Journal of Computational Physics 410 (2020) 109391

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

FFT-based high order central difference schemes for three-dimensional Poisson's equation with various types of boundary conditions

Hongsong Feng, Shan Zhao [*]

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

A R T I C L E   I N F O

Article history:
Received 16 April 2019
Received in revised form 19 February 2020
Accepted 4 March 2020
Available online 7 March 2020

Keywords:
Fast Poisson solver
Fast Fourier transform (FFT)
High order central difference schemes
Complex boundary conditions
Matched interface and boundary method

A B S T R A C T

In this paper, a unified approach is introduced to implement high order central difference schemes for solving Poisson's equation via the fast Fourier transform (FFT). Popular high order fast Poisson solvers in the literature include compact finite differences and spectral methods. However, FFT-based high order central difference schemes have never been developed for Poisson problems, because with long stencils, central differences require fictitious nodes outside the boundary, which poses a challenge to integrate boundary conditions in FFT computations. To overcome this difficulty, several layers of exterior grid lines are introduced to convert the problem to an immersed boundary problem with zero-padding solutions beyond the original cubic domain. Over the boundary of the enlarged cubic domain, the anti-symmetric property is naturally satisfied so that the FFT fast inversion is feasible, while the immersed boundary problem can be efficiently solved by the proposed augmented matched interface and boundary (AMIB) method. As the first fast Poisson solver based on high order central differences, the AMIB method can be easily implemented in any dimension, due to its tensor product nature of the discretization. As a systematical approach, the AMIB method can be made to arbitrarily high order in principle, and can handle the Dirichlet, Neumann, Robin or any combination of boundary conditions. The accuracy, efficiency, and robustness of the proposed AMIB method are numerically validated by considering various Poisson problems in two and three dimensions.

1. Introduction

Poisson's equation as a simple partial differential equation (PDE) plays a crucial role in the study of electrostatics, mechanical engineering and theoretical physics. Its numerical solution draws extensive attentions as it has many generalized forms [30] in practical utility. Efficient algorithm for the Poisson solution is desired, otherwise it may take the majority of computation time in many applications.

Fast Fourier transform (FFT) is one of the most successful fast Poisson solvers. On a spatial degree of freedom N in multi-dimensions, generic iterative solvers would require a complexity of order $O(N^2)$ in solving the sparse system. For a second order central difference discretization to a Poisson problem, FFT provides a solver of complexity $O(N \log N)$, which

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

https://doi.org/10.1016/j.jcp.2020.109391
0021-9991/© 2020 Elsevier Inc. All rights reserved.

results in a significant acceleration. The FFT solver came into being when Hockney [20] used a Fourier method to solve Poisson's equation. Since then, FFT has been further investigated in various situations and applications. Fourier transform algorithms [38] need to be modified for different boundary conditions and discretization on standard or staggered grid lines. With Fourier analysis and synthesis well formulated for certain problems, pre- and post-processing algorithms [38] have been constructed to convert non-periodic transforms into a periodic form, which fulfills FFT with the help of available software. Another approach deploys the symmetric FFTs proposed by Swartztrauber [41], which avoids pre- and post-processing stages. Both approaches enable the fast summation of Fourier analysis or synthesis. Theoretical analysis of FFT for second order accurate solution to Poisson's equation has been thoroughly studied over the past decades. Several FFT packages including FISHPACK [42] and FFTW [14] have been developed. A survey of FFT-based matrix decomposition algorithms for elliptic boundary value problems can be found in [4].

Besides FFT, other popular fast Poisson solvers include multigrid method, cyclic reduction, and FACR algorithm. The FACR algorithm [40] is a combination of cyclic reduction and Fourier analysis, and yields an asymptotic operation count that is slightly less than either cyclic reduction or FFT. Multigrid methods, particularly geometric multigrid methods, offer some of the best Poisson solvers with a computation time on the order $O(N)$, and can be applied on arbitrary regions and unstructured grids [54]. The alternating direction implicit (ADI) algorithm, originally developed for parabolic PDEs [12], can also be employed as a fast Poisson solver [23] with the operation count being $O(N)$. Typically, a fast Poisson solver refers to an algorithm with complexity being $O(N)$ or $O(N\log N)$, which is regarded as the fastest solution for Poisson problems known today.

This study is concerned with high order fast Poisson solvers, which not only maintain a complexity of $O(N)$ or $O(N\log N)$ in algebraic computations, but also deliver an order of accuracy higher than two in Poisson solution, such as four or six. High order discretizations are indispensable for problems associated with high frequency waves, such as electromagnetic and acoustic wave propagation and scattering, vibration analysis of engineering structures, and shock-vortex interaction in compressible fluid flows. To achieve the same precision, high order methods allow the use of coarse grids, whereas dense meshes have to be employed in low order methods. Thus, high order methods are cost-efficient and their combination with fast Poisson solvers could further accelerate the computational efficiency. Therefore, in recent years, there has been an increased interest in algorithms that combine high order with fast solvers.

Three major types of high order fast Poisson solvers have been developed in the literature, i.e., deferred corrections, compact finite differences, and spectral methods. Based on second order finite difference and deferred corrections, a fourth order accurate algorithm was proposed by Pereyra [31] in 1966. Later, this technique has been applied for two-dimensional (2D) Poisson problems built in the FISHPACK package. Similar works by deferred corrections were conducted for fourth order on nonrectangular planar domain [32] and sixth order of accuracy [34].

Compact finite differences have been widely employed in high order fast Poisson solvers, because only the nearest neighboring nodes are involved in the numerical differential kernel. This gives rise to a great convenience not only in handling different boundary conditions, but also in FFT or multigrid implementation. In 1978, Houstis and Papatheodorou [21] first applied multidimensional compact differences to obtain fourth order fast solution to Helmholtz-Dirichlet problem with help of a combination of cyclic reduction and Fourier analysis. Several fourth order compact finite difference schemes have been introduced in [5,37] to provide FFT solution of Helmholtz problems with Dirichlet, Neumann, or periodic boundary condition. Lai [24] generalized the fourth order compact difference scheme to a FFT Poisson solver on polar geometry. Various multigrid-based compact Poisson solvers have also been developed. In solving 2D Poisson equation with Dirichlet boundaries, a four order multigrid Poisson solver has been shown to achieve a dramatic improvement in efficiency when comparing with its counterpart in second order [16]. Multigrid implementation of fourth order compact difference with unequal mesh sizes has been considered in [17] for 3D Poisson's equation. Accelerated by Richardson extrapolation, sixth order of accuracy could be achieved in compact Poisson solvers in multigrid computations [45]. Typically, the compact finite difference scheme is directly formulated in a multi-dimensional setting, and yields a block-diagonal matrix structure for fast computation. It is pretty common that the generalization of such a compact difference from 2D to 3D requires additional works. Recently, a tensor-product style algorithm has been proposed for a fourth order compact scheme [44], and in solving Poisson's equation with Dirichlet boundaries, the same algorithm works for 2D, 3D, and even four dimensions. We also note that the implementation of Robin condition in the second order fast Poisson solver is already a tough issue, for which iterative methods are usually employed [33]. The only existing second order fast Poisson solver for dealing with Robin condition is introduced in 2D [19]. However, no extension is known in 3D or for high order compact differences.

Spectral methods are another powerful class of fast Poisson solvers. They are usually more accurate than compact finite differences, while keeping a high efficiency. A series of Fourier pseudospectral (FPS) methods [2,6,7] have been developed for solving Poisson and Helmholtz equations in multi-dimensions with Dirichlet, Neumann, or periodic boundary conditions. The FPS methods are of arbitrary order of accuracy in principle, while the practical numerical order is limited by the polynomial subtraction technique used in boundaries and corners implementation. For spectral-Galerkin methods, numerical solution of Poisson's equation can be realized via constructing modal basis functions as compact combinations of orthogonal polynomials so that boundary conditions are satisfied naturally [36]. The compact nature results in a sparse system in the frequency space, and the overall complexity of the Fourier and Chebyshev discrete transforms can be reduced to $O(N\log N)$ by using the FFT. For 1D Poisson problems, spectral-Galerkin methods provide a uniform treatment of Dirichlet, Neumann, and Robin boundary conditions. In multi-dimensional spectral-Galerkin methods, fast Poisson solvers are accomplished through the matrix decomposition/diagonalization method [18], which limits the choice of basis functions or boundary conditions in different Cartesian directions.

To the best of our knowledge, high order central difference schemes have never been applied in fast Poisson solvers, even though the standard fast Poisson solver is based on the second order central difference. This is perhaps because high order central differences have long stencils, so that fictitious points outside boundaries are inevitable. Such boundary treatments in the framework of FFT or geometrical multigrid solvers are highly non-trivial. Nevertheless, a fast Poisson solver based on high order central difference schemes enjoys several potential benefits in comparing with its counterpart based on compact finite difference or spectral methods.

First, for separable geometries, central difference approximations are carried out in a 1D manner, which can be easily generalized to multi-dimensions based on tensor-products. Sharing the same advantage, the extension of FPS methods from 2D to 3D is also straightforward [2,6,7]. Nevertheless, for most compact difference and spectral-Galerkin methods, the generalization from 2D to 3D could be nontrivial.

Second, central difference schemes usually can produce a higher order of accuracy than compact finite differences. Most of current compact differences achieve the fourth order of accuracy, while sixth order might be reached with the aid of Richardson extrapolation [45]. The central difference schemes can be systematically carried out when boundaries are appropriately treated [50,52], and can be made to arbitrarily high order in principle. Thus, the accuracy of central differences could be comparable to spectral methods.

Third, for separable geometries, the boundary treatments of central difference schemes can be processed in a 1D manner. Moreover, the treatments at left and right ends are also independent. Thus, without considering fast algebraic computations, boundary treatments become much easier than in compact finite differences and spectral/pseudospectral methods. In fact, a systematic approach for handling various general boundary conditions in arbitrarily high-order central difference schemes has been developed in the literature [50,52]. This boundary treatment is formulated based on the matched interface and boundary (MIB) method, originally developed for solving interface problems [48,53]. The MIB method can handle not only standard boundary conditions, including Dirichlet, Neumann, Robin and periodic types, but also non-standard ones, such as free-edged condition in vibration analysis [49] and perfect electric conducting condition in computation electromagnetics [51]. Another powerful boundary scheme is the inverse Lax-Wendroff (ILW) procedure developed in Shu's group for conservation laws and convection-diffusion equations [22,43,26]. Both MIB and ILW methods generate fictitious nodes outside the domain for supporting long stencils of translation-invariant central differences near boundaries. In ILW methods, this is mainly achieved by creating high order boundary conditions based on the prescribed boundary condition and the underlying PDE, while the MIB method avoids using high order boundary conditions by repeatedly enforcing one boundary condition. However, the existing boundary closure procedures of the MIB and ILW methods cannot be directly incorporated into the FFT or multigrid fast Poisson solvers.

The goal of this work is to develop high order fast Poisson solvers based on central difference schemes and MIB boundary closure [49–52]. We note that when the MIB method is simply coupled with the FFT algorithm, high order accuracy cannot be maintained near boundary. This is true even in the simplest case with a zero Dirichlet condition, i.e., $u = 0$ on boundary. Nevertheless, if the solution also satisfies the anti-symmetric property up to $(2m)$th order near both ends, the $(2m)$th order can be realized by central differences in Fourier transforms without any boundary adjustment. This motivates us to introduce a thin fictitious buffer zone or zero-padding region outside the domain so that the anti-symmetric property can be assumed up to the desired orders on the boundary of the extended domain. This converts the original boundary value problem to an immersed boundary problem. We note that numerous finite difference methods have been developed in the literature for solving elliptic interface problems, with the immerse interface method (IIM) being the most famous one [25,46,3]. The development of FFT or multigrid based fast interface algorithms is also a hot topic in the literature. Currently, only a few interface algorithms can asymptotically achieve $O(N)$ or $O(N \log N)$ complexity, i.e., multigrid IIM [1], augmented IIM [27], and augmented MIB (AMIB) [13]. Nevertheless, all existing fast algorithms can only achieve second order accuracy in resolving interface problems.

In this work, we will develop a novel high order AMIB method for boundary value problems by effectively combining the AMIB method with the FFT algorithm. This gives rise to a uniform approach for implementing high order central differences in multi-dimensions. Examples in 2D and 3D will be shown. The AMIB method is systematically carried out and can be made to arbitrarily high order in principle. Up to 8th order will be demonstrated in this work. Similar to the original MIB boundary closure [49–52], the AMIB method can handle periodic, Dirichlet, Neumann, Robin conditions and their arbitrary combinations.

