Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 227 (2007) 602–632
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Three-dimensional matched interface and boundary (MIB) method for treating geometric singularities

Sining Yu [a], G.W. 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

Received 22 March 2007; received in revised form 23 July 2007; accepted 4 August 2007
Available online 15 August 2007

Abstract

This paper reports the three-dimensional (3D) generalization of our previous 2D higher-order matched interface and boundary (MIB) method for solving elliptic equations with discontinuous coefficients and non-smooth interfaces. New MIB algorithms that make use of two sets of interface jump conditions are proposed to remove the critical acute angle constraint of our earlier MIB scheme for treating interfaces with sharp geometric singularities, such as sharp edges, sharp wedges and sharp tips. The resulting 3D MIB schemes are of second-order accuracy for arbitrarily complex interfaces with sharp geometric singularities, of fourth-order accuracy for complex interfaces with moderate geometric singularities, and of sixth-order accuracy for curved smooth interfaces. A systematical procedure is introduced to make the MIB matrix optimally symmetric and banded by appropriately choosing auxiliary grid points. Consequently, the new MIB linear algebraic equations can be solved with fewer number of iterations. The proposed MIB method makes use of Cartesian grids, standard finite difference schemes, lowest order interface jump conditions and fictitious values. The interface jump conditions are enforced at each intersecting point of the interface and mesh lines to overcome the staircase phenomena in finite difference approximation. While a pair of fictitious values are determined along a mesh at a time, an iterative procedure is proposed to determine all the required fictitious values for higher-order schemes by repeatedly using the lowest order jump conditions. A variety of MIB techniques are developed to overcome geometric constraints. The essential strategy of the MIB method is to locally reduce a 2D or a 3D interface problem into 1D-like ones. The proposed MIB method is extensively validated in terms of the order of accuracy, the speed of convergence, the number of iterations and CPU time. Numerical experiments are carried out to complex interfaces, including the molecular surfaces of a protein, a missile interface, and van der Waals surfaces of intersecting spheres.

Keywords: Elliptic equations; Discontinuous coefficients; Material interface; Geometric singularity; Immersed boundary method; Immersed interface method

* Corresponding author. Address: 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).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2007.08.003

1. Introduction

The numerical solution of elliptic equations with discontinuous coefficients and singular sources has attracted much attention in the applied mathematics and scientific computing [5–7,13,16,21,26–29,31,-32,34,36,41–43,46,61–63,65,66,68] since Peskin's pioneer work in 1977 [57]. This class of problems have demonstrated a growing impact to fluid dynamics [14,18,19,30,53], electromagnetics [23,24,33], material science [25,27] and biologic systems [20,44,70,75]. Due to the lack of regularity in the solution, ordinary numerical schemes, such as finite difference methods, finite element methods and spectral methods, cannot maintain the designed order of convergence and even diverge in the worst scenario. It is very difficult to construct highly accurate and efficient numerical methods for this class of problems. Apart from Peskin's immersed boundary method (IBM) [22,35,57–59], a number of other elegant methods have been proposed in the literature. Among them, the ghost fluid method (GFM) proposed by Fedkiw, Osher and coworkers [15,45] is a relatively simple and easy to use sharp interface approach. Finite element formulations [2,8,40,44] are more suitable for weak solutions. An upwinding embedded boundary method was proposed by Cai and Deng [9] for dielectric media. A second-order finite volume based method was proposed Oevermann and Klein [54], which uses bilinear ansatz functions on Cartesian grids and compact stencils. A relevant, while quite distinct approach is the integral equation method for embedding complex geometry [49–51]. A major advance in the field was due to LeVeque and Li [37], who proposed a remarkable second-order sharp interface scheme, the immersed interface method (IIM). Their method has been improved for computational efficiency and matrix optimization over the past decade [1,12,38,39,62] and widely applied in practical problems. A new trend in this field is the construction of higher-order (interface) methods [21,50,63,71,73,74] that are particularly desirable for problems involving both material interfaces and high frequency waves where conventional local adaptive refinement approaches do not work well. Typical examples are the interaction of turbulence and shock, and high frequency wave propagation in inhomogeneous media. Simple Cartesian grids are preferred in these situations because of the bypass of the mesh generation, better temporal stability and the availability of fast algebraic solvers. Recently, we have proposed a systematic higher-order method, the matched interface and boundary (MIB) method, for treating electromagnetic wave propagation and scattering in dielectric media [71]. In particular, we proposed a way to avoid the use of high order jump conditions, which require large stencils and are numerically unstable in higher-order schemes as analyzed in our work. We found that by repeatedly using only the lowest order jump conditions, an arbitrarily higher-order interface scheme can be constructed in principle [71]. This idea was applied to the fourth-order beam equation with free boundary conditions [72]. More recently, we have generalized the earlier MIB method for solving elliptic equations with curved interfaces [73,74]. The MIB method makes use of simple Cartesian grids, standard finite difference schemes, lowest order physical jump conditions and fictitious domains. The physical jump conditions are enforced at each intersecting point of the interface and the mesh lines. Because of the use of the subgrid information at the intersecting points, the MIB method overcomes the staircase phenomena of the finite difference approximation for complex geometry and can be of arbitrarily higher-order convergence in principle when the lowest order physical jump conditions are repeatedly enforced. MIB schemes of up to 16th-order accuracy have been constructed for straight interfaces [71,73]. For curved interfaces, fourth- and sixth-order MIB schemes have been demonstrated [73,74]. Most recently, we have proposed an interpolation formulation of the MIB method that does not need the use of the fictitious values [74]. We have shown that our interpolation formulation of the MIB method is equivalent to our earlier fictitious domain formulation [74]. We have constructed a three-dimensional (3D) MIB scheme that is of second-order convergence for complex interfaces free of geometric singularities [75].

While much attention was focused on elliptic equations with smooth interfaces, little work has been done on the mathematical treatment of geometric singularities, such as sharp-edged, sharp-wedged and sharp-tipped interfaces. This class of problems are ubiquitous in science and engineering, such as wave-guides analysis [52], electromagnetic wave scattering and propagation [10,55,56], friction modeling [64], plasma-surface interaction [47] and turbulent-flow [4]. The presence of geometric singularities, such as cusps and self-intersecting surfaces, is a major obstacle to the accuracy, convergence and stability of the numerical solution of the Poisson–Boltzmann equation for the electrostatic analysis of biomolecules [70]. Technically, it is much more challenging to construct efficient numerical methods for this class of problems due to the lack of regularity in

the interface. Because the gradient near the tip of a sharp-edged interface is not well defined in the classical sense, the aforementioned interface methods might not maintain the designed accuracy. A standard approach to these problems is the use of local adaptive approaches [52,56], which exhibit reduced convergence rates and require dramatic mesh refinements in the vicinity of geometric singularities [11]. These approaches fail to work if the solution is highly oscillatory due to the so called pollution effect [3]. A remarkable result, due to Hou and Liu [26], was of 0.8th-order convergence obtained with a 2D finite element formulation for elliptic interface problems with geometric singularities. Their method is convenient for geometric singularity induced weak solutions. Recently, we have developed a 2D second-order MIB treatment of sharp-edged interfaces [69]. Unlike the Galerkin formulation, our MIB scheme does not specifically handle the geometric singularity induced weak solutions. A standard approach that multiplies the solution with an appropriate polynomial [56] was incorporated in our MIB method to deal with geometry induced weak solutions [69]. The MIB is found to be insensitive to the location of tips. The domain extension on the tip can be computed from any direction by extrapolation without affecting the result [69]. An explanation is that although gradients on the tip extrapolated from different directions contradict each other, different sets of interface conditions from different directions near the tip are consistent with each other for a given domain extension.

Another challenge that is encountered in interface schemes, particularly in higher-order ones, is the asymmetric discretization matrices which render convergence problems in solving the resulting linear algebraic equations. This difficulty is deteriorated by complex interface geometry—the matrices usually become more asymmetric under complex geometric constraints. Moreover, the higher convergence order, the larger matrix bandwidth will be for an interface scheme. Furthermore, discretization matrices are usually not diagonally dominant due to the interface treatment. This causes slow convergence as well. A discrete maximum preserving scheme was proposed to accelerate the convergence of the IIM [12]. In our earlier MIB method, a large number of iterations is required to solve the linear algebraic equations due to the asymmetric matrix.

The objective of the present work is threefold. First, we extend our earlier 2D higher-order MIB schemes [71,73] to 3D. We report some first known sixth-order 3D results for elliptic problems with smooth interfaces. Such schemes are particularly efficient for problems involving both material interfaces and high frequency waves. Second, we generalize our earlier 2D second-order MIB method for geometric singularities [69] to 3D. Our earlier 3D MIB scheme does not maintain the designed second accuracy for molecular surfaces of proteins due to the presence of non-smooth interfaces [75]. In this work, we demonstrate the second-order convergence for arbitrarily complex interfaces with arbitrary geometric singularities. In fact, there is a critical acute angle restriction in our earlier 2D MIB scheme for sharp-edged interfaces [69]. A new algorithm, which makes use of two interface intersecting points for sharp-edged interfaces, is proposed to reserve this problem. We demonstrate that 3D fourth-order MIB schemes can be constructed for complex interfaces and interfaces with moderate geometric singularities. Finally, due to the constraint of complex interface geometry, it is impossible to create a high order MIB scheme with a symmetric discretization matrix. In this work, we have addressed this problem by making the MIB matrix optimally symmetric and banded. Here, by 'optimally symmetric and banded' we mean as symmetric and banded as possible under a given geometric constraint. The matrix optimization is achieved by an appropriate selection of auxiliary grid points. This optimization procedure dramatically improves the convergence property of the present MIB method. An essential feature of the proposed MIB method is that it locally simplifies a 2D or a 3D interface problem into 1D-like ones. As a result, the MIB method is able to systematically achieve higher-order convergence and to efficiently deal with geometric singularities. The proposed MIB algorithm, particularly its matrix acceleration and treatment of singularities, provides the crucial technical foundation to a new generation of Poisson–Boltzmann equation solvers for the electrostatic analysis of biomolecules [70].

In the next two sections, the theoretical formulation and the computational algorithm are given to the 3D MIB method for elliptic interface problems. Detailed consideration of topological variations is given, particularly for 3D geometric singularities. New algorithm is proposed for the sharp-edged interfaces and tips. The MIB matrix is made optimally symmetric and banded. A systematic procedure is introduced to construct 3D higher-order MIB schemes. The proposed MIB method is extensively validated by benchmark tests, such as the interfaces of missile, protein, flower and intersecting spheres. This paper ends with a conclusion to summarize the main points.

2. Theory and algorithm

2.1. General setting of the MIB method for elliptic interface problems

Consider a given interface Γ which divides an open bounded domain $\Omega \in \mathbb{R}^3$ into disjoint open subdomains, domain Ω^- and domain Ω^+, i.e., $\Omega = \Omega^- \cup \Omega^+ \cup \Gamma$. We solve the following 3D elliptic equation

$$\nabla \cdot (\beta(x,y,z)\nabla u(x,y,z)) - \kappa(x,y,z)u(x,y,z) = q(x,y,z), \quad x,y,z \in \Omega \tag{1}$$

with variable coefficients $\beta(x,y,z)$ and $\kappa(x,y,z)$ which may admit jumps at the interface Γ. To make the problem well-posed, this equation should be solved with appropriate boundary conditions and two interface jump conditions

$$[u] = u^+ - u^-, \tag{2}$$

$$[\beta u_n] = \beta^+ \nabla u^+ \cdot \mathbf{n} - \beta^- \nabla u^- \cdot \mathbf{n}, \tag{3}$$

where $\mathbf{n} = (n_x, n_y, n_z)$ is the normal direction of the interface. To achieve higher-order convergence, the interface jump conditions are to be rigorously enforced at each intersecting point of the interface and the Cartesian mesh lines, which means the incorporation of subgrid information for overcoming the staircase phenomena. In the MIB approach, we seek to reduce the 3D interface problem into 1D-like ones locally so that the standard higher-order finite difference schemes can be implemented for the discretization.

As the interface normal direction varies on the interface, it is necessary to define a set of local coordinate at each intersecting point of the interface and the Cartesian mesh. At a specific intersecting point, it is convenient to denote the local coordinates (ξ, η, ζ) such that ξ is along the normal direction and η is in the x–y plane. The coordinate transformation can be given as

$$\begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \mathbf{p} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{4}$$

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

$$\mathbf{p} = \begin{bmatrix} \sin\phi\cos\theta & \sin\phi\sin\theta & \cos\phi \\ -\sin\theta & \cos\theta & 0 \\ -\cos\phi\cos\theta & -\cos\phi\sin\theta & \sin\phi \end{bmatrix}. \tag{5}$$

Here θ and ϕ are the azimuth and zenith angles with respect to the normal direction $\mathbf{n}$, respectively. In the new coordinates, the jump condition (3) can be discretized as

$$[\beta u_\xi] = \beta^+(\sin\phi\cos\theta u_x^+ + \sin\phi\sin\theta u_y^+ + \cos\phi u_z^+) - \beta^-(\sin\phi\cos\theta u_x^- + \sin\phi\sin\theta u_y^- + \cos\phi u_z^-). \tag{6}$$

Without creating higher-order jump conditions, we can generate two additional jump conditions by differentiating Eq. (2) with respect to the tangential direction η and the binormal direction ζ

$$[u_\eta] = (-\sin\theta u_x^+ + \cos\theta u_y^+) - (-\sin\theta u_x^- + \cos\theta u_y^-), \tag{7}$$

$$[u_\zeta] = (-\cos\phi\cos\theta u_x^+ - \cos\phi\sin\theta u_y^+ + \sin\phi u_z^+) - (-\cos\phi\cos\theta u_x^- - \cos\phi\sin\theta u_y^- + \sin\phi u_z^-). \tag{8}$$

In principle, we can generate more jump conditions by further differentiating these low order jump conditions. However, a byproduct of such a procedure is the creation of higher-order derivatives and cross-derivatives, whose evaluation often involves larger stencils and is unstable for constructing higher-order interface schemes [71]. The proposed MIB method makes use of only lower-order jump conditions. Therefore, we end up with four interface jump conditions, i.e., Eqs. (2), (6), (7) and (8), which can be used to determine four desired unknown quantities related to the intersecting point of the interface and the mesh. In the MIB method, we extend the computational domains with fictitious values on both sides of the interface so that the standard finite difference scheme can be applied on a smooth domain near the interface without the loss of designed convergence. This is done along a mesh line at a time, so that our MIB method is locally a 1D-like scheme for a

higher dimensional interface. Fictitious values can be determined by the aforementioned four jump conditions. However, in association with four jump conditions there are six partial derivatives, $u_x^+, u_x^-, u_y^+, u_y^-, u_z^+$, and u_z^-, which have to be computed right on the intersecting point between the interface and the meshline. Numerically, these computations are carried out within appropriate subdomains near the interface, i.e., u_x^+, u_y^+ and u_z^+ in Ω^+, and u_x^-, u_y^- and u_z^- in Ω^-. For slightly curved smooth interfaces, it is usually quite easy to calculate all the six partial derivatives. However, for non-smooth and/or arbitrarily complex interfaces, it is often very difficult to compute all of these partial derivatives due to geometric constrains. To make the present MIB scheme applicable to real world problems, we eliminate two partial derivatives that are the most difficult to compute by using two appropriate jump conditions. The selection of two eliminated partial derivatives is pursued as follows.

1. All fictitious values on all irregular points will be determined. At an intersecting point between the interface and an x-mesh line, two fictitious values on the x-mesh line will be determined.
2. Two derivatives along this mesh line must be kept. In the above case, u_x^+ and u_x^- must be kept.
3. We therefore select two partial derivatives from u_y^+, u_z^+, u_y^- and u_z^-. We first check if the interface local geometry allows us to easily compute u_y^+ and u_z^+ in Ω^+. If so, we will eliminate u_y^- and u_z^-—it is done. If one of u_y^+ and u_z^+ cannot be computed, say u_z^+, we will check whether u_z^- can be compute in Ω^-. If yes, it is done. If u_z^- cannot be computed either, we will check whether u_y^- can be computed. If yes, it is done. If no, the scheme fails at this intersecting point. We will try to determine the nearby fictitious values from other nearby intersecting points.

