ADAPTIVE MESH REFINEMENT AND SUPERCONVERGENCE FOR TWO DIMENSIONAL INTERFACE PROBLEMS

HUAYI WEI*, LONG CHEN†, YUNQING HUANG*, AND BIN ZHENG‡

Abstract. Adaptive mesh refinement and the Börgers' algorithm are combined to generate a body-fitted mesh which can resolve the interface with fine geometric details. Standard linear finite element method based on such body-fitted meshes is applied to the elliptic interface problem and proven to be superclose to the linear interpolation of the exact solution. Based on this superconvergence result, a maximal norm error estimate of order $O(h^{1.5})$ is obtained without using the discrete maximum principle. The data structure and meshing algorithms, including local refinement and coarsening, are very simple. In particular, no tree structure is needed. An efficient solver for solving the resulting linear algebraic systems is also developed and shown be robust with respect to both the problem size and the jump of the diffusion coefficients.

Key words. body-fitted mesh, adaptive mesh refinement, superconvergence

AMS subject classifications. 65N15, 65N30, 65N50, 65N55

1. Introduction. Elliptic interface problems arise in many physical applications including fluid dynamics, material science and so on [23]. Let Ω be a bounded open domain in $\mathbb{R}^2$ and Γ a continuous simple closed curve imbedded in Ω. The curve Γ separates the domain Ω into two disjoint regions Ω^+ and Ω^-, and by convention Ω^- denotes the interior domain enclosed by Γ. A typical elliptic interface equation is

$$-\nabla\cdot(\beta(x)\nabla u(x)) = f(x),\ x\in\Omega\backslash\Gamma, \tag{1.1}$$

with prescribed jump conditions across the interface Γ:

$$[u]_\Gamma = u^+ - u^- = q_0, \quad [\beta u_n]_\Gamma = \beta^+ u_n^+ - \beta^- u_n^- = q_1, \tag{1.2}$$

and Dirichlet or Neumann boundary condition on $\partial\Omega$. Here u_n denotes the normal flux $(\nabla u)\cdot n$ with n being the unit outward (from Ω^- to Ω^+) normal vector of the interface Γ. The superscripts $+$ and $-$ stand for the restrictions of a function on Ω^+ and Ω^-, respectively. The diffusion coefficient $\beta(x)$ is assumed to be uniformly positive and smooth on each subdomain, but may be discontinuous across the interface. The discontinuity of the solution and/or the flux posed in (1.2) makes numerical methods for interface problems a challenging task.

*Hunan Key Laboratory for Computation and Simulation in Science and Engineering, Key Laboratory of Inteligent Computing & Information Processing Computing of Ministry Of Education, School of Mathematics and Computational Science, Xiangtan Uinversity Xiangtan 411105, Hunan, People's Republic of China (huayiwei1984@gmail.com, huangyq@xtu.edu.cn). The first author was supported by NSFC (Grant No. 11301449), in part by Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20134301120003), and in part by 2010-2011 UC Irvine Academic Senate Council on Research, Computing and Libraries (CORCL) award. The third author was supported by NSFC (Grant No.11031006), Program for Changjiang Scholars and Innovative Research Team in University (Grant No. IRT1179), International Science and Technology Cooperation Program of China (Grant No. 2010DFR00700).

†Department of Mathematics, University of California at Irvine, Irvine, CA 92697 (chenlong@math.uci.edu). The second author was supported by National Science Foundation (NSF) (DMS-0811272 and DMS-1115961), in part by 2009-2011 UC Irvine Academic Senate Council on Research, Computing and Libraries (CORCL) award, and in part by Department of Energy prime award # DE-SC0006903.

‡Pacific Northwest National Lab. P.O. Box 999 Richland, WA 99352(binzhengmath@gmail.com). The fourth author was supported in part by National Science Foundation (NSF) DMS-0811272.

Existing numerical methods for interface problems can be roughly classified into two categories by using either a Cartesian-type mesh or a body-fitted mesh in the discretization of the domain. Methods based on Cartesian meshes use Cartesian meshes of the whole domain Ω and modify the finite difference stencils or finite element basis functions on the vertices near the interface in order to deal with the jump conditions; see [15, 23, 24] and references therein. The main advantage of using Cartesian-type mesh is that the mesh generation is simple. In contrast, on body-fitted meshes, one can use the standard discretization while changing the mesh such that the grid points fitted to the interface. Therefore a crucial ingredient of this method is to have a simple, robust and fast mesh generator, which is the topic of this study.

We are interested in the semi-structured and body-fitted triangular mesh generation methods [4, 5, 26], and in particular the Börgers' algorithm [5]. The basic idea of the Börgers' algorithm is to use a Cartesian grid but perturb only the grid points near the interface onto the interface and choose an appropriate diagonal of perturbed quadrilaterals to fit the interface and maintain the mesh quality. The final body-fitted mesh is shape regular and topologically equivalent to the Cartesian grid. This topological structure will be beneficial in the numerical computation, e.g., on designing multigrid type solvers. Application of Börgers' algorithm to elliptic interface problems can be found in [30, 31]. For general unstructured body-fitted triangular mesh generation methods, the readers are referred to Triangle [28] and Distmesh [25].

When the interface contains fine geometric features, methods based on a Cartesian mesh or its perturbation, require a very fine mesh to resolve the interface and in turn increase the computational cost. The cost can be dramatically reduced by applying adaptive mesh refinement (AMR) near the interface. We should distinguish two types of mesh adaptivity: one is to resolve the geometry of the interface and the other is to resolve the singularity of the solution [13]. Here we focus on the former since usually the singularity occurs near the interface and resolving the interface is enough to resolve the singularity. Even if further singularity could exist in other locations, the standard adaptive mesh refinement based on a posteriori error estimator can be applied once the initial mesh resolves the interface.

In this study, we combine the adaptive mesh refinement and the Börgers' algorithm to generate a body-fitted mesh which resolves the interface with fine geometric details while maintaining the hierarchical structure. Since the feature of a curve can be characterized by its large curvature. Our mesh refinement will be guided by a discrete curvature estimator. Interface elements, which are triangles across the interface, will be refined if the estimated curvature is big. The refinement stops until no such elements. Then Börgers' algorithm [5] is applied to adjust the mesh points near the interface to get a body-fitted mesh. Finally edge swapping and mesh smoothing technique is applied to further improve the quality of the mesh. The obtained body-fitted mesh will be used a coarse grid and finite element approximation of the interface problem will be based on a sequence of meshes based on the uniform refinement of this coarse mesh. The new vertices on the interface edges introduced in each refinement will be projected onto the interface.

An important feature of our approach is that the data structure and meshing algorithms, including local refinement and coarsening, are simple and fast. In particular, no tree structure is needed, and only standard data structure for the Finite Element Methods (FEMs) is used. The hierarchical structure of the mesh is implicitly stored in the ordering of triangles. This is in contrast to other adaptive methods

where sophisticated quad-tree structure is used [34]. In addition, unlike the quad-tree grids, the generated mesh is conforming without hanging vertices that need special treatment.

The implicitly built-in hierarchical structure can be used to construct a multigrid-type preconditioner. Together with the Preconditioned Conjugate Gradient (PCG) method, the linear algebraic systems resulting from the finite element discretization on our body-fitted meshes can be solved very efficiently. Numerical tests show that the solver is robust with respect to both the problem size and the jump of the diffusion coefficients.

For moving interface problems, the coarsening algorithm is more important. After the interface is moved, we can efficiently adjust the mesh to fit the new interface by the combination of the coarsening, refinement and Börgers' algorithm. In this paper, we focus on the stationary elliptic equations and leave the generalization to moving interface problems in a forthcoming work.

Besides the numerical experiments, we provide a superconvergence analysis for the methods we developed. We show that the linear finite element solution of (1.1) is superclose to the nodal interpolation of the true solution under some practically assumption on the meshes. As a consequence we obtain an $O(h^{1.5})$ error estimate in the maximal norm. Note that although optimal order of convergence in the energy norm is well known [32, 14], the maximal norm error analysis without using discrete maximum principle, which is difficult to impose in practice [22], is new to two dimensional interface problems.

The rest of the paper is organized as follows. In Section 2, we present an adaptive mesh algorithm for the generation of body-fitted meshes. In Section 3, we introduce the finite element formulation of (1.1), and discuss the superclose and maximal norm estimate. In Section 4, we provide two numerical experiments to show the effectiveness of our method. Finally, we give several concluding remarks in Section 5.

2. A Mesh Generation Algorithm. In this section, we present a mesh generation algorithm to generate a body-fitted triangular mesh for a given smooth interface. We begin with discussion on the structure and quality of meshes and then introduce our semi-structured mesh generation algorithm. We explain details of our algorithm through a concrete example.

2.1. Overview. Structured meshes, for example, the Cartesian grid or topologically tensor product grids, can use a simple index pointer. Physical location of each vertex and function values at each vertex can be addressed by index (i, j) in two dimensions or (i, j, k) in three dimensions. This structure enable simplification in the implementation of numerical methods. Furthermore the natural hierarchical structure of tensor product meshes is ideal for multigrid solvers. Unlike structured grids, unstructured grids require a list of the connectivity which specifies the way of a given set of vertices making up individual elements. The indexing is more complicated and the hierarchical structure is usually unavailable.

The mesh quality measures the shape of elements. For triangles, usually the aspect ratio of the diameter over the inscribed radius is used. In two dimensions, this is equivalent to the minimal angle. Among various mesh qualities [20], in this paper we use the minimal angle. The mesh quality is important to control the interpolation error in H^1 norm [1] and the condition number of the stiffness matrix [18].

There are a number of body-fitted triangular mesh generation methods, different in the trade-off of the mesh structure and the mesh quality. Scarifying the structure, one can get body-fitted meshes with high quality. A popular approach is through

Delaunay-refinement and/or mesh smoothing. One noticeable example is the software package Triangle [28] which can generate an unstructured mesh fitted to given polygon curves with a guaranteed minimal angles $\geq 30^\circ$. Another approach is through mesh optimization. For example, DistMesh [25], can usually produce body-fitted meshes with high quality triangles.

In our application, the hierarchical structure is more important since efficient multigrid preconditioner can be developed and coarsening is possible which fascinate the re-meshing for moving interface problems. Therefore we would like to maintain a hierarchical structure. We still use unstructured grid indexing while record the hierarchical structure implicitly by the ordering of the triangles and vertices using the ideas in [12]. We call such grids semi-structure.

