Computers and Mathematics with Applications 68 (2014) 719–745

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

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

A Galerkin formulation of the MIB method for three dimensional elliptic interface problems

Kelin Xia [a], Guo-Wei Wei [a,b,*]

[a] Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA

[b] Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA

ARTICLE INFO

Article history:
Received 20 January 2014
Received in revised form 16 May 2014
Accepted 21 July 2014
Available online 18 August 2014

Keywords:
Elliptic interface problems
Galerkin formulation
Matched interface and boundary
Nonsmooth interfaces
Low regularity solutions
Proteins and multiprotein complex

ABSTRACT

We develop a three dimensional (3D) Galerkin formulation of the matched interface and boundary (MIB) method for solving elliptic partial differential equations (PDEs) with discontinuous coefficients, i.e., the elliptic interface problem. The present approach builds up two sets of elements respectively on two extended subdomains which both include the interface. As a result, two sets of elements overlap each other near the interface. Fictitious solutions are defined on the overlapping part of the elements, so that the differentiation operations of the original PDEs can be discretized as if there was no interface. The extra coefficients of polynomial basis functions, which furnish the overlapping elements and solve the fictitious solutions, are determined by interface jump conditions. Consequently, the interface jump conditions are rigorously enforced on the interface. The present method utilizes Cartesian meshes to avoid the mesh generation in conventional finite element methods (FEMs). We implement the proposed MIB Galerkin method with three different elements, namely, rectangular prism element, five-tetrahedron element and six-tetrahedron element, which tile the Cartesian mesh without introducing any new node. The accuracy, stability and robustness of the proposed 3D MIB Galerkin are extensively validated over three types of elliptic interface problems. In the first type, interfaces are analytically defined by level set functions. These interfaces are relatively simple but admit geometric singularities. In the second type, interfaces are defined by protein surfaces, which are truly arbitrarily complex. The last type of interfaces originates from multiprotein complexes, such as molecular motors. Near second order accuracy has been confirmed for all of these problems. To our knowledge, it is the first time for an FEM to show a near second order convergence in solving the Poisson equation with realistic protein surfaces. Additionally, the present work offers the first known near second order accurate method for C^1 continuous or H^2 continuous solutions associated with a Lipschitz continuous interface in a 3D setting.

1. Introduction

Material interfaces are ubiquitous in nature as well as in man-made structures, devices and equipments. Mathematical modeling of material interface often leads to elliptic equations with discontinuous interfaces and singular sources. This class of problems is commonly called elliptic interface problems. The development of numerical algorithms for elliptic interface

* Corresponding author at: Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA. Tel.: +1 517 353 4689; fax: +1 517 432 1562.

E-mail address: wei@math.msu.edu (G.-W. Wei).

http://dx.doi.org/10.1016/j.camwa.2014.07.022
0898-1221/

problems has been an important field in the past few decades. Indeed, without complex interface and geometric boundary, it is very easy to construct numerical schemes for solving partial differential equations (PDEs) in two or three-dimensional (3D) settings by using any of commonly known mathematical techniques, including finite difference, finite element, finite volume, wavelet, radial basis function, meshless and spectral methods. However, none of these ordinary numerical techniques works well directly for elliptic interface problems. Special interface schemes are required to handle interface jump conditions, which are necessary for the well-posedness of the interface problem. To this end, Peskin pioneered the immersed boundary method (IBM) in 1977 [1–3]. The IBM has been very popular in the numerical simulation of structure-flow interactions and many other engineering problems. In 1984, Mayo introduced an integral equation approach for this class of problems [4,5]. The immersed interface method (IIM) proposed by LeVeque and Li in 1994 is the first second order elliptic interface scheme that does not smear the interface jump [6–8]. In the past fifteen years, a few other interesting elliptic interface methods have been reported, including the ghost fluid method (GFM) proposed by Fedkiw, Osher and coworkers [9,10], finite-volume-based methods [11], a virtual node method [12] and a piecewise-polynomial discretization that utilizes a Krylov-accelerated multigrid solver to speed up the convergence [13]. Recently, Beale and Layton have provided a proof of the second order convergence of the IIM for smooth interfaces [14]. However, rigorous convergence analysis of most other interface schemes is yet to be developed.

Complementary to finite difference and finite volume methods, finite element methods (FEMs) are also a class of efficient approaches to the elliptic interface problems. The first FEM solution of elliptic interface problems was constructed by Babuška in 1970 [15]. In the past forty years, a variety of new FEM approaches has been developed for elliptic interface problems [16–20]. With extra number of degrees of freedom, non-conforming FEM [17] and discontinuous Galerkin (DG) FEM [21,18] appear naturally suitable for solving elliptic equations with discontinuous coefficients. Recently, a novel Galerkin formulation, the Wang–Ye Galerkin FEM, has been proposed for solving PDEs [22]. Unlike other FEMs, the Wang–Ye Galerkin method employs *a discrete gradient* in the finite element procedure, which leads to desirable flexibility in dealing with geometric complexities, boundary conditions and interface jump conditions. Indeed, the Wang–Ye Galerkin FEM has found its success in solving a class of difficult elliptic interface problems [23]. According to the topological relation between elements and the interface, FEM based elliptic interface methods can be categorized into two major classes: interface-fitted FEMs and immersed FEMs. Interface-fitted FEMs, or body-fitted FEMs, allocate unstructured element meshes to align with the interface [24,25,17,21,18]. This is a natural approach to complex interfaces and irregular boundaries as standard h-refinement techniques, such as a priori and/or a posteriori error estimation based local mesh refinements, can be employed to improve the accuracy. The performance of interface-fitted FEMs is mainly impacted by the Galerkin formulation and the quality of element meshes near the interface. However, in this type of methods, mesh generation and refinement can be a technically demanding and computationally time consuming process in 3D when the interface frequently changes its shape during the time evolution or numerical iteration. Additionally, many numerical algorithms, such as geometric multigrid methods, are not automatically applicable to unstructured meshes. Nevertheless, rigorous mathematical analysis of FEM based interface schemes has been established, which enhances our understanding of Galerkin FEM approximations [19,21,26,27].

To avoid mesh generation processes, immersed FEMs, or embedded FEMs, have been introduced to allow the use of simple structured Cartesian meshes for solving elliptic interface problems in the Galerkin framework [16,28–31]. In this type of approaches, the interface must cut through elements in general so that appropriate interface algorithms which are similar to those used in the finite-difference based elliptic interface methods are required to deal with embedded interface geometries. Consequently, the performance of embedded FEMs typically depends on the effectiveness of interface schemes for complex interface geometries. One may regard embedded FEMs as the Galerkin formulations of finite difference based interface schemes. A major advantage of this type of methods is their computational efficiency for problems involving evolving interface geometries. However, it is not easy for embedded FEMs to perform the h-refinement locally.

In the past decade, we have developed matched interface and boundary (MIB) method [32–36] for solving elliptic interface problems. Based on high order Lagrange polynomials, the MIB method is of arbitrarily high-order accuracy in principle. MIB schemes up to 16th order accurate have been constructed for simple interface geometry [34,35], and sixth-order accurate MIB schemes have been developed for complex interfaces in two-dimensional (2D) [35] and three-dimensional (3D) domains [33,35]. Some essential ideas behind the MIB method, i.e., the use of fictitious nodes and the iterative use of low order interface jump conditions, were introduced respectively in our earlier work on the discrete singular convolution [37,38] and on the solution of Maxwell's equations [34]. A comparison of the GFM, IIM and MIB approaches is discussed in our earlier work [35,36]. The development of the MIB methodology is motivated by the practical needs in scientific and engineering applications, such as optical molecular imaging [39], nano-electronic devices, [40], vibration analysis of plates [41], wave propagation [42,43], geodynamics [44] and electrostatic potential in proteins [45–48].

Indeed, elliptic interface problems have a variety of applications in science and engineering, including fluid dynamics [49–55], electromagnetic wave propagation [56–59,34,42], materials science, [60,61] and biological systems [62,45–48]. The past few decades have witnessed intensive research effort in interface problems [63–68,61,69–71,59,54,72–80]. These references are just the tip of a iceberg of whole literature. There have been numerous advances in elliptic interface problems in the past decades. As a result, the research in interface methods and their applications has evolved into a well defined field in applied and computational mathematics, due to its mathematical complexity and practical importance in science and engineering.

As a well defined field, it is important to understand its status of the art and standing open problems so as to further advance the field. At present, the existing knowledge in the literature regarding the basic methodology makes it very easy

to construct second order accurate elliptic interface schemes for complex but smooth interfaces in 2D and 3D domains, based on any of well-known standard approaches, namely, finite difference, finite volume, finite element, meshless, and special methods. Nonetheless, there are still many unsolved open problems in the field. One of these open problems is to develop higher order interface schemes motivated by applications associated with high frequency waves. Well known examples include the vibration analysis of engineering structures, the propagation and scattering of electromagnetic/acoustic waves, shock–vortex interactions in compressible fluid flows, turbulence and combustion. The current status is that sixth order 3D schemes were reported for spherical interfaces [33]. However, to design high order convergence interface schemes for arbitrarily complex interface geometries in 3D domains is still a challenge. The construction of sixth-order 3D interface schemes for arbitrarily curved smooth interfaces has been posed as a standing open problem in the field [33].

Another problem in the elliptic interface research is the development of second order or higher order schemes for elliptic problems with nonsmooth interfaces, such as interfaces with Lipschitz continuity or C^1 continuity [13,69,30,32,33,46,81]. Most of earlier elliptic interface methods are developed for smooth interfaces, which fail on nonsmooth interfaces. Nonsmooth interfaces, also being referred as geometric singularities [81,82,32,33,46], have received relatively less attention in the field. However, in many practical problems, such as electromagnetic devices, molecular surfaces in biological systems, and nano-technology, one frequently encounters geometric singularities. Numerically, geometric singularities constrain the available information that could be utilized for designing interface schemes. Therefore, it is much more difficult to achieve high order convergence in dealing with nonsmooth interfaces. The first known second order accurate scheme for 2D nonsmooth interfaces was constructed using the MIB method [32]. Since then, a few other interesting second order methods have also been reported for this class of problems in 2D [13,30,81,23]. Currently, the best results in 2D domains in the literature are still limited to the second order convergence for arbitrarily nonsmooth interfaces [13,30,32,81,83,23]. As such, another open problem in the field is the construction of robust 2D interface schemes of numerical orders higher than two for arbitrarily nonsmooth interfaces. In fact, it is not clear whether such a scheme is feasible at all due to the constrained information in geometric singularities.

Since most realistic applications are in 3D settings, the development of high order schemes for nonsmooth interfaces in 3D domains is one of most important tasks in elliptic interface problems. For instance, the electrostatic potentials of biomolecules, such as proteins and DNA, are described by the Poisson–Boltzmann equation in the implicit solvent theory. The solvent excluded surfaces of proteins and other biomolecules admit geometric singularities, including sharp edges, cusps and self-intersecting surfaces [46,47]. The practical needs in biomolecular modeling have been a major driven force for the development of second order MIB schemes for 3D elliptic PDEs with arbitrarily non-smooth interfaces or geometric singularities [33]. Such schemes have found success in the electrostatic analysis of biomolecular systems [46–48,84,85] and light propagation in biomedical imaging [39]. Mathematically, it is relatively easier to deal with geometric singularities in a 3D setting than in a 2D setting because higher dimension often offers more geometric information or more geometric flexibility. Therefore, a fourth order MIB scheme for 3D elliptic PDEs with nonsmooth interfaces has also been constructed [33]. However, the success of such MIB schemes is limited to just a few special interface geometries. Consequently, one of the most rewarding open problems in the field is to develop robust fourth order schemes for arbitrarily non-smooth interfaces in 3D domains.

In many physical situations, the aforementioned geometric singularities induce solution singularities, i.e., divergence of the solution or its gradient of the governing equation at the geometric singularity. This problem originates naturally from electromagnetic theory—the electric field diverges near the geometric singularities, such as tips of electrodes, antenna and elliptic cones, and sharp edges of planar conductors. The associated physical phenomenon is known as tip-geometry effects. These effects are often utilized in the design of electric devices, such as atomic force microscopy and lightning rod, also called lightning attractor. Mathematical modeling of electrostatic potentials lead to the Poisson equation, whose solution has a lower regularity, specifically, the gradient of the solution may not exist at geometric singularities. Technically, the elliptic interface problems with geometric singularity induced solution singularity are more challenging to handle for obvious reasons. For this class of problems, different mathematical formulations make a vital difference in their performances. In general, collocation formulations cannot be directly applied to the problem of solution singularities, although some special regularization treatments may be used for isolated geometric singularities [46,81]. In contrast, Galerkin formulations are more suitable to deal with solution singularities. Recently, Hou et al. have constructed a 2D immersed FEM method which is of first order convergence in the solution and 0.7th order convergence in the gradient of the solution when the solution of the Poisson equation is C^1 continuous and the interface is Lipschitz continuous [30]. More recently, the Wang–Ye Galerkin method was found to be particularly suitable for this problem in 2D settings. It is able to deliver at least 1.75th order convergence in the solution and 1st order convergence in the gradient [23]. The weak Galerkin nature of the Wang–Ye Galerkin method [22] may have played an essential role in making such an important progress.

Motivated by the success of Galerkin methods for resolving low regularity solution problems, a 2D MIB Galerkin method has been proposed in a recent work [83]. The main idea is to combine the flexibility of the MIB method for geometric singularities with the power of the Galerkin formulation for solution singularities. To by-pass mesh generation and refinement, Cartesian grid based triangular elements are employed. To simplify the formulation, classical continuous finite element basis functions are utilized throughout the whole computational domain. Since complex interfaces cut through elements, two sets of overlapping elements, called MIB elements, are defined in the vicinity of the interface to ensure the continuity of the basis functions across the interface. Consequently, the differential operators near the interface are evaluated as if there is no discontinuous coefficients, a typical MIB strategy [32–36]. Since the overlapping domains lead to extra nodes and elements

which are not admissible in the classical sense of FEM partitions, additional treatment is required to determine extra number of degrees of freedom. Interface jump conditions are utilized to uniquely determine function values on overlapping MIB elements. Similar to the original MIB method, the new MIB Galerkin method utilizes a set of lowest order interface jump conditions to enhance the stability of the interface scheme. The MIB Galerkin performed well for problems with geometric singularities induced solution singularities [83].

The main objective of the present work is to develop second order 3D MIB Galerkin schemes for practical elliptic interface problems. Another goal of the present work is to construct the first known second order 3D method for elliptic interface problems with arbitrarily complex interface geometries and geometric singularities induced solution singularities. The extension from our 2D MIB Galerkin schemes [83] to 3D ones is nontrivial. First, it is not obvious to design efficient 3D finite elements and associated basis functions which satisfy the Cartesian mesh constraint. Additionally, it is perhaps quite simple to construct a new second order elliptic interface method for simple 3D geometric shapes. However, is very difficult to develop a 3D elliptic interface method that delivers second order or near second order accuracy for arbitrarily complex interfaces, such as the geometric shapes of biomolecules. We construct 3D MIB Galerkin schemes with three different Cartesian based elements, including a rectangular prism element and two tetrahedral elements obtained respectively by triangulating each cube into five and six tetrahedra. The performance of these MIB Galerkin FEMs is examined for many realistic applications, including protein interfaces.

The rest of this paper is organized as follows. Section 2 is devoted to the general theory of the MIB Galerkin formulation. We first describe essential ideas of the MIB Galerkin method. The finite element partition, trial and test spaces are defined before constructing the Galerkin approximation. A unique feature of the present method is the use of two sets of overlapping MIB elements to describe the approximation of the true solution and enforcement of interface jump conditions. The solution algorithms for fictitious values on the irregular domains in vicinity of the interface are given in Section 3. Continuous basis functions are used for three different elements, i.e., a rectangular prism element and two tetrahedra elements. Coefficients of the solution on the normal nodes are determined in the classical manner. However, coefficients for the interpolation function defined on the irregular domains are determined by using interface jump conditions, which in turn solves fictitious values. In Section 4, we carry out extensive numerical experiments to validate the order of accuracy and demonstrate the performance of the proposed MIB Galerkin formulation. We first consider a few relatively simple interface geometries defined by level set functions, such as a sphere, four contacting spheres and four contacting cylinders. After establishing the second order accuracy for these simple cases, we consider interface geometries from realistic biomolecular systems, such as proteins and multiprotein complexes. Our MIB Galerkin method achieves the second order convergence for these truly arbitrarily complex interface geometries. To our knowledge, it is the first time that a near second order Galerkin method has been confirmed with realistic protein interfaces, although many FEM based elliptic interface algorithms have been applied to biomolecules in the literature. Finally, we examine the performance of the present MIB Galerkin method for 3D elliptic interface problems with low regularity solutions induced by geometric singularities. We report near second order convergence solution for this class of problems. This paper ends with a conclusion.

