**Example 1.** We consider a typical corner vertex in regular pattern, see the solid dot point in Fig. 3.6. In this case, the corner point belongs to only one element, to which there is no interior vertex attached. According to strategy 1, we fit a quadratic polynomial $p_{\tilde{z}}(x, y)$ at $\tilde{z}$ instead of fitting a quadratic polynomial of $p_z(x, y)$ at $z$, where $\tilde{z}$ is the closest interior vertex to $z$, i.e. the solid dot point in Fig. 3.5. Note that Fig. 3.5 shows the patch of the interior vertex $\tilde{z}$ instead of $z$. Applying the least squares fitting procedure described in [6], we obtain

$$\begin{aligned} p_{\tilde{z}}(x, y) &= u_0 + \frac{1}{6h}(2u_1 + u_2 - u_3 - 2u_4 - u_5 + u_6)x + \frac{1}{6h}(-u_1 + u_2 + 2u_3 + u_4 - u_5 - 2u_6)y \\ &\quad + \frac{1}{6h^2}(-6u_0 + 3u_1 + 3u_4)x^2 + \frac{1}{6h^2}(-6u_0 + 3u_3 + 3u_6)y^2 \\ &\quad + \frac{1}{6h^2}(6u_0 - 3u_1 + 3u_2 - 3u_3 - 3u_4 + 3u_5 - 3u_6)xy. \end{aligned}$$

Differentiating with respect to $x$ and $y$, we get

$$\begin{aligned} \frac{\partial p_{\tilde{z}}}{\partial x} &= \frac{1}{6h}(2u_1 + u_2 - u_3 - 2u_4 - u_5 + u_6) + \frac{1}{3h^2}(-6u_0 + 3u_1 + 3u_4)x \\ &\quad + \frac{1}{6h^2}(6u_0 - 3u_1 + 3u_2 - 3u_3 - 3u_4 + 3u_5 - 3u_6)y; \\ \frac{\partial p_{\tilde{z}}}{\partial y} &= \frac{1}{6h}(-u_1 + u_2 + 2u_3 + u_4 - u_5 - 2u_6) + \frac{1}{3h^2}(-6u_0 + 3u_3 + 3u_6)y \\ &\quad + \frac{1}{6h^2}(6u_0 - 3u_1 + 3u_2 - 3u_3 - 3u_4 + 3u_5 - 3u_6)x. \end{aligned}$$

Evaluating $\frac{\partial p_{\tilde{z}}}{\partial x}$ and $\frac{\partial p_{\tilde{z}}}{\partial y}$ at $z$ yields

$$G_h^1 u(z) = \frac{1}{6h}\begin{pmatrix} -18u_0 + 11u_1 - 2u_2 + 2u_3 + 7u_4 - 4u_5 + 4u_6 \\ 18u_0 - 4u_1 + 4u_2 - 7u_3 - 2u_4 + 2u_5 - 11u_6 \end{pmatrix}, \tag{3.5}$$

as depicted in Fig. 3.5. Using a computer algebra system such as **Mathematica**, we have the following Taylor expansion

$$G_h^1 u(z) = \begin{pmatrix} u_x(z) - \frac{h^2}{6}(2u_{xxx}(z) - 7u_{xxy}(z) + 2u_{xyy}(z)) + O(h^3) \\ u_y(z) - \frac{h^2}{6}(2u_{xxy}(z) - 7u_{xyy}(z) + 2u_{yyy}(z)) + O(h^3) \end{pmatrix}, \tag{3.6}$$

which is a second order finite difference scheme approximating $\nabla u(z)$.

Now we turn to Strategy 2. It fits a quadratic polynomial

$$\hat{p}_z(\xi, \eta) = (1, \xi, \eta, \xi^2, \xi\eta, \eta^2)(\hat{a}_1, \ldots, \hat{a}_6)^T,$$

in the least-squares sense at $z$, see the solid dot point in Fig. 3.6, with respect to eight nodal values in $(\xi, \eta)$ coordinates

$$\vec{\xi} = (0, 0, 0, -1, -2, -1, -1, -2)^T, \qquad \vec{\eta} = (0, 1, 2, 1, 0, 0, 2, 1).$$

We obtain

$$\begin{aligned} p_z(x, y) &= \frac{1}{42}(38u_0 + 6u_1 - 2u_2 - 8u_3 - 2u_4 + 6u_5 + 2u_6 + 2u_7)\frac{1}{42h}(44u_0 + 11u_1 + 8u_2 - 38u_3 - 6u_4 \\ &\quad - 38u_5 - 8u_6 + 27u_7)x\frac{1}{42h}(-44u_0 + 38u_1 + 6u_2 + 38u_3 - 8u_4 - 11u_5 - 27u_6 + 8u_7)y \\ &\quad \times \frac{1}{42h^2}(12u_0 + 3u_1 + 6u_2 - 18u_3 + 6u_4 - 18u_5 - 6u_6 + 15u_7)x^2\frac{1}{42h^2}(-18u_0 + 6u_1 + 12u_2 \\ &\quad + 6u_3 + 12u_4 + 6u_5 - 12u_6 - 12u_7)xy\frac{1}{42h^2}(12u_0 - 18u_1 + 6u_2 - 18u_3 + 6u_4 + 3u_5 + 15u_6 - 6u_7)y^2. \end{aligned}$$

