Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **11**, No. 1, pp. 216-240

DOI: 10.4208/aamm.OA-2018-0099
February 2019

A Bilinear Petrov-Galerkin Finite Element Method for Solving Elliptic Equation with Discontinuous Coefficients

Liqun Wang[1], Songming Hou[2], Liwei Shi[3,*] and Ping Zhang[1]

[1] *Department of Mathematics, College of Science, China University of Petroleum (Beijing), Beijing 102249, China*
[2] *Department of Mathematics and Statistics, Louisiana Tech University, Ruston, LA 71272, USA*
[3] *Department of Science and Technology Teaching, China University of Political Science and Law, Beijing 102249, China*

Received 7 April 2018; Accepted (in revised version) 23 August 2018

Abstract. In this paper, a bilinear Petrov-Galerkin finite element method is introduced to solve the variable matrix coefficient elliptic equation with interfaces using non-body-fitted grid. Different cases the interface cut the cell are discussed. The condition number of the large sparse linear system is studied. Numerical results demonstrate that the method is nearly second order accurate in the L^∞ norm and L^2 norm, and is first order accurate in the H^1 norm.

AMS subject classifications: 65N30

Key words: Petrov-Galerkin finite element method, jump condition, bilinear.

1 Introduction

Interface problem has attracted much attention from researchers of various disciplines because it is involved in many research areas, such as environmental science, physics, fluid dynamics and biological mathematics. Elliptic problem with internal interfaces is the basic form of interface problem. The most challenging part for solving interface problem is that its governing equations has discontinuous coefficients at interfaces and sometimes singular source term exists. Standard finite element or finite difference method are not suitable for this situation if non-body-fitted grid is used. Designing highly accurate and efficient methods for these problems are desired. Since the pioneering work of Peskin in 1977 [1], a large number of numerical methods are designed to solve interface problems.

*Corresponding author.
Emails: wliqunhmily@gmail.com (L. Q. Wang), shou@latech.edu (S. M. Hou), sliweihmily@gmail.com (L. W. Shi), zhangpingby@126.com (P. Zhang)

The immersed boundary method was proposed by Peskin to study blood flow through heart valves [2]. This method uses a numerical approximation of the delta-function, which smears out the solution on a thin finite band around the interface. In [3], it was combined with the level set method, resulting in a first order numerical method that is simple to implement, even in multiple spatial dimensions. In [4–6], efforts were made to achieve higher order accuracy of the immersed boundary method. Second order convergence are obtained by considering the interaction of a viscous incompressible flow and an anisotropic incompressible viscoelastic shell. This method has been extensively used in engineering computations due to its simplicity, efficiency and robustness [7–9].

In [10–14], LeVeque and Li proposed the immersed interface method (IIM) to solve elliptic equations with discontinuous coefficients and singular source term. This method is based on the finite difference method under the Cartesian grid. Standard finite difference or finite element method is employed away from the interface, while the grid points or elements near the interface are amended by the interface condition. This method has been successfully applied to incompressible Stokes equation and Navier-Stokes equations with singular source term.

Wei et al. pays attention to interface problems with geometry singularity and developed the second order accurate method called matched interface and boundary method [15–17]. This method has been successfully applied to biomathematics research on molecular level. Besides, there are other numerical methods for interface problems from the finite difference/volume perspective, including the boundary condition capturing method [18], the embedded boundary method [19], the Cartesian grid method [41], and so on. The Cartesian grid method [41] developed by Johansen and Colella is a second-order finite volume method on Cartesian grids for the variable coefficient Poisson equation on irregular domains. In [40], the finite volume method is used to solve elliptic equations with variable and discontinuous coefficients. With nonhomogeneous jump conditions, the method can deliver a second order accurate result in the L^2 and L^∞ norm.

Researchers also propose some methods from the finite element perspective. In [20], Chen and Zou considered the finite element method with fitted mesh for solving second order elliptic and parabolic interface problems, sub-optimal error estimates can be achieved for smooth interfaces. The immersed finite element method [34–37] is based on uniform triangulations of Cartesian grids, while the local basis functions are constructed according to the interface jump conditions. In [33], the immersed finite element method is developed to solve elliptic interface problems with non-homogeneous jump conditions. The basic idea is to locally add piecewise polynomials that can approximate the non-homogeneous flux jump condition. For the adaptive immersed interface method [22] and the extended finite element method [23–25], the mesh generation does not rely on the interface, while the construction of finite element space relies on the jump condition of the interface. In [26,28], the non-traditional finite element method is proposed to solve interface problems with variable coefficient and sharp-edged interface. In [27,31,32], the method is further analyzed and extended to three dimensions. The non-traditional finite element method is simple and easy to implement, it is extended to solve elasticity interface problems in [29] and multi-domain interface problems in [30]. Gradient recovery technique is discussed in [42,43] to improve the accuracy of gradient of solution.

In [38], the approximation capability of a bilinear immersed finite element space is discussed, which is proved to be similar to that of the standard bilinear FE space. In [39], the algebraic multigrid solver is employed by the immersed interface method to get bilinear and linear solutions for both stationary and moving interface problems, optimal convergence in both L^2 and semi-H^1 norms is achieved. The unfitted finite element method [21] modifies the bilinear form near the interface by penalizing the jump of the solution value without general flux jump across the interface. Overmann et. al proposed the Cartesian grid finite volume method [40] for elliptic equations with variable, discontinuous coefficients and solution discontinuities on irregular domains, bilinear ansatz functions on Cartesian grids is employed in this method, second order of accuracy can be achieved in the L^∞ and L^2 norm.

In this paper, we propose a bilinear Petrov-Galerkin finite element method for solving the variable coefficient elliptic equation with interfaces. This is different from the bilinear Immersed Finite Element Method because the test and trial function basis are different. It is based on the ideas of our earlier work [26, 28] using triangular grids and it is the first time that our previous method is extended to rectangular grids. Our method is easy to implement for inhomogeneous jump conditions. The rest of this article is organized as follows. In Section 2, we present the bilinear Petrov Galerkin finite element method for elliptic interface problems. Different cases the interface cuts the grids are discussed in details. In Section 3, extensive numerical results are presented to demonstrate the accuracy of our method. We conclude with Section 4.

2 Formulation and numerical method

2.1 Problem definition and weak formulation

In this paper, we solve the elliptic equation with discontinuous variable matrix coefficients along the interface. Consider a rectangular domain $\Omega=(x_{\min},x_{\max})\times(y_{\min},y_{\max})$. Γ is an interface prescribed by the zero level-set $\{(x,y)\in\Omega|\phi(x,y)=0\}$ of a level-set function $\phi(x,y)$. The unit normal vector of Γ is

$$\boldsymbol{n}=\left[\begin{array}{c} n_x \\ n_y \end{array}\right]=\frac{\nabla\phi}{|\nabla\phi|}$$

pointing from $\Omega^-=\{(x,y)\in\Omega|\phi(x,y)<0\}$ to $\Omega^+=\{(x,y)\in\Omega|\phi(x,y)>0\}$, see Fig. 1. Now the governing equation reads

$$-\nabla\cdot(\beta(x,y)\nabla u(x,y))=f(x,y) \quad \text{in} \quad \Omega\backslash\Gamma, \tag{2.1}$$

in which ∇ is the gradient operator. The coefficient

$$\beta(x,y)=\left[\begin{array}{cc} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{array}\right]$$

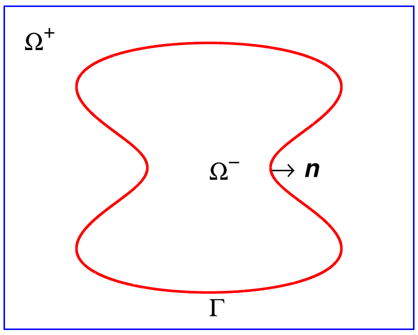


Figure 1: A rectangular domain $\Omega=\Omega^+\cup\Omega^-\cup\Gamma$.

is assumed to be a matrix that is uniformly elliptic on each disjoint subdomain, Ω^- and Ω^+, and its components are continuously differentiable on each disjoint subdomain, but they may be discontinuous across the interface Γ. The right-hand side $f(x,y)$ is assumed to lie in $L^2(\Omega)$.

Given functions a and b along the interface Γ, we prescribe the jump conditions on Γ

$$\begin{aligned} a(x,y) &= [u]_\Gamma(x,y) \equiv u^+(x,y)-u^-(x,y), &\text{(2.2a)}\\ b(x,y) &= [(\beta\nabla u)\cdot\boldsymbol{n}]_\Gamma(x,y) \\ &\equiv \boldsymbol{n}\cdot(\beta^+(x,y)\nabla u^+(x,y))-\boldsymbol{n}\cdot(\beta^-(x,y)\nabla u^-(x,y)), &\text{(2.2b)}\end{aligned}$$

the superscripts "$\pm$" refer to limits taken from within the subdomains $\Omega^\pm$.

Finally, we prescribe the boundary condition

$$u(x,y)=g(x,y) \tag{2.3}$$

for a given function g on the boundary $\partial\Omega$.

In this paper, we use non-traditional finite element method [27,28] to solve the elliptic equation with interface jump conditions described by Eqs. (2.1)-(2.3).

By multiplying both sides of Eq. (2.1) with the test function $\psi\in H_0^1(\Omega)$ and integrating over the subdomain Ω^+, we deduce from Green's theorem that

$$\begin{aligned} \int_\Omega f\psi &= \int_{\Omega^+\cup\Omega^-}\beta\nabla u\cdot\nabla\psi-\int_\Gamma\left(\beta^+\psi\frac{\partial u^+}{\partial\boldsymbol{n}^+}+\beta^-\psi\frac{\partial u^-}{\partial\boldsymbol{n}^-}\right)\\ &= \int_{\Omega^+\cup\Omega^-}\beta\nabla u\cdot\nabla\psi-\int_\Gamma\left(\beta^+\frac{\partial u^+}{\partial\boldsymbol{n}}-\beta^-\frac{\partial u^-}{\partial\boldsymbol{n}}\right)\psi\\ &= \int_{\Omega^+\cup\Omega^-}\beta\nabla u\cdot\nabla\psi+\int_\Gamma b\psi, \end{aligned}\tag{2.4}$$

where $-\boldsymbol{n}^+=\boldsymbol{n}^-=\boldsymbol{n}$.

