Journal of Scientific Computing (2021) 86:7
https://doi.org/10.1007/s10915-020-01379-7

# FFT-Based High Order Central Difference Schemes for Poisson's Equation with Staggered Boundaries

**Hongsong Feng[1] · Guangqing Long[2] · Shan Zhao[1]**

Received: 13 July 2020 / Revised: 28 November 2020 / Accepted: 30 November 2020


**Abstract**
This work concerns with the development of fast and high order algorithms for solving a single variable Poisson's equation with rectangular domains and uniform meshes, but involving staggered boundaries. Here the staggered boundary means that the boundary is located midway between two adjacent grid nodes. Due to the popularity of staggered grids in scientific computing for solving multiple variables partial differential equations (PDEs), the planned development deserves further studies, but is rarely reported in the literature, because grand challenges exist for spectral methods, compact finite differences, and fast Fourier transform (FFT) algorithms in handling staggered boundaries. A systematic approach is introduced in this paper to attack various open problems in this regard, which is a natural generalization of a recently developed Augmented Matched Interface and Boundary (AMIB) method for non-staggered boundaries. Formulated through immersed boundary problems with zero-padding solutions, the AMIB method combines arbitrarily high order central differences with the FFT inversion. Over staggered boundaries, the proposed AMIB method can handle Dirichlet, Neumann, Robin or any combination of boundary conditions. Convergence orders in four, six and eight are numerically validated for the AMIB method in both two and three dimensions. Moreover, the proposed AMIB method performs well for some challenging problems, such as low regularity solution near boundary, PDE solution not satisfying the boundary condition, and involving both staggered and non-staggered boundaries on two ends.

**Keywords** Fast Poisson solver · Fast Fourier transform (FFT) · High order central difference schemes · Matched interface and boundary (MIB) method · Staggered grid

**Mathematics Subject Classification** 65N06 · 65T50 · 65N85

✉ Shan Zhao
szhao@ua.edu

Hongsong Feng
hfeng5@crimson.ua.edu

Guangqing Long
longgq@nnnu.edu.cn

[1] Department of Mathematics, University of Alabama, Tuscaloosa, AL 35487, USA

[2] Department of Mathematics, Nanning Normal University, Nanning 530001, People's Republic of China

Published online: 01 January 2021

## 1 Introduction

Motivated by the extensive applications of Poisson's equation in a wide range of fields including fluid dynamics, electrostatics, and mechanical engineering, the development of *fast* and *high order* Poisson solvers has attracted a considerable interest in the literature.

A fast Poisson solver refers to an algorithm whose complexity is $O(N)$ or $O(N \log N)$, for solving a linear system with a degree of freedom $N$ arising from a numerical discretization of Poisson's equation over rectangular domains. This is significantly faster than a generic iterative solver whose complexity is on order of $O(N^2)$. The fast Fourier transform (FFT) of complexity $O(N \log N)$ is the most commonly employed tool for building fast Poisson solvers. In second order central difference approximation to Poisson problems, Fourier transform algorithms can be modified for discretization on non-staggered or staggered boundary conditions to meet special needs [21]. With Fourier analysis and synthesis well formulated for certain boundaries, pre- and post-processing [21] or symmetric FFTs [24] have been developed to achieve corresponding fast summation of complexity $O(N \log N)$. Multigrid method with computational complexity $O(N)$ is another type of algorithm widely adopted to design fast Poisson solvers. It removes the low-frequency and high-frequency of the numerical errors by successively adopting a couple of mesh levels. Then the solution on the coarse levels can be interpolated to fine levels to improve the solutions. Generally, this process needs to be done recursively before the improved solution reaches desired precision.