For the second order central difference discretization, the FACR algorithm [40] is slightly more efficient than the standard FFT Poisson solver, while its implementation becomes complicated. Partly due to such coding complexity, the FACR Poisson solver has never been extended to high order in the literature. In principle, the proposed AMIB method can be combined with the FACR algorithm to deliver high order, even though many details need to sorted out.

A uniform grid is employed throughout this work. In practice, a non-uniform grid with a finer resolution near boundaries could help to reduce the overall error. However, the generalization of the AMIB method to non-uniform grids is highly nontrivial, because the FFT inversion is no longer available. Moreover, high order central differences lose the simplicity of translation invariant feature, when formulated on non-uniform grids.

In this study, we focus mainly on solving Poisson boundary value problems over rectangular domains. A more challenging study is to solve immersed boundary problems with complex shaped domains, for which several advance algorithms have been developed in the literature [15,29,47,39] to achieve high order accuracy, as well as to attain high efficiency with the aid of FFT or multigrid. More importantly, by using the Fourier continuation (FC) method for the resolution of the Gibbs phenomenon [8,9], a fast solver based on alternating direction (FC-AD) has been developed in [10,28] for solving parabolic, elliptic, and hyperbolic PDEs over general smooth domains with Dirichlet boundary conditions. Obviously, such a FC-AD algorithm can be simply applied to rectangular domains to give rise to a high order fast Poisson solver.

We note that the proposed FFT-AMIB method could be generalized to solve irregular domain Poisson problems. In fact, a second order FFT-AMIB has been developed in [13] for solving 2D elliptic interface problems with complex geometries. Moreover, a fourth order MIB boundary closure method has been developed previously in [51] to impose boundary conditions over curved boundaries in 2D. This MIB method could be extended to Poisson boundary value problems over general shaped domains, and its acceleration via augmentation and FFT is under our investigation.

The proposed FFT-AMIB method share some similarities with the FC-AD method [10,28], including using the FFT for Fourier analysis, achieving $O(N\log N)$ high efficiency, and attaining high order convergence. However, there are some differences between them. First, the AMIB method is based on high order central difference discretization, while the FC-AD method solves 1D boundary value problems by Fourier series type ODE solutions. Second, the Gibbs oscillation is directly treated in the FC [8–10], by constructing a smooth periodic function extension. In the AMIB method, with zero padding, the solution is still discontinuous. The Gibbs oscillation is indirectly bypassed in the present study by introducing functions jumps at boundaries as auxiliary variables, so that in the Schur complement procedure of an enlarged linear system, the function discontinuity is not sensed in the FFT inversion part. Finally, the AMIB method can handle Dirichlet, Neumann, and Robin conditions, while the FC-AD method is mainly formulated for Dirichlet boundaries [10,28]. We note that by means of overlapping domain decomposition and curvilinear coordinate transformation, the FC method can handle Neumann boundary condition [11]. Nevertheless, the application of such boundary treatment to more efficient FC-AD method has not been reported yet.

The rest of the paper is organized as follows. In section 2, the Poisson's problem is reviewed and high order central differences are presented. Section 3 focuses on high order fast Poisson solvers via fast Sine transform (FST) with the anti-symmetric solution across the boundary. Following this, in section 4, the idea of immersed boundary is deployed to satisfy the anti-symmetry requirement. High order corrected differences are derived for solving the interface problem from immersed boundary. In section 5, the augmented system is exhibited as the core of the method. And implementation process is described. In section 6, numerical examples are carried out to demonstrate the proposed method. Finally, a summary will be discussed at the end of the paper.

2. Preliminary

In this work, we focus on high order central difference schemes for solving Poisson's equation in both two dimensions (2D) and three dimensions (3D). The governing equation is

$$\triangle u = f(\vec{x}), \quad \vec{x} \in \Omega \tag{1}$$

where $\vec{x} = (x_1, x_2, \cdots, x_d)$ is the spatial vector, Ω is the domain of interest, and $\triangle$ is the Laplacian operator defined by

$$\triangle = \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}.$$

In this paper, d is chosen to be 2 or 3.

Since the periodic boundary condition can be simply handled in the Fourier transform method, we will only focus on three standard boundary conditions in this work, i.e.

1. Dirichlet boundary condition:

$$u = \phi_j \text{ on } \Gamma_j,$$

2. Neumann boundary condition:

$$\frac{\partial u}{\partial n} = \phi_j \text{ on } \Gamma_j,$$

3. Robin boundary condition:

$$ku + \frac{\partial u}{\partial n} = \phi_j \text{ on } \Gamma_j,$$

where Γ_j is one of the boundaries of the computational domain as shown in Fig. 1, and $\frac{\partial}{\partial n}$ denotes the outward normal derivative.

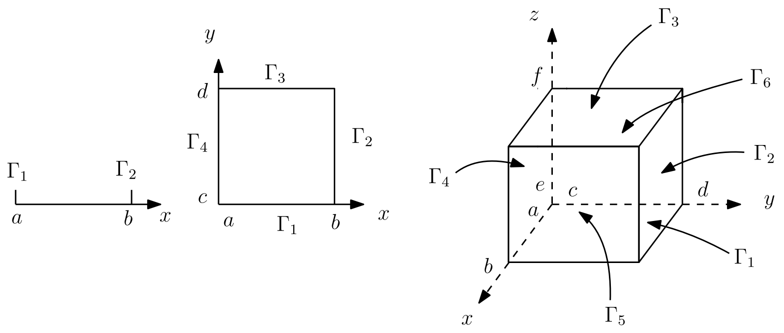

Fig. 1. Illustration of boundary notation in 1D, 2D and 3D.

We are concerned with the solution to Poisson's equation on domain $\Omega = [a, b] \times [c, d] \times [e, f]$, which is partitioned into M, N, P equally spaced intervals in $x-$, $y-$ and $z-$directions such that the mesh sizes are $h_x = (b - a)/M, h_y = (d - c)/N$, and $h_z = (f - e)/P$ respectively. If only a 2D problem is studied, we just neglect the partition in the $z-$direction. Therefore, the grid node coordinates are defined as

$$\begin{aligned} x_i =& a + ih_x,\ y_j = c + jh_y,\ z_p = e + ph_z, \\ & i = 0, \cdots, M,\ j = 0, \cdots, N,\ p = 0, \cdots, P. \end{aligned} \tag{2}$$

Central finite differences are utilized for approximating the Laplacian operator in this work. As the partial derivatives are defined in a tensor product style for separable geometries, we decompose approximation to Laplacian operator in several directions separately. Without loss of generality, high order central differences to second order partial derivative with respect to x are adopted for demonstration. For simplicity, we just concern ourselves with $u = u(x)$.

The fourth order central difference to $u_{xx}(x_i)$ holds with truncation error $O(h_x^4)$:

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

and sixth order central difference takes forms as below with truncation error $O(h_x^6)$:

$$\begin{aligned} u_{xx}(x_i) \approx \frac{1}{h_x^2}[&\frac{1}{90}u(x_{i-3}) - \frac{3}{20}u(x_{i-2}) + \frac{3}{2}u(x_{i-1}) - \frac{49}{18}u(x_i) + \frac{3}{2}u(x_{i+1}) \\ &- \frac{3}{20}u(x_{i+2}) + \frac{1}{90}u(x_{i+3})], \end{aligned} \tag{4}$$

along with the standard eighth order central difference with truncation error $O(h_x^8)$:

$$\begin{aligned} u_{xx}(x_i) \approx& \frac{1}{h_x^2}[-\frac{1}{560}u(x_{i-4}) + \frac{8}{315}u(x_{i-3}) - \frac{1}{5}u(x_{i-2}) + \frac{8}{5}u(x_{i-1}) - \frac{205}{70}u(x_i) \\ &+ \frac{8}{5}u(x_{i+1}) - \frac{1}{5}u(x_{i+2}) + \frac{8}{315}u(x_{i+3}) - \frac{1}{560}u(x_{i+4})]. \end{aligned} \tag{5}$$

Likewise, the fourth, sixth and eighth order finite differences could be defined for u_{yy}, u_{zz} in a similar fashion.

3. Fast Poisson solver

In the literature, many high order fast Poisson solvers have been designed based on the compact differences. In this work, we will take advantage of the aforementioned central differences to construct fast Poisson solvers. As a translation invariant application of high order central finite differences will inevitably use points outside the boundary, we extend the domain by exterior grids in addition to (2) defined by

$$\begin{aligned} x_{-i} &= a - ih_x, x_{M+i} = b + ih_x, \qquad i = 1, 2, 3, \\ y_{-j} &= c - jh_y, y_{N+j} = d + jh_y, \qquad j = 1, 2, 3, \\ z_{-p} &= e - ph_z, z_{P+p} = f + ph_z, \qquad p = 1, 2, 3. \end{aligned}$$

Ghost values can then be defined on these grids for application of high order central difference schemes.

In the first step of constructing our algorithm, we will consider a simple Poisson problem with Dirichlet zero boundary condition. We note that for this simple Poisson problem, a direct application of central differences with FFT fast Poisson solver still cannot maintain high order accuracy. A remedy will be offered so that a FFT fast Poisson solver can be well constructed for the Dirichlet zero boundary. The generalization to real boundary conditions will be addressed in the next section. For simplicity, we will only consider a one dimension (1D) Poisson's equation in this section to illustrate the ideas.

3.1. Fourth order central difference scheme

Consider the fourth order finite difference (FD) scheme for a 1D Poisson's equation:

$$u_{xx}(x) = f, \tag{6}$$

subject to Dirichlet zero boundary condition. In order to apply FFT directly, we further assume anti-symmetry property up to fourth order for ghost values outside the domain

$$u(x_0) = 0, \quad u(x_{-1}) = -u(x_1), \tag{7}$$

$$u(x_M) = 0, \quad u(x_{M+1}) = -u(x_{M-1}). \tag{8}$$

Here, we denote u_i as the approximation to the exact value of $u(x_i)$ and f_i as $f(x_i)$:

$$u_i \approx u(x_i), \quad i = -1, 0, \cdots, M+1,$$
$$f_i = f(x_i), \quad i = 1, 2, \cdots, M-1.$$

To achieve higher order of accuracy, a higher order anti-symmetry property is assumed. Examples with orders being six and eight are given in Appendix. We note that the anti-symmetry properties like (7) and (8) may not hold for a general Dirichlet zero boundary. A systematic procedure to bypass this constraint will be constructed later.

The fourth order FD discretization (3) for equation (6) gives an equation system:

$$\frac{1}{h_x^2}\left(-\frac{1}{12}u_{i-2} + \frac{4}{3}u_{i-1} - \frac{5}{2}u_i + \frac{4}{3}u_{i+1} - \frac{1}{12}u_{i+2}\right) = f_i, \quad \text{for } i = 1, \cdots, M-1. \tag{9}$$

Due to the assumption of anti-symmetry of solution across the boundary, system (9) could be reduced to a matrix-vector form

$$AU = F, \tag{10}$$

where $A \in \mathbb{R}^{M-1,M-1}$, $U \in \mathbb{R}^{M-1}$ and $F \in \mathbb{R}^{M-1}$ are given by

$$A = \frac{1}{h_x^2}\begin{pmatrix} -\frac{5}{2}+\frac{1}{12} & \frac{4}{3} & -\frac{1}{12} & 0 & 0 & 0 & \cdots & 0 \\ \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} & 0 & 0 & \cdots & 0 \\ -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} & 0 & \cdots & 0 \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ 0 & & & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} \\ \vdots & & & & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} \\ 0 & 0 & \cdots & & & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2}+\frac{1}{12} \end{pmatrix}, \tag{11}$$

together with

$$U = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{M-1} \end{pmatrix} \quad \text{and} \quad F = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{M-1} \end{pmatrix}.$$

Note that the entries at the corners of matrix A are due to the anti-symmetry assumption.

