An Edge-based Anisotropic Mesh Refinement Algorithm and its Application to Interface Problems

Duan Wang[*], Ruo Li[†] and Ningning Yan[‡]

November 20, 2009

Abstract

Based on an error estimate in terms of element edge vectors on arbitrary unstructured simplex meshes, we propose a new edge-based anisotropic mesh refinement algorithm. As the mesh adaptation indicator, the error estimate involves only the gradient of error rather than higher order derivatives. The preferred refinement edge is chosen to reduce the maximal term in the error estimate. The algorithm is implemented in both two and three dimensional cases, and applied to the singular function interpolation and the elliptic interface problem. The numerical results illustrate that the convergence order can be higher using the proposed anisotropic mesh refinement algorithm than the isotropic one.

Subj class: 65N22, 65N50, 65N55
Keywords: Adaptive finite element method; Anisotropic mesh refinement; Elliptic interface problem; Non-homogeneous jump; *A posteriori* error estimate.

1 Introduction

In the singular or nearly singular problems, the structures of singularity often exhibit "low-dimensional" feature that the solutions vary significantly in some directions but mildly in other directions. To numerically approximate such solutions efficiently, no doubt we prefer anisotropic meshes, which are of different length scales in different directions and fit the anisotropic feature in the solutions. Numerous examples, including pervasive layer structures and interface discontinuities, have shown the efficiency of anisotropic elements in reducing computational cost and improving approximation accuracy [1, 5, 6, 21, 26, 28]. The main focus of this paper is the elliptic interface problem with homogeneous and non-homogeneous jump conditions, which attracts a lot of interests as it is omnipresent in many

[*]LMAM & School of Mathematical Sciences, Peking University, 100871, Beijing, P. R. China, Email: doreenwd@pku.edu.cn.

[†]CAPT, LMAM & School of Mathematical Sciences, Peking University, 100871, Beijing, P. R. China, Email: rli@math.pku.edu.cn.

[‡]LSEC, Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, 100080, Beijing, P. R. China, Email: ynn@amss.ac.cn.

scientific and engineering problems, including multi-phase flows, nano-electronic devices, electromagnetic wave propagation in heterogeneous waves, implicit solvent models in structural biology, biological membrane, etc. To resolve this layer anisotropy, we develop an anisotropic refinement algorithm, which can be effective not only for the interface problem, but also for problems with global anisotropy.

Compared with isotropic elements, the description of anisotropic meshes needs more information. Take two-dimensional triangular element as example, its anisotropy can be measured in two main aspects [8]. One is orientation, which is roughly the direction of its longest side. The other is the aspect ratio, which measures how thin the triangle is. The first quantity is supposed to be more crucial to the success of anisotropic element. Goodman et al [15] once gave an example showing that a wrong direction may lead to non-convergence. In the past decades, great improvement has been made in numerical analysis of linear interpolation on anisotropic triangular meshes [18, 22, 3, 7]. The main conclusion can be roughly stated as: given the area of a triangular element τ, the error (in L^p-norm) for the linear interpolation of a function u at the vertices of τ is nearly the minimum when τ is aligned with the eigenvector (associated with the smaller eigenvalue) of the Hessian $\nabla^2 u$, and the aspect ratio (or stretch ratio) of τ is about the square root of the ratio of the greater eigenvalue of $\nabla^2 u$ to the smaller one. For quadratic interpolation, the anisotropic orientation depends on $\nabla^3 u$ [8] and higher order interpolation may have similar properties. Based on this analysis, some anisotropic mesh optimization methodologies have been developed [9, 4, 16], which try to minimize the error by relocating nodes. From a practical point of view, since the solution of the problem is unknown, a crucial point in these methods is how to approximate high order derivatives as $\nabla^2 u$ or $\nabla^3 u$ efficiently and accurately. When the solution is not regular enough, the accuracy in recovery of the high order derivatives can be misleading.

As an effort to remedy the difficulty of requiring high order derivatives in the above mesh adaptive algorithm, the method we proposed depends on only the first order derivatives of $u(u_h)$. For each element with indicator above the given tolerance, one edge is chosen as the preferred refinement edge. The affine map from the reference element to the actual element plays an essential role in anisotropic error analysis. In [11, 12], Formaggia et al proved that the sum of error gradient projection onto two principal axes of the affine map is an upper bound of the element error. We project the error gradient onto the element edges instead to find the preferred refinement edge. Since the Jacobian matrix of the affine map can be expressed by edge vectors when we use the unit reference triangle, the sum of error gradient projection onto the three edges is again an upper bound of the element error. To reduce this upper bound error estimate, the most efficient mesh adaptation is to refine the edge with the maximal contribution to the estimate. The algorithm is first validated in the interpolation of given functions. The H^1 error between the function and its interpolation is used as the adaptive indicator. The example for two-dimensional smooth function shows the advantage of anisotropic refinement in saving degrees of freedom. The examples for the gradient discontinuous functions demonstrate the capability of the anisotropic algorithm in improving the convergence order. Then as an application, we solve a 2nd order elliptic immersed-interface problem with homogeneous and non-homogeneous jump conditions. In this case, the simple recovery-type Zienkiewicz-Zhu error estimator [29] is adopted as the adaptation indicator. The error convergence is similar as the interpolation of the weakly discontinuous functions.

The layout of this paper is as follows: in section 2, we introduce the model problem and its finite element discretization for elliptic interface problem, including homogeneous and non-homogeneous cases. In section 3, we describe the algorithm of adaptive anisotropic mesh refinement. In section 4, we give the edge-based error analysis and the strategy to choose the preferred refinement edge. In section 5, numerical experiments are presented to demonstrate the performance of our method on function interpolation and elliptic interface problems. Concluding remarks in section 6 close this paper.

2 Model problem and finite element discretization

Consider the following elliptic interface problem (see, e.g., [14]):

$$\left\{\begin{array}{rcll} -\nabla\cdot\beta^-\nabla u &=& f & \text{in} \quad \Omega^-, \\ -\nabla\cdot\beta^+\nabla u &=& f & \text{in} \quad \Omega^+, \\ [\![u]\!]_\Gamma &=& w, & \\ [\![\beta\dfrac{\partial u}{\partial \vec{n}}]\!]_\Gamma &=& Q, & \\ u|_{\partial\Omega} &=& g, & \end{array}\right. \tag{2.1}$$

where Ω is a bounded domain with its boundary $\partial\Omega$, Ω^- and Ω^+ are sub-domains of Ω such that $\Omega^- \bigcap \Omega^+ = \emptyset$ and $\overline{\Omega^-} \bigcup \overline{\Omega^+} = \overline{\Omega}$, and

$$\beta(x) = \left\{\begin{array}{ll} \beta^- & x \in \Omega^- \\ \beta^+ & x \in \Omega^+ \end{array}\right. .$$

For simplicity, we assume that $\overline{\Omega^-} \bigcap \partial\Omega = \emptyset$. We denote $\Gamma = \overline{\Omega^-} \bigcap \overline{\Omega^+}$ to be the interface separating Ω^- and Ω^+, and n to be the unit vector normal of Γ pointing from Ω^- to Ω^+ (see, e.g., Figure(2.1)). As usual, $[\![\cdot]\!]_\Gamma$ denotes the jump across Γ, i.e., for any function ξ,

$$[\![\xi]\!]_\Gamma(x) = \lim_{y\to x,\ y\in\Omega^+} \xi(y) - \lim_{y\to x,\ y\in\Omega^-} \xi(y), \quad x \in \Gamma.$$

We assume that Γ is sufficiently smooth, $\beta(x) \geq \beta_0 > 0$, $f \in L^2(\Omega)$, w, Q and g are all smooth and bounded given functions.

If the jump conditions are homogeneous, i.e., $w = 0$ and $Q = 0$, the weak solution $u \in H^1(\Omega)$ and satisfies

$$\int_\Omega \beta\nabla u\cdot\nabla v dx = \int_\Omega f v dx \quad \forall v \in H_0^1(\Omega), \tag{2.2}$$

where $u|_{\partial\Omega} = g$.

In general, the weak solution $u \notin H^1(\Omega)$ for the non-homogeneous problem. In this case, we follow [14] and extend the non-homogeneous jump $w : \Gamma \to \mathbb{R}$ to a piecewise smooth function $\hat{w} : \Omega \to \mathbb{R}$ such that

$$[\![\hat{w}]\!]_\Gamma = w.$$

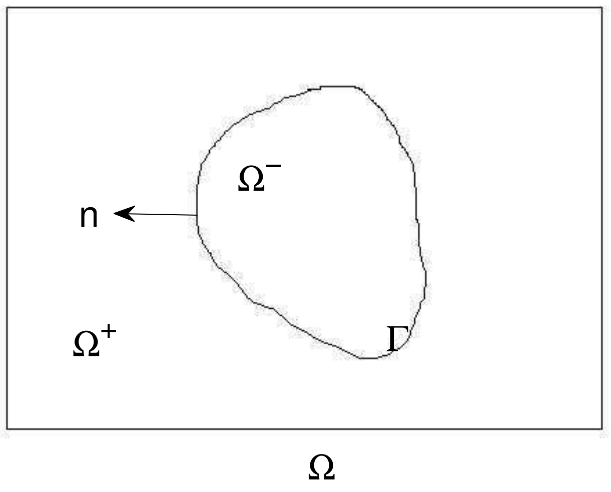

Figure 2.1: Domain of the interface problem.

Then the problem (2.1) is equivalent to finding $u = q + \hat{w}$, where $q \in H^1(\Omega)$, such that $q|_{\partial\Omega} = g - \hat{w}$, and

$$\int_\Omega \beta \nabla q \cdot \nabla v dx = \int_\Omega f v dx - \int_\Gamma Q v ds - \int_\Omega \beta \nabla \hat{w} \cdot \nabla v dx \quad \forall v \in H_0^1(\Omega). \tag{2.3}$$

Let $\mathcal{T}_h$ be a triangulation on Ω, then we have the standard piecewise polynomial finite element space $V_h \subset C(\Omega)$ on this partition. The finite element discretization of the homogeneous jump problem (2.2) is to *find* $u_h \in V_{h,g}$, *such that*

$$(\beta \nabla u_h, \nabla v_h)_\Omega = (f, v_h)_\Omega, \quad \forall v_h \in V_{h,0}, \tag{2.4}$$

where

$$V_{h,\psi} = \{v_h \in V_h : \ v_h|_{\partial\Omega} = \psi\},$$

and $(\cdot,\cdot)_\Omega$ *denotes the* L^2 *inner product on* Ω*.*

The corresponding finite element discretization of the non-homogeneous jump problem (2.3) is to *find* $q_h \in V_{h,g-\hat{w}}$, *such that*

$$(\beta \nabla q_h, \nabla v_h)_\Omega = (f, v_h)_\Omega - \int_\Gamma Q v_h ds - (\beta \nabla \hat{w}, \nabla v_h)_\Omega, \quad \forall v_h \in V_{h,0}. \tag{2.5}$$

For above schemes, there are only very loose constraints (see Assumption (1) in section 4) on the partition except that the elements are required to be triangles without hanging nodes. The highly anisotropic triangles are accepted in the partition for efficiency of the numerical approximation. We start from a quasi-uniform background mesh and adaptively generate the anisotropic mesh based on the indicators and algorithms provided in section 3 and 4.

3 Algorithm of adaptive anisotropic mesh refinement

We first present the adaptive anisotropic refinement algorithm assuming the preferred refinement edges are given. Let the domain Ω be an n-dimensional ($n = 2$ or 3) polygonal domain, and a partition $\mathcal{T}_h$ on Ω with triangle elements or tetrahedron elements be quasi-uniform, as the background mesh of our adaptive algorithm. As the common framework, there is a finite element space V_h on this partition and a finite element solution u_h in the space V_h, as the approximation of a function u, which can be the solution of a PDE. Based on the information of u_h, a number can often be calculated on every element $e \in \mathcal{T}_h$, denoted as $C_e(u_h)$, which shows the distribution of error. This $C_e(u_h)$ can be used as the indicator of mesh adaptation and commonly calculated by an *a posteriori* error estimate.