The mathematical operation of the elimination is derived below. From Eq. (4), we have

$$\begin{bmatrix} u_\xi \\ u_\eta \\ u_\zeta \end{bmatrix} = \mathbf{p} \cdot \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}. \tag{9}$$

Therefore, Eqs. (6)–(8) can be rewritten as follows:

$$\begin{bmatrix} [\beta u_\xi] \\ [u_\eta] \\ [u_\zeta] \end{bmatrix} = \mathbf{C} \cdot \begin{bmatrix} u_x^+ \\ u_x^- \\ u_y^+ \\ u_y^- \\ u_z^+ \\ u_z^- \end{bmatrix}, \tag{10}$$

where

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \\ \mathbf{C}_3 \end{bmatrix} = \begin{bmatrix} p_{11}\beta^+ & -p_{11}\beta^- & p_{12}\beta^+ & -p_{12}\beta^- & p_{13}\beta^+ & -p_{13}\beta^- \\ p_{21} & -p_{21} & p_{22} & -p_{22} & p_{23} & -p_{23} \\ p_{31} & -p_{31} & p_{32} & -p_{32} & p_{33} & -p_{33} \end{bmatrix}. \tag{11}$$

Here p_{ij} is the ijth component of the transformation matrix $\mathbf{p}$ and $\mathbf{C}_i$ represents the ith row of $\mathbf{C}$. After the elimination of the lth and mth elements of the array $(u_x^+, u_x^-, u_y^+, u_y^-, u_z^+, u_z^-)$, Eq. (10) becomes

$$a[\beta u_\xi] + b[u_\eta] + c[u_\zeta] = (a\mathbf{C}_1 + b\mathbf{C}_2 + c\mathbf{C}_3) \cdot \begin{bmatrix} u_x^+ \\ u_x^- \\ u_y^+ \\ u_y^- \\ u_z^+ \\ u_z^- \end{bmatrix}, \tag{12}$$

where

$$\begin{aligned} a &= C_{2l}C_{3m} - C_{3l}C_{2m}, \\ b &= C_{3l}C_{1m} - C_{1l}C_{3m}, \\ c &= C_{1l}C_{2m} - C_{2l}C_{1m}. \end{aligned} \tag{13}$$

Therefore, two remaining jump conditions, Eqs. (2) and (12), are to be used to determine two fictitious values near the interface along a specific mesh line at a time. To construct higher-order interface schemes, the above procedure can be systematically repeated to determine fictitious values along other two mesh lines and at other interface locations. A more explicit and alternative description of this elimination procedure can be found in Ref. [75].

Alternatively, for very simple geometry, one may determine three or four fictitious values at a time by using three or four jump conditions. However, such a method will have to approximate more partial derivatives and involve more grid values. Consequently, the resulting matrix will be less symmetric and more iterations will be required to solve the linear algebraic equation.

In the present work, we propose a novel algorithm to deal with sharp-edged interfaces. Specifically, when there is a very narrow interface or a thin region enclosed by two interfaces, two sets of interface jump conditions at two intersecting points will be used to determine a few fictitious values together, see Section 2.5. This algorithm is developed to remove the acute angle constraint in our earlier 2D MIB method [69]. It gives us flexibility to handle arbitrarily complex geometry and geometric singularities. In summary, we determine fictitious values along a mesh line at time, and locally reduce a 3D interface problem into 1D-like ones.

2.2. Treatment of smooth interfaces—an illustration

Considering a geometry illustrated in Fig. 1, the interface intersects the kth y-mesh line on the y–z plane at point (x_0, y_0, z_0). Two fictitious values $f_{i,j,k}$ and $f_{i,j+1,k}$ are to be determined on irregular grid points. We denote the left domain as Ω^+ and the right one as Ω^-. Here, u^+, u^-, u_y^+ and u_y^- are easily expressed as interpolations and standard finite difference (FD) schemes from information in Ω^- and Ω^+, respectively

$$\begin{aligned} u^+ &= (w_{0,j-1}, w_{0,j}, w_{0,j+1}) \cdot (u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}}, \\ u^- &= (w_{0,j}, w_{0,j+1}, w_{0,j+2}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k})^{\mathrm{T}}, \\ u_y^+ &= (w_{1,j-1}, w_{1,j}, w_{1,j+1}) \cdot (u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}}, \\ u_y^- &= (w_{1,j}, w_{1,j+1}, w_{1,j+2}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k})^{\mathrm{T}}, \end{aligned} \tag{14}$$

where $w_{n,m}$ denote FD weights, which are generated by using the standard Lagrange polynomials [17]. The first subscript n represents either the interpolation ($n = 0$) or the first order derivative ($n = 1$) at interface point (x_o, y_o, z_o), while their second subscript is for the node index. We only need to compute two of the remaining

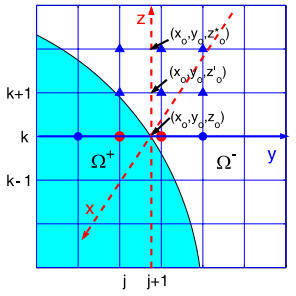

Fig. 1. Illustration of a smooth interface. The kth y mesh line intersects the interface at point (x_o, y_o, z_o). A pair of irregular points (i,j,k) and $(i,j+1,k)$, in red ●, is on the kth y-mesh line. Type I auxiliary points are in blue ▲. Type II auxiliary points are in blue ●. Auxiliary lines, in red dashed lines along the x- and z-directions, are used to calculate u_x^-, u_x^+, u_z^- and u_z^+. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

four interface quantities. If u_x^- and u_z^- can be conveniently computed, then u_x^+ and u_z^+ will be eliminated by using Eqs. (12) and (13) with $l = 1$ and $m = 5$.

To approximate u_z^+ or u_z^-, we need three u values along the auxiliary line $y = y_o$ on the y–z plane, see Fig. 1. Unfortunately, these values are unavailable on the grid and have to be approximated by interpolation schemes along the y-direction. This means six more auxiliary points are involved. In practice, one of these two derivatives can be easily found. In the situation shown in Fig. 1, u_z^- can be represented as follows:

$$u_z^- = [w_{1,k}, w_{1,k+1}, w_{1,k+2}] \cdot \begin{bmatrix} w_{0,j} & w_{0,j+1} & w_{0,j+2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w'_{0,j} & w'_{0,j+1} & w'_{0,j+2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w^*_{0,j} & w^*_{0,j+1} & w^*_{0,j+2} \end{bmatrix} \times [f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j,k+1}, u_{i,j+1,k+1}, u_{i,j+2,k+1}, u_{i,j,k+2}, u_{i,j+1,k+2}, u_{i,j+2,k+2}]^{\mathrm{T}}, \tag{15}$$

where we use superscripts on w to denote different sets of FD weights [17]. Similarly, u_x^- can be computed along the auxiliary line $y = y_o$ on the x–y plane, the dashed line shown in Fig. 1. By choosing $l = 1$ and $m = 5$ in Eq. (13), we can eliminate u_x^+ and u_z^+. Then, by solving Eqs. (2) and (12) together, two fictitious values $f_{i,j,k}$ and $f_{i,j+1,k}$ can be easily represented in terms of 16 function values around them.

Note that the calculation of fictitious values strongly depends on whether auxiliary points can be found in proper subdomains. At the second-order accuracy, this calculation is quite easy for smooth interfaces and relatively dense meshes. However, it is a challenge to determine all the fictitious values for higher-order schemes and with the presence of geometric singularities. In order to discuss this issue clearly in the following sections, we define type I auxiliary points as the auxiliary points that are only used to interpolate u values on auxiliary lines passing through the intersecting point. Type II auxiliary points are defined as the auxiliary points that are in the same line with two irregular points. For instance, in the situation depicted in Fig. 1, there are six type I auxiliary points used to interpolate u value at (x_o, y_o, z_o^*) and (x_o, y_o, z'_o), which are used to calculate u_z^-. Another six type I auxiliary points are used to calculate u_x^- in a similar way. Here, $(i,j-1,k)$ and $(i,j+2,k)$ are two type II auxiliary points.

2.3. Matrix optimization

In the calculation of each pair of fictitious values, 12 type I auxiliary points are involved. In most situations, the locations of type I auxiliary points are not unique. The utility of this non-uniqueness was not explored in our previous MIB schemes. In fact, the selection of type I auxiliary points has a substantial impact to the convergence property of the MIB matrix. Essentially, it is important to make the matrix optimally symmetric and banded so as to accelerate the speed of the convergence of linear algebraic solvers. However, in practice, it is not an easy task to choose type I auxiliary points because of the complex geometric constraints. In our earlier MIB method, denoted MIB I, our consideration was dictated by geometric complexities and little attention was paid to the matrix optimization. Since the locations of type I auxiliary points are very flexible, a systematical strategy is introduced to select optimal type I auxiliary points so as to maximize the speed of matrix convergence and minimize the numerical error of the interface scheme. This new approach, denoted as MIB II, is described as follows.

A typical interface topology is depicted in Fig. 2. In this case, 6 type I auxiliary points are required to calculate u_z^+ or u_z^-. Here u_z^+ requires 6 type I auxiliary points all from Ω^+ while u_z^- requires 6 type I auxiliary points all from Ω^-. Since normally only one of these two derivatives are computed, one could select either Ω^+ or Ω^-.

The next step is to decide which two mesh lines should be used to accommodate for these 6 type I auxiliary points. The present MIB II method prefers using mesh lines z_{k-1} and z_{k+1} together, while the earlier MIB I method typically does not choose this kind of combinations. The strategic difference of MIB I and MIB II in this step is indicated in Fig. 2. In the present work, the essential idea is to select type I auxiliary points as close to the intersecting point as possible. The CPU time can be significantly reduced in MIB II by using the new strategy, especially when the MIB method are applied to large computational domains and/or complicated interfaces.

In order to optimize the proposed MIB II method for accelerating the matrix convergence and minimizing numerical errors, the selection of type I auxiliary points is separated into three steps:

- Step 1: Select a subdomain, Ω^+ or Ω^-. In solving the Poisson–Boltzmann equation, there is a tendency to select the out subdomain such that the interference with the charge singularity can be avoided. In general, an appropriate subdomain is selected to optimize the matrix property.

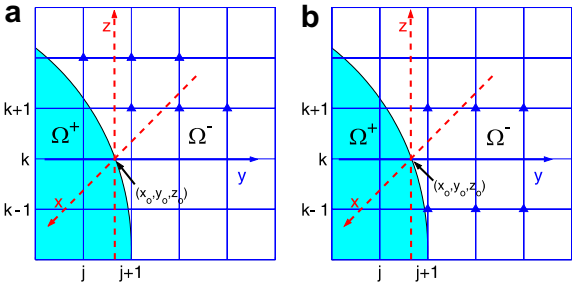

Fig. 2. (a) Type I auxiliary points ▲ typically selected in MIB I; (b) type I auxiliary points ▲ typically selected in MIB II.

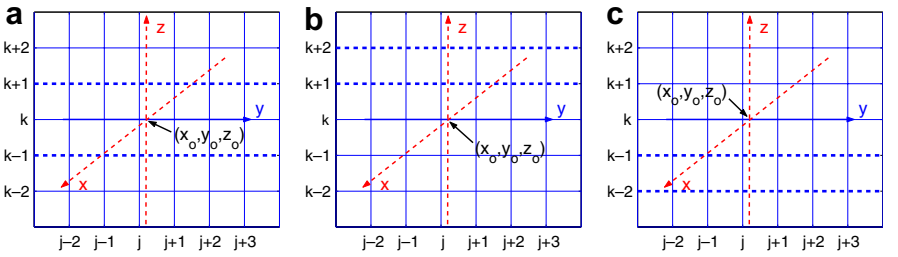

Fig. 3. Three different sets of mesh lines from which type I auxiliary points are selected. Set (a): mesh lines z_{k-1} and z_{k+1}; set (b): mesh lines z_{k+1} and z_{k+2}; set (c): mesh lines z_{k-1} and z_{k-2}.

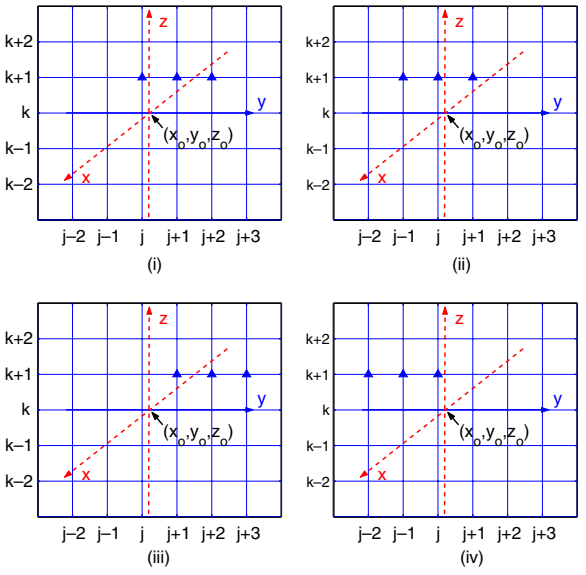

Fig. 4. Four sets of type I auxiliary points on a single mesh line z_{k+1}.

- Step 2: Select a set of mesh lines, i.e., a pair of mesh lines, as shown in Fig. 3a–c. In normal cases, the scheme in Fig. 3a is preferred in our MIB II methods.
- Step 3: Select two sets of grid points from 4 different options shown in Fig. 4i–iv. Note that an option can be used twice. In normal cases, choice of 4i or ii, or their combination is preferred. However, the selected grid points must be within the same selected subdomain.

The process of selecting type I auxiliary points is described in details in the following pseudocode:

A pseudocode for the selection of Type I auxiliary points

```
1: SD = Ω+;                    ▷ Select a subdomain (SD) to which all 6 points should belong
2: MLS = (a);                  ▷ Select a preferred mesh line set MLS
3: Search for 3 points in SD on each mesh line in MLS, following the order of (i), (ii), (iii), and (iv).
4: if all 6 points are found on these two mesh lines then
5:     Return;
6: else
7:     MLS = (b);
8:     Search for 3 points in the SD on each mesh line in the MLS, following the same order described
       above.
9:     if all 6 points are found on these two mesh lines then
10:        Return;
11:    else
12:       MLS = (c);
13:       Search for 3 points in the SD on each mesh line in the MLS, following the same order described
          above.
14:       if all 6 points are found on these two mesh lines then
15:          Return;
16:       else
17:          Goto 1, choose SD = Ω− and repeat the rest;
18:       end if
19:    end if
20: end if
```

2.4. Type I auxiliary points

Indeed, the proposed selection strategy of type I auxiliary points not only leads to optimally symmetric MIB matrices, but also introduces additional flexibility to the MIB method. For instance, in the situation depicted in Fig. 5a, type I auxiliary points can only be $(i,j-2,k-1)$, $(i,j-1,k-1)$, $(i,j,k-1)$ on mesh line $z=z_{k-1}$ and $(i,j-2,k+1)$, $(i,j-1,k+1)$, $(i,j,k+1)$ on mesh line $z=z_{k+1}$. Therefore, the present MIB II allows one to use mesh lines from both upper and lower quadrants, i.e., $z=z_{k-1}$ and $z=z_{k+1}$ in this situation, simultaneously.