In recent years, there has been an increased interest in boosting fast Poisson solvers to high order, because high order methods allow the use of coarse grids so that the computational efficiency could be further accelerated. Based on FFT or multigrid algorithms, compact differences and spectral methods are frequently used to fulfill high accuracy on cubic domains. Numerous high order compact finite differences with FFT fast solvers [3,16,23,27] or multigrid fast solvers [12,13,28] have been developed in the literature. The popularity of compact finite differences is attributed to adoption of nearest neighboring nodes in the numerical differential kernel in compact schemes, which makes it relatively easier to handle different boundary conditions, and to incorporate the schemes in FFT or multigrid procedures. A series of Fourier pseudospectral (FPS) methods have been developed in [2,4,5] for solving multi-dimensional Poisson and Helmholtz equations with various boundary conditions. It can be of arbitrary order accuracy in principle, while the practical numerical order is determined by the polynomial subtraction technique for boundaries and corners treatment. Spectral-Galerkin methods can also provide fast Poisson solutions with complexity of $O(N \log N)$ [22], in which boundary conditions are satisfied by specially choosing basis functions as compact combinations of orthogonal polynomials. Recently, an Augmented Marched Interface and Boundary (AMIB) approach has been introduced in [8] as the first fast Poisson solver in the literature that combines FFT with arbitrarily high order central difference schemes. The AMIB method not only can handle the Dirichlet, Neumann, Robin or any combination of boundary conditions, but also can be easily generalized to two dimensions (2D) or three dimensions (3D). Such properties are usually not shared by compact difference and spectral methods.

This work concerns with the development of fast and high order Poisson solvers for rectangular domains and uniform meshes with staggered boundaries. It is known that staggered grids are widely used in scientific computing to solve partial differential equations (PDEs) with multiple dependent variables, such as incompressible Navier-Stokes equations [11,30], Maxwell's equations [31], and wave equations [6,15]. By defining different variables at different grid locations such as centers or edges, a staggered grid could avoid the odd-even

decoupling error of a collocated grid arrangement. Moreover, staggered grids enjoy some advantages in handling multiple variables, including better computational efficiency, smaller memory requirement and easier implementation.

In this paper, we will focus only on a single variable Poisson's equation with a uniform mesh, but allow the boundaries of the cubic domain to be staggered with the grid, i.e., the boundary locates midway between two adjacent grid nodes. We will call such a boundary as a staggered boundary, on which Dirichlet, Neumann, or Robin boundary condition needs to be satisfied. The problem to be solved can be regarded as one component needed in solving multiple variable PDEs on staggered grids. For example, the Poisson problem modeling pressure on staggered grids is derived in the projection method on unsteady incompressible Navier-Stokes equation, where solution from fast Poisson solver is desired to accelerate the computation process [11]. Moreover, high order methods have been applied to tackle many Poisson-related problems on staggered grids [6,15,30]. It is thus interested to develop fast and high order Poisson solvers for staggered boundaries.

In spite of a great success of compact difference and spectral methods for designing high order fast Poisson solvers on usual grids, few of them have been exploited for treating staggered boundaries. Considerable difficulties may arise not only in designing high order discretizations over staggered grids, but also in combining high order schemes with efficient linear solvers. For example, since a spectral method uses a global basis for approximating Poisson solutions, it seems not in the same philosophy of a shifted discretization for staggered grids, especially when the Chebyshev polynomial has roots clustered around the boundaries. As a result, high order Poisson solvers based on spectral methods and FFT implementation may not be achievable for staggered boundaries. In fact, even for second order versions, only a few fast Poisson solvers have been reported in the literature for staggered boundaries. A cyclic reduction method was proposed to obtain a second order solution of Poisson equation with staggered Neumann boundary conditions in [20]. Based on second order central difference, various Fourier transform algorithms have been constructed in [21] for different non-staggered or staggered boundary conditions. Nevertheless, there are still boundary combinations remaining unsolved for second order FFT Poisson solvers. Multigrid methods also have been exploited to fulfill second order finite differences or finite volume schemes in approximating Poisson solutions on the staggered or non-staggered boundary conditions [26,29]. The Poisson equation with staggered Neumann boundary conditions has received a special attention [1,37] due to its derivation from projection methods used in numerical simulation of incompressible fluid flows. The only known high order Poisson solver [1] in the literature combines multigrid method with fourth order compact scheme to deal with staggered Neumann boundary conditions. This is highly nontrivial, because staggered boundaries introduced additional difficulty when incorporating boundary conditions in the process of interpolation and restriction of multigrid. In summary, for Poisson's equation with staggered boundaries, the multigrid type high order solver is available only for Neumann boundary with all other conditions untouched, while FFT-based high order solvers are completely open.

