Journal of Computational Physics 448 (2022) 110762

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A FFT accelerated high order finite difference method for elliptic boundary value problems over irregular domains

Yiming Ren, Hongsong Feng, Shan Zhao *

Department of Mathematics, University of Alabama, Tuscaloosa, AL 35487, USA

ARTICLE INFO

Article history:
Available online 6 October 2021

Keywords:
Elliptic boundary value problem
Irregular domains
Mixed boundary conditions
Fast Fourier transform (FFT)
High order central difference schemes
Matched interface and boundary (MIB) method

ABSTRACT

For elliptic boundary value problems (BVPs) involving irregular domains and Robin boundary condition, no numerical method is known to deliver a fourth order convergence and $O(N \log N)$ efficiency, where N stands for the total degree-of-freedom of the system. Based on the matched interface and boundary (MIB) and fast Fourier transform (FFT) schemes, a new finite difference method is introduced for such problems, which involves two main components. First, a ray-casting MIB scheme is proposed to handle different types of boundary conditions, including Dirichlet, Neumann, Robin, and their mix combinations. By enclosing the concerned irregular domain by a large enough cubic domain, the ray-casting MIB scheme generates necessary fictitious values outside the irregular domain by imposing boundary conditions along the normal direction of the boundary, so that a high order central difference discretization of the Laplacian can be formed. Second, an augmented MIB formulation is built, in which Cartesian derivative jumps are reconstructed on the boundary as auxiliary variables. By treating such variables as unknowns, the discrete Laplacian can be efficiently inverted by the FFT algorithm, in the Schur complement solution of the augmented system. The accuracy and efficiency of the proposed augmented MIB method are numerically examined by considering various elliptic BVPs in two and three dimensions. Numerical results indicate that the new algorithm not only achieves a fourth order of accuracy in treating irregular domains and complex boundary conditions, but also maintains the FFT efficiency.

1. Introduction

This paper is dedicated to a *fast* and *high order* numerical scheme for solving elliptic boundary value problems (BVPs) in complex-shaped multi-dimensional domains. The elliptic BVPs are modeled as

$$\Delta u + \kappa u = f(\mathbf{x}), \ \mathbf{x} \in \Omega, \tag{1}$$

where function $u(\mathbf{x})$ together with the corresponding source term $f(\mathbf{x})$ depend on a vector variable $\mathbf{x} = (x_1, x_2, \cdots, x_d)$ for $d = 2$ or $d = 3$ on a complex shaped domain Ω, and the coefficient κ of the reaction term is a constant. The generic boundary conditions imposed on the boundary $\Gamma = \partial\Omega$ take a form as

* Corresponding author.
E-mail address: szhao@ua.edu (S. Zhao).

https://doi.org/10.1016/j.jcp.2021.110762
0021-9991/

$$\alpha u + \beta \frac{\partial u}{\partial n} = \phi \quad \text{on } \Gamma, \tag{2}$$

where $\frac{\partial}{\partial n}$ denotes the outward normal derivative and ϕ is a given function. Equation (2) represents three commonly used boundary conditions, i.e., Dirichlet ($\alpha \neq 0, \beta = 0$), Neumann ($\alpha = 0, \beta \neq 0$), and Robin ($\alpha \neq 0, \beta \neq 0$).

Various methods for solving elliptic problems over irregular domain have been investigated in the past several decades [3,14,15,22,23,29,30,35,37,43]. Motivated by the great success achieved by fast Poisson solvers over rectangular domains [17], many irregular domain methods are also equipped with fast algorithms. In [37], the authors proposed a new fast Poisson solver for irregular, multiply-connected regions with a complexity $O(N \log N + M)$ in algebraic computations, where N is the number of interior grid points and M is the number of points on the boundary. In this integral equation method, numerical solutions for Laplace's equation can be realized via computing volume integrals and evaluating layer potentials, so that the resulting integral equation can be solved by a multipole-accelerated iterative procedure. Reference [29] employed multigrid iteration, along with an adaptive mesh refinement (AMR) procedure for Poisson's equation with Dirichlet boundary conditions. In this method, conservative differencing of the second order fluxes was used on each volume. Thus, their method was second-order accurate, even in a very complicated boundary geometries. Taking advantage of a scheme which captures the boundary condition via easily implemented source terms, Towers introduced a source term method [43] for solving a Poisson equation in the presence of an irregular interface where Dirichlet, Neumann, and Robin or any combination of the boundary conditions can be imposed. A second order accurate discretization of the embedded Poisson problem is implemented after introducing a domain embedding method and a FFT-based fast Poisson solver was carried out to solve the system of equations iteratively. In [3], accelerated by fast multipole acceleration, Askham presented a fast, adaptive Poisson solver for complex two-dimensional geometries. In [15], based on finite difference ghost-cell techniques, a simple discretization solved by a multigrid approach was proposed in arbitrary domains embedded in a regular square grid. An efficient Fourier-based solver for the Poisson problem around an arbitrary geometry in an unbounded 3D domain is presented in [25].

Besides the fast algorithms, the development of high order methods has also received many interests. The high order methods are typically more efficient than second order ones, because accurate results can be realized by using coarse grids. In [23], Gibou devised a finite difference discretization subject to the Dirichlet boundary conditions for the Laplace equation on arbitrary domains. By means of defining ghost values via high order polynomials, third and fourth order of accuracy could be achieved. A review of high order finite difference methods for elliptic and parabolic problems is given in [24]. The spectral methods can also be applied to elliptic PDEs in complex domains by means of a fictitious domain approach. The success lies in a smooth extension of the PDE coefficients and data functions from the original domain to an enlarged rectangular domain and a suitable variational formulation to ensure a smoothly extended solution. For instance, two powerful and efficient spectral methods have been proposed in [27] for problems with smooth and weakly singular exact solutions, in which the order of convergence is only determined by the how smooth the solution is, so that these spectral methods are of arbitrary high order of accuracy in principle. Other effective Cartesian grid methods for elliptic boundary value problems over irregular domains include interface boundary integral method [31], immersed boundary smooth extension [41], the embedded finite difference method with Dirichlet boundary conditions [30].

It becomes more challenging when one wishes to combine fast algorithms with high order discretizations for elliptic BVPs over irregular domains. Because such combinations could further accelerate the computational efficiency, several exceptional studies have been conducted in the literature to advance along this direction. In 1977, a fourth order finite difference scheme for general bounded regions has been introduced in [39] for Laplace's equation with Dirichlet boundary conditions by means of deferred correction to achieve high accuracy. Based on the results obtained by a second order accurate method and fast Poisson solver, only a slightly additional use of computer time is required to attain highly accurate solutions.

In combining fast solvers and high order convergence, one of the major breakthroughs is a FC-AD method [9,10], designed for solving elliptic BVPs subject to the Dirichlet boundary conditions over general smooth domains. The FC-AD method implements the alternating direction implicit (ADI) algorithm as a fast Poisson solver with the operation count being $O(N)$, and combines the Fourier Continuation (FC) method for resolving the Gibbs oscillation. The FC-AD method [11,12] gives rise to an unconditional stability, high order convergence, and $O(N \log N)$ high efficiency, for general domains without resorting to a domain mapping. In fact, fifth order of accuracy could be achieved in unconditionally stable FC-AD solvers for elliptic and parabolic problems over smooth spatial domains and for arbitrarily fine discretizations.

Another fast algorithm, which is also compatible with high order accuracy and the use of a fast Poisson solver in arbitrarily shaped domains, is based on a combination of the Correction Function Method (CFM) and Boundary Integral Method (BIM) [36]. For BVPs with Dirichlet or Neumann boundary conditions, the authors converted the BIM solution as a series of Poisson problems in rectangular domains, whereas these Poisson equations are solved using the CFM with finite differences and a FFT based fast Poisson solver. The numerical solutions could converge with either 3rd or 4th order of accuracy, depending on the characteristics of the problem.

We note that for the approaches discussed above that combine fast Poisson solvers with high order numerical solutions, none of them considered the Robin boundary condition. In fact, the second order implementation of the Robin boundary conditions is already a tough issue for solving elliptic BVPs. Besides the source term method [43] mentioned above, two finite volume schemes have been successfully developed in [2,6,38] to secure a second order solution in solving the Poisson-type equations with Robin boundary condition over complex domains. One scheme results in a symmetric linear system, while the other is nonsymmetric but achieves a second-order accuracy in the L^∞ norm for gradients. Moreover, based on

the level set framework, these finite volume methods can handle irregular domains with piecewise smooth boundaries [6]. To the best of our knowledge, within the context of the irregular domains, high order schemes with fast Poisson solvers have never been reported to handle the Robin boundary conditions.

There exist two related areas, in which various fast and high order Poisson solvers have been successfully developed. First, for elliptic interface problems with a curved interface embedded in a rectangular domain, the multigrid acceleration has been explored in various second order interface algorithms, such as [1,5,13]. The use of the FFT Poisson solver in second order immersed interface method (IIM) was first introduced by Li in 1998, yielding the augmented IIM (AIIM) scheme [33]. By introducing auxiliary variables to form an augmented system, the Laplacian operator can be approximated by the standard finite difference stencil in the AIIM, which results in a symmetric and diagonally dominant matrix for FFT inversion. The AIIM has been generalized to solve variable coefficient elliptic interface problems with multigrid acceleration [34]. The first elliptic interface algorithm that not only achieves a fourth order of accuracy in dealing with interfaces and boundaries, but also maintains the FFT complexity of $O(N \log N)$ is the augmented matched interface and boundary (AMIB) method [18]. Built on an augmented formulation similar to the AIIM method [33,34], the AMIB method [16,18] employs a different type of auxiliary variables and enforces the interface jump conditions in a totally different manner. Second, for elliptic BVPs over rectangular domains, several advanced algorithms have been introduced in the literature [4,7,8,17,21,26,28,32,42,46] to achieve high order or spectral accuracy, as well as to attain high efficiency with the help of FFT or multigrid. In [32], Lai presented an efficient compact fourth-order FFT Poisson solver on polar geometry to produce an accurate solution, up to fourth order for the problem on an annulus and third order for the problem on a disk. For the sake of efficiency and convergence acceleration, a fourth order multgrid-based compact Poisson solver has been considered in [28]. Multigrid implementation of sixth order compact difference with the aid of Richardson extrapolation has been devised in [44]. As the first fast Poisson solver based on high order central differences, the AMIB method [17,19] can be made to arbitrarily high order in principle, and can handle the Dirichlet, Neumann, Robin or any combination of boundary conditions over cubic domains.

The goal of this paper is to develop high order central finite difference schemes with the FFT acceleration for solving elliptic BVPs on general shaped domains where Dirichlet, Neumann, and Robin or any combination of the boundary conditions can be imposed. To this end, two new matched interface and boundary (MIB) methods will be developed. First, a fourth order ray-casting MIB boundary scheme will be proposed in this work for solving elliptic BVPs over irregular domains. For rectangular domains, the MIB boundary scheme [48,50,51] is known to be a systematic approach for handling various general boundary conditions in arbitrarily high-order central differences, and has been applied for solving elliptic, parabolic, and hyperbolic partial differential equations (PDEs). The application of the MIB boundary method for treating complex domains has been reported once in the literature [49], in which a ray-casting MIB method is designed for perfectly electric conducting (PEC) boundary conditions in solving Helmholtz eigenvalue problems. The proposed ray-casting MIB scheme will be reformulated from the MIB PEC treatment [49], and will be generalized for solving elliptic BVPs with Dirichlet, Neumann, or Robin boundary conditions. For a comparison, a Cartesian MIB method will also be studied by generating fictitious values outside the irregular domain along the Cartesian directions. Owning to the geometric fact that the normal direction makes it possible for us to reach more grid points inside the domain compared to the Cartesian direction, the ray-casting MIB method is able to handle boundaries of complex shapes, and thus is much more flexible than the Cartesian MIB method. Moreover, the ray-casting MIB method could be generalized to deliver even higher order, such as six or eight.

Second, a fourth order augmented MIB (AMIB) scheme will be developed by coupling the ray-casting MIB boundary method with the augmented formulation [16,18]. We note that for rectangular domains, the augmented version of the MIB boundary scheme [48,50,51] has been presented in [17,19] for solving elliptic BVPs with various boundary conditions, in which the FFT fast inversion is applied for this first time in the literature to high order central differences. In this work, zero-padding zones will be similarly introduced inside or outside the irregular domains to form a cubic domain so that original boundaries become immersed boundaries and the FFT inversion can be applied. Auxiliary variables will be introduced on immersed boundaries and will be solved together with the unknown solution in the Schur complement procedure of the augmented system. Because the increment of the iteration number for solving the auxiliary variables weakly depends on the mesh size, the proposed AMIB scheme achieves the FFT efficiency, while maintaining the fourth order accuracy of the ray-casting MIB scheme.

The proposed ray-casting AMIB method shares some similarities with the fourth order finite difference method developed in [23] for the Laplace equation, including using a simple Cartesian grid, implementing the standard central difference schemes, shifting of the interpolation in the case of Dirichlet boundaries, attaining high order convergence, as well as dealing with irregular domains. However, there are some differences between them. First, the AMIB method can handle Dirichlet, Neumann, as well as Robin conditions, while the fourth order scheme in [23] is designed for Dirichlet boundary only. Second, in this study, the ray-casting MIB method is equipped with a fast algorithm and the resulting FFT-AMIB produces a high efficiency with $O(N \log N)$ complexity, while no fast Poisson solver is concerned in [23].

Compared with the existing AMIB schemes [16–19], the present AMIB method has two unique features. First, the present AMIB method is designed for BVPs over an irregular domain, while the previous AMIB algorithm [17,19] for elliptic BVPs only works for rectangular domains. Second, in the proposed ray-casting MIB scheme, the boundary condition is applied only along the normal direction for fictitious value generation, while in the preview MIB schemes [16–19], fictitious values are constructed along Cartesian directions through certain decomposition of the boundary/interface conditions.