Now we focus on fast solver for the solution of above equation system. Based on discrete Sine transform, the solution could be expanded as

$$u_j = \sum_{l=1}^{M-1} \hat{u}_l \sin(\frac{lj\pi}{M}), -1 \le j \le M+1. \tag{12}$$

where

$$\hat{u}_l = \frac{2}{M} \sum_{j=1}^{M-1} u_j \sin(\frac{lj\pi}{M}), l = 1, 2, \cdots, M-1. \tag{13}$$

Besides, the source term f_j has discrete Sine transform

$$f_j = \sum_{l=1}^{M-1} \hat{f}_l \sin(\frac{lj\pi}{M}), 1 \le j \le M-1, \tag{14}$$

where

$$\hat{f}_l = \frac{2}{M} \sum_{j=1}^{M-1} f_j \sin(\frac{lj\pi}{M}), l = 1, 2, \cdots, M-1. \tag{15}$$

Plugging (12) and (14) into (9), we obtain

$$\begin{aligned}
&\sum_{l=1}^{M-1} \hat{u}_l \frac{1}{h_x^2} [-\frac{1}{12} \sin(\frac{l(j-2)\pi}{M}) + \frac{4}{3} \sin(\frac{l(j-1)\pi}{M}) - \frac{5}{2} \sin(\frac{lj\pi}{M}) \\
&\quad + \frac{4}{3} \sin(\frac{l(j+1)\pi}{M}) - \frac{1}{12} \sin(\frac{l(j+2)\pi}{M})] \\
&= \sum_{l=1}^{M-1} \hat{u}_l \frac{1}{h_x^2} [-\frac{1}{3} (\cos(\frac{l\pi}{M}) - 1)(\cos(\frac{l\pi}{M}) - 7)] \sin(\frac{lj\pi}{M}) \\
&= \sum_{l=1}^{M-1} \hat{f}_l \sin(\frac{lj\pi}{M}), \quad \text{for} \quad j = 1, 2, \cdots, M-1.
\end{aligned}$$

The above equation system is well-posed, and hence by equating the coefficients of the same terms, we get the relation

$$\hat{u}_l = \frac{\hat{f}_l}{\lambda_l}, \quad \text{for} \quad l = 1, 2, \cdots, M-1,$$

where $\lambda_l = -\frac{1}{3h_x^2}[\cos(\frac{l\pi}{M}) - 1][\cos(\frac{l\pi}{M}) - 7]$.

A procedure for the FFT solution can be described as below:

1. Compute Sine transform for $f(x_j)$ via inverse fast Sine transform (IFST).

$$\hat{f}_l = \frac{2}{M} \sum_{j=1}^{M-1} f(x_j) \sin(\frac{lj\pi}{M}), \quad \text{for} \quad l = 1, \cdots, M-1.$$

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

3.2. Numerical validation

We consider two 1D Poisson problems with Dirichlet zero boundary condition, with one satisfying anti-symmetry and the other one not. Two numerical implementations of fourth order central difference will be considered. The first one employs the FFT procedure presented above. The second one further assumes that the analytical values on ghost nodes outside the boundaries are known. Then one can move such ghost values to the right-side hand of algebraic system, and solves the system by the LU decomposition.

Example 1 in 1D. Consider Poisson's equation

$$u_{xx} = 16\pi^2 \sin(4\pi x) \quad \text{for} \quad x \in [0, 1]$$

Table 1
Numerical errors of the FFT and LU methods for Example 1 in 1D. Here N denotes the number of partitions on the given interval.

N	FFT		LU	
	L_∞ Error	Order	L_∞ Error	Order
32	2.607E-4	–	2.607E-4	–
64	1.646E-5	3.99	1.646E-5	3.99
128	1.031E-6	4.00	1.031E-6	4.00
256	6.450E-8	4.00	6.450E-8	4.00
512	4.032E-9	4.00	4.032E-9	4.00

Table 2
Numerical errors of the FFT and LU methods for Example 2 in 1D.

N	FFT		LU	
	L_∞ Error	Order	L_∞ Error	Order
32	1.321E-2	–	5.229E-4	–
64	3.235E-3	2.03	3.294E-5	3.99
128	8.046E-4	2.00	2.063E-6	4.00
256	2.009E-4	2.00	1.290E-7	4.00
512	5.020E-5	2.00	8.061E-9	4.00

with solution $u(x) = \sin(4\pi x)$ satisfying anti-symmetry. The numerical results are shown in Table 1. It is obvious that both FFT and LU methods attain the same precision and fourth order of convergence is achieved in both cases. In algebraic computations, the FFT method is much faster than the LU decomposition.

Example 2 in 1D. We next consider Poisson's equation

$$-u_{xx} = 16\pi^2 \cos(4\pi x) \quad \text{for} \quad x \in [0, 1]$$

with solution $u(x) = -\cos(4\pi x) + 1$ satisfying the homogeneous Dirichlet boundary condition $u = 0$ at $x = 0$ and $x = 1$. Nevertheless, the anti-symmetry property is not held. Table 2 lists the errors of the FFT and LU methods for fourth order central difference scheme. It is observed that the FFT method only provides second order accuracy while the LU gives fourth order accuracy. This demonstrates the importance of anti-symmetry of solution when using the above FFT Poisson solver.

Remark 3.1. So far, a fourth order fast Poisson solver via FST has been formulated for the Poisson problem with Dirichlet zero boundary condition. A crucial assumption here is that one layer of ghost value outside of domain satisfies the anti-symmetry property across the boundary. Higher order fast Poisson solvers can be constructed in this manner. In Appendix, by assuming two and three layers of ghost values satisfying anti-symmetry respectively, sixth and eighth order central difference schemes can be formulated by the same FFT procedure.

Remark 3.2. For the two dimensional (2D) or three dimensional (3D) Poisson's equation, we continue to assume the homogeneous Dirichlet boundary and anti-symmetry property in each dimension as described in 1D. By the tensor product approach, the discretization and FFT procedure for high dimensional problems can be similarly completed.

Remark 3.3. The aforementioned fast Poisson solvers only apply to the case of homogeneous Dirichlet condition with anti-symmetric solution across the boundary. However, in practice the anti-symmetry property may not hold or is not easily to be verified for homogeneous Dirichlet boundary conditions. Moreover, general Poisson problems face more complex boundary conditions. Hence, it is necessary to explore fast Poisson solvers for general solutions. We will construct a systematic procedure to bypass the anti-symmetric issue.

4. Immersed boundary problem and corrected differences

In the previous section, fast Poisson solvers based on high order central differences have been constructed with the assumption of anti-symmetry of solution across the boundary. However, such assumption is generally not valid for Poisson problems. This is essentially the obstacle that prevents the application of FFT Poisson solver to high order central difference schemes. To resolve this difficulty, an immersed boundary problem will be introduced in this paper. To be more clear, the original domain is embedded in a larger domain such that fictitious solution equals to 0 on the extended domain. The aforementioned fast Poisson solvers can then be facilitated, since the zero solution automatically meets the requirement of anti-symmetry on the boundary of extended domain. However, the entire solution consisting of original solution and fictitious solution is discontinuous on the original boundary or on the interface of the immersed boundary problem. The direct FFT computation of discontinuous solutions will suffer Gibbs oscillations. Instead, corrected differences will be employed in

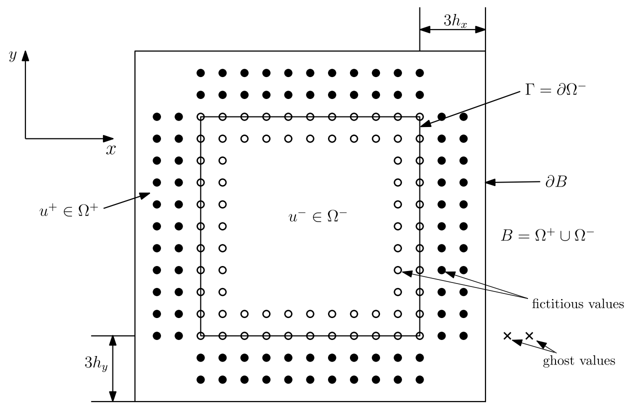

Fig. 2. A demonstration of immersed boundary problem for fourth order central difference scheme with two interior and exterior layers of *fictitious values* around the interface $\Gamma = \partial\Omega^-$. Here filled and open circles stand for fictitious values to be used in approximation of jump quantities from corrected differences. Meanwhile, *ghost values* (depicted as crossings) are zero padding values beyond the expanded domain satisfying anti-symmetry.

the proposed augmented matched interface and boundary (AMIB) method, so that central differences are preserved with some correction terms being used to restore the desired accuracy.

4.1. Immersed boundary

With a uniform grid spacing in each dimension, we embed the original domain Ω in a larger rectangle or cube B such that the distance between interior boundary $\partial\Omega$ and the exterior boundary ∂B of B is s multiples of h, where s is a positive integer. The value of s varies according to the desired accuracy of central difference approximation for efficiency concern. Technically, just a minimum integer number s is needed to carry out the approximation of desired accuracy, otherwise extra computational efforts will be wasted. Fig. 2 illustrates the case in 2D for fourth order central difference scheme with $s = 3$. As a consequence, the grid node coordinates are redefined as below in the domain B

$$\begin{aligned} x_i =& a + (i - s)h_x, \;\; y_j = c + (j - s)h_y, \;\; z_k = e + (k - s)h_z, \\ & i = 0, \cdots, M + 2s, \;\; j = 0, \cdots, N + 2s, \;\; k = 0, \cdots, P + 2s, \end{aligned} \tag{16}$$

where h_x, h_y, h_z are identically defined as (2).

From now on, we denote the original domain Ω as Ω^- and the extended subdomain as Ω^+, yielding the whole domain $B = \Omega^+ \cup \Omega^-$. Correspondingly, the solution in the original domain is defined as u^- in Ω^-, and solution in the extended domain Ω^+ as u^+. Source term is defined as $f^- \in \Omega^-$ and $f^+ \in \Omega^+$. Then the immersed boundary problem can be modeled as

$$\triangle u(\vec{x}) = f, \quad \vec{x} \in B \tag{17}$$

with source term being

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

Note that as the artificial solution $u(\vec{x}) = 0$ for $\vec{x} \in \Omega^+$, $f(\vec{x}) = 0$ for $\vec{x} \in \Omega^+$. Further beyond the exterior boundary ∂B, u is considered as 0 such that the anti-symmetry property is now satisfied on boundary ∂B. In other words, ghost values beyond ∂B are set to be 0. This facilitates the utilization of high order central differences and FFT Poisson solvers. However, this remodeled problem introduces an interface $\Gamma = \partial\Omega^-$, across which discontinuity of solution and source term emerges.

It is well known that central difference approximations will suffer accuracy reduction near the interface Γ when the approximation involves nodes from both sides of Γ. Grid nodes in such situation are called irregular points, otherwise regular points. To be more clear, we take the case of fourth order central difference for instance. In general, assume the interface intersects the grid line $y = y_j$ at some point α between x_i and x_{i+1}. As the fourth order central difference for nodes x_{i-1}, x_i, x_{i+1} and x_{i+2} will use the node information on the other side of the interface, these four nodes are called irregular points. For sixth and eighth order central difference with even wider stencil, more irregular points are defined along the grid line accordingly. In the literature, many methods have been proposed to correct finite difference approximation for interface problems [25,27,46,3,48,53]. In this study, we will follow the AMIB method [13] that uses corrected finite differences [46] to tackle such irregular points problem. The corrected differences incorporate several interface jump quantities to compensate the discontinuity, while central differences are still used for regular points.

4.2. Corrected finite differences for Laplacian

Taking advantage that central difference approximation to Laplacian is carried out dimension by dimension, we again just focus on the derivation of corrected differences in 1D, by taking $u = u(x)$ and $h = h_x$. Assume the smoothness of the piecewise function $u \in C^{l+1}$ in each side of the interface. Function values at two nodes could be related by corrected Taylor expansion with some jump conditions included, from which corrected finite difference of different orders could be derived. In the first place, the following two important results (18) and (19) can be cited from *Lemma 1* and *Remark 2* of [46]. Here one considers the corrected Taylor expansion for two adjacent points on each side of the interface:

Assume the interface $x = \alpha$ lies in between grid points x_j and x_{j+1}, then it follows that

$$u(x_{j+1}) = \sum_{k=0}^{l} \frac{h^k}{k!} u^{(k)}(x_j) + \sum_{k=0}^{l} \frac{(h^+)^k}{k!} [u^{(k)}]|_{x=\alpha} + O(h^{l+1}) \tag{18}$$

and

$$u(x_j) = \sum_{k=0}^{l} \frac{(-h)^k}{k!} u^{(k)}(x_{j+1}) - \sum_{k=0}^{l} \frac{(h^-)^k}{k!} [u^{(k)}]|_{x=\alpha} + O(h^{l+1}) \tag{19}$$

where $h^- = x_j - \alpha$, $h^+ = x_{j+1} - \alpha$, and $[u^{(m)}]_\alpha = \lim_{x\to\alpha^+} u^{(m)}(x) - \lim_{x\to\alpha^-} u^{(m)}(x)$.