The goal of this paper is to develop high order fast Poisson solvers which can handle staggered boundaries together with any boundary condition. Such a development shall be a milestone in the literature by resolving all aforementioned open problems for staggered boundaries. On the other hand, this development is a natural generalization of our recently developed AMIB method [8], which provides fast Poisson solvers on usual grids by combining arbitrarily high order central differences with FFT inversion. We note that before the AMIB approach [8], high order central difference schemes have never been applied for designing fast Poisson solvers, even though the most popular FFT fast Poisson solver is based on a second order central difference. This is because central differences with long

stencils require fictitious nodes outside the boundary, which poses a challenge to integrate boundary conditions in FFT computations. This is true in both staggered and non-staggered boundary conditions. Motivated by the AMIB scheme developed in [7,9] for solving 2D elliptic interface problems with complex geometries, a Poisson boundary value problem over a rectangular domain is converted to an immersed boundary problem with zero-padding solutions beyond the original cubic domain in [8]. Consequently, the anti-symmetric property needed in high order FFT inversions is simply satisfied over the boundary of the enlarged cubic domain, while the immersed boundary problem can be efficiently solved by a Schur complement procedure with iterative algorithms [7–9].

In this paper, we will adopt the same AMIB approach to construct high order fast Poisson solvers for staggered boundaries. Like the original AMIB method [8], the new Poisson solver can handle the Dirichlet, Neumann, and Robin boundary conditions, and can be easily implemented in multidimensions, due to its tensor product nature of discretization. Furthermore, the proposed AMIB method can be made to arbitrarily high order in principle. Finally, since the AMIB treatments at left and right boundaries are independent, problems with a regular boundary on one end and a staggered boundary on another end will be considered with any boundary condition loaded on each end. Such a study fully demonstrates the flexibility of the proposed algorithm.

The paper contains several sections as follows. The Sect. 2 reviews a systematic direct high order Poisson solver with the assumption of anti-symmetric solution across the boundary and a brief generalization for real problems. Section 3 focuses on the details about immersed boundary treatment that artificially allows for anti-symmetric solution, and the corrected differences to make high order Poisson solver feasible in the context of staggered boundary conditions. Besides, truncation error analysis of the corrected differences is presented to ensure the high order accuracy of the Poisson solver. Following this, in Sect. 4, the augmented system is formulated as the core of the method. In Sect. 5, numerical experiments are carried out to demonstrate the robustness and efficiency of the proposed method. Finally, a summary will be discussed at the end of the paper.

## 2 Background

An Augmented Matched Interface and Boundary (AMIB) method has been developed in [8] as the first fast Poisson solver that combines FFT with arbitrarily high order central difference schemes. The key ideas of the AMIB Poisson solver are briefly reviewed in this section.

### 2.1 Anti-symmetric Solution

Consider a rectangular domain with an usual grid and non-staggered boundaries. Across each boundary, assume the Poisson solution satisfying the anti-symmetric property. For such a setting, a direct FFT-based Poisson solver via high order central difference can be constructed up to arbitrary order [8]. The formulation for a fourth order solver is discussed below.

It is sufficient to limit our discussion to a one dimensional (1D) Poisson's equation

$$u_{xx}(x) = f, \quad \text{for} \quad x \in [a, b] \tag{1}$$

subject to Dirichlet zero boundary condition. Suppose a uniform partition on the domain $x \in [a, b]$ with $h_x = \frac{b-a}{M}$, where $M$ is the partition number on the given domain. As in

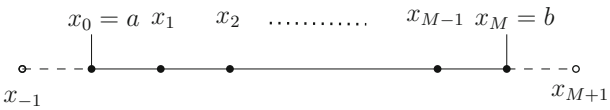


**Fig. 1** the standard grid setting in 1D

Fig. 1, the standard grid points are defined as

$$R = \{x_i | x_i = a + ih_x, i = -1, 0, 1, \ldots, M, M+1.\}.$$

Two extra grid points outside the given domain are defined due to the need of two ghost values deployed in the fourth order central difference