The rest of the paper is organized as follows. In section 2, the ray-casting MIB method will be presented for solving elliptic BVPs. Then, the augmented MIB scheme will be formulated. Section 3 is dedicated to the numerical results to demonstrate the performance of the proposed algorithm. A summary and future plan will be discussed at the end of this paper.

2. Theory and algorithm

In this work, we are concerned with fast and high order numerical solution of elliptic equation (1) on a complex domain Ω with a smooth boundary subject to various boundary conditions (2). In order for us to fulfill finite difference discretization to the concerned problem, the given complex domain Ω is embedded into a rectangular domain $D = [a, b] \times [c, d]$ or a cube $D = [a, b] \times [c, d] \times [e, f]$. With a uniform grid spacing in each dimension, the domain D is partitioned into n_x, n_y, n_z equally spaced intervals in $x-$,$y-$, and $z-$ directions respectively with mesh increments $h_x = (b-a)/n_x$, $h_y = (d-c)/n_y$, and $h_z = (f-e)/n_z$. If only 2D problems are considered, the partition in the $z-$ direction can be ignored. The grid coordinates in 3D are therefore defined as

$$x_i = a + ih_x,\ y_j = c + jh_y,\ z_k = e + kh_z, i = 0, \cdots, n_x,\ j = 0, \cdots, n_y,\ k = 0, \cdots, n_z.$$

We will take advantage of the central differences to approximate the Laplacian operator in the elliptic equation (1). Due to a tensor product style for the partial derivatives, the approximation to Laplacian operator can be decomposed in several directions separately. Without loss of generality, we consider the standard fourth order finite difference discretization for second order partial derivative of x with a truncation error $O(h_x^4)$ as below:

$$u_{xx}(x_i) \approx \frac{1}{h_x^2}[-\frac{1}{12}u(x_{i-2}) + \frac{4}{3}u(x_{i-1}) - \frac{5}{2}u(x_i) + \frac{4}{3}u(x_{i+1}) - \frac{1}{12}u(x_{i+2})]. \tag{3}$$

Analogously, stencil could be obtained for $y-$ and $z-$ partial derivatives.

2.1. Immersed boundary

It has found in our recent work [17] that the discrete Laplacian matrix generated by the fourth order central difference can be inverted by the FFT algorithm, only if an anti-symmetric property is satisfied in the solution across the boundaries of the rectangular domain. Thus, several layers of zero-padding solutions are introduced outside the domain in the AMIB scheme [17], so that the anti-symmetric property is trivially valid at the extend boundaries. Following the same idea, we will assume the solution outside the domain Ω being zero too. Moreover, in order to achieve the desired order of accuracy, the extended domain D needs to be large enough such that sufficient fictitious values are available for the MIB treatment. Now we take fourth order central difference scheme in $2D$ shown in Fig. 1 for a demonstration. The shortest distance between the original boundary Γ and the exterior boundary ∂D should be great than $2h_x$ or $2h_y$ in each direction to ensure the fourth order accuracy.

From now on, we define the original domain Ω as Ω^- and the extended domain as Ω^+, yielding the whole domain $D = \Omega^- \cup \Omega^+$. Correspondingly, the solution in the original domain Ω^- is denoted as u^- with the source term as f^-, and the solution in the extended domain Ω^+ is simply defined as $u^+ = 0$, with the related source term as $f^+ = 0$. The boundary $\Gamma = \Omega^- \cap \Omega^+$ is defined by a level set function $\Gamma = \{(x, y), \varphi(x, y) = 0\}$, with $\varphi(x, y) < 0$ in Ω^- and $\varphi(x, y) > 0$ in Ω^+. We assume Γ to be C^1 continuous and f^- a smooth function. Within the boundary Γ, the solution u is assumed to be sufficiently smooth in Ω^-. For instance, to allow for a fourth order convergence, u is assume to be at least C^6 continuous in Ω^-. Then, the immersed boundary problem can be modeled as

$$\triangle u + \kappa u = f(\mathbf{x}),\ \mathbf{x} \in D, \tag{4}$$

with the source term being

$$f = \begin{cases} f^-, \mathbf{x} \in \Omega^-, \\ f^+, \mathbf{x} \in \Omega^+. \end{cases}$$

The remodeled problem introduces the extended boundary ∂D, beyond which ghost values are considered as 0 such that the anti-symmetry property is trivially satisfied across the exterior boundary ∂D, which lays a foundation for the aforementioned FFT inversion.

Referring to Fig. 1, away from the boundary Γ, the fourth order central difference approximation given in Eq. (3) will be employed at all regular points. However, for irregular points near the boundary, the central difference will need function values at some nodes outside Ω^-. A formulation for irregular points in the fourth order case can be given as below. Define min and max functions at a node (x_i, y_j) as

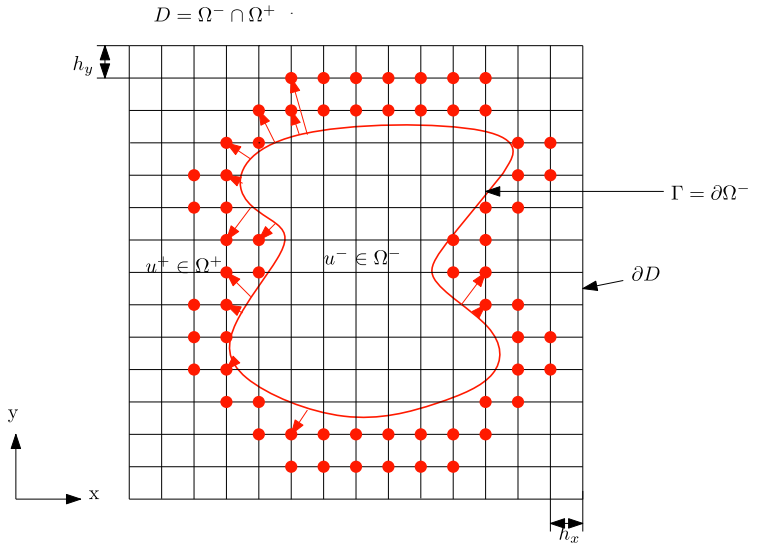

Fig. 1. The original problem is converted into an immersed boundary problem. The red curve indicates the smoothly curved boundary Γ and ∂D is the introduced exterior boundary. Red dots represent the fictitious nodes for the corrected fourth order central difference. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

$$\varphi_{ij}^{min} = \min\{\varphi_{i-2,j}, \varphi_{i-1,j}, \varphi_{i,j}, \varphi_{i+1,j}, \varphi_{i+2,j}, \varphi_{i,j-2}, \varphi_{i,j-1}, \varphi_{i,j+1}, \varphi_{i,j+2}\},$$
$$\varphi_{ij}^{max} = \max\{\varphi_{i-2,j}, \varphi_{i-1,j}, \varphi_{i,j}, \varphi_{i+1,j}, \varphi_{i+2,j}, \varphi_{i,j-2}, \varphi_{i,j-1}, \varphi_{i,j+1}, \varphi_{i,j+2}\}.$$

Recall that the interface Γ is represented by the zero level set $\varphi(x, y) = 0$. When $\varphi_{i,j}^{min}\varphi_{i,j}^{max} < 0$, we call the grid node (x_i, y_j) a irregular point, otherwise regular point. For the fourth order central difference, the irregular points consist of two layers of nodes inside Γ, while sixth and eighth order central difference approximations with even wider stencils involve more irregular points. The central difference approximation at irregular points has to be corrected [16,17].

Remark 2.1. With some modifications, the proposed numerical algorithm can be applied to handle an exterior irregular domain problem, in which the PDE is defined in Ω^+ and the zero padding is applied in Ω^- to obtain $u^- = 0$. See Example 6 in Section 3 for instance. Moreover, we can handle a triple-layers problem with the PDE defined in the middle layer, and the solution is trivially zero for inside and outside layers, see Example 5.

2.2. Fictitious values formulation and ray-casting MIB method

To correct the high order central difference approximation at irregular points, fictitious values outside the domain Ω^- are employed in the MIB boundary schemes [48–51]. Such fictitious values can be regarded as smooth extension of solution across the boundary, but will be rigorously determined according to boundary conditions. To complete the fourth, sixth, and eighth order finite difference approximation respectively, 2, 3, and 4 layers of fictitious values outside Γ^- are required to achieve the desired accuracy. For example, in Fig. 1, two layers of fictitious nodes are shown.

Besides treating boundaries of rectangular domains [48,50,51], the MIB boundary method has been developed to deal with curved boundaries in [49]. In particular, a fourth order ray-casting MIB is constructed for perfectly electric conducting (PEC) boundary conditions in solving Helmholtz eigenvalue problems [49]. In this present study, the ray-casting MIB method will be reformulated for solving the Robin condition, and is generalized to sixth and eighth orders, as well as to three-dimensional (3D) elliptic BVPs.

An illustration of the fourth order ray-casting MIB scheme is given in Fig. 2. In this case, two layers of fictitious values are needed outside Γ by means of the ray-casting treatment. In Fig. 2, red filled squares stand for the first layer of fictitious points, while green crosses represent the second layer of fictitious points. For each fictitious node, we will locate an interface point such that the fictitious node is along the normal direction of this interface point. With the assumption that the interface Γ is C^1 continuous, such an inverse normal projection is always feasible. In the proposed ray-casting MIB method, the fictitious value will be generated by enforcing the boundary condition at the corresponding interface point. Moreover, as shown in Fig. 2, all fictitious values will be determined independently, by using different interface points. This is different from the regular MIB boundary method [48,50,51], in which the boundary condition at one interface point will be repeatedly enforced to calculate several fictitious values.

We first consider how every possible fictitious value is calculated using the ray-casting MIB treatment. Referring to the fictitious node (x_i, y_j) in Fig. 3, an inversely projected normal line or auxiliary normal line is generated, which not only

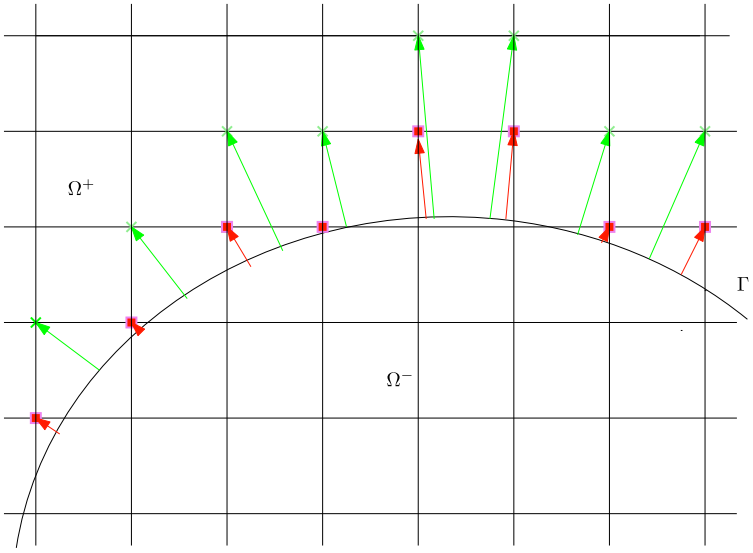

Fig. 2. An illustration of the ray-casting MIB scheme.

passes through the fictitious point (x_i, y_j), but also is perpendicular to the boundary Γ at an intersection point P. As shown in Fig. 3, this auxiliary line meets a x grid line first, yielding four auxiliary nodes on four different x grid lines. Then, four auxiliary points inside the boundary Γ together with the fictitious node (x_i, y_j) serve as a five-point stencil, which is sufficient to ensure the fourth order of accuracy. We define the four auxiliary nodes as (x_{M_1}, y_{j-1}), (x_{M_2}, y_{j-2}), (x_{M_3}, y_{j-3}), (x_{M_4}, y_{j-4}), which will be utilized to discretize the boundary condition. Correspondingly, we denote $\hat{u}_{i,j}$ as a fictitious value at (x_i, y_j). The function values at the four auxiliary nodes are defined as $u_{M_1}, u_{M_2}, u_{M_3}, u_{M_4}$. Denote the angle between the outward normal line and the positive x direction to be θ.

Now, it is sufficient to illustrate the ray-casting formulation of fictitious values with generic boundary conditions in 2D,

$$\alpha u + \beta \frac{\partial u}{\partial n} = \phi \quad \text{on } \Gamma. \tag{5}$$

We generate fictitious values by imposing the boundary conditions along the normal direction or the auxiliary line. Firstly, one fictitious value $\hat{u}_{i,j}$ is constructed to approximate the boundary condition Eq. (5):

$$\alpha(C_1^{(0)}\hat{u}_{i,j} + \sum_{i=2}^{5} C_i^{(0)} u_{M_{i-1}}) + \beta(C_1^{(1)}\hat{u}_{i,j} + \sum_{i=2}^{5} C_i^{(1)} u_{M_{i-1}}) = \phi, \tag{6}$$

where $C_i^{(m)}$ for $m = 0, 1$ and $i = 1, 2, \cdots, 5$ are finite difference (FD) weights [20]. Here the superscript m represents interpolation ($m = 0$) or the first order derivative approximation ($m = 1$) at the intersection point P by using a stencil $\{\hat{u}_{i,j}, u_{M_1}, u_{M_2}, u_{M_3}, u_{M_4}\}$. Hence, the unknown $\hat{u}_{i,j}$ can be represented as the combination of u_{M_i} for $i = 1, \cdots, 4$ and ϕ. However, the four auxiliary nodes are not generally located on the Cartesian grid nodes. Secondly, to get a full Cartesian grid approach, we will further seek to determine fictitious values using the function values on the grid nodes. To be more clear, for each auxiliary point, five nearest grid nodes exclusively inside the domain Ω^- are used to interpolate the auxiliary point along each x line, see Fig. 3. As a consequence, each function value at an auxiliary point can be represented as

$$u_{M_i} = \sum_{k=1}^{5} w_{i,k}^{(0)} u_{i_k, j-i}, \quad i = 1, \cdots, 4, \tag{7}$$

where $w_{i,k}^{(0)}$ for $k = 1, 2, \cdots, 5$ are FD weights [20] to approximate function value at each auxiliary point (x_{M_i}, y_{j-i}) using a stencil $\{u_{i_1, j-i}, \cdots, u_{i_5, j-i}\}$ in which i_k for $k = 1, \cdots, 5$ are the indexes of x coordinates of five nearest grid nodes within the domain with i_k for $k = 1, 2, \cdots, 5$ being positive integers. In this manner, we have obtained a representation of function value at each auxiliary point (x_{M_i}, y_{j-i}) for $i = 1, 2, \cdots, 4$ in terms of some grid nodes.