2. Three-dimensional MIB Galerkin method

Let us consider an open bounded domain $\Omega \subset \mathbb{R}^3$ with a given interface Γ, which divides the domain into two subdomains, Ω^+ and Ω^-. The boundary $\partial\Omega$ and interfaces Γ may be nonsmooth, such as Lipschitz continuous. The interface can be characterized by a piecewise smooth level-set function $\varphi \in \Omega$, such that $\Gamma = \{\mathbf{x}|\varphi(\mathbf{x}) = 0,\ \forall \mathbf{x} \in \Omega\}$. As such, two subdomains can be given by $\Omega^+ = \{\mathbf{x}|\varphi(\mathbf{x}) \geq 0,\ \forall \mathbf{x} \in \Omega\}$ and $\Omega^- = \{\mathbf{x}|\varphi(\mathbf{x}) < 0,\ \forall \mathbf{x} \in \Omega\}$. In some practical applications, the interface may be given as an isovalue of a scalar field. The 3D elliptic interface problem can be described as

$$-\nabla \cdot \beta(\mathbf{x})\nabla u(\mathbf{x}) = g(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega \tag{1}$$

$$u = g_b, \quad \forall \mathbf{x} \text{ on } \partial\Omega \tag{2}$$

where $g(\mathbf{x})$ is a piecewise continuous function, g_b is the boundary value, and $\beta(\mathbf{x})$ is a variable coefficient that is discontinuous on the interface Γ. As a result, two jump conditions are required to make the problem well posed

$$[u] = u^+ - u^- = \Phi, \quad \forall \mathbf{x} \text{ on } \Gamma \tag{3}$$

$$[\beta u_n] = \beta^+ u_n^+ - \beta^- u_n^- = \Psi, \quad \forall \mathbf{x} \text{ on } \Gamma \tag{4}$$

where u^+, u_n^+ and β^+ denote their limiting value from the Ω^+ side of the interface Γ, and u^-, u_n^- and β^- denote their limiting value from the Ω^- side of the interface Γ. The derivatives u_n^+ and u_n^- are evaluated along the normal direction on the interface. Here $\Phi(\mathbf{x})$ and $\Psi(\mathbf{x})$ is at least C^1 continuous. Eqs. (1)–(4) define the elliptic interface problem to be solved in the present work.

2.1. Essential ideas of the MIB Galerkin method

One of important issues of the present MIB Galerkin method is its mesh. Like the original MIB method, the MIB Galerkin formulation admits the Cartesian type of meshes to avoid the mesh generation procedure. In the earlier 2D MIB Galerkin

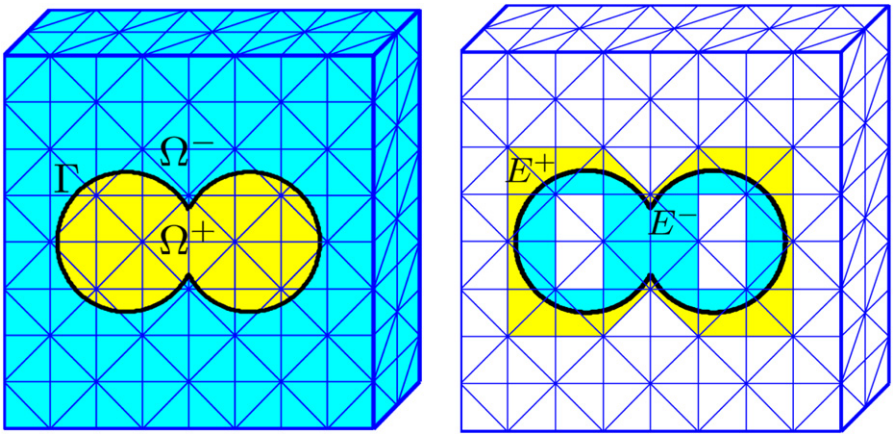

Fig. 1. A cross section showing the domain decomposition via interface Γ (the double circle) in a tetrahedral Cartesian mesh. Left chart: partition of the domain Ω into Ω^+ and Ω^- by interface Γ; Right chart: illustration of irregular domains E^+ and E^-.

method, the triangular Cartesian meshes are automatically generated by introducing diagonal lines to the rectangular Cartesian meshes. In the present 3D formulation, we consider Cartesian meshes encompassing either rectangular prism elements or tetrahedral elements generated from further triangulating the rectangular prism element. Note that the element shape is very important in the present Galerkin method as numerically investigated in Section 4. Since our partition is based on the Cartesian mesh, we can also directly employ the Cartesian mesh in the present MIB Galerkin formulation.

Another important issue of the MIB method is the systematical classification of grid nodes or elements into regular type and irregular type. In the original MIB method, we classify mesh nodes into regular and irregular ones. In the MIB Galerkin method, we classify elements into regular and irregular ones. Fig. 1 illustrates a cross section of a 3D interface problem. The interface Γ and the relation among domains Ω, Ω^+ and Ω^- are depicted. Because of the use of the Cartesian type of meshes, a complex interface inevitably cuts through certain elements, as shown in Fig. 1. For simplicity, we define an element as irregular element if its vertices locate on more than one side of the interface. Let us denote domain E as an irregular domain that constitutes all the irregular elements. Obviously, irregular elements contain regions from both Ω^+ and Ω^-, as shown in the right chart of Fig. 1. The interface divides the irregular domain E into two subdomains, labeled as E^+ and E^-, where $E^+ = \Omega^- \cap E$ and $E^- = \Omega^+ \cap E$, as shown in Fig. 1. With irregular elements, we can define irregular nodes as the vertices of irregular elements. Note that unlike in the finite difference based MIB schemes, the definition of regular and irregular elements is the same for both second or higher order MIB Galerkin schemes.

The next issue in the MIB method is about domain extension. In the present 3D MIB Galerkin method, we extend domains Ω^+ and Ω^- across the interface Γ as shown in Fig. 2. The extended domain Ω_e^+ constitutes the original domain Ω^+ and the irregular domain E^+, i.e., $\Omega_e^+ = \Omega^+ \cup E^+$. Similarly, extended domain Ω_e^- constitutes the original domain (Ω^-) and the irregular domain (E^-), i.e., $\Omega_e^- = \Omega^- \cup E^-$. As a result, we have two sets of overlapping elements, i.e., MIB elements, on the locations of irregular elements near the interface.

To properly define the elliptic interface problem on extended domains, we need to extend discontinuous coefficient $\beta_e(\mathbf{x})$ over the extended domains

$$\beta_e^+(\mathbf{x}) = \begin{cases} \beta^+(\mathbf{x}), & \forall \mathbf{x} \in \Omega^+ \\ \beta_{E^+}^+(\mathbf{x}), & \forall \mathbf{x} \in E^+, \end{cases} \tag{5}$$

and

$$\beta_e^-(\mathbf{x}) = \begin{cases} \beta^-(\mathbf{x}), & \forall \mathbf{x} \in \Omega^- \\ \beta_{E^-}^-(\mathbf{x}), & \forall \mathbf{x} \in E^-, \end{cases} \tag{6}$$

where $\beta_{E^+}^+(\mathbf{x})$ and $\beta_{E^-}^-(\mathbf{x})$ are the smooth extensions of the coefficient $\beta^+(\mathbf{x})$ and $\beta^-(\mathbf{x})$ over domains E^+ and E^-, respectively. Similarly, we also define extended singular source function $g_e(\mathbf{x})$ over extended domains

$$g_e^+(\mathbf{x}) = \begin{cases} g^+(\mathbf{x}), & \forall \mathbf{x} \in \Omega^+ \\ g_{E^+}^+(\mathbf{x}), & \forall \mathbf{x} \in E^+, \end{cases} \tag{7}$$

and

$$g_e^-(\mathbf{x}) = \begin{cases} g^-(\mathbf{x}), & \forall \mathbf{x} \in \Omega^- \\ g_{E^-}^-(\mathbf{x}), & \forall \mathbf{x} \in E^-, \end{cases} \tag{8}$$

where $g_{E^+}^+(\mathbf{x})$ and $g_{E^-}^-(\mathbf{x})$ are the smooth extensions of the functions $g^+(\mathbf{x})$ and $g^-(\mathbf{x})$ over domains $E^+(\mathbf{x})$ and $E^-(\mathbf{x})$, respectively. With these extensions, we can define the original interface problems on MIB elements.

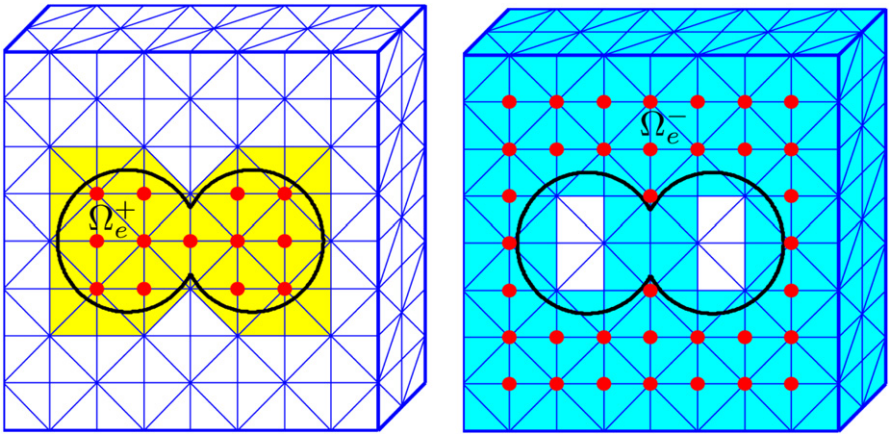

Fig. 2. A cross section illustration of extended domain Ω_e^+ (left chart) and extended domain Ω_e^- (right chart) across the interface Γ. Solid dots are vertices.

A common practice in the MIB method [34,45–48] and the earlier DSC algorithm [37,38] is the use of fictitious values defined on irregular nodes in the irregular domains. As the solution and the gradient of the solution may be discontinuous across the interface, the MIB method smoothly extends the solution across the interface to the nodes on extended domains. These extended values are called fictitious values, which are to be used in the discretization of the original PDE on the MIB elements near the interface. By means of fictitious values, derivatives across the interface are computed as if the solution was smooth.

The selection of basis functions is an important issue for FEMs in general. The quality and performance of an FEM solution to an elliptic interface problem often depend on its basis function. There are many different basis functions, such as those associated with continuous elements, non-conforming elements, mixed finite elements and discontinuous Galerkin FEM. Typically, by using more complicated FEM techniques, one gains capability and flexibility in dealing with complex boundary, interface, and/or ill-posed governing equations. In the present MIB Galerkin method, the use of MIB elements and the existence of fictitious values allow us to employ the continuous FEM basis to solve discontinuous problems. Therefore, the classical continuous FEM basis functions are employed in the present work, as described in detail in Section 3.

To solve an interface problem, one needs to rigorously enforce interface jump conditions. In the original MIB method, these conditions, or set of equations, are utilized to numerically determine fictitious values on extended domains. The same idea is utilized in the present MIB Galerkin method. The detailed implementation of this idea is described in Section 3. Note that in high order MIB schemes, the set of interface jump conditions is iteratively used so as to determine as many fictitious values as needed. This approach can also be used to create high order MIB Galerkin schemes.

Finally, it is noted that the situation illustrated in Fig. 1 is a special situation. In general, the interface Γ may pass through the domain boundary $\partial\Omega$. Therefore the boundaries of both extended domains may also be part of the boundary of the computational domain, i.e., $\partial\Omega_e^- \cap \partial\Omega \neq \emptyset$ and $\partial\Omega_e^+ \cap \partial\Omega \neq \emptyset$. This general setting does not affect our method. Additionally, the irregular elements shown in Fig. 1 are simple cases. Fig. 3 demonstrates a special situation where element edges cut through the interface while the element is still a regular element, because all of its vertices locate in the same subdomain.

2.2. MIB Galerkin formulation

Denote $\mathcal{T}_h$ as a partition of the domain Ω with mesh size h. The partition is regrouped into two sets denoted by $\mathcal{T}_h^+ = \mathcal{T}_h \cap \Omega_e^+$ and $\mathcal{T}_h^- = \mathcal{T}_h \cap \Omega_e^-$, respectively. Obviously, $\mathcal{T}_h^\pm$ is finite element partitions for subdomains $\Omega_e^\pm$. Denote $K^\pm \in \mathcal{T}_h^\pm$ the MIB elements resulting from $\mathcal{T}_h^\pm$. On these elements, $P_j(K^+)$ and $P_k(K^-)$ are two sets of polynomials with degrees no more than j and k, respectively. The choice of j and k determines the order of the 3D MIB Galerkin FEMs. Let us define two trial MIB element spaces

$$U_h^+ := \left\{ u_h^+ \in H^1(\Omega_e^+) : \ u_h^+|_{K^+} \in P_j(K^+), \forall K^+ \in \mathcal{T}_h^+ \right\} \tag{9}$$

$$U_h^- := \left\{ u_h^- \in H^1(\Omega_e^-) : \ u_h^-|_{K^-} \in P_k(K^-), \forall K^- \in \mathcal{T}_h^- \right\}. \tag{10}$$

For a given MIB element $K^\pm \in \mathcal{T}_h^\pm$, let $\phi_i^\pm, \ i = 1, \ldots, N^\pm$ be a set of basis functions for $P_j(K^+)$ and $P_k(K^-)$, respectively. A trial function $u_h^\pm$ can be constructed by basis functions

$$u_h^\pm(\mathbf{x}) = \sum_{i=1}^{N^\pm} u_{h,i}^\pm \phi_i^\pm(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega_e^\pm, \tag{11}$$

where $\{u_{h,i}^\pm\}$ are expansion coefficients.

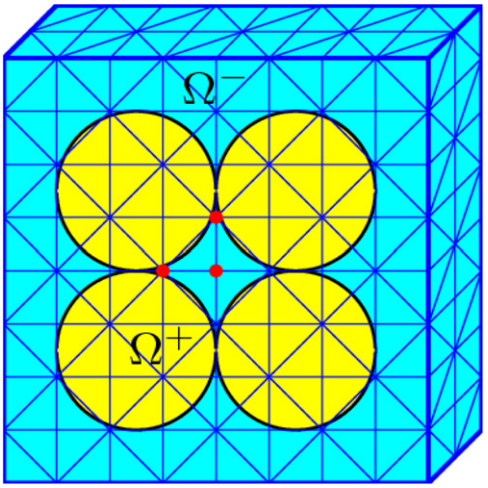

Fig. 3. Illustration of special regular elements. All the three vertices denoted by red dots locate in the Ω^- domain, although one of their edges intersects the interface. Therefore, this element is still a regular element, as far as it is judged from its three vertices in the present cross section.

In general, one test space is required in a Galerkin formulation. In the present method, we need two test spaces. A simple choice is to construct these spaces via U_h^+ and U_h^-

$$U_h^{0,\pm} := \left\{ v^\pm \in U_h^\pm : \ v^\pm = 0 \text{ on } \partial\Omega_e^\pm \right\}. \tag{12}$$

To solve Eqs. (1)–(4) with a Galerkin procedure, we multiply the original equation with a test function $v^\pm \in U_h^{0,\pm}$ and then carry out the integration over domain $\Omega_e^\pm$

$$-\int_{\Omega_e^\pm} \nabla \cdot (\beta_e^\pm(\mathbf{x}) \nabla u_h^\pm(\mathbf{x})) v^\pm d\mathbf{x} = \int_{\Omega_e^\pm} g_e^\pm(\mathbf{x}) v^\pm d\mathbf{x}, \quad \forall \mathbf{x} \in \Omega_e^\pm. \tag{13}$$

After integrating by parts, we have

$$-\int_{\partial\Omega_e^\pm} \beta_e^\pm(\mathbf{x}) \frac{\partial u_h^\pm(\mathbf{x})}{\partial n^\pm} v^\pm ds + \int_{\Omega_e^\pm} \beta_e^\pm(\mathbf{x}) \nabla u_h^\pm(\mathbf{x}) \cdot \nabla v^\pm d\mathbf{x} = \int_{\Omega_e^\pm} g_e^\pm v^\pm d\mathbf{x}, \quad \forall \mathbf{x} \in \Omega_e^\pm. \tag{14}$$

The above equation can be simplified according to the property of the test space (12),

$$\int_{\Omega_e^\pm} \beta_e^\pm(\mathbf{x}) \nabla u_h^\pm(\mathbf{x}) \cdot \nabla v^\pm d\mathbf{x} = \int_{\Omega_e^\pm} g_e^\pm(\mathbf{x}) v^\pm d\mathbf{x}, \quad \forall \mathbf{x} \in \Omega_e^\pm, \ \forall v^\pm \in U_h^{0,\pm}. \tag{15}$$

Due to the overlapping feature of the MIB elements, partitions $\mathcal{T}_h^+$ and $\mathcal{T}_h^-$ are not admissible, and Eqs. (15) is not well posed in the usual sense. Specifically, the number of unknowns is greater than the number of equations. To resolve this problem, we define a set of normal solutions U_h as a restriction of $\{u_h^+\}$ and $\{u_h^-\}$ to the domain Ω^+ and Ω^-, respectively

$$U_h := \left\{ u_h = u_h^\pm(\mathbf{x}) : \ \forall \mathbf{x} \in \Omega^\pm \right\} \tag{16}$$

where $\{u_h\}$ are to be determined by Eqs. (15). It is clear that the total number of unknowns in the set $\{u_h\}$ equals the total number of node points in the inner domain $(\Omega \setminus \partial\Omega)$. However, the number of unknowns in Eqs. (15), which are defined on extended domains, is larger than the total number of nodes in the inner domain $(\Omega \setminus \partial\Omega)$. To resolve this problem, we define two sets of fictitious values

$$F_h^\pm := \left\{ f_h^\pm(\mathbf{x}) = u_h^\pm(\mathbf{x}) : \ \forall \mathbf{x} \in E^\pm \right\}. \tag{17}$$

If these fictitious values, denoted as $F_h = F_h^+ \cup F_h^-$, are determined by other means, then the solution to Eqs. (15) becomes unique. In the MIB Galerkin method, we use interface jump conditions to determine all fictitious values. In the original MIB method, the set of low order interface jump conditions is repeatedly used to construct high order schemes. The same approach is used for the MIB Galerkin method. The specific detail of determining F_h is described in Section 3.