In the present study, the above results have to be extended for high order central difference schemes which involve longer stencils. In particular, besides x_j and x_{j+1}, jump corrected Taylor expansions are needed for nodes further away from interface $x = \alpha$.

$$u(x_{j+j_2}) = \sum_{k=0}^{l} \frac{[(j_2 - j_1)h]^k}{k!} u^{(k)}(x_{j-j_1}) + \sum_{k=0}^{l} \frac{[(j_2 - 1)h + h^+]^k}{k!} [u^{(k)}]|_{x=\alpha} + O(h^{l+1}) \tag{20}$$

and

$$u(x_{j-j_1}) = \sum_{k=0}^{l} \frac{[(j_1 - j_2)h]^k}{k!} u^{(k)}(x_{j+j_2}) - \sum_{k=0}^{l} \frac{[-j_1 h + h^-]^k}{k!} [u^{(k)}]|_{x=\alpha} + O(h^{l+1}) \tag{21}$$

where integers $j_2 \geq 1$ and $j_1 \geq 0$ are index increments while j denotes the index location. Here we still assume that $x_j \leq \alpha \leq x_{j+1}$, $h^- = x_j - \alpha$, and $h^+ = x_{j+1} - \alpha$.

Equations (20) and (21) are generalized form of (18) and (19). The proof of these two equations is just a minor modification of the proof for (18) and (19). Thus it is omitted here. For details, refer to [46]. When $j_2 = 1$, and $j_1 = 0$, equations (20) and (21) are consistent with (18) and (19). With aid of formula (20) and (21), fourth order corrected differences can be obtained as below:

Theorem 1. Corrected fourth differences. Let $x_j \leq \alpha < x_{j+1}, h^- = x_j - \alpha$, 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 $O(h^4)$:

$$\begin{aligned} u_{xx}(x_{j-1}) \approx{} & \frac{1}{h^2}[-\frac{1}{12}u(x_{j-3}) + \frac{4}{3}u(x_{j-2}) - \frac{5}{2}u(x_{j-1}) + \frac{4}{3}u(x_j) - \frac{1}{12}u(x_{j+1})] \\ & + \frac{1}{12h^2} \sum_{m=0}^{5} \frac{(h^+)^m}{m!}[u^{(m)}], \end{aligned} \tag{22}$$

$$\begin{aligned} u_{xx}(x_j) \approx{} & \frac{1}{h^2}[-\frac{1}{12}u(x_{j-2}) + \frac{4}{3}u(x_{j-1}) - \frac{5}{2}u(x_j) + \frac{4}{3}u(x_{j+1}) - \frac{1}{12}u(x_{j+2})] \\ & - \frac{4}{3h^2} \sum_{m=0}^{5} \frac{(h^+)^m}{m!}[u^{(m)}] + \frac{1}{12h^2} \sum_{m=0}^{5} \frac{(h+h^+)^m}{m!}[u^{(m)}], \end{aligned} \tag{23}$$

$$\begin{aligned} u_{xx}(x_{j+1}) \approx{} & \frac{1}{h^2}[-\frac{1}{12}u(x_{j-1}) + \frac{4}{3}u(x_j) - \frac{5}{2}u(x_{j+1}) + \frac{4}{3}u(x_{j+2}) - \frac{1}{12}u(x_{j+3})] \\ & + \frac{4}{3h^2} \sum_{m=0}^{5} \frac{(h^-)^m}{m!}[u^{(m)}] - \frac{1}{12h^2} \sum_{m=0}^{5} \frac{(h^- - h)^m}{m!}[u^{(m)}], \end{aligned} \tag{24}$$

$$\begin{aligned} u_{xx}(x_{j+2}) \approx{} & \frac{1}{h^2}[-\frac{1}{12}u(x_j) + \frac{4}{3}u(x_{j+1}) - \frac{5}{2}u(x_{j+2}) + \frac{4}{3}u(x_{j+3}) - \frac{1}{12}u(x_{j+4})] \\ & - \frac{1}{12h^2} \sum_{m=0}^{5} \frac{(h^-)^m}{m!}[u^{(m)}]. \end{aligned} \tag{25}$$

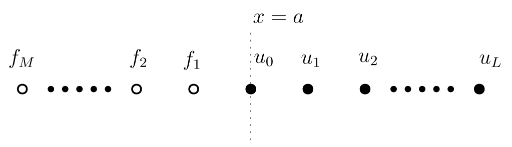

Fig. 3. An illustration of fictitious values near the left boundary.

Remark 4.1. We note that one may just take m ranging from 0 to 4 to achieve the third order accuracy locally at irregular points. Since the fourth order central difference is applied at all regular points except for at a few irregular points, a global fourth order of convergence can still be maintained. Thus, for a better efficiency, we will take $m=0,\ldots,4$ in the present study.

Remark 4.2. The fourth order corrected differences have been illustrated above in 1D manner. In higher dimensional study, such operators are applicable in each dimension once intersections of grid line with interface are encountered. Besides the fourth order case, sixth or eighth order corrected difference can be derived with the details described in the Appendix.

Remark 4.3. The local distances $h^-=x_j-\alpha$ and $h^+=x_{j+1}-\alpha$ defined in the corrected differences should be determined before approximation. In our expanded domain, the interface Γ aligns with the grid line. Thus either h^- or h^+ is equal to zero. For convenience, we consider all grid nodes on the interface as inside the interface. Therefore, there are only two situations in terms of the local distance. For the interface close to the left (bottom) boundary, $h^-=-h$ and $h^+=0$. We name this situation as type I. On the other hand, for the interface close to the right (upper) boundary, $h^-=0$ and $h^+=h$. We name this situation as type II.

4.3. Reconstructing the Cartesian derivative jumps

So far we have established corrected differences with some derivative jumps $[u^{(m)}]$ undetermined. In the proposed AMIB method, such jump quantities will be calculated through enforcing the given boundary conditions. A systematic MIB boundary closure [49–52] has been developed in the literature for implementing high order central difference schemes with any combination of Dirichlet, Neumann, and Robin conditions. In the present study, this MIB method will be employed to enforce boundary conditions and correspondingly generate a sufficient number of fictitious values outside $\partial\Omega^-$ or interface Γ. Unlike the original MIB [49–52], one will not directly modify the discrete Laplacian approximation by using fictitious values. Instead, the fictitious values will be utilized for calculating the derivative jumps $[u^{(m)}]$, as in the AMIB method [13].

A minimal number of fictitious nodes is determined as follows for the high order central difference schemes. As mentioned above, it is sufficient to use one less order locally in the corrected jump quantities to guarantee the high orders. To ensure a third, fifth, and seventh order convergence locally for the irregular points for fourth, sixth, and eighth corrected differences respectively, polynomials of degree 4, 6, and 8 are needed for approximation. As a consequence, 2, 3, and 4 fictitious values outside Γ are combined with 3, 4, and 5 real function values on the other side of the interface to give desired polynomials respectively for each case. Fig. 2 shows two layers of fictitious values needed on each side of the interface Γ for the fourth order corrected differences. We note that the interface Γ is regarded as in Ω^- in the present notation.

4.3.1. Fictitious values formulation

To reconstruct the Cartesian derivative jumps, we first review how fictitious values are calculated in the MIB method [52]. A systematic procedure has been developed in [52] for handling Dirichlet, Neumann, and Robin boundary conditions. This procedure iteratively enforces the boundary condition to determine fictitious values as smooth extension of solution around the interface Γ. The MIB formulation of fictitious values is explained in great details for various boundary conditions in [52].

In the present study, it is sufficient to illustrate the MIB formulation of fictitious values in the case of Robin boundary condition in 1D,

$$ku+\frac{\partial u}{\partial n}=ku-\frac{\partial u}{\partial x}=\phi \qquad \text{on} \quad \Gamma_1. \tag{26}$$

We generate fictitious values iteratively by repeatedly matching the boundary conditions across the boundary. Referring to Fig. 3, to generate M fictitious values f_i for $i=1,2,\cdots,M$ outside the domain, $L+1$ function values u_j for $j=0,1,\cdots,L$ inside the domain are to be used. Firstly, one fictitious value f_1 is constructed according to approximation to boundary (26):

$$ku_0+C_{2,1}^{(1)}f_1+\sum_{i=2}^{L+2}C_{2,i}^{(1)}u_{i-2}=\phi,$$

where $C_{2,i}^{(1)}$ are finite difference (FD) weights in approximating first derivative at $x=a$ by using a stencil $\{f_1,u_0,\cdots,u_L\}$. The notation 2 in first subscript of $C_{2,i}^{(1)}$ is because u_0 is the second point in the FD stencil. As a result, unknown f_1 can

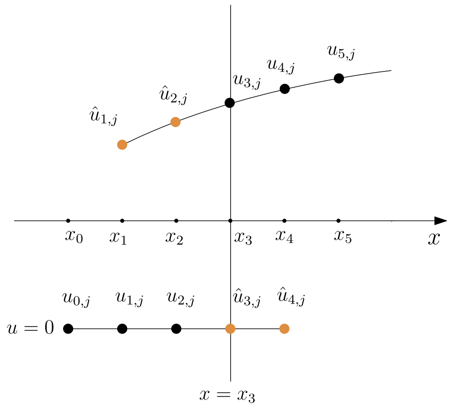

Fig. 4. An illustration of jumps approximation via two Lagrange polynomials from two sides of the interface $x = x_3$ for the fourth order approximation. The black dots stand for the real values while the yellow dots denote the fictitious values. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

be represented as the combination of u_i for $i = 0, 1, \cdots, L$ and ϕ. Taking f_1 as known, we may further seek to determine fictitious value f_2 using one iterative step

$$ku_0 + C^{(1)}_{3,1} f_2 + C^{(1)}_{3,2} f_1 + \sum_{i=3}^{L+3} C^{(1)}_{3,i} u_{i-3} = \phi,$$

where $C^{(1)}_{3,i}$ are FD weights to approximate first derivative at $x = a$ using a stencil $\{f_2, f_1, u_0, u_1, \cdots, u_L\}$. In this manner, we can further obtain representation of fictitious values $f_3, f_4, \cdots, f_M$ in terms of u_i for $i = 0, 1, \cdots, L$. Even though it is flexible to choose the value for L, we expect to set L so that high-order accuracy is achieved in boundary implementation. As suggested in [52], the condition $L + 1 = 2M$ is sufficient to guarantee the order requirement when M fictitious values are needed in high order central differences near the boundary. For example, in the fourth order central difference, two layers of fictitious values are needed. We may adopt $L = 3$ for formulation of these two fictitious values.

The MIB implementation of Neumann boundary condition can be similarly formulated. For the Dirichlet condition, we will follow the idea in [52] to generate a numerical boundary condition of the form

$$\frac{\partial^2 u}{\partial n^2} = \phi, \tag{27}$$

which will be iteratively enforced to calculate fictitious values.

Finally, the fictitious values calculated from Dirichlet, Neumann, and Robin boundary conditions can be rewritten into a general form in 2D. Denote $\hat{u}_{i,j}$ as a fictitious value at (x_i, y_j). We have

$$\hat{u}_{i,j} = \sum_{(x_I, y_J) \in \mathbb{S}_{i,j}} W_{I,J} u_{I,J} + W_0[u] + W_1[u_x] + W_2[u_y] + W_3\phi, \tag{28}$$

which is a linear combination of functions values on a chosen set $\mathbb{S}_{i,j}$, and known boundary data. Here the function jump $[u]$ can be explicitly given by the Dirichlet boundary condition. For Neumann boundary condition, $[u_x]$ or $[u_y]$ can be analytically given. For Robin and other mixed form of boundary conditions, non-homogeneous term ϕ is usually known. Hence, the linear combination form (28) represents the most general setup for all types of boundary conditions. The node set $\mathbb{S}_{i,j}$ can be determined by the choice of M and L before calculating the MIB weights $W_{I,J}$. Such a MIB formulation can be easily extended to 3D.

4.3.2. *Approximation to derivative jumps*

Considering jump quantities in the corrected differences, we want to build polynomials of certain degree to calculate limiting derivatives from each side of the interface Γ. The jump quantity at $x = \alpha$ is defined as

$$[\frac{\partial^m u}{\partial x^m}]|_{x=\alpha} = \lim_{x \to \alpha^+} \frac{\partial^m u}{\partial x^m} - \lim_{x \to \alpha^-} \frac{\partial^m u}{\partial x^m} \tag{29}$$

Fig. 4 shows the general fourth order numerical approximation to jumps by two polynomials near the interface. We note that in our immersed boundary problem, the approximation of up to certain order derivative can just be completed by one-sided polynomial from inside of Ω^- together with the external function $u^+ = 0$. By taking derivatives on one-sided polynomials, we can approximate the jump values at intersecting point (x^*, y_j) with regard to x-partial derivatives. Two ways of approximation should be formulated for two types of interface as mentioned in Remark 4.3. An example for fourth order approximation is shown below, which could be generalized to even higher order cases.