Finally, by plugging the function values at the four auxiliary points Eq. (7) into Eq. (6), the fictitious values can be rewritten into a general form in 2D. We have

$$\hat{u}_{i,j} = \sum_{(x_I, y_J) \in \mathbb{S}_{i,j}} W_{I,J} u_{I,J} + W_3 \phi, \tag{8}$$

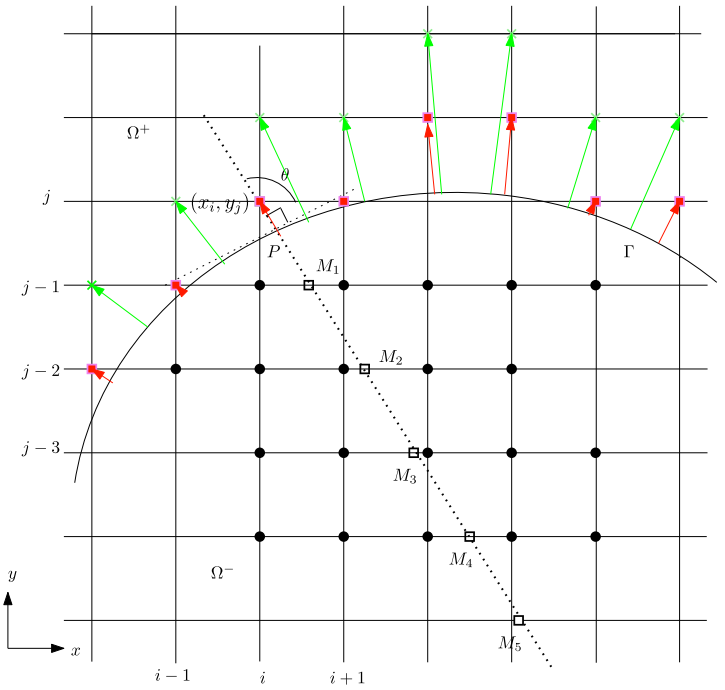

Fig. 3. Consider the representation of one fictitious point (x_i, y_j) along normal direction, open squares represent auxiliary nodes. Filled circles stand for the chosen Cartesian grid points to calculate the fictitious value at (x_i, y_j).

where $\mathbb{S}_{i,j}$ is a chosen node set which contains the required Cartesian grid nodes within the domain. Eq. (8) is a linear combination of functions values on the chosen set $\mathbb{S}_{i,j}$, and known boundary data. In fact, Eq. (8) is applicable for Dirichlet, Neumann, Robin boundary conditions and even any other mixed form of boundary conditions, because non-homogeneous term ϕ is usually known.

Remark 2.2. There are two scenarios where the auxiliary line intersects with the x or y grid line first within Ω^-. This can be determined by figuring out the relation between the mesh angle $\arctan(h_y/h_x)$ and θ. For example, if $h_x = h_y$, when $\frac{\pi}{4} \leqslant \theta \leqslant \frac{3\pi}{4}$ (as shown in Fig. 3) or $\frac{5\pi}{4} \leqslant \theta \leqslant \frac{7\pi}{4}$, the auxiliary line meets the x grid line first, so the auxiliary line intersecting with four x grid lines gives us four auxiliary nodes. Otherwise, the four auxiliary points are chosen as the intersection points of the auxiliary lines with y grid lines, yielding a number of grid points within the domain for formulating fictitious values in a similar fashion.

Remark 2.3. In our computation, the auxiliary nodes are generated adaptively. We consider that there are cases in which certain x or y grid line should be skipped due to the unavailability of grid nodes inside Ω^- for the interpolation of certain auxiliary nodes. The new auxiliary node shall be set as the intersection of the auxiliary line with the next x or y grid line. In other words, a down shifting of all involved nodes in certain grid line needs to be carried out. This could usually happen when calculating the second layer of fictitious values. The first auxiliary node could be outside of the domain, so the first auxiliary line can be ignored, see the fictitious point at (x_i, y_{j+1}) shown in Fig. 3. We repeat the down shifting process at most twice until there are enough grid nodes available at each grid line for interpolating the auxiliary nodes. If the down shifting is carried out more than twice, the chosen grid nodes are far away from the targeted fictitious point, which makes the construction of fictitious values lack of the desired accuracy to some extent.

Remark 2.4. The ray-casting MIB scheme can be simply generalized to 3D, in which the boundary condition is still enforced along an auxiliary normal line. The difference is that each auxiliary node will be interpolated in a 2D plane, not along one grid line.

2.3. Dirichlet boundary condition and Cartesian MIB method

We note that there are two options in dealing with Dirichlet boundary conditions. First, the ray-casting MIB method as depicted in Fig. 3 can be applied. Second, since there is no directional derivative in the Dirichlet boundary condition, the condition can be discretized along any direction. Thus, one can simply choose x or y direction to reduce the FD stencil size. For example, for the previously studied case with a fictitious node (x_i, y_j), a Cartesian MIB boundary treatment is shown

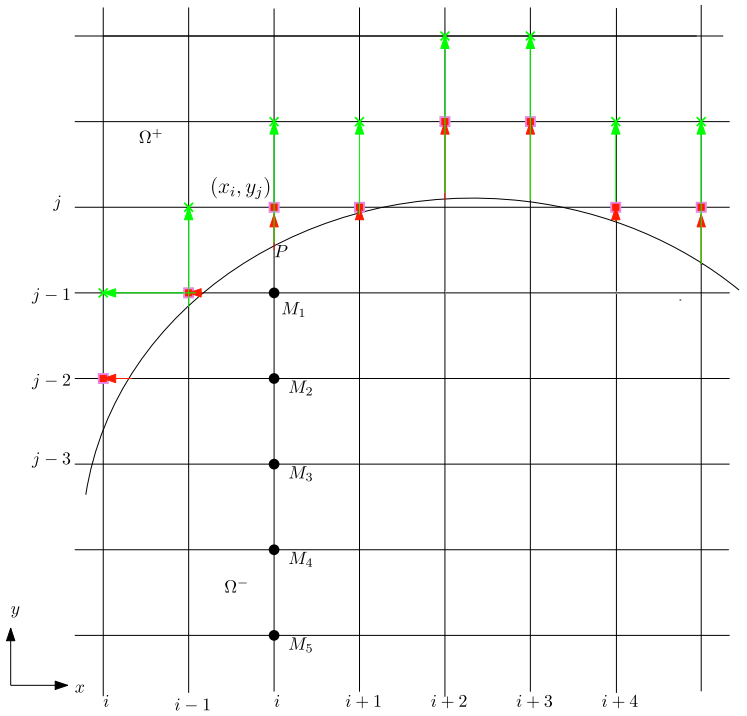

Fig. 4. Consider the formation of fictitious point (x_i, y_j) along Cartesian x or y direction. Here M_1, M_2, M_3, M_4 represent the chosen grid nodes to generate the fictitious value at (x_i, y_j). Red filled squares stand for the first layer of fictitious points, while green crosses represent the second layer of fictitious nodes.

in Fig. 4. In particular, the Dirichlet boundary condition at the interface point P can be discretized as in Eq. (6) by taking $\beta = 0$. Moreover, the auxiliary nodes are on-grid now, so that one does not need to further interpolate them. The generated fictitious value can also be represented as in Eq. (8), except that the set $\mathbb{S}_{i,j}$ involves much fewer nodes. We note that the idea behind the Cartesian MIB method is essentially the same as that proposed in [23]. In fact, Ref. [23] is the first study in the literature on developing the fourth order finite difference algorithm for solving elliptic BVPs with Dirichlet boundary conditions, in which the fictitious values are calculated by generating a fitting polynomial in [23].

There are some differences between the Ray-casting and Cartesian MIB schemes for the Dirichlet boundary condition. The fictitious node at (x_i, y_j) is generated along y direction for the Cartesian method as shown in Fig. 4, while the formation of the fictitious point at (x_i, y_j) is implemented along normal direction for the ray-casting scheme as shown in Fig. 3. By the geometric nature of the normal direction, all fictitious values are independent from each other by means of the ray-casting MIB treatment, while the Cartesian MIB scheme determines two fictitious values in pair usually. It can be seen in Fig. 4, the second layer of fictitious values is formulated based on the first layer for Cartesian MIB scheme. Also, for certain fictitious points, one can impose Dirichlet boundary condition in either x or y direction. In this case, an optimal choice of the Cartesian directions can be considered in the Cartesian MIB method, while there is no alternative direction for ray-casting scheme.

The Cartesian MIB method enjoys potential benefits in comparing with the ray-casting MIB method for simply shaped domains. First, fewer grid nodes are usually required along Cartesian directions to determine fictitious values. Each fictitious value depends on 20 function values as shown in Fig. 3 for the ray-casting MIB discretization, while only four grid nodes M_1, M_2, M_3, M_4 are needed to generate each fictitious value along y direction as shown in Fig. 4. Second, the grid nodes for the extrapolation along Cartesian direction are closer to the intersection node P on the interface, which is more accurate compared with the ray-casting method. Nevertheless, the ray-casting method is applicable to deal with complex shaped domains. The normal direction makes it possible for us to reach more grid points within irregular domains compared to Cartesian direction such that a minimal number of grid points are attained to generate fictitious points.

An important technique suggested in [23] will also be adopted in the present study. It is found in [23] that, in case of Dirichlet boundary conditions, if a grid node is very close to the interface point P, the accuracy of the generated fictitious value will be seriously reduced. Thus, a simple shifting trick is introduced in [23] to avoid the accuracy reduction. The same trick will be applied to both Cartesian MIB and ray-casting MIB methods. As shown in Fig. 3 and Fig. 4, we define Θ as $\frac{|P-M_1|}{|M_1-M_2|}$. If Θ is less than a given tolerance S, we will shift the all auxiliary nodes one position towards the interior of domain Ω^-. For example, for both Fig. 3 and Fig. 4, the Dirichlet boundary condition will be discretized by using a stencil $\{\hat{u}_{i,j}, u_{M_2}, u_{M_3}, u_{M_4}, u_{M_5}\}$ instead of $\{\hat{u}_{i,j}, u_{M_1}, u_{M_2}, u_{M_3}, u_{M_4}\}$. As noticed in [23], this shifting technique will reduce the

condition number of the final finite difference discretization matrix, and this will be verified in the present study too. In practice, S will be chosen to be a small number, so that the shifting is conducted only when necessary.

2.4. The augmented MIB (AMIB) method

Different from the original MIB boundary schemes [48–51], an augmented MIB (AMIB) method will be developed for solving elliptic BVPs over irregular domains. Following the AMIB methods [16–19] for interface and boundary value problems over rectangular domains, the fictitious values generated above will not be directly used in the Laplacian approximation. The proposed AMIB method consists of two stages. First, the Laplacian operator will be discretized by using the standard central differences, subject to the availability of Cartesian derivative jumps $[u^{(k)}]$ at various interface points. Second, the derivative jumps $[u^{(k)}]$ will be approximated by using the MIB fictitious values. Details are offered in the following two parts.

2.4.1. Correcting central differences

Thanks to the fact that central difference approximation to the Laplacian is carried out dimension by dimension, it is sufficient to discuss how the central differences will be corrected by considering only a 1D problem. Denote $u = u(x)$ and assume that the interface Γ intersects the x grid line at $x = \alpha$. The following theorem provides necessary formulas to correct the fourth order central difference.

Theorem 1 *(Corrected fourth order finite differences [18,45]). Let $x_j \le \alpha \le x_{j+1}$, $h^- = x_j - \alpha$, and $h^+ = x_{j+1} - \alpha$. Suppose $u \in C^6[x_j - 2h, \alpha) \bigcap C^6(\alpha, x_{j+1} + 2h]$, with derivative extending continuously up to the interface α. Then the following approximations hold to $O(h^4)$ when $K = 5$:*

$$
\begin{aligned}
u_{xx}(x_{j-1}) &\approx \frac{1}{h^2}[-\frac{1}{12}u(x_{j-3}) + \frac{4}{3}u(x_{j-2}) - \frac{5}{2}u(x_{j-1}) + \frac{4}{3}u(x_j) - \frac{1}{12}u(x_{j+1})] \\
&\quad + \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h^+)^k}{k!}[u^{(k)}], \\
u_{xx}(x_j) &\approx \frac{1}{h^2}[-\frac{1}{12}u(x_{j-2}) + \frac{4}{3}u(x_{j-1}) - \frac{5}{2}u(x_j) + \frac{4}{3}u(x_{j+1}) - \frac{1}{12}u(x_{j+2})] \\
&\quad - \frac{4}{3h^2}\sum_{k=0}^{K}\frac{(h^+)^k}{k!}[u^{(k)}] + \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h+h^+)^k}{k!}[u^{(k)}], \\
u_{xx}(x_{j+1}) &\approx \frac{1}{h^2}[-\frac{1}{12}u(x_{j-1}) + \frac{4}{3}u(x_j) - \frac{5}{2}u(x_{j+1}) + \frac{4}{3}u(x_{j+2}) - \frac{1}{12}u(x_{j+3})] \\
&\quad + \frac{4}{3h^2}\sum_{k=0}^{K}\frac{(h^-)^k}{k!}[u^{(k)}] - \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h^- - h)^k}{k!}[u^{(k)}], \\
u_{xx}(x_{j+2}) &\approx \frac{1}{h^2}[-\frac{1}{12}u(x_j) + \frac{4}{3}u(x_{j+1}) - \frac{5}{2}u(x_{j+2}) + \frac{4}{3}u(x_{j+3}) - \frac{1}{12}u(x_{j+4})] \\
&\quad - \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h^-)^k}{k!}[u^{(k)}],
\end{aligned}
$$

where $[u^{(k)}] = [u^{(k)}]_\alpha = \lim_{x\to\alpha^+} u^{(k)}(x) - \lim_{x\to\alpha^-} u^{(k)}(x)$.