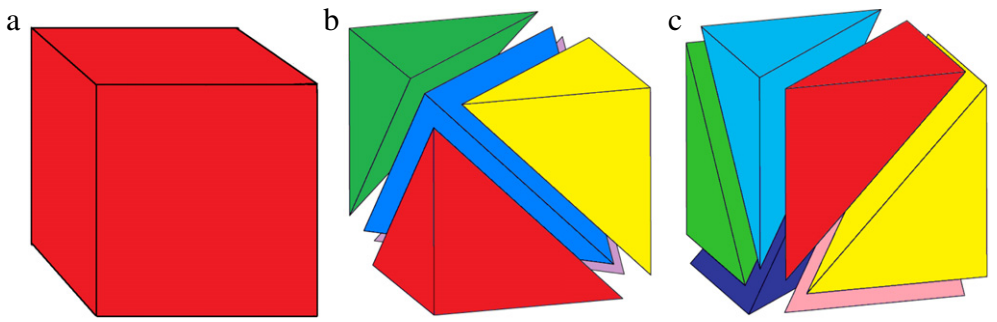

Fig. 4. Illustration of three types of finite elements. (a) Rectangular prism element; (b) Five-tetrahedron element; (c) Six-tetrahedron element.

3. Solution algorithms

3.1. Three types of elements

The above MIB Galerkin formulation is independent of elements. In this work, we employ a regular Cartesian mesh. Different elements can be chosen to furnish the Cartesian mesh on condition that all of their vertices coincide with the original Cartesian nodes. Consequently, no new node is introduced by elements. We discuss three simple choices of elements. First, each Cartesian grid naturally gives rise to a rectangular prism element, which is employed in the present work. Additionally, we consider tetrahedral elements generated from the Cartesian mesh. Essentially, there are two ways to subdivide a rectangular prism into simple tetrahedral elements without introducing new node as depicted in Fig. 4. One subdivision leads to a five-tetrahedron element, and the other produces a six-tetrahedron element. For the six-tetrahedron element, the subdivision method should be exactly the same for each Cartesian grid in the whole mesh, so that the edges of the tetrahedral elements in adjacent grids do not intersect with each other. However, for five-tetrahedron element, adjacent two Cartesian grids are discretized differently as demonstrated in Fig. 5.

For both five-tetrahedron elements and six-tetrahedron elements, the basis function $\phi_j(\mathbf{x})$ can be constructed from the traditional tetrahedral volume coordinates [86]. For a prism element, we first map its eight nodes coordinates (x_j, y_j, z_j) to a reference coordinate (ξ_j, η_j, ζ_j). In such a coordinate, the basis function for a reference element can be expressed as,

$$\phi_j(\xi, \eta, \zeta) = \frac{1}{8}(1 + \xi\xi_j)(1 + \eta\eta_j)(1 + \zeta\zeta_j), \tag{18}$$

where ξ_j, η_j and ζ_j are integer numbers with value either -1 or 1.

3.2. Basic solution algorithm

This section concerns the solution of discretization equations (15) using basis functions described above. To simplify the notation, we introduce the following bilinear form

$$a(u_h, v_h) = (\beta\nabla u_h, \nabla v_h), \tag{19}$$

where the inner product (a, b) represents the integration of the product of ab over the region where the functions are defined. Therefore, we can recast Eqs. (15) into simplified representations

$$a(u_h^+, v_h^+) = (g_e^+, v_h^+), \tag{20}$$

$$a(u_h^-, v_h^-) = (g_e^-, v_h^-). \tag{21}$$

3.2.1. Matrix equations for rectangular prism elements

In the mesh of rectangular prism elements, for a regular node (or vertex) $\mathbf{x}_i \in \Omega_e^+ \setminus E$, we also need to consider all 26 adjacent nodes (or vertices). A total of 27 nodes is denoted as $i_1, i_2, \ldots, i_{27}$. We use the approximation in Eq. (11) to discretize Eq. (20)

$$(a(\phi_{i_1}^+, \phi_i^+), a(\phi_{i_2}^+, \phi_i^+), \ldots, a(\phi_{i_{27}}^+, \phi_i^+)) \cdot \begin{pmatrix} u_{h,i_1}^+ \\ u_{h,i_2}^+ \\ \cdots \\ u_{h,i_{27}}^+ \end{pmatrix} = (g_e^+, \phi_i^+), \tag{22}$$

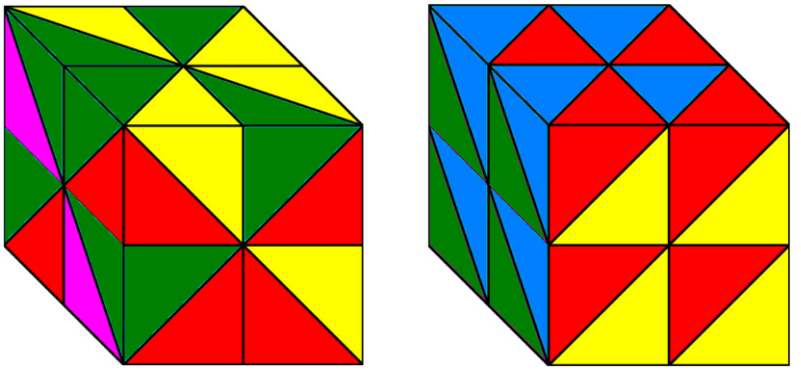

Fig. 5. Illustration of Cartesian meshes associated with five-tetrahedron element (left) and six-tetrahedron element (right).

where $\{u^+_{h,i},\ i = i_1, i_2, \ldots, i_{27}\}$ are coefficients on 27 related nodes. Similarly, for a regular node $\mathbf{x}_j \in \Omega_e^- \setminus E$,

$$(a(\phi^-_{j_1}, \phi^-_j), a(\phi^-_{j_2}, \phi^-_j), \ldots, a(\phi^-_{j_{27}}, \phi^-_j)) \cdot \begin{pmatrix} u^-_{h,j_1} \\ u^-_{h,j_2} \\ \cdots \\ u^-_{h,j_{27}} \end{pmatrix} = (g^-_e, \phi^-_j), \tag{23}$$

where $\{u^-_{h,j},\ j = j_1, j_2, \ldots, j_{27}\}$ are coefficients on 27 related nodes. However, Eqs. (22) and (23) cannot be solved directly because of the lack of boundary condition on ∂E.

For an irregular node $\mathbf{x}_i \in E^-$ near the interface, a similar Galerkin approximation can be obtained. However, fictitious values are now involved in the discretization scheme

$$(a(\phi^+_{i_1}, \phi^+_i), a(\phi^+_{i_2}, \phi^+_i), \ldots, a(\phi^+_{i_{27}}, \phi^+_i)) \cdot \begin{pmatrix} u^+_{h,i_1} \\ u^+_{h,i_2} \\ \cdots \\ f^+_{h,i_{27}} \end{pmatrix} = (g^+_e, \phi^+_i), \tag{24}$$

where $f^+_{h,i_{27}}$ is a fictitious value. Here we give a general form of the function values and fictitious values. The number of fictitious values in each discretization scheme of the irregular node depends on the local geometry, i.e., the relation between the interface and the mesh. So does the number of function values in the scheme. However, the sum of these two numbers equals 27. Similarly, for an irregular node $\mathbf{x}_j \in E^+$, we have matrix elements

$$(a(\phi^-_{j_1}, \phi^-_j), a(\phi^-_{j_2}, \phi^-_j), \ldots, a(\phi^-_{j_{27}}, \phi^-_j)) \cdot \begin{pmatrix} u^-_{h,j_1} \\ u^-_{h,j_2} \\ \cdots \\ f^-_{h,j_{27}} \end{pmatrix} = (g^-_e, \phi^-_j). \tag{25}$$

As the same reason we have described above, here the number of fictitious values also varies according to the local geometry. Eqs. (22)–(25) describe all of the matrix elements in the present MIB Galerkin method with rectangular prism elements. The treatment of fictitious values is discussed in Section 3.3.

3.2.2. Matrix equations for five-tetrahedron elements

For a five-tetrahedron element mesh, two types of vertices have different numbers of related vertices. A Type1 vertex is related to 6 nearest adjacent vertices. Therefore the discretization form of Eq. (20) for a Type1 vertex can be expressed as

$$(a(\phi^+_{i_1}, \phi^+_i), a(\phi^+_{i_2}, \phi^+_i), \ldots, a(\phi^+_{i_7}, \phi^+_i)) \cdot \begin{pmatrix} u^+_{h,i_1} \\ u^+_{h,i_2} \\ \cdots \\ u^+_{h,i_7} \end{pmatrix} = (g^+_e, \phi^+_i). \tag{26}$$

A Type2 vertex shares its edges with 18 nearest adjacent vertices. The detailed geometry of these 18 nearest adjacent vertices is demonstrated in Fig. 6. Therefore the discretization form of Eq. (20) for a Type2 vertex is

$$(a(\phi^+_{i_1}, \phi^+_i), a(\phi^+_{i_2}, \phi^+_i), \ldots, a(\phi^+_{i_{19}}, \phi^+_i)) \cdot \begin{pmatrix} u^+_{h,i_1} \\ u^+_{h,i_2} \\ \cdots \\ u^+_{h,i_{19}} \end{pmatrix} = (g^+_e, \phi^+_i). \tag{27}$$

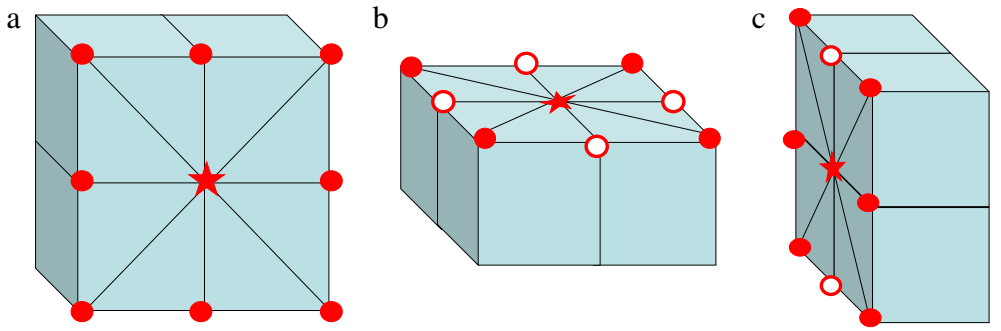

Fig. 6. An illustration of 18 nearest adjacent vertices of a Type2 vertex in a five-tetrahedron element. The vertex of interest is labeled with a star. Its nearest adjacent vertices in three cross sections are denoted by either solid dots or circles. Here, solid dots are to be counted while circles have already been counted in other cross sections. (a) Cross section at $y = y_0$; (b) Cross section at $z = z_0$; (c) Cross section at $x = x_0$.

Similarly, for a regular vertex $\mathbf{x}_j \in \Omega_e^- \setminus E$, two discretization forms can be attained for two types of vertices

$$(a(\phi_{j_1}^-, \phi_j^-), a(\phi_{j_2}^-, \phi_j^-), \ldots, a(\phi_{j_7}^-, \phi_j^-)) \cdot \begin{pmatrix} u_{h,j_1}^- \\ u_{h,j_2}^- \\ \cdots \\ u_{h,j_7}^- \end{pmatrix} = (g_e^-, \phi_j^-), \tag{28}$$

and

$$(a(\phi_{j_1}^-, \phi_j^-), a(\phi_{j_2}^-, \phi_j^-), \ldots, a(\phi_{j_{19}}^-, \phi_j^-)) \cdot \begin{pmatrix} u_{h,j_1}^- \\ u_{h,j_2}^- \\ \cdots \\ u_{h,j_{19}}^- \end{pmatrix} = (g_e^-, \phi_j^-). \tag{29}$$

3.2.3. Matrix equations for six-tetrahedron elements

Matrix equations for the six-tetrahedron element are constructed as follows. For a regular vertex $\mathbf{x}_i \in \Omega_e^+ \setminus E$, Eq. (11) is employed to discretize Eq. (20). For a six-tetrahedron element mesh, a regular vertex is related to 14 adjacent vertices and a total of 15 vertices is denoted as $i_1, i_2, \ldots, i_{15}$. The discretization formulation is given by

$$(a(\phi_{i_1}^+, \phi_i^+), a(\phi_{i_2}^+, \phi_i^+), \ldots, a(\phi_{i_{15}}^+, \phi_i^+)) \cdot \begin{pmatrix} u_{h,i_1}^+ \\ u_{h,i_2}^+ \\ \cdots \\ u_{h,i_{15}}^+ \end{pmatrix} = (g_e^+, \phi_i^+), \tag{30}$$

where $\{u_{h,i}^+,\ i = i_1, i_2, \ldots, i_{15}\}$ are coefficients of 15 related vertices. Similarly, for a regular vertex $\mathbf{x}_j \in \Omega_e^- \setminus E$, one has

$$(a(\phi_{j_1}^-, \phi_j^-), a(\phi_{j_2}^-, \phi_j^-), \ldots, a(\phi_{j_{15}}^-, \phi_j^-)) \cdot \begin{pmatrix} u_{h,j_1}^- \\ u_{h,j_2}^- \\ \cdots \\ u_{h,j_{15}}^- \end{pmatrix} = (g_e^-, \phi_j^-), \tag{31}$$

where $\{u_{h,j}^-,\ j = j_1, j_2, \ldots, j_{15}\}$ are coefficients of 15 related vertices. However, Eqs. (30) and (31) cannot be solved directly because of the lack of boundary conditions on ∂E.

3.2.4. Matrix equations for irregular tetrahedral vertices

From above discussions, it is easy to find out that the discretization form for a given vertex is determined by adjacent vertices which all share edges with the particular vertex. There are three types of tetrahedral vertices, namely, Type1 vertices of five-tetrahedron elements, Type2 vertices of five-tetrahedron elements, and six-tetrahedron element vertices. Let us denote the number of adjacent vertices as $n - 1$ for these three cases. Then a uniform representation for Eq. (20) can be expressed as

$$(a(\phi_{i_1}^+, \phi_i^+), a(\phi_{i_2}^+, \phi_i^+), \ldots, a(\phi_{i_n}^+, \phi_i^+)) \cdot \begin{pmatrix} u_{h,i_1}^+ \\ u_{h,i_2}^+ \\ \cdots \\ u_{h,i_n}^+ \end{pmatrix} = (g_e^+, \phi_i^+), \tag{32}$$

where $n = 16$, 7 and 19 are respectively for Type1 vertices of five-tetrahedron elements, Type2 vertices of five-tetrahedron elements, and six-tetrahedron element vertices.

For an irregular vertex $\mathbf{x}_i \in E^-$, a similar discretization can be obtained. However, fictitious values are now involved in the discretization scheme

$$(a(\phi_{i_1}^+, \phi_i^+), a(\phi_{i_2}^+, \phi_i^+), \ldots, a(\phi_{i_n}^+, \phi_i^+)) \cdot \begin{pmatrix} u_{h,i_1}^+ \\ u_{h,i_2}^+ \\ \cdots \\ f_{h,i_n}^+ \end{pmatrix} = (g_e^+, \phi_i^+), \tag{33}$$

where f_{h,i_n}^+ is a fictitious value. The number of fictitious values depends on the local geometry, i.e., the relation between the interface and the mesh. Similarly, for an irregular vertex $\mathbf{x}_j \in E^+$, we have matrix elements

$$(a(\phi_{j_1}^-, \phi_j^-), a(\phi_{j_2}^-, \phi_j^-), \ldots, a(\phi_{j_n}^-, \phi_j^-)) \cdot \begin{pmatrix} u_{h,j_1}^- \\ u_{h,j_2}^- \\ \cdots \\ f_{h,j_n}^- \end{pmatrix} = (g_e^-, \phi_j^-). \tag{34}$$

Combined with the regular vertex discretization formulation, all of the matrix elements in the present MIB Galerkin are attained for tetrahedral elements.

3.3. Algorithms for determining fictitious values

Fictitious values in the above matrix equations are to be resolved before the FEM solution can be obtained. This issue is discussed in the present section. We first describe the interface jump conditions, which is needed at each intersecting point between element edges and the interface.