The standard chart flow of the local mesh refinement algorithm can now be described as an iterative procedure as follows:

```
Algorithm 1:
  1. set $k = 0$, and $\mathcal{T}_h^{(0)} = \mathcal{T}_h$;
  2. solve for $u_h$ on $\mathcal{T}_h^{(k)}$, and compute the refinement indicator $C_e(u_h)$;
  3. judge if the quality of the numerical solution is good enough based on $C_e(u_h)$: if
     yes, stop;
  4. generate a new mesh based on $u_h$ and $\mathcal{T}_h^{(k)}$;
  5. let $k := k + 1$ and set the new mesh as $\mathcal{T}_h^{(k)}$, then go to step (2);
```

We focus on step (4) of this procedure. There are two main differences between the anisotropic and isotropic mesh refinement algorithms. The first is that it is necessary to choose the preferred refinement edge in the anisotropic case, which is presented in the next section. The second is how to update the local mesh structure, which is discussed below.

Suppose for elements in $\mathcal{T}_h$ with its indicators above the given tolerance, marks are already set on the preferred refinement edges. We demonstrate how the mesh structure is updated. For two-dimensional case, the following Algorithm 2 bisects all marked edges while the mesh remains to be free from hanging nodes.

```
Algorithm 2:
    1. denote $\mathcal{S} := \{e \in \mathcal{T}_h | \text{at least one edge of } e \text{ is marked}\}$ to be initial set of elements
       for refinement;
    2. stop if $\mathcal{S} = \emptyset$;
    3. fetch one element $e$ from $\mathcal{S}$ and let $\mathcal{S} \to \mathcal{S} \backslash \{e\}$;
    4. set the current edge to $e$'s preferred refinement edge, if any, or to a randomly chosen
       marked edge of e;
    5. bisect $e$ to two elements $e_1$ and $e_2$ along the current edge;
    6. add element $e_i$, $i = 1, 2$, to the set $\mathcal{S}$ if it has marked edges;
    7. goto step (2);
```

In this algorithm, the element number of the waiting set $\mathcal{S}$ is not monotonically decreasing, while the number of marked edges is monotonically decreasing, by one every step. Since every element in $\mathcal{S}$ should have at least one marked edge, the algorithm will stop in a finite number of steps.

Precisely, the above refinement algorithm produces three cases, as illustrated in Figure (3.1):

1. Elements with only one marked edge are bisected as in Figure (3.1)(a);

2. Elements with two marked edges are divided into three elements after two operations as shown in Figure (3.1)(b). The edge to refine first is the preferred refinement edge, if any, or randomly chosen from the two marked edges. In both cases, exactly one of the two elements created by the first refinement operation has one marked edge, which will be handled by case (1);

3. Elements with three edges marked are divided into four elements after three operations, as shown in Figure (3.1)(c). The edge to refine first is the preferred refinement edge, if any, or a randomly chosen one. The first refinement operation creates two elements with one marked edge, which will be handled by case (1).

We implement the mesh refinement algorithm for three-dimensional(3D) mesh with tetrahedral elements, too. Similarly as in 2D case, our discussion is based on element. For each tetrahedral element with indicator above the given tolerance, suppose one of its six edges is labeled as the preferred refinement edge and the new freedom is located on the midpoint. First we demonstrate the basic bisection operation for an element with one marked edge in Figure (3.2).

Different from the 2D mesh update, the mesh update in 3D space is much more involved due to the complex geometric configuration and the above edge-based bisection. For a tetrahedral element, it can share a node, an edge or a surface with its neighbor. Mesh

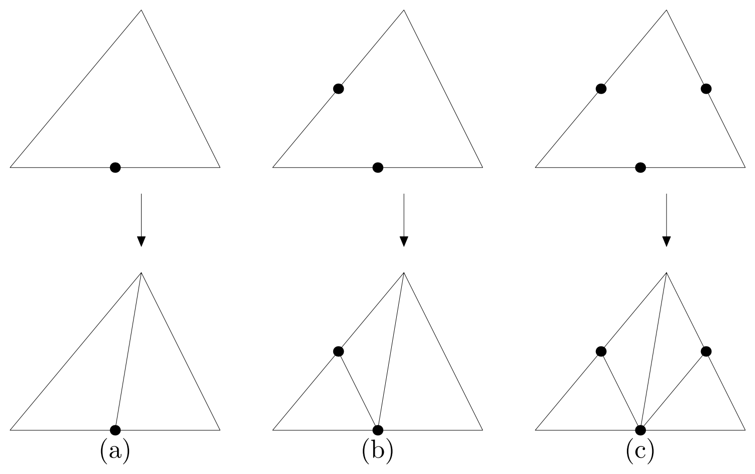

Figure 3.1: The three cases of refinement for one element in two-dimensional space.

confusion may happen on the common surface of two neighboring elements. A unique sequence number is needed for each labeled edge to guarantee that the bisection is ordered operated and mesh confusion is avoided. Even though, the algorithm can be implemented following the basic procedure of Algorithm 2 with slight modification.

```
Algorithm 2*:
  1. denote $\mathcal{S} := \{e \in \mathcal{T}_h | \text{at least one edge of } e \text{ is marked}\}$ to be initial set of elements for refinement;
  2. give each marked edge a unique sequence number;
  3. stop if $\mathcal{S} = \emptyset$;
  4. fetch one element $e$ from $\mathcal{S}$ and let $\mathcal{S} \to \mathcal{S} \backslash \{e\}$;
  5. set the current edge to $e$'s marked edge with the smallest sequence number;
  6. bisect $e$ to two elements $e_1$ and $e_2$ along the current edge;
  7. add element $e_i$, $i = 1, 2$, to the set $\mathcal{S}$ if it has marked edges;
  8. goto step (3);
```

Similarly, the above algorithm will end in finite steps.

4 Indicators for adaptive anisotropic mesh refinement

In this section, we present the strategy to choose the preferred refinement edge based on an error estimate in terms of the element edge vectors. The error estimate can be given

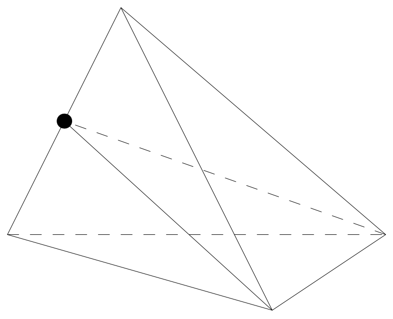

Figure 3.2: The bisection operation for tetrahedron with one labeled edge.

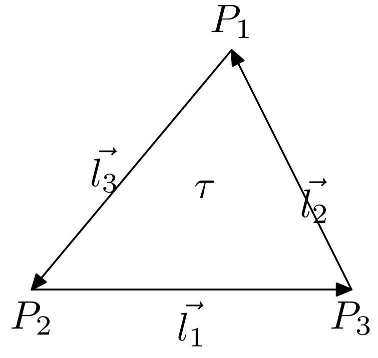

Figure 4.1: Notations on a triangle τ.

for both the function interpolation and the model elliptic equation.

We consider the error estimate of the linear interpolation at first. Let u be a quadratic polynomial, and u_I be a piecewise linear Lagrangian interpolation of u. For a given element τ, denote $P_i = (x_i, y_i)$ and $\vec{l_i}$, $1 \le i \le 3$, to be its three vertices and edges as illustrated in Figure (4.1), and c_i, $1 \le i \le 3$, to be the linear nodal basis functions (barycenter coordinates). Then u_I on the element τ can be expressed as

$$u_I|_\tau = \sum_{i=1,2,3} u(x_i, y_i)c_i.$$

Goodman provided an edge-based L^∞ error estimate for quadratic function u firstly in [15] as

$$\frac{1}{8} \max_{i=1,2,3} \vec{l_i}^T H \vec{l_i} \le \max_{x\in\tau} |u - u_I| \le \frac{1}{6} \max_{i=1,2,3} \vec{l_i}^T H \vec{l_i},$$

where H is the Hessian matrix of u. Naturally, controlling the error $E(\vec{l_i}) = \vec{l_i}^T H \vec{l_i}$ leads to an edge-based anisotropic refinement algorithm if H is known.

Let us denote $\vec{E} = (E_1, E_2, E_3)^T$ with $E_i = E(\vec{l_i})$, $1 \le i \le 3$, Cao established the L^2 interpolation error in [7] as

$$\int_\tau |u - u_I|^2 dxdy = \frac{1}{4} \vec{E}^T B_0 \vec{E},$$

and Bank et al computed the H^1 interpolation error in [4] as

$$\int_\tau |\nabla u - \nabla u_I|^2 dxdy = \frac{1}{4}\vec{E}^T B_1 \vec{E},$$

where matrices B_0 and B_1 are usually positive definite and diagonal dominant and dependent on τ. Thus the L^2 and H^1 interpolation error can be approximated as

$$\int_\tau |u - u_I|^2 dxdy \approx C(E_1^2 + E_2^2 + E_3^2)$$

and

$$\int_\tau |\nabla u - \nabla u_I|^2 dxdy \approx C_1(E_1^2 + E_2^2 + E_3^2),$$

where constants C and C_1 are dependent on the triangle. Suppose an element is selected by some means to be refined, we can regard C and C_1 as constants on this element. The most efficient way to reduce the error of this element is to refine the edge with biggest E_i. So if the function u is locally quadratic that H can be approximated with enough accuracy, an edge-based anisotropic refinement algorithm is very likely to succeed by the above method. Bank et al [4] used this idea in their mesh smoothing method by solution of a minimization problem.

Let us try to deduce the error estimate in terms of the element edge vectors without the information of the Hessian matrix. With the first order derivatives, Formaggia et al [11, 12] analyzed the linear L^2 interpolation error, based on the eigenvalues and eigenvectors of the affine map from the reference triangle to the actual triangle. For a triangle τ, let $T_\tau : \hat{\tau} \longrightarrow \tau$ be the affine transformation which maps the reference triangle $\hat{\tau}$ into τ. Let M_τ be the Jacobian matrix of the map

$$\vec{x} = T_\tau(\vec{\hat{x}}) = M_\tau \vec{\hat{x}} + \vec{t}_\tau,$$

where $\vec{x}$, $\vec{\hat{x}}$ are coordinate vectors of actual points and reference points. Since M_τ is invertible, it admits a singular value decomposition (SVD): $M_\tau = R_\tau^T \Lambda_\tau P_\tau$. We set

$$\Lambda_\tau = \begin{pmatrix} \lambda_{1,\tau} & 0 \\ 0 & \lambda_{2,\tau} \end{pmatrix}, \quad R_\tau = \begin{pmatrix} \vec{r}_{1,\tau}^{\,T} \\ \vec{r}_{2,\tau}^{\,T} \end{pmatrix}, \quad \lambda_{1,\tau} \geq \lambda_{2,\tau},$$

and the geometric interpretation of this decomposition is in Figure (4.2). Let I_h be the linear interpolation operator of Clément type (see, e.g., [25]): $H^1(\Omega) \rightarrow V_h$. We assume that the partition $\mathcal{T}_h$ satisfies

Assumption 1. *The diameter of the reference patch $\Delta_{\hat{\tau}} = T_\tau^{-1}\Delta_\tau$ is independent of the mesh geometry, where $\Delta_\tau = \bigcup_{T \in \mathcal{T}_h, \bar{T}\cap\bar{\tau}\neq\emptyset} T$.*

It was then pointed out in [11, 12] that there exists a constant C depending only on the reference element $\hat{\tau}$ (we use this notation for C from now on) such that, for all $\tau \in \mathcal{T}_h$ and an arbitrary function $v \in H^1(\Omega)$,

$$\begin{cases} \|v - I_h v\|_{L^2(\tau)} & \leq & C\omega_\tau(v), \\ \|v - I_h v\|_{L^2(\vec{l}_i)} & \leq & C\sqrt{\dfrac{l_i}{\lambda_{1,\tau}\lambda_{2,\tau}}}\,\omega_\tau(v) \quad i = 1, 2, 3, \end{cases}$$

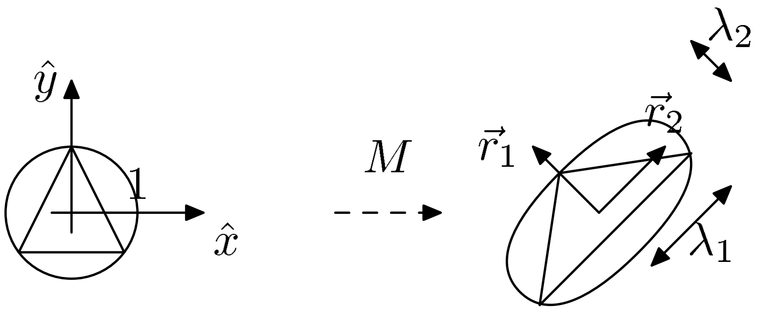

Figure 4.2: The transformation when the reference triangle is equilateral. The unit circle is mapped into an ellipse with directions $\vec{r}_1$ and $\vec{r}_2$, the amplitude of stretching being λ_1 and λ_2.