The above corrected differences focus on x-direction derivatives. Corrected differences for y-direction and z-direction can be defined in a similar fashion. One can combine corrected differences from different directions to approximate the Laplacian operator dimension by dimension. The corrected difference consists of different jump quantities of derivatives and maintains the standard central difference stencil. In fact, such correction terms vanish at regular points, while correction terms occur for the irregular points case.

In order to achieve fourth order accuracy, we need jump quantities up to five order derivative with a truncation error $O(h^4)$. Nevertheless, it is sufficient to take $K = 4$ in the fourth order corrected differences with a third order local truncation error $O(h^3)$ at irregular points. In this way, global fourth order of convergence can still be maintained.

The corrected fourth order finite differences are applicable in dealing with a complicated curved interface, for which a grid line may cut through the interface twice within a relatively short distance. The treatment to approximation of corrected finite differences depends on the amount of grid points between two intersection points. If there is only one grid point located between two adjacent intersection points, a grid refinement is required. If there are at least three grid points available between two intersection points, two interface points can be treated independently and the above corrected formulas can be safely applied. We name this situation as Type 1. However, if only two adjacent grid points are available between two intersection points, the corrected differences are applied in a different mode. We name this situation as Type 2. As shown in Fig. 6, with $y = y_j$ cutting Γ twice at α_1 and α_2, two grid points are located along x direction inside Γ. The formulas for multiple corrections are derived in x-direction for demonstration as following:

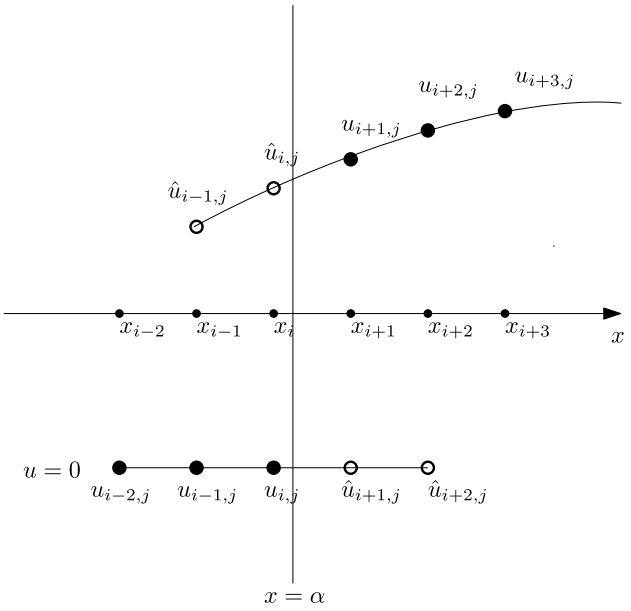

Fig. 5. An illustration of the general fourth order numerical approximation to Cartesian derivative jumps by two polynomials near the interface $x = \alpha$. The black dots stand for the real values, while the circles denote the fictitious values.

Theorem 2 *(Fourth order multiple corrections [18]). Let $x_{j-1} \le \alpha_1 \le x_j \le x_{j+1} \le \alpha_2 \le x_{j+2}, h_1^- = x_{j-1} - \alpha_1, h_1^+ = x_j - \alpha_1$ and $h_2^- = x_{j+1} - \alpha_2, h_2^+ = x_{j+2} - \alpha_2$. Suppose $u \in C^6[x_j - 2h, \alpha) \bigcap C^6(\alpha_1, \alpha_2) \bigcap C^6(\alpha, x_{j+1} + 2h]$, with derivative extending continuously up to boundaries of the subintervals. Then the following fourth order approximations to second order derivative hold with truncation error $O(h^4)$ when $K = 5$:*

$$
\begin{aligned}
u_{xx}(x_j) \approx{}& \frac{1}{h^2}[-\frac{1}{12}u(x_{j-2}) + \frac{4}{3}u(x_{j-1}) - \frac{5}{2}u(x_j) + \frac{4}{3}u(x_{j+1}) - \frac{1}{12}u(x_{j+2})] + \\
&+ \frac{4}{3h^2}\sum_{k=0}^{K}\frac{(h_1^-)^k}{k!}[u^{(k)}]_{\alpha_1} - \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h_1^- - h)^k}{k!}[u^{(k)}]_{\alpha_1} + \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h_2^+)^k}{K!}[u^{(k)}]_{\alpha_2}, \\
u_{xx}(x_{j+1}) \approx{}& \frac{1}{h^2}[-\frac{1}{12}u(x_{j-1}) + \frac{4}{3}u(x_j) - \frac{5}{2}u(x_{j+1}) + \frac{4}{3}u(x_{j+2}) - \frac{1}{12}u(x_{j+3})] \\
&- \frac{4}{3h^2}\sum_{k=0}^{K}\frac{(h_2^+)^k}{k!}[u^{(k)}]_{\alpha_2} + \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h + h_2^+)^k}{k!}[u^{(k)}]_{\alpha_2} - \frac{1}{12h^2}\sum_{k=0}^{K}\frac{(h_1^-)^k}{k!}[u^{(k)}]_{\alpha_1}.
\end{aligned}
$$

All the above corrected differences in the case of single interface or multiple interfaces are concerned with x-direction derivatives, which applies to y-and z-direction in a similar fashion.

2.4.2. Approximation to Cartesian derivative jumps

The Cartesian derivative jumps involved in the corrected central differences are actually unknown. We need to further construct these jump quantities by approximating limiting derivatives. The jump quantity at $x = \alpha$ is defined as

$$
[\frac{\partial^k u}{\partial x^k}]|_{x=\alpha} = \lim_{x \to \alpha^+}\frac{\partial^k u}{\partial x^k} - \lim_{x \to \alpha^-}\frac{\partial^k u}{\partial x^k}, \tag{9}
$$

where $k = 0, 1, \cdots, 4$. Note that the one-sided limits here are right-hand side limit $x \to \alpha^+$ and left-hand side limit $x \to \alpha^-$. The superscripts here are different from those used in the inside subdomain Ω^- and outside subdomain Ω^+.

In our computation, all intersection points of Γ with either x or y grid lines will be determined first. Then on these interface points, the needed Cartesian derivative jumps will be approximated by using the MIB fictitious values calculated above. The one-side limit in Eq. (9) will be approximated by building a Lagrange polynomial of degree 4. See Fig. 5 for an illustration. Two layers of fictitious values are available outside Ω^-. For the zero solution defined in Ω^+, we can trivially assume two layers of fictitious values as well. In this way, two layers of fictitious nodes are available on both sides of the interface. For each side limit, Lagrange polynomials can be obtained with the aid of three real values on one side and two fictitious values on the other side of the interface. By taking derivative on these one-sided polynomials, each order derivative can be computed.

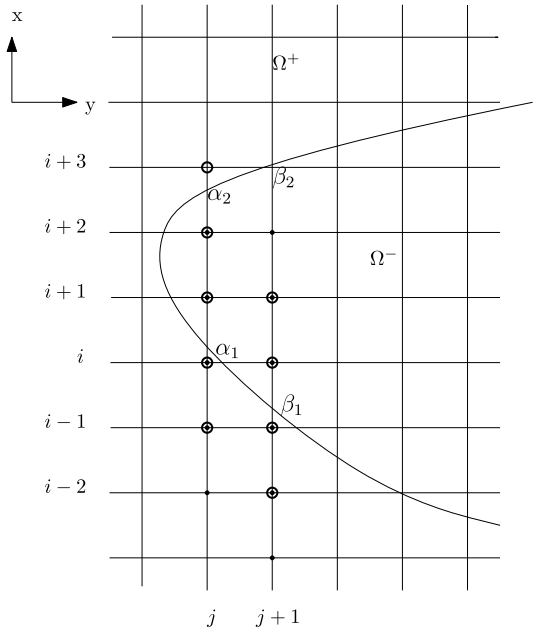

Fig. 6. Derivative jump approximation in two scenarios. Empty circles stand for the fictitious values at the grid points, while black dots denote real function values.

For the case shown in Fig. 5, the Cartesian derivative jumps are approximated by

$$[\frac{\partial^k u}{\partial x^k}]|_{x=\alpha} \approx (w^k_{i-1,j}\hat{u}_{i-1,j} + w^k_{i,j}\hat{u}_{i,j} + \sum_{l=1}^{3} w^k_{i+l,j}u_{i+l,j}) - (\sum_{l=1}^{3} w^k_{i-3+l,j}u_{i-3+l,j} + w^k_{i+1,j}\hat{u}_{i+1,j} + w^k_{i+2,j}\hat{u}_{i+2,j}), \tag{10}$$

where $w^k_{p,q}$ represents the FD weights approximating at $x=\alpha$, and (p,q) stands for the location (x_p, y_q). The above approximation Eq. (10) is actually for the general interface problems. In this paper, the function value beyond Ω^- is zero, limit from outside of the interface Γ is trivially zero. Hence, only approximation from the inside of the interface Γ is involved, and Eq. (10) can be rewritten as

$$[\frac{\partial^k u}{\partial x^k}]|_{x=\alpha} \approx (w^k_{i-1,j}\hat{u}_{i-1,j} + w^k_{i,j}\hat{u}_{i,j} + \sum_{l=1}^{3} w^k_{i+l,j}u_{i+l,j}) - 0.$$

As mentioned in the preceding subsection, the amount of grid points within Γ has to be monitored for applying the corrected difference. This issue has to be taken into account again when constructing derivative jumps by Eq. (10). Fig. 6 shows two scenarios. For Type 1 where at least three grid nodes locate between β_1 and β_2, Eq. (10) can be applied at each intersection point in a safe mode. For Type 2 in which only two grid nodes are available between α_1 and α_2, more fictitious values are needed in approximation to derivative jumps. For instance, in order to approximate derivative jumps $\alpha=\alpha_1$, a simple treatment is needed by replacing real function value $u_{i+3,j}$ with fictitious value $\hat{u}_{i+3,j}$ in Eq. (10).

2.5. Formulation of augmented system and FFT inversion

In the AMIB method, derivative jumps are treated as auxiliary variables to be solved to together with the unknown function values. With these auxiliary variables along with the function values, one augmented system can be constructed.

2.5.1. Auxiliary variables

In the last subsection, the representation of derivative jumps in terms of fictitious values is formulated at each intersection point between a grid line and Γ. Taking $[\frac{\partial^k u}{\partial x^k}]$ $k=0,1,\cdots,4$ as auxiliary variables, we plug fictitious value representation Eq. (8) into Eq. (10), which gives us equivalent forms:

$$\sum_{(x_I, y_J)\in\mathbb{S}_{i,j}} C_{I,J} u_{I,J} + [\frac{\partial^k u}{\partial x^k}] = C_0\phi, \tag{11}$$

where $C_{I,J}$ is the corresponding weights of function value $u_{I,J}$ in approximation to jump quantities $[\frac{\partial^k u}{\partial x^k}]$ $k = 0, 1, \cdots, 4$, and ϕ is the known interface quantities. Equivalent form for jump quantities $[\frac{\partial^k u}{\partial y^k}]$ could be derived in a similar fashion in the $y-$ direction. Moreover, we denote Q as a vector of the introduced auxiliary variables $[u]_i, [\frac{\partial u}{\partial x}]_i, [\frac{\partial^2 u}{\partial x^2}]_i, [\frac{\partial^3 u}{\partial x^3}]_i, [\frac{\partial^4 u}{\partial x^4}]_i$ for $i = 1, 2, \cdots$, and $[u]_j, [\frac{\partial u}{\partial y}]_j, [\frac{\partial^2 u}{\partial y^2}]_j, [\frac{\partial^3 u}{\partial y^3}]_j, [\frac{\partial^4 u}{\partial y^4}]_j$, for $j = 1, 2, \cdots$, at all intersection points between interface Γ and grid lines. Hence, the total number of auxiliary variables, denoted as N_2, is 5 times that of all intersection points. After generalizing Eq. (11) from one point to all interface points, a relation between numerical solution and auxiliary variable can be obtained as

$$CU + IQ = \Phi, \tag{12}$$

where C represents the weights coefficients matrix containing the corresponding finite difference weights for U, U stands for all function variables, I is the identity matrix, and Φ is composed of terms after moving the known quantities to the right-hand side.

2.5.2. Augmented system

We are concerned with approximation to 2D elliptic equation. Let $U_{i,j}$ indicate the discrete solution of analytical solution $u(x_i, y_j)$ at (x_i, y_j). Based on the corrected difference analysis, the problem (4) can be discretized as

$$L_h U_{i,j} + C_{i,j} = f_{i,j},\ 1 \le i \le n_x - 1,\ 1 \le j \le n_y - 1, \tag{13}$$

where $C_{i,j}$ is the correction term, and $L_h U_{i,j}$ is the standard high order central difference approximation to Laplacian with a degree of freedom $N_1 = (n_x - 1) \times (n_y - 1)$. Taking advantage of the introduced variables Q in Eq. (12), Eq. (13) can be written as

$$AU + BQ = F, \tag{14}$$

where A is a symmetric, diagonally dominant matrix of dimension N_1 by N_1 consisting of coefficients from standard fourth order central difference approximation, and B is composed of coefficients from correction terms. Due to the fact that the correction term is nonzero only at the two layers of grid points surrounding the interface which are irregular nodes, B is a sparse matrix of dimension N_1 by N_2. Here F is a vector with entries being $f_{i,j}$.