Note that the derivation is similar to our previous work [28] using triangular grids. The main difference will be in the construction of local system to be shown when we present our numerical method.

2.2 Domain discretization

In this paper, we restrict ourselves to a rectangular domain $\Omega=(x_{\min},x_{\max})\times(y_{\min},y_{\max})$ in the plane. Given positive integers I and J, set $\Delta x=(x_{\max}-x_{\min})/I$ and $\Delta y=(y_{\max}-y_{\min})/J$. Define $(x_i,y_j)=(x_{\min}+i\Delta x,y_{\min}+j\Delta y)$ for $i=0,\cdots,I$ and $j=0,\cdots,J$ as a uniform Cartesian grid. Each (x_i,y_j) is called a grid point. For the case $i=0,I$ or $j=0,J$, a grid point is called a boundary point, otherwise it is called an interior point. If $\phi(x_i,y_j)<0$, we count the grid point (x_i,y_j) as in Ω^-; if $\phi(x_i,y_j)>0$, we count the grid point (x_i,y_j) as in Ω^+; otherwise we count it as on the interface Γ. The grid size is defined as $h=max(\Delta x,\Delta y)>0$, see Fig. 2. The rectangular mesh here has some advantage compared with the triangular mesh. For the same grid size, the number of cells is half of the triangular mesh. It will be convenient for adaptive refinement or multi-scale computation.

Two sets of grid functions are needed and they are denoted by

$$H^{1,h}=\{\omega^h=(\omega_{i,j}):0\le i\le I,\ 0\le j\le J\},$$

and

$$H_0^{1,h}=\{\omega^h=(\omega_{i,j})\in H^{1,h}:\omega_{i,j}=0 \text{ if } i=0,I \texttt{ or } j=0,J\}.$$

In this way, we obtain the rectangulation $\mathcal{T}_h$ of the domain Ω. There are two types of cells in our method: uniform cell and nonuniform cell. In order to define the interface cell, we need to have the two following assumptions:

1. In a cell, the interface is assumed to be a piecewise straight line segment and all of its end points are enforced to be on the edges of this cell.

2. In a cell, the interface can only have one intersection point with each edge of a cell.

According to the subdomains that the four vertices of a cell belong to, we will define the regular cell and nine kinds of interface cells. In reality, the interfaces are curves and

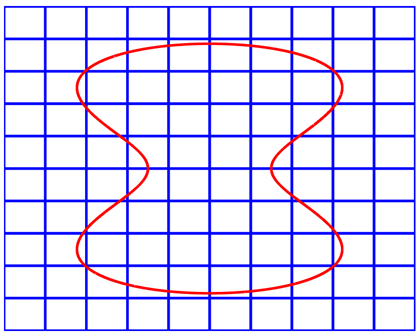


Figure 2: A uniform rectangulation.

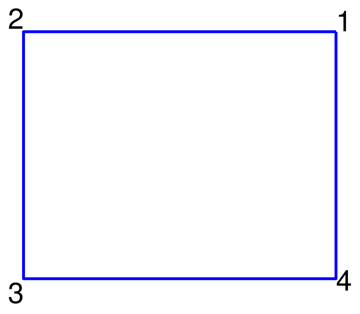


Figure 3: Regular cell.

could intersect the cell in many different ways. However, for targeting on proposing a second order accurate method, it is reasonable to make the above assumptions. A cell K is called a regular cell if all of its vertices belong to the same subdomain, see Fig. 3; otherwise it is called an interface cell. In this paper, we consider the elliptic equation with Dirichlet jump condition $[u]=a\neq 0$ along the interface Γ. Based on the assumptions above, there are nine kinds of interface cells, as shown in Fig. 4. In an interface cell, we write $K=K^+\bigcup K^-$. K^+ and K^- are approximations of the regions $K\bigcap\Omega^+$ and $K\bigcap\Omega^-$, respectively. K^+ and K^- are separated by a straight line segment, denoted by Γ_K^h. If a cell is divided into three parts by interfaces, like in Fig. 4(h), then we denote them by $K^{+,1}$ K^- and $K^{+,2}$ or $K^{-,1}$ K^+ and $K^{-,2}$. The two end points of the line segment Γ_K^h are located on interface Γ and their locations are calculated from the linear interpolations of the discrete level-set functions $\phi^h=\phi(x_i,y_j)$.

In the following discussion, two extension operators are needed. For any $\psi^h\in H_0^{1,h}$, define $T^h(\psi^h)$ as a standard continuous piecewise bilinear polynomial, which is a bilinear polynomial in each cell (both regular cell and interface cell), and $T^h(\psi^h)$ matches ψ^h on grid points. For any $u^h\in H^{1,h}$ with $u^h=g^h$ on boundary points, $U^h(u^h)$ is a piecewise bilinear polynomial and matches u^h on grid points. It is a bilinear polynomial in each regular cell, just like the first extension operator $U^h(u^h)=T^h(u^h)$ in a regular cell. In each interface cell, $U^h(u^h)$ consists of two or three pieces of bilinear polynomial, depending on the shape of interface in the cell. In cases (c) to (g) of Fig. 4, $U^h(u^h)$ consists of two pieces of bilinear polynomial, one is on K^+ and the other is on K^-. In cases (h) and (i) of Fig. 4, $U^h(u^h)$ consists of three pieces of bilinear polynomial; if $\phi(x_4,y_4)<0$, they are on $K^{+,1}$, K^- and $K^{+,2}$, otherwise they are on $K^{-,1}$, K^+ and $K^{-,2}$. The location of discontinuity in the interface cell is the straight line segment Γ_K^h. Note that two end points of the line segment are located on the interface Γ, and hence the interface condition $[u]=a$ could be and is enforced exactly at these two end points. In each interface cell, the interface condition $[\beta\nabla u\cdot n]=b$ is enforced with the value b at two end points of Γ_K^h, the same as in [40]. Note that we do not need to enforce the jump conditions everywhere on the interface. Redundant information would not improve the accuracy, as our method is already about 2nd order accurate in the L^∞ and L^2 norm.

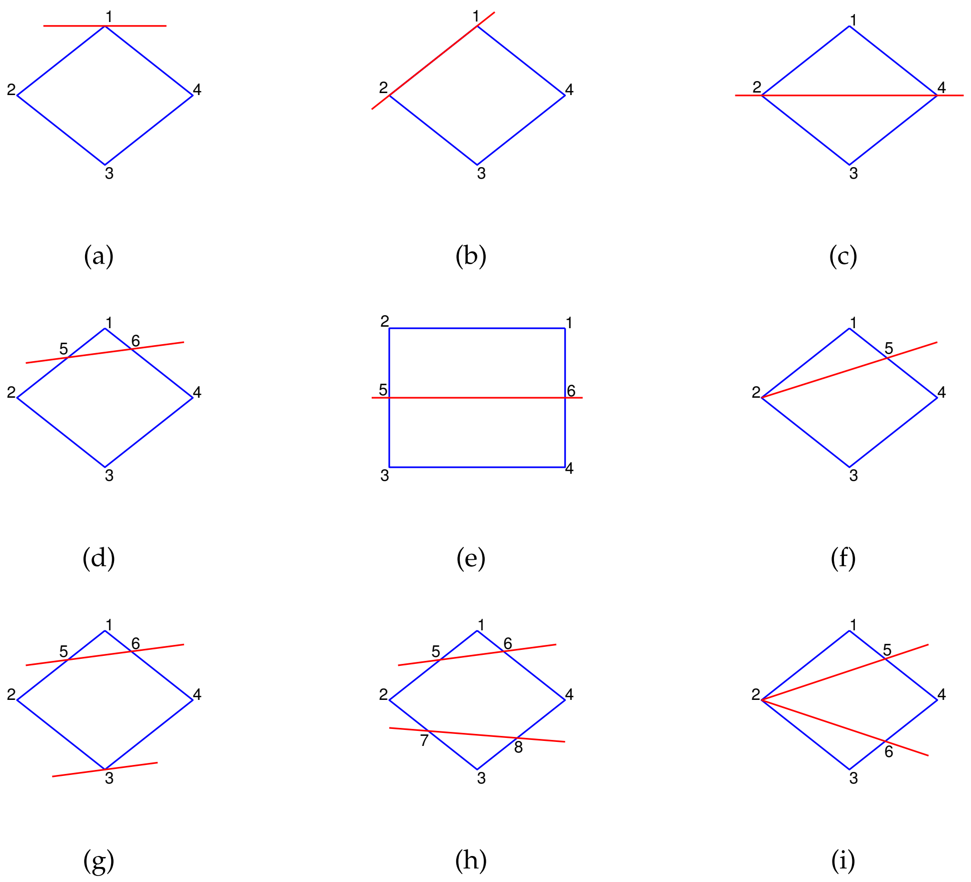


Figure 4: Nine cases of interface cells.

2.3 Numerical method

We shall construct an approximate solution to the interface problem taking into account the jump conditions. Note that the Neumann jump condition $[(\beta\nabla u)\cdot\boldsymbol{n}]=b$ along the interface Γ has already been absorbed into the weak formulation, hence, we only need to take care of the Dirichlet jump condition $[u]=a$ along the interface Γ. We shall seek an approximate solution which is continuous piecewise bilinear on both subdomains Ω^- and Ω^+, but discontinuous along the interface Γ when $[u]\neq 0$. Clearly, in cases (a), (b), (c), (f), (g) and (i) of Fig. 4, when a vertex (x,y) of the interface cell K is on the interface, we need to get two solutions

$$u^h(x,y)=\begin{cases} u^+(x,y), \\ u^-(x,y), \end{cases}$$

defined at the same point. To this end, we introduce a globally piecewise bilinear approximation $u^h(x,y)$ defined below:

$$u^h(x,y)=\begin{cases} u^+(x,y), & \text{if } \phi(x,y)\geq 0,\\ u^-(x,y), & \text{if } \phi(x,y)<0. \end{cases}$$

In this paper, we use standard bilinear finite element functions in all regular cells:

$$u(x,y)=c_1x+c_2y+c_3xy+c_4.$$

The four unknown coefficients c_1, c_2, c_3 and c_4 are uniquely determined by the four corner values of u, see Fig. 3. Given piecewise bilinear distribution of $u(x,y)$, the integration on the left hand side of Eq. (2.4) is straightforward.