where $l_i = |\vec{l_i}|$ (the length of $\vec{l_i}$) and

$$\begin{aligned} \omega_\tau^2(v) &= \lambda_{1,\tau}^2(\vec{r}_{1,\tau}^{\,T} G_\tau(v)\vec{r}_{1,\tau}) + \lambda_{2,\tau}^2(\vec{r}_{2,\tau}^{\,T} G_\tau(v)\vec{r}_{2,\tau}), \\ G_\tau(v) &= \int_{\Delta_\tau} \begin{pmatrix} \dfrac{\partial v}{\partial x}\dfrac{\partial v}{\partial x} & \dfrac{\partial v}{\partial x}\dfrac{\partial v}{\partial y} \\ \dfrac{\partial v}{\partial x}\dfrac{\partial v}{\partial y} & \dfrac{\partial v}{\partial y}\dfrac{\partial v}{\partial y} \end{pmatrix} dxdy = \int_{\Delta_\tau} \nabla v \cdot (\nabla v)^T dxdy \end{aligned} \tag{4.1}$$

Based on the above interpolation error estimate, Picasso [20] obtained an *a priori* error estimate for the finite element approximation of the elliptic equation

$$\begin{cases} -\nabla\cdot(A\nabla u) &= f, \quad \text{in } \Omega \\ u|_{\partial\Omega} &= 0, \end{cases} \tag{4.2}$$

where $f \in L^2(\Omega)$. Let u_h be the linear finite element solution and $e = u - u_h$, it was proved in [20]

$$\begin{aligned} \int_\Omega A\nabla e\cdot\nabla e \;\; &\le \;\; C\sum_{\tau\in\mathcal{T}_h}\Big(\|f + div(A\nabla u_h)\|_{L^2(\tau)} \\ &\quad + \frac{1}{2}\sum_{i=1}^{3}\sqrt{\frac{l_i}{\lambda_{1,\tau}\lambda_{2,\tau}}}\|[\![A\nabla u_h\cdot\vec{n}]\!]\|_{L^2(\vec{l_i})}\Big) \times \omega_\tau(e), \end{aligned} \tag{4.3}$$

where $\vec{n}$ is the unit out normal on edge $\vec{l_i}$.

Denote $E'(\vec{l}) \triangleq \vec{l}^{\,T} G_\tau(e)\vec{l}$, the two terms $E'(\lambda_{i,\tau}\vec{r}_{i,\tau}) = \lambda_{i,\tau}^2\vec{r}_{i,\tau}^{\,T} G_\tau(e)\vec{r}_{i,\tau} (1 \le i \le 2)$ in $\omega_\tau^2(e)$ can actually be written as

$$\lambda_{i,\tau}^2\vec{r}_{i,\tau}^{\,T} G_\tau(e)\vec{r}_{i,\tau} = \int_{\Delta_\tau}\left|\left(\frac{\partial e}{\partial x},\frac{\partial e}{\partial y}\right)\cdot\lambda_{i,\tau}\vec{r}_{i,\tau}\right|^2 dxdy.$$

Recall the geometry property of $\vec{r}_{i,\tau}$ and $\lambda_{i,\tau}$, $E'(\lambda_{i,\tau}\vec{r}_{i,\tau})$ is then the projection of the error gradient on the principal axis $\vec{r}_{i,\tau}$ with length to be $\lambda_{i,\tau}$. Since the three edge vectors can be represented by linear compositions of the principal axes, it is suggested to examine the projection of the error gradient on the edges of the triangle. Specifically, we set the reference element to be the unit triangle $\hat{\tau} = \{(\hat{x}, \hat{y}), 0 \leq \hat{y} \leq 1 - \hat{x}, \hat{x} \in [0, 1]\}$, with its vertices as $\hat{p}_1 = (0, 0)$, $\hat{p}_2 = (1, 0)$ and $\hat{p}_3 = (0, 1)$. The affine map $T_\tau : \hat{\tau} \to \tau$ satisfies:

$$\vec{x} = T_\tau(\vec{\hat{x}}) = M_\tau \vec{\hat{x}} + \vec{t}_\tau,$$

and the Jacobian matrix is

$$M_\tau \times \begin{pmatrix} 1-0, & 0-0 \\ 0-0, & 1-0 \end{pmatrix} = \begin{pmatrix} \vec{l}_3 & -\vec{l}_2 \end{pmatrix},$$

i.e., $M_\tau = (\vec{l}_3, -\vec{l}_2)$. We have

Lemma 1. *Let $v \in H^1(\tau)$, and $\hat{v}$ be the corresponding function defined on the reference element $\hat{\tau}$. Then*

$$\frac{\lambda_{2,\tau}}{\lambda_{1,\tau}} \|\nabla v\|^2_{L^2(\tau)} \leq \|\hat{\nabla} \hat{v}\|^2_{L^2(\hat{\tau})} \leq |M_\tau|^{-1} \sum_{i=1,2,3} \|\vec{l}_i^{\,T} \cdot \nabla v\|^2_{L^2(\tau)}, \tag{4.4}$$

where $|M_\tau|$ denotes the determinant of M_τ.

Proof. Noting that

$$\hat{\nabla} \hat{v} = M_\tau^T \nabla v,$$

we have

$$\|\hat{\nabla} \hat{v}\|^2_{L^2(\hat{\tau})} = |M_\tau|^{-1} \int_\tau (\vec{l}_3^T \nabla v)^2 + (\vec{l}_2^T \nabla v)^2 \leq |M_\tau|^{-1} \sum_{i=1,2,3} \|\vec{l}_i^{\,T} \nabla v\|^2_{L^2(\tau)}$$

The upper bound for $\|\nabla v\|_{L^2(\tau)}$ in terms of $\|\hat{\nabla} \hat{v}\|_{L^2(\hat{\tau})}$ is given by Formaggia et al [11] in Lemma 2.2. □

The L^2 error of the interpolation of Clément type for $v \in H^1(\Omega)$ is estimated as

Proposition 1. *Let $v \in H^1(\Omega)$. Then*

$$\|v - I_h v\|^2_{L^2(\tau)} \leq C \sum_{i=1,2,3} \vec{l}_i^{\,T} G_\tau(v) \vec{l}_i, \tag{4.5}$$

where $G_\tau(v)$ is given in (4.1).

Proof. Considering $v \in H^1(\tau)$, let $\hat{v}$ be the corresponding function defined on the reference element $\hat{\tau}$. Using the normal relation between $\hat{\tau}$ and τ, we have

$$\|v - I_h v\|^2_{L^2(\tau)} = |M_\tau| \|\hat{v} - \hat{I}_h \hat{v}\|^2_{L^2(\hat{\tau})},$$

Applying the well known Bramble-Hilbert lemma (see, e.g. [10]), it can be concluded that

$$\|\hat{v} - \hat{I}_h \hat{v}\|_{L^2(\hat{\tau})} \leq C |\hat{v}|_{H^1(\Delta_{\hat{\tau}})}.$$

Thus,

$$\|v - I_h v\|^2_{L^2(\tau)} \le C|M_\tau||\hat{v}|^2_{H^1(\Delta_{\hat{\tau}})} = C|M_\tau| \sum_{\hat{T}\in\Delta\hat{\tau}} |\hat{v}|^2_{H^1(\hat{T})}.$$

Now the semi-norms $|\hat{v}|_{H^1(\hat{T})}$ can be replaced by inequality (4.4) in Lemma 1. It follows that

$$\|v - I_h v\|^2_{L^2(\tau)} \;\le\; C \sum_{T\in\Delta_\tau} \sum_{i=1,2,3} \|\vec{l_i}^T \cdot \nabla v\|^2_{L^2(T)} = C \sum_{i=1,2,3} \vec{l_i}^T G_\tau(v)\vec{l_i}.$$

□

Proposition 2. *Let $v \in H^1(\Omega)$. Then*

$$|v - I_h v|^2_{H^1(\tau)} \le C\frac{1}{\lambda^2_{2,\tau}} \sum_{i=1,2,3} \vec{l_i}^T G_\tau(e)\vec{l_i}, \tag{4.6}$$

where $e = v - I_h v$.

Proof. It follows from inequality (4.4) that

$$\begin{aligned}
|v - I_h v|^2_{H^1(\tau)} &\le \frac{\lambda_{1,\tau}}{\lambda_{2,\tau}}|\hat{v} - \hat{I}_h\hat{v}|^2_{H^1(\hat{\tau})} \\
&= \frac{\lambda_{1,\tau}}{\lambda_{2,\tau}}\|\hat{\nabla}\hat{e}\|^2_{L^2(\hat{\tau})} \\
&\le \frac{\lambda_{1,\tau}}{\lambda_{2,\tau}}|M_\tau|^{-1} \sum_{i=1,2,3} \|\vec{l_i}^T \nabla e\|^2_{L^2(\tau)} \\
&\le \frac{\lambda_{1,\tau}}{\lambda_{2,\tau}}|M_\tau|^{-1} \sum_{i=1,2,3} \vec{l_i}^T G_\tau(e)\vec{l_i}.
\end{aligned}$$

Note that $|M_\tau| = \lambda_{1,\tau}\lambda_{2,\tau}$, we can finish the proof. □

Proposition 3. *Let $v \in H^1(\Omega)$. Then*

$$\|v - I_h v\|^2_{L^2(\vec{l_j})} \le C\frac{l_j}{|M_\tau|} \sum_{i=1,2,3} \vec{l_i}^T G_\tau(v)\vec{l_i}, \quad j = 1, 2, 3. \tag{4.7}$$

Proof. By (4.4) we have that

$$\begin{aligned}
\|v - I_h v\|^2_{L^2(\vec{l_j})} &= \frac{l_j}{\hat{l_j}}\|\hat{v} - \hat{I}_h\hat{v}\|^2_{L^2(\hat{\vec{l_j}})} \\
&\le l_j\|\hat{v} - \hat{I}_h\hat{v}\|^2_{L^2(\partial\hat{\tau})} \\
&\le Cl_j \sum_{\hat{T}\in\Delta_{\hat{\tau}}} |\hat{v}|^2_{H^1(\hat{T})} \\
&\le Cl_j|M_\tau|^{-1} \sum_{i=1,2,3} \vec{l_i}^T G_\tau(v)\vec{l_i}.
\end{aligned}$$

□

We denote $\omega'_\tau(e) = (\sum_{i=1,2,3} E'(\vec{l_i}))^{1/2} = \left(\sum_{i=1,2,3} \vec{l_i}^T G_\tau(e)\vec{l_i}\right)^{1/2}$ and follow [20] and Proposition 1 and 3, the error estimate (4.3) for the finite element approximation of the elliptic equation can be revised as

$$\begin{aligned}\int_\Omega A\nabla e\cdot\nabla e \;\; &\le \;\; C\sum_{\tau\in\mathcal{T}_h}\Big(\|f+div(A\nabla u_h)\|_{L^2(\tau)} \\ &\quad +\frac{1}{2}\sum_{i=1}^{3}\sqrt{\frac{l_i}{|M_\tau|}}\|[\![A\nabla u_h\cdot\vec{n}]\!]\|_{L^2(\vec{l_i})}\Big)\times\omega'_\tau(e),\end{aligned} \tag{4.8}$$

with $e = u - u_h$.