Furthermore, keeping Eq. (12) in mind, matrix C is a sparse matrix of dimension $N_2 \times N_1$, and I is a N_2 by N_2 identity matrix. An augmented matrix is therefore obtained by combining Eq. (14) and Eq. (12)

$$KW = R, \tag{15}$$

where

$$K = \begin{pmatrix} A & B \\ C & I \end{pmatrix}, W = \begin{pmatrix} U \\ Q \end{pmatrix}, \quad \text{and} \quad R = \begin{pmatrix} F \\ \Phi \end{pmatrix}.$$

2.5.3. FFT inversion

The inversion of matrix A can be efficiently carried out by the FFT algorithm. As demonstrated in [16,17], an anti-symmetric property across the boundary of the extended domain is trivially satisfied in the proposed AMIB formulation, which provides a foundation for FFT inversion. Thus, the above matrix A facilitates FFT Poisson solver. Taking advantage that the multi-dimensional FFT inversion can be carried out based on one-dimensional (1D) FFT inversion in a tensor product style, the following 1D FFT algorithm is sufficient for the FFT inversion of A. Without loss of generality, consider a 1D linear system with dimension P,

$$Au = f, \tag{16}$$

where A is a symmetric, pentadiagonal matrix obtained through fourth order central difference discretization of u_{xx} [16,17]. The FFT inversion of (16) is obtained via the fast Sine transform as presented below:

1. Compute Sine transform for $f(x_j)$ via inverse fast Sine transform (IFST).
   $\hat{f}_l = \frac{2}{P+1} \sum_{j=1}^{P} f_j \sin(\frac{lj\pi}{P+1}),$ for $l = 1, \cdots, P.$
2. $\hat{u}_l = \frac{\hat{f}_l}{\lambda_l},$ for $l = 1, 2, \cdots, P$, where $\lambda_l = -\frac{1}{3h_x^2}[\cos(\frac{l\pi}{P+1}) - 1][\cos(\frac{l\pi}{P+1}) - 7].$
3. Compute u_j via fast Sine transform (FST): $u_j = \sum_{l=1}^{P} \hat{u}_l \sin(\frac{lj\pi}{P+1}),$ for $j = 1, \cdots, P.$

2.5.4. Schur complement

The augmented system (15) is solved by the Schur complement method. First, a biconjugate gradient method could be utilized to determine Q. In particular, eliminating U in Eq. (15) gives a linear system with dimension N_2-by-N_2 for Q,

$$(I - CA^{-1}B)Q = \Phi - CA^{-1}F. \tag{17}$$

The details of implementation on Schur complement system (17) with a biconjugate gradient iteration is described as below:

1. One obtains $\hat{F} = \Phi - CA^{-1}F$ simply by performing FFT on F for $A^{-1}F$ and some easy algebraic operations.
2. For the left hand, the matrix vector product of $(I - CA^{-1}B)Q$ is achieved by $(I - CA^{-1}B)Q$. Based on that, a FFT is carried out on $A^{-1}BQ$, then it is followed with simple multiplication and summation for $IQ - CA^{-1}(BQ)$. The transpose of $(I - CA^{-1}B)^T Q$ is completed by following the same strategy on $IQ - B^T A^{-1} C^T Q$.
3. A initial guess $Q = (0, 0, \cdots, 0)^T$ starts the biconjugate gradient iteration, and Q will be updated to start a new iteration until either the maximal iteration number 5000 or error tolerance equaling to 10^{-12} is reached.

After calculating Q, the numerical solution U could be solved by enforcing FFT inversion again

$$AU = F - BQ.$$

The efficiency of the proposed ray-casting AMIB method depends on two factors. First, the inversion of matrix A is realized by the FFT algorithm in all steps, which has a complexity of the order $O(N_1 \log N_1)$. Second, the iterative solution of Q in the Schur complement depends on the dimension N_2 and the actual iteration number. Recall that N_2 represents the number of interface points that Γ intercepts with all grid lines. Hence, N_2 is one-dimensionally smaller than N_1. Moreover, as to be shown in the next section that in most AMIB computations, the iteration number grows slowly with respect to the increment of N_1. Thus, the overall complexity of the AMIB method is also close to $O(N_1 \log N_1)$.

3. Numerical experiments

In this section, we will examine the accuracy and efficiency of the proposed augmented matched interface and boundary (AMIB) method in solving two or three dimensional elliptic equations with various boundary shapes and conditions. We will mainly focus on the ray-casting AMIB scheme, while the performance of AMIB employing fictitious value generated in Cartesian direction [51] will be examined too. The latter one is denoted as Cartesian AMIB when some comparisons are made in the following discussions. We denote AMIB4 as the fourth order AMIB method, and similar notation is used for AMIB6 and AMIB8. For simplicity, the domain D will be assumed to be a square domain with equally spaced nodes $n = n_x = n_y = n_z$.

The numerical accuracy and convergence of the numerical solutions in $2D$ problems are tested by calculating errors under the maximum norm and L_2 norm defined as

$$L_\infty = \max_{(x_i, y_j) \in \Omega^-} |u(x_i, y_j) - u_h(x_i, y_j)|,$$

$$L_2 = \sqrt{\frac{1}{n_{\Omega^-}^2} \sum_{(x_i, y_j) \in \Omega^-} |u(x_i, y_j) - u_h(x_i, y_j)|^2},$$

where $u(x_i, y_j)$ and $u_h(x_i, y_j)$ are respectively analytical and numerical solution inside the given domain Ω^- of interest, and the number n_{Ω^-} in L_2 norm denotes the total grid numbers in Ω^-. The error norms in 3D can be similarly defined.

The convergence rate of the scheme will be examined by the formula

$$\text{order} = \frac{\log(||E_1||/||E_2||)}{\log(h_1/h_2)},$$

where $||E_i||$ is the numerical error on mesh of spacing h_i for $i = 1, 2$, using the above defined norms on n by n mesh for the computational domain D. The adopted FFT subroutine is from Numerical Recipes [40]. For both the AMIB and MIB computations, the iteration numbers in the biconjugate gradient iterative algorithm are reported in the discussion below.

All the experiments were carried out on Lenovo Laptop with Intel 1.80 GHz Intel Core i5.

3.1. Numerical examples in 2D

Example 1. This example focuses on the comparison of ray-casting AMIB and ray-casting MIB for solving Poisson's equation

$$u_{xx} + u_{yy} = e^{-x^2 - 0.5y^2}(4x^2 + y^2 - 3),$$

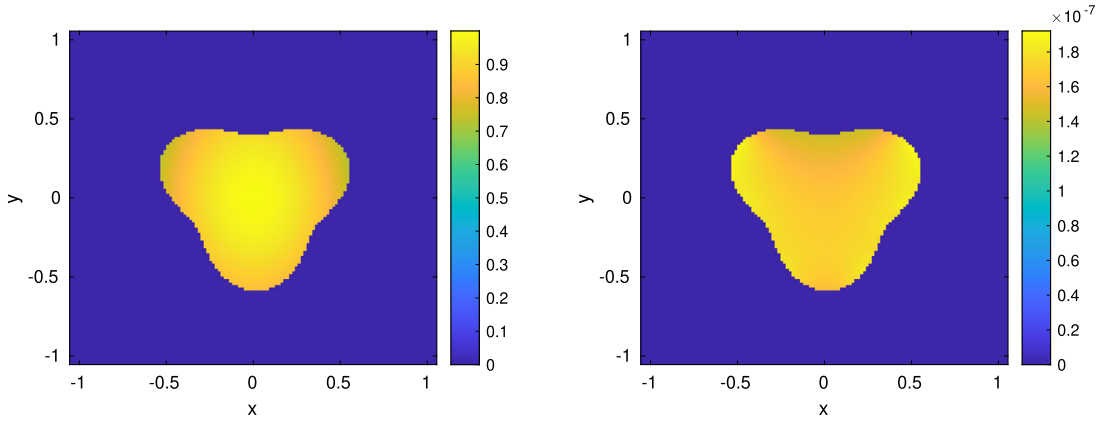

Fig. 7. The numerical solution (left) and error (right) of Example 1 on a mesh with $n = 129$.

Table 1
Example 1 – Fourth order numerical error analysis of AMIB vs MIB with $tol = 10^{-10}$ and 10^{-12}, respectively, for Robin boundary condition and Dirichlet boundary condition.

$[n_x, n_y]$	Robin boundary – AMIB4					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	2.143E-6	–	2.983E-6	–	21	4.69E-2
[129, 129]	1.672E-7	3.6799	1.920E-7	3.9576	21	0.14
[257, 257]	8.1466E-9	4.3592	9.515E-9	4.3347	22	0.719
[513, 513]	5.525E-10	3.8822	6.362E-10	3.9027	26	3.094
$[n_x, n_y]$	Robin boundary – MIB4					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	9.263E-7	–	2.967E-6	–	155	0.125
[129, 129]	7.197E-8	3.6860	1.9186E-7	3.9509	314	1.0156
[257, 257]	3.503E-9	4.3607	9.537E-9	4.3304	624	7.766
[513, 513]	2.342E-10	3.9028	6.313E-10	3.9171	1398	69.9
$[n_x, n_y]$	Dirichlet boundary – AMIB4					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	5.162E-8	–	1.347E-7	–	47	6.25E-2
[129, 129]	2.759E-9	4.2257	5.122E-9	4.7169	61	0.328
[257, 257]	1.637E-10	4.0750	3.092E-10	4.0501	89	2.25
[513, 513]	1.000E-11	4.0330	1.896E-11	4.0275	116	12.5
$[n_x, n_y]$	Dirichlet boundary – MIB4					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	2.230E-8	–	1.637E-7	–	195	0.141
[129, 129]	1.166E-9	4.2574	5.383E-9	4.9265	343	1.0625
[257, 257]	6.975E-11	4.0632	3.087E-10	4.1241	749	9.86
[513, 513]	5.8662E-12	3.5717	3.695E-11	3.0626	1594	80.9

with an embedded boundary $\Gamma : r = 0.5(1 + 0.2\sin(3\theta))$ in a computational domain $D : [-\pi/3, \pi/3] \times [-\pi/3, \pi/3]$. The equation is subject to Dirichlet boundary conditions or Robin boundary conditions on Γ determined by analytical solution $u(x, y) = e^{-x^2-0.5y^2}$. The numerical solution and error of the AMIB scheme for the Robin boundary are shown in Fig. 7.

The performance comparison between the AMIB4 and MIB4 with Dirichlet or Robin boundary conditions could be observed under each of the two norms in Table 1. It can be seen that all four methods achieve fourth order of convergence. Moreover, the iteration numbers of the four methods are also reported. The iteration numbers of AMIB4 with both Dirichlet and Robin boundary conditions increase slower and are less dependent on the mesh size in comparing with those of MIB4. In the case of Robin boundary conditions, when the mesh size is doubled for three times, the iteration number of AMIB4 increases very little, while the iteration number of MIB4 becomes 4.5 times larger. Correspondingly, the CPU cost of the

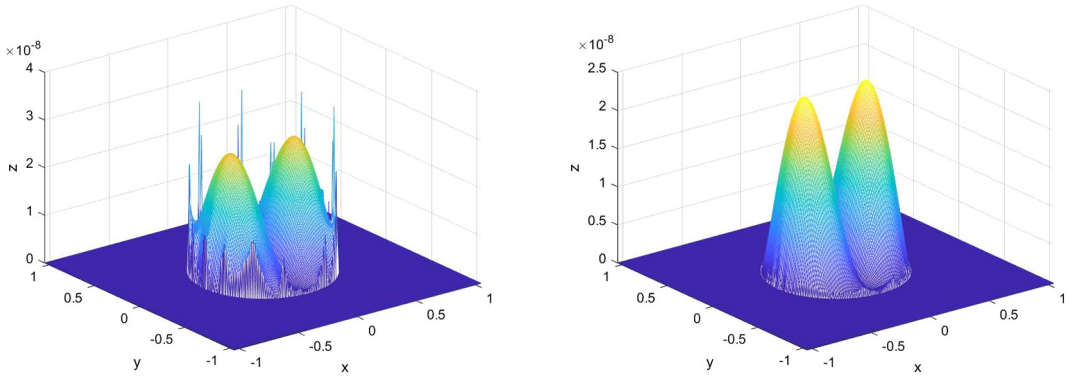

Fig. 8. The computed error of ray-casting method (left) and Cartesian method (right) in Example 2a on a mesh with $n = 257$.

Table 2
Example 2 – fourth order numerical error analysis of Ray-casting vs Cartesian AMIB schemes with $k = 5$ and $tol = 10^{-10}$.

$[n_x, n_y]$	Ray-casting AMIB					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	9.830E-6	–	3.609E-5	–	44	9.375E-2
[129, 129]	1.987E-7	5.6285	6.115E-7	5.8831	53	0.359
[257, 257]	1.335E-8	3.8957	3.339E-8	4.1949	62	1.67
[513, 513]	7.820E-10	4.0935	1.617E-9	4.3680	94	10.3
$[n_x, n_y]$	Cartesian AMIB					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	2.549E-6	–	5.752E-6	–	57	0.125
[129, 129]	1.721E-7	3.8886	3.815E-7	3.9143	82	0.516
[257, 257]	1.107E-8	3.9585	2.419E-8	3.9792	108	2.75
[513, 513]	7.089E-10	3.9649	1.540E-9	3.9734	132	14.27

AMIB4 is less expensive than that of the MIB4. In particular, on mesh 513 by 513, the AMIB4 is about 23 times faster than the MIB4.

In the case of Dirichlet boundary conditions, the numerical results show that the AMIB4 is still faster than the MIB4. However, the iteration number of the AMIB4 scheme becomes lager compared with that of Robin boundary conditions. Such an increment is related to the issue discussed in section 2.2, when some grid points are very close to the interface Γ. Referring to [23], a good rule of thumb is to shift the interpolation towards the inside, so that the grid point very close the interface is not used. In this paper, we apply the shifting treatment for both AMIB4 and MIB4. Following section 2.2, the shifting condition is chosen to be $\Theta < 0.2$ in this example. Table 1 shows the numerical results after shifting. We note that the growth rate of iteration number of AMIB4 with respect to mesh size n is still moderate, so that the efficiency of the AMIB4 is not reduced.