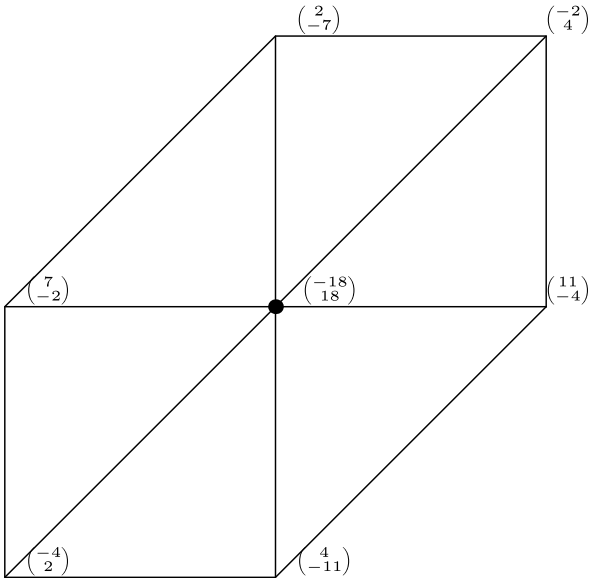


**Fig. 3.5.** Denominator $42h$.

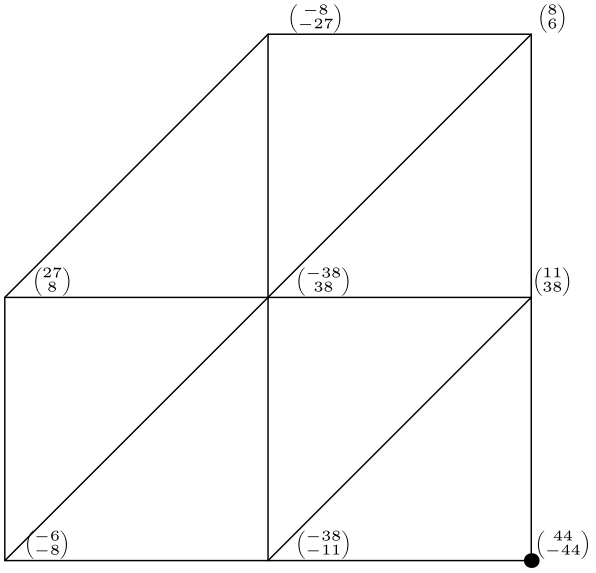


**Fig. 3.6.** Denominator $6h$.

It indicates that

$$\begin{aligned}\frac{\partial p_z}{\partial x} &= \frac{1}{42h}(44u_0+11u_1+8u_2-38u_3-6u_4-38u_5-8u_6+27u_7)\frac{1}{21h^2}(12u0+3u_1+6u_2-18u_3\\ &\quad +6u_4-18u_5-6u_6+15u_7)x\frac{1}{42h^2}(-18u_0+6u_1+12u_2+6u_3+12u_4+6u_5-12u_6-12u_7)y;\\ \frac{\partial p_z}{\partial y} &= \frac{1}{42h}(-44u_0+38u_1+6u_2+38u_3-8u_4-11u_5-27u_6+8u_7)\frac{1}{42h^2}(-18u_0+6u_1+12u_2+6u_3\\ &\quad +12u_4+6u_5-12u_6-12u_7)x\frac{1}{21h^2}(12u_0-18u_1+6u_2-18u_3+6u_4+3u_5+15u_6-6u_7)y.\end{aligned}$$

Then we obtain the recovered gradient at boundary vertex $z$ (see Fig. 3.6)

$$G_h^2u(z)=\frac{1}{42h}\begin{pmatrix}44u_0+11u_1+8u_2-38u_3-6u_4-38u_5-8u_6+27u_7\\ -44u_0+38u_1+6u_2+38u_3-8u_4-11u_5-27u_6+8u_7\end{pmatrix}. \tag{3.7}$$

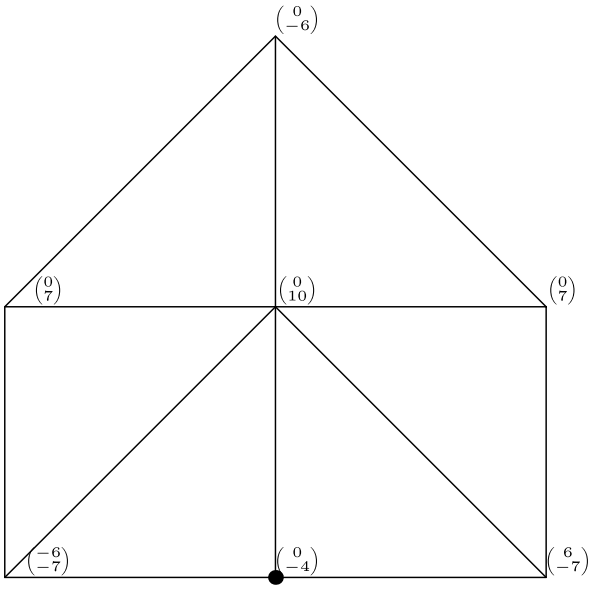


**Fig. 3.7.** Denominator $12h$.

Also, we have the following Taylor expansion

$$G_h^2 u(z) = \begin{pmatrix} u_x(z) - \dfrac{h^2}{42}(14u_{xxx}(z) - 27u_{xxy}(z) - 8u_{xyy}(z)) + O(h^3) \\ u_y(z) + \dfrac{h^2}{42}(8u_{xxy}(z) + 27u_{xyy}(z) - 14u_{yyy}(z)) + O(h^3) \end{pmatrix}; \tag{3.8}$$

which is a second-order finite difference scheme as well.

**Remark 3.3.** The main difference between Strategy 1 and Strategy 2 is that the former fits quadratic polynomials at some interior vertices near $z$ but the latter fits a quadratic polynomial at the very boundary vertex $z$.

**Example 2.** In this example, a typical boundary vertex, as plotted in Fig. 3.7, in chevron pattern mesh is considered. Firstly, we employ Strategy 1 to this case. Repeating the same procedure as in Example 1, we find that

$$G_h^1 u(z) = \frac{1}{12h}\begin{pmatrix} -6u_4 + 6u_6 \\ 10u_0 + 7u_1 - 6u_2 + 7u_3 - 7u_4 - 4u_5 - 7u_6 \end{pmatrix} \tag{3.9}$$

as shown in Fig. 3.7. It is straightforward to verify that

$$G_h^1 u(z) = \begin{pmatrix} u_x(z) - \dfrac{h^2}{6}u_{xxx}(z) + O(h^3) \\ u_y(z) + \dfrac{h^2}{12}(7u_{xxy}(z) - 4u_{yyy}(z)) + O(h^3) \end{pmatrix}; \tag{3.10}$$

which provides a second-order approximation to the exact gradient $\nabla u$.