We verify that $\omega'(e)$ can be bounded from above by $\omega(e)$. Precisely for each edge $\vec{l_i}$,

$$\vec{l_i} = a_i\lambda_1\vec{r}_1 + b_i\lambda_2\vec{r}_2, \quad |a_i|\le 2, \; |b_i|\le 2,$$

we have that

$$E'(\vec{l_i}) \le 2\max(a_i^2, b_i^2)(E'(\lambda_1\vec{r}_1)+E'(\lambda_2\vec{r}_2)) \le 8(E'(\lambda_1\vec{r}_1)+E'(\lambda_2\vec{r}_2)),$$

thus

$$\omega'(e)\le 2\sqrt{6}\;\omega(e).$$

Now we are in the position to set the preferred refinement edge for the elements to be refined in the mesh. The elements with the error indicator above the tolerance are marked as the first step. The error indicator can be the residual type *a posteriori* error estimates (see, e.g., [20]) or the recovery type *a posteriori* error estimates (see, e.g., [27]). With Proposition 2 and error estimate (4.8), we weight each edge $\vec{l_i}$ of a marked element with $w_i = \vec{l_i}^T G_\tau(e)\vec{l_i}$, $i = 1, 2, 3$ and mark the edge with the greatest weight. The matrix $G(e)$ in $E'(\vec{l_i})$ plays the same role as Hessian matrix H in $E(\vec{l_i})$ in certain way.

Remark 1. *In the numerical examples of the interpolation problem (see the next section), we can use the exact error e as our refinement indicators. While solving PDEs, we still need a posteriori error estimates, such as residual-based a posteriori error estimates or recovery type a posteriori error estimates (see, e.g., [29]), and $G(e)$ can only be computed by approximation. The gradient recovery of u_h (see [27] and [29] for more details) can be applied to replace the gradient of the solution u in $G(e)$.*

Remark 2. *From the proofs, it can be seen the above propositions valid due to the fact that the linear Clément interpolation satisfies $\widehat{I_h v} = \hat{I}_h\hat{v}$ and $\|\hat{v}-\hat{I}_h\hat{v}\|_{L^2(\hat{\tau})}\le C|\hat{v}|_{H^1(\Delta_{\hat{\tau}})}$. It is trivial to verify that the propositions can be extended to any interpolation with these properties, including the Clément type interpolation of higher order (see, e.g., [25]) and the Lagrangian interpolation.*

Remark 3. *On rectangular meshes, Apel et al [2] developed an anisotropic refinement method based upon error gradients by comparison of the two quantities $\left\|\frac{\partial e}{\partial x}\right\|$ and $\left\|\frac{\partial e}{\partial y}\right\|$, and obtained anisotropic meshes with hanging nodes. This method works well for straight sharp layers on structured meshes. Our method can be regarded as an extension of Apel's method on unstructured meshes.*

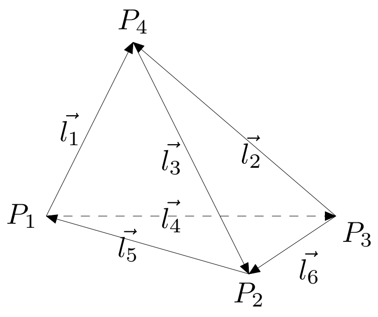

Figure 4.3: Notations on a tetrahedron τ.

Next, let us consider problems in 3D space. For tetrahedral elements, we have similar L^2 interpolation error estimate as in Proposition 1. Precisely, we set the reference element to be the unit tetrahedron $\hat{\tau}$ with vertices as $\hat{p}_1 = (0,0,0)$, $\hat{p}_2 = (1,0,0)$, $\hat{p}_3 = (0,1,0)$ and $\hat{p}_4 = (0,0,1)$. Then for an actual tetrahedral element τ as in Figure(4.3), the affine map from the reference tetrahedron to it is

$$\vec{x} = T_\tau(\vec{\hat{x}}) = M_\tau \vec{\hat{x}} + \vec{t}_\tau,$$

and the Jacobian matrix M_τ satisfies

$$M_\tau \times \begin{pmatrix} 1-0, & 0-0, & 0-0 \\ 0-0, & 1-0, & 0-0 \\ 0-0, & 0-0, & 1-0 \end{pmatrix} = \begin{pmatrix} -\vec{l_5} & \vec{l_4} & \vec{l_1} \end{pmatrix},$$

i.e., $M_\tau = (-\vec{l_5}, \vec{l_4}, \vec{l_1})$.

Proposition 4. *Suppose that Ω is a three-dimensional polygon domain, $v \in H^1(\Omega)$ and $I_h v$ is the linear Clément interpolation of v on a partition with tetrahedral elements. Then*

$$\|v - I_h v\|^2_{L^2(\tau)} \le C \sum_{i=1}^{6} \vec{l_i}^T G_\tau(v) \vec{l_i}, \tag{4.9}$$

where $G_\tau(v)$ is defined as follows

$$G_\tau(v) = \int_{\Delta_\tau} \nabla v \cdot (\nabla v)^T dxdydz, \quad \Delta_\tau = \bigcup_{T \in \mathcal{T}_h, \bar{T} \cap \bar{\tau} \neq \emptyset} T.$$

Proof. The proof is similar as in two-dimensional case. Let $\hat{v}$ be the corresponding function defined on the reference element $\hat{\tau}$. Using the normal relation between $\hat{\tau}$ and τ, we have

$$\|v - I_h v\|^2_{L^2(\tau)} = |M_\tau| \|\hat{v} - \hat{I}_h \hat{v}\|^2_{L^2(\hat{\tau})}.$$

Again by the Bramble-Hilbert lemma (see,e.g.[10]), we obtain

$$\|\hat{v} - \hat{I}_h \hat{v}\|_{L^2(\hat{\tau})} \le C |\hat{v}|_{H^1(\Delta_{\hat{\tau}})}.$$

Thus,

$$\begin{aligned}\|v - I_h v\|^2_{L^2(\tau)} &\leq C|M_\tau||\hat{v}|^2_{H^1(\Delta_{\hat{\tau}})} \\ &= C|M_\tau| \sum_{\hat{T}\in\Delta\hat{\tau}} \|\hat{\nabla}\hat{v}\|^2_{L^2(\hat{T})}\end{aligned}$$

Notice

$$\hat{\nabla}\hat{v} = M_\tau^T \nabla v$$

still holds in 3D case, then

$$\begin{aligned}\|v - I_h v\|^2_{L^2(\tau)} &\leq C \sum_{T\in\Delta_\tau} (\|\vec{l_5}^T \nabla v\|^2_{L^2(T)} + \|\vec{l_4}^T \nabla v\|^2_{L^2(T)} + \|\vec{l_1}^T \nabla v\|^2_{L^2(T)}) \\ &\leq C \sum_{T\in\Delta_\tau} \sum_{i=1}^{6} \|\vec{l_i}^T \cdot \nabla v\|^2_{L^2(T)} = C \sum_{i=1}^{6} \vec{l_i}^T G_\tau(v) \vec{l_i}.\end{aligned}$$

$\square$

By this estimate, our algorithm may be extended to the 3D space. We can then choose the one of the six edges with the greatest weight $\vec{l_i}^T G \vec{l_i}$ as the preferred refinement edge. Example 3(6) in the next section is presented to show the numerical behavior of the 3D anisotropic interpolation(interface problem).

5 Numerical Experiments

The proposed algorithm has been implemented by using C++ programming language based on the adaptive finite element package AFEPack [17]. We examine the numerical efficiency of the algorithm for the interpolation of functions, and then apply the method to the 2nd order elliptic interface problem.

For comparison, we also implement the isotropic refinement algorithm using the longest edge bisection strategy [23, 24], which bisect the longest edge first. Both algorithms use the same error indicator for adaptive finite element mesh refinement, except that edge indicator provided in this paper is adopted to choose the preferred refined edge for adaptive anisotropic mesh refinement.

For all examples below, the domain Ω are set as the unit square $[0, 1] \times [0, 1]$ in 2D space or the unit cube $[0, 1] \times [0, 1] \times [0, 1]$ in 3D space if not specified.

Before presenting the numerical results, we have the optimal convergence order of the interpolation error in the number of degree of freedoms(#DOF) in Table (1). The convergence order is defined as the constant α in

$$e \approx N^{-\alpha},$$

where e is the error and N is the #DOF. The weakly discontinuous functions considered here are the functions with $O(1)$ discontinuities in its gradients on a $(D - 1)$-dimensional manifold, where $D = 2, 3$ is the dimension of the domain.

	smooth		weakly discontinuous	
	H^1 error	L^2 error	H^1 error	L^2 error
2D linear	1/2	1	1/2	3/2
2D quadratic	1	3/2	1/2	3/2
3D linear	1/3	2/3	1/4	3/4
3D quadratic	2/3	1	1/4	3/4

Table 1: The optimal convergence order of standard interpolation operators on isotropic meshes for smooth functions and weakly discontinuous functions.

	H^1 error	L^2 error
2D linear	1/2	1
2D quadratic	(1/2,1)	3/2
3D linear	(1/4,1/3)	2/3
3D quadratic	(1/4,2/3)	(3/4,1)

Table 2: The expected convergence order of standard interpolation operators using anisotropic algorithm for weakly discontinuous functions. The cell with an interval (α_0, α_1) means the order α should satisfy $\alpha_0 \le \alpha \le \alpha_1$.

The expected convergence order for weakly discontinuous function interpolation using anisotropic algorithm is presented in Table (2). For the cases the optimal order for smooth functions is less than or equal to that for the weakly discontinuous functions, the anisotropic convergence order will degenerate to the optimal order for smooth functions. For the cases the optimal order for smooth functions is greater than that for the weakly discontinuous functions, the anisotropic algorithm could improve the convergence order to be higher than the optimal convergence order of the isotropic algorithm, and it could achieve the optimal convergence order for smooth functions.

For smooth functions with sharp layers, the anisotropic algorithm could reduce the #DOF, though there can not be improvements in convergence order. Let us demonstrate this point in Example 1 at first.

5.1 Interpolation of functions

We consider the interpolation of functions with layer structures or gradient discontinuities to examine the efficiency of our anisotropic mesh refinement algorithm.
Example 1 Let the function $f(x, y)$ to be

$$f(x, y) = \tanh((x - y)/\epsilon) \tanh((1 - y - x)/\epsilon), \quad \epsilon = 0.005.$$

This function has two straight sharp layers along $y = x$ and $x + y = 1$ and the two sharp layers intersect at $(0.5, 0.5)$ and form a crossing. The background mesh is a uniform triangulation with element size as $0.1\sqrt{2}$, plotted in the left picture in Figure (5.1). We use both piecewise linear and piecewise quadratic polynomials to approximate this function

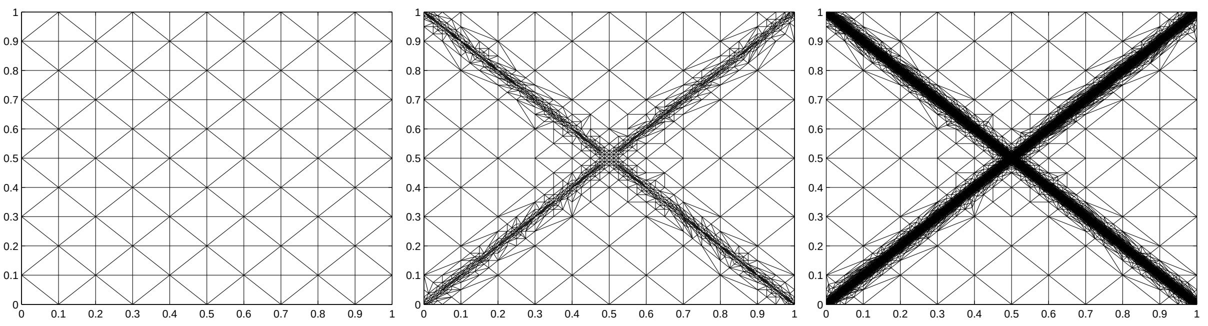

Figure 5.1: The anisotropic refinement process for the function $f(x, y)$ in Example 1. From left to right, the background mesh, the mesh obtained after 8 rounds of refinement, and the mesh after 16 rounds of refinement.