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

for numerical approximation at grid points neighboring the boundaries. Assume the anti-symmetry property for ghost values up to order 6:

$$u(x_0) = 0, \qquad u(x_{-1}) = -u(x_1) + O(h_x^6), \tag{3}$$

$$u(x_M) = 0, \qquad u(x_{M+1}) = -u(x_{M-1}) + O(h_x^6). \tag{4}$$

Denote $u_i$ as the approximation to the exact value of $u(x_i)$ and $f_i$ as $f(x_i)$:

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

Substituting anti-symmetric properties (3) and (4) into (2) and dropping the fourth order local truncation errors, we arrive at equations system of fourth order discretization for equation (1) :

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

Concisely, system (5) has a matrix-vector form

$$AU = F, \tag{6}$$

where $A \in \mathbb{R}^{M-1,M-1}$, $U \in \mathbb{R}^{M-1}$ and $F \in \mathbb{R}^{M-1}$. Due to the anti-symmetric solution across the boundary, coefficients of the ghost values are absorbed into coefficients of their counterparts inside the domain. The matrix $A$ is a symmetric matrix [8]. The system (6) can then be efficiently inverted using following FFT procedure with flops $O(M \log(M))$:

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

***Remark 2.1*** Thanks to the tensor product style of central difference approximation, the above fourth order one dimensional fast solvers can be easily generalized to multi-dimensions over

cubic domains. When the above assumptions are satisfied in each dimension, the resulting discretization matrix system $AU = F$ can all be quickly inverted with flops $O(M^d \log(M))$, where $d$ denotes the dimension and uniform degree of freedom of $M$ in each dimension is assumed.

***Remark 2.2*** In general, if the solution satisfies anti-symmetric property up to $(2m + 2)$th order across both ends with a positive integer $m$, then $(2m)$th order FFT solution can be realized by $(2m)$th order central differences without any boundary adjustment as in the fourth order solver. In principle, arbitrarily high order can be fulfilled using corresponding central differences with the assumption of anti-symmetric solution holding. It can be observed that coefficient matrix $A$ arising from the finite difference discretization can automatically allow for fast solution of $Ax = f$ for any right side due to the fact that $A$ is inherent to the FFT-based Poisson solvers construction. Besides, $A$ is a symmetric coefficient matrix, making its transpose still available for FFT solutions.

### 2.2 Generalization to Real Boundary Conditions

The anti-symmetric assumptions (3) and (4) are generally not true for real problems. Thus the high order AMIB method [8] introduces a thin fictitious buffer zone or zero-padding region outside the domain such that the anti-symmetric property can be automatically satisfied up to the desired orders on the boundary of the extended domain. In this way, the original boundary value problem is converted to an immersed boundary problem, as the original boundary becomes a closed interface in the extended domain, which requires high order corrected differences to restore the accuracy across the interface and to facilitate FFT inversion. The overall efficiency via FFT is accomplished in the framework of AMIB method, where FFT is used to obtain both the values of jump quantities in corrected differences and the function values. As such immersed boundary treatment can be naturally applied in staggered boundary problem, details for designing the fourth order fast solvers are shown in the following sections.

## 3 Poisson Solver for Staggered Boundaries

Consider Poisson's equation over cubic domains

$$\triangle u = f(\vec{x}), \quad \vec{x} \in \Omega \tag{7}$$

where $\vec{x} = (x_1, x_2, \ldots, x_d)$ is the spatial vector ($d = 2, 3$ for two and three dimensions), $\Omega$ is the domain of interest, and $\triangle$ is the Laplacian operator defined by

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

On staggered boundaries, three standard boundary conditions are considered :