One semi-structured and shape regular meshing algorithm is proposed by Börgers [5] for Cartesian grids. He showed that one can generate a quasi-uniform and shape regular grid from a Cartesian grid by perturbing some nearby grid points to the interface and connecting suitable diagonals of the quadrilaterals. The tensor product structure of the Cartesian grid is thus preserved. The accuracy of the approximation of the interface Γ is $\mathcal{O}(h^2)$ for $\Gamma \in \mathcal{C}^2$; see Theorem 3 in [5]. If the initial Cartesian grid is uniform, the mesh generated by Börgers algorithm has a theoretical lower bound 18.4349°.

In this study, we combine the adaptive mesh refinement and the Börgers' algorithm to generate a body-fitted mesh which resolves the interface with fine geometric details while maintains the hierarchical structure. Our mesh refinement is guided by a discrete curvature estimator. If we begin with a mesh consisting of isosceles right triangles, the final mesh generated by our algorithm also has a lower bound 18.4349°. In practice, the minimum angle is usually slightly larger than this theoretical bound.

In our algorithm, we store the coordinates of vertices in *node* array and the topological connectivity of vertices in *elem* array, which is the minimal information required for any finite element implementation. Instead of using the binary refinement tree explicitly, the hierarchical structure is implicitly stored in a special ordering of the triangles and vertices; see [12] for details.

One closely related mesh generation algorithm is AMR through quad-tree grids which recursively subdivides rectangles into four smaller rectangles with equal size. Once the mesh resolves the fine details of the geometry, vertex perturbation is applied to fit the interface. Further post-processing is needed to improve the skew triangles near the boundary or interface; see [17, 33] and references therein.

There are some drawbacks of the quad-tree method which limit its application [34]. First, the data structure of quad-trees is more sophisticated than ours in terms of storing and retrieving mesh information. Furthermore, it suffers from the existence of hanging nodes and is relatively poor in fitting the boundary. As a consequence, mesh generation algorithm based on quad-tree needs post-processing in order to obtain a mesh with good quality; see [16, 17].

2.2. Algorithm. For simplicity, we assume that the domain Ω is a square with four vertices p_1, p_2, p_3, and p_4 shown in Fig. 2.1 (a). We also assume that an initial mesh $\mathcal{T}_0$ of Ω consisting of isosceles right triangles is given. The vertices of each triangle are listed in counter-clockwise order and the first vertex of each triangle is opposite to its longest edge. If Ω is the unit square, the *node* and *elem* arrays of $\mathcal{T}_0$ in Fig. 2.1(a) are

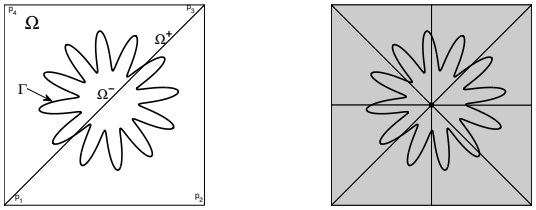
node = [0 0; 1 0; 1 1; 0 1]; elem = [2 3 1; 4 1 3];

We label the first vertex of each triangle of $\mathcal{T}_0$ as the newest vertex, and use the

newest vertex bisection method. The rule of newest vertex bisection includes:

(1) a triangle is bisected to two new children triangles by connecting the newest vertex to the midpoint of the opposite edge;
(2) the new vertex created at the midpoint of a bisected edge is assigned to be the newest vertex of the children.

Once the labeling is done for an initial triangulation, the decent triangulations inherit the label by the rule (2) such that the bisection process can proceed. Note that when triangles are all isosceles, the newest vertex bisection coincides with the longest edge bisection. By storing the first vertex of triangles as the newest vertex, however, there is no need to compute the edge length. A simple implementation of newest vertex bisection method in MATLAB can be found in *i*FEM [9].



(a) (b)

Fig. 2.1: The initial mesh and the mesh obtained by bisection twice. (a) The initial mesh $\mathcal{T}_0$. (b) Uniformly bisect $\mathcal{T}_0$ twice.

Let Γ be a smooth interface imbedded in Ω. Assume Γ can be represented by the zero-level set of a function $\phi(x)$, i.e., $\Gamma = \{x \in \Omega : \phi(x) = 0\}$. The interface Γ separates Ω into two subdomains $\Omega^+ := \{x \in \Omega : \phi(x) > 0\}$ and $\Omega^- := \{x \in \Omega : \phi(x) < 0\}$. Given a triangular mesh $\mathcal{T}$ of Ω, a vertex p is said to be *inside* if $\phi(p) < 0$, *outside* if $\phi(p) > 0$, or *on* Γ if $\phi(p) = 0$; an element $\tau \in \mathcal{T}$ is called an *interface element* if the function values of ϕ at its three vertices have at least two different signs. All vertices of interface elements are called *interface vertices*.

During the refinement process, there will appear two types of elements in the mesh. An element is said to be of *Type A* if its two legs are parallel to either x-axis or y-axis; see Fig. 2.6. A *Type B* element is one such that its hypotenuse is parallel to x-axis or y-axis; see Fig. 2.7. If all elements containing a vertex p are of Type A, their legs locally form a Cartesian grid around p for which Börgers' algorithm [5] can apply. Therefore the type of elements will be carefully controlled in our algorithm. Note that one bisection will change the type of the bisected element.

Our algorithm is described as follows:

ALGORITHM 2.1. $\mathcal{T} = \text{interface}(\mathcal{T}_0, \phi)$
Input: $\mathcal{T}_0$ an initial mesh of Ω; ϕ the level set function describing Γ.
Output: $\mathcal{T}$ a body-fitted mesh.
Step 1 Uniformly bisect $\mathcal{T}_0$ till there exists at least one inside vertex.
Step 2 Bisect interface elements with large curvature estimator.
Step 3 Update the type of interface elements.
Step 4 Move nearby interface vertices to Γ.
Step 5 Apply edge swapping and mesh smoothing.

2.3. An example. We describe Alg. 2.1 in detail by the following example. Consider the square domain $\Omega = (-1, 1)^2$ and an interface Γ given in the parametrized

form:

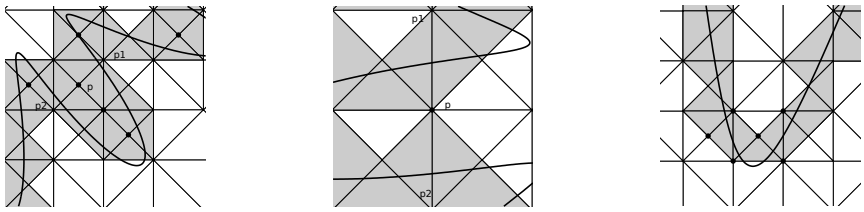
$$X(\theta) = 0.02\sqrt{5} + r(\theta)\cos(\theta), Y(\theta) = 0.02\sqrt{5} + r(\theta)\cos(\theta), r(\theta) = 0.5 + 0.2\sin(12\theta),$$

for $0 \le \theta \le 2\pi$; see Fig. 2.1(a). Given a point $x_0 = (X, Y) \in \Omega$, the level set of Γ is:

$$\phi(x_0) = r_0^2 - r(\theta_0)^2,$$

where (r_0, θ_0) is the polar coordinates of x_0.

Step 1. Uniformly bisect $\mathcal{T}_0$ several times to get a finer mesh with at least one vertex inside Γ; see Fig. 2.1(b). Find all interface elements by checking the sign of the level set functions evaluated on each vertex; see the elements with gray color in Fig. 2.1(b).



(a) (b) (c)

Fig. 2.2: Two cases of interface vertices with large curvature estimator. In (a) and (b), for an interface vertex p not on the interface, there exist other two interface vertices p_1 and p_2, which are neighbors of p, such that p_1, p, p_2 are collinear and the line segment (p_1, p_2) is crossed twice by the interface Γ ; In (c) the interface elements form a "belt" domain whose "width" is only one interface element and boundaries are two poly-lines along the interface.

Step 2. The accuracy of the poly-line approximation of a curve is usually controlled by its curvature and the local edge length. Typically, more mesh vertices (and thus shorter edge length) are needed at places with big curvature in order to have a better approximation. In this step, we will repeat the following loop until no elements marked for refinement:

$$\text{ESTIMATE} \to \text{MARK} \to \text{REFINE}. \tag{2.1}$$

For every interface vertex p, we estimate the curvature of Γ near p, mark interface elements which contains a interface vertex with large curvature estimator, and refine these elements.

ESTIMATE. We estimate the curvature at an interface vertex p by the configuration and angles of the interface elements containing p, and construct a curvature estimator η_p.

There are two possible configurations of interface vertex. For the first case, let p be a interface vertex which is not on Γ. If there exist other two interface vertices p_1 and p_2 which are neighbors of p(namely (p_1, p) and (p_2, p) are two edges in current mesh), furthermore p_1, p, p_2 are collinear and the line segment (p_1, p_2) intersects the interface Γ twice, then we set $\eta_p = \infty$, see the vertices marked with black dots in Fig. 2.2 (a) and (b). Because this case indicates that the interface is folded in some place (not necessarily near p) which make two parts of the interface near p very close, so we need to bisect the elements containing p to separate this two parts. Through local mesh refinement, the interface vertices of the first case will disappear, since the

distance between p and Γ is fixed and the local element size around p will discrease when local refinement is applied.

In the second case, the interface elements form a "belt" domain whose "width" is only one interface element and boundaries are two poly-lines along the interface; see Fig. 2.2 (c). And the trend of such a "belt" domain should reflect the trend of the interface. Consequently the discrete curvature of its two poly-line boundaries should reflect the curvature of the interface. So we introduce our curvature estimator based on the discrete curvature of these two poly-line boundaries. Here the discrete curvature κ_p at a vertex p is defined as the change of the angles of two neighboring line segments following the positive direction of the poly-line; see Fig. 2.3. The curvature estimator η_p at p is defined as:

$$\eta_p := \left| \sum_{q \in \mathcal{J}(p)} \kappa_q \right|,$$

where $\mathcal{J}(p)$ is the neighbor interface vertex set of p. Here the sum introduces some average and smoothing effect and make the estimator to reflect the varying of the real curvature better.

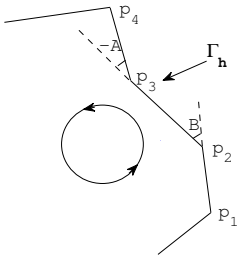


Fig. 2.3: The discrete curvature of poly-line Γ_h at its vertex. Counter clockwise is the positive direction. From line segment (p_1, p_2) to line segment (p_2, p_3), Γ_h turns left and the discrete curvature of Γ_h at p_2 is B degree; from line segment (p_2, p_3) to line segment (p_3, p_4), Γ_h turns right and the discrete curvature of Γ_h at p_3 is $-A$ degree.