In dealing with complex interface geometry, there are a few cases in which type I auxiliary points cannot all be found by following the aforementioned schemes. For example, in the situation depicted in Fig. 5b, type I auxiliary points can only be $(i,j-1,k-1)$, $(i,j+1,k-1)$, $(i,j+2,k-1)$ on mesh line $z=z_{k-1}$, and $(i,j-2,k+1)$, $(i,j-1,k+1)$ and $(i,j,k+1)$ on mesh line $z=z_{k+1}$. Similarly, there is another kind of cases in which type I auxiliary points cannot be found in consecutive mesh lines. One may need to use $z=z_{k-1}$ and $z=z_{k+2}$ together for the calculation of u_z^- or u_z^+ instead of mesh lines $z=z_{k-1}$ and $z=z_{k+1}$. To resolve these two kinds of difficulties, we first check whether primary fictitious values can be computed from other directions or other intersecting points. If this is true, then we just compute the primary fictitious values from another direction. If it is not true, then a new scheme is introduced to allow the use of non-consecutive type I auxiliary points as in Fig. 5b or non-consecutive z-mesh lines as in Fig. 5c. Note that when non-consecutive type I auxiliary points or non-consecutive z-mesh lines are used in interpolations, the matrix becomes less

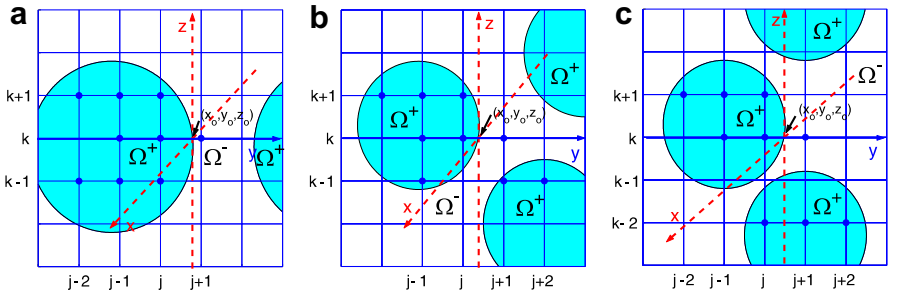

Fig. 5. Type I auxiliary points and other three points on mesh line z_k, all in blue ●, are used together to derive u_z^+. *Situation* (a): Type I auxiliary points locate on mesh lines z_{k-1} and z_{k+1}; *Situation* (b): Type I auxiliary points locate separately on mesh line z_{k-1}; *Situation* (c): Type I auxiliary points locate on nonconsecutive mesh lines z_{k-2} and z_{k+1}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

symmetric and the accuracy of the interpolations would be slightly reduced. However, our numerical tests indicate that the accuracy of the fictitious values solved by using the new scheme is still comparable to that of other fictitious values solved by our previous schemes. An explanation is that the interpolation by type I auxiliary points only plays a minor role in the whole calculation of fictitious values. In addition, the situation depicted in Fig. 5b and c often disappears when the mesh is refined. Nonetheless, schemes similar to those in Fig. 5b and c should be avoided whenever possible.

2.5. Type II auxiliary points

Unlike type I auxiliary points which are only used in Eq. (12) and have many options as described above, type II auxiliary points are directly involved in all the jump condition equations and have unique locations once the corresponding irregular points are selected. Furthermore, in the above scheme for smooth interfaces, type II auxiliary points and the irregular points next to them must be in the same subdomain. For instance, to find the fictitious values at irregular points (i,j,k) and $(i,j+1,k)$, auxiliary point $(i,j-1,k)$ and irregular point (i,j,k) must be in the same subdomain, and similarly, auxiliary point $(i,j+2,k)$ and irregular point $(i,j+1,k)$ must be in the same subdomain too.

However, when the interface has large curvatures, sharp edges, sharp wedges, tips or corners, there are 'improper' regions [69] where the required auxiliary points cannot be found in the same subdomain. For example, as depicted in Fig. 6, irregular point (i,j,k) has no auxiliary point associated with it inside the same subdomain. This 'improper' region cannot be eliminated by refining the mesh. The same situation occurs in the

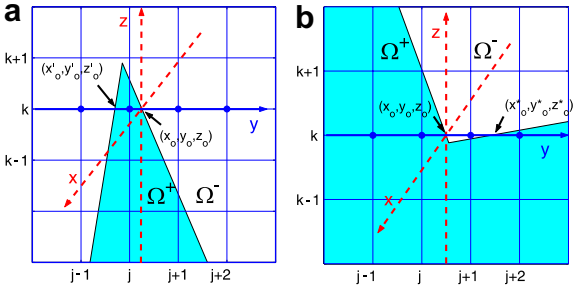

Fig. 6. Interface singularities and type II auxiliary points. Type II auxiliary points and two irregular points, all in blue ●, are used in all discretization of jump conditions about interface point (x_o,y_o,z_o) and an additional jump condition about the second interface point. *Situation* (a): Type II auxiliary point $(i,j-1,k)$ and irregular point (i,j,k) are in different subdomains, which creates a second interface point (x'_o,y'_o,z'_o); *Situation* (b): Type II auxiliary point $(i,j+2,k)$ and irregular point $(i,j+1,k)$ are in different subdomains, which creates a second interface point (x^*_o,y^*_o,z^*_o). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

neighboring mesh lines if the mesh is doubled. The auxiliary points in the 'improper' region lead to inconsistence between the number of unknowns and the number of jump conditions.

The MIB schemes for sharp-edged interfaces [69] were developed to deal with this situation. In Ref. [69], the secondary fictitious points were introduced to replace the role of type II auxiliary points in the 'improper' region. Secondary fictitious values are computed by appropriate interpolations and are used as auxiliary points to determine primary fictitious values. Consequently, the number of unknowns can be reduced, and the resulting unknowns can be solved successfully. This MIB scheme gives satisfactory results and has been used in the present work for geometric singularities. However, it has a critical acute angle limitation because the scheme depends on a priori calculation of secondary fictitious values. When edges become very sharp, it is likely that no secondary fictitious value can be computed around the primary fictitious point. Here, we present a new scheme that deals with sharp-edged problems by using an additional interface condition to solve primary fictitious value directly. This new scheme therefore removes the critical acute angle limitation in our earlier MIB method [69].

2.5.1. Second-order scheme

Consider the situation depicted in Fig. 6a, type II auxiliary point $(i,j-1,k)$ and irregular point (i,j,k) are not on the same subdomain. Fictitious values cannot be calculated directly from Eqs. (2) and (12). Assuming that point $(i,j-1,k)$ is in Ω^- with secondary fictitious value $f_{i,j-1,k}$. Then, Eq. (2) can be discretized with primary fictitious values, $f_{i,j,k}$, $f_{i,j+1,k}$ and secondary fictitious value $f_{i,j-1,k}$

$$[u] = (w_{0,j-1}, w_{0,j}, w_{0,j+1}) \cdot (f_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}} - (w_{0,j}, w_{0,j+1}, w_{0,j+2}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k})^{\mathrm{T}}. \tag{16}$$

Eq. (12) can be discretized in the same manner by simply replacing all the $u_{i,j-1,k}$ with the secondary fictitious value $f_{i,j-1,k}$.

To solve for three unknowns, $f_{i,j,k}$, $f_{i,j+1,k}$ and $f_{i,j-1,k}$, one more equation is needed. For this purpose, we make use of another interface intersecting point $(x'_{\mathrm{o}}, y'_{\mathrm{o}}, z'_{\mathrm{o}})$ located between $(i,j-1,k)$ and (i,j,k). Note that this interface intersecting point always exists in the situation that $(i,j-1,k)$ and (i,j,k) are not on the same subdomain. We should use the jump condition of the function, which, unlike the flux jump condition, does not introduce any additional derivative. The jump condition at the intersecting point $(x'_{\mathrm{o}}, y'_{\mathrm{o}}, z'_{\mathrm{o}})$ can be discretized as

$$[u]' = (w'_{0,j-1}, w'_{0,j}, w'_{0,j+1}) \cdot [(f_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}} - (u_{i,j-1,k}, f_{i,j,k}, u_{i,j+1,k})^{\mathrm{T}}]. \tag{17}$$

Combining Eqs. (16), (17) and modified Eq. (12), the primary fictitious values $f_{i,j,k}$ and $f_{i,j+1,k}$ can be represented in terms of $u_{i,j-1,k}$, $u_{i,j,k}$, $u_{i,j+1,k}$, $u_{i,j+2,k}$ and other *12* auxiliary points located in x–y and y–z planes.

In the situation depicted in Fig. 6b, auxiliary point $(i,j+2,k)$ and irregular point $(i,j+1,k)$ are not on the same subdomain. The secondary fictitious value $f_{i,j+2,k}$ is used to replace $u_{i,j+2,k}$ in the discretization of Eqs. (2) and (3). By use of the jump condition on the interface intersecting point $(x^*_{\mathrm{o}}, y^*_{\mathrm{o}}, z^*_{\mathrm{o}})$ located between $(i,j+1,k)$ and $(i,j+2,k)$, the third equation can be discretized as

$$[u]^* = (w^*_{0,j}, w^*_{0,j+1}, w^*_{0,j+2}) \cdot [(u_{i,j,k}, f_{i,j+1,k}, u_{i,j+2,k})^{\mathrm{T}} - (f_{i,j,k}, u_{i,j+1,k}, f_{i,j+2,k})^{\mathrm{T}}]. \tag{18}$$

Eq. (2) can then be discretized with primary fictitious values, $f_{i,j,k}$, $f_{i,j+1,k}$ and secondary fictitious value $f_{i,j+2,k}$

$$[u] = (w_{0,j-1}, w_{0,j}, w_{0,j+1}) \cdot (u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}} - (w_{0,j}, w_{0,j+1}, w_{0,j+2}) \cdot (f_{i,j,k}, u_{i,j+1,k}, f_{i,j+2,k})^{\mathrm{T}}. \tag{19}$$

Eq. (12) can be discretized in the same manner by replacing all the $u_{i,j+2,k}$ by the secondary fictitious value $f_{i,j+2,k}$.

Combining Eqs. (18), (19) and the modified Eq. (12), the primary fictitious values $f_{i,j,k}$ and $f_{i,j+1,k}$ can still be represented in terms of $u_{i,j-1,k}$, $u_{i,j,k}$, $u_{i,j+1,k}$, $u_{i,j+2,k}$ and other 12 auxiliary points located in x–y and y–z planes.

Note that the above scheme assumes that either $u_{i,j+1,k}$ and $u_{i,j+2,k}$ locate on the same subdomain as shown in Fig. 6a, or $u_{i,j-1,k}$ and $u_{i,j,k}$ locate on the same subdomain as shown in Fig. 6b. That is, there is only one secondary fictitious value needed, which is $f_{i,j-1,k}$ in Fig. 6a or $f_{i,j+2,k}$ in Fig. 6b. Therefore, one more equation, Eq. (17) or Eq. (18), is enough to resolve the situation depicted in Fig. 6a or Fig. 6b, respectively. This is generally true in our second-order MIB scheme.

2.5.2. Third-order scheme

The main obstacle in constructing higher-order interface schemes for arbitrarily complex interfaces is the presence of geometric singularities. It is important to understand what high order interface schemes one can construct for a sharp-edged and sharp-tipped interface depicted in Fig. 6a. To this end, we discuss the feasibility of a third-order scheme. However, we have to relax some rules in the MIB method in order to construct a third-order one. Specifically, we need to admit asymmetric or one-sided discretization schemes near the interface, instead of purely central finite differences as employed in our earlier MIB method. To discretize Eq. (1) up to the third-order accuracy at a typical point (i,j,k) around the singularity depicted in Fig. 6a, three fictitious values are required for a non-central finite difference scheme. Let us denote them as $f_{i,j+2,k}, f_{i,j+1,k}$ and $f_{i,j-1,k}$. In addition, fictitious value $f_{i,j,k}$ is required for the third-order non-central discretizations at points $(i,j-1,k)$ and $(i,j+1,k)$. Therefore, around the typical singular point (i,j,k), four primary fictitious values are to be solved. Consequently, we need to discretize four jump conditions around two interface intersecting points, (x,y,z) and $(x'_{\mathrm{o}},y'_{\mathrm{o}},z'_{\mathrm{o}})$. The first two equations are

$$\begin{aligned} [u] &= u^+ - u^-, \\ [u'] &= u'^+ - u'^-, \end{aligned} \tag{20}$$

which can be discretized with function values and four unknowns, $f_{i,j,k}$, $f_{i,j+1,k}$, $f_{i,j-1,k}$ and $f_{i,j+2,k}$

$$\begin{aligned} u^+ &= (w_{0,j-1}, w_{0,j}, w_{0,j+1}, w_{0,j+2}) \cdot (f_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k}, f_{i,j+2,k})^{\mathrm{T}}, \\ u^- &= (w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k})^{\mathrm{T}}, \\ u'^+ &= (w'_{0,j-1}, w'_{0,j}, w'_{0,j+1}, w'_{0,j+2}) \cdot (f_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k}, f_{i,j+2,k})^{\mathrm{T}}, \\ u'^- &= (w_{0,j-2}, w_{0,j-1}, w_{0,j}, w_{0,j+1}) \cdot (u_{i,j-2,k}, u_{i,j-1,k}, f_{i,j,k}, u_{i,j+1,k})^{\mathrm{T}}. \end{aligned} \tag{21}$$

The other two jump conditions are given by Eq. (12) at points $(x_{\mathrm{o}},y_{\mathrm{o}},z_{\mathrm{o}})$ and $(x'_{\mathrm{o}},y'_{\mathrm{o}},z'_{\mathrm{o}})$, respectively. Two pairs of partial derivatives, $u_x^-, u_z^-, u_x'^+$, and $u_z'^+$ at two intersecting points are eliminated. Here, $u_y^+, u_y^-, u_y'^+$ and $u_y'^-$ can be discretized in a similar way as that in Eq. (21) for u^+, u^-, u'^+ and u'^-. The discretizations of u_z^+ (or u_z^-) and u_x^+ (or u_x^-) involve $2\times2\times3$ type I auxiliary points around $(x_{\mathrm{o}},y_{\mathrm{o}},z_{\mathrm{o}})$. Similarly, the approximations of $u_z'^+$ (or $u_z'^-$) and $u_x'^+$ (or $u_x'^-$) take another $2\times2\times3$ type I auxiliary points around $(x'_{\mathrm{o}},y'_{\mathrm{o}},z'_{\mathrm{o}})$. After obtained all the fictitious values, the third-order discretization of the elliptic equation can be achieved without introducing additional difficulty. The fourth-order central finite difference discretization should be used away from the interface. While near the interface, third-order non-central finite difference discretization can be used. This scheme is as feasible as the second-order MIB scheme for geometric singularities. However, it is not clear at this point whether a truly fourth-order MIB scheme can be constructed for arbitrarily complex interfaces with sharp geometric singularities. In the following subsection, we describe 3D higher-order MIB schemes for smooth interfaces and for interfaces with moderate geometric singularities.

2.6. Higher-order MIB algorithms

The MIB method was originally proposed as an arbitrarily higher-order interface algorithm [71]. The fourth- and sixth-order MIB schemes in 2D have been developed for curved interfaces [73,74]. The 3D generalization of higher-order MIB schemes is described in this work. In brief, an nth-order MIB method means the use of the nth-order finite difference discretization. Therefore, more irregular points, and thus more fictitious values are required in our fictitious domain formulation, although an equivalent interpolation formulation can be constructed to avoid the use of fictitious values. On the other hand, because we avoid using higher-order interface jump conditions, the number of interface conditions remains the same. Therefore, lower-order jump conditions are repeatedly used as proposed in our earlier paper [71]. The basic procedure is to first generate a pair of fictitious values of required nth-order accuracy around an interface intersecting point with given lower-order interface jump conditions. By repeatedly using the same set of jump conditions, we then add two more fictitious values at a time until all the required fictitious values on a mesh line are determined. This procedure is repeated for other interface intersecting points and other mesh lines. Finally, the elliptic equation is solved by the nth-order central finite difference scheme with appropriate fictitious values.

2.6.1. Generation of the first pair of fictitious values

Let us consider a 3D fourth-order MIB scheme. The first step is to calculate the first pair of fictitious values by using fourth-order finite difference stencils to discretize interface jump conditions.