For type I, the jump quantities are approximated by (refer to Fig. 4)

$$\begin{aligned}[\frac{\partial^m u}{\partial x^m}] &= \frac{\partial^m u^-}{\partial x^m} - 0 \\ &\approx w^m_{i-1,j}\hat{u}_{i-1,j} + w^m_{i,j}\hat{u}_{i,j} + \sum_{k=1}^{3} w^m_{i+k,j} u_{i+k,j}\end{aligned} \tag{30}$$

and for type II, the jump quantities are approximated by

$$\begin{aligned}[\frac{\partial^m u}{\partial x^m}] &= 0 - \frac{\partial^m u^-}{\partial x^m} \\ &\approx -\sum_{k=1}^{3} w^m_{i-3+k,j} u_{i-3+k,j} - w^m_{i+1,j}\hat{u}_{i+1,j} - w^m_{i+2,j}\hat{u}_{i+2,j}\end{aligned} \tag{31}$$

where $u^- \in \Omega^-$, and $w^m_{p,q}$ stands for the weights in central finite difference approximation at $x=\alpha$. In $w^m_{p,q}$, subscript (p,q) denotes the location (x_p, y_q), and $m = 0, 1, \cdots, 4$ indicates the mth derivative. In these two types, we take $x_i = \alpha$. The best accuracy can be obtained in the framework of central finite difference.

Plugging fictitious values representation (28) into (30) or (31) gives equivalent forms:

$$\sum_{(x_I, y_J)\in\mathbb{S}} C_{I,J} u_{I,J} + [\frac{\partial^m u}{\partial x^m}] = C_0[u] + C_1[u_x] + C_2\phi, \tag{32}$$

where $C_{I,J}$ is the corresponding weights of function value $u_{I,J}$ in approximation to jump quantity $[\frac{\partial^m u}{\partial x^m}]$, and $\mathbb{S}$ is the set of needed grid points in (30) or (31). Equivalent form for jump quantities $[\frac{\partial^m u}{\partial y^m}]$ could be derived in a similar fashion in the $y-$ direction.

$$\sum_{(x_I, y_J)\in\mathbb{S}} \hat{C}_{I,J} u_{I,J} + [\frac{\partial^m u}{\partial y^m}] = \hat{C}_0[u] + \hat{C}_1[u_y] + \hat{C}_2\phi. \tag{33}$$

The original boundary conditions are automatically satisfied then. In equation (32) and (33), $[u], [u_x]$ or $[u_y]$ is known, but $u_{I,J}$ are unknowns to be solved. In the AMIB method, we take $[\frac{\partial^m u}{\partial x^m}]$ and $[\frac{\partial^m u}{\partial y^m}], m = 0, 1, \cdots, 4$ as auxiliary variables. Equation (32) or (33) can be rewritten in one matrix-vector form:

$$CU + IQ = \Phi, \tag{34}$$

where C stands for the weights coefficients matrix, U represents the function variables, I is the identity matrix, Q is the introduced auxiliary variables on all boundary nodes, and Φ is from the known boundary quantities $[u]$, $[u_x]$, $[u_y]$, and ϕ.

5. Augmented system

5.1. formulation of the augmented system

In the preceding section, corrected differences are introduced, while the correction terms are approximated using fictitious values from the MIB method. With correction terms defined as auxiliary variables along with the function values, one augmented system can be constructed. Now we take approximation to 2D Poisson's equation for demonstration. Let $U_{i,j}$ indicate the discrete solution of $u(x_i, y_j)$ at (x_i, y_j). Based on the corrected difference analysis, the problem (1) is discretized as

$$L_h U_{i,j} + C_{i,j} = f_{i,j}, \quad 1 \le i \le M + 2s - 1, \quad 1 \le j \le N + 2s - 1, \tag{35}$$

where $C_{i,j}$ is the correction term, and $L_h U_{i,j}$ is the standard high order central difference scheme to Laplacian with a degree of freedom $N1 = (M + 2s - 1) \times (N + 2s - 1)$. The s has the same meaning in (16).

Below we take the fourth order study for instance. The parameter s is equal to 3. Then $N1 = (M + 5) \times (N + 5)$. Note that correction term only exists at irregular points but vanishes at regular points. In other words, $C_{i,j}$ exists for approximation at the each two layers of grid points surrounding the interface. Otherwise, $C_{i,j}$ vanishes for the other interior grid points inside the interface as shown in Fig. 2. Via Eq. (35), a relation between numerical solution and auxiliary variables in x- or y-partial derivatives is obtained as

$$AU + BQ = F, \tag{36}$$

where A is a symmetric, diagonally dominant matrix of dimension $N1$ by $N1$, B is a matrix of coefficients from correction terms and F is a vector with entries being f_{ij}. We note that the inversion of matrix A can be facilitated with the above FFT algorithm. Variable U stands for the numerical solution to the immersed boundary problem and Q is a vector of 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 the intersection of interface and grid lines. The total number of auxiliary variables, denoted as $N2$, is 5 times that of all intersection points, while the latter is proportional to M or N. That means $N2$ is one dimension smaller than $N1$. Matrix B is a sparse matrix of dimension $N1 \times N2$, and it is composed of the coefficients from the correction terms in the corrected differences.

Moreover, inspired by formula (34), approximation to jump quantities could be formulated as

$$CU + IQ = \Phi, \tag{37}$$

where notations in (37) have the same meaning in (34). Matrix C is a sparse matrix of dimension $N2 \times N1$, and I is a $N2$ by $N2$ identity matrix. An augmented MIB matrix is therefore obtained by combining (36) and (37),

$$KW = R, \tag{38}$$

where

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

5.2. *Schur complement*

We are concerned with solving the linear system of equation efficiently. The symmetric coefficient matrix A enables us to take advantage of fast Poisson solvers in the implementation. Eliminating U in (38) gives a Schur complement system for Q,

$$(I - CA^{-1}B)Q = \Phi - CA^{-1}F, \tag{39}$$

which has a much smaller dimension. To solve for Q in (39), a biconjugate gradient method could be utilized. Once Q is determined, U could be solved by one more FFT inversion

$$AU = (F - BQ). \tag{40}$$

The Schur complement system (39) takes the majority of computation time. A detailed explanation on the implementation of the conjugate gradient iteration is described below:

1. The determination of the right side $\hat{F} = \Phi - CA^{-1}F$ involves one step FFT for $A^{-1}F$ and a few simple algebraic operations.
2. For the left hand, the matrix vector product of $(I - CA^{-1}B)Q$ is completed in a few steps. Based on the equivalence to $IQ - CA^{-1}BQ$, first a product of BQ is obtained. Then FFT is carried out on $A^{-1}BQ$. Finally it is finished by simple multiplication and summation for $(I - CA^{-1}B)Q$. The transpose of $(I - CA^{-1}B)^T Q$ is done on $IQ - B^T A^{-1} C^T Q$ by the same approach.
3. A initial guess $Q = (0, 0, \cdots, 0)^T$ starts the biconjugate gradient iteration, and either the maximal iteration number being 5000 or error tolerance less than 10^{-15} terminates the iteration. In following numerical experiments, the preceding criteria are used unless specified.

Remark 5.1. Note that selection of such small error tolerance in step 3 is reasonable for validating high order scheme. In real application, a larger tolerance could be used based on the practical needs.

6. Numerical experiments

In this section, we examine the performance of the proposed augmented matched interface and boundary (AMIB) method in solving two or three dimensional Poisson's equation with an arbitrary combination of Dirichlet, Neumann, and Robin boundary conditions. The AMIB approach is primarily designed for high order central difference schemes, but the same approach is applicable to second order central difference as well. In the present study, we will test numerical performance of AMIB2, AMIB4, AMIB6, and AMIB8 methods, with the number after AMIB standing for the designed order of accuracy. For a comparison, the standard MIB boundary closure method [49–52] will be employed too. Without Schur complement and FFT components, a biconjugate gradient iteration is simply employed in the MIB to solve the sparse system, and the termination criteria are identical to those in the AMIB algorithm. Like the AMIB, one can also specify the order of accuracy for the MIB method.

For simplicity, a square domain with uniform mesh is used such that there are same amount of grids in x, y or z direction in the given domain, respectively. Fig. 1 gives an illustration of the boundary notations. In the following presentation, we will focus mainly on 2D setting. The numerical accuracy and convergence will be tested by comparing the numerical and exact solutions under the maximum norm and L_2 norm defined as

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

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

where $u_h(x_i, y_j), u(x_i, y_j)$ are numerical and analytical solution, respectively. The set S_{Ω^-} stands for the set of grid points on the given domain before the proposed domain expansion, and N is the number of grid points in each direction of set S_{Ω^-}. The purpose of error analysis on such set is because the solution on the given domain is of concern, but not on Ω^+.

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

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

where $||E_i||$ is the error of mesh h_i for $i = 1, 2$ using either of the above two defined norms on N by N mesh. The solution calculation is facilitated by a FFT subroutine from Numerical Recipes [35] with 2^n summation. Due to the limitation of partition number equal to 2^n in the subroutine, non-bisection refinement of interval size is used in the error analysis.

All experiments were carried out on MacBook Pro with 8.00 GB RAM and Intel 2.3 GHz Intel Core i5.

6.1. Accuracy studies

In this section, several comparisons on numerical accuracy between MIB and AMIB methods are studied. In each table below, N_x, N_y and N_z for AMIB and MIB stand for grid numbers on the domain Ω^- in each direction.

Example 1. We first examine the second and fourth order accuracy of the proposed AMIB method for Dirichlet boundary conditions. Tests on 2D Poisson's equation defined in a domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$ are studied for this purpose. Here Poisson's equation is given as

$$u_{xx} + u_{yy} = e^x(2 + y^2 + 2\sin(y) + 4x\sin(y)),$$

with analytical solution $u = e^x(x^2\sin(y) + y^2)$. The Dirichlet boundary condition along the four boundaries are determined according to the exact solution.

The second and fourth order convergence could be observed under each of the two norms in Table 3. As the computing time is related to the iteration number for solving the auxiliary variables in Schur complement, we report the iteration numbers in the second last column in Table 3. The last column is the CPU time in seconds for each case. For the present study, the iteration number has no change or just increases a little when mesh size is doubled. Consequently, the AMIB method is very efficient. By comparing the second and fourth accurate results, the AMIB4 is superior to AMIB2 in terms of accuracy and computing time.

Example 2. This example is dedicated to the comparison of AMIB and MIB for solving Poisson's equation

$$u_{xx} + u_{yy} = 2e^x\cos(x + y) - e^x\sin(x + y)$$

on domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$, subject to Dirichlet boundary conditions determined by the exact solution $u = e^x\sin(x + y)$. Table 4 indicates that both methods produce a fourth order of convergence, while the AMIB4 is much more efficient than the MIB4, especially when a dense mesh is considered.

Example 3. This example focuses on validating the high-order convergence of our new methods. Restricted on a domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$, a 2D Poisson's equation is defined as

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

subject to mixed boundary conditions:

$$u = \sin(kx)\cos(\frac{k\pi}{3}) \qquad \text{on} \quad \Gamma_1,$$

$$u + \frac{\partial u}{\partial n} = (\sin(\frac{k\pi}{3}) + k\cos(\frac{k\pi}{3}))\cos(ky) \quad \text{on} \quad \Gamma_2,$$

Table 3
Example 1 – Second and fourth order numerical error analysis.

$[N_x, N_y]$	AMIB2					
	L_∞		L_2		iter no.	CPU time (s)
	Error	Order	Error	Order		
[29, 29]	7.529E-4	–	2.965E-4	–	11	6.95E-3
[61, 61]	1.547e-4	2.03	6.333E-5	2.09	12	2.64E-2
[125, 125]	3.575E-5	1.99	1.496E-5	1.99	12	9.64E-2
[253, 253]	8.631E-6	1.99	3.651E-6	1.99	12	0.325
[509, 509]	2.122E-6	1.99	9.030E-7	1.99	12	1.463
$[N_x, N_y]$	AMIB4					
	L_∞		L_2		iter no.	CPU time (s)
	Error	Order	Error	Order		
[27, 27]	1.336E-6	–	5.251E-7	–	20	1.142E-2
[59, 59]	5.485E-8	3.98	2.301E-8	3.90	20	3.954E-2
[123, 123]	2.892E-9	3.96	1.264E-9	3.90	20	0.151
[251, 251]	1.673E-10	3.97	7.461E-11	3.94	20	0.518
[507, 507]	1.088E-11	3.88	4.669E-12	3.93	21	2.406

Table 4
Example 2 – Fourth order numerical error analysis of AMIB vs MIB.