MARK. Mark elements which contain an interface vertex with $\eta_p \geq 180°$.

REFINE. If there exist marked elements, bisect all marked elements with possible more neighboring elements to get a conforming triangulation; else step 2 terminates.

Let us consider a special case –the interface is a straight line–to show why Step 2 can terminate. See Fig. 2.4, for a straight line, all interface elements form a "belt" domain whose "width" is only one interface element and boundaries form two poly-lines along the straight line, and these two poly-line boundaries neither cross over nor be far from the straight line. Under such case, one can show that the discrete curvature estimator of every interface vertex must be less than 180°. If not, the poly-lines must cross over or depart from the straight line.

For general interface, under small enough local mesh size, the interface locally can be thought as a straight line. So the bisection process based on the discrete curvature estimator η_p will terminate when the local mesh size is small enough. For the mesh and its local detail, see Fig. 2.5.

Step 3. For an interface vertex p if all interface elements containing p are of Type A, the legs of these elements locally form a Cartesian grid, see Fig. 2.6, so

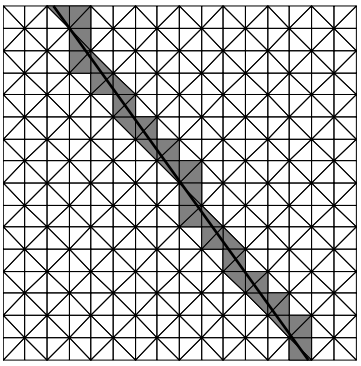
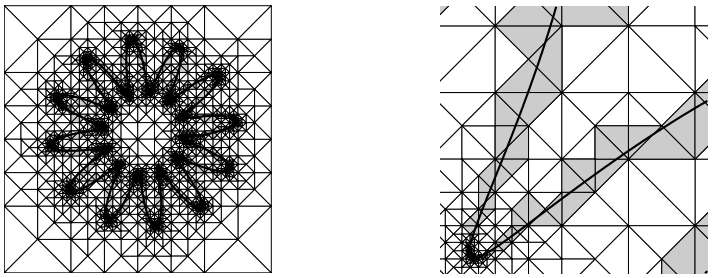


Fig. 2.4: . The interface Γ is a straight line. The gray elements are the interface elements.



(a) (b)

Fig. 2.5: The mesh (a) and its local details (b) after step 2.

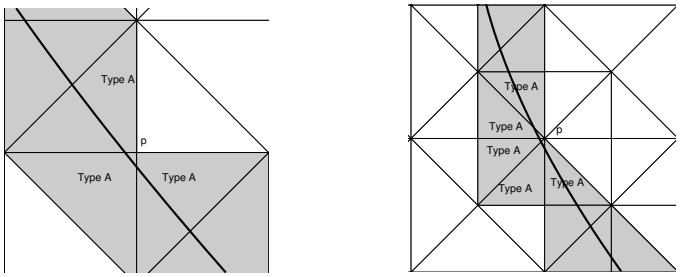


Fig. 2.6: Examples of Type A interface elements. The legs of these elements locally form a uniform Cartesian grid.

that Börgers' algorithm can be applied. For Type B interface elements it can be transformed into Type A by one bisection. However, there is no need to transform all Type B interface elements to Type A. In fact, as a property of locally adaptive meshes, the type difference between neighboring elements is necessary for having varying mesh size. We shall just bisect some Type B interface elements and keep others in the mesh.

Two triangles are called neighbors of each other if they share a common edge.

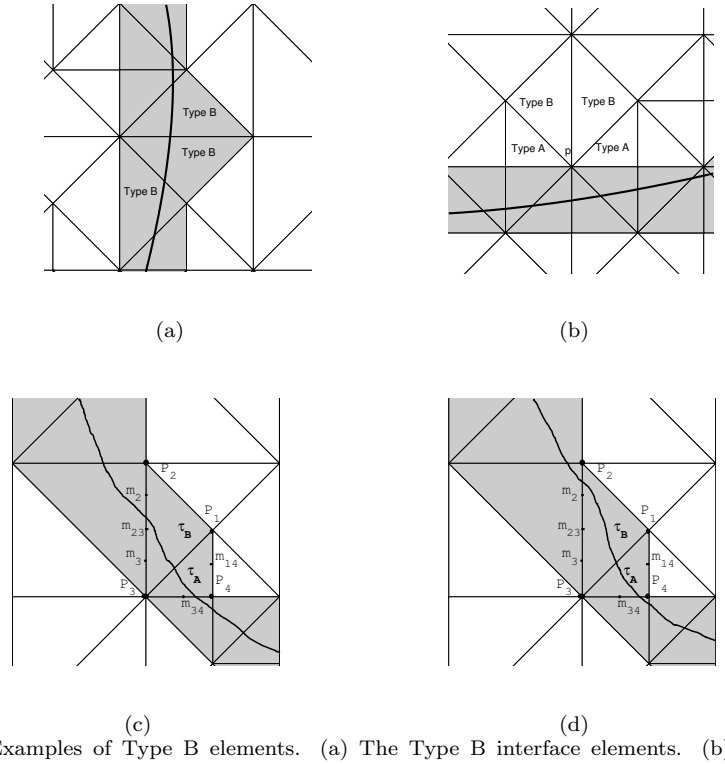


Fig. 2.7: Examples of Type B elements. (a) The Type B interface elements. (b) The Type B element pair near the interface which are not interface elements. (c) The transition Type B interface element τ_B which satisfies the condtion $\phi(m_2)\phi(m_3) < 0$. (d) The transition Type B interface element τ_B which satisfies the condition $\phi(m_{14})\phi(m_{34}) < 0$ and $\phi(m_2)\phi(p_2) < 0$.

Let the neighbors of an element be indexed in such a way that the i-th neighbor is opposite to its i-th vertex. A Type B interface element is called a *transition* Type B if its first neighbor is a Type A interface element, and its second or third neighbor is also of Type A interface element; see Fig. 2.7(c). We shall bisect some transition Type B elements according the rule described below.

Let us take the transition Type B interface element $\tau_B : (p_1, p_2, p_3)$ in Fig. 2.7(c) and (d) as an example to illustrate our criterion. The Type A interface element $\tau_A : (p_4, p_1, p_3)$ is the second neighbor of τ_B. Let m_{23}, m_{14} and m_{34} be the middle points of edge (p_2, p_3), (p_1, p_4) and (p_3, p_4), respectivly. Furthermore, let $m_2 = (p_2 + m_{23})/2$ and $m_3 = (m_{23} + p_3)/2$. We will bisect τ_B if one of the following two conditions is satisfied

(C1): $\phi(m_2)\phi(m_3) < 0$,

(C2): $\phi(m_{14})\phi(m_{34}) < 0$ and $\phi(m_2)\phi(p_2) < 0$.

Let p be the intersection point of Γ and edge (p_2, p_3). Condition **(C1)** means that $d(m_{23}, p) < d(p, p_2)$ and $d(m_{23}, p) < d(p, p_3)$, where $d(\cdot, \cdot)$ is the Euclidean distance of two points. So τ_B is bisected such that m_{23} is a mesh vertex and thus the perturbation (of vertices to intersection) is smaller; see Fig. 2.7 (c).

When condition **(C2)** is satisfied, the vertices p_2 and p_4 will be perturbed onto Γ while p_1 and p_3 will not be perturbed onto Γ by Börgers' algorithm. So edge (p_1, p_3) will cross the interface. The interface will be better approximated by swapping the

diagonal (p_1, p_3) with $(p2, p4)$ in the quadrilateral formed by p_1, p_2, p_3 and p_4. But for quadrilateral formed by different type of elements, the edge swapping would introduce an element with very small angle (for example, the triangle (p_1, p_2, p_4)). Therefore bisection of τ_B will be helpful; see Fig. 2.7 (d).

In order to enable the edge swapping in Step 5, we also need to bisect the pairs of Type B elements which are close to but not crossed by the interface, i.e. they are not interfaces elements but neighbors of interface elements; see Fig. 2.7(b).

Step 4. Following Börgers' algorithm in [5], we find all edges parallel to the coordinate axes and crossing the interface. For each such edge, we compute the distances from the two end vertices to the intersection point and replace the closer vertex by the intersection. If an interface vertex can be replaced by more than one intersection point, we choose the closest one; see Fig. 2.8.

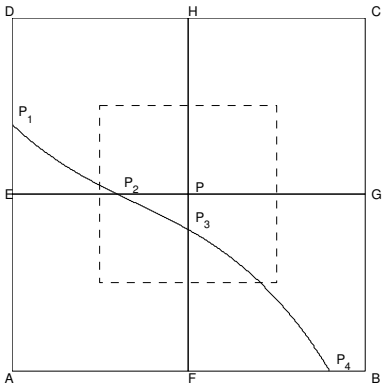
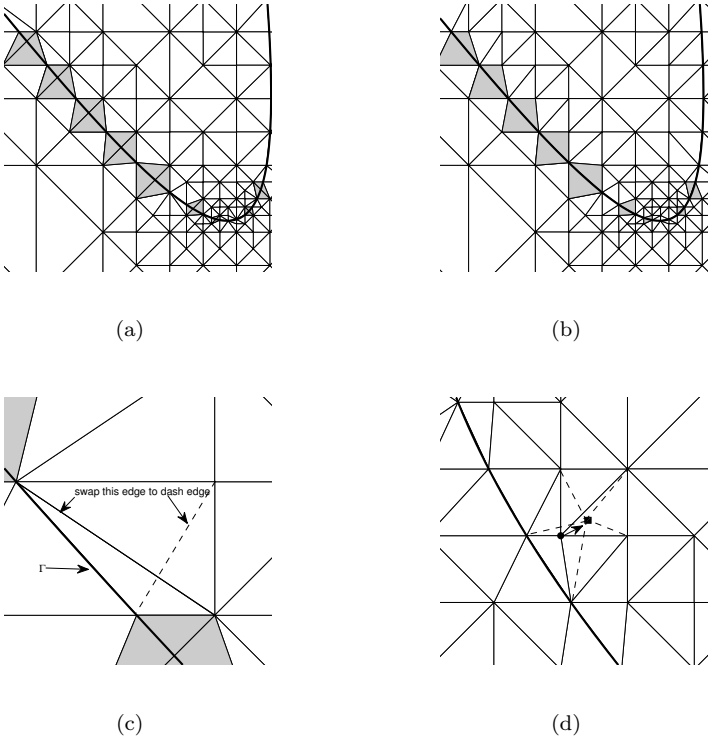


Fig. 2.8: Börgers' algorithm considers the intersections of the interface and the edges parallel with x or y coordinates and choose the smaller perturbation. In the above figure, because $|PP_2| < |EP_2|$ and $|PP_3| < |FP_3|$, so P can be moved to P_2 or P_3. Furthermore because of $|PP_3| < |PP_2|$, Börgers' algorithm will move P to P_3.