Then we consider Strategy 2. The patch $\mathcal{K}_z$ of $z$ is shown in Fig. 3.8. Following the same procedure as Example 1, we derive that

$$G_h^{2x} u(z) = \frac{1}{140h}\left(-28u_5 - 14u_6 + 14u_8 + 28u_9\right),$$

and

$$G_h^{2y} u(z) = \frac{1}{140h}\left(66u_0 + 61u_1 - 70u_2 + 61u_3 + 46u_4 - 52u_5 - 37u_6 - 37u_7 - 37u_8 - 52u_9 + 46u_{10}\right);$$

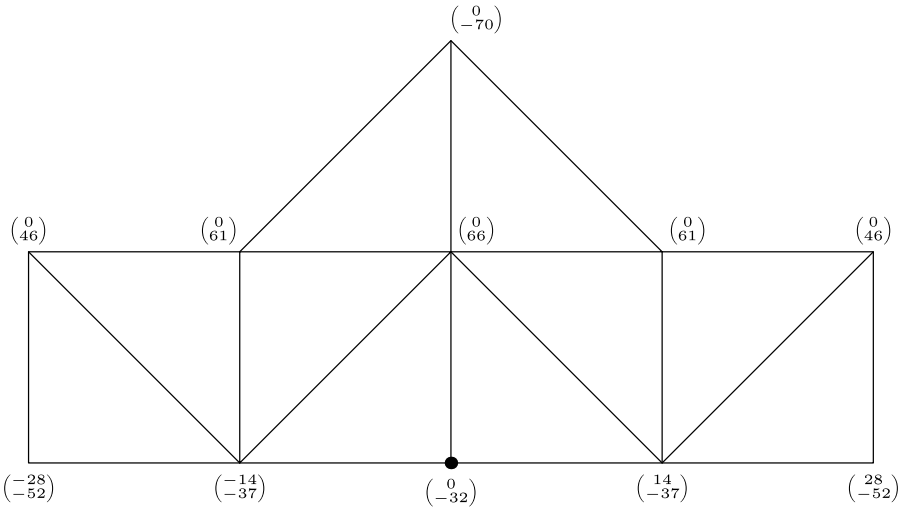


**Fig. 3.8.** Denominator 140$h$.

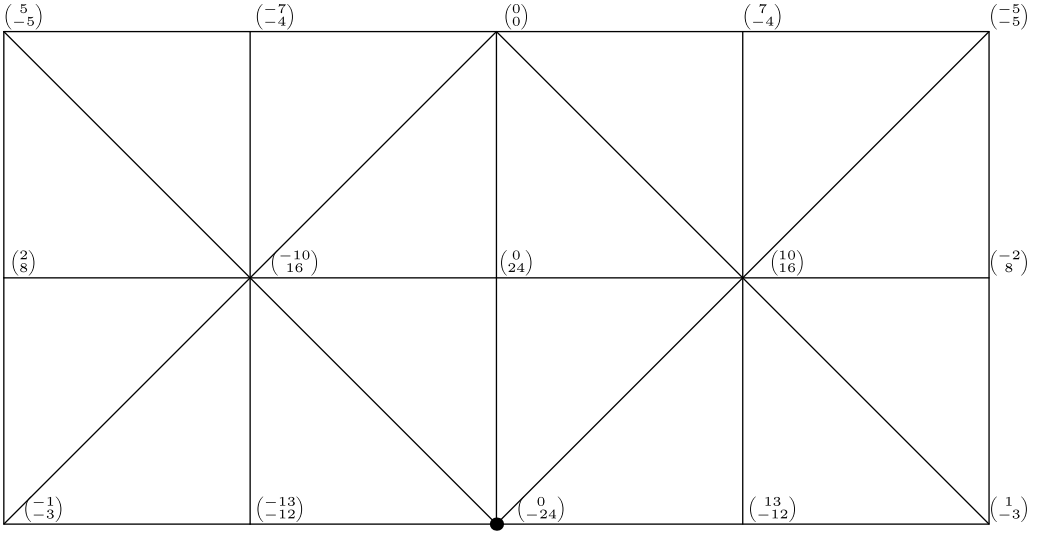


**Fig. 3.9.** Denominator 36$h$.

where $G_h^{2x}$ and $G_h^{2y}$ represent the first and second rows of $G_h^2$ respectively. Note that Strategy 2 uses larger patch, see Fig. 3.8, but it also produces a second-order finite difference scheme. Actually, we have

$$G_h^2 u(z) = \begin{pmatrix} u_x(z) - \dfrac{17h^2}{30} u_{xxx}(z) + O(h^3) \\ u_y(z) + \dfrac{h^2}{12}(21u_{xxy}(z) - 4u_{yyy}(z)) + O(h^3) \end{pmatrix}. \tag{3.11}$$

**Example 3.** This example demonstrates that $G_h^1$ and $G_h^2$ may involve the same vertices but produce different finite difference schemes. Let $z$ be a boundary vertex as plotted in Fig. 3.9. As for Strategy 1, we need to fit three least square polynomials at three interior vertices $z_0$, $z_1$ and $z_2$ connecting $z$ and then take average. Simple calculation verifies that

$$G_h^{1x} u(z) = \frac{1}{36h}\left(-10u_0 - 7u_3 + 5u_4 + 2u_5 - u_6 - 13u_7 + 10u_9 - 2u_{10} - 5u_{11} + 7u_{12} + 13u_{13} + u_{14}\right),$$

and

$$\begin{aligned} G_h^{1y} u(z) = \frac{1}{36h}\big(&-10u_0 + 24u_1 - 4u_3 - 5u_4 + 8u_5 - 3u_6 - 12u_7 \\ &- 24u_8 + 16u_9 + 8u_{10} - 5u_{11} - 4u_{12} - 12u_{13} - 3u_{14}\big), \end{aligned}$$