For a regular cell K (see Fig. 3), no vertex or edge is on the interface, the cell belongs to a subdomain Ω^+ or Ω^-. For an interface cell as in Case 1 (see Fig. 4(a)), there is only one vertex on the interface, this cell K belongs to a subdomain Ω^+ or Ω^-, depending on which subdomain the other three vertices belong to. For these two kinds of cases, no extra treatment is needed.

In Case 2 (see Fig. 4(b) or any situation with at least one edge on the interface), the interface cuts the cell K along an edge, this cell belongs to a subdomain Ω^+ or Ω^-, depending on which subdomain the other two vertices belong to. For this kind of cell, we need to take the Neumann jump condition into consideration.

Special piecewise bilinear polynomials satisfying interface jump conditions are employed only in interface cells. In Cases 3-7 (see Figs. 4(c)-(g)), the interface cells are separated into two different pieces K^+ and K^- by the interface segment. The key idea of our method for treating these interface cells is to construct a piecewise function by two bilinear polynomials defined on K^+ and K^-,

$$u(x,y)=\begin{cases} c_1^+x+c_2^+y+c_3^+xy+c_4^+, & \text{if } (x,y)\in K^+,\\ c_1^-x+c_2^-y+c_3^-xy+c_4^-, & \text{if } (x,y)\in K^-. \end{cases}$$

In order to get the unknown coefficients c_1^+, c_2^+, c_3^+, c_4^+, c_1^-, c_2^-, c_3^- and c_4^- for different kinds of interface cells, different local systems need to be constructed. For simplicity, in the following discussion, we suppose $\phi(x_1,y_1)>0$ in Cases 3-9 (see Figs. 4(c)-(i)).

For ease of discussion, we define the following notation:

$$\begin{aligned} &d_1^{+,k}=n_x^k\beta_{11}^{+,k}+n_y^k\beta_{21}^{+,k}, \quad d_2^{+,k}=n_x^k\beta_{12}^{+,k}+n_y^k\beta_{22}^{+,k},\\ &d_3^{+,k}=(n_x^k\beta_{11}^{+,k}+n_y^k\beta_{21}^{+,k})y_k+(n_x^k\beta_{12}^{+,k}+n_y^k\beta_{22}^{+,k})x_k,\\ &d_1^{-,k}=n_x^k\beta_{11}^{-,k}+n_y^k\beta_{21}^{-,k}, \quad d_2^{-,k}=n_x^k\beta_{12}^{-,k}+n_y^k\beta_{22}^{-,k},\\ &d_3^{-,k}=(n_x^k\beta_{11}^{-,k}+n_y^k\beta_{21}^{-,k})y_k+(n_x^k\beta_{12}^{-,k}+n_y^k\beta_{22}^{-,k})x_k, \end{aligned}$$

where the superscripts k refer to the point p_k.

In Case 3, from the above assumption, the jump conditions are enforced at the two end points of the interface segment $l_{2,4}$, see Fig. 4(c). And then, the local system can be constructed as follows:

$$\begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 \\ x_2 & y_2 & x_2y_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_3 & y_3 & x_3y_3 & 1 \\ x_4 & y_4 & x_4y_4 & 1 & 0 & 0 & 0 & 0 \\ x_2 & y_2 & x_2y_2 & 1 & -x_2 & -y_2 & -x_2y_2 & -1 \\ x_4 & y_4 & x_4y_4 & 1 & -x_4 & -y_4 & -x_4y_4 & -1 \\ d_1^{+,2} & d_2^{+,2} & d_3^{+,2} & 0 & d_1^{-,2} & d_2^{-,2} & d_3^{-,2} & 0 \\ d_1^{+,4} & d_2^{+,4} & d_3^{+,4} & 0 & d_1^{-,4} & d_2^{-,4} & d_3^{-,4} & 0 \end{bmatrix} \begin{bmatrix} c_1^+ \\ c_2^+ \\ c_3^+ \\ c_4^+ \\ c_1^- \\ c_2^- \\ c_3^- \\ c_4^- \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ a_2 \\ a_4 \\ b_2 \\ b_4 \end{bmatrix}.$$

In Case 4, the jump conditions are enforced at the two end points of the interface segment $l_{5,6}$, see Fig. 4(d). And then, the local system can be constructed as follows:

$$\begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & x_2y_2 & 1 \\ 0 & 0 & 0 & 0 & x_3 & y_3 & x_3y_3 & 1 \\ 0 & 0 & 0 & 0 & x_4 & y_4 & x_4y_4 & 1 \\ x_5 & y_5 & x_5y_5 & 1 & -x_5 & -y_5 & -x_5y_5 & -1 \\ x_6 & y_6 & x_6y_6 & 1 & -x_6 & -y_6 & -x_6y_6 & -1 \\ d_1^{+,5} & d_2^{+,5} & d_3^{+,5} & 0 & d_1^{-,5} & d_2^{-,5} & d_3^{-,5} & 0 \\ d_1^{+,6} & d_2^{+,6} & d_3^{+,6} & 0 & d_1^{-,6} & d_2^{-,6} & d_3^{-,6} & 0 \end{bmatrix} \begin{bmatrix} c_1^+ \\ c_2^+ \\ c_3^+ \\ c_4^+ \\ c_1^- \\ c_2^- \\ c_3^- \\ c_4^- \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ a_5 \\ a_6 \\ b_5 \\ b_6 \end{bmatrix}.$$

In Case 5, the jump conditions are enforced at the two end points of the interface segment $l_{5,6}$, see Fig. 4(e). And then, the local system can be constructed as follows:

$$\begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 \\ x_2 & y_2 & x_2y_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_3 & y_3 & x_3y_3 & 1 \\ 0 & 0 & 0 & 0 & x_4 & y_4 & x_4y_4 & 1 \\ x_5 & y_5 & x_5y_5 & 1 & -x_5 & -y_5 & -x_5y_5 & -1 \\ x_6 & y_6 & x_6y_6 & 1 & -x_6 & -y_6 & -x_6y_6 & -1 \\ d_1^{+,5} & d_2^{+,5} & d_3^{+,5} & 0 & d_1^{-,5} & d_2^{-,5} & d_3^{-,5} & 0 \\ d_1^{+,6} & d_2^{+,6} & d_3^{+,6} & 0 & d_1^{-,6} & d_2^{-,6} & d_3^{-,6} & 0 \end{bmatrix} \begin{bmatrix} c_1^+ \\ c_2^+ \\ c_3^+ \\ c_4^+ \\ c_1^- \\ c_2^- \\ c_3^- \\ c_4^- \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ a_5 \\ a_6 \\ b_5 \\ b_6 \end{bmatrix}.$$

In Case 6, the jump conditions are enforced at the two end points of the interface segment $l_{2,5}$, see Fig. 4(f). And then, the local system can be constructed as follows:

$$\begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 \\ x_2 & y_2 & x_2y_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_3 & y_3 & x_3y_3 & 1 \\ 0 & 0 & 0 & 0 & x_4 & y_4 & x_4y_4 & 1 \\ x_2 & y_2 & x_2y_2 & 1 & -x_2 & -y_2 & -x_2y_2 & -1 \\ x_5 & y_5 & x_5y_5 & 1 & -x_5 & -y_5 & -x_5y_5 & -1 \\ d_1^{+,2} & d_2^{+,2} & d_3^{+,2} & 0 & d_1^{-,2} & d_2^{-,2} & d_3^{-,2} & 0 \\ d_1^{+,5} & d_2^{+,5} & d_3^{+,5} & 0 & d_1^{-,5} & d_2^{-,5} & d_3^{-,5} & 0 \end{bmatrix} \begin{bmatrix} c_1^+ \\ c_2^+ \\ c_3^+ \\ c_4^+ \\ c_1^- \\ c_2^- \\ c_3^- \\ c_4^- \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ a_2 \\ a_5 \\ b_2 \\ b_5 \end{bmatrix}.$$

In Case 7, the jump conditions are enforced at the two end points of the interface segment $l_{5,6}$, see Fig. 4(g). And then, the local system can be constructed as follows:

$$\begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & x_2y_2 & 1 \\ x_3 & y_3 & x_3y_3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_4 & y_4 & x_4y_4 & 1 \\ x_5 & y_5 & x_5y_5 & 1 & -x_5 & -y_5 & -x_5y_5 & -1 \\ x_6 & y_6 & x_6y_6 & 1 & -x_6 & -y_6 & -x_6y_6 & -1 \\ d_1^{+,5} & d_2^{+,5} & d_3^{+,5} & 0 & d_1^{-,5} & d_2^{-,5} & d_3^{-,5} & 0 \\ d_1^{+,6} & d_2^{+,6} & d_3^{+,6} & 0 & d_1^{-,6} & d_2^{-,6} & d_3^{-,6} & 0 \end{bmatrix} \begin{bmatrix} c_1^+ \\ c_2^+ \\ c_3^+ \\ c_4^+ \\ c_1^- \\ c_2^- \\ c_3^- \\ c_4^- \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ a_5 \\ a_6 \\ b_5 \\ b_6 \end{bmatrix}.$$

By solving the local systems of Cases 3-7, we get all the coefficients, c_1^+, c_2^+, c_3^+, c_4^+, c_1^-, c_2^-, c_3^- and c_4^-.