Step 5. After the perturbation, in Börgers' algorithm, one need to choose a suitable diagonals of the quadrilaterals containing interface vertices. In our setting it is equivalent to the edge swapping. The first case is a quadrilateral formed by two interface elements and the diagonal of this quadrilateral cross the interface. We apply edge swapping to make them to fit the interface; see (a) and (b) in Fig. 2.9. The second case is a quadrilateral formed by the element pair which has at least one vertex on the interface and share a common longest edge. We apply the edge swapping if the minimal angle can be improved and the new edge does not intersect with the interface; see Fig. 2.9(c). After edge swapping, the minimal angle is 19.1052° for this example, which is better than the theoretical lower bound 18.4349°.

Mesh smoothing can be further applied to improve the mesh quality for elements near the interface. The mesh smoothing scheme used here is the CPT smoothing which is a variant of ODT smoothing proposed in [7, 8, 10]. For any bad element with an edge fitting to the interface, relocate the vertex opposite to the fitting edge to the weighted average of barycenters of all triangles containing this vertex; see Fig. 2.9(d). The mesh smoothing improve the minimum angle to 24.0143° for this example.

Finally, we obtain a shape regular and body-fitted mesh which captures the geometric details of Γ and preserve some structure of the adaptive mesh; see Fig. 2.10.

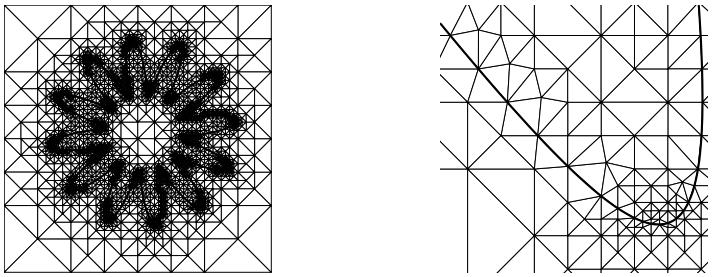


(a) (b)

(c) (d)

Fig. 2.9: Edge swapping and mesh smoothing are applied to fit the interface (the thicker curve) and improve the mesh quality. (a) The interface element pairs after step 4. For every pair, the common longest edge of two elements is crossed by the interface. (b) Apply edge swapping for the interface element pairs in (a) to make the common edge fit the interface. (c) Apply edge swapping for the element pair near interface to improve the minmal angle of the element pair. (d) Local CPT mesh smoothing. Relocate the vertex to the weighted average of barycenters of all triangles containing this vertex.

We emphasize again the obtained body- fitted mesh will be used a coarse grid and linear finite element approximation of the interface problem will be based on a sequence of meshes using the uniform refinement, i.e., each triangle is divided into four similar small triangles, of this coarse mesh.



(a) (b)

Fig. 2.10: The obtained mesh (a) and its local details (b).

3. FEMs for Interface Problems. In this section we consider linear finite element approximation of (1.1) based on a sequence of meshes obtained by several uniform refinement of the body-fitted mesh generated by Alg. 2.1. We provide a superconvergence analysis to prove that the linear finite element solution is superclose to the linear interpolant of the exact solution. A maximal norm estimate is obtained based on this superclose result.

For the simplicity of exposition, we assume the domain Ω is a square, the function value jump condition $[u]_\Gamma = 0$, and the Dirichlet boundary condition $u|_{\partial\Omega} = 0$. Furthermore, we assume the coefficient function $\beta(x)$ is positive and piecewise constant. The results in this section, however, can be extended to general elliptic interface problems without essential difficulty. In addition, we use $x \lesssim y$ to indicate $x \leq Cy$.

3.1. The Sobolev spaces and the weak formulation. Let D denote a bounded open and connected set in $\mathbb{R}^2$ and $W^{k,p}(D)$ the usual Sobolev space with standard norm $\|\cdot\|_{k,p,D}$ and seminorm $|\cdot|_{k,p,D}$. For $p = 2$, we denote $W^{k,2}(D)$ by $H^k(D)$ and the corresponding norm and seminorm by $\|\cdot\|_{k,D} = \|\cdot\|_{k,2,D}$ and $|\cdot|_{k,D} = |\cdot|_{k,2,D}$, respectively. The space $H_0^1(D)$ is a subspace of $H^1(D)$ whose elements have zeros trace on ∂D. Let $(\cdot,\cdot)_D$ and $\langle\cdot,\cdot\rangle_{\partial D}$ denote the standard L^2 inner products of $L^2(D)$ and $L^2(\partial D)$, respectively.

For (1.1), the domain Ω can be decomposed as $\Omega = \Omega^- \cup \Gamma \cup \Omega^+$ (see Fig. 2.1(a)). We denote by $W^{k,p}(\Omega^- \cup \Omega^+)$ the Sobolev space consisting of function w such that $w|_{\Omega^-} \in W^{k,p}(\Omega^-)$ and $w|_{\Omega^+} \in W^{k,p}(\Omega^+)$ equipped with norm

$$\|w\|_{k,p,\Omega^-\cup\Omega^+} = \left(\|w\|_{k,p,\Omega^-}^p + \|w\|_{k,p,\Omega^+}^p\right)^{1/p},$$

and seminorm

$$|w|_{k,p,\Omega^-\cup\Omega^+} = \left(|w|_{k,p,\Omega^-}^p + |w|_{k,p,\Omega^+}^p\right)^{1/p},$$

with standard modification for $p = \infty$.

The weak formulation of (1.1) is as follows: find $u \in H_0^1(\Omega)$ such that:

$$(\beta\nabla u, \nabla v)_\Omega = (f, v)_\Omega - \langle q_1, v\rangle_\Gamma \quad \text{for all } v \in H_0^1(\Omega). \tag{3.1}$$

The existence and uniqueness of the solution can be easily proved by the Lax-Millegram theorem. For the solution u of (3.1), we have the following regularity result [27]: $u \in H^r(\Omega^- \cup \Omega^+)$

$$\|u\|_{r,\Omega^-\cup\Omega^+} \lesssim \|f\|_{0,\Omega} + \|q_1\|_{r-3/2,\Gamma},$$

where $0 \leq r \leq 2$.

Remark 3.1. If the function value contains a jump $[u]_\Gamma = q_0 \neq 0$, we can find $\phi^- : \Omega^- \to \mathbb{R}$ with $\phi^- = q_0$ on $\partial\Omega^-$ and $\phi^- \in H^1(\Omega^-)$. The zero extension of ϕ^-, denoted by ϕ satisfies $\phi \in H^1(\Omega^- \cup \Omega^+)$ with $\phi = \phi^-$ on Ω^- and $\phi = 0$ on Ω^+. The model (1.1) is equivalent to: find $u = q - \phi$ with $q \in H_0^1(\Omega)$ such that:

$$(\beta\nabla q, \nabla v)_\Omega = (f, v)_\Omega - \langle q_1, v\rangle_\Gamma + (\beta\nabla\phi, \nabla v)_{\Omega^-}$$

for all $v \in H_0^1(\Omega)$. Notice that the choice of ϕ is not unique and can be approximated by a finite element function [19].

3.2. Finite Element Approximation. Let $\mathcal{T}_h$ be a body-fitted and shape regular triangular mesh. For each $\tau \in \mathcal{T}_h$, let h_τ denote its diameter and $h = \max_{\tau\in\mathcal{T}_h} h_\tau$. The vertices on Γ forms a polygon Γ_h approximation of Γ. The polygon Γ_h also splits Ω into two subdomains: Ω_h^- and Ω_h^+ which are the approximations of Ω^- and Ω^+, respectively. Each triangle $\tau \in \mathcal{T}_h$ is either in Ω_h^+ or Ω_h^-, and has at most two vertices on Γ. The triangulation $\mathcal{T}_h$ can be decomposed into three parts:

$$\begin{aligned} \mathcal{T}_h^+ &:= \{\tau \in \mathcal{T}_h | \tau \subset \Omega_h^+, \tau \text{ has at most one vertex on } \Gamma\}, \\ \mathcal{T}_h^- &:= \{\tau \in \mathcal{T}_h | \tau \subset \Omega_h^-, \tau \text{ has at most one vertex on } \Gamma\}, \\ \mathcal{T}_h^0 &:= \{\tau \in \mathcal{T}_h | \tau \text{ has two vertices on } \Gamma\}. \end{aligned} \tag{3.2}$$

Let $h_\Gamma = \max_{\tau\in\mathcal{T}_h^0} h_\tau$ denote the mesh size near the interface. Assume Γ is of class $\mathcal{C}^2$, it is easy to show that the distance between every edge $E_h \in \Gamma_h$ and Γ is $\mathcal{O}(h_{E_h}^2)$, where h_{E_h} is the length of E_h. Furthermore, for each triangle $\tau \in \mathcal{T}_h^0$, let $\tau^+ = \tau \cap \Omega^+$ and $\tau^- = \tau \cap \Omega^-$, and since $\Gamma \in \mathcal{C}^2$, we have either $|\tau^+| \lesssim h_\tau^3$ or $|\tau^-| \lesssim h_\tau^3$; see Fig. 3.1.

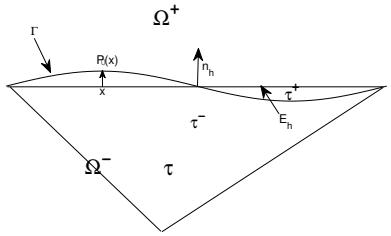


Fig. 3.1: The projection $\mathcal{P}_0$. In this figure, $\tau^+ = \tau \cap \Omega^+$ and $\tau^- = \tau \cap \Omega^-$, and $|\tau^+| \lesssim h_\tau^3$.

Let E_h be an edge of Γ_h and n_h the unit normal of E_h pointing from Ω_h^- to Ω_h^+. We can define a projection $\mathcal{P}_0$ from E_h to Γ [6] (see Fig. 3.1):

$$\mathcal{P}_0(x) = x + d(x)n_h \quad \text{for all } x \in E_h,$$

where $\mathcal{P}_0(x) \in \Gamma$ and $|d(x)|$ is the distance between x and Γ along n_h. We assume the length of E_h is small enough so that $\mathcal{P}_0$ and its inverse $\mathcal{P}_0^{-1}$ are all well defined.