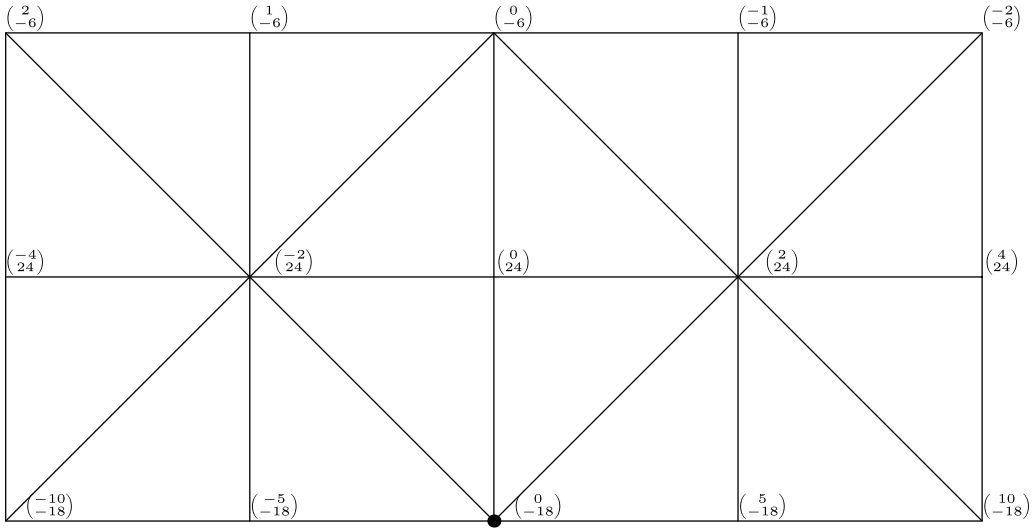


**Fig. 3.10.** Denominator $60h$.

where $G_h^{1x}$ and $G_h^{1y}$ are two rows of $G_h^1$. Using standard Taylor expansion, we get

$$G_h^1 u(z) = \begin{pmatrix} u_x(z) - \dfrac{h^2}{6}(u_{xxx}(z) + u_{xyy}(z)) + O(h^3) \\ u_y(z) - \dfrac{h^2}{3}u_{yyy}(z) + O(h^3) \end{pmatrix} \tag{3.12}$$

which clearly indicates that $G_h^1$ provides a second order approximation to the exact gradient $\nabla u(z)$.

To see how Strategy 2 works, we construct patch $\mathcal{K}_z$, as shown in Fig. 3.10, in two steps. Using all vertices in $\mathcal{K}_z$, fit a quadratic polynomial at $z$ which yields

$$G_h^{2x} u(z) = \frac{1}{60h}\left(2u_1 + 4u_2 - 2u_3 - u_4 + u_6 + 2u_7 - 4u_8 - 2u_9 - 10u_{10} - 5u_{11} + 5u_{13} + 10u_{14}\right),$$

and

$$G_h^{2y} u(z) = \frac{1}{10h}\left(4u_0 + 4u_1 + 4u_2 - u_3 - u_4 - u_5 - u_6 - u_7 + 4u_8 + 4u_9 - 3u_{10} - 3u_{11} - 3u_{12} - 3u_{13} - 3u_{14}\right),$$

where $G_h^{2x}$ and $G_h^{2y}$ have the same meaning as previous example. Taylor expansion results in

$$G_h^2 u(z) = \begin{pmatrix} u_x(z) + \dfrac{h^2}{30}(17u_{xxx}(z) - 5u_{xyy}(z)) + O(h^3) \\ u_y(z) + \dfrac{h^2}{3}(3u_{xxy}(z) - u_{yyy}(z)) + O(h^3) \end{pmatrix}. \tag{3.13}$$

This means that $G_h^2 u(z)$ is also a second order approximation of the exact gradient $\nabla u(z)$.

**Remark 3.4.** Comparing the computational complexity of Strategy 1 and Strategy 2, we see that Strategy 1 needs to perform three least square fittings with three $9 \times 6$ matrices. On the other hand, Strategy 2 does one least square fittings with one $15 \times 6$ matrix. Thus the computational costs of those two strategies are comparable.

**Remark 3.5.** We have discussed three cases to illustrate proposed two strategies for PPR on boundary. Indeed, both $G_h^1 u$ and $G_h^2 u$ converge to $\nabla u$ with second-order rate for all boundary vertices of arbitrary mesh due to the polynomial preserving property.

## 4. Numerical experiments

In this section, we provide four numerical examples to verify superconvergence and robustness of our boundary recovery strategies and also compare the results with COMSOL Multiphysics integrated '*ppr*' command. In order to detect boundary

**Table 4.1**
Example 1: $\|\cdot\|_{L_\infty(\mathcal{N}_{h,1})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 2.37e−01 | – | 1.54e−02 | – | 1.54e−02 | – | 2.58e−02 | – |
| 4 841 | 1.27e−01 | 0.46 | 5.48e−03 | 0.76 | 5.48e−03 | 0.76 | 1.66e−02 | 0.32 |
| 19 121 | 6.55e−02 | 0.48 | 2.34e−03 | 0.62 | 2.34e−03 | 0.62 | 8.02e−03 | 0.52 |
| 76 001 | 3.33e−02 | 0.49 | 1.11e−03 | 0.54 | 1.11e−03 | 0.54 | 3.73e−03 | 0.56 |