Example 2. This example is devoted to the comparison on numerical accuracy of the proposed AMIB method with fictitious values being generated along Cartesian direction [51] (Cartesian scheme) and normal direction (ray-casting scheme) for solving a $2D$ Poisson's equation

$$u_{xx} + u_{yy} = -2k^2 \sin(kx)\cos(ky),$$

with a circular interface defined by $\Gamma : x^2 + y^2 = 0.5^2$ in $D : [-\pi/3, \pi/3] \times [-\pi/3, \pi/3]$, subject to Dirichlet boundary condition on Γ. Similar to Example 1, shifting the grid points is necessary in dealing with Dirichlet boundary conditions. The exact solution to this problem is $u(x, y) = \sin(kx)\cos(ky)$, in which the wave number k is a free parameter.

By taking $k = 5$, Table 2 indicates that both Cartesian and ray-casting AMIB schemes produce a fourth order of convergence, and the Cartesian AMIB scheme is slightly more accurate. As can be seen in Fig. 8, the ray-casting AMIB scheme produces some error spikes around Γ, while the error pattern of the Cartesian AMIB is pretty regular. Nevertheless, Table 2 shows that the ray-casting scheme is more efficient than the classic Cartesian method. One motivation of designing ray-casting fictitious values is due to the unavailability of grid points to generate a fictitious point along Cartesian direction.

This example is also employed to validate the high-order convergence of our new method with the above circular interface Γ in $D : [-2\pi/3, 2\pi/3] \times [-2\pi/3, 2\pi/3]$. The parameter k is uniformly set to be 7 for testing the AMIB4, AMIB6, and

Table 3
Example 2 – high order numerical error analysis with the Robin boundary condition and $k = 7$. Here $tol = 10^{-10}$ for AMIB4 and AMIB6, and $tol = 10^{-12}$ for AMIB8.

$[n_x, n_y]$	AMIB4					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	1.654E-2	–	3.815E-2	–	15	3.125E-2
[129, 129]	4.227E-4	5.2902	8.818E-4	5.4351	18	0.141
[257, 257]	1.590E-5	4.7325	3.271E-5	4.7526	19	0.56
[513, 513]	6.497E-7	4.6131	1.848E-6	4.1457	22	2.67
$[n_x, n_y]$	AMIB6					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	7.531E-3	–	1.763E-2	–	17	6.25E-2
[129, 129]	5.937E-5	6.9870	1.438E-4	6.9378	20	0.203
[257, 257]	6.245E-7	6.5709	1.266E-6	6.8276	24	0.844
[513, 513]	3.134E-9	7.6386	7.197E-9	7.4587	30	3.953
$[n_x, n_y]$	AMIB8					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	1.379E-3	–	3.604E-3	–	18	7.8E-2
[129, 129]	6.545E-6	7.7190	1.821E-5	7.6287	26	0.297
[257, 257]	2.924E-8	7.8063	6.228E-8	8.1917	42	1.422
[513, 513]	3.792E-11	9.5908	3.611E-10	7.4302	59	7.375

Table 4
Example 3 – Fourth order numerical error analysis for mixed boundary conditions with $tol = 10^{-10}$.

$[n_x, n_y]$	AMIB					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	6.228E-5	–	2.419E-4	–	18	4.688E-2
[129, 129]	3.004E-6	4.3738	8.560E-6	4.8207	22	0.172
[257, 257]	2.047E-7	3.8753	7.461E-7	3.5202	24	0.781
[513, 513]	9.720E-9	4.3964	2.997E-8	4.6378	25	3.344

AMIB8. Here the Robin boundary condition is assumed. It is seen from Table 3 that the AMIB4, AMIB6, and AMIB8 achieves, separately, the fourth, sixth and eighth order of convergence. The AMIB8 is the most accurate among the methods.

Example 3. We consider a complex domain Ω^- in $D : [-\pi/3, \pi/3] \times [-\pi/3, \pi/3]$, which is generated as the union of two disks. In particular, the embedded interface Γ consists of two pieces: $\Gamma_1 = \{(x, y), x < 0, (x + 0.2)^2 + y^2 = 0.5^2\}$ and $\Gamma_2 = \{(x, y), x > 0, (x - 0.2)^2 + y^2 = 0.5^2\}$. A 2D elliptic equation is solved in Ω^-

$$u_{xx} + u_{yy} - k^2 u = -3k^2 \sin(kx)\cos(ky),$$

subject to mixed Robin and Neumann boundary conditions:

$$u + \frac{\partial u}{\partial n} = \sin(kx)\cos(ky) + k\cos(kx)\cos(ky)\hat{n}_x - k\sin(kx)\sin(ky)\hat{n}_y \quad \text{on } \Gamma_1,$$
$$\frac{\partial u}{\partial n} = k\cos(kx)\cos(ky)\hat{n}_x - k\sin(kx)\sin(ky)\hat{n}_y \quad \text{on } \Gamma_2,$$

where $(\hat{n}_x, \hat{n}_y)$ determines the normal direction. The exact solution is prescribed as $u(x, y) = \sin(kx)\cos(ky)$ with $k = 5$.

By considering mixed boundary conditions, the numerical results of the AMIB4 are listed in Table 4. In terms of accuracy and efficiency, the present results are very close to those in Example 1 in case of Robin boundary conditions. Over rectangular domains, the AMIB scheme [17] can also handle mixed boundary conditions. Table 4 shows that the ray-casting AMIB4 method is equally successful in dealing with mixed boundary conditions over irregular domains. In fact, the present ray-casting AMIB4 method is capable of handing any type of general boundary conditions.

The numerical solution and error of the AMIB method for the Example 3 are depicted in Fig. 9. It can be seen that the boundary Γ is non-smooth at two joint points of two circles. Note that the AIMB scheme is designed for C^1 continuous Γ. Fortunately, these two cusps are not severe in this example, so that the AMIB method can still deliver the desired order of accuracy. If the distance between the centers of two disks is larger, the geometric singularity at two cusps becomes severe.

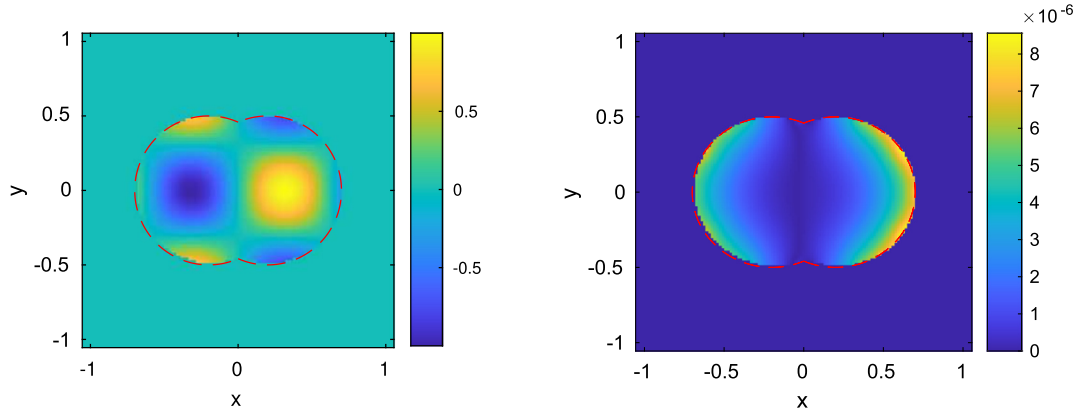

Fig. 9. The numerical solution (left) and error (right) of Example 3 on a mesh with $n = 129$. Red dash lines represent boundaries.

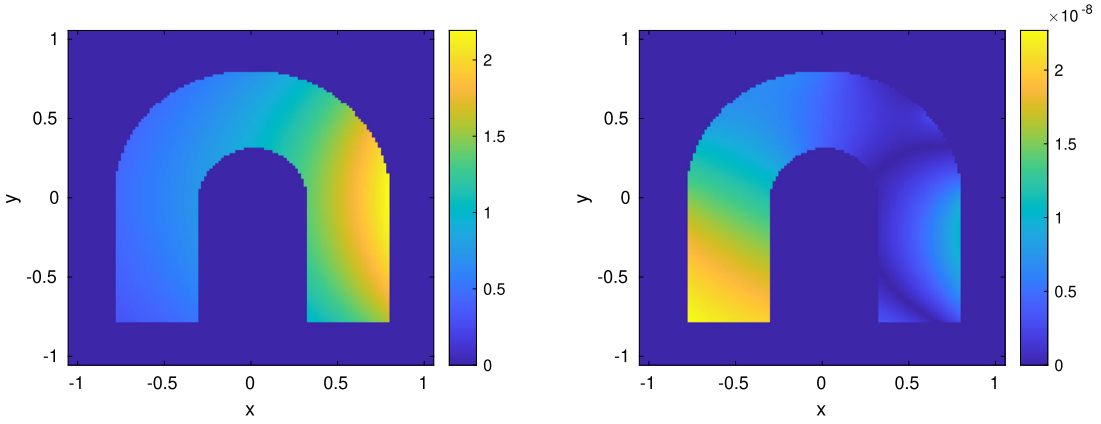

Fig. 10. The numerical solution (left) and error (right) of Example 4 on a mesh with $n = 129$.

Then, the fourth order convergence is impossible to be accomplished for the AMIB or any other known methods. In such a case, a second order of accuracy can still be maintained by using the original MIB method, see [47].

Example 4. We consider an arch domain Ω^- embedded in a computation domain $D : [-\pi/3, \pi/3] \times [-\pi/3, \pi/3]$. Here the boundary Γ involves two half-circles $\Gamma_1 = \{(x, y), y = \sqrt{0.3^2 - x^2}\}$ and $\Gamma_2 = \{(x, y), y = \sqrt{0.8^2 - x^2}\}$. The other pieces of Γ are the sides of two base columns, which are rectangles $[-0.8, -0.3] \times [-0.8, 0]$ and $[0.3, 0.8] \times [-0.8, 0]$. See Fig. 10. A 2D Laplace's equation is solved in Ω^-

$$u_{xx} + u_{yy} = 0,$$

subject to the mixed Robin and Neumann boundary conditions:

$$u + \frac{\partial u}{\partial n} = e^x \cos(y) + e^x \cos(y)\hat{n}_x - e^x \sin(y)\hat{n}_y \quad \text{on } \Gamma \cap \{(x, y), x > 0\},$$
$$\frac{\partial u}{\partial n} = e^x \cos(y)\hat{n}_x - e^x \sin(y)\hat{n}_y \quad \text{on } \Gamma \cap \{(x, y), x < 0\},$$

which are determined by analytical solution $u(x, y) = e^x \cos(y)$ within Ω^-. The normal direction is defined by $(\hat{n}_x, \hat{n}_y)$. For a comparison, we also study the same BVP by replacing the Neumann boundary condition with the Dirichlet boundary condition for $x < 0$.

The numerical results are summarized in Table 5. Similar to the previous study, the impact of Dirichelt boundary conditions can be clearly observed in Table 5. The iteration number of the AMIB4 becomes larger when Neumann boundary conditions are replaced with Dirichlet boundary conditions in the mixed setting. Here the shifting condition $\Theta < 0.2$ is also employed in case of the Dirichlet boundary condition.

Table 5
Example 4 – Fourth order numerical error analysis for the arch domain with $tol = 10^{-13}$.

$[n_x, n_y]$	mixed Robin and Neumann					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	2.146E-7	–	4.246e-7	–	38	3.125E-2
[129, 129]	9.941E-9	4.4321	2.268E-8	4.2266	48	0.4375
[257, 257]	7.501E-10	3.7282	1.573E-9	3.8498	49	1.5
[513, 513]	1.5070E-11	5.6335	3.057E-11	5.6853	44	8.98
$[n_x, n_y]$	mixed Robin and Dirichlet					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	4.288E-8	–	1.216E-7	–	86	0.266
[129, 129]	2.629E-9	4.0277	1.089E-8	3.4811	120	0.875
[257, 257]	1.661E-10	3.9844	5.770E-10	4.2383	201	8.656
[513, 513]	3.341E-12	5.6356	1.427E-11	5.3375	215	48.61

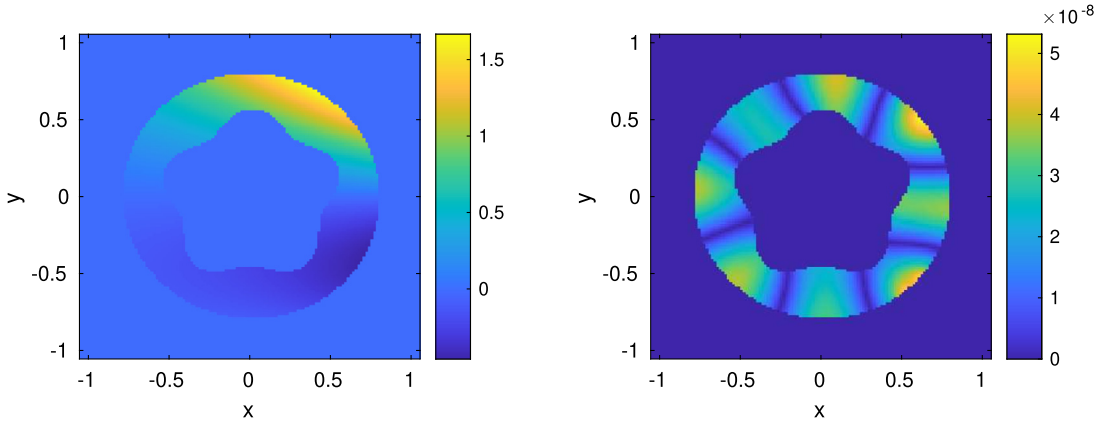

Fig. 11. The numerical solution (left) and error (right) of Example 5 on a mesh with $n = 129$.