Let V_h be the linear finite element space on $\mathcal{T}_h$. The linear finite element approximation of (3.1) is: find $u_h \in V_h \cap H_0^1(\Omega)$ such that:

$$(\beta_h \nabla u_h, \nabla v_h)_\Omega = (f, v_h)_\Omega - \langle \bar{q}_1, v_h \rangle_{\Gamma_h} \quad \text{for all } v_h \in V_h \cap H_0^1(\Omega), \tag{3.3}$$

where $\bar{q}_1(x) = q_1(\mathcal{P}_0(x))$ for all $x \in \Gamma_h$ and $\beta_h|_\tau = \beta^+$ for all $\tau \in \Omega_h^+$ and $\beta_h|_\tau = \beta^-$ for all $\tau \in \Omega_h^-$. We can find the following nearly optimal L^2-norm and H^1-norm estimates; see [14, 32].

THEOREM 3.1. *Let u be the solution of* (1.1) *and u_h the linear element solution of* (3.3). *We have*

$$\|\nabla u - \nabla u_h\|_{0,\Omega} \lesssim h \left|\log h\right|^{1/2} (\|f\|_{0,\Omega} + \|q_1\|_{2,\Omega}), \tag{3.4}$$

$$\|u - u_h\|_{0,\Omega} \lesssim h^2 \left|\log h\right| (\|f\|_{0,\Omega} + \|q_1\|_{2,\Omega}). \tag{3.5}$$

3.3. Superconvergence. Let E be an interior edge of $\mathcal{T}_h$ and Ω_E the patch of E consisting of two triangles sharing the edge E. We say Ω_E is an $\mathcal{O}(h^2)$ approximate parallelogram if the lengths of any two opposite edges differ only by $\mathcal{O}(h^2)$. We adapt the definition of the $\mathcal{O}(h^{2\sigma})$ irregular grids (see [2, 11]) to the body-fitted mesh.

DEFINITION 3.2. *Let $\mathcal{E} = \mathcal{E}_1 \oplus \mathcal{E}_2$ denote the set of interior edge of $\mathcal{T}_h$. The triangulation $\mathcal{T}_h$ is $\mathcal{O}(h^{2\sigma})$ irregular, for some $\sigma > 0$, if for each $E \in \mathcal{E}_1$, E is not an edge of Γ_h and Ω_E forms an $\mathcal{O}(h^2)$ approximate parallelogram, while $\sum_{E\in\mathcal{E}_2} |\Omega_E| = \mathcal{O}(h^{2\sigma})$.*

Taking the body-fitted mesh generated by Alg. 2.1 as the initial mesh, we uniformly refine it, i.e., divide each triangle into four similar small triangles, to get a sequence of meshes of Ω. By computing $\sum_{E\in\mathcal{E}_2} |\Omega_E|$ of each mesh in the sequence, one can show that the mesh sequence obtained is $\mathcal{O}(h^{2\sigma})$ irregular, and numerically $\sigma \approx 0.5$.

LEMMA 3.3. *Let $u \in W^{1,\infty}(\Omega^- \cup \Omega^+)$; then*

$$|(\beta\nabla u, \nabla v_h)_\Omega - (\beta_h \nabla u, \nabla v_h)_\Omega| \lesssim h_\Gamma^{3/2} \|u\|_{1,\infty,\Omega^-\cup\Omega^+} |v_h|_{1,\Omega} \text{ for all } v_h \in V_h. \tag{3.6}$$

Proof. Given $\tau \in \mathcal{T}_h^0$, let $\tau^* = \operatorname{supp}(\beta - \beta_h) \cap \tau$. Then $|\tau^*| \lesssim h_\tau^3$ and

$$\begin{aligned}
&|(\beta\nabla u, \nabla v_h)_\Omega - (\beta_h \nabla u, \nabla v_h)_\Omega| \\
\le &\sum_{\tau\in\mathcal{T}_h^0} \int_{\tau^*} |[\beta]\nabla u \cdot \nabla v_h| \lesssim \sum_{\tau\in\mathcal{T}_h^0} \|u\|_{1,\infty,\Omega^-\cup\Omega^+} \int_{\tau^*} |\nabla v_h| \\
\lesssim &\sum_{\tau\in\mathcal{T}_h^0} h_\tau^{3/2} \|u\|_{1,\infty,\Omega^-\cup\Omega^+} |v_h|_{1,\tau^*} \lesssim h_\Gamma^2 \|u\|_{1,\infty,\Omega^-\cup\Omega^+} \sum_{\tau\in\mathcal{T}_h^0} |v_h|_{1,\tau} \\
\lesssim &h_\Gamma^{3/2} \|u\|_{1,\infty,\Omega^-\cup\Omega^+} |v_h|_{1,\Omega},
\end{aligned}$$

where we have used the fact that ∇v_h is constant on τ. Obviously, the constant in $\lesssim$ is $C|\beta^- - \beta^+|$. ∎

The following lemma is slightly different with Lemma 2.2 in [14]. Here we use the extension $\bar{q}_1 = q_1(\mathcal{P}_0(x))$ to replace the linear interpolation of q_1 on Γ_h in [14].

LEMMA 3.4. *Assume $q_1 \in W^{0,\infty}(\Gamma)$ and $\Gamma \in C^2$; then*

$$|\langle q_1, v_h\rangle_\Gamma - \langle \bar{q}_1, v_h\rangle_{\Gamma_h}| \lesssim h_\Gamma^{3/2} \|q_1\|_{0,\infty,\Gamma} \|v_h\|_{1,\Omega} \quad \text{for all } v_h \in V_h, \tag{3.7}$$

where $\bar{q}_1(x) = q_1(\mathcal{P}_0(x))$ for all $x \in \Gamma_h$.

Proof.

$$\begin{aligned}
&\langle q_1, v_h\rangle_\Gamma - \langle \bar{q}_1, v_h\rangle_{\Gamma_h} \\
= &\sum_{E_h\in\Gamma_h} \int_{E_h} q_1(\mathcal{P}_0(x)) v_h(\mathcal{P}_0(x)) J(x) \mathrm{d}s - \int_{E_h} q_1(\mathcal{P}_0(x)) v_h(x) \mathrm{d}s \\
= &\sum_{E_h\in\Gamma_h} \left[\int_{E_h} q_1(\mathcal{P}_0(x))(v_h(\mathcal{P}_0(x)) - v_h(x)) J(x) \mathrm{d}s + \int_{E_h} q_1(\mathcal{P}_0(x)) v_h(x)(J(x) - 1) \mathrm{d}s\right] \\
= &\sum_{E_h\in\Gamma_h} [I_1 + I_2],
\end{aligned}$$

where $J(x)$ is the Jacobian of the projection $\mathcal{P}_0$.

For I_1, since v_h is piecewise linear function, we have

$$|I_1| \lesssim \|q_1\|_{0,\infty,\Gamma} \int_{E_h} |\nabla v_h(\mathcal{P}_0(x)) \cdot (\mathcal{P}_0(x) - x)| \lesssim h_\Gamma^2 \|q_1\|_{0,\infty,\Gamma} \int_{E_h} |\nabla v_h(\mathcal{P}_0(x))|$$
$$\lesssim h_\Gamma \|q_1\|_{0,\infty,\Gamma} \int_{\Omega_{E_h}} |\nabla v_h| \lesssim h_\Gamma^2 \|q_1\|_{0,\infty,\Gamma} |v_h|_{1,\Omega_{E_h}} .$$

For I_2, since $\Gamma \in C^2$, we have $|J(x) - 1| \lesssim h^2$. Then we have

$$|I_2| \lesssim h_\Gamma^2 \|q_1\|_{0,\infty,\Gamma} \int_{E_h} |v_h| \lesssim h_\Gamma^{5/2} \|q_1\|_{0,\infty,\Gamma} \|v_h\|_{0,E_h} \lesssim h_\Gamma^2 \|q_1\|_{0,\infty,\Gamma} \|v_h\|_{1,\Omega_{E_h}} ,$$

where we used the scaled trace theorem. Then we have

$$|\langle q_1, v_h\rangle_\Gamma - \langle \bar{q}_1, v_h\rangle_{\Gamma_h}| \lesssim h_\Gamma^2 \|q_1\|_{0,\infty,\Gamma} \sum_{E_h \in \Gamma_h} \|v_h\|_{1,\Omega_{E_h}} \lesssim h_\Gamma^{3/2} \|q_1\|_{0,\infty,\Gamma} \|v_h\|_{1,\Omega} .$$

□

We are in the position to present the main superconvergence results.

THEOREM 3.5. *Suppose the body-fitted triangulation $\mathcal{T}_h$ is $\mathcal{O}(h^{2\sigma})$ irregular. Let u be the solution of (1.1), u_h the linear element solution of (3.3) and u_I the linear interpolation of u in V_h. If $u \in H^1(\Omega) \cap H^3(\Omega^- \cup \Omega^+) \cap W^{2,\infty}(\Omega^- \cup \Omega^+)$ and Γ is of class $\mathcal{C}^2$; then for all $v_h \in V_h$,*

$$\begin{aligned}(\beta_h \nabla(u - u_I), \nabla v_h)_\Omega \lesssim & h^{1+\min\{1,\sigma\}} (\|u\|_{3,\Omega^- \cup \Omega^+} + \|u\|_{2,\infty,\Omega^- \cup \Omega^+}) |v_h|_{1,\Omega} \\ & + h_\Gamma^{3/2} \|u\|_{2,\infty,\Omega^- \cup \Omega^+} |v_h|_{1,\Omega} ,\end{aligned} \tag{3.8}$$

and

$$\begin{aligned}\left\| \beta_h^{1/2} (\nabla u_h - \nabla u_I) \right\|_{0,\Omega} \lesssim & h^{1+\min\{1,\sigma\}} (\|u\|_{3,\Omega^- \cup \Omega^+} + \|u\|_{2,\infty,\Omega^- \cup \Omega^+}) \\ & + h_\Gamma^{3/2} (\|u\|_{2,\infty,\Omega^- \cup \Omega^+} + \|q_1\|_{0,\infty,\Gamma}).\end{aligned} \tag{3.9}$$

Proof. Let u^- and u^+ be the restrictions of u on Ω^- and Ω^+, repectively. We have $u^- \in H^3(\Omega^-) \cap W^{2,\infty}(\Omega^-)$ and $u^+ \in H^3(\Omega^+) \cap W^{2,\infty}(\Omega^+)$. Since Γ is of class $\mathcal{C}^2$, by the extension theorem in [29], we can extend u^- onto the whole domain Ω and still denote the extension by u^-. We have the extension $u^- \in H^3(\Omega) \cap W^{2,\infty}(\Omega)$ and

$$\left\|u^-\right\|_{3,\Omega} \lesssim \left\|u^-\right\|_{3,\Omega^-} \text{ and } \left\|u^-\right\|_{2,\infty,\Omega} \lesssim \left\|u^-\right\|_{2,\infty,\Omega^-} .$$