$[N_x, N_y]$	AMIB4					
	L_∞		L_2		CPU time(s)	iter no.
	Error	Order	Error	Order		
[11, 11]	4.408E-5	–	1.549E-5	–	4.823E-3	20
[27, 27]	1.096E-6	3.87	4.712E-7	3.65	1.290E-2	20
[59, 59]	4.662E-8	3.94	2.209E-8	3.81	4.056E-2	20
[123, 123]	2.442E-9	3.97	1.212E-9	3.90	0.139	21
[251, 251]	1.400E-10	3.98	7.111E-11	3.95	0.529	21
[507, 507]	7.771E-12	4.10	4.004E-12	4.08	2.403	21
$[N_x, N_y]$	MIB4					
	L_∞		L_2		CPU time(s)	
	Error	Order	Error	Order		
[13, 13]	4.070E-5	–	1.841E-5	–	2.00E-3	
[29, 29]	1.076E-6	4.28	5.267E-7	4.19	2.466E-2	
[61, 61]	4.644E-8	4.12	2.345E-8	4.08	0.207	
[125, 125]	2.444E-9	4.06	1.252E-9	4.04	1.836	
[253, 253]	1.423E-10	4.00	7.329E-11	4.00	15.386	
[509, 509]	3.363E-11	2.06	1.417E-11	2.34	148.41	

$$u + \frac{\partial u}{\partial n} = (\cos(\frac{k\pi}{3}) - k\sin(\frac{k\pi}{3}))\sin(kx) \quad \text{on} \quad \Gamma_3,$$
$$u_x = k\cos(\frac{k\pi}{3})\cos(ky) \qquad \text{on} \quad \Gamma_4.$$

It has analytical solution $u = \sin(kx)\cos(ky)$, in which the wave number k could be chosen as a large number. blue Basically, only for highly oscillating solutions, high-order convergence can be detected numerically. In this example, the parameter k is uniformly set to be 15 for testing the AMIB4, AMIB6, and AMIB8 with mixed boundary conditions.

The high-order convergence of the AMIB method is validated in Table 5. It is clear that the AMIB4, AMIB6, and AMIB8 achieves, respectively, the fourth, sixth, and eighth order of convergence. Moreover, the AMIB8 is obviously the most accurate scheme.

The iteration numbers are also reported in Table 5. Two comments are in order. First, the iteration number increases with respect to the mesh size for this Poisson problem. But such increment is slow and weakly depends on the mesh size - when the mesh size is doubled for four times, the iteration number is doubled once. Second, the iteration number seems to not depend on the order of AMIB method. This demonstrates the robustness of the AMIB method.

Example 4. This example is devoted to test the AMIB method for solving the Helmholtz equation

$$u_{xx} + u_{yy} + k^2 u = e^{-x^2-\frac{y^2}{2}}(4x^2 + y^2 + k^2 - 3)$$

subject to Robin boundary conditions:

$$u + \frac{\partial u}{\partial n} = (1 - \frac{\pi}{3})e^{-x^2-\frac{\pi^2}{18}} \quad \text{on} \quad \Gamma_1,$$

Table 5
Example 3 – High order numerical error analysis.

$[N_x, N_y]$	AMIB4 and $k = 15$				
	L_∞		L_2		iter no.
	Error	Order	Error	Order	
[27, 27]	6.125E-2	-	1.368E-2	-	30
[59, 59]	3.000E-3	3.76	5.575E-4	3.99	30
[125, 125]	1.476E-4	4.05	2.848E-5	4.00	34
[251, 251]	8.521E-6	3.97	1.614E-6	4.00	43
[507, 507]	5.072E-7	4.00	9.612E-8	4.00	57
$[N_x, N_y]$	AMIB6 and $k = 15$				
	L_∞		L_2		iter no.
	Error	Order	Error	Order	
[25, 25]	0.135	–	2.001E-2	–	31
[57, 57]	9.283E-4	5.87	1.496E-4	5.78	36
[123, 123]	7.120E-6	6.39	1.078E-6	6.47	41
[249, 249]	8.167E-8	6.15	1.051E-8	6.38	50
[505, 505]	1.070E-9	6.11	1.317E-10	6.18	69
$[N_x, N_y]$	AMIB8 and $k = 15$				
	L_∞		L_2		iter no.
	Error	Order	Error	Order	
[23, 23]	0.140	-	2.354E-2	-	34
[55, 55]	3.103E-4	6.81	4.482E-5	6.97	42
[121, 121]	4.131E-7	8.47	6.780E-8	8.31	48
[247, 247]	9.587E-10	8.25	1.242E-10	8.57	57
[503, 503]	3.173E-12	8.01	3.760E-13	8.13	70

Table 6
Example 4 – Fourth order numerical error analysis for the Helmholtz equation with $k = 20$.

$[N_x, N_y]$	AMIB4				
	L_∞		L_2		iter no.
	Error	Order	Error	Order	
[27, 27]	6.255E-7	–	2.218E-7	–	53
[59, 59]	1.1732E-8	4.96	5.777E-9	4.55	38
[123, 123]	9.682E-10	3.35	3.307E-10	3.85	64
[251, 251]	5.751E-11	3.94	2.014E-11	3.90	75
[507, 507]	3.502E-12	3.97	1.246E-12	3.95	94

$$u + \frac{\partial u}{\partial n} = (1 + \frac{2\pi}{3})e^{-\frac{\pi^2}{9} - \frac{y^2}{2}} \quad \text{on } \Gamma_2,$$
$$u + \frac{\partial u}{\partial n} = (1 + \frac{\pi}{3})e^{-x^2 - \frac{\pi^2}{18}} \quad \text{on } \Gamma_3,$$
$$u + \frac{\partial u}{\partial n} = (1 - \frac{2\pi}{3})e^{-\frac{\pi^2}{9} - \frac{y^2}{2}} \quad \text{on } \Gamma_4,$$

which are determined by analytical solution $u = e^{-x^2 - \frac{y^2}{2}}$ on domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$. Parameter k is chosen to be 20 in the test. Table 6 shows that the AMIB4 method is equally successful in solving the Helmholtz equation with Robin boundary conditions.

Example 5. Both boundary condition implementation and solution of algebraic system are conducted in dimension by dimension manner in the AMIB method. Thus, the generalization of 2D AMIB method to 3D is straightforward. Here the AMIB4 method is verified by the following 3D Poisson problem:

$$u_{xx} + u_{yy} + u_{zz} = \sin(kx)e^{y+z}(2 - k^2)$$

with Dirichlet boundary conditions determined by solution $u = \sin(kx)e^{y+z}$ on the domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$. Parameter k is set to be 5 in the 3D test. The fourth order convergence of the AMIB4 for 3D Poisson problem is numerically verified through Table 7. Moreover, the iteration number increases very little when the mesh size is doubled in each dimension.

Table 7
Example 5 –Fourth order numerical error analysis of the AMIB4 on 3D Poisson's equation.

$[N_x, N_y, N_z]$	AMIB4					
	L_∞		L_2		iter no.	CPU times(s)
	Error	Order	Error	Order		
$[11, 11, 11]$	0.870	–	0.127	–	37	4.722E-2
$[27, 27, 27]$	3.337E-2	3.81	3.347E-3	3.80	49	0.392
$[59, 59, 59]$	1.426E-3	4.13	1.216E-4	4.13	52	3.191
$[123, 123, 123]$	7.436E-5	4.10	5.760E-6	4.10	55	28.792
$[251, 251, 251]$	4.270E-6	4.06	3.132E-7	4.06	55	296.97

Table 8
Computational efficiency comparison.

$[N_x, N_y]$	FISHPACK					
	L_∞		L_2		CPU time (s)	
	Error	Order	Error	Order		
$[65, 65]$	2.384E-3	–	9.717E-4	–	1.098E-3	
$[129, 129]$	6.073E-4	1.97	2.464E-4	1.98	4.837E-3	
$[257, 257]$	1.530E-4	1.99	6.207E-5	1.99	1.665E-2	
$[513, 513]$	3.840E-5	1.99	1.558E-5	1.99	6.084E-2	
$[1025, 1025]$	9.617E-6	2.00	3.902E-6	2.00	0.243	
$[N_x, N_y]$	AMIB2					
	L_∞		L_2		iter no.	CPU time (s)
	Error	Order	Error	Order		
$[61, 61]$	2.790E-3	–	1.087E-3	–	10	1.92E-2
$[125, 125]$	6.565E-4	1.99	2.604E-4	1.97	10	7.432E-2
$[253, 253]$	1.590E-4	2.00	6.380E-5	1.98	12	0.311
$[509, 509]$	3.915E-5	2.00	1.579E-5	1.99	12	1.471
$[1021, 1021]$	9.711E-6	2.00	3.929E-6	2.00	12	6.267
$[N_x, N_y]$	AMIB4					
	L_∞		L_2		iter no.	CPU time (s)
	Error	Order	Error	Order		
$[59, 59]$	1.273E-5	–	4.764E-6	–	19	4.100E-2
$[123, 123]$	6.609E-7	3.98	2.570E-7	3.93	19	0.127
$[251, 251]$	3.768E-8	3.99	1.496E-8	3.97	20	0.516
$[507, 507]$	2.250E-9	4.00	9.03E-10	3.98	20	2.284
$[1019, 1019]$	1.374E-10	4.00	5.544E-11	4.00	21	10.613

6.2. *Computational efficiency*

In this section, the computational efficiency of the AMIB method will be further investigated. A comparison between the results from FISHPACK [42], AMIB2 and AMIB4 will be considered in 2D. In the FISHPACK, a direct FFT implementation is conducted for the second order central difference. Theoretically, the computational cost of FISHPACK is of $O(N^2 \log N)$, where N is the degree of freedom in each dimension on a squared domain. In the proposed AMIB method, an iterative solution is needed for the Schur complement system, whose size is one-dimensionally smaller. Because the iteration number weakly depends on N, the computational cost of the AMIB method could be on the order of $O(N^2 \log N)$ too. This will be numerically verified in this subsection.

We note two things in our flop order test. First, the change of $\log N$ in comparing with N is almost negligible in the total computational cost. So we have to drop this term in complexity in flop order test. In particular, we will fit the CPU time to the form of $O(N^r)$ by the least squares method. The numerically detected flop order r will characterize the efficiency of the AMIB method. Second, we re-studied Example 3 from the last section by changing boundary conditions. This is because FISHPACK cannot handle Robin condition. Instead, Dirichlet boundary conditions determined by exact solutions are employed.

In the experiment, the parameter k is set to be 5. We adopt 10^{-10} as the error tolerance for the AMIB2, as it already guarantees the expected convergence. For the AMIB4, the tolerance is still 10^{-15}. From Table 8, we can see that both AMIB2 and FISHPACK produce second order accuracy. The AMIB4 has the expected fourth order accuracy. To analyze the flop order, the CPU time is plotted against the degree of freedom in each dimension, i.e., N, in Fig. 5. For a given N, AMIB algorithm is more expensive than FISHPACK, while the CPU time difference between AMIB2 and AMIB4 is small. Moreover, least squares fitting is conducted to compute flop order r in $O(N^r)$ setting. From Fig. 5, it can be observed that the flop order r for each method is around 2. Thus, the computational cost of the AMIB method is roughly on the order of $O(N^2 \log N)$ too, whereas the AMIB method has larger proportionality constants on the operation count $O(N^2 \log N)$ compared to FISHPACK.

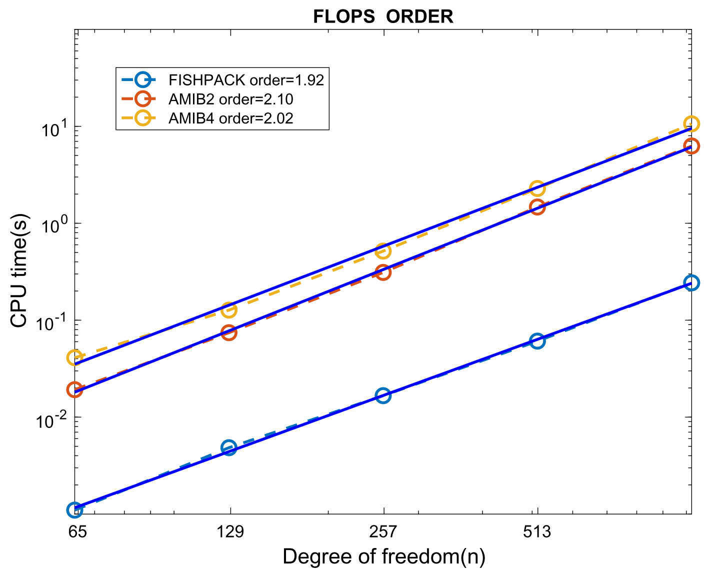

Fig. 5. A comparison of computational cost between three methods, and flop orders are tested to be around 2 for each method.