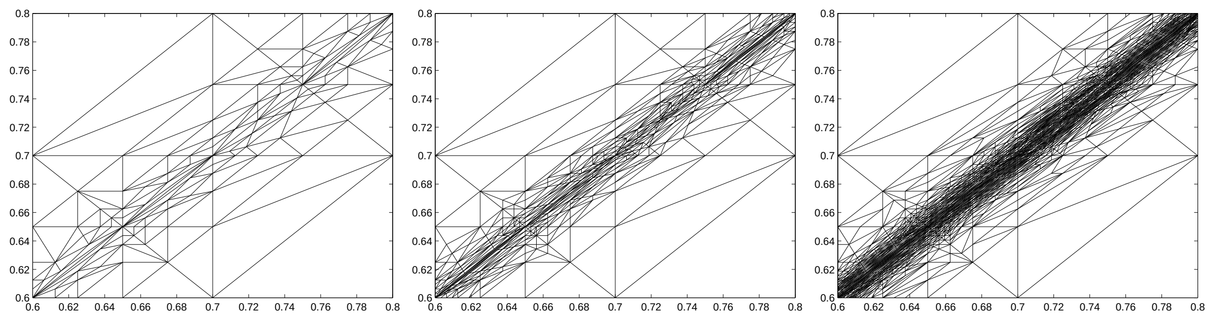

Figure 5.2: The detailed mesh structure in the square $[0.6, 0.8] \times [0.6, 0.8]$ obtained after 8, 12 and 16 rounds of anisotropic refinement for the function $f(x, y)$ of Example 1 from left to right.

and using H^1 error as indicator. Each step, we refine 30 percent elements of the old mesh, whose indicators are larger.

Figure (5.1) displays the background mesh and the meshes obtained after 8 and 16 rounds of anisotropic refinement. The refined elements of the domain are aligned along two sharp layers. The refinement procedures in two local regions of the domain are plotted in Figure (5.2) and Figure (5.3) to show the detailed mesh structures. In Figure (5.2), the mesh is refined gradually along the sharp layer and highly anisotropic elements are generated with the correct refinement direction. In Figure (5.3), the mesh structure at the crossing is isotropic, indicating that the structure of f there is actually isotropic. Such behaviors demonstrate that our algorithm can resolve the mixed structures in the function with both anisotropic and isotropic singularities.

Using the longest edge bisection strategy, we obtain the locally isotropic meshes in Figure (5.4). The refined elements exhibit wider transient bands comparing with the anisotropic mesh shown in Figure (5.1), thus the anisotropic refinement could achieve the same accuracy with less DOFs.

The log-log plots of the H^1 and L^2 error versus the #DOF for comparison of linear and quadratic interpolation are in Figure (5.5), which illustrate that the anisotropic refinement greatly improves the approximation efficiency compared to the isotropic case. In these two pictures, all the curves tend to straight lines, indicating that the error is reciprocal to the

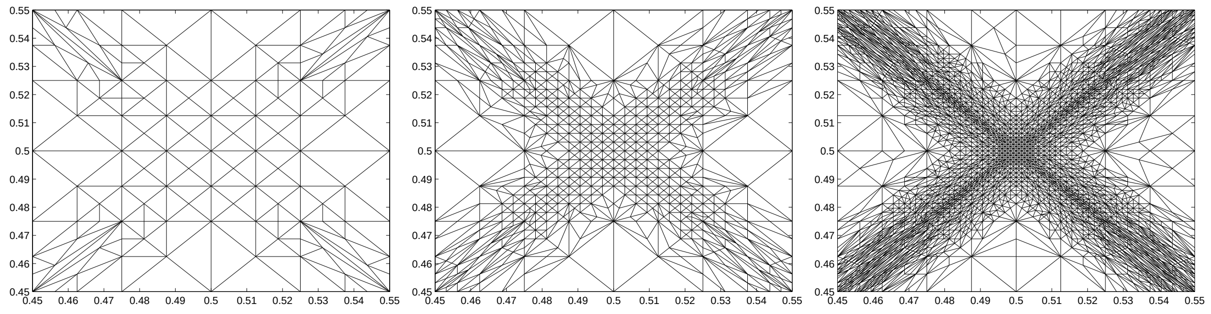

Figure 5.3: The detailed mesh structure at the crossing of the two sharp layers in the square $[0.45, 0.55] \times [0.45, 0.55]$ obtained after 8, 12 and 16 rounds of anisotropic refinement for the function $f(x, y)$ of Example 1 from left to right.

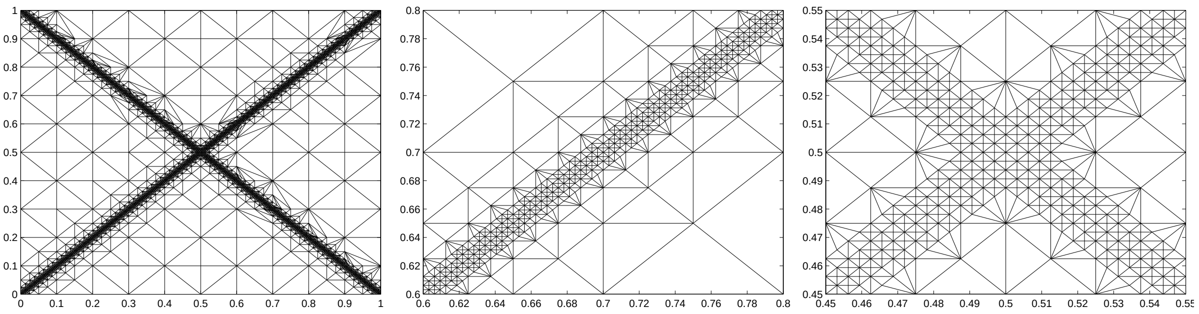

Figure 5.4: The total mesh(left) and two detailed mesh structures in $[0.6, 0.8] \times [0.6, 0.8]$(middle) and $[0.45, 0.55] \times [0.45, 0.5]$(right) obtained after 12 rounds of isotropic refinement for the function $f(x, y)$ of Example 1.

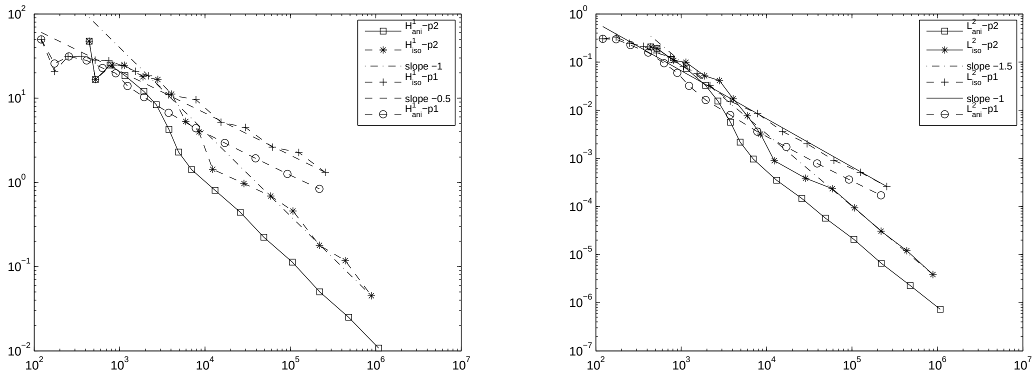

Figure 5.5: The H^1(left) and L^2(right) error in the #DOF for the comparison of quadratic and linear interpolation of f in Example 1 in log-log scale.

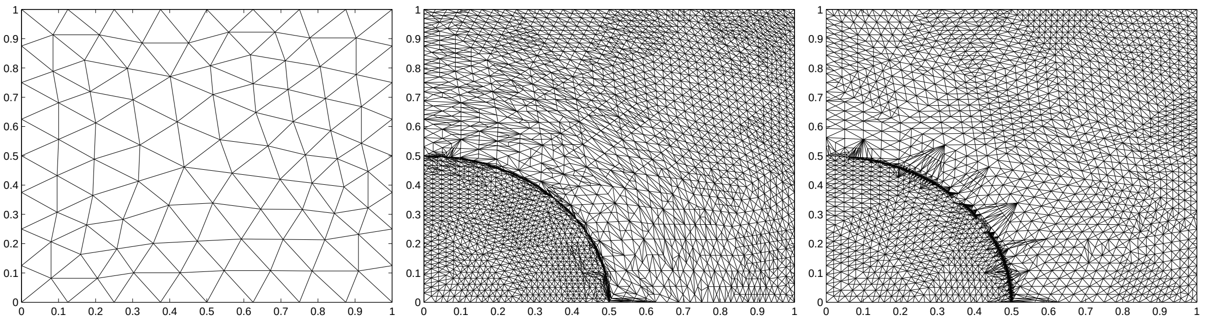

Figure 5.6: The initial mesh with 105 nodes(left), the anisotropic mesh with 4141 nodes (middle) and the isotropic mesh with 4056 nodes (right) for linear interpolation of Example 2.

#DOF and is equally distributed on all DOFs asymptotically. Both the anisotropic and isotropic L^2 linear/quadratic interpolation error curves in the right picture of Figure (5.5) tend to have the same asymptotical slope -1 and $-3/2$. So do the H^1 linear/quadratic curves in the left picture with the asymptotical slope to be $-1/2$ and -1. Both H^1 error and L^2 error achieve the optimal convergence order.

Example 2. We consider a function

$$d_1(x,y) = \begin{cases} \alpha(r^2 + r^3) & r < 0.5 \\ \alpha(-r^2 + r^3 + 0.5) & r >= 0.5 \end{cases} \qquad r = \sqrt{x^2+y^2}, \alpha = 100$$

with a jump of the gradient on $r = 0.5$. We use H^1 error as indicator. In each round of refinement, 30 percent elements with larger indicator are refined.

Figure (5.6) is the background mesh and the meshes obtained after 13 rounds of anisotropic and isotropic refinement, using piecewise linear finite element. The refined elements of the domain are located along the curve. Figure (5.7) shows the detailed mesh structure in $[0.16, 0.32] \times [0.36, 0.52]$.

The log-log plot of the error versus the degrees of freedom is in Figure (5.8). The asymptotic slope of the isotropic H^1 error curves(including linear and quadratic curves) is $-1/2$, which agrees with the optimal convergence order of the isotropic interpolation

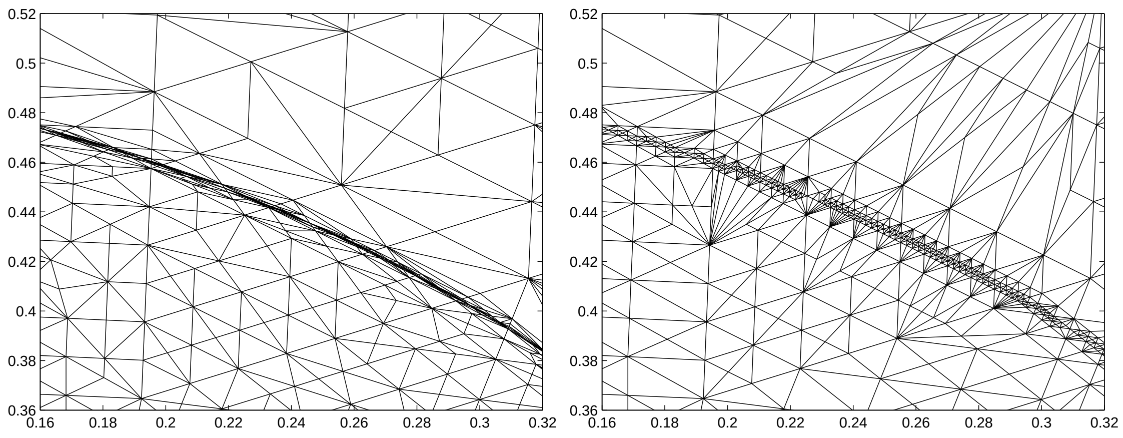

Figure 5.7: The detailed mesh structure for Example 2, in the square $[0.16, 0.32] \times [0.36, 0.52]$ for anisotropic (left) and isotropic (right) refinement.