Unlike in the second-order MIB scheme, u^+, u^-, u_y^+ and u_y^- are now represented by five-point finite difference schemes

$$\begin{aligned}
u^+ &= (w_{0,j-3}, w_{0,j-2}, w_{0,j-1}, w_{0,j}, w_{0,j+1}) \cdot (u_{i,j-3,k}, u_{i,j-2,k}, u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}},\\
u^- &= (w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k}, u_{i,j+4,k})^{\mathrm{T}},\\
u_y^+ &= (w_{1,j-3}, w_{1,j-2}, w_{1,j-1}, w_{1,j}, w_{1,j+1}) \cdot (u_{i,j-3,k}, u_{i,j-2,k}, u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k})^{\mathrm{T}},\\
u_y^- &= (w_{1,j}, w_{1,j+1}, w_{1,j+2}, w_{1,j+3}, w_{1,j+4}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k}, u_{i,j+4,k})^{\mathrm{T}}.
\end{aligned} \tag{22}$$

To approximate u_z^+ or u_z^- with the fourth-order accuracy, we need five u values along the auxiliary line $y = y_o$ on the y–z plane, see Fig. 7. These values are approximated by 20 more auxiliary points. In the situation shown in Fig. 7, u_z^- can be represented as follows:

$$u_z^- = (w_{1,k-1}, w_{1,k}, w_{1,k+1}, w_{1,k+2}, w_{1,k+3}) \cdot (u_{i,o,k-1}, u_{i,o,k}, u_{i,o,k+1}, u_{i,o,k+2}, u_{i,o,k+3})^{\mathrm{T}}, \tag{23}$$

where

$$\begin{aligned}
u_{i,o,k-1} &= (w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}, w_{0,j+5}) \cdot (u_{i,j+1,k-1}, u_{i,j+2,k-1}, u_{i,j+3,k-1}, u_{i,j+4,k-1}, u_{i,j+5,k-1})^{\mathrm{T}},\\
u_{i,o,k} &= (w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}) \cdot (f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k}, u_{i,j+4,k})^{\mathrm{T}},\\
u_{i,o,k+1} &= (w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}, w_{0,j+5}) \cdot (u_{i,j+1,k+1}, u_{i,j+2,k+1}, u_{i,j+3,k+1}, u_{i,j+4,k+1}, u_{i,j+5,k+1})^{\mathrm{T}},\\
u_{i,o,k+2} &= (w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}) \cdot (u_{i,j,k+2}, u_{i,j+1,k+2}, u_{i,j+2,k+2}, u_{i,j+3,k+2}, u_{i,j+4,k+2})^{\mathrm{T}},\\
u_{i,o,k+3} &= (w_{0,j-1}, w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}) \cdot (u_{i,j-1,k+3}, u_{i,j,k+3}, u_{i,j+1,k+3}, u_{i,j+2,k+3}, u_{i,j+3,k+3}, u_{i,j+4,k+3})^{\mathrm{T}}.
\end{aligned}$$

Similarly, u_x^- can also be computed along the auxiliary line $y = y_o$ on the x–y plane, the dashed line shown in Fig. 7. Then, by solving Eqs. (2) and (12) with $l = 1$ and $m = 5$, two fictitious values $f_{i,j,k}$ and $f_{i,j+1,k}$ can be easily represented in terms of the function values of 48 nearby grid points.

2.6.2. Iterative generation of other fictitious values

Let us consider a typical case as shown in Fig. 7, in order to discretize u_{yy} at point (i,j,k) by central finite difference schemes with nth-order accuracy, $\frac{n}{2}$ fictitious values on the right side of point (i,j,k) are required. An iterative scheme is applied to determine the rest of fictitious values. The order of generating fictitious values is illustrated in Fig. 8.

Here, u^+, u^-, u_y^+ and u_y^- are represented by six-point finite difference schemes, which include the second pair of fictitious values $f_{i,j-1,k}$ and $f_{i,j+4,k}$

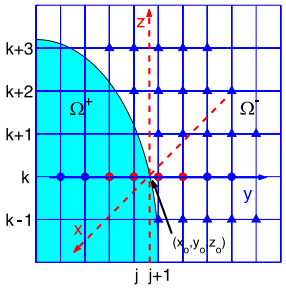

Fig. 7. Illustration of a fourth-order MIB scheme. The y-mesh line intersects the interface at (x_o, y_o, z_o). A set of irregular points $(i,j-1,k)$, (i,j,k), $(i,j+1,k)$ and $(i,j+2,k)$, in red ●, is on the y-mesh line. Type I auxiliary points are in blue ▲. Type II auxiliary points are in blue ●. Auxiliary lines, in red dashed lines along the x- and z-directions, are used for computing u_x and u_z. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

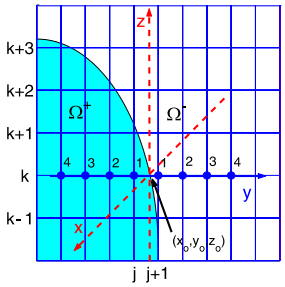

Fig. 8. Illustration of iterative schemes. The y-mesh line intersects the interface at (x_o, y_o, z_o). The fictitious values at ● points are generated pairwisely. The order of generating fictitious values is indicated by the numbers on the upper right corner of ●.

$$
\begin{aligned}
u^+ &= (w_{0,j-3}, w_{0,j-2}, w_{0,j-1}, w_{0,j}, w_{0,j+1}, w_{0,j+2}) \cdot (u_{i,j-3,k}, u_{i,j-2,k}, u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k}, f_{i,j+2,k})^{\mathrm{T}},\\
u^- &= (w_{0,j-1}, w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}) \cdot (f_{i,j-1,k}, f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k}, u_{i,j+4,k})^{\mathrm{T}},\\
u_y^+ &= (w_{1,j-3}, w_{1,j-2}, w_{1,j-1}, w_{1,j}, w_{1,j+1}, w_{1,j+2}) \cdot (u_{i,j-3,k}, u_{i,j-2,k}, u_{i,j-1,k}, u_{i,j,k}, f_{i,j+1,k}, f_{i,j+2,k})^{\mathrm{T}},\\
u_y^- &= (w_{1,j-1}, w_{1,j}, w_{1,j+1}, w_{1,j+2}, w_{1,j+3}, w_{1,j+4}) \cdot (f_{i,j-1,k}, f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k}, u_{i,j+4,k})^{\mathrm{T}}.
\end{aligned}
\tag{24}
$$

In these equations, $f_{i,j+1,k}$ and $f_{i,j,k}$ are resolved earlier, and are used as secondary fictitious values, i.e., the fictitious values that are used to resolve other fictitious values.

The approximation of u_z^+ or u_z^- is still given by Eq. (23). The only difference is that here $u_{i,o,k}$ is given by the following equation

$$
u_{i,o,k} = (w_{0,j-1}, w_{0,j}, w_{0,j+1}, w_{0,j+2}, w_{0,j+3}, w_{0,j+4}) \cdot (f_{i,j-1,k}, f_{i,j,k}, u_{i,j+1,k}, u_{i,j+2,k}, u_{i,j+3,k}, u_{i,j+4,k})^{\mathrm{T}}.
$$

Similarly, u_x^- can be computed along the auxiliary line $y = y_o$ in the x–y plane. Then, by solving Eqs. (2) and (12) with $l = 1$ and $m = 5$ again, two more fictitious values $f_{i,j-1,k}$ and $f_{i,j+2,k}$ can be easily represented in terms of the function values of 48 points around them and the secondary fictitious values $f_{i,j,k}$ and $f_{i,j+1,k}$. In addition, it may be noticed that $f_{i,j,k}$ and $f_{i,j+1,k}$ are also functions of the same 48 surrounding points. Therefore, the newly found fictitious values $f_{i,j-1,k}$ and $f_{i,j+2,k}$ can be discretized in terms of the function values of 48 surrounding grid points.

All the other fictitious values labeled in Fig. 8 can be systematically found in a similar manner. It is important to note that the accuracy of the ith pair of fictitious values is restricted by the discretization accuracy of the first pair of fictitious values. Therefore, in order to obtain sixth-order schemes, a seven-point finite difference stencil should be used to generate the first pair of fictitious values.

2.6.3. High order discretization of elliptic equations

The fourth-order discretization of Eq. (1) at point (i,j,k) is then given by

$$
\beta_x u_x + \beta u_{xx} + \beta_y u_y + \beta u_{yy} + \beta_z u_z + \beta u_{zz} - \kappa u(i,j,k) = q(i,j,k), \tag{25}
$$

where

$$
\begin{aligned}
u_x &= (w_{1,i-2}, w_{1,i-1}, w_{1,i}, w_{1,i+1}, w_{1,i+2}) \cdot (\tilde{u}_{i-2,j,k}, \tilde{u}_{i-1,j,k}, u_{i,j,k}, \tilde{u}_{i+1,j,k}, \tilde{u}_{i+2,j,k})^{\mathrm{T}},\\
u_{xx} &= (w_{2,i-2}, w_{2,i-1}, w_{2,i}, w_{2,i+1}, w_{2,i+2}) \cdot (\tilde{u}_{i-2,j,k}, \tilde{u}_{i-1,j,k}, u_{i,j,k}, \tilde{u}_{i+1,j,k}, \tilde{u}_{i+2,j,k})^{\mathrm{T}},\\
u_y &= (w_{1,j-2}, w_{1,j-1}, w_{1,j}, w_{1,j+1}, w_{1,j+2}) \cdot (\tilde{u}_{i,j-2,k}, \tilde{u}_{i,j-1,k}, u_{i,j,k}, \tilde{u}_{i,j+1,k}, \tilde{u}_{i,j+2,k})^{\mathrm{T}},\\
u_{yy} &= (w_{2,j-2}, w_{2,j-1}, w_{2,j}, w_{2,j+1}, w_{2,j+2}) \cdot (\tilde{u}_{i,j-2,k}, \tilde{u}_{i,j-1,k}, u_{i,j,k}, \tilde{u}_{i,j+1,k}, \tilde{u}_{i,j+2,k})^{\mathrm{T}},\\
u_z &= (w_{1,k-2}, w_{1,k-1}, w_{1,k}, w_{1,k+1}, w_{1,k+2}) \cdot (\tilde{u}_{i,j,k-2}, \tilde{u}_{i,j,k-1}, u_{i,j,k}, \tilde{u}_{i,j,k+1}, \tilde{u}_{i,j,k+2})^{\mathrm{T}},\\
u_{zz} &= (w_{2,k-2}, w_{2,k-1}, w_{2,k}, w_{2,k+1}, w_{2,k+2}) \cdot (\tilde{u}_{i,j,k-2}, \tilde{u}_{i,j,k-1}, u_{i,j,k}, \tilde{u}_{i,j,k+1}, \tilde{u}_{i,j,k+2})^{\mathrm{T}}
\end{aligned}
\tag{26}
$$

and

$$\tilde{u}_{l,m,n} = \begin{cases} u_{l,m,n} & \text{if } (l,m,n) \text{ and } (i,j,k) \text{ in the same subdomain,} \\ f_{l,m,n} & \text{otherwise.} \end{cases} \tag{27}$$

Obviously, the 3D MIB procedure outlined here is systematic and is of arbitrarily high order accuracy, in principle. In practical applications, complex interface and geometric singularities give rise to the difficulty of finding required auxiliary points. A wide variety of MIB techniques developed in our earlier work [69,73,74] and in the present work will alleviate this difficulty.

As shown in an earlier section, it is possible to slightly alleviate the difficulty of finding required auxiliary points by using non-central or one-sided finite difference discretizations near the interface. This is similar to our MIB treatment of boundary conditions [72]. However, the difficulty persists for sharp-edged and sharp-tipped interfaces. In the following section, we explore the capability and examine the limitation of the proposed MIB method.

3. Results and discussion

In this section, we examine the convergence order, test the accuracy, validate the matrix optimization and demonstrate the capability of the proposed MIB II schemes. The impact of the proposed algorithm acceleration is studied with the molecular surfaces of a diatom and a protein. Molecular surfaces are generated with the MSMS program [60] with probe radius 1.4 and density 10. Reported CPU time is recorded on an AMD Turion 64™ MT-30 laptop with 1.6 GHz clock speed. The ability of the proposed MIB II for handling geometric singularities is tested with systems of two, three and eighteen intersecting spheres (van der Waals surfaces). Both molecular surfaces and van der Waals surfaces are very important to the theoretical modeling of biophysics and structural biology. Comparison is made to our earlier MIB technique, the MIB I [75], in which the matrix is not optimized and interface singularities are not treated. A challenging test case with a missile interface is provided to demonstrate the robustness of the MIB II on severe geometric singularities. The numerical tests on the fourth-order MIB method are carried out. Five different interfaces are used and oscillatory solutions are compared. Moreover, cases with variable diffusion coefficients and non-zero linear term are studied. Finally, we demonstrate the sixth-order MIB scheme with three test cases. A detailed comparison on the performance of second-, fourth- and sixth-order MIB schemes has been provided. To analyze the numerical performance of our MIB schemes, we use two error measurements, the maximum absolute error L_∞ and the surface maximum absolute error E_1

$$\begin{aligned} L_\infty &= \max_{\Omega} |u(x,y,z) - u_{\text{ex}}(x,y,z)|, \\ E_1 &= \max_{\Gamma} |u(x,y,z) - u_{\text{ex}}(x,y,z)|, \end{aligned} \tag{28}$$

where u and u_{ex} are numerical and exact solutions, respectively. Here E_1 is computed over all the irregular points near the interface Γ where the modified difference schemes are used. In many surface error maps, the difference, $u(x,y,z) - u_{\text{ex}}(x,y,z)$, is plotted. The standard preconditioned biconjugate gradient (BiCG) is used for solving the linear algebraic equation system and the tolerance of the BiCG iterations is set to be 1.0×10^{-6} in all test cases of second-order accuracy, and 1.0×10^{-14} in all test cases of higher-order accuracy. The diagonal of the matrix is used as the preconditioner. The number of BiCG iterations, N_{BiCG}, is listed for some test cases to access the speed of convergence. The order of convergence is reported for the L_∞ errors in many cases. In this section, we denote the subdomain enclosing the subjects of interest, such as molecules and missiles, as Ω^- and the rest as Ω^+.

3.1. Case 1: Matrix optimization

The different selections of subdomains, sets of mesh lines and sets of auxiliary points are compared in the following test case. In this study, a smooth interface as shown in Fig. 9 is used to avoid the effect of interface singularities. The exact solutions as shown in Fig. 9a are given by

$$u^- = 10\cos x \cos y \cos z + 20, \quad u^+ = 10(x+y+z) + 1 \tag{29}$$

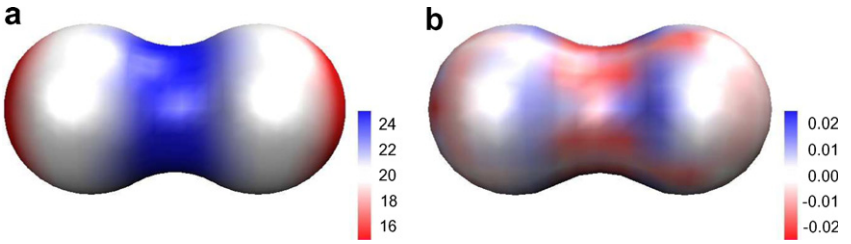

Fig. 9. Surface maps of the exact solution, (a), and numerical errors, (b), for the molecular surface of a diatom of atomic radius 1.5 Å with atoms centered at $(-1.52,0,0)$ and $(1.35,0,0)$, respectively.

with coefficients

$$\beta^- = 1, \quad \beta^+ = 80, \quad \text{and} \quad \kappa = 0. \tag{30}$$

The source term $q(x,y,z)$ and interface conditions $[u]$, $[\beta u_n]$ can be easily derived from the exact solutions. The computational domain is set to $[-6,6]\times[-4,5]\times[-4,4]$.