Example 5. We explore the performance of the ray-casting AMIB for a more complicated geometry. Restricted in a domain $D : [-\pi/3, \pi/3] \times [-\pi/3, \pi/3]$, the interface Γ is composed of $\Gamma_1 : r = 0.5 + 1/18 \sin(5\theta)$ and $\Gamma_2 : r = 0.8$. See Fig. 11. Here, the Helmholtz equation is given as

$$u_{xx} + u_{yy} + k^2 u = e^x(k^2(\sin(y) + y^2) + y^2 + 2),$$

with the exact solutions $u(x, y) = e^x(\sin(y) + y^2)$. With the parameter k being set as 6, it is subject to mixed Robin and Neumann boundary conditions:

$$u + \frac{\partial u}{\partial n} = e^x(\sin(y) + y^2) + e^x(\sin(y) + y^2)\hat{n}_x + e^x(\cos(y) + 2y)\hat{n}_y \quad \text{on } \Gamma_1,$$
$$\frac{\partial u}{\partial n} = e^x(\sin(y) + y^2)\hat{n}_x + e^x(\cos(y) + 2y)\hat{n}_y \quad \text{on } \Gamma_2,$$

where $(\hat{n}_x, \hat{n}_y)$ represents the normal direction. Table 6 shows that the AMIB4 method is equally capable of solving the Helmholtz equation with mixed boundary conditions. The fourth order convergences are again confirmed. The efficiency of the AMIB4 is further validated.

Example 6. We consider an external problem, where the artificial solution u^- within the internal domain Ω^- is 0, and the physical solution is defined over Ω^+ with an expression $u^+(x, y) = e^{x+y}$. Here $D = \Omega^- \cup \Omega^+ = [-\pi/3, \pi/3] \times [-\pi/3, \pi/3]$ and the interior boundary is a three-leaf shaped interface $\Gamma_1 : r = 0.6 + 1/12 \sin(3\theta)$, and the exterior boundary Γ_2 consists of four sides of D. The Helmholtz equation is solved over Ω^+

$$u_{xx} + u_{yy} + k^2 u = (2 + k^2)e^{x+y},$$

where $k = 5$. The Robin boundary condition is assumed across the interface Γ_1, and Dirichlet boundary conditions are used on Γ_2.

Table 6
Example 5 – Fourth order numerical error analysis for two types of mixed boundary conditions with $tol = 10^{-10}$.

$[n_x, n_y]$	mixed Robin and Neumann					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	3.402E-7	–	7.897E-7	–	34	9.375E-2
[129, 129]	2.292E-8	3.8917	5.313E-8	3.8937	39	0.344
[257, 257]	1.552E-9	3.8844	3.461E-9	3.9403	43	1.375
[513, 513]	1.133E-10	3.7759	2.381E-10	3.8615	40	5.266

Table 7
Example 6 – Fourth order numerical error analysis for the case where $u^- = 0$ with $tol = 10^{-10}$.

$[n_x, n_y]$	AMIB4					
	L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[65, 65]	9.760E-7	–	3.187E-6	–	37	7.8125E-2
[129, 129]	4.604E-8	4.4059	1.528E-7	4.3825	32	0.281
[257, 257]	2.568E-9	4.1642	8.261E-9	4.2092	36	1.109
[513, 513]	1.547E-10	4.0531	5.136E-10	4.0076	34	4.453

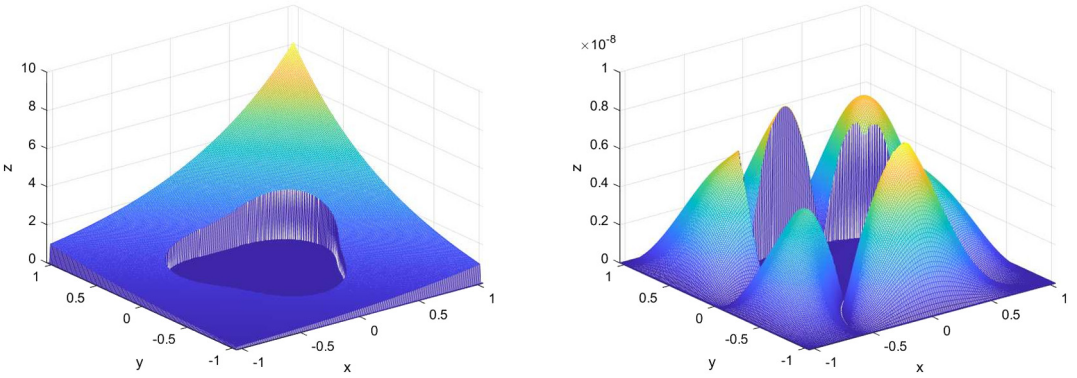

Fig. 12. The numerical solution (left) and error (right) of Example 6 on a mesh with $n = 257$.

As mentioned in the previous section, in order to satisfy the anti-symmetric property across Γ_2, we will regard Γ_2 as an embedded interface too. A zero padding zone is needed in an extended domain Ω^e outside D, see [17] for more details. For achieving fourth order of accuracy, we need to equip the extended region Ω^e with a width equaling to $2h_x$ or $2h_y$ for x- or y-direction outside the domain D. In treating Dirichlet boundary conditions over Γ_2, the mesh size for the entire domain $\Omega = \Omega^- \cup \Omega^+ \cup \Omega^e$ is taken as (n + 5) by (n + 5). A systematic approach is constructed at both Γ_1 and Γ_2 to build the fourth order central difference and a uniform augmentation framework is utilized to accelerate the computation at both Γ_1 and Γ_2. The present problem has some similarity to the elliptic interface problems studied in [18]. Nevertheless, this example considers the Robin boundary condition for the interior interface Γ_1 instead of jump conditions, which is different from [18].

For the present example, the fourth order convergence of the AMIB4 is numerically verified through Table 7. Again, the efficiency of the AMIB4 is further validated. In particular, on a mesh 513 by 513, the execution time of the AMIB4 is only 4.453 seconds. The numerical solution and error are depicted in Fig. 12.

3.2. Numerical example in 3D

Example 7. We consider a 3D example in this subsection. Consider a Poisson equation

$$u_{xx} + u_{yy} + u_{zz} = -3k^2 \sin(kx)\cos(ky)\sin(kz),$$

over a spherical domain with boundary Γ being $r^2 : x^2 + y^2 + z^2 = 0.60^2$. The Robin boundary condition is employed, which is calculated by the exact solution $u(x, y) = \sin(kx)\cos(ky)\sin(kz)$. Here, the parameter k is set to be 2 in this 3D test. The computational domain is $D = [-2\pi/3, 2\pi/3] \times [-2\pi/3, 2\pi/3] \times [-2\pi/3, 2\pi/3]$.

The comparison of numerical results of AMIB4 with those of MIB4 are presented in Table 8. Obviously, both AMIB4 and MIB4 for 3D Poisson problem can achieve fourth order convergence. Similar to the Example 1 in 2D, the AMIB method is well conditioned such that the iteration number only weakly depends on the mesh size n.

Table 8
Example 7 – Fourth order numerical error analysis of the AMIB4 vs MIB4 on 3D Poisson's equation with $tol = 10^{-10}$.

$[n_x, n_y, n_z]$	AMIB L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[33, 33, 33]	6.302E-4	–	2.159E-3	–	9	0.491
[65, 65, 65]	4.710E-5	3.9221	1.882E-4	3.5200	15	3.75
[129, 129, 129]	3.107E-6	3.9221	1.229E-5	3.9367	17	39.00
[257, 257, 257]	1.425E-7	4.4465	5.851E-7	4.3927	19	344.12
$[n_x, n_y, n_z]$	MIB L_2		L_∞		iter no.	CPU time (s)
	Error	Order	Error	Order		
[33, 33, 33]	8.881E-5	–	2.160E-3	–	67	0.272
[65, 65, 65]	5.885E-6	3.9156	1.882E-4	3.5207	101	2.686
[129, 129, 129]	3.676E-7	4.0008	1.229E-5	3.9367	165	39.573
[257, 257, 257]	1.639E-8	4.4872	5.849E-7	4.3932	266	514.141

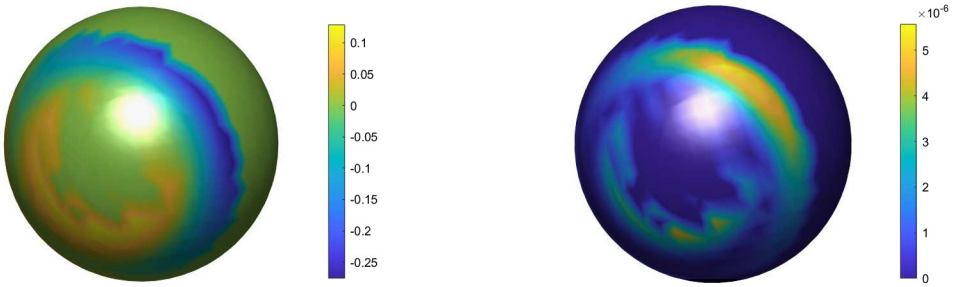

Fig. 13. The numerical solution (left) and error (right) of Example 7 on a mesh with $n = 129$.

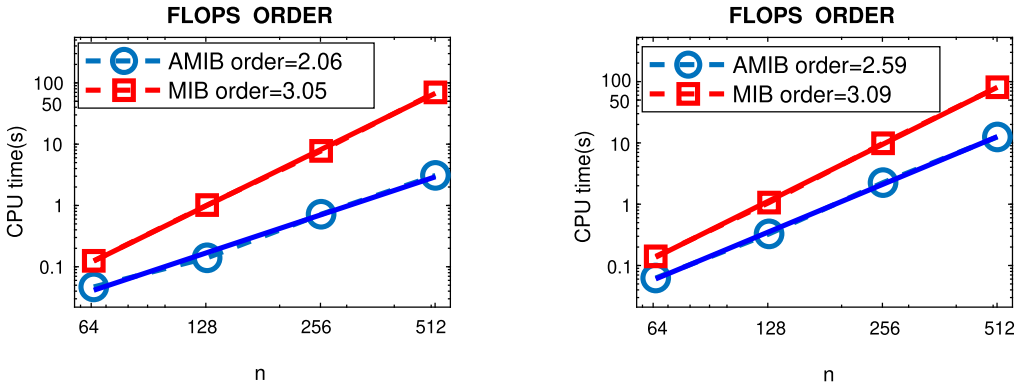

Fig. 14. Flops order in CPU time is examined for AMIB4 and MIB4 in 2D with Robin boundary conditions (left) and Dirichlet boundary conditions (right) in Example 1. Here n represents the degree of freedom in each direction.

In Fig. 13, surface plots are shown, in which the numerical solution and error are projected to the sphere. It can be seen that larger errors usually occur at regions where the solution changes rapidly.

3.3. Computational efficiency

In this subsection, we would like to investigate the computational efficiency of the AMIB method. Based on the computational time of the AMIB4 and MIB4 methods in Table 1 and Table 8, the complexity of both methods can be numerically quantified. For this purpose, Fig. 14 compares the computational efficiency of AMIB4 and MIB4 in 2D in case of Robin and Dirichlet boundary conditions by plotting CPU time versus n in a log-log manner. For simplicity, we regard the degree of freedom in each direction as n. A least squares fitting is conducted to express the CPU time in the form of n^r. It can be observed that the complexity of the MIB4 is at least $O(n^3)$. For the AMIB4 scheme, such r value is slightly above 2 in case of Robin boundary conditions, whereas the ray-casting AMIB4 method becomes slow in case of Dirichlet boundary conditions. Similarly, Fig. 15 compares the computational efficiency of AMIB4 and MIB4 for 3D BVPs with Robin boundary conditions.

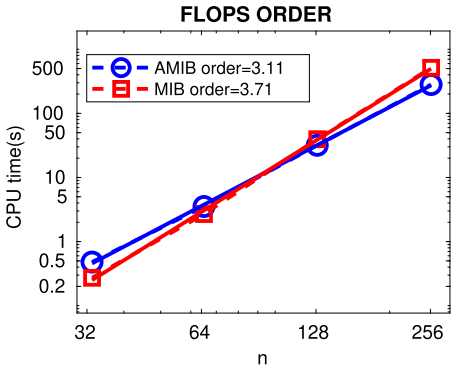

Fig. 15. Flops order in CPU time is examined for AMIB4 and MIB4 in 3D with Robin boundary conditions.

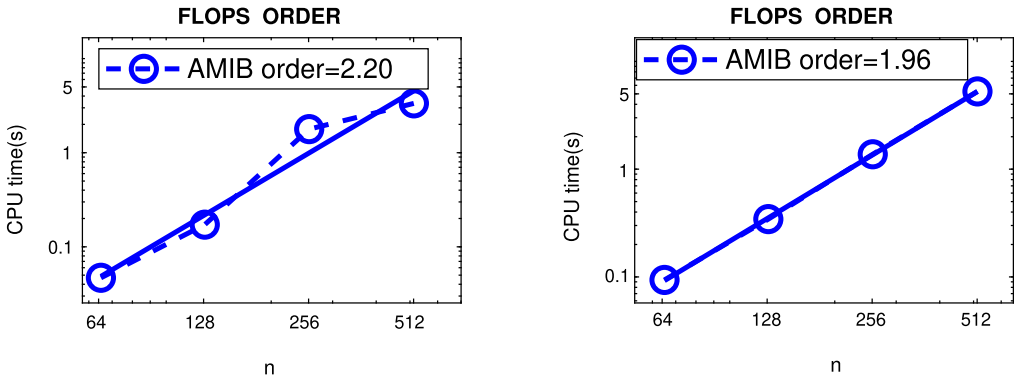

Fig. 16. Flops order in CPU time is examined for Example 3 (left) and Example 5 (right). Flop orders are tested to be around 2 for each example.

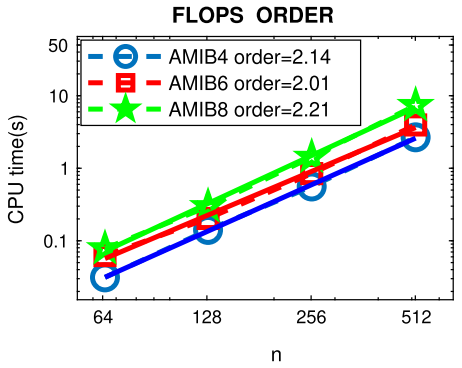

Fig. 17. A comparison of computational cost between three methods, and flop orders are tested to be around 2 for each method.