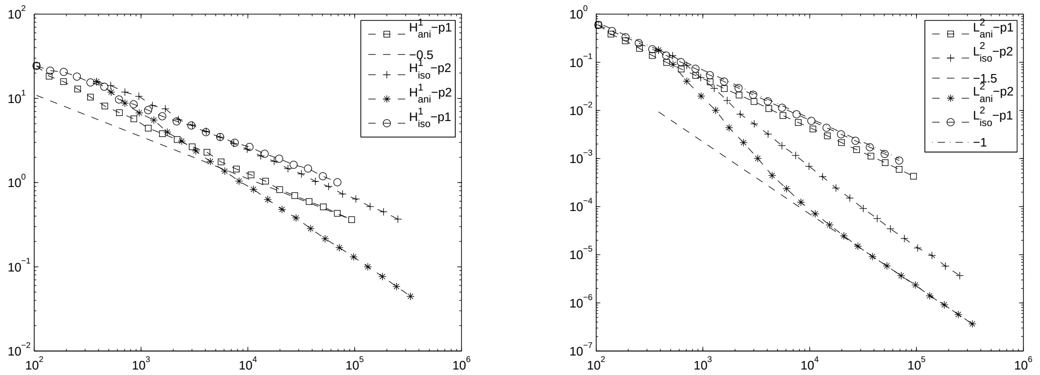

Figure 5.8: The H^1(left) and L^2(right) error in the #DOF for comparison of linear and quadratic interpolation of d_1 in Example 2 in log-log scale.

error for weakly discontinuous functions in 2D. The anisotropic H^1 curve using piecewise linear finite element first converges faster than the corresponding isotropic one, and when the dominant part of the error locates in the smooth domain(far from $r = 0.5$) where the optimal H^1 error asymptotic slope is $-1/2$, the curve tends to have the final asymptotic slope as $-1/2$. The asymptotic slope of the H^1 error curve using the quadratic anisotropic algorithm is about -0.85, higher than the isotropic one. For L^2 error, with the increasing of the #DOF, both the isotropic and anisotropic curves obtain the same asymptotic slope, i.e., -1 for linear interpolation and $-3/2$ for quadratic interpolation.

Example 3. We consider a 3D weakly discontinuous function

$$d_2(x, y, z) = \begin{cases} 10(-r+1) + e^{x+y+z} & r < 0.5 \\ 10r + e^{x+y+z} & r >= 0.5 \end{cases}, \quad r = \sqrt{x^2 + y^2 + z^2},$$

with the jump of the gradient on the sphere $r = 0.5$, to examine the numerical behavior of our algorithm in 3D space. The 3D background mesh is a quasi-uniform one generated

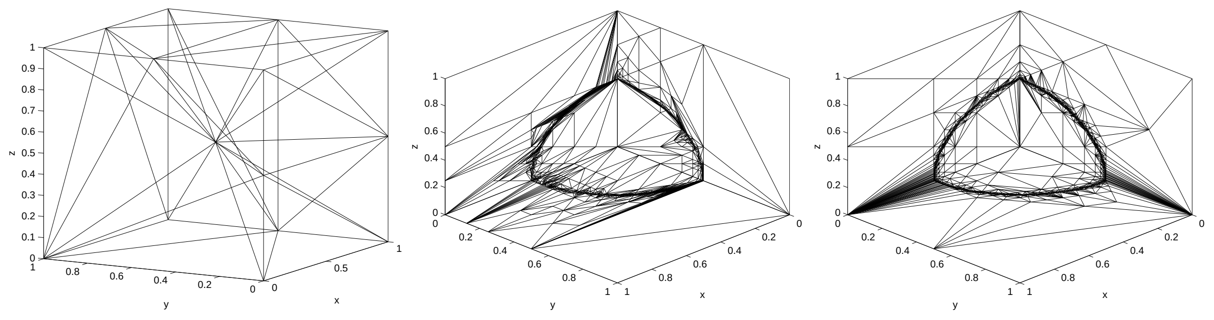
Figure 5.9: The background tetrahedral mesh (left), the mesh structure on the coordinate planes from diagonal view for anisotropic (middle) and isotropic (right) linear interpolation of Example 3.

by the 3D mesh generator Gmsh [13], as shown in the left picture in Figure (5.9). The typical element size in the background mesh is about 0.5. We use H^1 error as indicator. Each step, we refine the elements with indicator above $\sqrt{1.5}|d_2 - I_h d_2|_{H^1}$.

The middle picture in Figure (5.9) is the anisotropic mesh with 213253 elements and the right is the isotropic one with 368442 elements. The refined elements of the domain are located along the sphere. In Figure (5.10), we present part of the mesh on the coordinate plane yoz to show the quality of the generated mesh. It is observed that the 3D anisotropic refinement algorithm produces highly anisotropic elements stretching along the right direction.

The log-log plot of the error versus the #DOF is in Figure (5.11). The asymptotic slopes of the H^1 error curve using isotropic refinement for both linear and quadratic interpolation are $-1/4$. The slopes agree with the optimal convergence order of the isotropic interpolation of 3D weakly discontinuous functions. The linear/quadratic anisotropic H^1 error curve, which has an asymptotic slope as about $-1/3$ and -0.42, has a higher convergence order than the corresponding isotropic one. For L^2 error, the linear isotropic and anisotropic curves tend to have the same asymptotic slope of $-2/3$, while the quadratic anisotropic/isotropic error curve has the asymptotic slope of -0.8 and $-3/4$.

5.2 Elliptic interface problems

In this subsection, we solve the model equation of interface problems introduced in section 2. The examples are from [14]. We use the Zienkiewicz-Zhu (ZZ) [29] error estimator as the refinement indicator, which is based on the local reconstruction of the solution gradient. The ZZ error estimator works well for both anisotropic and isotropic elements in our numerical experiments. For a piecewise linear finite element approximation u_h on an element τ, the ZZ estimator of energy norm is

$$\eta_\tau^{ZZ} = \|\nabla u_h - \Pi_h \nabla u_h\|_{L^2(\tau)},$$

where $\Pi_h \nabla v$ is a local $L^2(\Omega)$ projection of ∇v onto V_h:

$$\Pi_h \left(\frac{\partial v}{\partial x}\right)(P) = \frac{1}{\sum_{K\in\mathcal{T}_h, P\in K} |K|} \sum_{T\in\mathcal{T}_h, P\in K} |K| \left(\frac{\partial v}{\partial x}\right)\Big|_K,$$

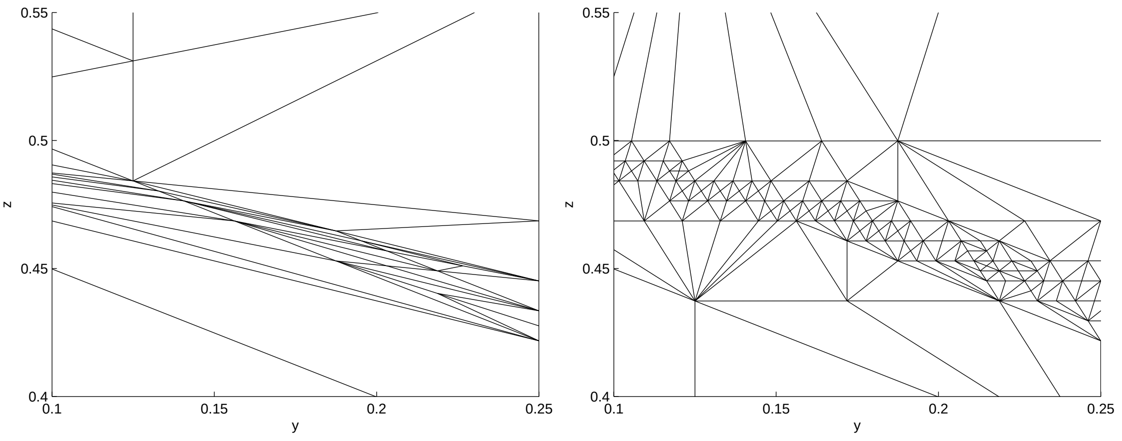

Figure 5.10: The mesh structure for Example 3, in $[0.1, 0.25] \times [0.4, 0.55]$ on the *yoz* plane, for linear anisotropic (left) refinement and isotropic (right) refinement.

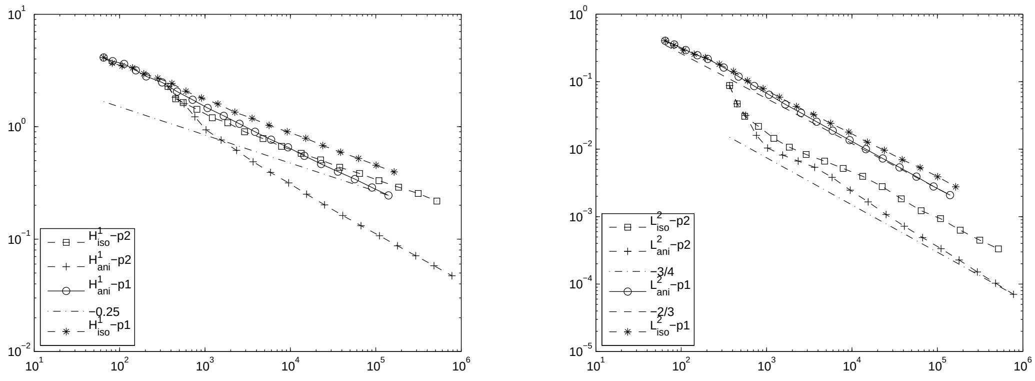

Figure 5.11: The H^1(left) and L^2(right) error in the #DOF for comparison of linear and quadratic interpolation of d_2 in Example 3 in log-log scale.

$$\Pi_h\left(\frac{\partial v}{\partial y}\right)(P)=\frac{1}{\sum_{K\in\mathcal{T}_h,P\in K}|K|}\sum_{T\in\mathcal{T}_h,P\in K}|K|\left(\frac{\partial v}{\partial y}\right)\Big|_K.$$

For the quadratic finite element approximation, similar patch recovery operation is used. The difference is to recover a piecewise quadratic gradient instead of a piecewise linear one. The gradient $\Pi_h\nabla u_h$ obtained by patch recovery is then used to substitute the gradient of the solution ∇u in $\nabla e=\nabla u-\nabla u_h$, thus the term $G_\tau(e)$ in the estimate is approximated.

The numerical convergence order of the elliptic interface problem could be different from the function interpolation, due to the linear regression phenomenon. For the L^2 error using the piecewise quadratic approximation, the numerical convergence order of the L^2 error degenerates to 2 on uniform meshes as pointed out in [14]. In the following examples, we observe the similar behavior on isotropic locally refined mesh, too. The asymptotic slope of the L^2 error/#DOF curve in the log-log scale is -1, instead of $-3/2$ as in the function interpolation. By using the anisotropic refinement algorithm, the degeneration of the convergence order can be remedied.

The background mesh generated by Easymesh [19] is quasi-uniform with typical element size 0.1, as shown in the middle picture in Figure (5.12).

Example 4 Homogeneous jump problem. Consider the problem (2.1) with the jumps $w=0$ and $Q=0$. We set $\Omega=[-1,1]\times[-1,1]$ and the interface Γ being the circle centered at point $(0,0)$ with radius $R=0.5$, and $\beta^-=1$, $\beta^+=100$ as the left picture in Figure(5.12). The source term $f(x,y)$ and the Dirichlet boundary condition g are calculated from the solution $u(x,y)$:

$$u(x,y)=\begin{cases}\dfrac{r^3}{\beta^-} & \text{if } r\le R,\\ \dfrac{r^3}{\beta^+}+(\dfrac{1}{\beta^-}-\dfrac{1}{\beta^+})R^3 & \text{othewise,}\end{cases}$$

where $r=\sqrt{x^2+y^2}$.

During the process of refinement, we use double of the average ZZ indicator on element as tolerance. The right picture in Figure (5.12) is the anisotropic mesh obtained after 8 rounds of refinement, using piecewise quadratic approximation. The preferred refinement edges are correctly chosen using the ZZ error estimator. We plot a small part of the mesh structure in Figure (5.13) for comparison of anisotropic and isotropic refinement.

The log-log plot of error versus the #DOF is in Figure (5.14). Due to the discontinuity in gradient, the quadratic isotropic H^1 error curve can only have the asymptotic slope as $-1/2$. But the asymptotic slope of the quadratic anisotropic H^1 error curve is -0.75, much higher than the isotropic case. Similarly, the corresponding anisotropic L^2 error curve asymptotically achieve the same optimal slope as $-3/2$, while the isotropic L^2 error curve only has the linear regression asymptotical slope of -1. It is interesting that though the ZZ error estimator is rigorously proved only on almost-uniform meshes, the use of ZZ error estimator as refinement indicator in this example does not adversely affect the quality of the generated meshes with increasing anisotropy.