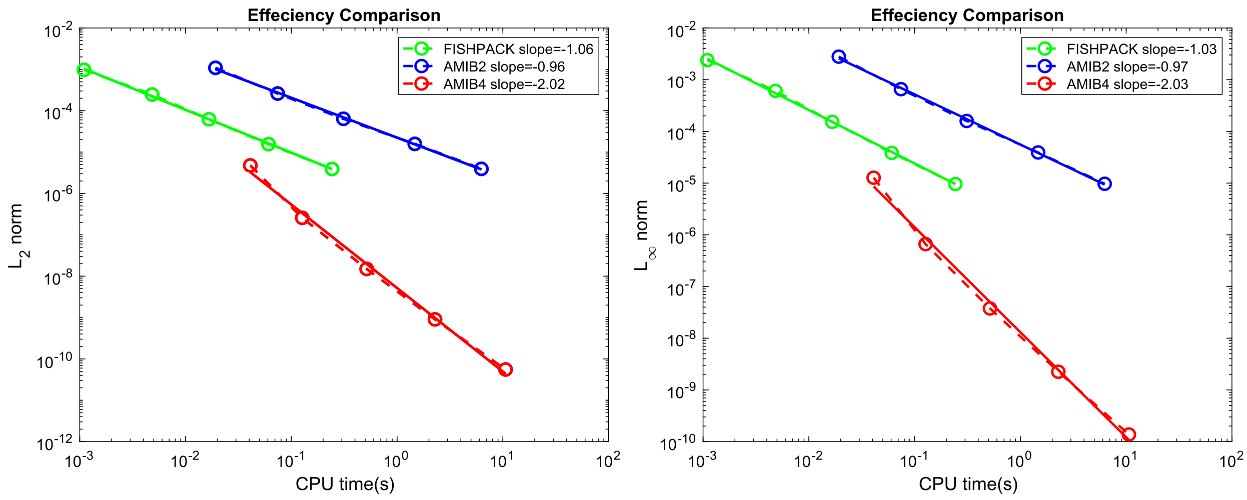

Fig. 6. A comparison between the three methods on accuracy VS. CPU time under two norms.

On the other hand, in terms of cost-efficiency, high order methods have the advantage of providing better accuracy than lower order methods on the same mesh size. Table 8 shows that the AMIB4 has accuracy significantly improved than the second order methods. In fact, the errors of the AMIB4 on a coarse mesh 65×65 are smaller than those of second order AMIB2 and FISHPACK on 513×513. To demonstrate the cost-efficiency, we plot the accuracy against CPU time in Fig. 6 for both error norms. Obviously, the AMIB2 scheme as a byproduct of this study, is dominated by the FISHPACK - they are of the same accuracy level, while the AMIB2 is slower. Nevertheless, it can be observed in this figure that to achieve a precision around 10^{-5}, the proposed AMIB4 method is more efficient than the FISHPACK. Moreover, the efficiency gain of the AMIB4 becomes more significant when a better accuracy is targeted. For log-log lines shown in Fig. 6, we calculated the slopes by least squares fitting. The AMIB4 yields a slope around -2, while that of FISHPACK is about -1. Based on these results, it is projected that to achieve a precision around 10^{-10}, the FISHPACK would take about 10^5 seconds of CPU time (assume memory is large enough), which is about $10,000$ times more expensive than the proposed AMIB4 scheme.

7. Summary

In this work, a FFT-based high order fast Poisson solver is developed for a cubic type geometry in multi-dimensions. By introducing a thin fictitious buffer zone or zero-padding region outside the domain, the Poisson boundary value problem is converted into an immersed boundary problem, so that FFT inversion of high order central difference schemes becomes feasible. The boundary conditions then become interface conditions, which are discretized by the matched interface and boundary (MIB) method for generating fictitious nodes. The fictitious values are then utilized to approximate Cartesian derivative jumps on the boundary, which are introduced as auxiliary variables in the augmented MIB (AMIB) formulation. The Schur complement method allows us to iteratively solve these auxiliary variables over a one-dimensionally smaller algebraic system. Moreover, the iteration number for the Schur complement system weakly depends on the degree of freedom N in each dimension. Hence, even though the AMIB method involves an iteration process, which is essentially not a direct fast solver, it still gives a fast Poisson solution. The numerical flop counts of the AMIB method are the order of $O(N^2 \log N)$ for a two-dimensional problem.

Traditional high-order fast Poisson solvers are mostly based on compact finite difference or spectral methods. This study is the first of its kind based on high order central differences. As a systematical approach, the AMIB method can easily handle the usual Dirichlet, Neumann, Robin or any combination of boundary conditions. Moreover, based on tensor product principle, the AMIB method can be implemented in multi-dimensions. Furthermore, the AMIB method can be made to arbitrarily high order in principle. Up to 8th order is demonstrated in this work. Finally, we note that due to broad application of high order central difference schemes, the AMIB method has potential to be applied to many problems with ease to gain a fast solution, including more complicated boundary conditions, boundary value problems over irregular domains, and interface problems. The future plan is to use the AMIB method to tackle irregular domain and interface problems.

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) grant DMS-1812930 and the Simons Foundation award 524151.

Appendix A

A.1. An extension on high order algorithms via FFT

The fourth order algorithms via FFT have been demonstrated in great details. High order algorithms, for example sixth and eighth order, can be derived with minor modification on assumptions and parameters. Below lists the procedures for sixth and eighth order schemes in one dimensional manner. Reader interested in such high order algorithm could easily go through the mathematical formulation in two or three dimensions.

A.1.1. Sixth order FFT solution for one-dimensional Poisson equation

The anti-symmetry for sixth order study can be assumed as

$$u_0 = 0, u_{-1} = -u_1, u_{-2} = -u_2,$$
$$u_M = 0, u_{M+1} = -u_{M-1}, u_{M+2} = -u_{M-2}.$$

The procedure for one dimensional FFT solution of sixth order can be developed as below:

1. Compute Sine transform for $f(x_j)$ via IFST $\hat{f}_l = \frac{2}{M} \sum_{j=1}^{M-1} f(x_j) \sin(\frac{lj\pi}{M})$, for $l = 1, \cdots, M-1$.
2. $\hat{u}_l = \frac{\hat{f}_l}{\lambda_l}$, for $l = 1, \cdots, M-1$, where $\lambda_l = \frac{1}{h_x^2}[\frac{1}{45}\cos(\frac{3l\pi}{M}) - \frac{3}{10}\cos(\frac{2l\pi}{M}) + 3\cos(\frac{l\pi}{M}) - \frac{49}{18}]$.
3. Compute u_j via FST: $u_j = \sum_{l=1}^{M-1} \sin(\frac{lj\pi}{M})$, for $j = 1, \cdots, M-1$.

A.1.2. Eighth order FFT solution for one-dimensional Poisson equation

For the eighth order study, we further assume the following anti-symmetry

$$u_0 = 0, u_{-1} = u_1, u_{-2} = -u_2, u_{-3} = -u_3,$$
$$u_M = 0, u_{M+1} = -u_{M-1}, u_{M+2} = -u_{M-2}, u_{M+3} = -u_{M-3}.$$

Therefore the procedure for the FFT solution of eighth order is formulated as below:

1. Compute Sine transform for $f(x_j)$ via IFST $\hat{f}_l = \frac{2}{M} \sum_{j=1}^{M-1} f(x_j) \sin(\frac{lj\pi}{M})$, for $l = 1, \cdots, M-1$.

2. $\hat{u}_l = \frac{\hat{f}_l}{\lambda_l}$, for $l = 1, \cdots, M-1$,

$$\lambda_l = \frac{1}{h_x^2}[-\frac{1}{280}\cos(\frac{4l\pi}{M}) + \frac{16}{315}\cos(\frac{3l\pi}{M}) - \frac{2}{5}\cos(\frac{2l\pi}{M}) + \frac{16}{5}\cos(\frac{l\pi}{M}) - \frac{205}{72}].$$

3. Compute u_j via FST: $u_j = \sum_{l=1}^{M-1} \sin(\frac{lj\pi}{M})$, for $j = 1, \cdots, M-1$.

A.2. An extension on high order corrected differences

Fourth order corrected differences have been explained in the body part of the work. An further extension on high order corrected differences are illustrated below.

Theorem 2. Corrected sixth differences. Let $x_j \le \alpha < x_{j+1}, h^- = x_j - \alpha$, and $h^+ = x_{j+1} - \alpha$. Suppose $u \in C^8[x_j - 3h, \alpha) \bigcap C^8(\alpha, x_{j+1} + 3h]$, with derivative extending continuously up to the interface α. Then the following approximations hold to $O(h^6)$:

$$\begin{aligned}
u_{xx}(x_{j-2}) \approx & \frac{1}{h^2}[\frac{1}{90}u(x_{j-5}) - \frac{3}{20}u(x_{j-4}) + \frac{3}{2}u(x_{j-3}) - \frac{49}{18}u(x_{j-2}) + \frac{3}{2}u(x_{j-1}) \\
& -\frac{3}{20}u(x_j) + \frac{1}{90}u(x_{j+1})] - \frac{1}{90h^2}\sum_{m=0}^{7}\frac{(h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j-1}) \approx & \frac{1}{h^2}[\frac{1}{90}u(x_{j-4}) - \frac{3}{20}u(x_{j-3}) + \frac{3}{2}u(x_{j-2}) - \frac{49}{18}u(x_{j-1}) + \frac{3}{2}u(x_j) \\
& -\frac{3}{20}u(x_{j+1}) + \frac{1}{90}u(x_{j+2})] \\
& -\frac{1}{90h^2}\sum_{m=0}^{7}\frac{(h+h^+)^m}{m!}[u^{(m)}] + \frac{3}{20h^2}\sum_{m=0}^{7}\frac{(h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_j) \approx & \frac{1}{h^2}[\frac{1}{90}u(x_{j-3}) - \frac{3}{20}u(x_{j-2}) + \frac{3}{2}u(x_{j-1}) - \frac{49}{18}u(x_j) + \frac{3}{2}u(x_{j+1}) \\
& -\frac{3}{20}u(x_{j+2}) + \frac{1}{90}u(x_{j+3})] - \frac{1}{90h^2}\sum_{m=0}^{7}\frac{(2h+h^+)^m}{m!}[u^{(m)}] \\
& +\frac{3}{20h^2}\sum_{m=0}^{7}\frac{(h+h^+)^m}{m!}[u^{(m)}] - \frac{3}{2h^2}\sum_{m=0}^{7}\frac{(h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j+1}) \approx & \frac{1}{h^2}[\frac{1}{90}u(x_{j-2}) - \frac{3}{20}u(x_{j-1}) + \frac{3}{2}u(x_j) - \frac{49}{18}u(x_{j+1}) + \frac{3}{2}u(x_{j+2}) \\
& -\frac{3}{20}u(x_{j+3}) + \frac{1}{90}u(x_{j+4})] + \frac{1}{90h^2}\sum_{m=0}^{7}\frac{(h^- - 2h)^m}{m!}[u^{(m)}] \\
& -\frac{3}{20h^2}\sum_{m=0}^{7}\frac{(h^- - h)^m}{m!}[u^{(m)}] + \frac{3}{2h^2}\sum_{m=0}^{7}\frac{(h^-)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j+2}) \approx & \frac{1}{h^2}[\frac{1}{90}u(x_{j-1}) - \frac{3}{20}u(x_j) + \frac{3}{2}u(x_{j+1}) - \frac{49}{18}u(x_{j+2}) + \frac{3}{2}u(x_{j+3}) \\
& -\frac{3}{20}u(x_{j+4}) + \frac{1}{90}u(x_{j+5})] \\
& +\frac{1}{90h^2}\sum_{m=0}^{7}\frac{(h^- - h)^m}{m!}[u^{(m)}] - \frac{3}{20h^2}\sum_{m=0}^{7}\frac{(h^-)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j+3}) \approx & \frac{1}{h^2}[\frac{1}{90}u(x_j) - \frac{3}{20}u(x_{j+1}) + \frac{3}{2}u(x_{j+2}) - \frac{49}{18}u(x_{j+3}) + \frac{3}{2}u(x_{j+4}) \\
& -\frac{3}{20}u(x_{j+5}) + \frac{1}{90}u(x_{j+6})] + \frac{1}{90h^2}\sum_{m=0}^{7}\frac{(h^-)^m}{m!}[u^{(m)}].
\end{aligned}$$

Here note that we may just make m range from 0 to 6 to ensure sixth order accuracy for elliptic problem globally despite local fifth order accuracy.