The r value is slightly above 3 for AMIB scheme and close to 4 for MIB scheme. In Fig. 16, the computational efficiencies of the AMIB4 for complicated 2D BVPs with mixed boundary conditions are studied. The flops orders are all around 2. In summary, in treating most boundary conditions, the complexity of the ray-casting AMIB scheme is about $O(n^d \log n)$ where d is the dimension number. Such complexity order becomes larger when Dirichlet boundary conditions are dealt with.

We next examine the cost-efficiency. From Table 3, we can see that AMIB4, AMIB6, and AMIB8 produce fourth, sixth, and eighth order accuracy, respectively. To analyze the flops order, the CPU time is plotted against the degree of freedom n in each dimension in Fig. 17. For a given n, the higher order accuracy achieved, the more expensive AMIB algorithm is. From Fig. 17, it is clear that the flop order r for each method is around 2, which again verifies the expected complexity of $O(n^2 \log n)$ for our designed algorithm. On the other hand, it is obvious from Table 3 that the AMIB6 is more accurate than the fourth order method, while the AMIB8 is the most accurate method among them. To examine the cost-efficiency, we plot the accuracy against CPU time in Fig. 18 for both error norms. For log-log lines shown in Fig. 18, we calculated the slopes by the least squares fitting. For instance, the slope of AMIB4 is around -2, while that of AMIB8 is about -3.5 under L_∞. Based on these data, it is predicted that to achieve an accuracy around 10^{-10}, it would take AMIB4 about 105

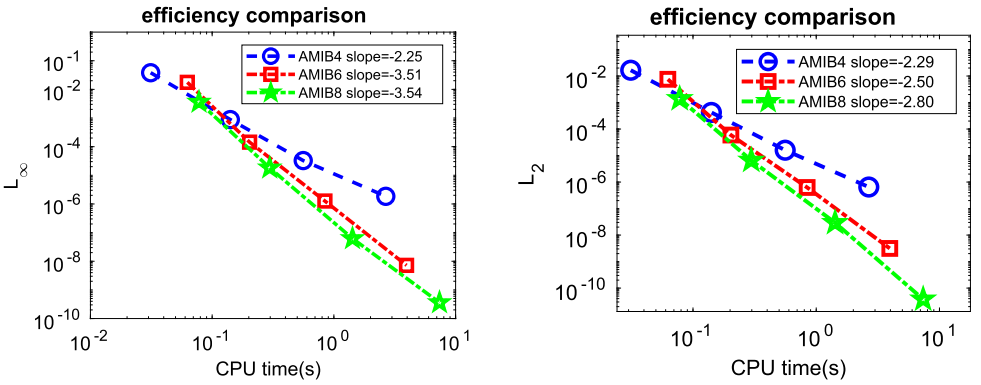

Fig. 18. A comparison between the three methods on accuracy vs CPU time under two norms.

seconds of CPU time provided that memory is large enough, which is about 15 times more expensive than the proposed AMIB8 scheme. The present study indicates that the higher order schemes are more cost-efficient.

4. Conclusion

In this work, an accurate and efficient finite difference method is proposed to solve elliptic BVPs with irregular domains and complex boundary conditions. The proposed ray-casting AMIB method is built based on our previous developments, including the MIB boundary method [48,50,51], the ray-casting MIB boundary method [49], and the augmented formulation for interface and boundary problems [16–19]. The proposed ray-casting MIB scheme is able to implement different types of boundary conditions, including Dirichlet, Neumann, Robin, and their mix combinations, in high order central difference discretizations. Moreover, in the augmented MIB formulation, the FFT algorithm will not sense the solution discontinuities, so that the discrete Laplacian can be efficiently inverted. In numerical tests with irregular domains, the proposed ray-casting AMIB method not only achieves a fourth order of accuracy, but also maintains the FFT efficiency, in treating Neumann and Robin boundary conditions. For the Dirichlet boundary, since the condition itself does not involve the normal direction, the complexity of the ray-casting AMIB method becomes larger. But, the ray-casting AMIB method is still faster than the Cartesian AMIB method for Dirichlet BVPs. When the domain is not too complicated, the order of accuracy of the ray-casting AMIB method can be improved to six or eight.

In the future, we will explore potential approaches to control the condition number of the augmented system so that the iterative solution in the Schur complement procedure could be faster. The generalization of the ray-casting MIB method for solving BVPs of vector Laplacian problems is under our investigation.

CRediT authorship contribution statement

Yiming Ren: Methodology, Software, Validation, Visualization, Writing – original draft. **Hongsong Feng:** Methodology, Software, Writing – review & editing. **Shan Zhao:** Conceptualization, Methodology, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research is partially supported by the National Science Foundation (NSF) under grants DMS-1812930 and DMS-2110914.

References

[1] L. Adams, Z.L. Li, The immersed interface/multigrid methods for interface problems, SIAM J. Sci. Comput. 24 (2002) 463–479.
[2] V. Arias, D. Bochkov, F. Gibou, Poisson equations in irregular domains with Robin boundary conditions - solver with second-order accurate gradients, J. Comput. Phys. 365 (2018) 1–6.
[3] T. Askham, A.J. Cerfon, An adaptive fast multipole accelerated Poisson solver for complex geometries, J. Comput. Phys. 344 (2017) 1–22.
[4] A. Averbuch, M. Israeli, L. Vozovoi, A fast Poisson solver of arbitrary order accuracy in rectangular regions, SIAM J. Sci. Comput. 19 (1998) 933–952.
[5] J. Bedrossian, J.H. von Brecht, S.W. Zhu, E. Sifakis, J.M. Teran, A finite element method for interface problems in domains with smooth boundaries and interfaces, J. Comput. Phys. 229 (2010) 6405–6426.

[6] D. Bochkov, F. Gibou, Solving Poisson-type equations with Robin boundary conditions on piecewise smooth interfaces, J. Comput. Phys. 376 (2019) 1156–1198.
[7] R.F. Boisvert, A fourth order accurate Fourier method for the Helmholtz equation in three dimensions, ACM Trans. Math. Softw. 13 (1987) 221–234.
[8] E. Braverman, M. Israeli, A. Averbuch, L. Vozovoi, A fast 3D Poisson solver of arbitrary order accuracy, J. Comput. Phys. 144 (1998) 109–136.
[9] O. Bruno, Fast, high-order, high-frequency integral methods for computational acoustics and electromagnetics, in: M. Ainsworth, P. Davies, D. Duncan, P. Martin, B. Rynne (Eds.), Topics in Computational Wave Propagation Direct and Inverse Problems Series, in: Lecture Notes in Computational Science and Engineering, vol. 31, 2003, pp. 43–82.
[10] O. Bruno, Y. Han, M. Pohlman, Accurate, high-order representation of complex three-dimensional surfaces via Fourier-continuation analysis, J. Comput. Phys. 227 (2007) 1094–1125.
[11] O. Bruno, M. Lyon, High-order unconditionally stable FC-AD solvers for general smooth domains I. Basic elements, J. Comput. Phys. 229 (2010) 2009–2033.
[12] O. Bruno, T. Elling, A. Sen, A Fourier continuation method for the solution of elliptic eigenvalue problems in general domains, Math. Probl. Eng. 2015 (2015) 184786.
[13] T. Chen, J. Strang, Piecewise-polynomial discretization and Krylov-accelerated multigrid for elliptic interface problems, J. Comput. Phys. 227 (2008) 7503–7542.
[14] H. Chen, C. Min, F. Gibou, A supra-convergent finite difference scheme for the Poisson and heat equations on irregular domains and non-graded adaptive Cartesian grids, J. Sci. Comput. 31 (2007) 19–60.
[15] A. Coco, G. Russo, Finite-difference ghost-point multigrid methods on Cartesian grids for elliptic problems in arbitrary domains, J. Comput. Phys. 241 (2013) 464–501.
[16] H. Feng, G. Long, S. Zhao, An augmented matched interface and boundary (MIB) method for solving elliptic interface problem, J. Comput. Appl. Math. 361 (2019) 426–443.
[17] H. Feng, S. Zhao, FFT-based high order central difference schemes for three-dimensional Poisson's equation with various types of boundary conditions, J. Comput. Phys. 410 (2020) 109391.
[18] H. Feng, S. Zhao, A fourth order finite difference method for solving elliptic interface problems with the FFT acceleration, J. Comput. Phys. 419 (2020) 109677.
[19] H. Feng, G. Long, S. Zhao, FFT-based high order central difference schemes for Poisson's equation with staggered boundaries, J. Sci. Comput. 86 (2021) 7.
[20] B. Fornberg, Classroom note: calculation of weights in finite difference formulas, SIAM Rev. 40 (1998) 685–691.
[21] Y. Ge, Multigrid method and fourth-order compact difference discretization scheme with unequal meshsizes for 3D Poisson equation, J. Comput. Phys. 229 (2010) 6381–6391.
[22] F. Gibou, R. Fedkiw, A second order accurate symmetric discretization of the Poisson equation on irregular domains, J. Comput. Phys. 176 (2003) 1–23.
[23] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problems, J. Comput. Phys. 202 (2005) 577–601.
[24] F. Gibou, C. Min, R. Fedkiw, High resolution sharp computational methods for elliptic and parabolic problems in complex geometries, J. Sci. Comput. 54 (2013) 369–413.
[25] T. Gillis, G. Winckelmans, P. Chatelain, Fast immersed interface Poisson solver for 3D unbounded problems around arbitrary geometries, J. Comput. Phys. 354 (2018) 403–416.
[26] L. Greengard, J-Y. Lee, A direct adaptive Poisson solver of arbitrary order accuracy, J. Comput. Phys. 125 (1996) 415–424, https://doi.org/10.1006/jcph.1996.0103.
[27] Y. Gu, J. Shen, Accurate and efficient spectral method for elliptic PDEs in complex domains, J. Sci. Comput. 83 (2020) 42.
[28] M.M. Gupta, J. Kouatchou, J. Zhang, Comparison of second and fourth order discretization multigrid Poisson solvers, J. Comput. Phys. 132 (1997) 226–232.
[29] H. Johansen, P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85.
[30] Z. Jomaa, C. Macaskill, The embedded finite difference method for the Poisson equation in a domain with an irregular boundary and Dirichlet boundary conditions, J. Comput. Phys. 202 (2005) 488–506.
[31] C. Kublik, N.M. Tanushev, R. Tsai, An implicit interface boundary integral method for Poisson's equation on arbitrary domains, J. Comput. Phys. 247 (2013) 279–311.
[32] M.C. Lai, A simple compact fourth-order Poisson solver on polar geometry, J. Comput. Phys. 182 (2002) 337–345.
[33] Z.L. Li, A fast iterative algorithm for elliptic interface problem, SIAM J. Numer. Anal. 35 (1998) 230–254.
[34] Z.L. Li, H. Ji, X. Chen, Accurate solution and gradient computation for elliptic interface problems with variable coefficients, SIAM J. Numer. Anal. 55 (2017) 670–697.
[35] X.D. Liu, R.P. Fedkiw, M. Kang, Boundary condition capturing method for Poisson's equation on irregular domain, J. Comput. Phys. 160 (2000) 151–178.
[36] A.N. Marques, J.-C. Nave, R.R. Rosales, High order solution of Poisson problems with piecewise constant coefficients and interface jumps, J. Comput. Phys. 335 (2017) 497–515.
[37] A. McKenney, L. Greengard, A. Mayo, A fast Poisson solver for complex geometries, J. Comput. Phys. 100 (1992) 236.
[38] J. Papac, F. Gibou, C. Ratsch, Efficient symmetric discretization for the Poisson, heat and Stefan-type problems with Robin boundary conditions, J. Comput. Phys. 229 (2010) 875–889.
[39] V. Pereyra, W. Preskurowski, O. Widlund, High order fast Laplace solvers for the Dirichlet problem on general regions, Math. Comput. 31 (1997) 1–16.
[40] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in Fortran: The Art of Scientific Computing, 2nd edition, Cambridge University Press, 1992.
[41] D.B. Stein, R.D. Guy, B. Thomases, Immersed boundary smooth extension: a high-order method for solving PDE on arbitrary smooth domains using Fourier spectral methods, J. Comput. Phys. 304 (2016) 252–274.
[42] X.H. Sun, Y. Zhuang, A High-Order Direct Solver for Helmholtz Equations with Neumann Boundary Conditions, Technical Report, Institute for Computer Applications in Science and Engineering (ICASE), 1997.
[43] J. Towers, A source term method for Poisson problems on irregular domains, J. Comput. Phys. 361 (2018) 424–441.
[44] Y. Wang, J. Zhang, Sixth order compact scheme combined with multigrid method and extrapolation technique for 2D Poisson equation, J. Comput. Phys. 228 (2009) 137–146.
[45] 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 (2004) 827–862.
[46] Y. Xie, W. Ying, A fourth-order kernel-free boundary integral method for the modified Helmholtz equation, J. Sci. Comput. 78 (2019) 1632–1658.
[47] S. Yu, Y. Zhou, G.W. Wei, Matched interface and boundary (MIB) method for elliptic problems with sharp-edged interfaces, J. Comput. Phys. 224 (2010) 729–756.
[48] S. Zhao, On the spurious solutions in the high-order finite difference methods, Comput. Methods Appl. Math. 196 (2007) 5031–5046.
[49] S. Zhao, A fourth order finite difference method for waveguides with curved perfectly conducting boundaries, Comput. Methods Appl. Math. 199 (2010) 2655–2662.

[50] 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.
[51] S. Zhao, G.W. Wei, Matched interface and boundary (MIB) for the implementation of boundary conditions in high order central finite differences, Int. J. Numer. Methods Eng. 77 (2009) 1690–1730.