3.3.1. 3D interface jump conditions

In the MIB method, a local coordinate system is needed on an interface point where interface jump conditions are enforced. The relation between the local coordinate system and the Cartesian mesh is described below. Denote (ξ, η, ζ) a local coordinate system on an interface point. Here ξ is along the normal direction and η is in the x–y plane. The transformation from coordinate (x, y, z) to (ξ, η, ζ) can be given as

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = \mathbf{P} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{35}$$

where the transformation matrix $\mathbf{P}$ has the form

$$\mathbf{P} = \begin{pmatrix} \sin\phi\cos\theta & \sin\phi\sin\theta & \cos\phi \\ -\sin\theta & \cos\theta & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi \end{pmatrix} \tag{36}$$

where θ and ϕ are the azimuth and zenith angles with respect to the normal direction n, respectively. To increase the number of interface conditions, we usually differentiate Eq. (3) along the tangential directions of the interface to obtain two additional interface conditions, $[u_\eta] = u_\eta^+ - u_\eta^-$, and $[u_\zeta] = u_\zeta^+ - u_\zeta^-$. In the local coordinate system, these two jump conditions can be expressed as

$$[u_\eta] = (-\sin\theta u_x^+ + \cos\theta u_y^+) - (-\sin\theta u_x^- + \cos\theta u_y^-) \tag{37}$$

$$[u_\zeta] = (-\cos\phi\cos\theta u_x^+ - \cos\phi\sin\theta u_y^+ + \sin\phi u_z^+) - (-\cos\phi\cos\theta u_x^- - \cos\phi\sin\theta u_y^- + \sin\phi u_z^-) \tag{38}$$

and the jump condition with respect to the normal direction can be given as

$$[\beta u_\xi] = \beta^+(\sin\phi\cos\theta u_x^+ + \sin\phi\sin\theta u_y^+ + \cos\phi u_z^+) - \beta^-(\sin\phi\cos\theta u_x^- + \sin\phi\sin\theta u_y^- + \cos\phi u_z^-). \tag{39}$$

These three jump conditions, together with the original jump condition (3), constitute a set of lowest order jump conditions that is usually enforced in the MIB method. The enforcement of jump conditions determines fictitious values to be used in the discretization of differentiation operators in a given PDE.

3.3.2. Basic algorithms

In this section we utilize a smooth interface as shown in Fig. 7 to demonstrate the MIB Galerkin treatment of fictitious values. A rectangular prism element mesh and a five-tetrahedron element mesh are discussed. For the rectangular prism

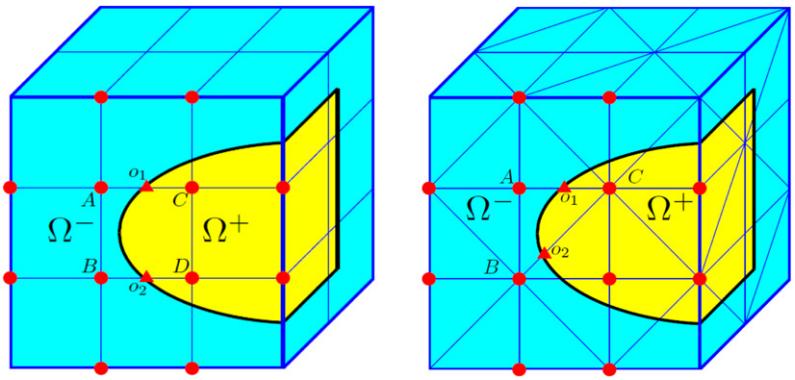

Fig. 7. A cross section illustrating MIB Galerkin treatments of a smooth interface. A rectangular prism element mesh (left) and a five-tetrahedron element mesh (right) are depicted. Solid dots are element vertices. In the left situation, A, B, C and D are irregular vertices in a rectangular prism element mesh, while in the right situation, A, B and C are irregular vertices in a five-tetrahedron element mesh. Intersecting points (i.e., o_1 and o_2) between element edges and the interface are labeled with triangles, where interface jump conditions are enforced to determine fictitious values on irregular nodes. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

element mesh, the basis function at each vertex involves seven other vertices. Therefore, if one or more of the rest seven vertices locate in the other subdomain, the discretization of matrix equation (21) at this vertex involves fictitious values. For example, in the left chart of Fig. 7, the yellow region and the green region represent subdomain Ω^+ and subdomain Ω^- respectively. The basis function for vertex A involves nodes C and D, which are in a different subdomain. Therefore, fictitious values are used for vertices C and D when we construct discretization matrix equation (21) for vertex A. For the same reason, the discretization matrix at vertex C include fictitious values for vertices A and B. Therefore, for this special rectangular prism element, we need to determine the fictitious value for all four vertices in this cross section.

For the tetrahedral mesh shown in the right chart of Fig. 7, the situation is quite similar to that in the right chart. Vertices A, B and C are all irregular vertices and corresponding fictitious values are needed in constructing the matrix equation. To determine these fictitious values, we construct two 3D second order polynomial functions, one in domain Ω_e^+ and the other one in domain Ω_e^-

$$u_h^{e+}(x,y,z) = a_0 + a_1x + a_2y + a_3z + a_4x^2 + a_5y^2 + a_6z^2 + a_7xy + a_8xz + a_9yz, \quad (x,y,z) \in \Omega_e^+ \tag{40}$$

$$u_h^{e-}(x,y,z) = b_0 + b_1x + b_2y + b_3z + b_4x^2 + b_5y^2 + b_6z^2 + b_7xy + b_8xz + b_9yz, \quad (x,y,z) \in \Omega_e^- \tag{41}$$

where $\{a_l\}_{l=0,1,\ldots,9}$ and $\{b_n\}_{n=0,1,\ldots,9}$ are a set of 20 coefficients. It is noted that both $u_h^{e+}(x,y,z)$ and $u_h^{e-}(x,y,z)$ are defined near the interface. In fact, $u_h^{e+}(x,y,z) = u_h^+(x,y,z)$ is a normal solution (i.e., $u_h^{e+}(x,y,z) \in U_h$) when $(x,y,z) \in \Omega^+$. However, $u_h^{e+}(x,y,z) = f_h^+(x,y,z)$ is a fictitious value when $(x,y,z) \in E^+$. Similarly, we have $u_h^{e-}(x,y,z) = u_h^-(x,y,z)$ when $(x,y,z) \in \Omega^-$, and $u_h^{e-}(x,y,z) = f_h^-(x,y,z)$ when $(x,y,z) \in E^-$.

Once the coefficients are determined in Eqs. (40) and (41), six derivatives, i.e., u_x^+, u_y^+, u_z^+, u_x^-, u_y^-, and u_z^-, in the interface jump conditions can be easily calculated. Additionally, all fictitious values can be evaluated by using either Eq. (40) or Eq. (41). To determine 20 coefficients in Eqs. (40) and (41), one needs to construct at least 20 linear equations. These equations are obtained in two ways. For example, by evaluating $u_h^{e+}(\mathbf{x})$ or $u_h^{e-}(\mathbf{x})$ at node $\mathbf{x}_k$ near the irregular element of interest in domain Ω_e^+, (or domain Ω_e^-), we have $u_h^{e+}(\mathbf{x}_k)$ (or $u_h^{e-}(\mathbf{x}_k)$), which gives rise to one linear equation. By selecting appropriate number of such nodes near the interface, we can determine many coefficients in Eqs. (40) and (41).

The other way to determine the coefficients is to enforce interface jump conditions. As discussed in Section 3.3.1, at each interface point, we have a set of four interface jump conditions, which determines four coefficients in principle. At least one set of interface jump conditions is enforced so that solution of the elliptic interface problem satisfies the interface jump conditions.

Question arises as how many nearby nodes and how many interface conditions are to be employed for the determination of 20 coefficients for each pair of interpolation functions $u_h^{e+}(\mathbf{x})$ and $u_h^{e-}(\mathbf{x})$. In fact, there is no unique way to select them. The proper choice depends on the local interface geometry. Due to the use of Cartesian mesh, topological relations among elements are straightforward, and such relations between an element and the interface are clear, which gives us the flexibility to select suitable regular nodes and interface points. Knowing these topological relations is advantageous when dealing with geometric singularities.

We illustrate in detail the algorithm of computing fictitious values in the rest of this section. Let us denote coefficients in Eqs. (40) and (41) as a vector $\mathbf{C}$

$$\mathbf{C} = (a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8, b_9)^T. \tag{42}$$

For node (i, j, k), we can have two linear equations

$$u_h^+(x_i, y_j, z_k) = (1, x_i, y_j, z_k, x_i^2, y_j^2, z_k^2, x_iy_j, x_iz_k, y_jz_k, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \cdot \mathbf{C}, \quad (x_i, y_j, z_k) \in \Omega_e^+ \tag{43}$$

$$u_h^-(x_i, y_j, z_k) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, x_i, y_j, z_k, x_i^2, y_j^2, z_k^2, x_iy_j, x_iz_k, y_jz_k) \cdot \mathbf{C}, \quad (x_i, y_j, z_k) \in \Omega_e^-. \tag{44}$$

It is possible to uniquely determine all 20 coefficients $\mathbf{C}$ by nodes selected from appropriate domains, which, however, lead to a solution in which the original interface jump conditions are not implemented. Therefore one needs to enforce at least one set of interface jump conditions.

At intersecting point o_1 between the interface and an element edge, we denote the local coordinate as $(x_{o_1}, y_{o_1}, z_{o_1})$. The derivatives at such a point can be expressed as

$$u_x^+ = (a_1 + 2a_4x + a_7y + a_8z) \tag{45}$$

$$u_x^- = (b_1 + 2b_4x + b_7y + b_8z) \tag{46}$$

$$u_y^+ = (a_2 + 2a_5y + a_7x + a_9z) \tag{47}$$

$$u_y^- = (b_2 + 2b_5y + b_7x + b_9z) \tag{48}$$

$$u_z^+ = (a_3 + 2a_6z + a_8x + a_9y) \tag{49}$$

$$u_z^- = (b_3 + 2b_6z + b_8x + b_9y). \tag{50}$$

We have four interface jump conditions at node o_1

$$\begin{aligned}
[u]_{o_1} &= (1, x_{o_1}, y_{o_1}, z_{o_1}, x_{o_1}^2, y_{o_1}^2, z_{o_1}^2, x_{o_1}y_{o_1}, x_{o_1}z_{o_1}, y_{o_1}z_{o_1}, \\
&\quad -1, -x_{o_1}, -y_{o_1}, -z_{o_1}, -x_{o_1}^2, -y_{o_1}^2, -z_{o_1}^2, -x_{o_1}y_{o_1}, -x_{o_1}z_{o_1}, -y_{o_1}z_{o_1}) \cdot \mathbf{C}, \qquad (51)\\
[\beta u_\xi]_{o_1} &= (0, \beta^+ \sin\phi\cos\theta, \beta^+ \sin\phi\sin\theta, \beta^+\cos\phi, 2\beta^+ \sin\phi\cos\theta x_{o_1}, 2\beta^+ \sin\phi\sin\theta y_{o_1}, 2\beta^+\cos\phi z_{o_1}, \\
&\quad \beta^+ \sin\phi\cos\theta y_{o_1} + \beta^+ \sin\phi\sin\theta x_{o_1}, \beta^+ \sin\phi\cos\theta z_{o_1} + \beta^+\cos\phi x_{o_1}, \\
&\quad \beta^+ \sin\phi\sin\theta z_{o_1} + \beta^+\cos\phi y_{o_1}, 0, -\beta^- \sin\phi\cos\theta, -\beta^- \sin\phi\sin\theta, -\beta^-\cos\phi, \\
&\quad -2\beta^- \sin\phi\cos\theta x_{o_1}, -2\beta^- \sin\phi\sin\theta y_{o_1}, -2\beta^-\cos\phi z_{o_1}, \\
&\quad -\beta^- \sin\phi\cos\theta y_{o_1} - \beta^- \sin\phi\sin\theta x_{o_1}, -\beta^- \sin\phi\cos\theta z_{o_1} - \beta^-\cos\phi x_{o_1}, \\
&\quad -\beta^- \sin\phi\sin\theta z_{o_1} - \beta^-\cos\phi y_{o_1}) \cdot \mathbf{C}, \\
[u_\eta]_{o_1} &= (0, -\sin\theta, \cos\theta, 0, -2\sin\theta x_{o_1}, 2\cos\theta y_{o_1}, 0, -\sin\theta y_{o_1} + \cos\theta x_{o_1}, -\sin\theta z_{o_1}, \cos\theta z_{o_1} \\
&\quad 0, \sin\theta, -\cos\theta, 0, 2\sin\theta x_{o_1}, -2\cos\theta y_{o_1}, 0, \sin\theta y_{o_1} - \cos\theta x_{o_1}, \sin\theta z_{o_1}, -\cos\theta z_{o_1}) \cdot \mathbf{C}, \\
[u_\zeta]_{o_1} &= (0, -\cos\phi\cos\theta, -\cos\phi\sin\theta, \sin\phi, -2\cos\phi\cos\theta x_{o_1}, -2\cos\phi\sin\theta y_{o_1}, 2\sin\phi z_{o_1}, \\
&\quad -\cos\phi\cos\theta y_{o_1} - \cos\phi\sin\theta x_{o_1}, -\cos\phi\cos\theta z_{o_1} + \sin\phi x_{o_1}, -\cos\phi\sin\theta z_{o_1} + \sin\phi y_{o_1}, \\
&\quad 0, \cos\phi\cos\theta, \cos\phi\sin\theta, -\sin\phi, 2\cos\phi\cos\theta x_{o_1}, 2\cos\phi\sin\theta y_{o_1}, -2\sin\phi z_{o_1}, \\
&\quad \cos\phi\cos\theta y_{o_1} + \cos\phi\sin\theta x_{o_1}, \cos\phi\cos\theta z_{o_1} - \sin\phi x_{o_1}, \cos\phi\sin\theta z_{o_1} - \sin\phi y_{o_1}) \cdot \mathbf{C}.
\end{aligned}$$

As demonstrated in Fig. 7, the interface intersecting points are chosen at the place where element edges intersect the interface surface. For different types of elements, the set of intersecting points is different too, see Fig. 7. Normally, two interface intersecting points are employed in our scheme, which results in eight jump conditions. Consequently, it appears that one just needs to choose 12 nodes in appropriate domains near the interface to determine 20 coefficients. In fact, such a choice may not work well because the resulting matrix may not be invertible for complex interface geometries. One way to resolve this problem is to select more than 12 nodes. Furthermore, to reduce the matrix condition number, it is important to select the nearest local nodes in a most symmetric manner [33]. To avoid confusion, the final matrix derived from our MIB Galerkin method is not a positive definite matrix as that obtained from the traditional finite element method. Therefore, to construct stable and accurate approximations, the number of nodes selected from Ω^+ should be similar to that from Ω^- when it is possible. In a normal case, nine nodes from each domain are chosen, which leads to eighteen additional linear equations. A linear matrix equation is constructed with eight interface jump conditions and eighteen node values,

$$\mathbf{A} \cdot \mathbf{C} = \mathbf{B}, \tag{52}$$

where $\mathbf{A}$ is a 26×20 matrix, and $\mathbf{B}$ is a vector containing function values at nearby nodes and interface jump conditions. Vector $\mathbf{C}$ comprises coefficients to be determined. As the linear matrix equation is over-determined, we make use of the least square method to compute an inverse matrix $\mathbf{A}'$ and represent vector $\mathbf{C}$ by

$$\mathbf{C} = (\mathbf{A}'\mathbf{A})^{-1}\mathbf{A}'\mathbf{B}. \tag{53}$$

Matrix $\mathbf{B}$ can be decomposed into two vectors $\mathbf{B}_1$ and $\mathbf{B}_2$, where vector $\mathbf{B}_1$ has eighteen components, i.e., function values at nodes. For the situation illustrated in Fig. 7, vector $\mathbf{B}_1$ contains function values at red node points. Here, vector $\mathbf{B}_2$ consists

of interface jump conditions at two interface-mesh intersecting points. For the situation illustrated in Fig. 7, vector $\mathbf{B}_2$ has components of interface jump conditions at point o_1 and o_2. Let us denote matrix $\mathbf{G}$ as

$$\mathbf{G} = (\mathbf{A}'\mathbf{A})^{-1}\mathbf{A}', \tag{54}$$

we can divide $\mathbf{G}$ into two matrices $\mathbf{G}_1$ and $\mathbf{G}_2$. Therefore, vector coefficient $\mathbf{C}$ can be expressed as

$$\{\mathbf{C}\}_{20\times 1} = (\{\mathbf{G}_1\}_{20\times 18}, \{\mathbf{G}_2\}_{20\times 8}) \cdot \begin{pmatrix} \{\mathbf{B}_1\}_{18\times 1} \\ \{\mathbf{B}_2\}_{8\times 1} \end{pmatrix} \tag{55}$$

$$\{\mathbf{C}\}_{20\times 1} = \{\mathbf{G}_1\}_{20\times 18}\{\mathbf{B}_1\}_{18\times 1} + \{\mathbf{G}_2\}_{20\times 8}\{\mathbf{B}_2\}_{8\times 1}. \tag{56}$$

After the determination of vector $\mathbf{C}$, fictitious values are evaluated at their locations. In Fig. 7, for the irregular element marked by vertices A, B, C and D, fictitious values can be represented as

$$f_h^+(x_A, y_A, z_A) = (1, x_A, y_A, z_A, x_A^2, y_A^2, z_A^2, x_Ay_A, y_Az_A, x_Az_A, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \cdot \mathbf{C} \tag{57}$$

$$f_h^+(x_B, y_B, z_B) = (1, x_B, y_B, z_B, x_B^2, y_B^2, z_B^2, x_By_B, y_Bz_B, x_Bz_B, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \cdot \mathbf{C} \tag{58}$$

$$f_h^-(x_C, y_C, z_C) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, x_C, y_C, z_C, x_C^2, y_C^2, z_C^2, x_Cy_C, y_Cz_C, x_Cz_C) \cdot \mathbf{C} \tag{59}$$

$$f_h^-(x_D, y_D, z_D) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, x_D, y_D, z_D, x_D^2, y_D^2, z_D^2, x_Dy_D, y_Dz_D, x_Dz_D) \cdot \mathbf{C}. \tag{60}$$

Denote $\mathbf{T}$ as a local coefficient vector. One can write a fictitious value as $f_h = \mathbf{T} \cdot \mathbf{C}$. By using Eqs. (55) and (56), one obtains a general expression for a fictitious value

$$\begin{aligned} f_h &= \{\mathbf{T}\}_{1\times 20} \cdot (\{\mathbf{G}_1\}_{20\times 18}\{\mathbf{B}_1\}_{18\times 1} + \{\mathbf{G}_2\}_{20\times 8}\{\mathbf{B}_2\}_{8\times 1}) \\ &= \{\mathbf{T}\}_{1\times 20}\{\mathbf{G}_1\}_{20\times 18}\{\mathbf{B}_1\}_{18\times 1} + \{\mathbf{T}\}_{1\times 20}\{\mathbf{G}_2\}_{20\times 8}\{\mathbf{B}_2\}_{8\times 1}. \end{aligned} \tag{61}$$

Once the mesh is set up for a given interface, the local information for each node and each interface intersecting point is known. As a result, vectors $\mathbf{B}_2$ and $\mathbf{T}$ are also known. Matrices $\mathbf{G}_1$ and $\mathbf{G}_2$ are evaluated from Eq. (54). Vector $\mathbf{B}_1$ contains approximation function values at mesh nodes. When the expression of the fictitious values is obtained, it is inserted into the discretization matrix of the governing equation, Eq. (24) or Eq. (25), to solve the linear matrix equation, which results in the solution at all the nodes.