Table 1 gives a comparison of numerical errors, convergence orders, number of iterations and CPU time for six different combinations of Step 1 and Step 2 as illustrated in Section 2.3. The preferred auxiliary point sets (i) and (ii) are used in this comparison whenever possible. The L_∞ orders of convergence are quite similar in all combinations and the designed second-order is numerically confirmed. In terms of numerical errors, different choices of mesh line sets in Step 2 do not affect much the accuracy. However, the selection of Ω^- in Step 1 leads to slightly larger errors, probably because of the geometric constraints of Ω^-. The most significant difference is in the number of biconjugate gradient iterations and CPU time. The proposed mesh line set (a) requires 30–50% fewer N_{BiCG} and less CPU time than other mesh line sets, indicating the impact of proposed optimization schemes. Moreover, the selection of Ω^- in Step 1 results in fewer N_{BiCG} in most cases, possibly because Ω^- is the subdomain enclosed by the interface, and the auxiliary points selected from Ω^- is therefore more compactly distributed than those selected from Ω^+.

Table 2 compares six different combinations of Step 2 and Step 3 as illustrated in Section 2.3 with a fixed subdomain Ω^+. It is seen that both iteration number N_{BiCG} and numerical errors increase slightly when non-compact auxiliary points are selected in Step 3. However, no significant difference is observed. Comparing Tables 1 and 2, the selection of Step 2 is the dominant factor for the speed of convergence. In order to achieve the best result in terms of both numerical accuracy and convergence speed, the combination of subdomain Ω^+, mesh line set (a) and auxiliary point sets (i) and (ii) is preferred. Fig. 9b illustrates the surface map of the numerical errors obtained from this preferred selection at $h = 0.25$. It is evident that most errors are distributed around the extrema of the solution.

Table 1
Comparison of different choices made for Step 1 (Ω^+ or Ω^-) and Step 2 (mesh line sets as shown in Fig. 3)

Step 2	h	Step 1: Ω^+					Step 1: Ω^-				
		L_∞	E_1	N_{BiCG}	CPU(s)	Order	L_∞	E_1	N_{BiCG}	CPU(s)	Order
(a)	0.5	1.3e−1	5.7e−2	83	0.6		1.3e−1	1.3e−1	90	0.6	
	0.25	3.4e−2	9.4e−3	168	8.8	1.9	3.9e−2	3.9e−2	169	8.8	1.7
	0.125	8.9e−3	1.5e−3	337	133.9	1.9	8.6e−3	8.6e−3	338	135.0	2.2
(b)	0.5	1.3e−1	5.7e−2	133	0.9		2.6e−1	2.6e−1	95	0.7	
	0.25	3.9e−2	3.9e−2	220	11.3	1.9	4.7e−2	4.7e−2	182	9.4	2.5
	0.125	8.8e−3	1.5e−3	444	175.9	2.0	9.4e−3	8.1e−3	348	139.0	2.3
(c)	0.5	1.3e−2	6.7e−2	169	1.1		2.6e−1	2.6e−1	95	0.7	
	0.25	3.4e−2	9.4e−3	218	11.2	1.9	4.7e−2	4.7e−2	182	9.3	2.5
	0.125	8.8e−3	1.5e−3	456	181.0	1.9	9.4e−3	8.1e−3	348	136.6	2.3

Auxiliary point sets (i) and (ii) as shown in Fig. 4 are used for Step 3 whenever they are available.

Table 2
Comparison of different choices made for Step 2 (mesh line sets as shown in Fig. 3) and Step 3 (auxiliary point sets as shown in Fig. 4)

Step 2	h	Step 3: (i) and (ii)					Step 3: (iii) and (iv)				
		L_∞	E_1	N_{BiCG}	CPU(s)	Order	L_∞	E_1	N_{BiCG}	CPU(s)	Order
(a)	0.5	1.3e−1	5.7e−2	83	0.6		1.3e−1	5.7e−2	85	0.6	
	0.25	3.4e−2	9.4e−3	168	8.8	1.9	3.4e−2	9.4e−3	172	8.8	1.9
	0.125	8.9e−3	1.5e−3	337	133.9	1.9	9.0e−3	1.4e−3	355	139.2	1.9
(b)	0.5	1.3e−1	5.7e−2	133	0.9		1.3e−1	5.7e−2	143	0.9	
	0.25	3.9e−2	3.9e−2	220	11.3	1.9	3.4e−2	9.4e−3	221	11.2	1.9
	0.125	8.8e−3	1.5e−3	444	175.9	2.0	8.9e−3	1.6e−3	449	175.0	1.9
(c)	0.5	1.3e−2	6.7e−2	169	1.1		1.5e−1	1.5e−1	201	1.3	
	0.25	3.4e−2	9.4e−3	218	11.2	1.9	3.4e−2	9.4e−3	224	11.3	2.1
	0.125	8.8e−3	1.5e−3	456	181.0	1.9	9.0e−3	1.7e−3	461	179.6	1.9

Subdomain Ω^+ is used.

Due to the simple geometry, the difference of various selections is not very obvious. To better test our method, we consider a more complex system, the molecular surface of a protein whose coordinates are obtained from the Protein Data Bank (PDB ID: 2pde). To obtain a full all-atom model for the protein, all attached water molecules are cleaned and hydrogen atoms were added. Atomic van der Waals radii defining the dielectric boundary were taken from the CHARMM22 force field [48], and the molecular surface is generated by the MSMS [60]. It is well-known that the molecular surface definition admits cusps and sharp self-intersecting surfaces [60]. This test is motivated by the electrostatic analysis of biomolecules in the implicit solvent model in structural biology. A detailed account of this aspect can be found in Ref. [70]. The exact solutions for this system are still given by Eqs. (29) and (30), and illustrated in Fig. 10a. It is seen that the molecular surface of protein 2pde with 667 atoms has a very irregular interface with geometric singularities. The computational domain is set to $[-16,16]\times[-20,15]\times[-17,19]$ Å^3, which is slightly larger than the domain of the molecular surface. All calculations are performed at the mesh size of $h = 0.5$ Å. Table 3 shows 12 different combinations of subdomains, mesh line sets and auxiliary point sets. The improvement by using mesh line set (a) over that of (b) or (c) is more obvious in this test case. Mesh line sets (b) and (c) require up to 30 times more N_{BiCG} than that required by mesh line set (a) given the same combination of Step 1 and Step 3. Moreover, mesh line set (a) is consistently more accurate than mesh line sets (b) and (c). Table 3 also confirms that the combination of subdomain Ω^+, mesh line set (a) and auxiliary point sets (i) and (ii) is the best choice for faster convergence and higher accuracy. A good alternative is the combination of Ω^-, mesh line set (a) and auxiliary point sets (i) and (ii).

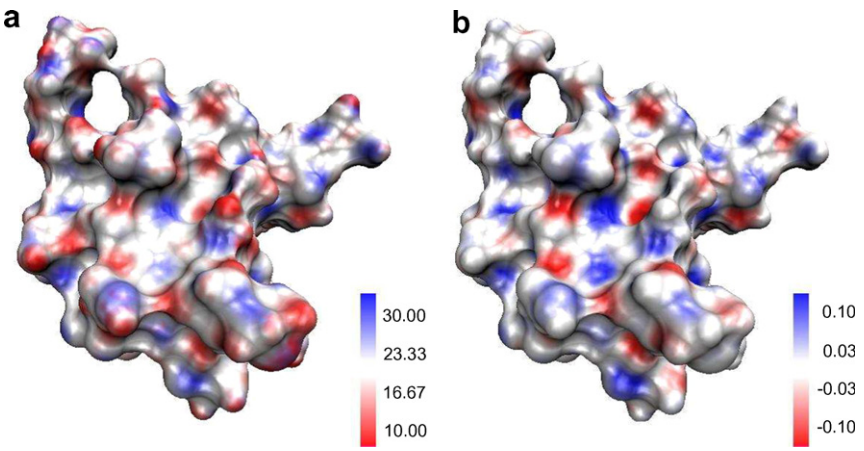

Fig. 10. Surface maps of the exact solution, (a), and numerical errors, (b), for the molecular surface of protein 2pde.

Table 3
Comparison of different choices made for subdomains, mesh line sets and auxiliary point sets

Step 3	Step 2	Step 1: Ω^+				Step 1: Ω^-			
		L_∞	E_1	N_{BiCG}	CPU(s)	L_∞	E_1	N_{BiCG}	CPU(s)
(i) and (ii)	(a)	2.29e−1	1.15e−1	294	77	7.65e−1	7.65e−1	285	75
	(b)	3.85e−1	3.85e−1	2051	579	9.51e−1	9.51e−1	331	88
	(c)	3.63e−1	3.63e−1	10812	2675	9.49e−1	9.49e−1	310	86
(iii) and (iv)	(a)	4.22e−1	4.22e−1	322	85	9.62e−1	9.62e−1	300	79
	(b)	6.66e−1	6.66e−1	3244	815	1.47e−0	1.47e−0	324	85
	(c)	5.96e−1	5.96e−1	4051	1010	1.67e−0	1.67e−0	316	84

The interface is defined by the molecular surface of protein 2pde, shown in Fig. 10. The exact solutions are given by Eqs. (29) and (30).

The L_∞ order of convergence is investigated for this test case in Table 4. Mesh line set (a) is used in all the cases in Table 4. It can be seen that second-order convergence is achieved in all four combinations although the numerical errors vary in different combinations. The combination of subdomain Ω^+ and auxiliary point sets (i) and (ii) is still the best choice while the combination of subdomain Ω^- and point sets (iii) and (iv) gives similar numerical errors at $h = 0.25$.

The exact solution is mapped onto the interface in Fig. 10a. The numerical errors obtained with the best combination and $h = 0.5$ Å are mapped on the interface in Fig. 10b. It can be seen that the numerical errors are about 0.3% of the original solution at $h = 0.5$ Å (note the difference in scale). Another interesting point is that there is a good correlation in the color distribution in two figures, indicating that numerical errors are mostly induced by the amplitude variation of the solution instead of the geometric variation of the interface. The matrix optimization and the treatment of geometric singularities proposed in this paper underpin our recent success in solving Poisson–Boltzmann equation for electrostatic analysis of biomolecules [70].

3.2. Case 2: Validation on interfaces with geometric singularities

The performance of the present MIB method on interface singularities is tested by using the van der Waals surfaces of three different systems shown in Fig. 11. These interfaces are composed of spheres with their intersecting sections removed. Singularities are generated at the intersecting wedges and tips as shown in Fig. 12, which gives the cross-section view of the interfaces. Fig. 12a indicates a sharp wedge generated by the intersecting of two spheres. Fig. 12b illustrates a sharp tip and sharp wedges generated by the intersection of three spheres.

The same exact solutions as those in Case 1, given by Eqs. (29) and (30), are employed in the present test examples. The computational domains are $[-5,4]\times[-3,3]\times[-3,3]$, $[-5,4]\times[-3,5]\times[-3,3]$ and $[-6,4]\times[-4,5]\times[-5,4]$ for three test cases whose interfaces are given in Fig. 11a–c, respectively. The numerical errors are computed at mesh sizes $h = 0.5$, 0.25 and 0.125 by using the present MIB method (MIB II) and our earlier MIB method (MIB I), in which the surface singularities are treated by simply replacing unknown fictitious values with the values at neighbor points. Table 5 lists the overall maximum

Table 4
Order of convergence test on selections of type I auxiliary points using different combinations of Step 1 and Step 3

Step 3	Step 1: Ω^+				Order	Step 1: Ω^-			Order
	h	L_∞	E_1	N_{BiCG}		L_∞	E_1	N_{BiCG}	
(i) and (ii)	0.5	2.29e−1	1.15e−1	294	2.0	7.65e−1	7.65e−1	285	1.9
	0.25	5.87e−2	1.86e−2	537		2.08e−1	2.08e−1	551	
(iii) and (iv)	0.5	4.22e−1	4.22e−1	322	2.8	9.62e−1	.62e−1	300	2.7
	0.25	5.86e−2	4.91e−2	575		1.49e−1	1.49e−1	552	

Mesh line set (a) is used in all the calculations. The interface is defined by the molecular surface of protein 2pde, shown in Fig. 10. The exact solutions are given by Eqs. (29) and (30).

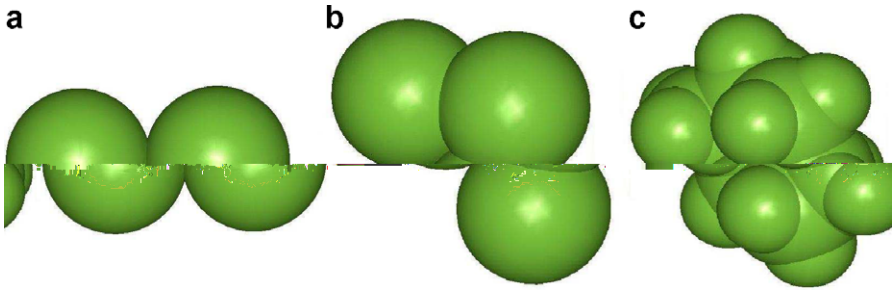

Fig. 11. Interface of intersecting spheres. (a) two intersecting spheres of radius 1.5 with centers $(-1.52,0,0)$ and $(1.35,0,0)$. (b) three intersecting spheres of radius 1.5 with centers $(-1.52,0,0)$, $(1.35,0,0)$ and $(0,1.62,0)$. (c) 18 intersecting spheres. The coordinates of the center and radius of each spheres are given in the form of (x,y,z,r) as follows: $(-2.0270,0.9540,-0.6510,1.7)$, $(-1.6690,0.2340,0.6650,1.7)$, $(-0.4530,-0.6870,0.4410,1.7)$, $(0.7510,0.1480,-0.0400,1.7)$, $(0.3930,0.8680,-1.3560,1.7)$, $(-0.8230,1.7880,-1.1320,1.7)$, $(-2.2840,0.2080,-1.4180,1.2)$, $(-2.8880,1.6170,-0.4830,1.2)$, $(-2.5270,-0.3680,0.9970,1.2)$, $(-1.4260,0.9800,1.4350,1.2)$, $(-0.1960,-1.1890,1.3850,1.2)$, $(-0.7010,-1.4410,-0.3200,1.2)$, $(1.0070,0.8940,0.7270,1.2)$, $(1.6120,-0.5150,\text{minus}0.2080,1.2)$, $(0.1490,0.1210,-2.1260,1.2)$, $(1.2510,1.4700,-1.6880,1.2)$, $(-1.0810,2.2910,-2.0760,1.2)$, and $(-0.5750,2.5430,-0.3710,1.2)$.

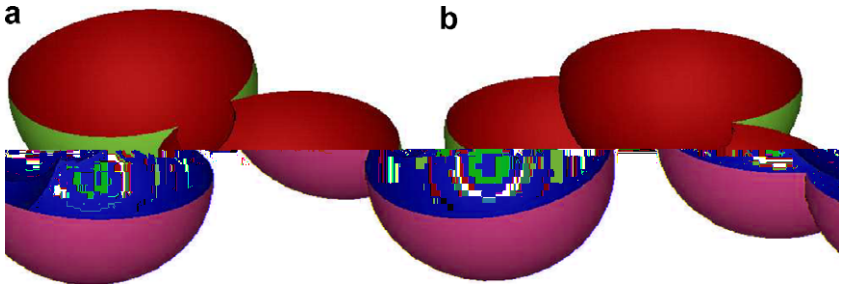

Fig. 12. Cross-section views of interface generated by intersecting spheres. (a) A cross-section of Fig. 11a; (b) a cross-section of Fig. 11b.