Similarly, we can obtain the extension of u^+ on Ω. We then split the target term as

$$\begin{aligned}& (\beta_h \nabla(u - u_I), \nabla v_h)_\Omega \\ = & \sum_{\tau \in \mathcal{T}_h^+} (\beta^+ \nabla(u - u_I), \nabla v_h)_\tau + \sum_{\tau \in \mathcal{T}_h^-} (\beta^- \nabla(u - u_I), \nabla v_h)_\tau \\ & + \sum_{\tau \in \mathcal{T}_h^0} [(\beta_h \nabla(u - u_I), \nabla v_h)_{\tau^+} + (\beta_h \nabla(u - u_I), \nabla v_h)_{\tau^-}]\end{aligned} \tag{3.10}$$

Let u_I^- and u_I^+ be the linear finite element interpolations of the extension u^- and u^+ in V_h, repectively. For $\tau \in \mathcal{T}_h^0$ with $|\tau^-| \lesssim h_\tau^3$, we have $\beta_h = \beta^+$ on τ and

$$\begin{aligned}& (\beta^+ \nabla(u - u_I), \nabla v_h)_{\tau^+} + (\beta^+ \nabla(u - u_I), \nabla v_h)_{\tau^-} \\ = & (\beta^+ \nabla(u^+ - u_I^+), \nabla v_h)_\tau - (\beta^+ \nabla(u^+ - u_I^+), \nabla v_h)_{\tau^-} \\ & + (\beta^+ \nabla(u - u_I), \nabla v_h)_{\tau^-} .\end{aligned}$$

For $\tau \in \mathcal{T}_h^0$ with $|\tau^+| \lesssim h_\tau^3$, we have a similar identity.

Then (3.10) can be written into the following form

$$\begin{aligned}
&(\beta_h \nabla(u - u_I), \nabla v_h)_\Omega \\
=&(\beta^- \nabla(u^- - u_I^-), \nabla v_h)_{\Omega_h^-} + (\beta^+ \nabla(u^+ - u_I^+), \nabla v_h)_{\Omega_h^+} \\
&+ \sum_{\tau \in \mathcal{T}_h^0, |\tau^-| \lesssim h_\tau^3} \left[(\beta^+ \nabla(u - u_I), \nabla v_h)_{\tau^-} - (\beta^+ \nabla(u^+ - u_I^+), \nabla v_h)_{\tau^-} \right] \\
&+ \sum_{\tau \in \mathcal{T}_h^0, |\tau^+| \lesssim h_\tau^3} \left[(\beta^- \nabla(u - u_I), \nabla v_h)_{\tau^+} - (\beta^- \nabla(u^- - u_I^-), \nabla v_h)_{\tau^+} \right] \\
=&I_1 + I_2 + \sum_{\tau \in \mathcal{T}_h^0, |\tau^-| \lesssim h_\tau^3} [I_{31} - I_{32}] + \sum_{\tau \in \mathcal{T}_h^0, |\tau^+| \lesssim h_\tau^3} [I_{41} - I_{42}] .
\end{aligned}$$

For I_1 and I_2, we have the following estimates which can be found in [11]:

$$\begin{aligned}
|I_1| &\lesssim h^{1+\min\{1,\sigma\}} (\left\|u^-\right\|_{3,\Omega^-} + \left\|u^-\right\|_{2,\infty,\Omega^-}) |v_h|_{1,\Omega} , \\
|I_2| &\lesssim h^{1+\min\{1,\sigma\}} (\left\|u^+\right\|_{3,\Omega^+} + \left\|u^+\right\|_{2,\infty,\Omega^+}) |v_h|_{1,\Omega} .
\end{aligned}$$

For the element $\tau \in \mathcal{T}_h^0$ with $|\tau^-| \lesssim h_\tau^3$, which means that the most part of τ is in Ω^+, we know that $u \in H^1(\tau) \cap W^{2,\infty}(\tau^- \cup \tau^+)$. By Taylor expansion [32], we have

$$|u - u_I|_{1,\infty,\tau^-} \lesssim \|u\|_{1,\infty,\tau^- \cup \tau^+} ,$$

and

$$|I_{31}| \lesssim \|u\|_{1,\infty,\tau^- \cup \tau^+} \int_{\tau^-} |\nabla v_h| \lesssim h_\tau^{3/2} \|u\|_{1,\infty,\tau^- \cup \tau^+} |v_h|_{1,\tau^-} \lesssim h_\tau^2 \|u\|_{1,\infty,\tau^- \cup \tau^+} |v_h|_{1,\tau} .$$

For I_{32}, since $u^+ \in H^2(\tau) \cap W^{2,\infty}(\tau)$, we have

$$|I_{32}| \lesssim \left\|\nabla u^+ - \nabla u_I^+\right\|_{0,\infty,\tau} \int_{\tau^-} |\nabla v_h| \lesssim h_\tau^{5/2} \left\|u^+\right\|_{2,\infty,\tau} |v_h|_{1,\tau^-} \lesssim h_\tau^3 \left\|u^+\right\|_{2,\infty,\tau} |v_h|_{1,\tau} .$$

Similarly, for the element $\tau \in \mathcal{T}_h^0$ with $|\tau^+| \lesssim h_\tau^3$, we can prove

$$|I_{41}| \lesssim h_\tau^2 \|u\|_{1,\infty,\tau^- \cup \tau^+} |v_h|_{1,\tau} , \quad |I_{42}| \lesssim h_\tau^3 \left\|u^-\right\|_{2,\infty,\tau} |v_h|_{1,\tau} .$$

By the fact that $\sum_{\tau \in \mathcal{T}_0^h} 1 \lesssim h_\Gamma^{-1}$, we have

$$\begin{aligned}
&\sum_{\tau \in \mathcal{T}_h^0, |\tau^-| \lesssim h_\tau^3} |I_{31} - I_{32}| + \sum_{\tau \in \mathcal{T}_h^0, |\tau^+| \lesssim h_\tau^3} |I_{41} - I_{42}| \\
\lesssim& h_\Gamma^2 \|u\|_{2,\infty,\Omega^- \cup \Omega^+} \sum_{\tau \in \mathcal{T}_h^0} |v_h|_{1,\tau} \lesssim h_\Gamma^{3/2} \|u\|_{2,\infty,\Omega^- \cup \Omega^+} |v_h|_{1,\Omega} .
\end{aligned}$$

The inequality (3.8) then follows.

We now prove (3.9) as

$$\begin{aligned}
&(\beta_h \nabla(u_h - u_I), \nabla v_h)_\Omega \\
=&(\beta_h \nabla(u - u_I), \nabla v_h)_\Omega + (\beta \nabla u, \nabla v_h)_\Omega - (\beta_h \nabla u, \nabla v_h)_\Omega \\
&+ \langle q_1, v_h \rangle_\Gamma - \langle \bar{q}_1, v_h \rangle_{\Gamma_h} .
\end{aligned}$$

Combining (3.8), Lemma 3.4 and 3.3, and taking $v_h = u_h - u_I$, we proved (3.9). ∎

At last, we give the the maximal norm error estimate. Here let $h_{\min}$ be the minimum element size of $\mathcal{T}_h$. By the discrete embedding result

$$\|v_h\|_{0,\infty,\Omega} \lesssim |\log h_{\min}|^{1/2} |v_h|_{1,\Omega} \qquad \text{for all } v_h \in V_h \cap H_0^1(\Omega),$$

we have the following corollary immediately:

COROLLARY 3.6. *Assume the same hypothesis in Theorem* 3.5*; then*

$$\begin{aligned} \|u_h - u_I\|_{0,\infty,\Omega} \lesssim |\log h_{\min}|^{1/2} \Big[& h^{1+\min\{1,\sigma\}} (\|u\|_{3,\Omega^-\cup\Omega^+} + \|u\|_{2,\infty,\Omega^-\cup\Omega^+}) \\ & + h_\Gamma^{3/2} (\|u\|_{2,\infty,\Omega^-\cup\Omega^+} + \|q_1\|_{0,\infty,\Gamma}) \Big]. \end{aligned} \tag{3.11}$$

Remark 3.2. For the body-fitted mesh $\mathcal{T}_h$ generated by Alg. 2.1, h_Γ is generally far less than h, the maximum size of $\mathcal{T}_h$; see the body-fitted meshes in Section 4. Therefore the second term of the right-hand side of (3.11) is a high order term.

4. Numerical experiments. In this section, we present two numerical examples to demonstrate the effectiveness of the mesh algorithm combined with the FEM. We use Matlab package *i*FEM [9] to do the experiment on a machine with 2.8 GHz Intel Xeon processor. To test the rate of convergence, we first generate a body-fitted triangular mesh using Alg. 2.1 and then uniformly refine it to get a sequence of meshes. Recall that here uniformly refinement means one triangle is subdivied into four by connecting the middle points of the three edges of the triangle. The new vertices on the interface edges introduced in each refinement will be projected onto the interface.

The coefficients $\beta^+(x)$ and $\beta^-(x)$ and the exact solution $u^+(x)$ and $u^-(x)$ are given such that the source term, boundary condition, and jump conditions can be determined accordingly. Let u_h^- and u_h^+ be the linear finte element approximations of u^- and u^+, respectively. We will test the following four errors:

$$\begin{aligned} \|u - u_h\|_0 &:= (\|u^- - u_h^-\|_{0,\Omega^-}^2 + \|u^+ - u_h^+\|_{0,\Omega^+}^2)^{1/2}, \\ |u - u_h|_1 &:= (|u^- - u_h^-|_{1,\Omega^-}^2 + |u^+ - u_h^+|_{1,\Omega^+}^2)^{1/2}, \\ \|u_h - u_I\|_A &:= (\|\beta_h^{1/2}\nabla(u_h^- - u_h^-)\|_{0,\Omega_h^-}^2 + \|\beta_h^{1/2}\nabla(u_h^+ - u_I^+)\|_{0,\Omega_h^+}^2)^{1/2}, \\ \|u_h - u_I\|_\infty &:= \max(\|u_h^- - u_I^-\|_{0,\infty,\Omega_h^-}, \|u_h^+ - u_I^+\|_{0,\infty,\Omega_h^+}). \end{aligned}$$

Remark 4.1. Notice that, u_h^- defined on Ω_h^- and u_h^+ on Ω_h^+. But with simple linear extension on the interface elements [6], we can extend them to Ω^- and Ω^+, respectively. In order to compute $\|u - u_h\|_0$ and $|u - u_h|_1$ accurately, we use quadratic curve to replace the straight edge in Γ_h.