In Cases 8 and 9 (see Figs. 4(h)-(i)), the interface cell is separated into three different pieces $K^{+,1}$, K^- and $K^{+,2}$ by two interface segments. The key idea of our method for treating these interface cells is to construct a piecewise function by three bilinear polynomials defined on $K^{+,1}$, K^- and $K^{+,2}$,

$$u(x,y)=\begin{cases} c_1^{+,1}x+c_2^{+,1}y+c_3^{+,1}xy+c_4^{+,1}, & \text{if } (x,y)\in K^{+,1}, \\ c_1^-x+c_2^-y+c_3^-xy+c_4^-, & \text{if } (x,y)\in K^-, \\ c_1^{+,2}x+c_2^{+,2}y+c_3^{+,2}xy+c_4^{+,2}, & \text{if } (x,y)\in K^{+,2}. \end{cases}$$

In Case 8, as the assumption above, the jump conditions are enforced at the two end points of the interface segments $l_{5,6}$ and $l_{7,8}$, see Fig. 4(h). Let

$$A_l=\begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & x_2y_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_3 & y_3 & x_3y_3 & 1 \\ 0 & 0 & 0 & 0 & x_4 & y_4 & x_4y_4 & 1 & 0 & 0 & 0 & 0 \\ x_5 & y_5 & x_5y_5 & 1 & -x_5 & -y_5 & -x_5y_5 & -1 & 0 & 0 & 0 & 0 \\ x_6 & y_6 & x_6y_6 & 1 & -x_6 & -y_6 & -x_6y_6 & -1 & 0 & 0 & 0 & 0 \\ d_1^{+,5} & d_2^{+,5} & d_3^{+,5} & 0 & d_1^{-,5} & d_2^{-,5} & d_3^{-,5} & 0 & 0 & 0 & 0 & 0 \\ d_1^{+,6} & d_2^{+,6} & d_3^{+,6} & 0 & d_1^{-,6} & d_2^{-,6} & d_3^{-,6} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -x_7 & -y_7 & -x_7y_7 & -1 & x_7 & y_7 & x_7y_7 & 1 \\ 0 & 0 & 0 & 0 & -x_8 & -y_8 & -x_8y_8 & -1 & x_8 & y_8 & x_8y_8 & 1 \\ 0 & 0 & 0 & 0 & d_1^{-,7} & d_2^{-,7} & d_3^{-,7} & 0 & d_1^{+,7} & d_2^{+,7} & d_3^{+,7} & 0 \\ 0 & 0 & 0 & 0 & d_1^{-,8} & d_2^{-,8} & d_3^{-,8} & 0 & d_1^{+,8} & d_2^{+,8} & d_3^{+,8} & 0 \end{bmatrix},$$

$$C_l=\begin{bmatrix} c_1^{+,1}\\ c_2^{+,1}\\ c_3^{+,1}\\ c_4^{+,1}\\ c_1^{-}\\ c_2^{-}\\ c_3^{-}\\ c_4^{-}\\ c_1^{+,2}\\ c_2^{+,2}\\ c_3^{+,2}\\ c_4^{+,2} \end{bmatrix},\quad B_l=\begin{bmatrix} u_1\\ u_2\\ u_3\\ u_4\\ a_5\\ a_6\\ b_5\\ b_6\\ a_7\\ a_8\\ b_7\\ b_8 \end{bmatrix}.$$

Then the local system can be constructed as follows:

$$A_l*C_l=B_l.$$

In Case 9, the interface segments $l_{2,5}$ and $l_{2,6}$ both have one endpoint on p_2, it is a little bit complicated than the interface cell of case 8. We choose two middle points p_{25} and p_{26} on these two interface segments. From the above assumption, the jump conditions are enforced at the two middle points p_{25}, p_{26} and two end points p_5, p_6. Let

$$A_l=\begin{bmatrix}
x_1 & y_1 & x_1y_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
x_2 & y_2 & x_2y_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_2 & y_2 & x_2y_2 & 1\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_3 & y_3 & x_3y_3 & 1\\
0 & 0 & 0 & 0 & x_4 & y_4 & x_4y_4 & 1 & 0 & 0 & 0 & 0\\
x_5 & y_5 & x_5y_5 & 1 & -x_5 & -y_5 & -x_5y_5 & -1 & 0 & 0 & 0 & 0\\
x_{25} & y_{25} & x_{25}y_{25} & 1 & -x_{25} & -y_{25} & -x_{25}y_{25} & -1 & 0 & 0 & 0 & 0\\
d_1^{+,5} & d_2^{+,5} & d_3^{+,5} & 0 & d_1^{-,5} & d_2^{-,5} & d_3^{-,5} & 0 & 0 & 0 & 0 & 0\\
d_1^{+,25} & d_2^{+,25} & d_3^{+,25} & 0 & d_1^{-,25} & d_2^{-,25} & d_3^{-,25} & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & -x_6 & -y_6 & -x_6y_6 & -1 & x_6 & y_6 & x_6y_6 & 1\\
0 & 0 & 0 & 0 & -x_{26} & -y_{26} & -x_{26}y_{26} & -1 & x_{26} & y_{26} & x_{26}y_{26} & 1\\
0 & 0 & 0 & 0 & d_1^{-,6} & d_2^{-,6} & d_3^{-,6} & 0 & d_1^{+,6} & d_2^{+,6} & d_3^{+,6} & 0\\
0 & 0 & 0 & 0 & d_1^{-,26} & d_2^{-,26} & d_3^{-,26} & 0 & d_1^{+,26} & d_2^{+,26} & d_3^{+,26} & 0
\end{bmatrix},$$

$$C_l = \begin{bmatrix} c_1^{+,1} \\ c_2^{+,1} \\ c_3^{+,1} \\ c_4^{+,1} \\ c_1^{-} \\ c_2^{-} \\ c_3^{-} \\ c_4^{-} \\ c_1^{+,2} \\ c_2^{+,2} \\ c_3^{+,2} \\ c_4^{+,2} \end{bmatrix}, \quad B_l = \begin{bmatrix} u_1 \\ u_2 \\ u_2 \\ u_3 \\ u_4 \\ a_5 \\ a_{25} \\ b_5 \\ b_{25} \\ a_6 \\ a_{26} \\ b_6 \\ b_{26} \end{bmatrix}.$$

Then the local system can be constructed as follows:

$$A_l * C_l = B_l.$$

Notice that the system we defined above is an overdetermined system, we use least squares method to solve this system.

By solving the local system of Cases 8 and 9, we can get all the coefficients $c_1^{+,1}$, $c_2^{+,1}$, $c_3^{+,1}$, $c_4^{+,1}$, c_1^{-}, c_2^{-}, c_3^{-}, c_4^{-}, $c_1^{+,2}$, $c_2^{+,2}$, $c_3^{+,2}$ and $c_4^{+,2}$.

Based on the above discussion, we propose the following method:

Method 2.1. Find a discrete function $u^{h,n} \in H^{1,h}$ with $u^{h,n} = g^{h,n}$ on boundary points such that for all $\psi^h \in H_0^{1,h}$, we have

$$\begin{aligned} &\sum_{K\in\mathcal{T}_h} \left(\int_{K^+} \beta \nabla U^h(u^h)\cdot\nabla T^h(\psi^h) + \int_{K^-} \beta\nabla U^h(u^h)\cdot\nabla T^h(\psi^h) \right) \\ =&\sum_{K\in\mathcal{T}_h} \left(\int_{K^+} f T^h(\psi^h) + \int_{K^-} f T^h(\psi^h) - \int_{\Gamma_K^h} b T^h(\psi^h) \right). \end{aligned} \tag{2.5}$$

Remark 2.1. In our implementation, the integrals are computed with Gaussian quadrature rules. Since the interface can separate a cell into many different polygon, when we are calculating the integral, we further cut it into a few triangles. For these triangles, the midpoint of each edge is denoted by p_{ij}. In numerical computation, the average of three $f(p_{ij})$ in each cell is utilized.

3 Numerical experiments

In this section, we present some numerical examples with known exact solutions to demonstrate the accuracy of our method. In all numerical experiments below, the level-set function $\phi(x,y)$, the coefficients $\beta^{\pm}(x,y)$ and the solutions

$$u=\begin{cases} u^{+}(x,y) & \text{in } \Omega^{+}, \\ u^{-}(x,y) & \text{in } \Omega^{-}, \end{cases}$$

are given. Hence

$$\begin{aligned}
&g(x,y)=u(x,y) && \text{on } \partial\Omega,\\
&f(x,y)=-\nabla\cdot(\beta(x,y)\nabla u(x,y)) && \text{in } \Omega\backslash\Gamma,\\
&a(x,y)=u^{+}(x,y)-u^{-}(x,y) && \text{on } \Gamma,\\
&b(x,y)=(\beta^{+}(x,y)\nabla u^{+}(x,y))\cdot\boldsymbol{n}-(\beta^{-}(x,y)\nabla u^{-}(x,y))\cdot\boldsymbol{n} && \text{on } \Gamma.
\end{aligned}$$

All the examples are defined on the domain $[-1,1]\times[-1,1]$. In the first two examples, all errors in solutions are measured in the L^{∞} norm in the whole domain Ω. In Example 3.1, errors in solutions are measured in the L^{∞} norm, L^{2} norm and H^{1} norm.

Example 3.1. This example is taken from [28]. In this example, the solution has large oscillation. We compare three kinds of interfaces (star, face and chess) to find out how the error changes when the interface gets more and more complicated. The solutions $u^{\pm}$ and the coefficients $\beta^{\pm}$ are given as follows:

$$\begin{aligned}
&\beta^{+}(x,y)=1,\\
&\beta^{-}(x,y)=2+\sin(x+y),\\
&u^{+}(x,y)=6+\sin(2\pi x)\sin(2\pi y),\\
&u^{-}(x,y)=x^{2}+y^{2}+\sin(x+y).
\end{aligned}$$

Case 1. When the level-set function $\phi(x,y)$ is given as

$$\begin{aligned}
&\phi(r,\theta)=\frac{R\sin(\theta_t/2)}{\sin(\theta_t/2+\theta-\theta_r-2\pi(i-1)/5)}-r,\\
&\theta_r+\pi(2i-2)/5\le\theta<\theta_r+\pi(2i-1)/5,\\
&\phi(r,\theta)=\frac{R\sin(\theta_t/2)}{\sin(\theta_t/2-\theta+\theta_r-2\pi(i-1)/5)}-r,\\
&\theta_r+\pi(2i-3)/5\le\theta<\theta_r+\pi(2i-2)/5,
\end{aligned}$$

with $\theta_t=\pi/5$, $\theta_r=\pi/7$, $R=6/7$ and $i=1,2,3,4,5$.

The numerical result with different grids are shown in Table 1. Figs. 5(a) and (b) show the numerical result and the numerical error of this example with a grid of 32×32. Figs. 5(c)-(e) shows the numerical errors of different grids and Fig. 5(f) shows the condition numbers of different grids.

Table 1: Result of Example 3.1 with the interface of "star" shape.