Table 5
Convergence test on interface of intersecting spheres

h		Case 2(a)		Case 2(b)		Case 2(c)	
		MIB II	MIB I	MIB II	MIB I	MIB II	MIB I
0.5	L_∞	2.57e−1	1.55	2.14e−1	1.05	2.38e−1	2.46
	E_1	2.57e−1	1.55	2.14e−1	1.05	1.09e−1	2.46
0.25	L_∞	3.21e−2	1.00	3.73e−2	1.21	5.83e−2	2.86
	E_1	2.28e−2	1.00	2.57e−2	1.21	4.96e−2	2.86
	Order	3.0	0.6	2.5	−0.2	2.0	−0.2
0.125	L_∞	7.21e−3	1.21	8.66e−3	1.16	1.42e−2	1.29
	E_1	5.23e−3	1.21	5.58e−3	1.16	3.82e−3	1.29
	Order	2.2	−0.3	2.1	0.1	2.0	1.2

Case 2(a): two intersecting spheres; Case 2(b): three intersecting spheres; Case 2(c): 18 intersecting spheres.

absolute error L_∞ and the surface maximum absolute error E_1. It is seen that the MIB II method achieves the designed second-order convergence in all three test cases. In fact, the accuracy of the MIB II is independent of the geometric complexity. At a given mesh size, the same level of accuracy is achieved by the MIB II in all three interfaces. In contrast, the MIB I scheme does not converge because of the geometric singularity. Moreover, L_∞ and E_1 errors are the same in MIB I, indicating that the largest error occurs at the interface. Since our 3D MIB I is of second-order convergence for smooth interfaces, these L_∞ and E_1 errors must be from geometric singularities. These results demonstrate the importance of special treatments

for geometric singularities in practical applications. Ordinary interface methods forfeit their power at sharp-edged and sharp-tipped interfaces.

In addition, Figs. 11a and 12a show clearly that the intersecting wedges are very sharp. The MIB II method handles this situation very well and is not subject to the critical angular limitation [69] any more. The following test case is designed to further confirm this point.

3.3. Case 3: A missile interface

To further verify the ability of the proposed MIB scheme for treating arbitrarily complex interface singularities, we consider a missile geometry which was reported by Wang and Srinivasan [67] as a challenge for grid generation. The missile tip is very sharp and the missile fins are very thin compared to the size of the geometry, see Fig. 13. These features normally take enormous amount of grids to reduce the local numerical errors. The computational domain is set to $[-5,49]\times[-12,11]\times[-12,11]$. Two sets of solutions are tested on this interface. The set of solutions of Case 3(a) is given by

$$u^- = \cos x \cos y \cos z, \quad u^+ = 0. \tag{31}$$

While Case 3(b) adopts the solution set given by Eq. (29). The coefficients β and κ in these two cases are the same as those given in Eq. (30).

The numerical errors of both cases are listed in Table 6. For both Case 2(a) and Case 2(b), their convergence order from $h = 0.5$ to $h = 0.25$ is slightly lower than what is designed because it is much more computationally challenging at $h = 0.25$ than at $h = 0.5$ in the following sense. At $h = 0.5$, there are a fewer irregular grid points located near the sharp edges of the missile. In other word, much sharp feature of the missile has not been recognized by a coarse mesh yet. While when the mesh is refined to $h = 0.25$, the sharp feature captures more grid points and there are more irregular points across the interface. When the mesh is refined further to $h = 0.125$, the number of irregular grid points increases again. It is seen that the proposed MIB scheme is of second-order convergence in the last mesh refinement from $h = 0.25$ to $h = 0.125$.

Fig. 13 provides the exact solution of Case 3(b) and errors obtained at three different mesh sizes. An important feature of our results is that, the errors are distributed around the extrema of the solution. This feature suggests that the thin fins and the sharp tip are properly treated and do not cause additional errors in our scheme. Indeed, there is little error on the tip and the fins due to our special treatment of singularities. The level of accuracy in Case 3(b) remains the same as that in Table 5, confirming that interface complexity and geometric singularities do not compromise the accuracy of the proposed MIB method.

3.4. Case 4: Fourth-order MIB scheme

Higher-order methods are particularly valuable for problems with high wavenumbers. Normally, it is difficult to construct higher-order methods for complex geometry. Moreover, the presence of interfaces will lead to low regularity in solutions and reduce the convergence order of common higher-order schemes. Furthermore, the presence of interface singularities will reduce the convergence order of traditional interface methods that are not designed for sharp edges and sharp tips, as shown in Table 5. Consequently, the best result in the literature was of 0.8th order [26] for 2D elliptic equations with interface singularities before year 2005, to our knowledge. In the present study, we demonstrate our fourth MIB method for elliptic interface problems with moderate interface singularities.

To examine the proposed method, we first employ three typical and relatively simple test interfaces, an ellipsoid, a cylinder and an oak acorn.

- Case 4(a): Ellipsoid

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1, \tag{32}$$

where $a = \frac{20}{7}$, $b = \frac{25}{14}$ and $c = \frac{25}{14}$. The computational domain is set to $[-5,5]\times[-5,5]\times[-5,5]$.

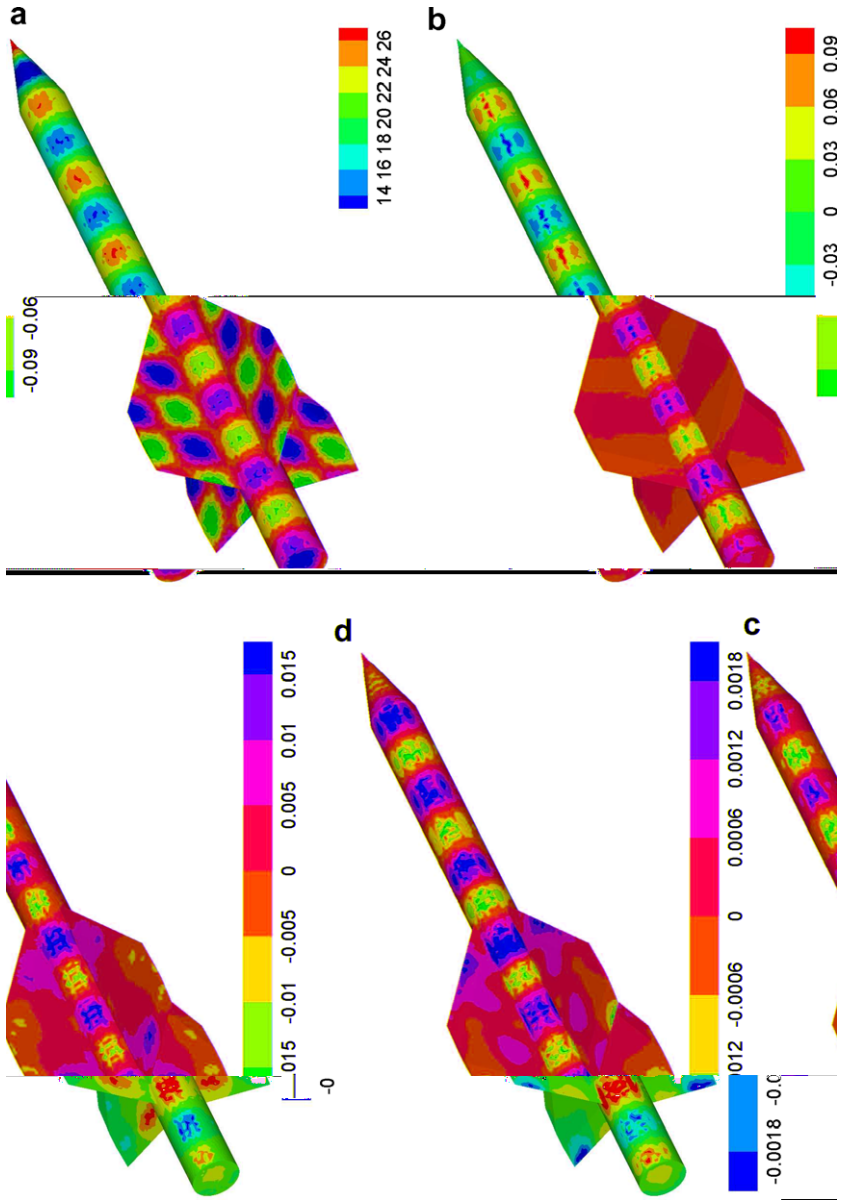

Fig. 13. A missile geometry with solution and error maps. Numerical errors are represented in seven scales. (a) exact solution mapped on the surface of a missile; (b) numerical error obtained at $h = 0.5$; (c) numerical error obtained at $h = 0.25$; (d) numerical error obtained at $h = 0.125$.

Table 6
Convergence test on missile interface shown in Fig. 13

h	Case 3(a)			Case 3(b)		
	L_∞	E_1	Order	L_∞	E_1	Order
0.5	1.63e−2	6.01e−3		1.64e−1	6.01e−2	
0.25	4.80e−3	2.52e−3	1.8	4.95e−2	2.62e−2	1.7
0.125	1.19e−3	4.44e−4	2.0	1.19e−2	4.44e−3	2.1

The exact solutions of Case 3(a) and Case 3(b) are respectively given by Eqs. (31) and (29), with coefficients defined in Eq. (30).

- Case 4(b): Cylinder of height 2π and diameter π. The computational domain is set to $[-4,4]\times[-4,4]\times[-2,8.4]$.
- Case 4(c): Oak acorn

$$\left(\frac{x}{d}\right)^2+\left(\frac{y}{d}\right)^2=(z-q)^2 \quad \text{if } z>0 \tag{33}$$

and

$$x^2+y^2+(z-g)^2=R^2 \quad \text{if } z\leqslant 0, \tag{34}$$

where $q=-\frac{6}{7}$, $g=\frac{1}{2}$, $R=\frac{15}{7}$ and $d=\sqrt{\frac{R^2-g^2}{q^2}}$. The computational domain is set to $[-5,5]\times[-5,5]\times[-5,5]$. We set solution as

$$u^-=10\cos kx\cos ky\cos kz+20, \quad u^+=10(x+y+z)+1, \tag{35}$$

with coefficients prescribed by Eq. (30). These three interfaces are depicted in Fig. 14 with surface maps of exact solutions and numerical errors computed at $k=3$. The cylinder has two corners and the oak acorn has an edge and a tip. However, these singularities are quite mild. The solution becomes oscillatory when a large k is used. As in earlier cases, there is a good correlation between the exact solutions and the numerical errors, indicating that the extrema of the surface errors are induced by the large amplitude of the solutions, rather than by the geometric features.

Table 7 shows the numerical errors given by the second- and fourth-order MIB schemes. It can be seen that the designed second- and fourth-order convergences are achieved in all the test cases. Under the same mesh size, the numerical errors obtained by the fourth-order method are up to 1000 times smaller than those obtained by the second-order method.

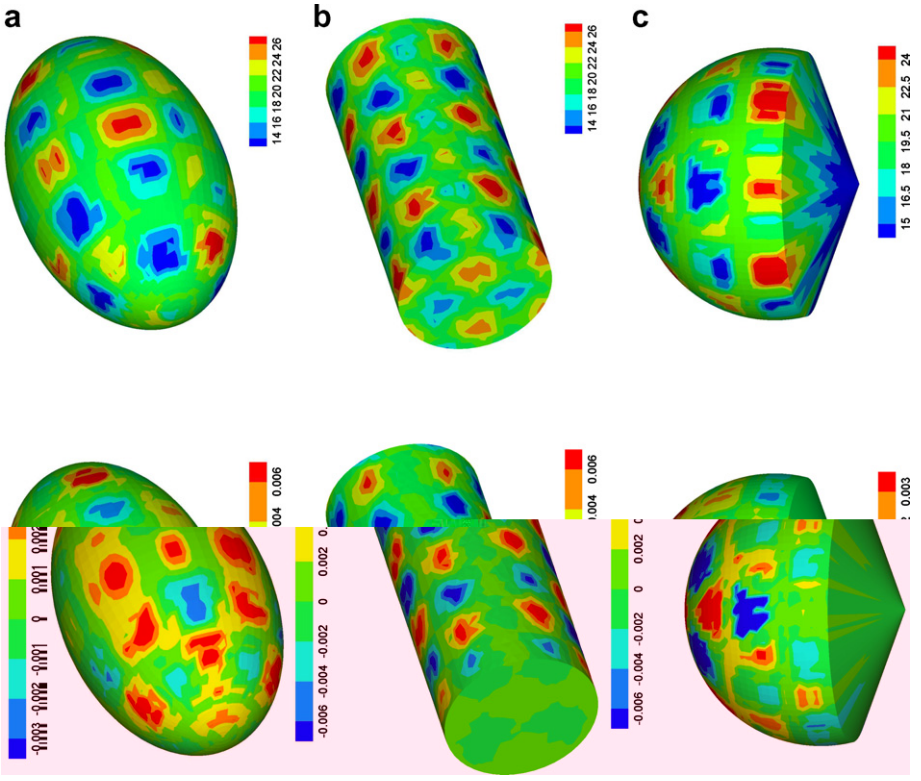

Fig. 14. The surface maps of exact solutions (Top row) and numerical errors (Bottom row). Numerical errors are obtained at $h=0.2$ and represented in seven scales. (a) Ellipsoid; (b) Cylinder; (c) Oak acorn.

Table 7
Convergence test of the fourth-order MIB scheme (Case 4)

		h	Second-order			Fourth-order		
			L_∞	E_1	Order	L_∞	E_1	Order
Ellipsoid	$k=1$	0.4	1.25e−1	2.30e−2		2.23e−3	2.23e−3	
		0.2	3.42e−2	3.54e−3	1.9	1.44e−4	1.13e−4	4.0
		0.1	8.70e−3	5.70e−4	2.0	1.04e−5	3.50e−6	3.8
	$k=3$	0.4	1.02e−0	8.29e−1		3.18e−1	3.18e−1	
		0.2	3.32e−1	1.01e−1	1.6	1.87e−2	1.87e−2	4.1
		0.1	8.45e−2	1.78e−2	2.0	1.06e−3	7.01e−4	4.1
Cylinder	$k=1$	0.4	1.13e−1	3.85e−2		2.11e−3	6.17e−4	
		0.2	3.01e−2	6.28e−3	1.9	1.28e−4	7.56e−5	4.0
		0.1	7.68e−3	5.68e−4	2.0	9.01e−6	2.79e−6	3.8
	$k=3$	0.4	1.08e−0	9.48e−1		2.57e−1	2.57e−1	
		0.2	3.17e−1	1.55e−1	1.8	1.71e−2	1.71e−2	3.9
		0.1	7.97e−2	2.50e−2	2.0	8.95e−4	4.77e−4	4.3
Acorn	$k=1$	0.2	2.24e−2	7.59e−3		1.05e−4	7.32e−5	
		0.1	5.69e−3	1.07e−3	2.0	6.84e−6	2.33e−6	3.9
		0.05	1.43e−3	1.56e−4	2.0	4.42e−7	1.21e−7	4.0
	$k=3$	0.2	3.10e−1	1.68e−1		1.51e−2	1.06e−2	
		0.1	7.81e−2	2.57e−2	2.0	9.59e−4	4.74e−4	4.0
		0.05	2.01e−2	3.52e−3	2.0	5.94e−5	2.22e−5	4.0

Three interfaces are shown in Fig. 14. The exact solutions are given by Eqs. (35) and (30).

It remains to know whether the proposed MIB method delivers fourth-order accuracy for more complex interface geometry and interfaces with more challenging geometric singularities. To this end, we consider two more test cases, an apple and a flower.

- Case 4(d): Apple

$$\rho = 1.9(1-\cos\phi), \tag{36}$$

where $\rho = \sqrt{x^2+y^2+z^2}$ and $\phi = \cos^{-1}\left(\frac{z}{\rho}\right)$. The computational domain is set to $[-5,5]\times[-5,5]\times[-8,4]$.

- Case 4(e): Flower

$$r = \frac{5}{2} + \frac{5}{7}\sin 5\theta, \quad -\frac{2}{3} \leqslant z \leqslant \frac{2}{3}, \tag{37}$$

where $r = \sqrt{x^2+y^2}$ and $\theta = \tan^{-1}\left(\frac{y}{x}\right)$. The computational domain is set to $[-5,5]\times[-5,5]\times[-2,2]$.