We also consider the weighted least square method in solving the Eq. (52). By using a diagonal weight matrix $\mathbf{W}$, which is 26×26 in dimension, the vector $\mathbf{C}$ can be represented as

$$\mathbf{C} = (\mathbf{A}'\mathbf{W}\mathbf{A})^{-1}\mathbf{A}'\mathbf{W}\mathbf{B}. \tag{62}$$

It is seen that if we set the corresponding diagonal element $W_{i,i}$ as zero for the two tangential direction jump conditions, then we only consider the traditional jump conditions in our schemes. We can also assign a given node a weight according to its distance from the interface to adjust its importance in the scheme.

For the situation of a geometric singularity, it may not be easy to find an enough number of nodes in a symmetric manner from both subdomains. So in each subdomain, we always choose nodes within nearest distance to the element of interest. A major concern is still the reversibility of the matrix $A'A$. As far as we know, there is no straightforward algorithm for nodes selection with the guaranteed matrix reversibility. Therefore, computationally, we systematically add a pair of nodes from two regions until the matrix is reversible. This procedure is found to work well.

3.3.3. Scheme for the derivative of the solution

Since the present MIB Galerkin method using the Cartesian mesh, the derivative of the solution on mesh nodes can be approximated by central difference schemes. For a regular node (x_i, y_j, z_k), the derivative of the solution can be written as

$$(u_h^+)_x(x_i, y_{j,}, z_k) = \frac{u_h^+(x_{i+1}, y_j, z_k) - u_h^+(x_{i-1}, y_j, z_k)}{2dx}, \tag{63}$$

$$(u_h^+)_y(x_i, y_j, z_k) = \frac{u_h^+(x_i, y_{j+1}, z_k) - u_h^+(x_i, y_{j-1}, z_k)}{2dy}, \tag{64}$$

$$(u_h^+)_z(x_i, y_j, z_k) = \frac{u_h^+(x_i, y_j, z_{k+1}) - u_h^+(x_i, y_j, z_{k-1})}{2dz}, \tag{65}$$

where dx, dy and dz are the grid sizes of the mesh at x, y and z directions, respectively. If node (x_i, y_j, z_k) is an irregular one and adjacent nodes (x_{i+1}, y_j, z_k), (x_i, y_{j+1}, z_k) and (x_i, y_j, z_{k+1}) locate on the other side of the interface. Fictitious values are used and central difference schemes are

$$(u_h^+)_x(x_i, y_j, z_k) = \frac{u_h^+(x_{i+1}, y_j, z_k) - f_h^+(x_{i-1}, y_j, z_k)}{2dx}, \tag{66}$$

$$(u_h^+)_y(x_i, y_j, z_k) = \frac{f_h^+(x_i, y_{j+1}, z_k) - u_h^+(x_i, y_{j-1}, z_k)}{2dy}, \tag{67}$$

$$(u_h^+)_z(x_i, y_j, z_k) = \frac{f_h^+(x_i, y_j, z_{k+1}) - u_h^+(x_i, y_j, z_{k-1})}{2dz}. \tag{68}$$

It is seen that if we calculate solutions u_h^+ and u_h^-, fictitious values can be attained through Eq. (61).

For convenience, let us denote $h = dx = dy = dz$. As we employ the second order polynomial to approximate the fictitious values in Eq. (40), we have the relation

$$u_h^{e+}(x_i, y_j, z_k) = u(x_i, y_j, z_k) + O(h^3). \tag{69}$$

This relation can be used to estimate the approximation order of the derivative scheme at an irregular node

$$\begin{aligned}(u_h^+)_x(x_i, y_j, z_k) &= \frac{f_h^+(x_{i+1}, y_j, z_k) - u_h^+(x_{i-1}, y_j, z_k)}{2h} \\ &\le \frac{|u_h^+(x_{i+1}, y_j, z_k) - u(x_{i+1}, y_j, z_k)| + |u(x_{i-1}, y_j, z_k) - f_h^+(x_{i-1}, y_j, z_k)|}{2h} \\ &\quad + \frac{|u(x_{i+1}, y_j, z_k) - u(x_{i-1}, y_j, z_k)|}{2h} \\ &= u_x(x_i, y_j, z_k) + O(h^2),\end{aligned} \tag{70}$$

where $u(x_{i+1}, y_j, z_k)$ and $u(x_{i-1}, y_j, z_k)$ are exact values and $u_x(x_i, y_j, z_k)$ is the exact derivative. We have taken the advantage of an error cancellation due to the symmetry of the central scheme. Similar estimations can be obtained for $(u_h^+)_y(x_i, y_j, z_k)$, $(u_h^+)_z(x_i, y_j, z_k)$, $(u_h^-)_x(x_i, y_j, z_k)$, $(u_h^-)_y(x_i, y_j, z_k)$, $(u_h^-)_z(x_i, y_j, z_k)$. Therefore, we can attain the second order accuracy for the derivatives even at irregular nodes.

4. Numerical studies

In this section, we examine validity and test the performance of the proposed MIB Galerkin FEM for solving the 3D Poisson equation with discontinuous coefficients. To avoid misunderstanding, the L_∞ error of the derivative of the solution is defined as:

$$\begin{aligned}L_\infty = \max_{\forall(x_i,y_j)\in\Omega\setminus\partial\Omega} \{&|u_x(x_i, y_j, z_k) - (u_h)_x(x_i, y_j, z_k)|, |u_y(x_i, y_j, z_k) - (u_h)_y(x_i, y_j, z_k)|, \\ &|u_z(x_i, y_j, z_k) - (u_h)_z(x_i, y_j, z_k)|\},\end{aligned} \tag{71}$$

where $u_x(x_i, y_j, z_k)$, $u_y(x_i, y_j, z_k)$ and $u_z(x_i, y_j, z_k)$ are exact derivative values.

To efficiently implement the present 3D MIB Galerkin method, we have made use of a FORTRAN90 library, FELIPPA (http://people.sc.fsu.edu/jburkardt/f_src/felippa/felippa.html), for finite element integrations. Among our three elements, the five-tetrahedron element and six-tetrahedron element share the same form of integration. The KEAST subroutine in the library is used for the element integration. For the rectangular prism element, the hexahedral integration form in the library is utilized.

Due to the introduction of fictitious values, the discretization matrix is not positive definite, nor is it symmetric. For an irregular node, we need to make use of 9 nodes from each subdomain, which leads to 18 nodes for every irregular node. However, we always select the nearest nodes in a symmetric manner in our scheme to make the discretization matrix as symmetric as possible. Additionally, since the interface is inherently a 2D surface in a 3D problem, the major portion of the discretization matrix remains symmetric. Therefore, the condition number of the final sparse matrix does not increase much because of our implementation of interface conditions. As a result, the final linear equation can be iteratively solved by many Krylov subspace based linear solvers. One of solvers that is often used is the biconjugate gradient (BICG) method. Many other matrix solvers in PETSc (http://www.mcs.anl.gov/petsc/) or SLATEC (http://sdphca.ucsd.edu/slatec_source/TOC.htm) can also be employed as well.

We carry out many numerical experiments with complex interfaces, geometric singularities and solution singularities. In Case 1, a spherical interface is employed to compare the performances of three different elements, i.e., rectangular prism element, five-tetrahedron element and six-tetrahedron element. As the former two give stable results, we further compare these two elements for more complex interfaces and geometric singularities in Case 2 and Case 3. Cases 4 and 5 are designed to demonstrate the performance of the present method for protein interfaces, which are arbitrarily complex. In Cases 6 and 7, complex interface geometries for protein complexes obtained from Electron Microscopy Data Bank (http://www.ebi.ac.uk/pdbe/emdb/) are utilized to further test the proposed MIB Galerkin method for potential applications in molecular biology. Different solutions and beta functions are tested. Finally, Case 8 is employed to examine the present method for the low regularity solution associated with geometric singularities. The present 3D MIB Galerkin method, equipped with the rectangular prism element, gives the second order convergence in these test problems.

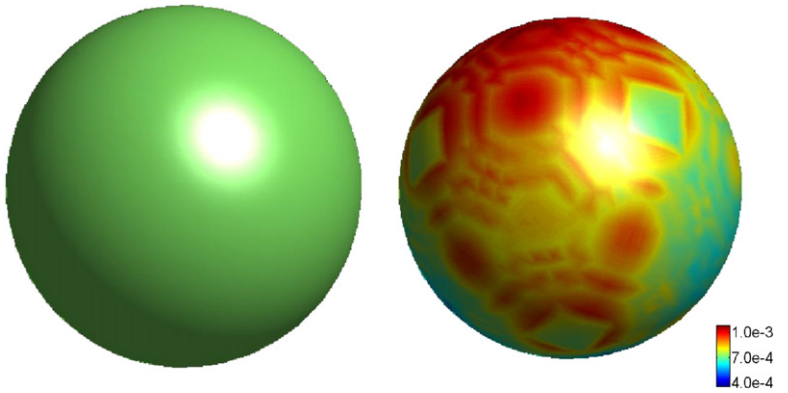

Fig. 8. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 1(a).

Table 1
Numerical errors and convergence orders for rectangular prism element results (Case 1a).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	1.347e−2		7.860e−3		1.600		2.690e−1	
$40 \times 40 \times 40$	3.797e−3	1.83	2.141e−3	1.88	5.740e−1	1.48	8.286e−2	1.70
$80 \times 80 \times 80$	1.112e−3	1.77	5.605e−4	1.93	1.729e−1	1.73	2.298e−2	1.85
$160 \times 160 \times 160$	2.760e−4	2.01	1.282e−4	2.13	4.753e−2	1.86	6.040e−3	1.93

Table 2
Numerical errors and convergence orders for five-tetrahedron element results (Case 1a).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	0.6117		8.549e−2		3.003		0.3737	
$40 \times 40 \times 40$	0.2147	1.51	2.255e−2	1.92	1.552	0.95	0.1278	1.55
$80 \times 80 \times 80$	6.378e−2	1.75	6.409e−3	1.81	0.7960	0.96	4.391e−2	1.54
$160 \times 160 \times 160$	1.739e−2	1.87	1.644e−3	1.96	0.5151	0.63	1.494e−2	1.56

Case 1. We first investigate a classical spherical interface problem. The 3D Poisson equation (1)–(4) is solved in domain $[-1, 1] \times [-1, 1] \times [-1, 1]$. To specify the spherical interface, we design level set function $\phi(x, y, z)$

$$\phi(x, y, z) = r_0^2 - (x^2 + y^2 + z^2), \tag{72}$$

where the radius $r_0 = 1/3$. The interface is given as $\Gamma = \{(x, y, z)|\phi = 0,\ \forall(x, y, z) \in \Omega\}$. Two subdomains are $\Omega^+ = \{(x, y, z)|\phi \geq 0,\ \forall(x, y, z) \in \Omega\}$, and $\Omega^- = \{(x, y, z)|\phi < 0;\ \forall(x, y, z) \in \Omega\}$. The solution in two different subdomains is chosen as

$$u^+(x, y, z) = 10(x + y + z), \quad \text{and} \quad u^-(x, y, z) = 5e^{(x^2+y^2+z^2)} + 20. \tag{73}$$

The discontinuous coefficients are given by

- **Case 1(a):**

$$\beta^+(x, y, z) = 4, \quad \text{and} \quad \beta^-(x, y, z) = 1. \tag{74}$$

- **Case 1(b):**

$$\beta^+(x, y, z) = 1 + \cos(x + y + z), \quad \text{and} \quad \beta^-(x, y, z) = 1 + 3\sin(x + y + z). \tag{75}$$

Here Case 1(b) is designed to study the ability of the present method for handling varying coefficients. The source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be given accordingly.

In this case, we examine the performance of three MIB Galerkin finite elements. Table 1 lists numerical results for the rectangular prism element. The geometry and error are depicted in Fig. 8. It is seen that errors are very small from the rectangular prism element at the fine mesh. The second order convergence is found.

Tables 2 and 3 show the results for five-tetrahedron element and six-tetrahedron element, respectively. It can be seen that the rectangular prism element and the five-tetrahedron element basically demonstrate the near second order accuracy. For the six-tetrahedron element, its order of convergence is just slightly better than one. In terms of accuracy, the rectangular prism element is much more accurate than the tetrahedral elements. The five-tetrahedron element is generally more accurate than six-tetrahedron element, although in the coarse grid the five-tetrahedron element method may have a larger L_∞ error.

Table 3
Numerical errors and convergence orders for six-tetrahedron element results (Case 1a).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	0.3913		0.1430		2.350		4.424e−1	
$40 \times 40 \times 40$	0.1778	1.14	5.485e−2	1.38	8.553e−1	1.46	1.533e−1	1.53
$80 \times 80 \times 80$	8.363e−2	1.09	2.325e−2	1.24	4.049e−1	1.08	5.496e−2	1.48
$160 \times 160 \times 160$	3.626e−2	1.21	9.560e−3	1.28	1.862e−1	1.12	2.035e−2	1.43

Table 4
Numerical errors and convergence orders for rectangular prism element results (Case 1b).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	3.231e−2		7.635e−3		1.679		2.687e−1	
$40 \times 40 \times 40$	8.174e−3	1.98	2.062e−3	1.89	5.919e−1	1.50	8.286e−2	1.70
$80 \times 80 \times 80$	2.042e−3	2.00	5.161e−4	2.00	1.761e−1	1.75	2.298e−2	1.85
$160 \times 160 \times 160$	5.542e−4	1.88	1.298e−4	1.99	4.801e−2	1.87	6.048e−3	1.93

Table 5
Numerical errors and convergence orders for five-tetrahedron element results (Case 1b).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	6.193e−1		7.952e−2		3.117		3.655e−1	
$40 \times 40 \times 40$	2.157e−1	1.52	2.198e−2	1.86	1.581	0.98	1.272e−1	1.52
$80 \times 80 \times 80$	6.388e−2	1.76	5.888e−3	1.90	8.012e−1	0.98	4.335e−2	1.55
$160 \times 160 \times 160$	1.740e−2	1.88	1.498e−3	1.97	5.161e−1	0.63	1.479e−2	1.55

Table 6
Numerical errors and convergence orders for six-tetrahedron element results (Case 1b).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	1.405		9.033e−2		13.76		4.630e−1	
$40 \times 40 \times 40$	6.366e−1	1.14	2.667e−2	1.76	12.47	*	1.833e−1	1.34
$80 \times 80 \times 80$	4.866e−1	0.39	1.122e−2	1.25	19.65	*	9.922e−2	0.89
$160 \times 160 \times 160$	3.192e−1	0.61	6.597e−3	0.77	17.38	*	5.623e−2	0.82

To further investigate these methods, we change the coefficients into a position dependent function in Case 1(b). Results are given in Tables 4–6, respectively for the rectangular prism element, the six-tetrahedron element and the six-tetrahedron element. Clearly, the rectangular prism element still demonstrates a near second order accuracy in both L_∞ and L_2 norms for the solution. Its accuracy order for the derivative is about 1.7 in both L_∞ and L_2 norms. The tetrahedral elements do not perform as well as the rectangular prism element. In particular, the six-tetrahedron element does not offer a convergent derivative in the L_∞ norm. There are two reasons for the six-tetrahedron element to lose its accuracy and convergence order. First, mesh quality of the six-tetrahedron element is generally not as good as that of five-tetrahedron element. It can been seen from Fig. 4 that many angles of the six-tetrahedron element are smaller than 45°. It is well known that equal-angled elements have the best mesh quality. Additionally, in the present interface method, interface jump conditions are enforced on a set of special points where the interface intersects element edges. For the six-tetrahedron element, these intersecting points may be too close to each other and cause the loss of accuracy in calculating the coefficients in Eq. (55). We drop the six-tetrahedron element in the rest of our numerical study.