$n_x \times n_y$	$\|u-u^h\|_\infty$	Order	$\|u-u^h\|_2$	Order	$\|u-u^h\|_1$	Order
20×20	0.3123		0.1309		6.1307	
40×40	0.0810	1.9467	0.0335	1.9666	3.1778	0.9480
80×80	0.0210	1.9575	0.0083	2.0185	1.6178	0.9740
160×160	0.0054	1.9562	0.0021	1.9974	0.8157	0.9879
320×320	0.0014	1.9813	0.0005	1.9988	0.4090	0.9959

Table 2: Result of Example 3.1 with the interface of "face" shape.

$n_x \times n_y$	$\|u-u^h\|_\infty$	Order	$\|u-u^h\|_2$	Order	$\|u-u^h\|_1$	Order
20×20	0.3636		0.1887		10.3995	
40×40	0.0808	2.1709	0.0535	1.8179	5.3899	0.9482
80×80	0.0203	1.9947	0.0138	1.9507	2.7392	0.9765
160×160	0.0051	2.0036	0.0035	1.9988	1.3771	0.9921
320×320	0.0013	1.9915	0.0009	2.0040	0.6899	0.9973

Case 2. When the level-set function $\phi(x,y)$ is given as

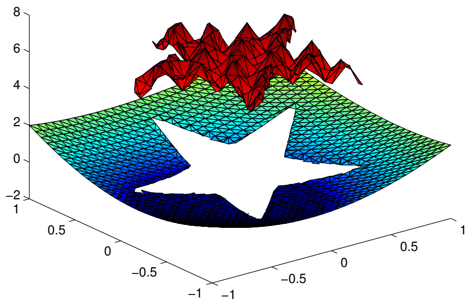
$$\begin{aligned}
&\phi(x,y)=\max(\min(\phi_1,\phi_2,\phi_3),\phi_4,\phi_5,\phi_6,\min(\phi_7,\phi_8)),\\
&\phi_1(x,y)=x^2+y^2-0.75^2-0.15^2,\\
&\phi_2(x,y)=(x-0.75)^2+y^2-0.15^2,\\
&\phi_3(x,y)=(x+0.75)^2+y^2-0.15^2,\\
&\phi_4(x,y)=-\frac{0.1}{0.12}(x-0.2)^2-\frac{0.12}{0.1}(y-0.22)^2+0.12\cdot 0.1,\\
&\phi_5(x,y)=-\frac{0.1}{0.12}(x+0.2)^2-\frac{0.12}{0.1}(y-0.22)^2+0.12\cdot 0.1,\\
&\phi_6(x,y)=-x^2-(y+0.08)^2+0.12^2,\\
&\phi_7(x,y)=-x^2-(y+0.625)^2+0.425^2,\\
&\phi_8(x,y)=-x^2-(y+0.25)^2+0.2^2.
\end{aligned}$$

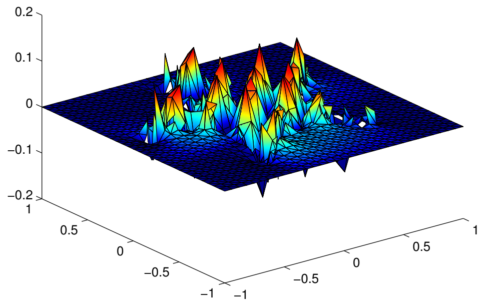
The numerical result with different grids are shown in Table 2. Figs. 6(a) and (b) show the numerical result and the numerical error of this example with a grid of 32×32. Figs. 6(c)-(e) shows the numerical errors of different grids and Fig. 6(f) shows the condition numbers of different grids.

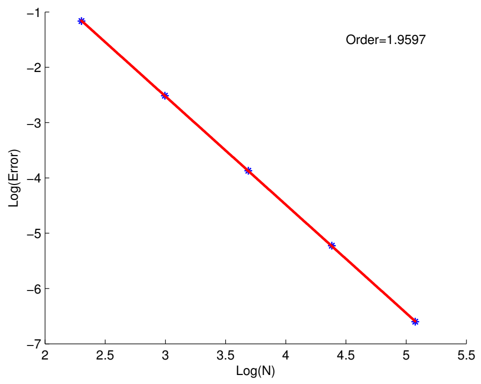
Case 3. When the level-set function $\phi(x,y)$ is given as

$$\phi(x,y)=-(\sin(5\pi x)-y)(\sin(5\pi y)+x).$$

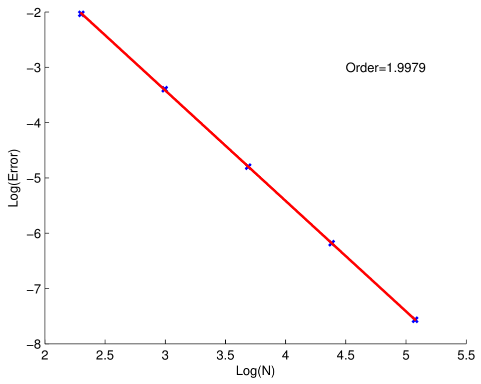
The numerical result with different grids are shown in Table 3. Figs. 7(a) and (b) show the numerical result and the numerical error of this example with a grid of 32×32. Figs. 7(c)-


(a) Numerical result

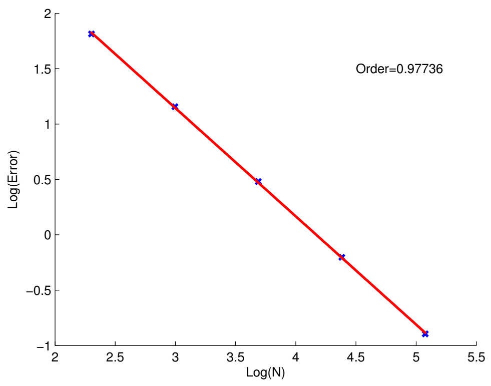

(b) Error



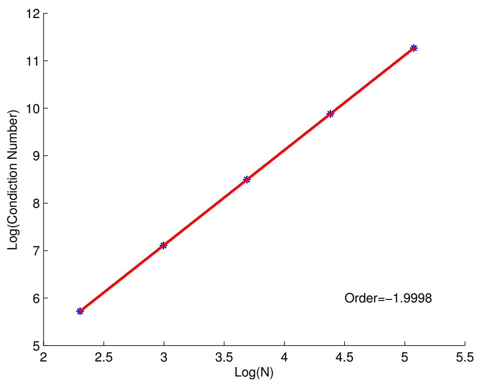
(c) L^{∞} norm



(d) L^{2} norm

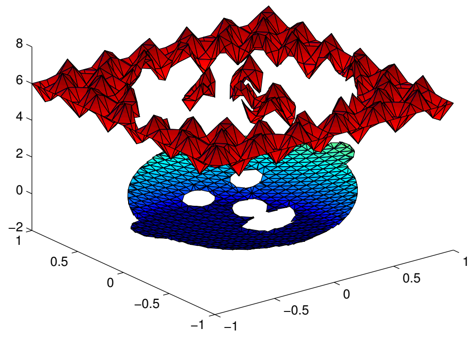


(e) H^{1} norm

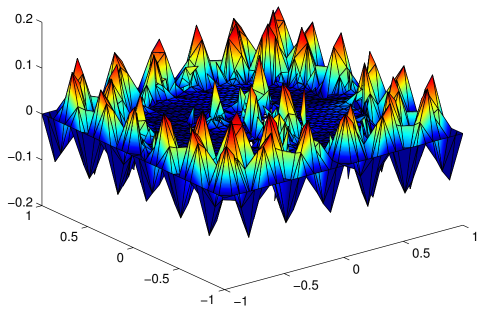


(f) Condition numbers

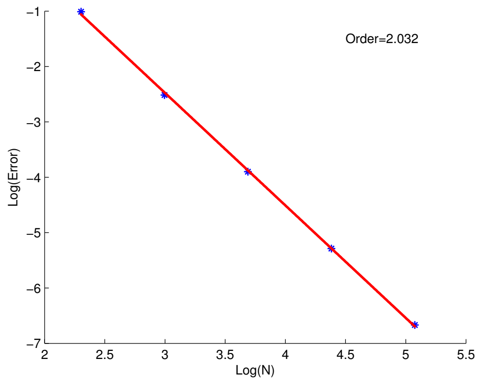
Figure 5: Numerical solution of Example 3.1 with the interface of "star" shape.



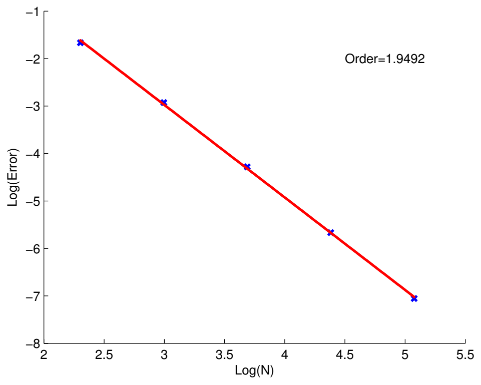
(a) Numerical result



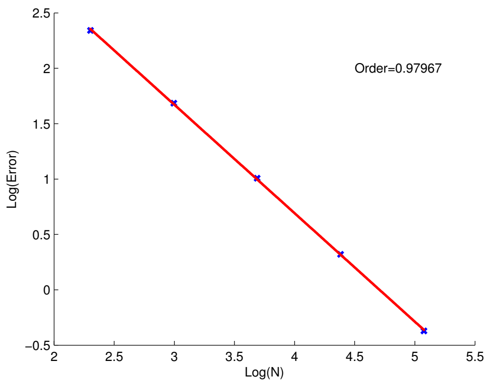
(b) Error



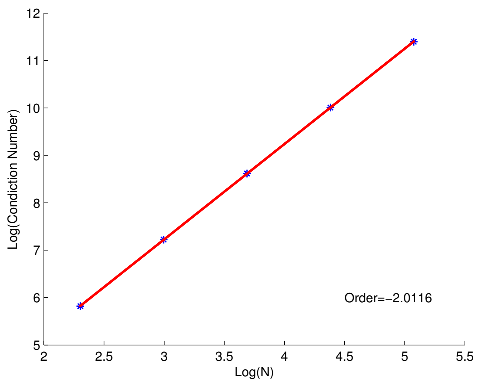
(c) L^{∞} norm



(d) L^2 norm



(e) H^1 norm



(f) Condition numbers

Figure 6: Numerical solution of Example 3.1 with the interface of "face" shape.