We adopt Eqs. (29) and (30) for exact solutions and parameters, respectively. These two interfaces are depicted in Fig. 15 with surface maps of exact solutions and numerical errors. The apple interface admits a singularity at the origin which is computationally challenging. The flower interface is complex and non-smooth. A 2D interface that is similar to the flower shape has been considered in our previous work [74] to test our 2D fourth-order MIB scheme for complex interface geometry.[1] Table 8 gives the convergence studies of the proposed MIB method for these two cases. Due to complexity and large curvature of the flower interface, a relatively dense grid is required to ensure the availability of auxiliary points and thus the designed fourth-order accuracy. In both cases, the designed fourth-order convergence is achieved. For the apple interface, L_∞ and E_1 are the identical at $h = 0.1$, indicating that the largest absolute error occurs at the origin i.e., the singular point. For the flower interface, large errors occur where there are large curvatures. It is to point out that the construction of 3D fourth-order interface methods for arbitrarily complex interface with geometric singularities is still an open problem.

[1] In Ref. [74], the interface of Example 5 should be $r = \frac{1}{2} + \frac{\sin(5\theta)}{7}$.

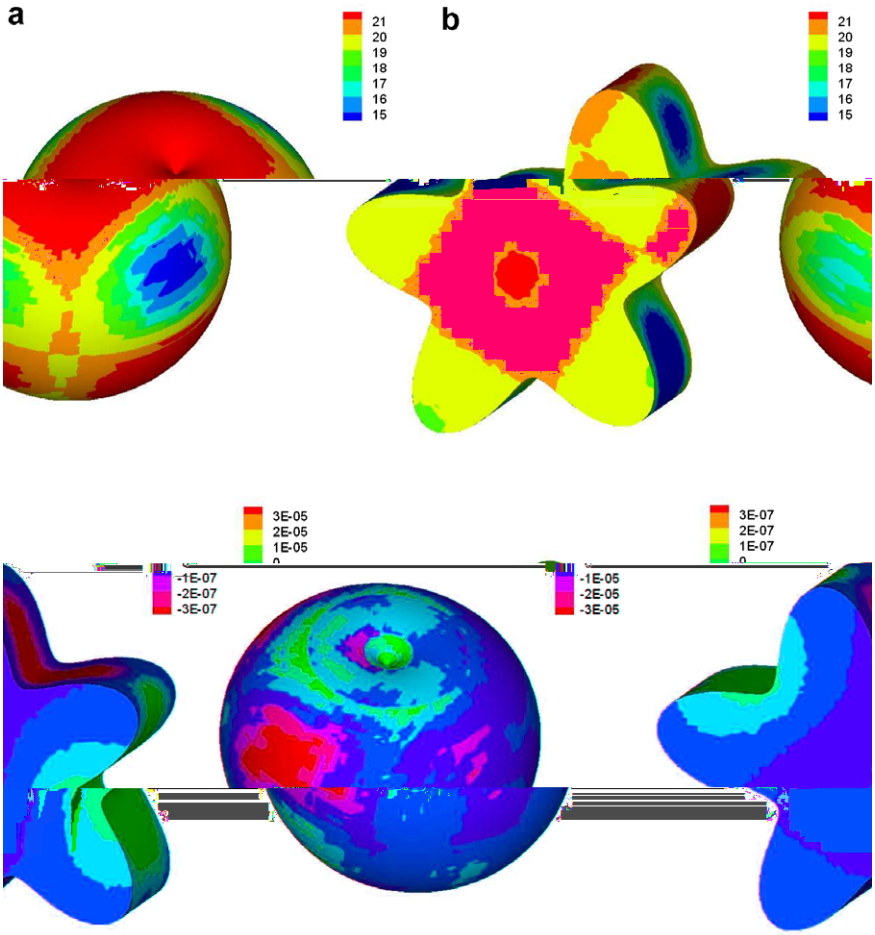

Fig. 15. The surface maps of exact solutions (Top row) and numerical errors (Bottom row). Numerical errors are obtained at $h = 0.2$ for the apple and at $h = 0.1$ for the flower. All images are plotted in seven scales. (a) Apple; (b) Flower.

Table 8
Convergence test of the fourth-order MIB scheme (Case 4(d) and (e)). Two interfaces are shown in Fig. 15

	h	L_∞	E_1	Order
Apple	0.4	9.83e−3	9.83e−3	
	0.2	1.00e−4	8.11e−5	6.6
	0.1	7.15e−6	7.15e−6	3.8
Flower	0.1	1.30e−5	1.30e−5	
	0.05	5.73e−7	5.73e−7	4.5
	0.025	1.79e−8	6.48e−9	5.0

The exact solutions are given by Eqs. (29) and (30).

3.5. Case 5: Variable diffusion coefficients

Our previous 2D MIB method has been tested for variable diffusion coefficients and for large contrast in diffusion coefficients. Here, we show that the present 3D MIB method works well for variable diffusion coefficients, which are piecewise smooth function of positions. In order to test the performance of the MIB method on different $\beta(x, y, z)$, the following diffusion coefficients are used together with the solution given in Eq. (35)

$$\beta^- = \frac{1}{2}(x+y) + 10, \quad \beta^+ = z + 15, \tag{38}$$

where we set $\kappa = 0$.

The interfaces of ellipsoid and acron shown in Fig. 14a and c are employed. The computational domains are set to $[-5,5]\times[-5,5]\times[-5,5]$ for both test cases. The numerical errors are given in Table 9. The computational results are of designed L_∞ order of convergence. The magnitudes of numerical errors are very similar to those in Table 7, indicating that the proposed method is robust against variable coefficients.

3.6. Case 6: Non-zero linear term

It is very important to include non-zero $\kappa(x,y,z)$ in many practical problems. The performance of the present MIB method is tested in the following two cases

- Case 6(a):

$$\kappa^-(x,y,z) = 10\exp(-x), \quad \kappa^+(x,y,z) = 20y^2. \tag{39}$$

- Case 6(b):

$$\kappa^-(x,y,z) = 1000\exp(-x), \quad \kappa^+(x,y,z) = 2000y^2. \tag{40}$$

The exact solution $u(x,y,z)$ is given in Eq. (35) with $k = 1$ and the diffusion coefficient $\beta(x,y,z)$ is given in Eq. (30). The source term $q(x,y,z)$ and jump conditions can therefore be easily derived. The second- and fourth-order MIB schemes are applied to solve these two cases with the acorn interface given in Eq. (33). The computational domain is set to $[-5,5]\times[-5,5]\times[-5,5]$.

Table 10 shows that the present MIB scheme achieves the designated order of convergence in both cases. It is interesting to note that when κ is increased 100 times from Case 6(a) to Case 6(b), the MIB method converges in Case 6(b) twice as fast as it does in Case 6(a), which is due to the fact that the matrix in Case 6(b) is more banded.

3.7. Case 7: Sixth-order MIB scheme

Finally, we test the convergence property of our sixth-order MIB scheme. Three different interfaces, a sphere, an ellipsoid and a diatomic molecule are considered. Different solutions are prescribed for these interfaces to further validate the present MIB method.

Table 9
Convergence test on variable diffusion coefficients (Case 5)

		h	Second-order			Fourth-order		
			L_∞	E_1	Order	L_∞	E_1	Order
Ellipsoid	$k=1$	0.4	2.00e−1	1.52e−1		1.76e−2	1.76e−2	
		0.2	4.76e−2	3.02e−2	2.1	1.03e−3	1.03e−3	4.1
		0.1	1.14e−2	5.25e−3	2.1	4.92e−5	4.92e−5	4.4
	$k=3$	0.4	2.39e−0	2.39e−0		1.03e−0	1.03e−0	
		0.2	4.57e−1	4.13e−1	2.4	1.25e−1	1.25e−1	3.0
		0.1	9.22e−2	6.87e−2	2.3	5.19e−3	5.19e−3	4.6
Acorn	$k=1$	0.2	3.79e−2	2.70e−2		4.82e−4	4.82e−4	
		0.1	9.17e−3	5.81e−3	2.1	2.52e−5	2.52e−5	4.3
		0.05	2.32e−3	1.39e−3	2.0	1.82e−6	1.82e−6	3.8
	$k=3$	0.2	4.01e−1	3.10e−1		7.09e−2	7.09e−2	
		0.1	9.74e−2	6.27e−2	2.0	3.67e−3	3.67e−3	4.3
		0.05	2.43e−2	1.27e−2	2.0	2.00e−4	2.00e−4	4.2

Two interfaces are shown in Fig. 14a and c. The exact solutions are given by Eqs. (35) and (38).

Table 10
Convergence test on the non-zero linear term with the acorn interface shown in Fig. 14c

	h	Second-order				Fourth-order			
		L_∞	E_1	N_{BiCG}	Order	L_∞	E_1	N_{BiCG}	Order
Case 6(a)	0.2	7.99e−3	3.28e−3	243		5.98e−5	5.98e−5	283	
	0.1	2.02e−3	4.72e−4	478	2.0	2.55e−6	2.34e−6	592	4.6
	0.05	5.07e−4	7.79e−5	969	2.0	1.62e−7	1.19e−7	1278	4.0
Case 6(b)	0.2	1.27e−3	1.27e−3	92		1.52e−5	1.52e−5	114	
	0.1	3.08e−4	3.08e−4	181	2.0	1.37e−6	1.37e−6	239	3.5
	0.05	5.14e−5	5.14e−5	389	2.6	7.98e−8	7.98e−8	518	4.1

The exact solutions are given by Eqs. (35) and (30) with $\kappa(x,y,z)$ redefined by Eqs. (39) and (40) for Case 6(a) and Case 6(b), respectively.

- Case 7(a): Spherical interface. The interface is a sphere of radius 2. The diffusion coefficients are given by Eq. (30). The solutions are given as

$$u^- = \cos kx \cos ky \cos kz, \quad u^+ = 0. \tag{41}$$

  The computational domain is set to $[-5,5]\times[-5,5]\times[-5,5]$.
- Case 7(b): Ellipsoidal interface. The ellipsoidal interface is given by Eq. (32). The exact solution and variable diffusion coefficients are given as the following:

$$u^- = \cos kx \cos ky \cos kz, \quad u^+ = \exp\left(-\frac{x^2+y^2+z^2}{20}\right), \tag{42}$$

$$\beta^- = (x+y)/2 + 10, \quad \beta^+ = z + 15, \quad \kappa = 0. \tag{43}$$

  The computational domain is set to $[-5,5]\times[-5,5]\times[-5,5]$.
- Case 7(c): Molecular surface of a diatom. This interface is defined by the molecular surface of a diatom, shown in Fig. 9. The exact solutions are given by Eqs. (35) and (30) with $k = 3$ (Note that these exact solutions are not the ones shown in Fig. 9a). Unlike two other cases, this interface contains concave surface curves.The computational domain is set to $[-6,6.16]\times[-4,4.96]\times[-4,4]$.

Table 11 provides the results of convergence tests for Case 7(a). It is seen that the designed L_∞ orders of convergence are achieved in these MIB schemes. The sixth-order method is up to 10^7 times more accurate that the second one, demonstrating the advantage of higher-order schemes. The L_∞ and E_1 errors in the sixth-order scheme are identical, indicating that maximum errors are generated from the interface.

The convergence results for Case 7(b) are listed Table 12. Similar to the last case, the designed orders of convergence are numerically confirmed for all MIB schemes. It is noted that L_∞ and E_1 errors become identical in both fourth- and sixth-order schemes. Therefore, the interface errors are the major error source in this case.

Table 11
Convergence comparison of the second-, fourth- and sixth-order MIB schemes on a spherical interface (Case 7(a))

	h	Second-order			Fourth-order			Sixth-order		
		L_∞	E_1	Order	L_∞	E_1	Order	L_∞	E_1	Order
$k=1$	0.2	3.5e−3	2.0e−4		1.5e−5	1.5e−5		5.3e−7	5.3e−7	
	0.1	9.0e−4	2.7e−5	2.0	1.0e−6	6.1e−7	3.8	5.4e−9	5.4e−9	6.6
	0.05	2.3e−4	3.8e−6	2.0	7.0e−8	2.4e−8	3.9	5.3e−11	5.3e−11	6.7
$k=3$	0.2	3.3e−2	9.8e−3		2.8e−3	2.8e−3		7.7e−4	7.7e−4	
	0.1	8.1e−3	1.6e−3	2.0	1.0e−4	1.0e−4	4.8	8.3e−6	8.3e−6	6.6
	0.05	2.0e−3	2.6e−4	2.0	5.8e−6	3.2e−6	4.1	5.8e−8	5.8e−8	7.2

The exact solutions are given by Eqs. (41) and (30).

Table 12
Convergence comparison of the second-, fourth- and sixth-order MIB schemes on an ellipsoid interface (Case 7(b)) as shown in Fig. 14a

	h	Second-order			Fourth-order			Sixth-order		
		L_∞	E_1	Order	L_∞	E_1	Order	L_∞	E_1	Order
$k=1$	0.2	4.9e−3	3.2e−3		9.9e−5	9.9e−5		3.5e−6	3.5e−6	
	0.1	1.2e−3	5.7e−4	2.1	4.7e−6	4.7e−6	4.4	4.2e−8	4.2e−8	6.4
	0.05	2.9e−4	1.0e−4	2.1	2.2e−7	2.2e−7	4.4	5.1e−10	5.1e−10	6.4
$k=3$	0.2	4.6e−2	4.1e−2		1.3e−2	1.3e−2		2.7e−3	2.7e−3	
	0.1	9.2e−3	6.8e−3	2.3	5.2e−4	5.2e−4	4.6	3.7e−5	3.7e−5	6.2
	0.05	2.0e−3	1.5e−3	2.2	2.1e−5	2.1e−5	4.7	3.5e−7	3.5e−7	6.7

The exact solutions are given by Eqs. (42) and (43).

Table 13
Convergence comparison of the second-, fourth- and sixth-order MIB schemes on a diatomic interface (Case 7(c)) as shown in Fig. 9

Scheme	h	L_∞	E_1	Order	N_{BiCG}
Second-order	0.16	2.11e−1	8.27e−2		296
	0.08	5.32e−2	1.08e−2	2.0	529
	0.04	1.34e−2	2.34e−3	2.0	2072
Fourth-order	0.16	1.12e−2	1.12e−2		329
	0.08	4.14e−4	2.04e−4	4.8	670
	0.04	2.62e−5	1.05e−5	4.0	2613
Sixth-order	0.16	5.20e−3	5.20e−3		409
	0.08	2.56e−5	2.56e−5	7.7	824
	0.04	1.13e−6	1.13e−6	4.5	3575

The exact solutions are given by Eqs. (35) and (30) with $k=3$.

Finally, Table 13 shows the comparison of three MIB schemes for the molecular surface of the diatom studied in Case 1. This problem is more difficult than the last two test cases because of the presence of concave curves in the interface. The mesh sizes are chosen slightly smaller than those used in the earlier examples to avoid the difficulty of finding insufficient number of auxiliary grid points. The number of iteration N_{BiCG} is also provided for each computation. It is seen that the fourth- and sixth-order MIB schemes require slightly larger N_{BiCG} than the second-order scheme does, because more auxiliary points are involved to represent a fictitious value in higher-order schemes. Under the same mesh size h, the CPU times required for all iterations are about the same since the size of the matrix is the same for a given h. The major factor that determines N_{BiCG} is the matrix size, instead of the order of the scheme. Therefore, the proposed higher-order MIB schemes do not cause additional convergence problems. At $h=0.16$, the sixth-order scheme delivers the same level of accuracy as the second-order scheme does at $h=0.04$, indicating the efficiency of higher-order methods. Similar to the last two cases, the L_∞ and E_1 errors in the sixth-order scheme are identical. This study suggests that the construction of sixth-order 3D interface methods for elliptic equations with arbitrarily curved smooth interfaces remains an open problem.