Case 2. We next consider the surface of four spheres that are packed in domain $\Omega : [-5, 5] \times [-5, 5] \times [-5, 5]$, as shown in Fig. 9. The interface can be characterized by a level set function $\phi(x, y, z)$

$$\begin{aligned}\phi(x, y, z) = &-(r_0 - \sqrt{(x - r_0)^2 + (y - r_0)^2 + z^2})(r_0 - \sqrt{(x - r_0)^2 + (y + r_0)^2 + z^2}) \\ &\times (r_0 - \sqrt{(x + r_0)^2 + (y - r_0)^2 + z^2})(r_0 - \sqrt{(x + r_0)^2 + (y + r_0)^2 + z^2}),\end{aligned} \tag{76}$$

where the radius $r_0 = 5/3$. The interface is given as $\Gamma = \{(x, y, z)|\phi = 0,\ \forall(x, y, z) \in \Omega\}$. Two subdomains of the box are $\Omega^+ = \{(x, y, z)|\phi \geq 0,\ \forall(x, y, z) \in \Omega\}$, and $\Omega^- = \{(x, y, z)|\phi < 0;\ \forall(x, y, z) \in \Omega\}$. The 3D Poisson equation (1)–(4) is solved on domain Ω. The discontinuous coefficients are given by

$$\beta^+(x, y, z) = 4, \quad \text{and} \quad \beta^-(x, y, z) = 1. \tag{77}$$

The solution in two different subdomains is chosen as

$$u^+(x, y, z) = 10(x + y + z), \quad \text{and} \quad u^-(x, y, z) = \cos(x)\cos(y)\cos(z) + 20. \tag{78}$$

The source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be given accordingly.

In this case, we further test the performance of rectangular prism element and five-tetrahedron element. A major challenge in this case is geometric singularities around the touching points of spheres. These geometry singularities are

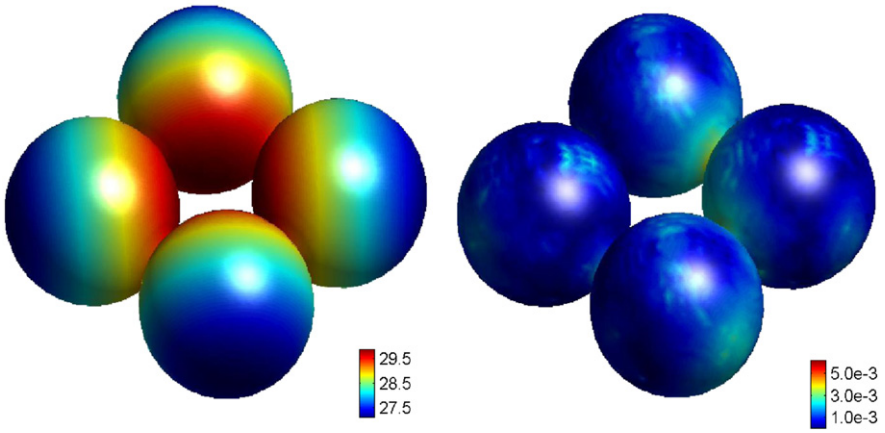

Fig. 9. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 2.

Table 7
Numerical errors and convergence orders for rectangular prism element results (Case 2).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	7.106e−2		9.685e−3		2.039e−1		1.226e−2	
$40 \times 40 \times 40$	1.627e−2	2.13	1.378e−3	2.81	8.952e−2	1.19	2.951e−3	2.05
$80 \times 80 \times 80$	6.433e−3	1.34	4.475e−4	1.62	3.627e−2	1.30	7.963e−4	1.89
$160 \times 160 \times 160$	2.058e−3	1.64	1.172e−4	1.93	2.443e−2	0.57	2.273e−4	1.81

Table 8
Numerical errors and convergence orders for five-tetrahedron element results (Case 2).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order	$L_\infty(\nabla u)$	Order	$L_2(\nabla u)$	Order
$20 \times 20 \times 20$	0.2010		3.249e−2		5.256e−1		2.837e−2	
$40 \times 40 \times 40$	4.891e−2	2.04	6.075e−3	2.42	2.838e−1	0.89	8.118e−3	1.81
$80 \times 80 \times 80$	1.449e−2	1.76	1.897e−3	1.68	1.319e−1	1.11	2.370e−3	1.78
$160 \times 160 \times 160$	5.669e−3	1.35	7.085e−4	1.42	5.334e−2	1.30	7.739e−4	1.61

generated by the close contact of spheres and generally tend to induce numerical instability in elliptic interface algorithms. Fig. 9 depicts the numerical solution and error on an $80 \times 80 \times 80$ mesh. From results demonstrated in Tables 7 and 8, it can be seen that the MIB Galerkin method can deliver a near second order accuracy, employing either the rectangular prism element or the five-tetrahedron element. Just like in Case 1, the rectangular prism element works slightly better than the five-tetrahedron element. Similar situations are found in more numerical studies. Therefore, we focus our attention to the rectangular prism element in the rest of this section.

Case 3. We next consider the surface of four parallel contacting cylinders, as shown in Fig. 9. This surface is immersed in domain $[-1, 1] \times [-1, 1] \times [-1, 1]$, in which the 3D Poisson equation (1)–(4) is solved. We design a level set function $\phi(x, y, z)$ to define the interface

$$\phi(x,y,z) = \begin{cases} -[r_0 - \sqrt{(x-r_0)^2 + (y-r_0)^2}][r_0 - \sqrt{(x-r_0)^2 + (y+r_0)^2}] & \\ \times[r_0 - \sqrt{(x+r_0)^2 + (y-r_0)^2}][r_0 - \sqrt{(x+r_0)^2 + (y+r_0)^2}] & |z| \le z_0 \\ z & |z| > z_0, \end{cases} \tag{79}$$

where the radius $r_0 = 1/3$, $z_0 = 2/3$. Discontinuous coefficients are two constants

$$\beta^+(x,y,z) = 4, \quad \text{and} \quad \beta^-(x,y,z) = 10. \tag{80}$$

The solution in two different subdomains is given by

$$u^+(x,y,z) = 10, \quad \text{and} \quad u^-(x,y,z) = \cos(x)\cos(y)\cos(z). \tag{81}$$

Based on this information, the source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be derived.

Similar to the last case, this problem involves geometric singularities. Additionally, parallel cylinders may lead to a singular matrix which does not have an inverse. In order to avoid singular matrix, parallel cylinders are tilted towards the body diagonal direction of the Cartesian mesh. If we define the azimuth angle θ and zenith angle ϕ, four cylinders as a whole will tilt around the original point $(0, 0, 0)$ with $\cos\theta = 0.3$ and $\cos\phi = 1.0$. In our study, we further explore the performance of the rectangular prism element. Fig. 10 depicts the numerical solution and error on an $80 \times 80 \times 80$ mesh.

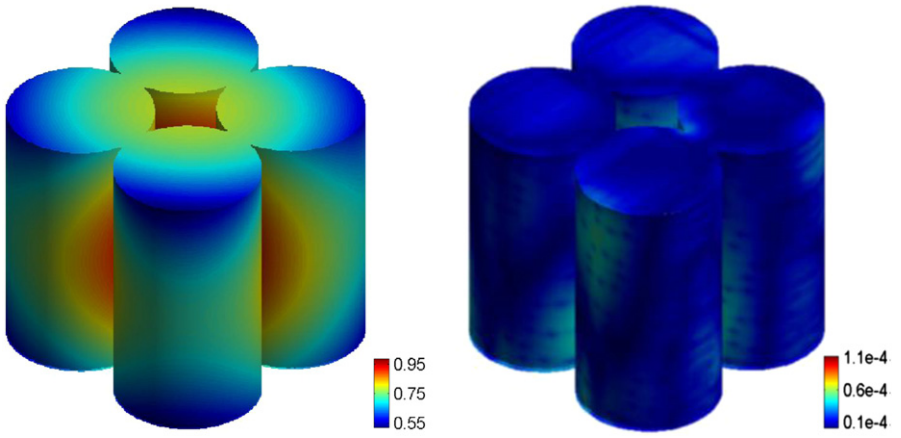

Fig. 10. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 3.

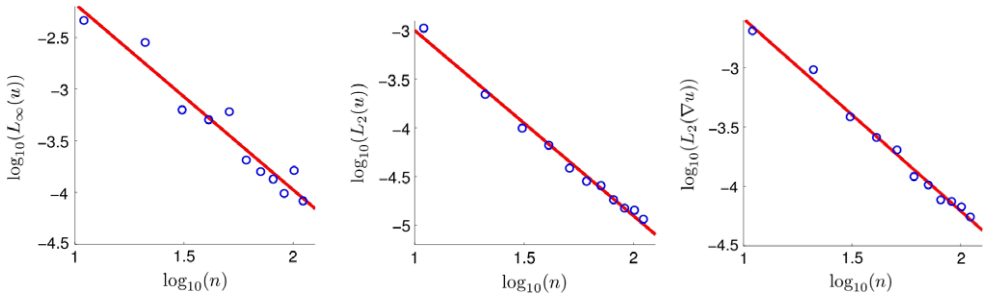

Fig. 11. L_∞ errors for solution (left chart), L_2 errors for solution (middle chart) and L_2 errors for derivative (right chart) for Case 3 ($n = n_x = n_y = n_z$).

Table 9
Numerical errors and convergence orders of MIB Galerkin method for protein 1ajj (Case 4).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order
$20 \times 20 \times 20$	4.702e−2		1.849e−2	
$40 \times 40 \times 40$	1.169e−2	2.01	4.523e−3	2.03
$80 \times 80 \times 80$	4.665e−3	1.33	1.215e−3	1.90
$160 \times 160 \times 160$	1.115e−3	2.06	3.330e−4	1.87

Fig. 11 shows errors of the rectangular prism element. The L_∞ convergence order for the solution is 1.80, the L_2 convergence for the solution is 1.91 and the L_2 convergence for the derivative is 1.62.

Case 4. In this case, we consider a truly arbitrarily complex interface geometry—the surface of a low-density lipoprotein receptor (LDLR) which is responsible for the uptake of cholesterol-containing lipoprotein particles into cells. The surface of LDLR is immersed in domain $[-1, 1] \times [-1, 1] \times [-1, 1]$, in which the 3D Poisson equation (1)–(4), with discontinuous coefficients given by

$$\beta^+(x, y, z) = 3 + x + y + z, \quad \text{and} \quad \beta^-(x, y, z) = 2(x^2 + y^2 + z^2). \tag{82}$$

The analytical solution in two different subdomains is set to

$$u^+(x, y, z) = 10(x^2 + y^2 + z^2), \quad \text{and} \quad u^-(x, y, z) = 5\cos(x^2 + y^2 + z^2). \tag{83}$$

The source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be derived accordingly.

The original information of the LDLR is obtained from the Protein Data Bank (http://www.rcsb.org, PDB ID:1ajj). We generate the protein surface by the minimal molecular surface (MMS) [87]. The 3D volumetric data used for generating the final MMS was cosmetically polished by 3 time steps. As it can be seen from Table 9, the second order accuracy is attained in both L_∞ and L_2 norms. Fig. 12 depicts the numerical solution and error on an $80 \times 80 \times 80$ mesh.

Many FEMs have been constructed for solving the Poisson equation for electrostatic analysis of proteins in the literature. Although many FEMs were tested against the spherical interface to validate their second order accuracy, their performance for realistic protein surfaces has not been directly validated, to our knowledge. Therefore, the proposed MIB Galerkin method

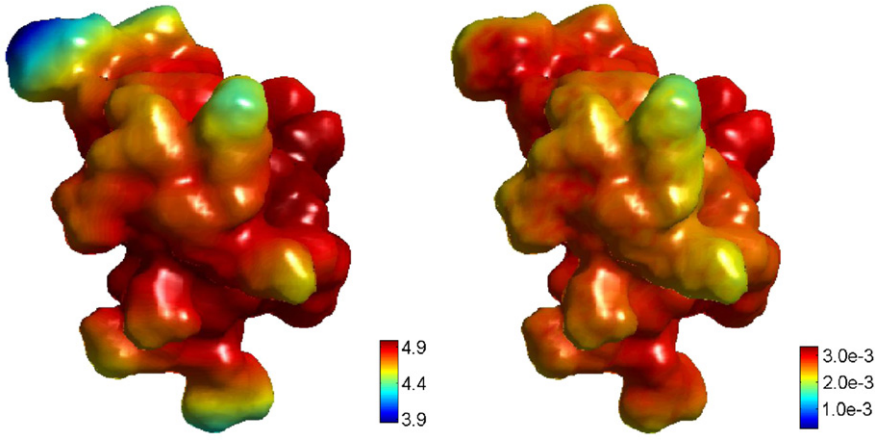

Fig. 12. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 4.

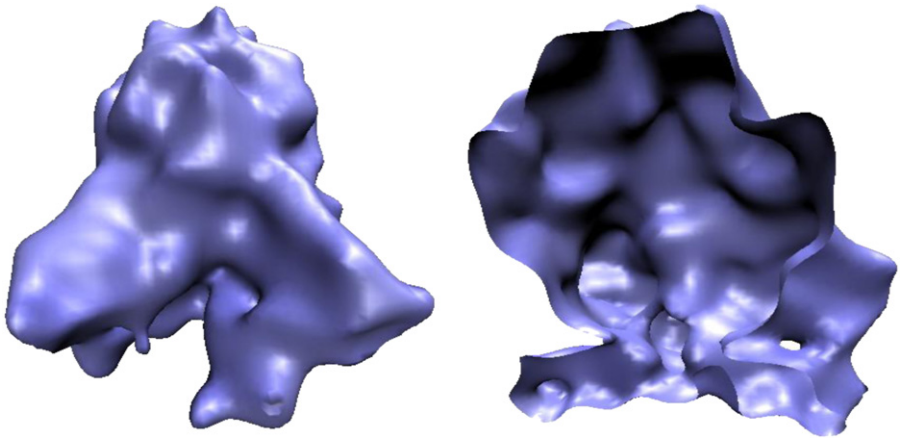

Fig. 13. Illustration of a BK channel protein surface (left chart) and its cross section (right chart) used in Case 5.

is the first FEM that has been validated against a realistic protein surface. The only other numerical method that has been confirmed its second order accuracy for protein surfaces in the literature is the original MIB method [46–48].

Case 5. We next consider a more complex protein surface, the BK channel protein (emd5114), to further study the proposed MIB Galerkin method. BK channels are essential for the regulation of many important physiological processes including neuronal excitation, muscle contraction, and the electrical tuning of hair cells. The surface of emd5114 is very complex as illustrated in Fig. 13. In this test, the emd5114 surface is immersed in domain $[-1, 1] \times [-1, 1] \times [-1, 1]$, on which the 3D Poisson equation (1)–(4) is solved. The discontinuous coefficients are given by two functions

$$\beta^+(x, y, z) = 3 + x + y + z, \quad \text{and} \quad \beta^-(x, y, z) = 2(x^2 + y^2 + z^2). \tag{84}$$

We design the solution in two different subdomains as

$$u^+(x, y, z) = 10(x^2 + y^2 + z^2), \quad \text{and} \quad u^-(x, y, z) = 5\cos(x^2 + y^2 + z^2) + 100. \tag{85}$$

The source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be derived accordingly.

The original data of the BK channel protein is downloaded from the Electron Microscopy Data Bank (http://emdatabank.org/index.html, EMDB ID: emd5114). The recommended contour value of 0.229 is used to generate the surface. However, due to the noise, we remove the noise in the original 3D data by 2 time steps with the mean curvature flow [87] before taking the isosurface. The numerical solution and error are depicted on an $80 \times 80 \times 80$ mesh in Fig. 14. More quantitative results are given in Table 10. Obviously, the second order accuracy is attained for this case.