Theorem 3. Corrected eighth differences. Let $x_j \le \alpha < x_{j+1}, h^- = x_j - \alpha$, and $h^+ = x_{j+1} - \alpha$. Suppose $u \in C^{10}[x_j - 4h, \alpha) \bigcap C^{10}(\alpha, x_{j+1} + 4h]$, with derivative extending continuously up to the interface α. Then the following approximations hold to $O(h^8)$:

$$\begin{aligned}
u_{xx}(x_{j-3}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-7}) + \frac{8}{315}u(x_{j-6}) - \frac{1}{5}u(x_{j-5}) + \frac{8}{5}u(x_{j-4}) - \frac{205}{72}u(x_{j-3}) \\
& + \frac{8}{5}u(x_{j-2}) - \frac{1}{5}u(x_{j-1}) + \frac{8}{315}u(x_j) - \frac{1}{560}u(x_{j+1})] + \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j-2}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-6}) + \frac{8}{315}u(x_{j-5}) - \frac{1}{5}u(x_{j-4}) + \frac{8}{5}u(x_{j-3}) - \frac{205}{72}u(x_{j-2}) \\
& + \frac{8}{5}u(x_{j-1}) - \frac{1}{5}u(x_j) + \frac{8}{315}u(x_{j+1}) - \frac{1}{560}u(x_{j+2})] \\
& - \frac{8}{315h^2}\sum_{m=0}^{9}\frac{(h^+)^m}{m!}[u^{(m)}] + \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(h+h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j-1}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-5}) + \frac{8}{315}u(x_{j-4}) - \frac{1}{5}u(x_{j-3})\frac{8}{5}u(x_{j-2}) - \frac{205}{72}u(x_{j-1}) \\
& + \frac{8}{5}u(x_j) - \frac{1}{5}u(x_{j+1}) + \frac{8}{315}u(x_{j+2}) - \frac{1}{560}u(x_{j+3})] + \frac{1}{5h^2}\sum_{m=0}^{9}\frac{(h^+)^m}{m!}[u^{(m)}] \\
& - \frac{8}{315h^2}\sum_{m=0}^{9}\frac{(h+h^+)^m}{m!}[u^{(m)}] + \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(2h+h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_j) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-4}) + \frac{8}{315}u(x_{j-3}) - \frac{1}{5}u(x_{j-2}) + \frac{8}{5}u(x_{j-1}) - \frac{205}{72}u(x_j) \\
& + \frac{8}{5}u(x_{j+1}) - \frac{1}{5}u(x_{j+2}) + \frac{8}{315}u(x_{j+3}) - \frac{1}{560}u(x_{j+4})] \\
& - \frac{8}{5}\frac{1}{h^2}\sum_{m=0}^{9}\frac{(h^+)^m}{m!}[u^{(m)}] + \frac{1}{5h^2}\sum_{m=0}^{9}\frac{(h+h^+)^m}{m!}[u^{(m)}] \\
& - \frac{8}{315h^2}\sum_{m=0}^{9}\frac{(2h+h^+)^m}{m!}[u^{(m)}] + \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(3h+h^+)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j+1}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-3}) + \frac{8}{315}u(x_{j-2}) - \frac{1}{5}u(x_{j-1}) + \frac{8}{5}u(x_j) - \frac{205}{72}u(x_{j+1}) \\
& + \frac{8}{5}u(x_{j+2}) - \frac{1}{5}u(x_{j+3}) + \frac{8}{315}u(x_{j+4}) - \frac{1}{560}u(x_{j+5})] \\
& + \frac{8}{5}\frac{1}{h^2}\sum_{m=0}^{9}\frac{(h^-)^m}{m!}[u^{(m)}] - \frac{1}{5h^2}\sum_{m=0}^{9}\frac{(h^- - h)^m}{m!}[u^{(m)}] \\
& + \frac{8}{315h^2}\sum_{m=0}^{9}\frac{(h^- - 2h)^m}{m!}[u^{(m)}] - \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(h^- - 3h)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j+2}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-2}) + \frac{8}{315}u(x_{j-1}) - \frac{1}{5}u(x_j) + \frac{8}{5}u(x_{j+1}) - \frac{205}{72}u(x_{j+2}) \\
& + \frac{8}{5}u(x_{j+3}) - \frac{1}{5}u(x_{j+4}) + \frac{8}{315}u(x_{j+5}) - \frac{1}{560}u(x_{j+6})] - \frac{1}{5h^2}\sum_{m=0}^{9}\frac{(h^-)^m}{m!}[u^{(m)}] \\
& + \frac{8}{315h^2}\sum_{m=0}^{9}\frac{(h^- - h)^m}{m!}[u^{(m)}] - \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(h^- - 2h)^m}{m!}[u^{(m)}],
\end{aligned}$$

$$
\begin{aligned}
u_{xx}(x_{j+3}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_{j-1}) + \frac{8}{315}u(x_j) - \frac{1}{5}u(x_{j+1}) + \frac{8}{5}u(x_{j+2}) - \frac{205}{72}u(x_{j+3}) \\
& + \frac{8}{5}u(x_{j+4}) - \frac{1}{5}u(x_{j+5}) + \frac{8}{315}u(x_{j+6}) - \frac{1}{560}u(x_{j+7})] \\
& + \frac{8}{315h^2}\sum_{m=0}^{9}\frac{(h^-)^m}{m!}[u^{(m)}] - \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(h^- - h)^m}{m!}[u^{(m)}], \\
u_{xx}(x_{j+4}) \approx & \frac{1}{h^2}[-\frac{1}{560}u(x_j) + \frac{8}{315}u(x_{j+1}) - \frac{1}{5}u(x_{j+2}) + \frac{8}{5}u(x_{j+3}) - \frac{205}{72}u(x_{j+4}) \\
& + \frac{8}{5}u(x_{j+5}) - \frac{1}{5}u(x_{j+6}) + \frac{8}{315}u(x_{j+7}) - \frac{1}{560}u(x_{j+8})] - \frac{1}{560h^2}\sum_{m=0}^{9}\frac{(h^-)^m}{m!}[u^{(m)}].
\end{aligned}
$$

Here note that we may just make m range from 0 to 8 to ensure eighth order accuracy for elliptic problem globally despite local seventh order accuracy.

References

[1] L. Adams, Z.L. Li, The immersed interface/multigrid methods for interface problems, SIAM J. Sci. Comput. 24 (2002) 463–479.

[2] 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.

[3] P.A. Berthelsen, A decomposed immersed interface method for variable coefficient elliptic equations with non-smooth and discontinuous solutions, J. Comput. Phys. 197 (2004) 364–386.

[4] B. Bialecki, G. Fairweather, A. Karageorghis, Matrix decomposition algorithms for elliptic boundary value problems: a survey, Numer. Algorithms 56 (2011) 253–295.

[5] R.F. Boisvert, A fourth order accurate Fourier method for the Helmholtz equation in three dimensions, ACM Trans. Math. Softw. 13 (1987) 221–234.

[6] E. Braverman, M. Israeli, A. Averbuch, L. Vozovoi, A fast 3D Poisson solver of arbitrary order accuracy, J. Comput. Phys. 144 (1998) 109–136.

[7] E. Braverman, M. Israeli, A. Averbuch, A fast spectral solver for a 3D Helmholtz equation, SIAM J. Sci. Comput. 20 (1999) 2237–2260.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] J. Douglas Jr., On the numerical integration of $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial u}{\partial t}$ by implicit method, J. Soc. Ind. Appl. Math. 3 (1995) 42–65.

[13] H. Feng, S. Zhao, An augmented matched interface and boundary (MIB) method for solving elliptic interface problem, J. Comput. Appl. Math. 361 (2019) 426–443.

[14] M. Frigo, S.G. Johnson, The design and implementation of FFTW3, Proc. IEEE 93 (2005) 216–231.

[15] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefen problem, J. Comput. Phys. 202 (2005) 577–601.

[16] M.M. Gupta, J. Kouatchou, J. Zhang, Comparison of second and fourth order discretization multigrid Poisson solvers, J. Comput. Phys. 132 (1997) 226–232.

[17] 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.

[18] D. Haidvoge, T. Zang, The accurate solution of Poisson's equation by expansion in Chebyshev polynomials, J. Comput. Phys. 30 (1979) 167–180.

[19] J. Hendrickx, R. Vandebril, M. Van Barel, A fast direct method for solving the two-dimensional Helmholtz equation with Robbins boundary conditions, in: Fast Algorithms for Structured Matrices: Theory and Applications, in: Contemp. Math., vol. 323, Amer. Math. Soc., Providence, Rhode Island, 2003, pp. 187–204.

[20] R.W. Hockney, A fast direct solution of Poisson's equation using Fourier analysis, J. Assoc. Comput. Mach. 12 (1965) 95–113.

[21] E.N. Houstis, T.S. Papatheordorou, High order fast elliptic equation solver, ACM Trans. Math. Softw. 5 (1979) 431–441.

[22] L. Huang, C.W. Shu, M. Zhang, Numerical boundary conditions for the fast sweeping high order WENO methods for solving the Eikonal equation, J. Comput. Math. 26 (2008) 1–11.

[23] S. Karaa, J. Zhang, High order ADI method for solving unsteady convection–diffusion problems, J. Comput. Phys. 198 (2004) 1–9.

[24] M.C. Lai, A simple compact fourth-order Poisson solver on polar geometry, J. Comput. Phys. 182 (2002) 337–345.

[25] R.J. LeVeque, Z.L. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.

[26] T. Li, C.W. Shu, M. Zhang, Stability analysis of the inverse Lax-Wendroff boundary treatment for high order central difference schemes for diffusion equations, J. Sci. Comput. 70 (2017) 576–607.

[27] Z.L. Li, A fast iterative algorithm for elliptic interface problem, SIAM J. Numer. Anal. 35 (1998) 230–254.

[28] M. Lyon, O. Bruno, High-order unconditionally stable FC-AD solvers for general smooth domains II. Elliptic, parabolic and hyperbolic PDEs; theoretical considerations, J. Comput. Phys. 229 (2010) 3358–3381.

[29] 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.

[30] J.R. Nagel, Solving the Generalized Poisson's Equation Using the Finite-Difference Method (FDM), Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, 2011.

[31] V. Pereyra, On improving the approximate solution of a functional equation by deferred correction, Numer. Math. 8 (1966) 376–391.

[32] V. Pereyra, W. Preskurowski, O. Widlund, High order fast Laplace solvers for the Dirichlet problem on general regions, Math. Comput. 31 (1997) 1–16.

[33] W.M. Pickering, P.J. Harley, On Robbins boundary conditions, elliptic equations and FFT methods, J. Comput. Phys. 122 (1995) 380–383.

[34] W. Preskurowski, Algorithm 593: a package for the Helmholtz equation in nonrectangular planar region, ACM Trans. Math. Softw. 9 (1983) 117–124.
[35] 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.
[36] J. Shen, T. Tang, L.L. Wang, Spectral Methods: Algorithm, Analysis and Application, Springer Series in Computational Mathematics, 2011.
[37] 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.
[38] U. Schumann, R. Sweet, Fast Fourier transforms for direct solution of Poisson's equation with staggered boundary conditions, J. Comput. Phys. 75 (1988) 123–137.
[39] 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.
[40] P.N. Swarztrauber, The methods of cyclic reduction, Fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle, SIAM Rev. 19 (1977) 490–501.
[41] P.N. Swarztrauber, Symmetric FFTs, Math. Comput. 47 (1986) 323–346.
[42] P. Swarztrauber, R. Sweet, Algorithm 541: efficient Fortran subprograms for the solution of separable elliptic partial differential equations, ACM Trans. Math. Softw. 5 (1979) 352–364.
[43] S. Tan, C.W. Shu, Inverse Lax-Wendroff procedure for numerical boundary conditions of conservation laws, J. Comput. Phys. 229 (2010) 8144–8166.
[44] H. Wang, Y. Zhang, X. Ma, J. Qiu, Y. Liang, An efficient implementation of fourth-order compact finite difference scheme for Poisson's equation with Dirichlet boundary conditions, Comput. Math. Appl. 71 (2016) 1843–1860.
[45] 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.
[46] 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.
[47] Y. Xie, W. Ying, A fourth-order kernel-free boundary integral method for the modified Helmholtz equation, J. Sci. Comput. 78 (2019) 1632–1658.
[48] S. Zhao, G.W. Wei, High-order FDTD methods via derivative matching for Maxwell's equations with material interfaces, J. Comput. Phys. 200 (2004) 60–103.
[49] 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.
[50] S. Zhao, On the spurious solutions in the high-order finite difference methods, Comput. Methods Appl. Math. 196 (2007) 5031–5046.
[51] S. Zhao, A fourth order finite difference method for waveguides with curved perfectly conducting boundaries, Comput. Methods Appl. Math. 199 (2010) 2655–2662.
[52] 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.
[53] Y.C. Zhou, S. Zhao, M. Feig, G.W. Wei, High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular source, J. Comput. Phys. 213 (2006) 1–30.
[54] Y. Zhu, A.C. Cangellaris, Multigrid Finite Element Methods for Electromagnetic Field Modeling, Wiley, 2006.