1. Dirichlet boundary condition:

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

2. Neumann boundary condition:

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

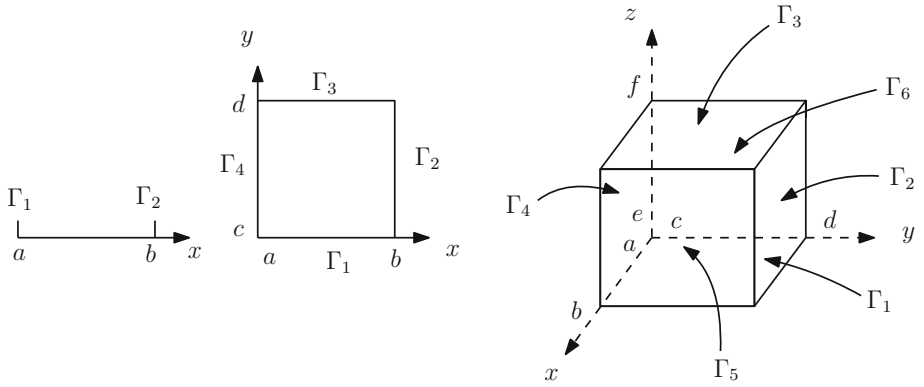


**Fig. 2** Illustration of boundary notation in 1D, 2D and 3D

3. Robin boundary condition:

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

where $\Gamma_j$ is one of the boundaries of the computational domain as shown in Fig. 2, and $\frac{\partial}{\partial n}$ denotes the outward normal derivative.

We are concerned with solution of Poisson's equation on domain $(x, y, z) \in \Omega = [a, b] \times [c, d] \times [e, f]$ with staggered boundary conditions. Choose integer $M$, $N$, $P$ such that spacing

$$h_x = \frac{b-a}{M}, h_y = \frac{d-c}{N}, h_z = \frac{f-e}{P} \tag{8}$$

are used in $x$-, $y$- and $z$-directions, respectively. There are a few key ingredients needed for high order fast Poisson solvers with staggered boundary conditions. As enough details about the FFT solvers have been given in the previous section, we now will focus on immersed boundary and corrected differences.

### 3.1 Immersed Boundary Problem

For staggered boundaries in 2D, the following grid node coordinates are defined

$$\begin{aligned} R^0 = \{(x_i, y_j) | x_i = a + (i - 0.5)h_x, \quad y_j = c + (j - 0.5)h_y, \\ i = 0, \ldots, M + 1, \quad j = 0, \ldots, N + 1\} \end{aligned} \tag{9}$$

where $h_x = \frac{b-a}{M}, h_y = \frac{d-c}{N}$ such that the boundaries of the given domain lie in midway between two grid points as shown in Fig. 3. However, as mentioned in Sect. 2.2, it is necessary to embed the original domain $\Omega$ into a larger rectangle such that the aforementioned FFT solvers can be applied when the zero solutions are defined on the extended part. The extended length of $2.5h_x$ or $2.5h_y$ for $x-$ or $y-$ direction is sufficient in supporting fourth order solvers, because the requirement of being anti-symmetric across the new boundary can be satisfied. A comparison between the current grid setting and the one for non-staggered boundaries is shown in Fig. 4. After extension, the grid coordinates are redefined as:

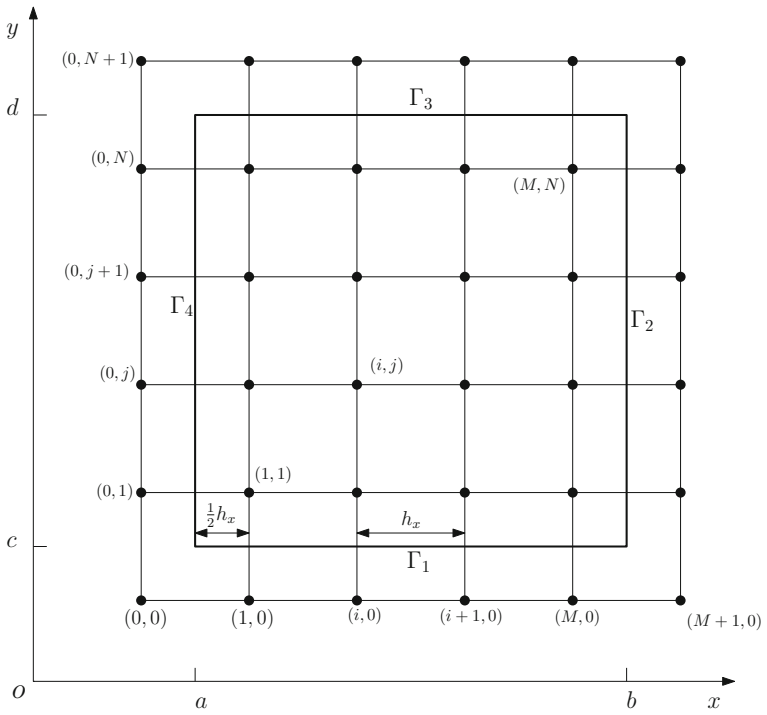


**Fig. 3** Poisson solutions on black dots in a uniform grid with staggered boundaries. Each boundary is located midway between two nodes