Table 3: Result of Example 3.1 with the interface of "chess board" shape.

$n_x \times n_y$	$\|u-u^h\|_\infty$	Order	$\|u-u^h\|_2$	Order	$\|u-u^h\|_1$	Order
22×22	0.2714		0.1444		7.3924	
44×44	0.0754	1.8479	0.0406	1.8313	4.3890	0.7522
88×88	0.0207	1.8654	0.0109	1.9026	2.2744	0.9484
176×176	0.0053	1.9523	0.0028	1.9697	1.1529	0.9803
352×352	0.0013	2.0295	0.0007	2.0036	0.5798	0.9917

(e) shows the numerical errors of different grids in L^∞ norm, L^2 norm and H^1 norm. Fig. 7(f) shows the condition numbers of different grids.

The interface geometry in this example is highly complicated, and the solution has large oscillation. Numerical results show that our method can achieve second order accuracy in the L^∞ norm and L^2 norm, and first order accuracy in the H^1 norm. The condition numbers of this example grow with order $\mathcal{O}(n^2)$, which is the same as the case without interface, and therefore a desired result.

Example 3.2. In this example, we compared three kinds of coefficient β (variable diagonal matrix, variable symmetric matrix and variable non-symmetric matrix) to find out how the error changes when the coefficients gets more and more complicated. The level-set function $\phi(x,y)$, and the solutions $u^\pm$ are given as follows:

$$\begin{aligned}
&\phi(x,y)=(x+1)^2-y-1,\\
&u^+(x,y)=\exp(2x+y^2-3),\\
&u^-(x,y)=x^2+\sin(y^2).
\end{aligned}$$

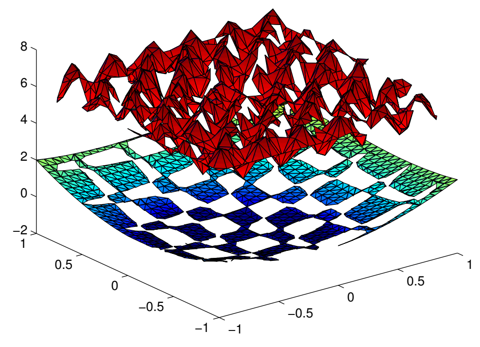
Case 1. When the coefficients $\beta^\pm$ are given as follows:

$$\begin{aligned}
\beta^+(x,y)&=\begin{pmatrix} xy+3 & 0\\ 0 & xy+5\end{pmatrix},\\
\beta^-(x,y)&=\begin{pmatrix} x^2-y^2+4 & 0\\ 0 & x^2-y^2+6\end{pmatrix}.
\end{aligned}$$

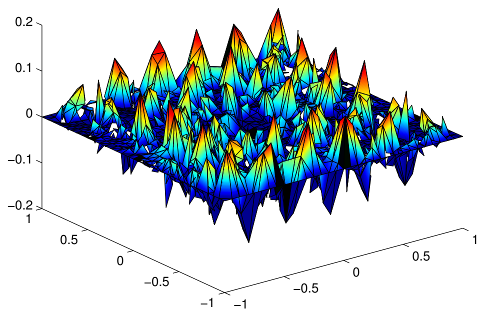
The numerical result with different grids are shown in Table 4. Figs. 8(a) and (b) show the numerical result and the numerical error of this example with a grid of 32×32. Figs. 8(c)-(e) shows the numerical errors of different grids in L^∞ norm, L^2 norm and H^1 norm. And Fig. 8(f) shows the condition numbers of different grids.

Case 2. When the coefficients $\beta^\pm$ are given as follows:

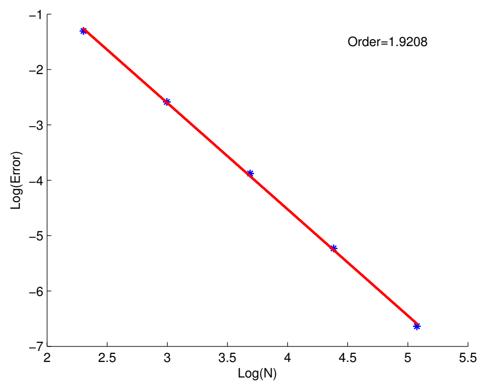
$$\begin{aligned}
\beta^+(x,y)&=\begin{pmatrix} xy+3 & \sin(xy)+1\\ \sin(xy)+1 & xy+5\end{pmatrix},\\
\beta^-(x,y)&=\begin{pmatrix} x^2-y^2+4 & x^2-y^2+1\\ x^2-y^2+1 & x^2-y^2+6\end{pmatrix}.
\end{aligned}$$



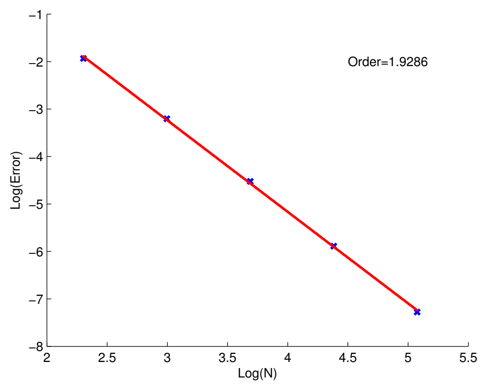
(a) Numerical result



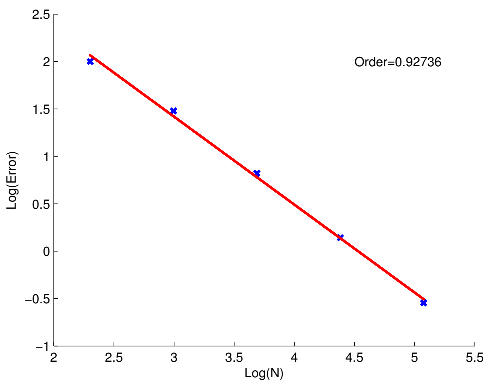
(b) Error



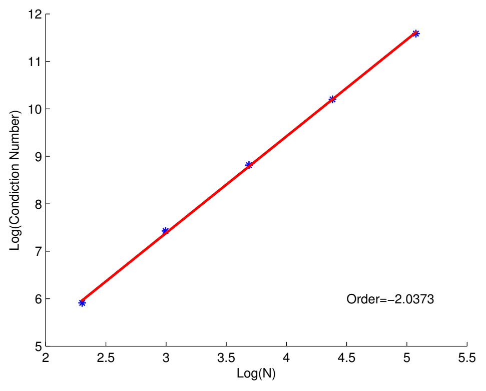
(c) L^∞ norm



(d) L^2 norm



(e) H^1 norm



(f) Condition numbers

Figure 7: Numerical solution of Example 3.1 with the interface of "chess board" shape.

Table 4: Result of Example 3.2, case 1.

$n_x \times n_y$	$\|u-u^h\|_\infty$	Order	$\|u-u^h\|_2$	Order	$\|u-u^h\|_1$	Order
20×20	1.8194e-3		9.7926e-4		1.2229e-1	
40×40	5.2993e-4	1.7796	2.6214e-4	1.9014	6.1101e-2	1.0011
80×80	1.4172e-4	1.9027	6.4786e-5	2.0166	3.0513e-2	1.0018
160×160	3.7821e-5	1.9058	1.5854e-5	2.0308	1.5246e-2	1.0010
320×320	9.7812e-6	1.9511	3.9865e-6	1.9916	7.6206e-3	1.0005

Table 5: Result of Example 3.2, case 2.

$n_x \times n_y$	$\|u-u^h\|_\infty$	Order	$\|u-u^h\|_2$	Order	$\|u-u^h\|_1$	Order
20×20	1.8934e-3		9.6459e-4		1.2205e-1	
40×40	5.1117e-4	1.8891	2.5260e-4	1.9330	6.1030e-2	0.9999
80×80	1.3785e-4	1.8907	6.2491e-5	2.0152	3.0496e-2	1.0009
160×160	3.6303e-5	1.9249	1.5432e-5	2.0178	1.5242e-2	1.0005
320×320	9.3939e-6	1.9503	3.8713e-6	1.9950	7.6196e-3	1.0003

Table 6: Result of Example 3.2, case 3.

$n_x \times n_y$	$\|u-u^h\|_\infty$	Order	$\|u-u^h\|_2$	Order	$\|u-u^h\|_1$	Order
20×20	1.3923e-3		9.9508e-4		1.2234e-1	
40×40	5.4198e-4	1.3611	2.6567e-4	1.9052	6.1102e-2	1.0016
80×80	1.5626e-4	1.7943	6.8053e-5	1.9649	3.0515e-2	1.0017
160×160	4.1812e-5	1.9020	1.6828e-5	2.0158	1.5247e-2	1.0010
320×320	1.0609e-5	1.9786	4.1008e-6	2.0369	7.6208e-3	1.0005

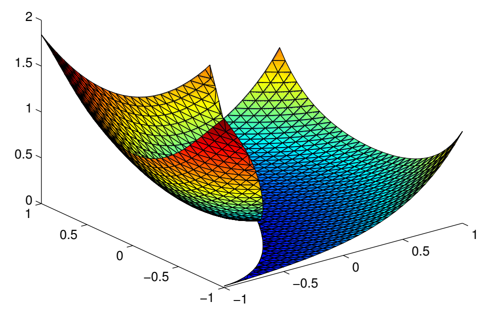
The numerical result with different grids are shown in Table 5. Figs. 9(a) and (b) show the numerical results and the numerical error of this example with a grid of 32×32. Figs. 9(c)-(e) shows the numerical errors of different grids and Fig. 9(f) shows the condition numbers of different grids.

Case 3. When the coefficients $\beta^{\pm}$ are given as follows:

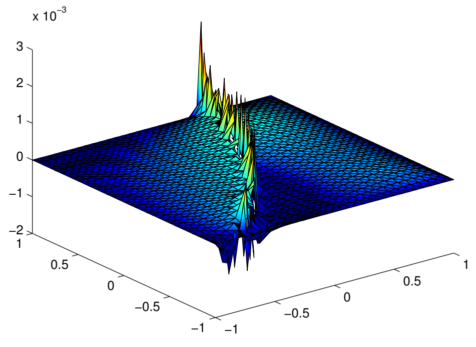
$$\beta^+(x,y)=\begin{pmatrix} xy+3 & \cos x \\ \sin y & xy+5 \end{pmatrix},$$
$$\beta^-(x,y)=\begin{pmatrix} x^2-y^2+4 & y \\ x & x^2-y^2+6 \end{pmatrix}.$$

The numerical result with different grids are show in Table 6. Fig. 10(a) shows the numerical result of this example with a grid of 32×32. Figs. 10(c)-(e) shows the numerical errors of different grids and Fig. 10(f) shows the condition numbers of different grids.