4. Conclusion

This paper introduces the three-dimensional (3D) higher-order matched interface and boundary (MIB) method for solving elliptic equations with discontinuous coefficients and geometric singularities. Our previous higher-order MIB methods in 1D [71] and 2D [73,74] have been generalized to 3D. The present MIB method is of fourth-order convergence for complex interfaces with moderate geometric singularities, and of six-order convergence for smooth interfaces. Moreover, a recently proposed 2D second-order MIB scheme [69] for solving elliptic equations with discontinuous coefficients and interface singularities is extended to a 3D MIB scheme of up to fourth-order convergence for interface singularities. Non-smooth interfaces are notoriously challenging and the best result in the literature is of 0.8th-order convergence due to Hou and Liu [26]. New

algorithms, particularly the use of jump conditions at two interface intersecting points, are proposed to effectively remove the critical acute angle restriction of our earlier MIB scheme [69]. The resulting MIB schemes are able to attain second-order convergence for arbitrarily complex interfaces with sharp edges, wedges, and tips. Furthermore, we propose a systematical procedure to make the MIB matrix optimally symmetric and banded. Consequently, our new MIB algorithm requires a fewer number of iterations to converge by using a standard preconditioned biconjugate gradient solver. The proposed 3D MIB schemes are extensively validated in terms of the speed of convergence, the order of accuracy, the number of iterations and CPU time. A large variety of numerical examples, including a protein molecular surface, a missile interface, and systems of multiple intersecting spheres, are utilized to validate the proposed MIB method. First known results are obtained for 3D elliptic equations in the following three categories: solutions of second-order accuracy for arbitrarily complex interfaces with sharp geometric singularities; solutions of fourth-order accuracy for complex interfaces with moderate singularities; and solutions of sixth-order accuracy for smooth curved interfaces. Some new open problems, such the construction of fourth-order 3D interface schemes for arbitrarily complex interfaces with sharp geometric singularities and the construction of sixth-order 3D interface schemes for arbitrarily curved smooth interfaces, are identified.

In summary, the main ideas of the present MIB method are follows. First, simple Cartesian grids are used to avoid the time consuming grid generation even if the interface is complex and irregular. Problems that are defined on complex geometric domains can be embedded in rectangular domains as interface problems. Second, the standard (higher-order) central finite difference (FD) schemes are used for the discretization of the elliptic equations in the entire domain, including the interface region. Consequently, the condition number of the discretization matrix is relatively low, and efficient linear algebraic equation solvers can be used. Moreover, MIB schemes automatically reduce to the central FD ones when there is no interface. Third, when the FD scheme reaches across the interface, fictitious values are used to ensure that the numerical derivative is locally computed on a ''smooth function'' near the interface, although the true solution is either discontinuous or continuous but not differentiable. Therefore, fictitious values are the smooth continuation of the non-smooth function across the interface. Fourth, the fictitious values are determined by using physical jump conditions that are prescribed to make the original elliptic problem well posed. Fifth, physical jump conditions are enforced on-interface. As such, subgrid information, i.e., the solution jump and the flux jump, is built in the interface scheme. As a result, the well-known staircase phenomenon of the FD scheme for irregular domains is eliminated. Sixth, only the lowest order jump conditions are used to avoid the involvement of higher-order derivatives, particularly the cross-derivative, and numerical instability. Jump conditions are discretized by using the standard Lagrange polynomials [17]. Seventh, to construct higher-order schemes, the lowest order jump conditions are utilized repeatedly. Eighth, although there are four available jump conditions in 3D, only two fictitious values on a mesh line are determined at a time to avoid the calculation of two most difficult partial derivatives at the interface. Consequently, a 2D or a 3D interface problem is locally reduced into 1D-like ones. Ninth, although the MIB matrix is non-symmetric because of the use of auxiliary points for calculating on-interface off-grid partial derivatives in the jump conditions, the flexibility in the auxiliary point selection is utilized to make the MIB matrix optimally symmetric and banded. Tenth, a wide variety of techniques are proposed in the MIB method to determine fictitious values [69,71,73,74], such as computing fictitious values from a different direction, using two sets of interface jump conditions, utilizing secondary fictitious values to evaluate primary fictitious values, etc. Finally, an interpolation formulation of the MIB method was proposed to avoid the use of fictitious values or fictitious subdomains. We have shown that such an approach is equivalent to our fictitious domain formulation [74]. It is believed that all results presented in the present paper can be obtained by using our interpolation formulation as well. The application of the present MIB method to the electrostatic analysis of biomolecules is reported elsewhere [70]. MIB based approaches for the simulation of electromagnetic wave propagation and scattering, and for the computational fluid dynamics are under our consideration.

Acknowledgements

This work was supported in part by NSF Grants DMS-0616704 and IIS-0430987. The authors thank Shan Zhao, Yongcheng Zhou and Weihua Geng for useful discussions.

References

[1] L. Adams, Z.L. Li, The immersed interface/multigrid methods for interface problems, SIAM J. Sci. Comput. 24 (2002) 463–479.
[2] I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, Computing 5 (1970) 207–213.
[3] I. Babuška, S.A. Sauter, Is the pollution effect of the FEM avoidable for the Helmholtz equation considering high wave number? SIAM J. Numer. Anal. 34 (1997) 2392–2423.
[4] F. Baetke, H. Werner, H. Wengle, Numerical-simulation of turbulent-flow over surface-mounted obstacles with sharp edges and corners, J. Wind Engng. Indust. Aerodyn. 35 (1990) 129–147.
[5] H. Ben Ameur, M. Burger, B. Hackl, Level set methods for geometric inverse problems in linear elasticity, Inverse Probl. 20 (2004) 673–696.
[6] P.A. Berthelsen, A decomposed immersed interface method for variable coefficient elliptic equations with non-smooth and discontinuous solutions, J. Comput. Phys. 197 (2004) 364–386.
[7] G. Biros, L.X. Ying, D. Zorin, A fast solver for the Stokes equations with distributed forces in complex geometries, J. Comput. Phys. 193 (2004) 317–348.
[8] 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.
[9] 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.
[10] S. Caorsi, M. Pastorino, M. Raffetto, Electromagnetic scattering by a conducting strip with a multilayer elliptic dielectric coating, IEEE T. Electromag. Compat. 41 (1999) 335–343.
[11] Z.J. Cendes, D.N. Shenton, H. Shahnasser, Magnetic field computation using Delaney triangulation and complementary finite element methods, IEEE Ttans. Magn. 19 (1983) 2251–2554.
[12] S.Z. Deng, K. Ito, Z.L. Li, Three-dimensional elliptic solvers for interface problems and applications, J. Comput. Phys. 184 (2003) 215–243.
[13] 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 (2003) 348–367.
[14] 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 (2000) 30–60.
[15] 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.
[16] A.L. Fogelson, J.P. Keener, Immersed interface methods for Neumann and related problems in two and three dimensions, SIAM J. Sci. Comput. 22 (2000) 1630–1654.
[17] B. Fornberg, Calculation of weights in finite difference formulas, SIAM Rev. 40 (1998) 685–691.
[18] M. Francois, W. Shyy, Computations of drop dynamics with the immersed boundary method, part 2: Drop impact and heat transfer, Numer. Heat Trans. Part B – Fund. 44 (2003) 119–143.
[19] M. Francois, E. Uzgoren, J. Jackson, W. Shyy, Multigrid computations with the immersed boundary technique for multiphase flows, Int. J. Numer. Methods Heat Fluid Flow 14 (2004) 98–115.
[20] W.H. Geng, S.N. Yu, G.W. Wei, Treatment of charge singularities in the implicit solvent models, J. Chem. Phys. 127 (2007) 114106.
[21] F. Gibou, R.P. Fedkiw, A forth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, J. Comput. Phys. 202 (2005) 577–601.
[22] 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.
[23] 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.
[24] J.S. Hesthaven, High-order accurate methods in time-domain computational electromagnetics. A review, Adv. Imaging Electron Phys. 127 (2003) 59–123.
[25] T.P. Horikis, W.L. Kath, Modal analysis of circular Bragg fibers with arbitrary index profiles, Optics Lett. 31 (2006) 3417–3419.
[26] S.M. Hou, X.-D. Liu, A numerical method for solving variable coefficient elliptic equation with interfaces, J. Comput. Phys. 202 (2005) 411–445.
[27] T.Y. Hou, Z.L. Li, S. Osher, H. Zhao, A hybrid method for moving interface problems with application to the Hele–Shaw flow, J. Comput. Phys. 134 (1997) 236–252.
[28] H. Huang, Z.L. Li, Convergence analysis of the immersed interface method, IMA J. Numer. Anal. 19 (1999) 583–608.
[29] J.K. Hunter, Z.L. Li, H. Zhao, Reactive autophobic spreading of drops, J. Comput. Phys. 183 (2002) 335–366.
[30] G. Iaccarino, R. Verzicco, Immersed boundary technique for turbulent flow simulations, Appl. Mech. Rev. 56 (2003) 331–347.
[31] S. Jin, X.L. Wang, Robust numerical simulation of porosity evolution in chemical vapor infiltration II. Two-dimensional anisotropic fronts, J. Comput. Phys. 179 (2002) 557–577.
[32] H. Johansen, P. Colella, A Cartesian grid embedding boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85.
[33] R. Kafafy, T. Lin, Y. Lin, J. Wang, Three-dimensional immersed finite element methods for electric field simulation in composite materials, Int. J. Numer. Methods Eng. 64 (2005) 940–972.
[34] J.D. Kandilarov, Immersed interface method for a reaction–diffusion equation with a moving own concentrated source, Lecture Notes Comput. Sci. 2542 (2003) 506–513.

[35] 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.
[36] L. Lee, R.J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (2003) 832–856.
[37] 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.
[38] Z.L. Li, A fast iterative algorithm for elliptic interface problems, SIAM J. Numer. Anal. 35 (1998) 230–254.
[39] Z.L. Li, K. Ito, Maximum principle preserving schemes for interface problems with discontinuous coefficients, SIAM J. Sci. Comput. 23 (2001) 339–361.
[40] Z.L. Li, T. Lin, X.H. Wu, New Cartesian grid methods for interface problems using the finite element formulation, Numer. Math. 96 (2003) 61–98.
[41] Z.L. Li, S.R. Lubkin, Numerical analysis of interfacial two-dimensional Stokes flow with discontinuous viscosity and variable surface tension, Int. J. Numer. Methods Fluid 37 (2001) 525–540.
[42] Z.L. Li, W.-C. Wang, I.-L. Chern, M.-C. Lai, New formulations for interface problems in polar coordinates, SIAM J. Sci. Comput. 25 (2003) 224–245.
[43] M.N. Linnick, H.F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, J. Comput. Phys. 204 (2005) 157–192.
[44] 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. Eng. 195 (2006) 1722–1749.
[45] 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.
[46] B. Lombard, J. Piraux, How to incorporate the spring-mass conditions in finite-difference schemes, SIAM J. Sci. Comput. 24 (2003) 1379–1407.
[47] E.B. Macak, W.D. Munz, J.M. Rodenburg, Plasma-surface interaction at sharp edges and corners during ion-assisted physical vapor deposition. Part I: Edge-related effects and their influence on coating morphology and composition, J. Appl. Phys. 94 (2003) 2829–2836.
[48] A.D. MacKerell Jr., D. Bashford, M. Bellott, J.D. Dunbrack, M.J. Evanseck, M.J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F.T.K. Lau, C. Mattos, S. Michnick, T. Ngo, D.T. Nguyen, B. Prodhom, W.E. Reiher, B. Roux, M. Schlenkrich, J.C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiorkiewicz-Kuczera, D. Yin, M. Karplus, All-atom empirical potential for molecular modeling and dynamics studies of proteins, J. Phys. Chem. 102 (1998) 3586–3616.
[49] A. Mayo, The fast solution of Poisson's and the biharmonic equations on irregular regions, SIAM J. Numer. Anal. 21 (1984) 285–299.
[50] A. Mayo, A. Greenbaum, Fourth order accurate evaluation of integrals in potential theory on exterior 3D regions, J. Comput. Phys. 220 (2007) 900–914.
[51] A. McKenney, L. Greengard, A. Mayo, A fast Poisson solver for complex geometries, J. Comput. Phys. 118 (1995) 348–355.
[52] R. Miniowitz, J.P. Webb, Covariant-projection quadrilateral elements for the analysis of wave-guides with sharp edges, IEEE Trans. Microwave Theory Techn. 39 (1991) 501–505.
[53] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 236–261.
[54] 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.
[55] G. Pan, M. Tong, B. Gilbert, Multiwavelet based moment method under discrete Sobolev-type norm, Microwave Opt. Techn. Lett. 40 (2004) 47–50.
[56] Z. Pantic-Tanner, J.Z. Savage, D.R. Tanner, A.F. Peterson, Two-dimensional singular vector elements for finite-element analysis, IEEE Trans. Microwave Theory Techn. 46 (1998) 178–184.
[57] C.S. Peskin, Numerical analysis of blood flow in heart, J. Comput. Phys. 25 (1977) 220–252.
[58] C.S. Peskin, Lectures on mathematical aspects of physiology, Lectures Appl. Math. 19 (1981) 69–107.
[59] C.S. Peskin, D.M. McQueen, A 3-dimensional computational method for blood-flow in the heart. 1. Immersed elastic fibers in a viscous incompressible fluid, J. Comput. Phys. 81 (1989) 372–405.
[60] M.F. Sanner, A.J. Olson, J.C. Spehner, Reduced surface: an efficient way to compute molecular surfaces, Biopolymers 38 (1996) 305–320.
[61] M. Schulz, G. Steinebach, Two-dimensional modelling of the river Rhine, J. Comput. Appl. Math. 145 (2002) 11–20.
[62] J.A. Sethian, A. Wiegmann, Structural boundary design via level set and immersed interface methods, J. Comput. Phys. 163 (2000) 489–528.
[63] A.K. Tornberg, B. Engquist, Numerical approximations of singular source terms in differential equations, J. Comput. Phys. 200 (2004) 462–488.
[64] B.J.E. van Rens, W.A.M. Brekelmans, F.P.T. Baaijens, Modelling friction near sharp edges using a Eulerian reference frame: application to aluminum extrusion, Int. J. Numer. Methods Engng. 54 (2002) 453–471.
[65] J.V. Voorde, J. Vierendeels, E. Dick, Flow simulations in rotary volumetric pumps and compressors with the fictitious domain method, J. Comput. Appl. Math. 168 (2004) 491–499.
[66] J.H. Walther, G. Morgenthal, An immersed interface method for the vortex-in-cell algorithm, J. Turbul. 3 (2002). Art. No. 039.
[67] Z.J. Wang, K. Srinivasan, An adaptive Cartesian grid generation method for 'dirty' geometry, Int. J. Numer. Methods Fluids 39 (2002) 703–717.

[68] 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 (2000) 827–862.
[69] 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 (2007) 729–756.
[70] S.N. Yu, W.H. Geng, G.W. Wei, Treatment of geometric singularities in the implicit solvent models, J. Chem. Phys. 126 (2007) 244108.
[71] S. Zhao, G.W. Wei, High order FDTD methods via derivative matching for Maxwell's equations with material interfaces, J. Comput. Phys. 200 (2004) 60–103.
[72] S. Zhao, G.W. Wei, Y. Xiang, DSC analysis of free-edged beams by an iteratively matched boundary method, J. Sound Vib. 284 (2005) 487–493.
[73] Y.C. Zhou, S. Zhao, M. Feig, G.W. Wei, High order matched interface and boundary (MIB) schemes for elliptic equations with discontinuous coefficients and singular sources, J. Comput. Phys. 213 (2006) 1–30.
[74] 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 (2006) 228–246.
[75] Y.C. Zhou, M. Feig, G.W. Wei, Highly accurate biomolecular electrostatics in continuum dielectric environments, J. Comput. Chem., in press.