For the solver, we use one multigrid $\mathcal{V}$-cycle as a preconditioner in PCG and call it MGCG method. The key component of such solver is a coarsening algorithm [12], with some modification for the mesh generated by Alg. 2.1, to construct a sequence of nested meshes. After the coarsening, the prolongation and restriction operators can be built algebraically and so is the standard $\mathcal{V}$-cycle.

For a 2-D quasi-uniform mesh $h \approx N^{-1/2}$ and $h^2 \approx N^{-1}$. In Fig. 4.2 and 4.4, the rate of convergence is obtained by the least square fitting of the errors to a straight-line in terms of $\log N$. The slope will be the rate of convergence in terms of N. From these tests, it is evident that

- $\|u-u_h\|_0$ and $|u-u_h|_1$ reach the optimal order N^{-1} and $N^{-1/2}$, respectively;
- $|u_I - u_h|_1$ has superconvergence $N^{-(1/2+\sigma/2)}$ with $\sigma \leq 1$;
- $\|u_I - u_h\|_\infty$ achieves the order of convergence $N^{-0.8}$;
- The MGCG solver scales linearly and is robust with respect to the jump of the diffusion coefficients.

4.1. Example 1. Our first example is borrowed from [21]. The domain Ω is $(-1,1)^2$ and the interface is defined by

$$X(\theta) = r(\theta)\cos(\theta) + x_c,\ Y(\theta) = r(\theta)\sin(\theta) + y_c,$$

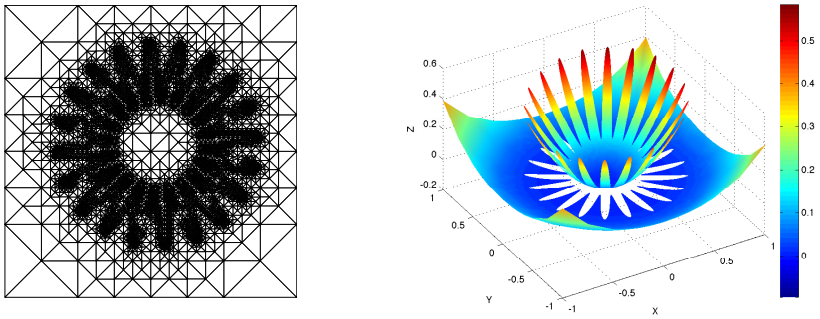
where $r(\theta) = r_0 + r_1\sin(\omega\theta),\ 0 \leq \theta < 2\pi$.

The parameters are set to $r_0 = 0.5$, $r_1 = 0.2$, $\omega = 20$, and $x_c = y_c = 0.02\sqrt{5}$. The analytic solution is given by

$$u^+ = (r^4 + C_0\log(2r))/\beta^+, \quad \text{and} \quad u^- = r^2/\beta^-,$$

where $\beta^+ = 10$, $\beta^- = 1$, $r = \sqrt{x^2+y^2}$, and $C_0 = -0.1$.

The body-fitted mesh generated by Alg. 2.1 is plotted in Fig. 4.1(a) and the linear finite element approximation of the solution is plotted in Fig. 4.1(b). The mesh in Fig. 4.1(a) resolves the interface very well and has a minimum angle 25.7137°, maximum element size 0.5 and minimum mesh size 6.7360e-05. The errors of $\|u-u_h\|_0$, $|u-u_h|_1$, $\|u_I-u_h\|_A$, and $\|u_I-u_h\|_\infty$ are presented in Table 4.1 and the rates of convergence are plotted in Fig. 4.2 (a) and (b).



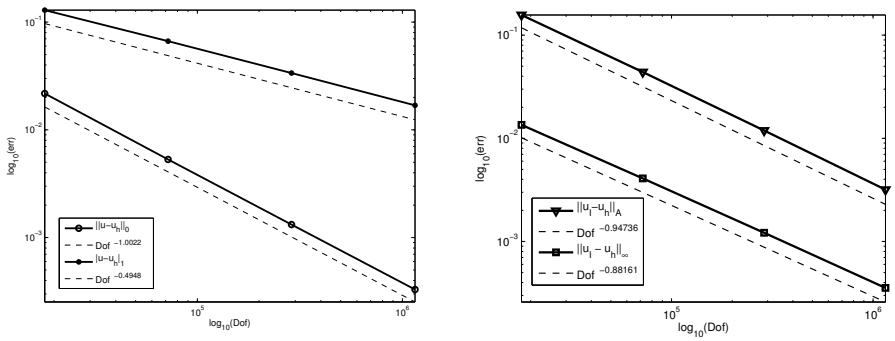
(a) (b)

Fig. 4.1: The mesh and solution of Example 1. (a) The initial body-fitted mesh with minimum angle 25.7137°. (b) The solution on the body-fitted mesh.

Dofs	$\|u-u_h\|_0$	$\|u-u_h\|_1$	$\|u_I-u_h\|_A$	$\|u_I-u_h\|_\infty$	$h_{\min}$
18004	2.1762e-02	1.2936e-01	1.5704e-01	1.3512e-02	6.7360e-05
71987	5.3077e-03	6.6627e-02	4.3957e-02	4.0870e-03	3.3680e-05
287893	1.3200e-03	3.3682e-02	1.1900e-02	1.2155e-03	1.6840e-05
1151465	3.2976e-04	1.6901e-02	3.1798e-03	3.5477e-04	8.4200e-06
Order	2.004	0.9896	1.8947	1.7632	

Table 4.1: Errors measured in H^1, L^2, and L^∞ norms for Example 1.

In Table 4.2, we present the variation of iteration steps and computational time with respect to the number of degree of freedoms (Dofs) and to the jump coefficients



(a) (b)

Fig. 4.2: The errors of Example 1. (a) The errors of $\|u - u_h\|_0$ and $|u - u_h|_1$. (b) The errors of $\|u_I - u_h\|_A$ and $\|u_I - u_h\|_\infty$.

by fixing $\beta^- = 1$ and varying β^+. For Example 1, since the analytic solution is dependent on the $\frac{1}{\beta^+}$ and $\frac{1}{\beta^-}$, all errors will decrease with the increased β^+ and fixed Dofs. The error of $\|u_I - u_h\|_A$ is weighted by β_h, so here we just list the error of $\|u_I - u_h\|_A$ with increased β^+, see Table. 4.3.

	$\beta^+ = 10$		$\beta^+ = 100$		$\beta^+ = 1000$		$\beta^+ = 10000$	
Dofs	#Iter	Time(s)	#Iter	Time(s)	#Iter	Time(s)	#Iter	Time(s)
18004	9	0.38	13	0.47	15	0.51	15	0.52
71987	10	0.51	13	0.88	15	0.96	15	0.96
287893	10	2.2	13	2.6	14	2.7	14	2.7
1151465	10	8.8	13	11	14	11	14	11

Table 4.2: Example 1, the iterations and computing time of MGCG with fixed $\beta^- = 1$ and increased β^+.

Dofs	$\beta^+ = 10$	$\beta^+ = 100$	$\beta^+ = 1000$	$\beta^+ = 10000$
18004	1.570387e-01	5.174507e-02	2.161404e-02	1.568107e-02
71987	4.395695e-02	1.461396e-02	6.404768e-03	4.873726e-03
287893	1.189950e-02	3.983592e-03	1.804054e-03	1.411514e-03
1151465	3.179841e-03	1.069657e-03	4.948437e-04	3.935619e-04
Order	1.8947	1.8863	1.8472	1.8154

Table 4.3: Example 1, the errors of $\|u_I - u_h\|_A$ with fixed $\beta^- = 1$ and increased β^+.

4.2. Example 2. The second example is taken from [3]. The domain Ω is $(-1, 1)^2$ and the interface is defined by

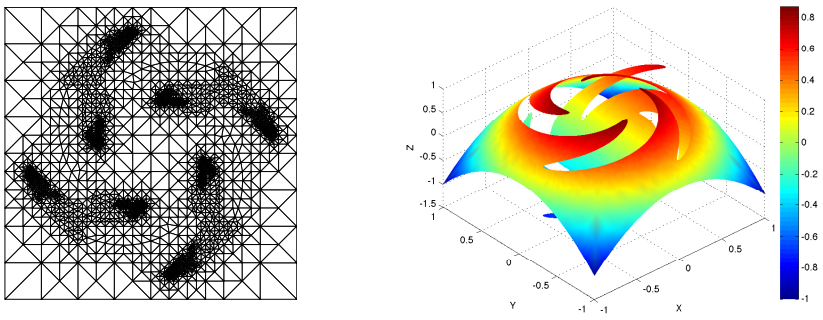
$$\theta(t) = t + \sin(4t), \quad r(t) = 0.60125 + 0.24012\cos(4t + \pi/2),$$
$$X(t) = r(t)\cos(\theta(t)), \quad Y(t) = r(t)\sin(\theta(t)),$$

where $0 \le t \le 2\pi$. The analytic solution is given by

$$u^- = \cos(y)\sin(x),\ u^+ = 1 - x^2 - y^2,$$

with $\beta^- = 4 + \sin(x + y)$ and $\beta^+ = 10000 + x^2 + y^2$.

For this example, the body-fitted mesh generated by Alg. 2.1 and the linear finite element approximation of the solution is plotted in Fig. 4.3. The mesh in Fig. 4.3(a) resolves the interface and has a minimum angle 26.2797°, maximum element size 0.3536 and minimum element size 1.0451e-03. The errors of $\|u - u_h\|_0$, $|u - u_h|_1$, $\|u_I - u_h\|_A$, and $\|u_I - u_h\|_\infty$ are presented in Table 4.4 and the rates of convergence are plotted in Fig. 4.4 (a) and (b).

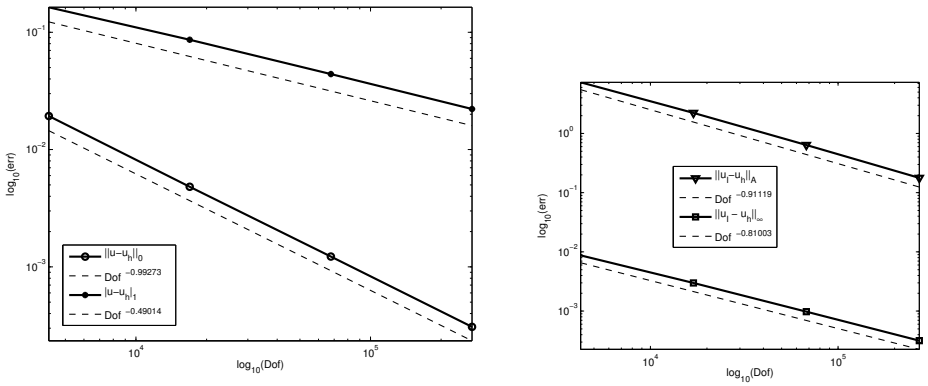


(a) (b)

Fig. 4.3: The mesh and solution of Example 2. (a) The initial body-fitted mesh with minimum angle 26.2797°. (b) The solution on the body-fitted mesh.