**Table 4.2**
Example 1: $\|\cdot\|_{L_\infty(\mathcal{N}_{h,2})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 2.87e−01 | – | 8.08e−03 | – | 2.83e−02 | – | 2.76e−02 | – |
| 4 841 | 1.45e−01 | 0.50 | 2.36e−03 | 0.90 | 7.08e−03 | 1.02 | 7.03e−03 | 1.00 |
| 19 121 | 7.25e−02 | 0.50 | 9.85e−04 | 0.64 | 1.77e−03 | 1.01 | 2.23e−03 | 0.83 |
| 76 001 | 3.63e−02 | 0.50 | 4.48e−04 | 0.57 | 4.48e−04 | 0.99 | 8.08e−04 | 0.74 |

influence, we split mesh nodes $\mathcal{N}_h$ into $\mathcal{N}_{h,1}$ and $\mathcal{N}_{h,2}$, where $\mathcal{N}_{h,2} = \{z \in \mathcal{N}_h : \text{dist}(z, \partial\Omega) \le L\}$ denotes all near boundary nodes and $\mathcal{N}_{h,1} = \mathcal{N}_h \setminus \mathcal{N}_{h,2}$ denotes rest interior nodes. Now, we can define

$$\Omega_{h,1} = \bigcup\{\tau \in \mathcal{T}_h : \tau \text{ has all of its vertices in } \mathcal{N}_{h,1}\},$$

and $\Omega_{h,2} = \Omega \setminus \Omega_{h,1}$, where $L$ is some small quantity to indicate the *width* of the boundary.

The notations used are the following:

$De = \nabla(u - u_h)$, where $u_h$ is the finite element solution.

$De^1 = \nabla u - G_h^1 u_h$, where $G_h^1 u_h$ is defined by PPR using Strategy 1.

$De^2 = \nabla u - G_h^2 u_h$, where $G_h^2 u_h$ is defined by PPR using Strategy 2.

$De^3 = \nabla u - G_h^3 u_h$, where $G_h^3 u_h$ is defined by PPR using COMSOL Multiphysics integrated '*ppr*' command.

All computations are carried out in COMSOL Multiphysics 3.5a on Delaunay triangulation. We perform three levels mesh refinement by connecting midpoints of each triangles. Here we choose $L = 0.1$.

**Example 1.** We first consider a symmetric and infinitely smooth case:

$$-\Delta u = 2\pi^2 \sin \pi x \sin \pi y, \quad \text{in } \Omega = [0, 1]^2,$$

with $u = 0$ on $\partial\Omega$. The exact solution is

$$u(x, y) = \sin \pi x \sin \pi y.$$

The maximum errors of $\nabla u - G_h u_h$ for interior nodes and near boundary nodes are depicted in Tables 4.1 and 4.2, respectively. It can be observed that after performing PPR by any of the three methods, the maximum error decreases significantly comparing to that without performing gradient recovery processing. In Table 4.1, the $L^\infty$ norm of $De^1$ and $De^2$ are identical since they have the same strategy for the interior nodes and only differ on the boundary. It is worth pointing out that to achieve the same accuracy, PPR 1 or PPR 2 requires approximately only $\frac{1}{4}$ degrees of freedom (DOF) of COMSOL Multiphysics integrated 'ppr' command.

In Table 4.2, we observe clearly superconvergence phenomena. Before recovery, $De$ shows a convergence rate $O(N^{-\frac{1}{2}})$. After PPR, our second strategy converges at a rate of $O(N^{-1})$. Moreover, to achieve the same level of accuracy, PPR 1 requires approximately $\frac{1}{4}$ degrees of freedom of COMSOL Multiphysics.

In addition, we report the $L_2$ error in Tables 4.3 and 4.4. As expected, it is observed that $\nabla(u - u_h)$ is $O(N^{-\frac{1}{2}})$. Concerning the convergence of recovered gradients, all three strategies show superconvergence at a rate of $O(N^{-1})$ in the interior domain and near the boundary region.

**Example 2.** Our second example is:

$$-\Delta u = 1, \quad \text{in } \Omega = [0, 1]^2,$$

with $u = 0$ on $\partial\Omega$. The exact solution is given by the infinite series

$$u(x, y) = \frac{x(1-x) + y(1-y)}{4} - \frac{2}{\pi^3} \sum_{m=0}^{\infty} \frac{1}{(2m+1)^3 (1 + e^{-(2m+1)\pi})} \cdot \{[e^{-(2m+1)\pi y} + e^{-(2m+1)\pi(1-y)}] \sin((2m+1)\pi x) + [e^{-(2m+1)\pi x} + e^{-(2m+1)\pi(1-x)}] \sin((2m+1)\pi y)\}.$$

This problem has weak singularities at four corners. In order to observe the asymptotic behavior of numerical approximations, we start from the second mesh level in the previous example and perform one more level mesh refinement.

**Table 4.3**
Example 1: $\|\cdot\|_{L_2(\Omega_{h,1})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 5.70e−02 | – | 6.92e−03 | – | 6.92e−03 | – | 5.57e−03 | – |
| 4 841 | 2.86e−02 | 0.505 | 1.80e−03 | 0.98 | 1.82e−03 | 0.98 | 1.63e−03 | 0.90 |
| 19 121 | 1.45e−02 | 0.497 | 4.83e−04 | 0.98 | 4.83e−04 | 0.97 | 4.10e−04 | 0.99 |
| 76 001 | 7.27e−03 | 0.500 | 1.26e−04 | 0.97 | 1.26e−04 | 0.97 | 1.09e−04 | 0.97 |

**Table 4.4**
Example 1: $\|\cdot\|_{L_2(\Omega_{h,2})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 5.17e−02 | – | 4.98e−03 | – | 7.07e−03 | – | 4.04e−03 | – |
| 4 841 | 2.58e−02 | 0.51 | 1.32e−03 | 0.97 | 1.59e−03 | 1.09 | 1.01e−03 | 1.01 |
| 19 121 | 1.27e−02 | 0.51 | 3.34e−04 | 1.00 | 3.69e−04 | 1.07 | 2.53e−04 | 1.01 |
| 76 001 | 6.33e−03 | 0.51 | 8.48e−05 | 0.99 | 8.92e−05 | 1.03 | 6.26e−05 | 1.01 |