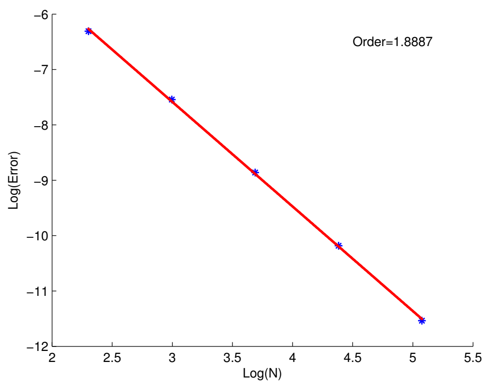
In this example, we validate our method by solving a problem with different matrix coefficients, including diagonal matrices, symmetric matrices and non-symmetric matrices. The numerical results show that our method can achieve second order accuracy in



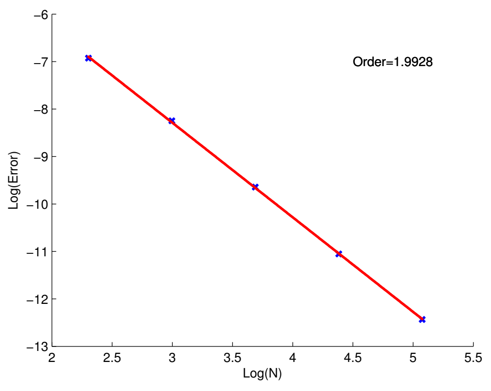
(a) Numerical result



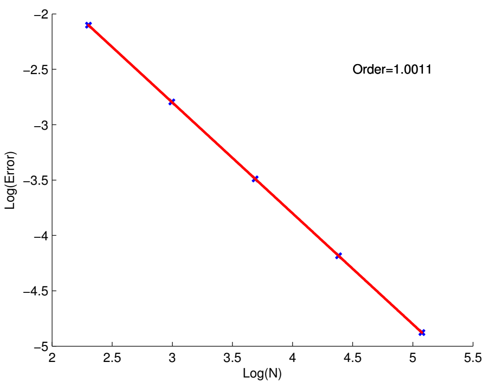
(b) Error



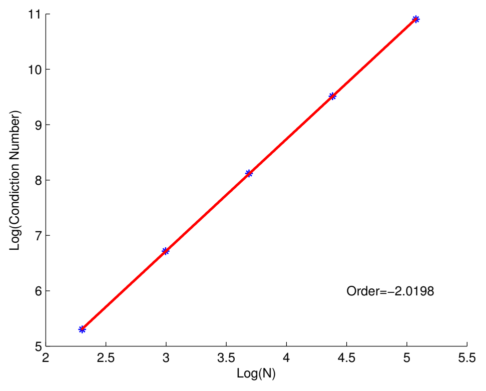
(c) L^{∞} norm



(d) L^2 norm

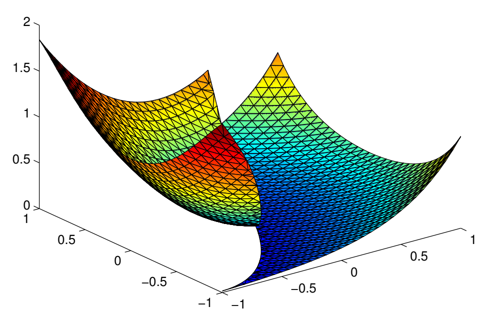


(e) H^1 norm

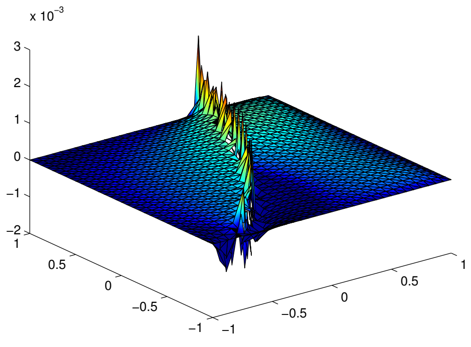


(f) Condition numbers

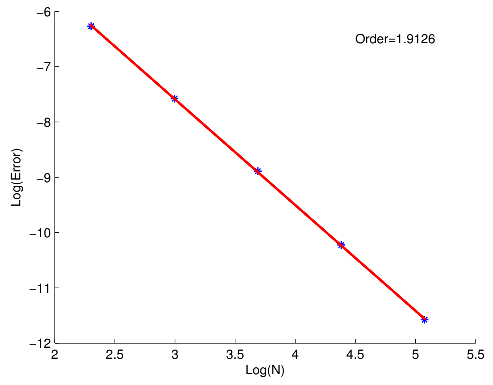
Figure 8: Numerical solution of Example 3.2, case 1.



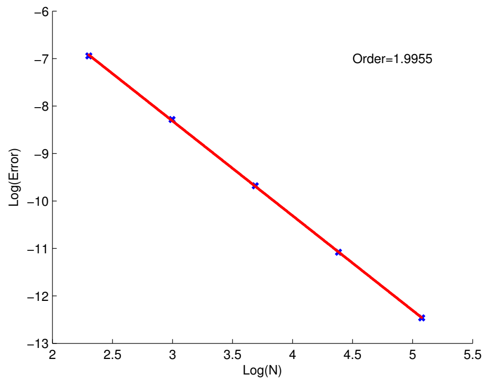
(a) Numerical result



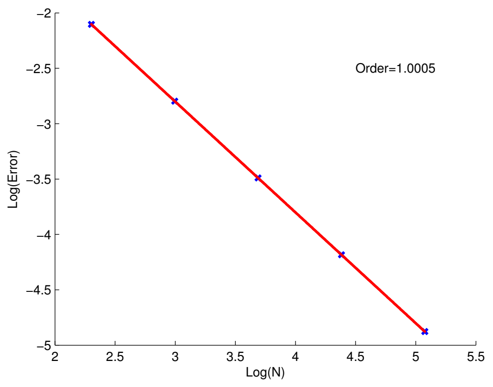
(b) Error



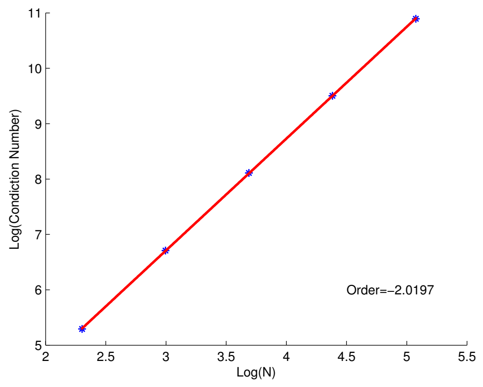
(c) L^∞ norm



(d) L^2 norm

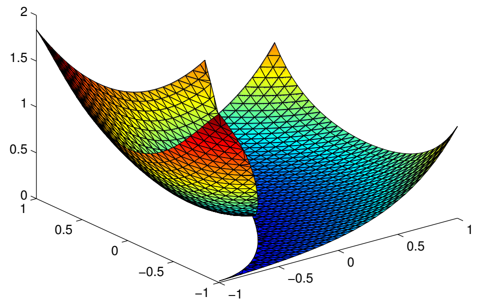


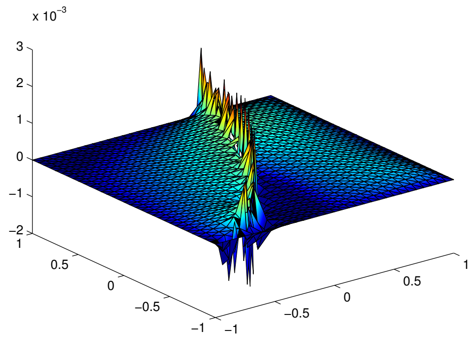
(e) H^1 norm

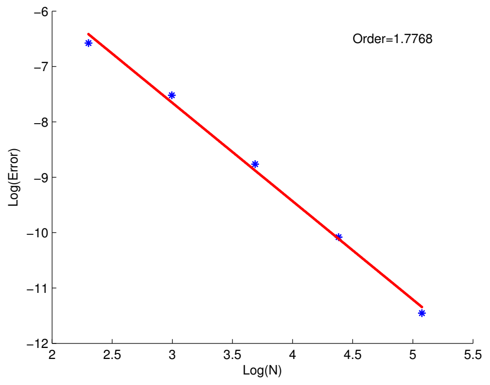


(f) Condition numbers

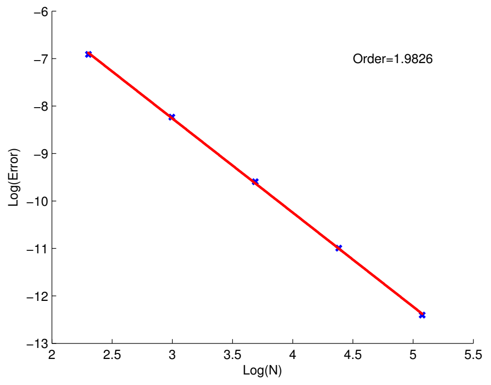
Figure 9: Numerical solution of Example 3.2, case 2.