Dofs	$\|u - u_h\|_0$	$\|u - u_h\|_1$	$\|u_I - u_h\|_A$	$\|u_I - u_h\|_\infty$	$h_{\min}$
4249	1.9326e-02	1.6337e-01	7.3234e+00	8.7266e-03	1.0451e-03
16953	4.8220e-03	8.6269e-02	2.2225e+00	2.9843e-03	5.2257e-04
67729	1.2256e-03	4.4058e-02	6.3928e-01	9.7703e-04	2.6129e-04
270753	3.0807e-04	2.2185e-02	1.7798e-01	3.1631e-04	1.3064e-04
Order	1.9855	0.9803	1.8223	1.6200	

Table 4.4: Errors measured in H^1, L^2, and L^∞ norms for Example 2.



(a) (b)

Fig. 4.4: The errors of Example 2. (a) The errors of $\|u - u_h\|_0$ and $|u - u_h|_1$. (b) The errors of $\|u_I - u_h\|_A$ and $\|u_I - u_h\|_\infty$.

In Table. 4.5, we also present the variation of iteration steps and computing time

with respect to Dofs for fixing $\beta^- = 4 + \sin(x + y)$ and varying β^+. For Example 2, since the analytic solution is not dependent on the β, the errors of $\|u - u_h\|_0$, $|u - u_h|_1$ and $\|u_I - u_h\|_\infty$ remain stable for different β^+. Here we just list the error of $\|u_I - u_h\|_A$ which is weighted by β_h. One can find that the error of $\|u_I - u_h\|_A$ increases with the increased β^+ and fixed Dofs, see Table. 4.6.

	$\beta^+ = 10 + r^2$		$\beta^+ = 100 + r^2$		$\beta^+ = 1000 + r^2$		$\beta^+ = 10000 + r^2$	
Dofs	#Iter	Time(s)	#Iter	Time(s)	#Iter	Time(s)	#Iter	Time(s)
4294	8	0.086	12	0.11	12	0.11	12	0.11
16953	9	0.16	12	0.19	13	0.21	12	0.19
67729	10	0.46	12	0.52	12	0.52	12	0.52
270753	10	1.9	12	2.2	12	2.2	12	2.2

Table 4.5: Example 2, the iterations and computing time of MGCG with fixed $\beta^- = 4 + \sin(x + y)$ and increased β^+.

Dofs	$\beta^+ = 10 + r^2$	$\beta^+ = 100 + r^2$	$\beta^+ = 1000 + r^2$	$\beta^+ = 10000 + r^2$
4249	2.435517e-01	7.356614e-01	2.316728e+00	7.323382e+00
16953	7.383809e-02	2.233129e-01	7.031052e-01	2.222478e+00
67729	2.121271e-02	6.422869e-02	2.022412e-01	6.392772e-01
270753	5.901919e-03	1.788140e-02	5.630591e-02	1.779810e-01
Order	1.8238	1.8224	1.8224	1.8223

Table 4.6: The error $\|u_I - u_h\|_A$ of Example 2 with fixed $\beta^- = 4 + \sin(x + y)$ and increased β^+.

5. Discussion and future work. we have improved Börgers' algorithm by newest vertex bisection refinement method, and make it more suitable for the interface with complex geometric details. At the same time, we maintain the hierarchical structure of the body-fitted mesh by a simple data structure. Based on this hierarchical structure, we use the coarsening algorithm in [12] to construct a sequence of nested meshes and then construct an efficient multigrid solver. By some superconvergence analysis, we immediately give an estimate in maximal norm to two dimensional interface problems.

For moving interface problems, we can efficiently adjust the mesh to fit the moved interface by the combination of the coarsening, bisection and the Börgers' algorithm, which will be our future work.

REFERENCES

[1] I. Babuška and A. K. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13(2):pp. 214–226, 1976.

[2] R. E. Bank and J. Xu. Asymptotically exact a posteriori error estimators, part I: Grids with superconvergence. *SIAM J. Numer. Anal.*, 41(6):pp. 2294–2312, 2003.

[3] J. Bedrossian, J. H. V. Brecht, S. Zhu, E. Sifakis, and J. M. Teran. A second order virtual node method for elliptic problems with interfaces and irregular domains. *J. Comput. Phys.*, 229(18):pp. 6405–6426, 2010.

[4] B. Bejanov, J. Guermond, and P. Minev. A grid-alignment finite element technique for incompressible multicomponent flows. *J. Comput. Phys.*, 227(13):pp. 6473–6489, 2008.

[5] C. Börgers. A triangulation algorithm for fast elliptic solvers based on domain imbedding. *SIAM J. Numer. Anal.*, 27(5):pp. 1187–1196, 1990.

[6] J. Bramble and J. King. A finite element method for interface problems in domains with smooth boundaries and interfaces. *Adv. Computat. Mathe.*, 6(1):109–138, 1996.

[7] L. Chen. Mesh smoothing schemes based on optimal Delaunay triangulations. In *13th International Meshing Roundtable*, pages 109–120, Williamsburg, VA, 2004. Sandia National Laboratories.
[8] L. Chen. *Robust and Accurate Algorithms for Solving Anisotropic Singularities*. PhD thesis, Department of Mathematics, The Pennsylvania State University, 2005.
[9] L. Chen. *i*FEM: An Integrated Finite Element Methods Package in MATLAB. *Technical Report, University of California at Irvine*, 2009.
[10] L. Chen and M. J. Holst. Efficient mesh optimization schemes based on Optimal Delaunay Triangulations. *Comput. Methods Appl. Mech. Engrg.*, 200(9-12):pp. 967–984, 2011.
[11] L. Chen and J. Xu. Topics on adaptive finite element methods. In T. Tang and J. Xu, editors, *in Adaptive Computations: Theory and Algorithms*, pages 1–31. Sci. Pr., Beijing, 2007.
[12] L. Chen and C. Zhang. A coarsening algorithm on adaptive grids by newest vertex bisection and its applications. *J. Comput. Math.*, 28(6):pp. 767–789, 2010.
[13] Z. Chen, Y. Xiao, and L. Zhang. The adaptive immersed interface finite element method for elliptic and maxwell interface problems. *J. Comput. Phys.*, 228(14):pp. 5000–5019, 2009.
[14] Z. Chen and J. Zou. Finite element methods and their convergence for elliptic and parabolic interface problems. *Numer. Math.*, 79(2):pp. 175–202, 1998.
[15] I. Chern and Y. Shu. A coupling interface method for elliptic interface problems. *J. Comput. Phys.*, 225(2):pp. 2138–2174, 2007.
[16] M. Ebeida, R. Davis, and R. Freund. A new fast hybrid adaptive grid generation technique for arbitrary two-dimensional domains. *Int. J. Numer. Methods Engrg.*, 84(3):pp. 305–329, 2010.
[17] P. Frey and L. Marechal. Fast adaptive quadtree mesh generation. In *Proceedings of the Seventh International Meshing Roundtable*. Citeseer, 1998.
[18] I. Fried. Condition of finite element matrices generated from nonuniform meshes. *AIAA J.*, 10:pp. 219–221, 1972.
[19] Y. Gong, B. Li, and Z. Li. Immersed-interface finite-element methods for elliptic interface problems with non-homogeneous jump conditions. *SIAM J. Numer. Anal*, 46(1):pp. 472–495, 2008.
[20] P. Knupp. Algebraic mesh quality metrics. *SIAM J. Sci. Comput.*, 23(1):193–218, 2001.
[21] Z. Li. A fast iterative algorithm for elliptic interface problems. *SIAM J. Numer. Anal.*, 35(1):pp. 230–254, 1998.
[22] Z. Li and K. Ito. Maximum principle preserving schemes for interface problems with discontinuous coefficients. *SIAM J. Sci. Comput.*, 23(1):pp. 339–361, 2001.
[23] Z. Li and K. Ito. *The immersed interface method: numerical solutions of PDEs involving interfaces and irregular domains*. SIAM, Philadelphia, PA, USA, 2006.
[24] Z. Li, T. Lin, and X. Wu. New cartesian grid methods for interface problems using the finite element formulation. *Numer. Math.*, 96(1):pp. 61–98, 2003.
[25] P. Persson and G. Strang. A simple mesh generator in matlab. *SIAM Rev.*, 46(2):pp. 329–345, 2004.
[26] C. Pflaum. Semi-unstructured grids. *Computing*, 67(2):pp. 141–166, 2001.
[27] J. Roĭtberg and Z. Šeftel. A theorem on homeomorphisms for elliptic systems and its applications. *Math. USSR Sbornik*, 7(3):pp. 439–465, 1969.
[28] J. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Comput. Geom.*, 22(1-3):pp. 21–74, 2002.
[29] E. Stein. *Singular integrals and differentiability properties of functions*, volume 30. Princeton Univ. Pr., 1970.
[30] H. Xie, K. Ito, Z. Li, and J. Toivanen. A finite element method for interface problems with locally modified triangulations. *Moving Interface Problems and Applications in Fluid Dynamics: January 8-March 1, 2007, the Institute for Mathematical Sciences, National University of Singapore*, 466:pp. 179–190, 2008.
[31] H. Xie, Z. Li, and Z. Qiao. A finite element method for elasticity interface problems with locally modified triangulations. *Int. J. Numer. Anal. Model.*, 8(2):pp. 189–200, 2011.
[32] J. Xu. Error estimates of the finite element method for the 2nd order elliptic equations with discontinuous coefficients. *J. Xiangtan University*, 1:pp. 1–5, 1982.
[33] M. Yerry and M. Shephard. A modified quadtree approach to finite element mesh generation. *IEEE Comput. Graph. Appl.*, 3(1):pp. 39–46, Jan. 1983.
[34] K. F. C. Yiu, D. M. Greaves, S. Cruz, A. Saalehi, and A. G. L. Borthwick. Quadtree grid generation: Information handling, boundary fitting and cfd applications. *Comput. Fluids*, 25(8):pp. 759–769, 1996.