**Table 4.5**
Example 2: $\|\cdot\|_{L_\infty(\mathcal{N}_{h,1})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 4 841 | 1.16e−02 | – | 3.34e−04 | – | 3.33e−04 | – | 1.27e−03 | – |
| 19 121 | 5.99e−03 | 0.48 | 1.58e−04 | 0.55 | 1.58e−04 | 0.55 | 5.41e−04 | 0.62 |
| 76 001 | 2.98e−03 | 0.51 | 8.03e−05 | 0.49 | 8.03e−05 | 0.49 | 2.57e−04 | 0.54 |
| 303 041 | 1.49e−03 | 0.50 | 4.04e−05 | 0.50 | 4.05e−05 | 0.50 | 1.30e−04 | 0.50 |

**Table 4.6**
Example 2: $\|\cdot\|_{L_\infty(\mathcal{N}_{h,2})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 4 841 | 3.07e−02 | – | 3.42e−03 | – | 8.51e−03 | – | 6.99e−03 | – |
| 19 121 | 1.77e−02 | 0.40 | 1.54e−03 | 0.58 | 4.22e−03 | 0.51 | 3.45e−03 | 0.51 |
| 76 001 | 9.95e−03 | 0.42 | 7.73e−04 | 0.50 | 2.18e−03 | 0.48 | 1.80e−03 | 0.47 |
| 303 041 | 5.44e−03 | 0.44 | 3.41e−04 | 0.59 | 1.04e−03 | 0.53 | 8.53e−04 | 0.54 |

**Table 4.7**
Example 2: $\|\cdot\|_{L_2(\Omega_{h,1})}$ on Delaunay triangulation.

| DOF | $De$ | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 4 841 | 2.24e−03 | – | 6.21e−05 | – | 6.21e−05 | – | 1.26e−04 | – |
| 19 121 | 1.14e−03 | 0.49 | 1.91e−05 | 0.86 | 1.91e−05 | 0.86 | 3.33e−05 | 0.97 |
| 76 001 | 5.75e−04 | 0.50 | 5.54e−06 | 0.90 | 5.54e−06 | 0.90 | 8.76e−06 | 0.97 |
| 303 041 | 2.89e−04 | 0.50 | 1.56e−06 | 0.92 | 1.56e−06 | 0.92 | 2.31e−06 | 0.96 |

The maximum error of gradient and convergence rates are reported in Tables 4.5 and 4.6. Due to the corner singularities, the maximum error occurs near the boundary and it is observed in Table 4.6. It can be seen that all strategies have enhanced the maximum error of gradient as expected. In Table 4.5, we can also observe that $De^1$ and $De^2$ on level 2 are comparable to $De^3$ on level 4. In Table 4.6, $De^1$ in level 3 is even smaller than $De^2$ and $De^3$ on level 4.

The $L_2$ errors are displayed in Tables 4.7 and 4.8. Inside the domain, Strategy 1 and Strategy 2 superconverge at rate $\approx O(N^{-0.9})$ while COMSOL Multiphysics integrated '*ppr*' command superconverges at rate $\approx O(N^{-1})$. However, we can observe smaller errors in both of our strategies than in COMSOL Multiphysics. Concerning the performing PPR near boundary, all three strategies are comparable and superconvergent.

**Example 3.** We now consider an anisotropic diffusion problem defined in the unit square $\Omega = (0, 1)^2$ as follows

$$\begin{cases} -\nabla\cdot(\mathcal{A}\nabla u) = f, & \text{in } \Omega \\ u = 0, & \text{on } \partial\Omega \end{cases}$$

where the diffusion matrix is given by

$$\mathcal{A} = \begin{pmatrix} k^2 & 0 \\ 0 & 1 \end{pmatrix},$$

and $f(x)$ is chosen such that the exact solution is $u = \sin(\pi x)\sin(k\pi y)$. We test the case $k = 10$. For anisotropic problems, it is more suitable to use anisotropic meshes or adaptive meshes. Nevertheless, for the sake of identifying the performance

**Table 4.8**
Example 2: $\|\cdot\|_{L_2(\Omega_{h,2})}$ on Delaunay triangulation.

| DOF | *De* | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 4 841 | 3.09e−03 | – | 1.22e−04 | – | 3.11e−04 | – | 2.47e−04 | – |
| 19 121 | 1.53e−03 | 0.51 | 3.33e−05 | 0.94 | 7.94e−05 | 0.99 | 6.71e−05 | 0.95 |
| 76 001 | 7.63e−04 | 0.50 | 9.04e−06 | 0.94 | 2.02e−05 | 0.99 | 1.75e−05 | 0.98 |
| 303 041 | 3.80e−04 | 0.50 | 2.44e−06 | 0.95 | 5.15e−06 | 0.99 | 4.59e−06 | 0.97 |

**Table 4.9**
Example 3: $\|\cdot\|_{L_\infty(\mathcal{N}_{h,1})}$ on Delaunay triangulation.

| DOF | *De* | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 1.82e+01 | – | 1.63e+01 | – | 1.63e+01 | – | 1.59e+01 | – |
| 4 841 | 1.08e+01 | 0.38 | 6.89e+00 | 0.63 | 6.89e+00 | 0.63 | 6.89e+00 | 0.62 |
| 19 121 | 5.10e+00 | 0.55 | 2.35e+00 | 0.78 | 2.35e+00 | 0.78 | 2.39e+00 | 0.77 |
| 76 001 | 2.44e+00 | 0.53 | 8.21e−01 | 0.76 | 8.21e−01 | 0.76 | 1.05e+00 | 0.60 |

**Table 4.10**
Example 3: $\|\cdot\|_{L_\infty(\mathcal{N}_{h,2})}$ on Delaunay triangulation.

| DOF | *De* | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 1.27e+01 | – | 4.84e+00 | – | 4.56e+00 | – | 5.14e+00 | – |
| 4 841 | 6.63e+00 | 0.48 | 1.92e+00 | 0.68 | 1.92e+00 | 0.63 | 1.92e+00 | 0.72 |
| 19 121 | 3.35e+00 | 0.50 | 6.02e−01 | 0.84 | 6.03e−01 | 0.84 | 6.99e−01 | 0.74 |
| 76 001 | 1.68e+00 | 0.50 | 2.38e−01 | 0.67 | 2.38e−01 | 0.67 | 4.55e−01 | 0.31 |