(a) Numerical result

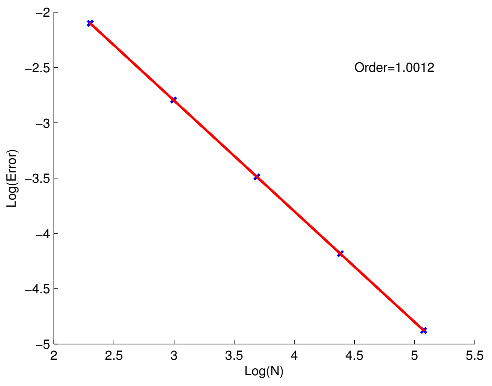

(b) Error



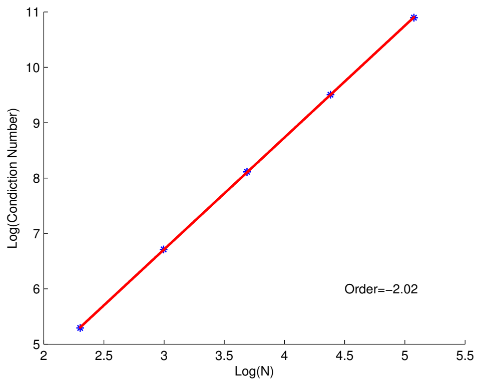
(c) L^∞ norm



(d) L^2 norm



(e) H^1 norm



(f) Condition numbers

Figure 10: Numerical solution of Example 3.2, case 3.

the L^∞ norm and L^2 norm, and first order accuracy in the H^1 norm. Again the condition number growth is the same as the case without the interface.

4 Conclusions

In the paper, we proposed a bilinear Petrov-Galerkin finite element method for solving the variable matrix coefficient elliptic interface problems. Compared with our previous work on the triangular elements, the new contribution of this paper is to handle the more complicated ways by which interface cut the rectangular cell. For the same grid size, the number of cells is half of the triangular mesh. The mesh is non-body-fitted. The method is easy to implement and can be further generalized in the future to solve more complicated problems. Numerical experiments demonstrate nearly second order accuracy in the L^∞ norm and L^2 norm, and first order accuracy in the H^1 norm. The accuracy and computational cost are comparable with the Petrov-Galerkin Finite Element Method using triangles and piecewise bilinear basis functions with non-body-fitted grids.

Acknowledgements

The author would like to thank the referees for the helpful suggestions. L. Shi's research is supported by National Natural Science Foundation of China (No. 11701569). L. Wang's research is supported by Science Foundation of China University of Petroleum–Beijing (No. 2462015BJB05). S. Hou's research is supported by Dr. Walter Koss Endowed Professorship. This professorship is made available through the State of Louisiana Board of Regents Support Funds.

References

[1] C. Peskin, *Numerical analysis of blood flow in the heart,* J. Comput. Phys., 25 (1977), pp. 220–252.

[2] C. Peskin, *The immersed boundary method,* Acta Numer., 11 (2001), pp. 479–517.

[3] M. Sussman, P. Smereka and S. Osher, *A level set approach for computing solutions to incompressible two-phase flow,* J. Comput. Phys., 114 (1994), pp. 146–154.

[4] B. E. Griffith and C. S. Peskin, *On the order of accuracy of the immersed boundary method: Higher order convergence rates for sufficiently smooth problems,* J. Comput. Phys., 208 (2005), pp. 75–105.

[5] M. C. Lai and C. S. Peskin, *An immersed boundary method with formal second-order accuracy and reduced numerical viscosity,* J. Comput. Phys., 160 (2000), pp. 705–719.

[6] Y. Mori and C. S. Peskin, *Implicit second order immersed boundary methods with boundary mass,* Comput. Methods Appl. Mech., 197 (2008), pp. 2049–2067.

[7] E. A. Fadlun, R. Verzicco, P. Orlandi and J. Mohd-Yusof, *Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations,* J. Comput. Phys., 161 (2000), pp. 30–60.

[8] M. FRANCOIS AND W. SHYY, *Computations of drop dynamics with the immersed boundary method, part 2: Drop impact and heat transfer,* Numer. Heat Trans. Part B-Fund., 44 (2003), pp. 119–143.

[9] G. IACCARINO AND R. VERZICCO, *Immersed boundary technique for turbulent flow simulations,* Appl. Mech. Rev., 56 (2003), pp. 331–347.

[10] R. J. LEVEQUE AND Z. LI, *The immersed interface method for elliptic equations with discontinuous coefficients and singular sources,* SIAM J. Numer. Anal., 31 (1994), pp. 1019–1044.

[11] K. ITO, Z. LI AND Y. KYEI, *Higher-order, cartesian grid based finite difference schemes for elliptic equations on irregular domains,* SIAM J. Sci. Comput., 27(1) (2005), pp. 346–367.

[12] R. J. LEVEQUE AND Z. LI, *The immersed interface method for elliptic equations with discontinuous coefficients and singular sources,* SIAM J. Numer. Anal., 34 (1994), pp. 1019–1044.

[13] Z. LI, *The immersed interface method using a finite element formulation,* Applied Numer. Math., 27 (1998), pp. 253–267.

[14] Z. LI AND K. ITO, *Maximum principle preserving schemes for interface problems with discontinuous coefficients,* SIAM J. Sci. Comput., 23 (2001), pp. 339–361.

[15] S. YU, Y. ZHOU AND G. WEI, *Matched interface and boundary (MIB) method for elliptic problems with sharp-edged interfaces,* J. Comput. Phys., 224 (2007), pp. 729–756.

[16] K. XIA, M. ZHAN AND G. WEI, *The matched interface and boundary (mib) method for multi-domain elliptic interface problems,* J. Comput. Phys., 230 (2011), pp. 8231–8258.

[17] Y. ZHOU, S. ZHAO, M. FEIG AND G. WEI, *High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular sources,* J. Comput. Phys., 213 (2006), pp. 1–30.

[18] X. D. LIU, R. P. FEDKIW AND M. KANG, *A boundary condition capturing method for Poisson's equation on irregular domains,* J. Comput. Phys., 160 (2000), pp. 151–178.

[19] D. W. HEWITT, *The embedded curved boundary method for orthogonal simulation meshes,* J. Comput. Phys., 138 (1997), pp. 585–616.

[20] Z. CHEN AND J. ZOU, *Finite element methods and their convergence for elliptic and parabolic interface problems,* Numer. Math., 79(2) (1998), pp. 175–202.

[21] A. HANSBO AND P. HANSBO, *An unfitted finite element method, based on nitsche's method, for elliptic interface problems,* Comput. Methods Appl. Mech. Eng., 191(47-48) (2002), pp. 5537–5552.

[22] Z. CHEN, Y. XIAO AND L. ZHANG, *The adaptive immersed interface finite element method for elliptic and Maxwell interface problems,* J. Comput. Phys., 228 (2009), pp. 5000–5019.

[23] N. MOES, J. DOLBOW AND T. BELYTSCHKO, *A finite element method for crack growth without remeshing,* Int. J. Numer. Methods Eng., 46(1) (1999), pp. 131–150.

[24] G. J. WAGNER, N. MOES, W. K. LIU AND T. BELYTSCHKO, *The extended finite element method for rigid particles in stokes flow,* Int. J. Numer. Methods Eng., 51(3) (2001), pp. 293–313.

[25] J. CHESSA AND T. BELYTSCHKO, *An extended finite element method for two-phase fluids: flow simulation and modeling,* J. Appl. Mech., 70(1) (2003), pp. 10–17.

[26] S. HOU AND X. LIU, *A numerical method for solving variable coefficient elliptic equations with interfaces,* J. Comput. Phys., 202 (2005), pp. 411–445.

[27] S. HOU, P. SONG, L. WANG AND H. ZHAO, *A weak formulation for solving elliptic interface problems without body fitted grid,* J. Comput. Phys., 249 (2013), pp. 80–95.

[28] S. HOU, W. WANG AND L. WANG, *Numerical method for solving matrix coefficient elliptic equation with sharp-edged interfaces,* J. Comput. Phys., 229 (2010), pp. 7162–7179.

[29] S. HOU, Z. LI, L. WANG AND W. WANG, *A numerical method for solving elasticity equations with interfaces,* Commun. Comput. Phys., 12(2) (2012), pp. 595–612.

[30] S. Hou, L. Wang and W. Wang, *A numerical method for solving the elliptic interface problem with multi-domains and triple junction points,* J. Comput. Math., 30(5) (2012), pp. 504–516.

[31] L. Wang, S. Hou and L. Shi, *An improved non-traditional finite element formulation for solving three-dimensional elliptic interface problems,* Comput. Math. Appl., 73 (2017), pp. 374–384.

[32] L. Wang, S. Hou and L. Shi, *A numerical method for solving three-dimensional elliptic interface problems with triple junction points,* Adv. Comput. Math., 44 (2018), pp. 175–193.

[33] X. He, T. Lin and Y. Lin, *Immersed finite element methods for elliptic interface problems with non-homogeneous jump conditions,* Int. J. Numer. Anal. Model., 8(2) (2011), pp. 284–301.

[34] Y. Gong and Z. Li, *Immersed interface finite element methods for elasticity interface problems with non-homogeneous jump conditions,* Numer. Math. Theory Methods Appl., 3 (2010), pp. 23–39.

[35] S. Adjerid and T. Lin, *A p-th degree immersed finite element for boundary value problems with discontinuous coefficients,* Appl. Numer. Math., 59(6) (2009), pp. 1303–1321.

[36] Z. Li, T. Lin, Y. Lin and R. C. Rogers, *An immersed finite element space and its approximation capability,* Numer. Methods Partial Differential Equations, 20(3) (2004), pp. 338–367.

[37] Z. Li, T. Lin and X. Wu, *New Cartesian grid methods for interface problems using the finite element formulation,* Numer. Math, 96(1) (2003), pp. 61–98.

[38] X. He, T. Lin, and Y. Lin, *Approximation capability of a bilinear immersed finite element space,* Numer. Methods Partial Differential Equations, 24(5) (2008), pp. 1265–1300.

[39] W. Feng, X. He, Y. Lin and X. Zhang, *Immersed finite element method for interface problems with algebraic multigrid solver,* Commun. Comput. Phys., 15(4) (2014), pp. 1045–1067.

[40] M. Oevermann and R. Klein, *A Cartesian grid finite volume method for elliptic equations with variable coefficients and embedded interfaces,* J. Comput. Phys., 219 (2006), pp. 749–769.

[41] H. Johansen and P. Colella, *A cartesian grid embedded boundary method for poisson's equation on irregular domains,* J. Comput. Phys., 147 (1998), pp. 60–85.

[42] H. Guo and X. Yang, *Gradient recovery for elliptic interface problem: I. body-fitted mesh,* To appear in Commun. Comput. Phys., 2017.

[43] H. Guo and X. Yang, *Gradient recovery for elliptic interface problem: II. immersed finite element methods,* J. Comput. Phys., 338 (2017), pp. 606–619.