Case 6. Having validated the second order accuracy for protein surfaces, we further examine the accuracy of the present MIB Galerkin method for a molecular motor—vacuolar ATPase complex, which is a key regulator of the acidification of endosomes, lysosomes, and the luminal compartments of several cell types, tissues, and organs. Original data are obtained from the Electron Microscopy Data Bank (http://emdatabank.org/index.html, ID:emd1590). The recommended isovalue of 0.05 is used to generate surface as shown in Fig. 15. The 3D Poisson equation (1)–(4) is solved in domain

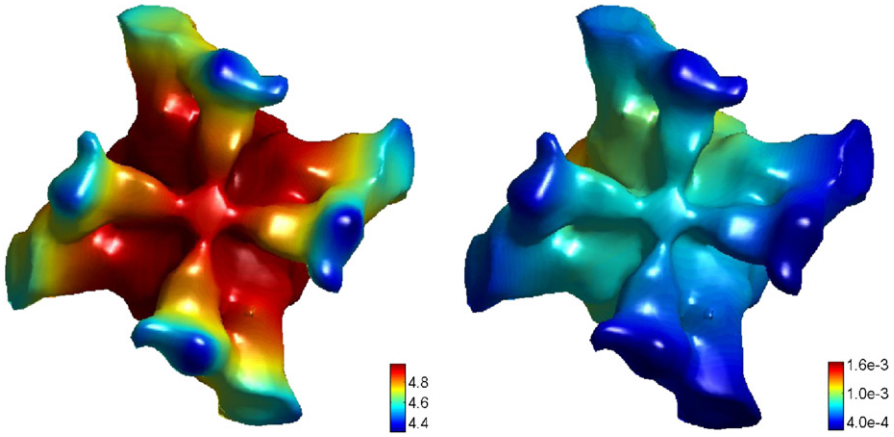

Fig. 14. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 5.

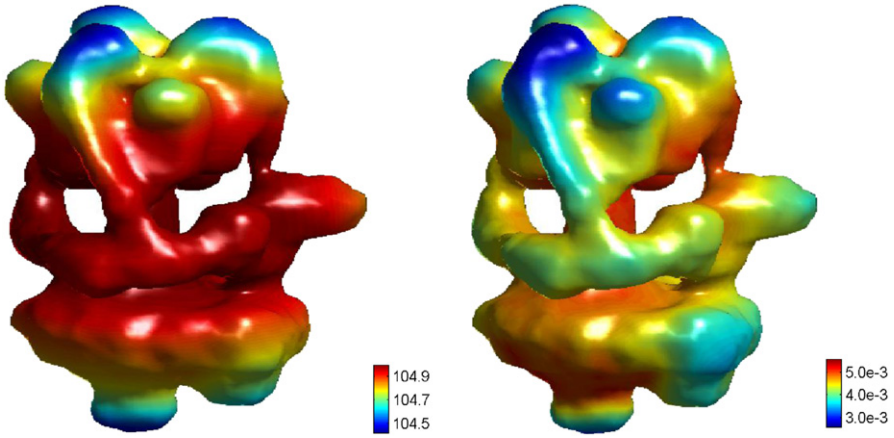

Fig. 15. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 6.

Table 10
Numerical errors and convergence orders of MIB Galerkin method for BK channel protein (Case 5).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order
$20 \times 20 \times 20$	2.612e−2		5.375e−3	
$40 \times 40 \times 40$	7.214e−3	1.86	1.544e−3	1.80
$80 \times 80 \times 80$	1.901e−3	1.92	4.224e−4	1.87
$160 \times 160 \times 160$	5.801e−4	1.71	1.077e−4	1.97

$[-1, 1]\times[-1, 1]\times[-1, 1]$ including the surface of the vacuolar ATPase complex. The discontinuous coefficients are given by

$$\beta^+(x, y, z) = 5 + x + y + z, \quad \text{and} \quad \beta^-(x, y, z) = 10(x^2 + y^2 + z^2). \tag{86}$$

The solution in two different subdomains is given as

$$u^+(x, y, z) = \sin(x)\sin(y)\sin(z), \quad \text{and} \quad u^-(x, y, z) = 5\cos(x^2 + y^2 + z^2) + 100. \tag{87}$$

The source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be given accordingly.

Our results for the solution are listed in Fig. 16. Demonstrated by the least square fitting, the convergence order for L_∞ is 2.08 and that for L_2 is 2.06. This test result indicates that the proposed MIB Galerkin performs well for multiprotein complexes.

Case 7. We next employ a Type III secretion system (T3SS), emd1617, to further validate the present MIB Galerkin method for multiprotein complexes. T3SSs are protein transport devices used by many Gram-negative bacteria to inject "effector" proteins into the plasma membrane of host cells so as to manipulate many important cell processes. The surface of emd1617 has a C_{12} symmetry as shown in Fig. 17 and is immersed in domain $[-1, 1] \times [-1, 1] \times [-1, 1]$. The 3D Poisson equation

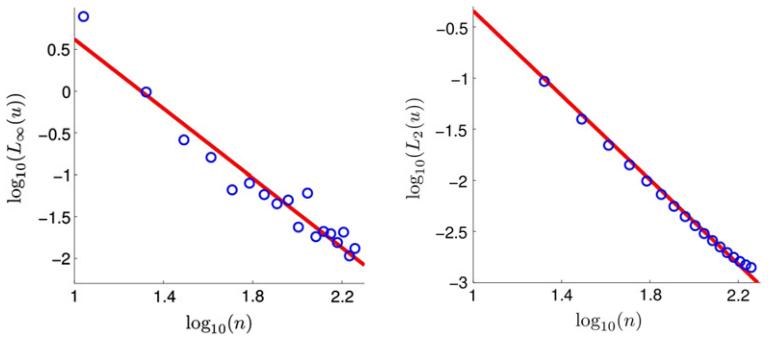

Fig. 16. L_∞ errors (left chart) and L_2 errors (right chart) for Case 6 ($n = n_x = n_y = n_z$).

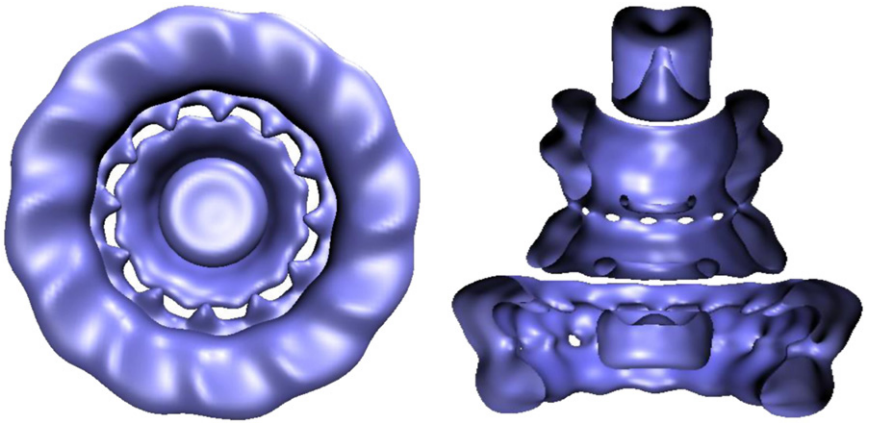

Fig. 17. Illustration of a Type III secretion system surface (left chart) and its cross section (right chart) used in Case 7.

(1)–(4) is solved on this domain with the discontinuous coefficients being given by

$$\beta^+(x, y, z) = 4, \quad \text{and} \quad \beta^-(x, y, z) = 1. \tag{88}$$

We set the solution in two different subdomains as

$$u^+(x, y, z) = \sin(x)\sin(y)\sin(z), \quad \text{and} \quad u^-(x, y, z) = 5\cos(x^2 + y^2 + z^2). \tag{89}$$

Accordingly, one can derive source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$.

The original data from the Electron Microscopy Data Bank are corrupted with much noise. We use our mean curvature flow method [87] to remove the noise in the Cryo-EM data with 50 time steps before extraction of the surface at the recommended contour value of 1.0. The numerical solution and error are depicted in Fig. 18 with an $80 \times 80 \times 80$ resolution. Quantitative error analysis is displayed in Fig. 19. With the least square method employed for data fitting, the convergence order of L_∞ is 1.72 and that of L_2 is 1.77. The results from this case and the last three examples indicate that the present method has a great potential to be used in the biomolecular modeling and computation of macromolecules.

Case 8. Having established the second order accuracy for arbitrarily complex interfaces from proteins and protein complexes, we are interested in examining the present method for low solution regularities associated with geometric singularities. Recently, the Wang–Ye Galerkin FEM [22] has established the first known second order convergence for this class of problems in 2D setting [23]. However, no second order accurate result has been reported in 3D setting, to our knowledge. In the present work, we design level set function $\phi(x, y, z)$

$$\phi(x, y, z) = \begin{cases} -1 & x > 0 \\ (x + y)(x - y) & x \le 0, \end{cases} \tag{90}$$

to characterize the interface of Lipschitz continuous, on which the 3D Poisson equation (1)–(4) is solved in domain $[-1, 1] \times [-1, 1] \times [-1, 1]$. We set the discontinuous coefficients to

$$\beta^+(x, y, z) = 4, \quad \text{and} \quad \beta^-(x, y, z) = 1. \tag{91}$$

We consider two solutions in our test study.

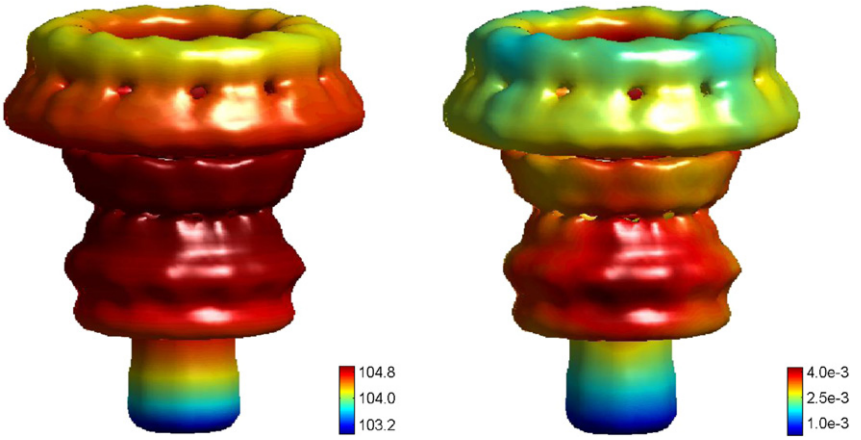

Fig. 18. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 7.

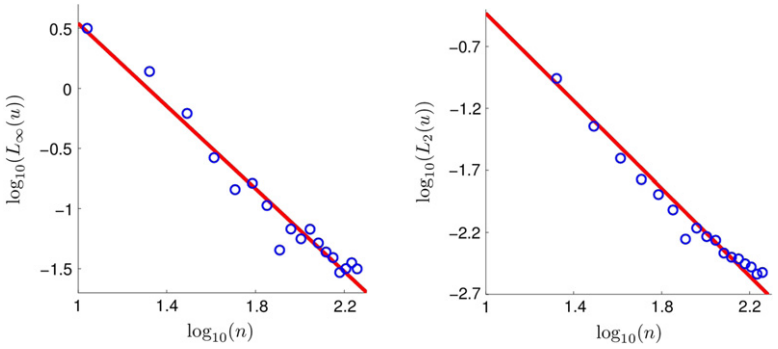

Fig. 19. L_∞ errors (left chart) and L_2 errors (right chart) for Case 7 ($n = n_x = n_y = n_z$).

- **Case 8(a):**

$$u^+(x, y, z) = 8, \tag{92}$$

$$u^-(x, y) = \begin{cases} x + y + z + 1 & x + y + z > 0 \\ \sin(x + y + z) + \cos(x + y + z) & x + y + z \le 0. \end{cases} \tag{93}$$

This solution is of C^1 continuous.

- **Case 8(b):**

$$u^+(x, y, z) = 8, \tag{94}$$

$$u^-(x, y, z) = (x + y + z)^{-5/6} + \sin(x + y + z). \tag{95}$$

This solution is of H^2 continuous.

The source term $g(x, y, z)$ and boundary value $g_b(x, y, z)$ can be given accordingly.

The final interface geometry used in our computation is obtained by tilting the subject around the origin $(0, 0, 0)$ with $\theta = \arccos(0.7)$ and $\phi = \arccos(0.4)$, where θ is the azimuth angle and ϕ is the zenith angle defined in Section 3.3.1. In Case 8a, the solution u^- is only C^1 continuous because the second order derivative is not continuous in plane $x + y + z = 0$. With Lipschitz continuous interface, this problem is difficult for collocation based methods. In Case 8b, the solution is H^2 continuous in Ω^- and at the original point, the second order derivative diverges. The present MIB Galerkin method works very well for these problems as shown in Tables 11 and 12. As in previous test cases, the second order accuracy is essentially obtained. Fig. 20 depicts the numerical solution and error on an $80 \times 80 \times 80$ mesh. Errors are very small for these test cases.

5. Conclusion

We present a three dimensional (3D) Galerkin formulation of the matched interface and boundary (MIB) method for the solution of elliptic partial differential equations (PDEs) with discontinuous coefficients, arising from modeling material interfaces in practical applications. This class of problems is also called elliptic interface problems. Much progress has been

Table 11
Numerical errors and convergence orders of MIB Galerkin method for C^1 continuous solutions (Case 8a).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order
$20 \times 20 \times 20$	2.020e−3		2.007e−4	
$40 \times 40 \times 40$	6.294e−4	1.68	6.219e−5	1.69
$80 \times 80 \times 80$	1.784e−4	1.82	1.637e−5	1.93
$160 \times 160 \times 160$	4.897e−5	1.87	4.096e−6	2.00

Table 12
Numerical errors and convergence orders of MIB Galerkin method for H^2 continuous solutions (Case 8b).

$n_x \times n_y \times n_z$	$L_\infty(u)$	Order	$L_2(u)$	Order
$20 \times 20 \times 20$	6.202e−3		3.140e−4	
$40 \times 40 \times 40$	1.929e−3	1.68	6.845e−5	2.20
$80 \times 80 \times 80$	5.874e−4	1.72	1.501e−5	2.19
$160 \times 160 \times 160$	1.773e−4	1.73	3.380e−6	2.15

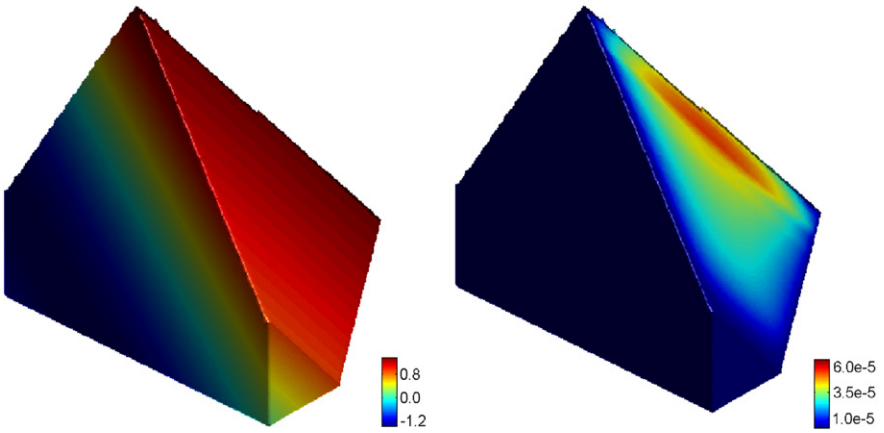

Fig. 20. The computed solution on an $80 \times 80 \times 80$ mesh (left chart) and the L_∞ error (right chart) for Case 8.

made in the development of advanced numerical methods for these problems in the past few decades. Indeed, it is fairly routine to construct 2D or 3D second order accurate numerical methods for solving elliptic interface problems at present. However, there are still many challenging issues that hinder the further progress in the field. Some of these challenges concern the development of higher order methods, namely, methods of convergence orders higher than those reported in the literature [33]. Another challenging issue in elliptic interface problems is the treatment of geometric singularities, i.e., nonsmooth interfaces, in high order schemes [33]. Many standing open problems regarding these issues were posed [33]. The MIB method has been shown to work well in dealing with the abovementioned challenges [33]. However, due to its collocation formulation, the original MIB method does not work well for low regularity solutions induced by or associated with geometric singularities, which commonly occur in practical problems, such as electromagnetic analysis. The recently introduced Wang–Ye Galerkin finite element method [22] has found its success in dealing with elliptic interface problems with low regularity solutions [23]. It has established essentially second order accuracy for six challenging cases proposed by Hou et al. [30]. Motivated by the success of Galerkin formulations, a 2D second order MIB Galerkin method using triangular elements has been proposed [83]. Indeed, the Galerkin formulation enables the MIB method to handle low regularity problems too.