**Table 4.11**
Example 3: $\|\cdot\|_{L_2(\Omega_{h,1})}$ on Delaunay triangulation.

| DOF | *De* | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 5.16e+00 | – | 5.38e+00 | – | 5.38e+00 | – | 4.38e+00 | – |
| 4 841 | 2.29e+00 | 0.60 | 1.96e+00 | 0.74 | 1.96e+00 | 0.74 | 1.69e+00 | 0.70 |
| 19 121 | 9.79e−01 | 0.62 | 5.81e−01 | 0.89 | 5.81e−01 | 0.89 | 4.83e−01 | 0.91 |
| 76 001 | 4.50e−01 | 0.56 | 1.58e−01 | 0.94 | 1.58e−01 | 0.94 | 1.40e−01 | 0.90 |

**Table 4.12**
Example 3: $\|\cdot\|_{L_2(\Omega_{h,2})}$ on Delaunay triangulation.

| DOF | *De* | Order | $De^1$ | Order | $De^2$ | Order | $De^3$ | Order |
|---|---|---|---|---|---|---|---|---|
| 1 241 | 1.94e+00 | – | 1.55e+00 | – | 1.58e+00 | – | 1.13e+00 | – |
| 4 841 | 9.54e−01 | 0.52 | 4.94e−01 | 0.84 | 5.00e−01 | 0.84 | 3.92e−01 | 0.78 |
| 19 121 | 4.63e−01 | 0.53 | 1.31e−01 | 0.97 | 1.32e−01 | 0.97 | 9.96e−02 | 1.00 |
| 76 001 | 2.29e−01 | 0.51 | 3.48e−02 | 0.96 | 3.49e−02 | 0.96 | 2.85e−02 | 0.91 |

of PPR, the same Delaunay meshes as in Example 1 would serve the purpose. The results are listed in Tables 4.9 and 4.10. The numerical results indicate that all three PPR strategies have improved the error on each mesh level.

As for $L_2$ error, it can observed from Tables 4.11 and 4.12 that all three strategies superconverge at rate of $O(N^{-1})$ asymptotically.

**Example 4.** In all previous examples, solutions are analytic. Let us consider the Laplace equation on the L-shaped domain $\Omega = (-1, 1) \times (-1, 1) \setminus (0, 1) \times (-1, 0)$. The Dirichlet boundary condition is imposed so that the true solution $u = r^{2/3} \sin(2\theta/3)$ in polar coordinates. In order to remove the pollution caused by the corner singularity, recovery based adaptive method [3] is employed. We start with an initial mesh shown in Fig. 4.1 and use Dörfler marking strategy [25] with $\theta = 0.3$.

Due to the corner singularity, the maximum error of $\nabla u - \nabla u_h$ is divergent. Hence we track $\|\nabla u - \nabla u_h\|_{0,\Omega}$ and $\|\nabla u - G_h u_h\|_{0,\Omega}$ instead. The numerical results are depicted in Fig. 4.2. For PPR with both Strategy 1 and Strategy 2, a superconvergence rate $O(N^{-1})$ is observed, where $N$ represents the total degrees of freedom. We also test the '*ppr*' command in COMSOL Multiphysics and obtain a superconvergence rate $O(N^{-0.9})$. In Fig. 4.2, a comparison among different strategies is made. It is observed that to achieve the same level of accuracy, both Strategy 1 and Strategy 2 require less degrees of freedom than PPR in COMSOL Multiphysics, and $De^1$ needs almost half less degrees of freedom than $De^3$.

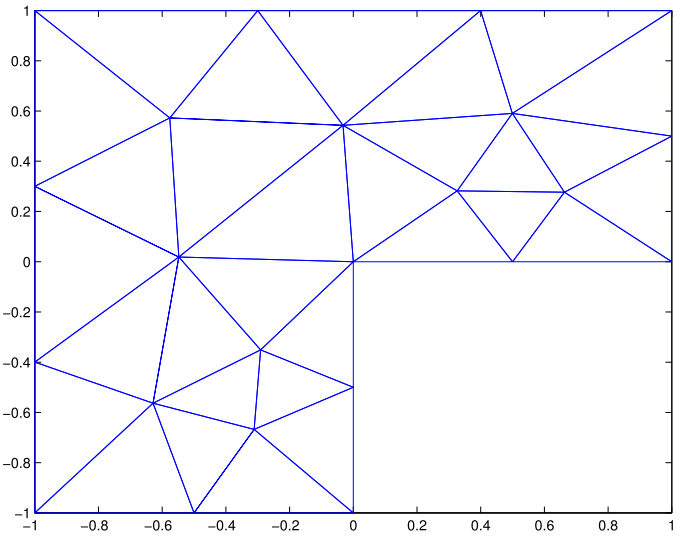


**Fig. 4.1.** Initial mesh for Example 4.

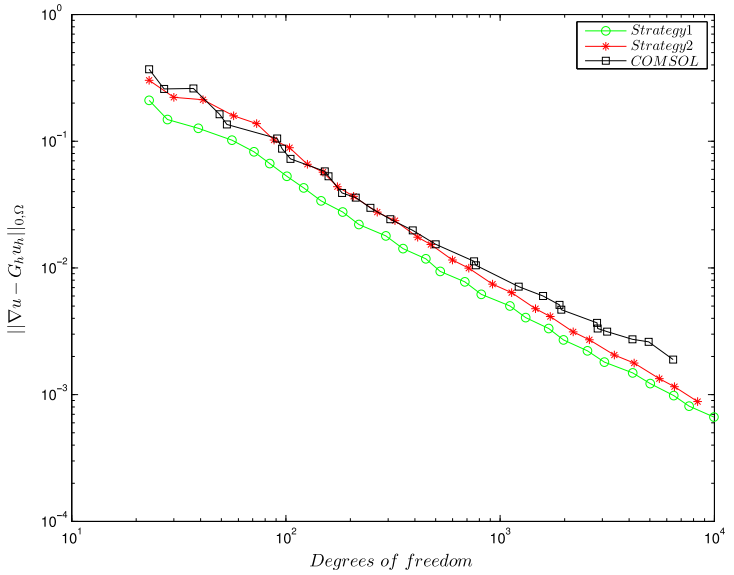


**Fig. 4.2.** Comparison of decay of error among different strategies.

## 5. Conclusion remarks

In this work, we have introduced two strategies to improve performance of PPR gradient recovery on boundary. Numerical tests provide convincing evidence that our methods inherit the superconvergence property of PPR in the interior of solution domains. We would expect that future versions of COMSOL Multiphysics may implement our methods and this in turn would benefit the scientific community.