Example 5 Non-homogeneous jump problem. Next, we consider the model problem (2.1) with $\Omega=[-1,1]\times[-1,1]$, and the interface being the zero level set of the function

$$\phi(x,y)=\sqrt{x^2+y^2}-r,$$

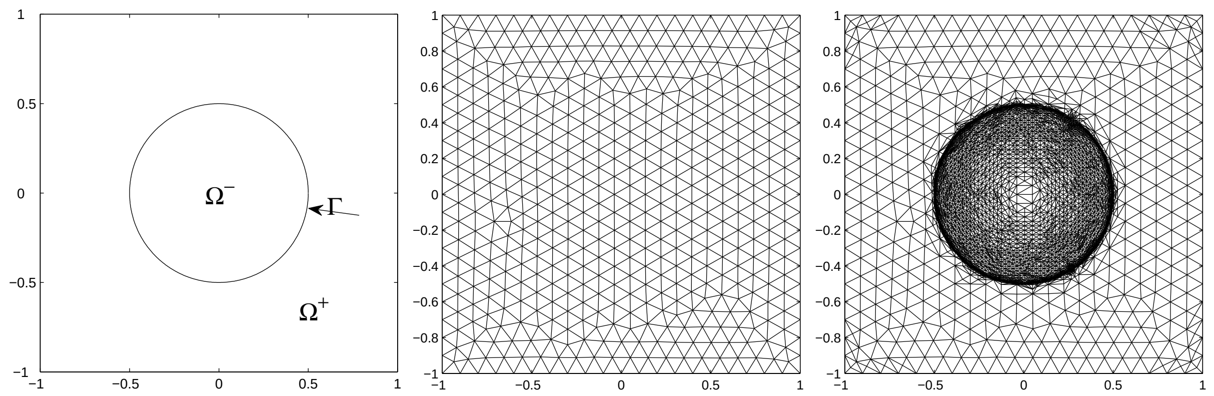

Figure 5.12: The domain and interface (left), the background mesh with 505 nodes (middle) and the anisotropic mesh (right) with 6130 nodes for Example 4.

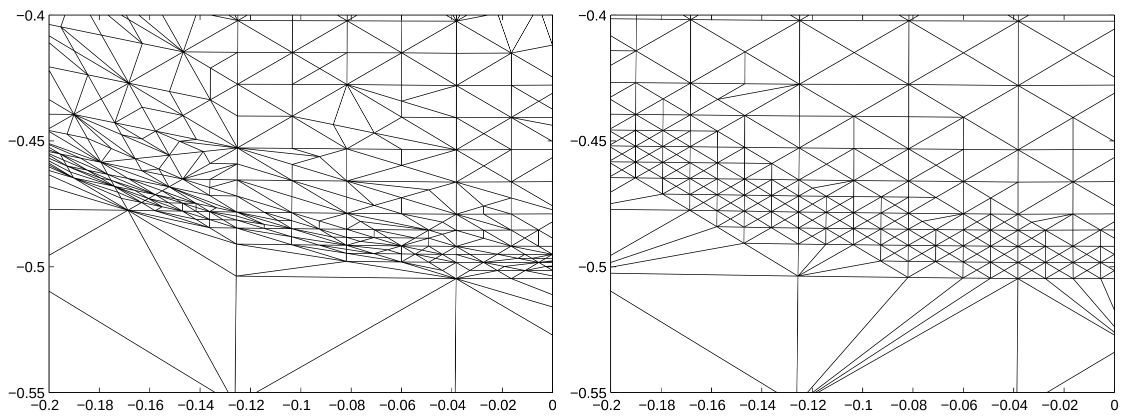

Figure 5.13: The anisotropic (left) and isotropic (right) mesh structures in $[-0.2, 0] \times [-0.55, -0.4]$ for Example 4, using ZZ error estimator.

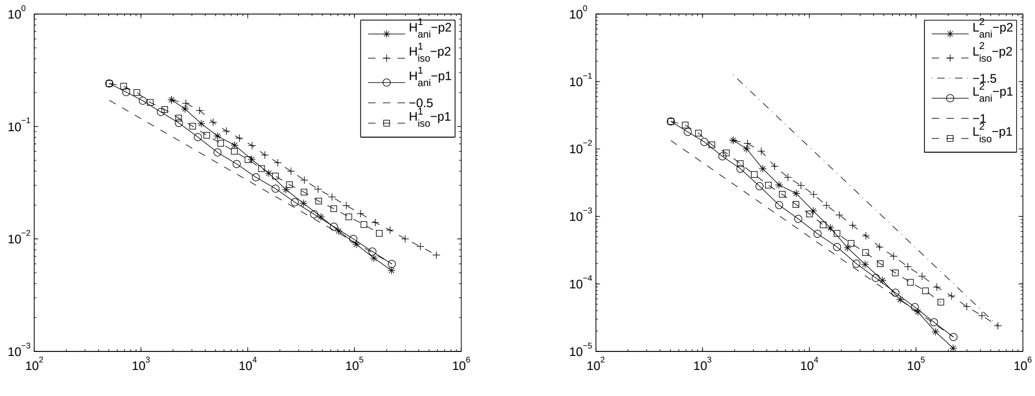

Figure 5.14: The H^1(left) and L^2(right) error in the #DOF for comparison of linear and quadratic approximation of Example 4 in log-log scale.

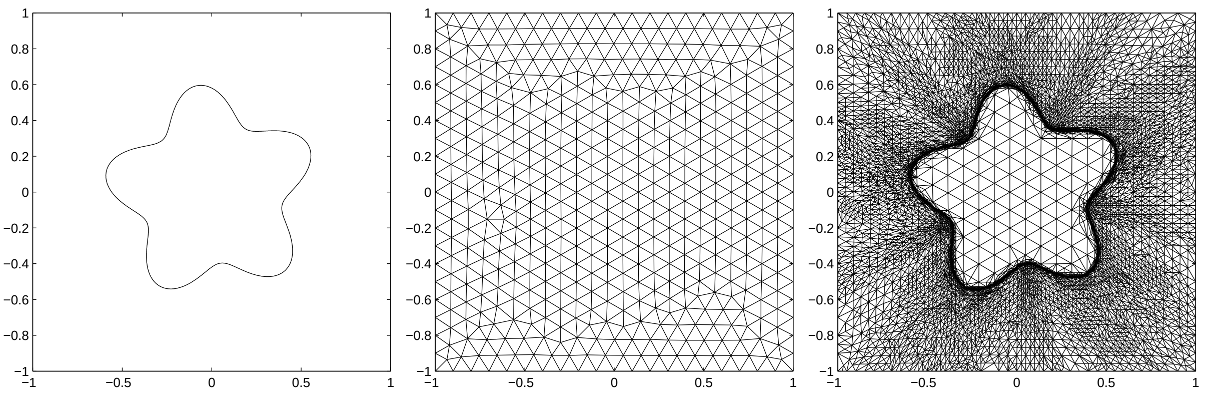
Figure 5.15: The interface and domain (left), the background mesh with 505 nodes (middle), and anisotropic mesh (right) with 6130 nodes for Example 5, using piecewise linear approximation.

where $r = 0.1\sin(5\theta - \pi/5) + 0.5$, $\tan\theta = y/x$ and $0 \le \theta < 2\pi$. The coefficient function β and right-hand side function f are

$$\beta(x,y) = \begin{cases} x^2+y^2+1 & \text{in } \Omega^-, \\ 0.1 & \text{in } \Omega^+, \end{cases} \qquad f(x,y) = \begin{cases} -4(2x^2+2y^2+1) & \text{in } \Omega^-, \\ 2\sin x\cos y & \text{in } \Omega^+. \end{cases}$$

The solution u is

$$u(x,y) = \begin{cases} x^2+y^2 & \text{in } \Omega^-, \\ 10(\sin x\cos y + \log\sqrt{x^2+y^2}) & \text{in } \Omega^+. \end{cases}$$

For simplicity, we choose $\hat{w}$ as

$$\hat{w}(x,y) = \begin{cases} 0 & \text{in } \Omega^-, \\ 10\sin x\cos y + 10\log r - r^2 & \text{in } \Omega^+. \end{cases}$$

During the process of refinement, the average ZZ indicator on element is chosen to be tolerance. In Figure (5.15), we plot the interface in the domain (left), the background mesh (middle) and the anisotropic mesh (right), using piecewise linear approximation. Figure (5.16) is the comparison of the detailed mesh structure using anisotropic and isotropic refinement algorithms.

In the log-log plot of the error versus the #DOF in Figure (5.17), the error behavior is similar as in Example 4. Both the isotropic H^1 error curves using piecewise linear and quadratic finite element tend to have the asymptotic slope of $-1/2$. The linear anisotropic H^1 error curve first converges faster and then tends to parallel with the isotropic one. The quadratic anisotropic H^1 error curve can have a higher asymptotic slope as -0.75. Again, for L^2 error convergence, the anisotropic linear/quadratic curves can asymptotically achieve the optimal slope as -1 and $-3/2$.

Example 6 Homogeneous jump problem in 3D. At last, we present a simple 3D interface problem with homogeneous jump. Let $\Omega = [0,1]\times[0,1]\times[0,1]$ and the interface Γ being the sphere centered at point $(0.5, 0.5, 0.5)$ with radius 0.25, and $\beta^- = 1/100$, $\beta^+ = 1$. The

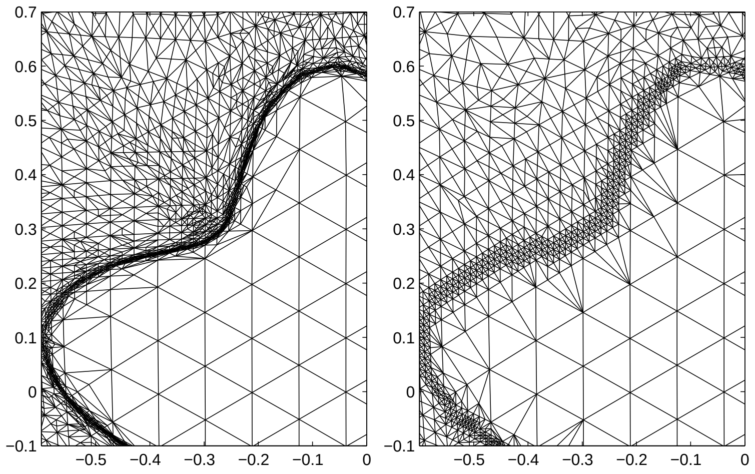

Figure 5.16: The anisotropic (left) and isotropic (right) mesh structures in $[-0.6, 0] \times [-0.1, 0.7]$ for Example 5.

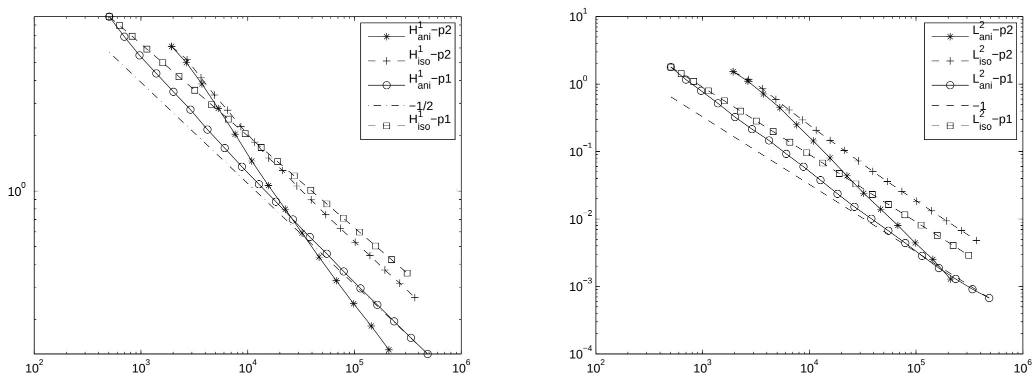

Figure 5.17: The H^1(left) and L^2(right) error in the #DOF for comparison of linear and quadratic approximation of Example 5 in log-log scale, using ZZ error estimator.