The good performance of the 2D MIB Galerkin method indicates this method may be useful for practical applications involving 3D realistic interface geometries. The present work constructs a 3D MIB Galerkin method for arbitrarily complex interface geometries and problems with low regularity solutions. In the present 3D Galerkin MIB method, we combine the advantages of the Galerkin formulation with the flexibility of the MIB method to build up a robust approach for handling geometric singularities and complex interfaces. We utilize the Cartesian mesh based elements to avoid the time consuming mesh generation and mesh refinement. We employ two sets of overlapping elements, i.e., MIB elements, on extended domains to deal with material interfaces. Confirming elements with piecewise polynomial basis functions are used in both extended domains. Fictitious solutions are defined on the overlapping elements so that differentiation operators can be evaluated near the interface as if there were no discontinuous coefficients. The full set of interface jump conditions is rigorously enforced on the interface, which in turn determines the fictitious solutions. To demonstrate the robustness of the proposed MIB Galerkin strategy, we realize it with three different elements, namely, rectangular prism element,

five-tetrahedron element and six-tetrahedron element. The performance of these elements is examined and compared. It is found that the simplest element, i.e., the rectangular prism element, has the best performance. A possible reason for this behavior is that the interface jump conditions are more efficiently used in the prism element, as the intersection points are only found on mesh lines. Further work can be done to improve the accuracy of tetrahedron element schemes by a better design of the interface scheme. It is also found that although our final matrix is not symmetric or positive definite, the symmetric selection of the nodes in fictitious schemes maintains certain regularity. Thus common Krylov subspace based linear solvers can be directly employed.

To explore its accuracy order and demonstrate its ability of handling geometric singularities, the present 3D method is extensively tested on three classes of complex interface geometries. One class of geometries is generated by analytically expressed level set functions, including a sphere, four spheres, and four cylinders. Another class of geometries is the surfaces of proteins, which are arbitrarily complexes. The other class of geometries is multiprotein complexes, such as molecular motors. The proposed MIB Galerkin method, particularly when realized with the rectangular prism element, achieves near second order accuracy in the L_∞ norm for all of these problems.

Finally, we have noted that the present work advances the field of elliptic interface problems in two aspects. First, it is the first time that an FEM has essentially established its near second order accuracy for elliptic interface problems defined on complex protein surfaces, to our best knowledge. Previously, many other FEMs have been developed for solving the Poisson equation for the electrostatic potential of proteins, in the framework of implicit solvent models. However, it appears that no FEM has been directly tested for its accuracy on protein surfaces. The only other numerical method that has been validated and confirmed the second order accuracy for truly complex proteins surfaces was the MIB method in its collocation formulation [46,47]. Additionally, the present work delivers the first known second order accuracy for low regularity solutions induced by or associated with geometric singularities in 3D settings. We demonstrate that the proposed MIB Galerkin method essentially achieves the second order convergence in solving problems with C^1 and H^2 continuous solutions, associated with a 3D Lipschitz continuous interface.

Acknowledgments

This work was supported in part by NSF grants IIS-1302285 and DMS-1160352, and NIH grant R01GM-090208.

References

[1] B.E. Griffith, 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) 75–105.
[2] M.C. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.
[3] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (3) (1977) 220–252.
[4] A. Mayo, The fast solution of Poisson's and the biharmonic equations on irregular regions, SIAM J. Numer. Anal. 21 (1984) 285–299.
[5] A. McKenney, L. Greengard, A. Mayo, A fast Poisson solver for complex geometries, J. Comput. Phys. 118 (1995) 348–355.
[6] R.J. LeVeque, Z.L. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.
[7] Z.L. Li, K. Ito, Maximum principle preserving schemes for interface problems with discontinuous coefficients, SIAM J. Sci. Comput. 23 (2001) 339–361.
[8] L. Adams, Z.L. Li, The immersed interface/multigrid methods for interface problems, SIAM J. Sci. Comput. 24 (2002) 463–479.
[9] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (1999) 457–492.
[10] X.D. Liu, R.P. Fedkiw, M. Kang, A boundary condition capturing method for Poisson's equation on irregular domains, J. Comput. Phys. 160 (2000) 151–178.
[11] M. Oevermann, R. Klein, A Cartesian grid finite volume method for elliptic equations with variable coefficients and embedded interfaces, J. Comput. Phys. 219 (2006) 749–769.
[12] J.L. Hellrung Jr., L.M. Wang, E. Sifakis, J.M. Teran, A second order virtual node method for elliptic problems with interfaces and irregular domains in three dimensions, J. Comput. Phys. 231 (2012) 2015–2048.
[13] T. Chen, J. Strain, Piecewise-polynomial discretization and Krylov-accelerated multigrid for elliptic interface problems, J. Comput. Phys. 16 (2008) 7503–7542.
[14] J.T. Beale, A.T. Layton, On the accuracy of finite difference methods for elliptic problems with interfaces, Commun. Appl. Math. Comput. Sci. 1 (2006) 91–119.
[15] I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, Computing 5 (1970) 207–213.
[16] R.E. Ewing, Z.L. Li, T. Lin, Y.P. Lin, The immersed finite volume element methods for the elliptic interface problems, Math. Comput. Simul. 50 (1999) 63–76.
[17] I. Ramiere, Convergence analysis of the q_1-finite element method for elliptic problems with non-boundary-fitted meshes, Internat. J. Numer. Methods Engrg. 75 (2008) 1007–1052.
[18] X. He, T. Lin, Y. Lin, Interior penalty bilinear IFE discontinuous Galerkin methods for elliptic equations with discontinuous coefficient, J. Syst. Sci. Complex. 23 (2010) 467–483.
[19] Z. Cai, X. Ye, S. Zhang, Discontinuous Galerkin finite element methods for interface problems: a priori and a posteriori error estimations, SIAM J. Numer. Anal. 49 (2011) 1761–1787.
[20] S.M. Hou, P. Song, L.Q. Wang, H.K. Zhao, A weak formulation for solving elliptic interface problems without body fitted grid, J. Comput. Phys. 249 (2013) 80–95.
[21] M. Dryjaa, J. Galvisb, M. Sarkisb, BDDC methods for discontinuous Galerkin discretization of elliptic problems, J. Complexity 23 (2007) 715–739.
[22] J.P. Wang, X. Ye, A weak Galerkin finite element method for second-order elliptic problems, 2011. arXiv:1104.2897v1.
[23] L. Mu, J. Wang, G.W. Wei, X. Ye, S. Zhao, Weak Galerkin method for second order elliptic interface problems, J. Comput. Phys. 250 (2013) 106–125.
[24] J. Bramble, J. King, A finite element method for interface problems in domains with smooth boundaries and interfaces, Adv. Comput. Math. 6 (1996) 109–138.
[25] E. Burman, P. Hansbo, Interior-penalty-stabilized Lagrange multiplier methods for the finite-element solution of elliptic interface problems, IMA J. Numer. Anal. 30 (2010) 870–885.

[26] R. Hiptmair, J. Li, J. Zou, Convergence analysis of finite element methods for H(div; Omega)-elliptic interface problems, J. Numer. Math. 18 (2010) 187–218.
[27] D.A. Wang, R. Li, N.N. Yan, An edge-based anisotropic mesh refinement algorithm and its application to interface problems, Commun. Comput. Phys. 8 (2010) 511–540.
[28] X.S. Wang, L.T. Zhang, L.W. Kam, On computational issues of immersed finite element methods, J. Comput. Phys. 228 (2009) 2535–2551.
[29] A. Hansbo, P. Hansbo, An unfitted finite element method, Comput. Methods Appl. Mech. Engrg. 191 (2002) 5537–5552.
[30] S.M. Hou, W. Wang, L.Q. Wang, Numerical method for solving matrix coefficient elliptic equation with sharp-edged interfaces, J. Comput. Phys. 229 (2010) 7162–7179.
[31] I. Harari, J. Dolbow, Analysis of an efficient finite element method for embedded interface problems, Comput. Math. 46 (2010) 205–211.
[32] S.N. Yu, Y.C. Zhou, G.W. Wei, Matched interface and boundary (MIB) method for elliptic problems with sharp-edged interfaces, J. Comput. Phys. 224 (2) (2007) 729–756.
[33] S.N. Yu, G.W. Wei, Three-dimensional matched interface and boundary (MIB) method for treating geometric singularities, J. Comput. Phys. 227 (2007) 602–632.
[34] S. Zhao, G.W. Wei, High-order FDTD methods via derivative matching for Maxwell's equations with material interfaces, J. Comput. Phys. 200 (1) (2004) 60–103.
[35] Y.C. Zhou, S. Zhao, M. Feig, G.W. Wei, High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular sources, J. Comput. Phys. 213 (1) (2006) 1–30.
[36] Y.C. Zhou, G.W. Wei, On the fictitious-domain and interpolation formulations of the matched interface and boundary (MIB) method, J. Comput. Phys. 219 (1) (2006) 228–246.
[37] G.W. Wei, Discrete singular convolution for the solution of the Fokker–Planck equations, J. Chem. Phys. 110 (1999) 8930–8942.
[38] G.W. Wei, Y.Q. Jia, Synchronization-based image edge detection, Europhys. Lett. 59 (6) (2002) 814–819.
[39] D. Chen, G.W. Wei, Modeling and simulation of electronic structure, material interface and random doping in nano-electronic devices, J. Comput. Phys. 229 (2010) 4431–4460.
[40] D. Chen, W.W. Guo, Modeling and simulation of electronic structure, material interface and random doping in nano-electronic devices, J. Comput. Phys. 229 (2010) 4431–4460.
[41] S.N. Yu, Y. Xiang, G.W. Wei, Matched interface and boundary (MIB) method for the vibration analysis of plates, Commun. Numer. Methods Engrg. 25 (2009) 923–950.
[42] S. Zhao, High order matched interface and boundary methods for the Helmholtz equation in media with arbitrarily curved interfaces, J. Comput. Phys. 229 (2010) 3155–3170.
[43] S. Zhao, Full-vectorial matched interface and boundary (MIB) method for the modal analysis of dielectric waveguides, IEEE/OSA J. Lighwave Technol. 26 (2008) 2251–2259.
[44] Y.C. Zhou, J.G. Liu, D.L. Harry, A matched interface and boundary method for solving multi-flow Navier–Stokes equations with applications to geodynamics, J. Comput. Phys. 231 (2012) 223–242.
[45] Y.C. Zhou, M. Feig, G.W. Wei, Highly accurate biomolecular electrostatics in continuum dielectric environments, J. Comput. Chem. 29 (2008) 87–97.
[46] S.N. Yu, W.H. Geng, G.W. Wei, Treatment of geometric singularities in implicit solvent models, J. Chem. Phys. 126 (2007) 244108.
[47] W. Geng, S. Yu, G.W. Wei, Treatment of charge singularities in implicit solvent models, J. Chem. Phys. 127 (2007) 114106.
[48] D. Chen, Z. Chen, C. Chen, W.H. Geng, G.W. Wei, MIBPB: a software package for electrostatic analysis, J. Comput. Chem. 32 (2011) 657–670.
[49] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (1) (2000) 35–60.
[50] M. Francois, W. Shyy, Computations of drop dynamics with the immersed boundary method, part 2: drop impact and heat transfer, Numer. Heat Transfer B 44 (2003).
[51] M. Francois, E. Uzgoren, J. Jackson, W. Shyy, Multigrid computations with the immersed boundary technique for multiphase flows, Internat. J. Numer. Methods Heat Fluid Flow 14 (2004) 98–115.
[52] G. Iaccarino, R. Verzicco, Immersed boundary technique for turbulent flow simulations, Appl. Mech. Rev. 56 (2003) 331–347.
[53] M. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 236–261.
[54] L. Lee, R.J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (3) (2003) 832–856.
[55] A.T. Layton, Using integral equations and the immersed interface method to solve immersed boundary problems with stiff forces, Comput. & Fluids 38 (2009) 266–272.
[56] G.R. Hadley, High-accuracy finite-difference equations for dielectric waveguide analysis I: uniform regions and dielectric interfaces, J. Lightwave Technol. 20 (2002) 1210–1218.
[57] J.S. Hesthaven, High-order accurate methods in time-domain computational electromagnetics. A review, Adv. Imaging Electron Phys. 127 (2003) 59–123.
[58] R. Kafafy, T. Lin, Y. Lin, J. Wang, Three-dimensional immersed finite element methods for electric field simulation in composite materials, Internat. J. Numer. Methods Engrg. 64 (2005) 940–972.
[59] J.D. Kandilarov, Immersed interface method for a reaction–diffusion equation with a moving own concentrated source, in: NMA'02: Revised Papers from the 5th International Conference on Numerical Methods and Applications, Springer-Verlag, London, UK, 2003, pp. 506–513.
[60] T.P. Horikis, W.L. Kath, Modal analysis of circular Bragg fibers with arbitrary index profiles, Opt. Lett. 31 (2006) 3417–3419.
[61] T.Y. Hou, Z.L. Li, S. Osher, H.K. Zhao, A hybrid method for moving interface problems with application to the Hele-Shaw flow, J. Comput. Phys. 134 (2) (1997) 236–252.
[62] W.K. Liu, Y. Liu, D. Farrell, L. Zhang, X. Wang, Y. Fukui, N. Patankar, Y. Zhang, C. Bajaj, X. Chen, H. Hsu, Immersed finite element method and its applications to biological systems, Comput. Methods Appl. Mech. Engrg. 195 (2006) 1722–1749.
[63] H.B. Ameur, M. Burger, B. Hackl, Level set methods for geometric inverse problems in linear elasticity, Inverse Problems 20 (3) (2004) 673–696.
[64] P.A. Berthelsen, A decomposed immersed interface method for variable coefficient elliptic equations with non-smooth and discontinuous solutions, J. Comput. Phys. 197 (1) (2004) 364–386.
[65] G. Biros, L. Ying, D. Zorin, A fast solver for the Stokes equations with distributed forces in complex geometries, J. Comput. Phys. 193 (1) (2004) 317–348.
[66] M.A. Dumett, J.P. Keener, An immersed interface method for solving anisotropic elliptic boundary value problems in three dimensions, SIAM J. Sci. Comput. 25 (1) (2003) 348–367.
[67] A.L. Fogelson, J.P. Keener, Immersed interface methods for neumann and related problems in two and three dimensions, SIAM J. Sci. Comput. 22 (5) (2001) 1630–1654.
[68] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, J. Comput. Phys. 202 (2) (2005) 577–601.
[69] S. Hou, X.-D. Liu, A numerical method for solving variable coefficient elliptic equation with interfaces, J. Comput. Phys. 202 (2) (2005) 411–445.
[70] S. Jin, X. Wang, Robust numerical simulation of porosity evolution in chemical vapor infiltration: II. Two-dimensional anisotropic fronts, J. Comput. Phys. 179 (2) (2002) 557–577.
[71] H. Johansen, P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1) (1998) 60–85.
[72] M.N. Linnick, H.F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, J. Comput. Phys. 204 (1) (2005) 157–192.
[73] B. Lombard, J. Piraux, How to incorporate the spring-mass conditions in finite-difference schemes, SIAM J. Sci. Comput. 24 (4) (2003) 1379–1407.
[74] M. Schulz, G. Steinebach, Two-dimensional modelling of the river Rhine, J. Comput. Appl. Math. 145 (1) (2002) 11–20.

[75] J.A. Sethian, Evolution, implementation, and application of level set and fast marching methods for advancing fronts, J. Comput. Phys. 169 (2) (2001) 503–555.
[76] A.-K. Tornberg, B. Engquist, Numerical approximations of singular source terms in differential equations, J. Comput. Phys. 200 (2) (2004) 462–488.
[77] J.J. Vande Voorde, J. Vierendeels, E. Dick, Flow simulations in rotary volumetric pumps and compressors with the fictitious domain method, J. Comput. Appl. Math. 168 (1–2) (2004) 491–499.
[78] G. Morgenthal, J.H. Walther, An immersed interface method for the vortex-in-cell algorithm, Comput. Struct. 85 (11–14) (2007) 712–726.
[79] A. Wiegmann, K.P. Bube, The explicit-jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions, SIAM J. Numer. Anal. 37 (3) (2000) 827–862.
[80] W. Cai, S.Z. Deng, An upwinding embedded boundary method for Maxwell's equations in media with material interfaces: 2D case, J. Comput. Phys. 190 (2003) 159–183.
[81] K.L. Xia, M. Zhan, G.-W. Wei, The matched interface and boundary (MIB) method for multi-domain elliptic interface problems, J. Comput. Phys. 230 (2011) 8231–8258.
[82] K.L. Xia, M. Zhan, D.C. Wan, G.W. Wei, Adaptively deformed mesh based matched interface and boundary (MIB) method for elliptic interface problems, J. Comput. Phys. 231 (2012) 1440–1461.
[83] K.L. Xia, M. Zhan, G.W. Wei, MIB Galerkin method for elliptic interface problems, J. Comput. Appl. Math. 272 (2014) 195–220.
[84] W. Geng, G.W. Wei, Multiscale molecular dynamics using the matched interface and boundary method, J. Comput. Phys. 230 (2) (2011) 435–457.
[85] Q. Zheng, D. Chen, G.W. Wei, Second-order Poisson–Nernst–Planck solver for ion transport, J. Comput. Phys. 230 (2011) 5239–5262.
[86] Q.Y. Li, J.R. Zheng, G. Liao, Y. Jin, Approach on area coordinate, volume coordinate and their application in true 3DGIS, J. Earth Sci. Eng. 1 (1) (2011) 52–59.
[87] P.W. Bates, G.W. Wei, S. Zhao, Minimal molecular surfaces and their applications, J. Comput. Chem. 29 (3) (2008) 380–391.