We would like to emphasize that both strategies are problem independent and method independent just as PPR itself. In order to obtain recovered gradient on the boundary, all we need are numerical data nearby. It does not matter what is the original problem, even though the quality of the recovery might be influenced by the underlying problem, the method itself is universal. Although our technique is demonstrated for the finite element method, it can be well applied to other methods, such as finite difference method and finite volume method, as long as numerical data are provided at some sampling points.

Finally, boundary recovery technique can be used at an interface, where the solution or its gradient has jumps. In other words, we treat an interface (if the location is known *a priori*) as a boundary when performing gradient recovery.

## Acknowledgments

The research of the first author was supported in part by U.S. National Science Foundation through grant DMS-1419040. The research of the second author was supported in part by the National Natural Science Foundation of China under grants 11471031, 91430216, and the U.S. National Science Foundation through grant DMS-1419040. The research of the fourth author was supported in part by the National Natural Science Foundation of China under grants 11171359 and 11428103 and by Guangdong Natural Science Foundation under grant 2014A030313179.

## References

[1] H. Guo, Z. Zhang, Gradient recovery for the Crouzeix–Raviart element, J. Sci. Comput. 64 (2) (2015) 456–476.
[2] Y. Huang, N. Yi, The superconvergent cluster recovery method, J. Sci. Comput. 44 (3) (2010) 301–322.
[3] A. Naga, Z. Zhang, A posteriori error estimates based on the polynomial preserving recovery, SIAM J. Numer. Anal. 42 (4) (2004) 1780–1800.
[4] B. Pouliot, M. Fortin, A. Fortin, E. Chamberland, On a new edge-base gradient recovery technique, Internat. J. Numer. Methods Engrg. 93 (1) (2013) 52–65.
[5] L. Song, Zhimin Zhang, Superconvergence property of an over-penalized discontinuous Galerkin finite element gradient recovery method, J. Comput. Phys. 299 (2015) 1004–1020.
[6] Z. Zhang, A. Naga, A new finite element gradient recovery method: superconvergence property, SIAM J. Sci. Comput. 26 (4) (2005) 1192–1213.
[7] J.Z. Zhu, O.C. Zienkiewicz, Superconvergence recovery technique and a posteriori error estimators, Internat. J. Numer. Methods Engrg. 30 (1990) 1321–1339.
[8] O.C. Zienkiewicz, J.Z. Zhu, The superconvergent patch recovery (SPR) and adaptive finite element refinement, Comput. Methods Appl. Mech. Engrg. 101 (1992) 207–224.
[9] O.C. Zienkiewicz, J.Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. I. The recovery technique, Internat. J. Numer. Methods Engrg. 33 (7) (1992) 1331–1364.
[10] O.C. Zienkiewicz, J.Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. II. Error estimates and adaptivity, Internat. J. Numer. Methods Engrg. 33 (1992) 1365–1382.
[11] W. Cao, Superconvergence analysis of the linear finite element method and a gradient recovery postprocessing on anisotropic meshes, Math. Comp. 84 (291) (2015) 89–117.
[12] W. Huang, R.D. Russell, Adaptive Moving Mesh Methods, Springer, New York, 2011.
[13] L. Kamenski, W. Huang, How a nonconvergent recovered Hessian works in mesh adaptation, SIAM J. Numer. Anal. 52 (2014) 1692–1708.
[14] G.M. Porta, S. Perotto, F. Ballio, Anisotropic mesh adaptation driven by a recovery-based error estimator for shallow water flow modeling, Internat. J. Numer. Methods Fluids 70 (2012) 269–299.
[15] L. Shen, A. Zhou, A defect correction scheme for finite element eigenvalues with applications to quantum chemistry, SIAM J. Sci. Comput. 28 (2006) 321–338.
[16] A. Naga, Z. Zhang, A. Zhou, Enhancing eigenvalue approximation by gradient recovery, SIAM J. Sci. Comput. 28 (2006) 1289–1300.
[17] A. Naga, Z. Zhang, Function value recovery and its application in eigenvalue problems, SIAM J. Numer. Anal. 50 (1) (2012) 272–286.
[18] H. Guo, Z. Zhang, R. Zhao, Superconvergent two-grid methods for elliptic eigenvalue problems, arXiv:1405.4641.
[19] COMSOL Multiphysics User's Guide, Version 3.5a, 2008.
[20] COMSOL Multiphysics User's Guide, Version 4.3, 2012.
[21] S.C. Brenner, L.R. Scott, The Mathematical Theory of Finite Element Methods, third ed., in: Texts in Applied Mathematics, Springer, New York, 2008.
[22] P.G. Ciarlet, The Finite Element Method for Elliptic Problems, North-Holland, Amsterdam, 1978.
[23] 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) 769–798.
[24] J.R. Shewchuk, Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator, in: M.C. Lin, D. Manocha (Eds.), Applied Computational Geometry: Towards Geometric Engineering, in: Lecture Notes in Comput. Sci., vol. 1148, Springer-Verlag, Berlin, 1996, pp. 203–222.
[25] W. Dörfler, A convergent adaptive algorithm for Poisson's equation, SIAM J. Numer. Anal. 33 (1996) 1106–1124.