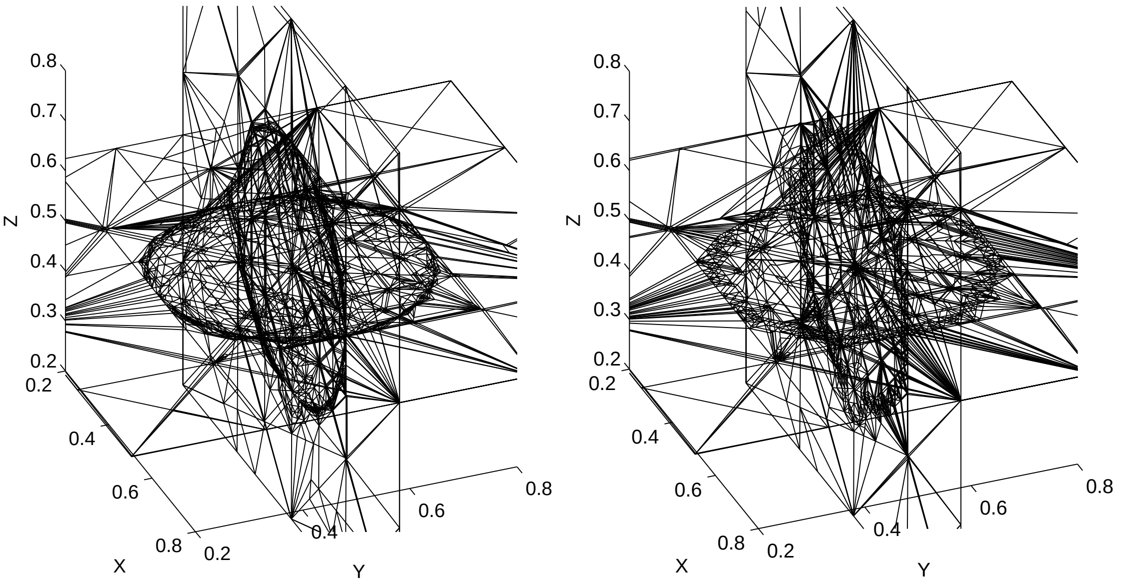

Figure 5.18: The mesh structure on two slices for anisotropic (left) and isotropic (right) linear interpolation of Example 6.

source term $f(x, y)$ and the Dirichlet boundary condition g are calculated from the exact solution

$$u(x,y,z) = \begin{cases} 100r^3 & r < 0.25 \\ r^3 + (100-1)/64 & r >= 0.25 \end{cases} \quad r = \sqrt{(x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2}.$$

The 3D background mesh with element size 0.5 is the same as the one in the left picture in Figure (5.9). We use ZZ estimate as indicator and the refinement tolerance is double of the average ZZ error.

Figure (5.18) is the local 3D mesh structure in horizontal and vertical slices, the left picture is the anisotropic mesh with 274451 elements and the right is the isotropic one with 304278 elements. The refined elements of the domain are located along the sphere. In Figure (5.19), we present local mesh on plane $z = 0.5$, and clearly the anisotropic mesh fits the solution better.

The log-log plot of the error versus the #DOF is in Figure (5.20). The asymptotic slopes of the H^1 error curve for both linear and quadratic isotropic refinement are $-1/4$. The linear/quadratic anisotropic H^1 error curve, which has an asymptotic slope as about -0.3 and -0.4, has a higher convergence order than the corresponding isotropic one. For L^2 error, the linear and quadratic isotropic curves tend to have the same asymptotic slope of $-1/2$, while the linear/quadratic anisotropic curve has the asymptotic slope of $-2/3$ and -0.85.

6 Concluding Remarks

In this paper, a new edge-based anisotropic mesh refinement methodology on unstructured meshes for elliptic interface problems has been developed. The preferred refinement edge is

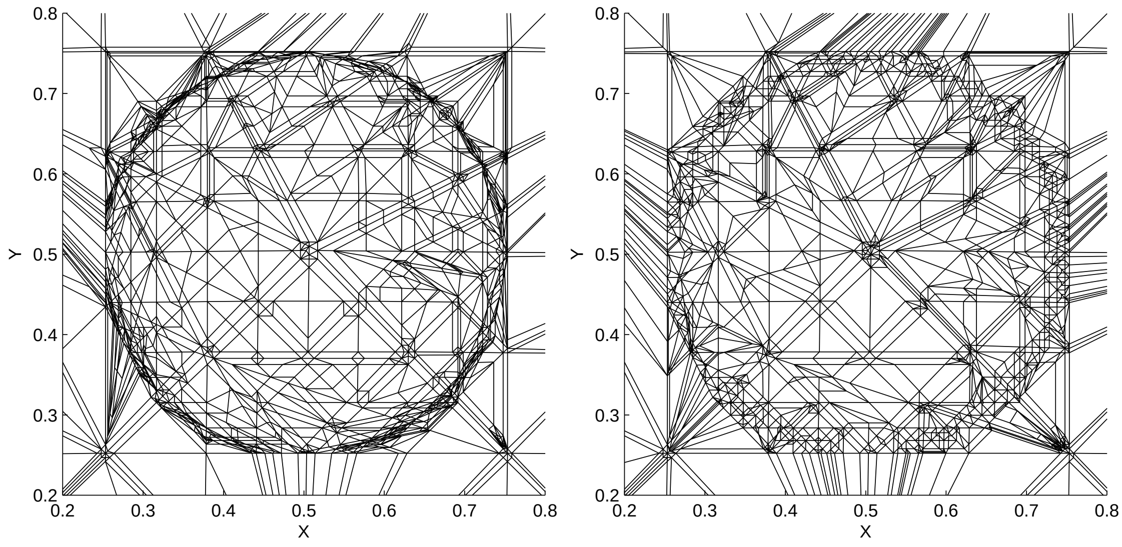

Figure 5.19: The local mesh structure on $z = 0.5$ in the domain $[0.2, 0.8]\times[0.2, 0.8]$, for anisotropic (left) and isotropic (right) linear interpolation of Example 6.

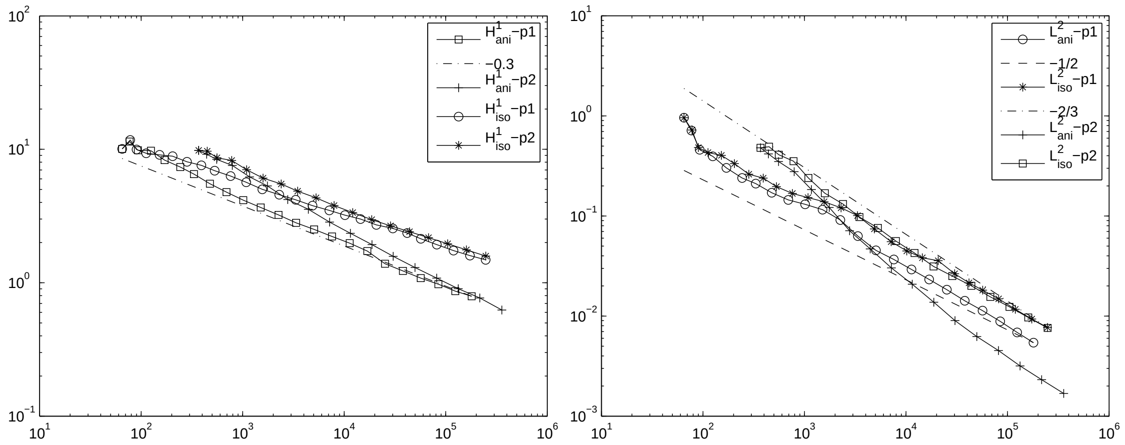

Figure 5.20: The H^1(left) and L^2(right) error in the #DOF for comparison of linear and quadratic approximation of Example 6 in log-log scale, using ZZ error estimator.

obtained by comparing the projection of the error gradient onto the edges. The algorithm can be valid with information of only first order derivatives. Numerical experiments for the interpolation of functions with different types of anisotropic singularities and the solution of interface problems have illustrated the effectiveness of our method.

Acknowledgements

The research was supported by the National Basic Research Program under the Grant 2005CB321701, and the National Science Foundation of China under the grant 10771008 and 10771211. Moreover, the research of Li was partial supported by A Foundation for the Author of National Excellent Doctoral Dissertation of PRC.

References

[1] T. Apel. *Anisotropic finite elements: local estimates and applications.* Stuttgart, Teubner, 1999.

[2] T. Apel, S. Grosman, P. K. Jimack, and A. Meyer. A new methodology for anisotropic mesh refinement based upon error gradients. *Appl. Numer. Math.*, 50:329–341, 2004.

[3] E. F. D' Azevedo and R. B. Simpson. On optimal triangle meshes for minimizing the gradient error. *Numer. Math.*, 59(1):321–348, 1991.

[4] R. E. Bank and R. K. Simth. Mesh smoothing using a posteriori estimates. *SIAM J. Numer. Anal.*, 34:979–997, 1997.

[5] E. Bänsch, F. Haußer, O. Lakkis, B. Li, and A. Voigt. Finite element method for epitaxial growth with attachment-detachment kinetics. *J. Comput. Phys.*, 194:409–434, 2004.

[6] R. E. Caflisch and B. Li. Analysis of island dynamics in epitaxial growth of thin films. *Multiscale Model. and Anal.*, 1:150–171, 2003.

[7] W. M. Cao. On the error of linear interpolation and the orientation, aspect ratio, and internal angles of a triangle. *SIAM J. Numer. Anal.*, 43(1):19–40, 2005.

[8] W. M. Cao. Anisotropic measures of third order derivatives and the quadratic interpolation error on triangular elements. *SIAM J. Sci. Comput.*, 29(2):756–781, 2007.

[9] L. Chen, P. T. Sun, and J. C. Xu. Optimal anisotropic meshes for minimizing interpolation errors in L^p-norm. *Math. Comput.*, 76(257):179–204, 2006.

[10] P. G. Ciarlet. *The finite element method for elliptic problems.* Elsevier, 1978.

[11] L. Formaggia and S. Pertto. New anisotropic a priori error estimates. *Numer. Math.*, 89(4):641–667, 2001.

[12] L. Formaggia and S. Pertto. Anisotropic error estimates for elliptic problems. *Numer. Math.*, 94:67–92, 2003.

[13] C. Geuzaine and J. Remacle. `http://www.geuz.org/gmsh/`.

[14] Y. Gong, B. Li, and Z. L. Li. Immersed-interface finite-element methods for elliptic interface problems with nonhomogeneous jump conditions. *SIAM J. Numer. Anal.*, 46:472–495, 2007.

[15] J. Goodman, K. Samuelsson, and A. Szepessy. Anisotropic refinement algorithms for finite elements. `http://citeseer.ist.psu.edu/goodman96anisotropic.html`, 1996.

[16] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M. -G. Vallet. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles. *Internat. J. Numer. Methods Fluids*, 32:725–744, 2000.

[17] R. Li and W. B. Liu. `http://circus.math.pku.edu.cn/AFEPack`.

[18] E. J. Nadler. *Piecewise interpolation on triangulations of a planar region.* PhD thesis, Division of Applied Mathematics, Brown University, Providence, 1985.

[19] B. Niceno. `http://www-dinma.univ.trieste.it/nirftc/research/easymesh/`.

[20] M. Picasso. An anisotropic error indicator based on Zienkiewicz-Zhu error estimator. *SIAM J. Sci. Comput.*, 24:1328–1355, 2003.

[21] W. Rachowicz. An anisotropic h-adaptive finite element method for compressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 146:231–252, 1997.

[22] S. Rippa. Long and thin triangles can be good for linear interpolation. *SIAM J. Numer. Anal.*, 29:257–270, 1992.

[23] M. C. Rivara. Algorithms for refining triangular grids suitable for adaptive and multigrid techniques. *Int. J. Numer. Meth. Engrg.*, 20:745–756, 1984.

[24] M. C. Rivara. Design and data structure of fully adaptive, multigrid, finite element software. *ACM Trans. Math. Software*, 10:242–264, 1984.

[25] L. R. Scott and S. Zhang. Finite element interpolation of non-smooth functions satisfying boundary conditions. *Math. Comput.*, 54(190):483–493, 1990.

[26] T. Skalicky and H. G. Roos. Anisotropic mesh refinement for problems with internal and boundary layers. *Internat. J. Numer. Methods Engrg.*, 46:1933–1953, 1999.

[27] J. C. Xu and Z. M. Zhang. Analysis of recovery type a posteriori error estimators for mildly structured grids. *Math. Comput.*, 73:1139–1152, 2004.

[28] O. C. Zienkiewicz and J. Wu. Automatic directional refinement in adaptive analysis of compressible flows. *Int. J. Numer. Methods Engrg.*, 37:2189–2210, 1994.

[29] O. C. Zienkiewicz and J. Z. Zhu. A simple error estimator and adaptive procedure for practical engineering analysis. *Int. J. Numer. Methods Engrg.*, 24:337–357, 1987.