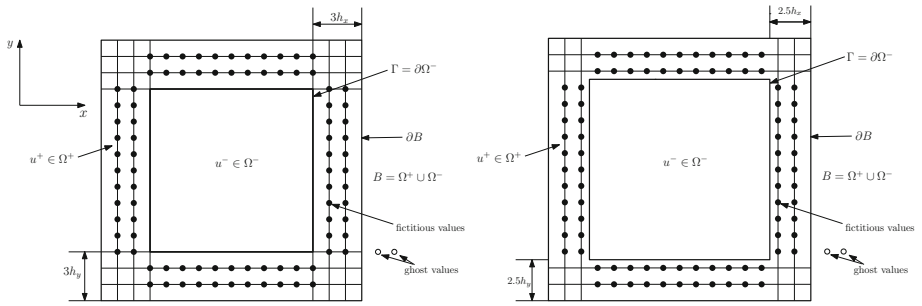


**Fig. 4** A comparison on two kinds of grids construction for immersed boundary problem with fourth order central difference scheme. The left is for usual grid, and the right one is for staggered grid. Filled circles stand for fictitious values used in approximation of jump quantities from corrected differences. *Ghost values* (depicted as open circles) are zero padding values beyond the expanded domain satisfying anti-symmetry

$$R = \{(x_i, y_j) | x_i = a + (i - 2.5)h_x, \quad y_j = c + (j - 2.5)h_y, \\ i = 0, \ldots, M + 5, \quad j = 0, \ldots, N + 5\}, \tag{10}$$

where $h_x, h_y$ are the same as above. The figure of grids in (10) would be equipped with two more layers of grid points outside each side of the domain in comparison with Fig. 3.

From now on, we denote the original domain $\Omega$ as $\Omega^-$ and the extended subdomain as $\Omega^+$, yielding the whole domain $B = \Omega^+ \cup \Omega^-$. Correspondingly, the solution in the original

domain is defined as $u^-$ in $\Omega^-$, and solution in the extended domain $\Omega^+$ as $u^+ = 0$. Source term is defined as $f^- \in \Omega^-$ and $f^+ = 0 \in \Omega^+$. Then the immersed boundary problem can be modeled as

$$\triangle u(\vec{x}) = f, \quad \vec{x} \in B \tag{11}$$

with source term being

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

However, this remodeled problem introduces an interface $\Gamma = \partial\Omega^-$, across which discontinuity of solution and source term emerges. As regular central difference approximations for Laplacian will suffer accuracy reduction near a discontinuous interface, corrected difference will be adopted with some interface quantities to compensate the discontinuity. One core part of corrected difference is that it preserves the high order translation-invariant central difference stencil, enabling formation of special matrix structure of the FFT matrix $A$ as above.

Without the loss of generality, discussion can be focused on 1D case, as the generalization to multi-dimension is simply by virtue of tensor product. Hence, we may just concern ourselves with $u = u(x)$. Generally, in the fourth order central difference approximation, corrected differences are only needed at grid nodes, where the central difference stencil involves node information on the other side of the interface. To be more clear, assume the interface intersects the grid line $y = y_j$ at some point $\alpha$ between $x_i$ and $x_{i+1}$, then at nodes $x_{i-1}, x_i, x_{i+1}$ and $x_{i+2}$, corrected differences are required. We call these four nodes as irregular points, while other nodes as regular points.

### 3.2 Corrected Finite Differences for Laplacian

Corrected differences of different orders have been formulated in [8]. As we just focus on fourth order approximation in this paper, fourth order corrected differences are included here.

**Theorem 1** *Fourth order corrected differences.*

Let $x_j \le \alpha < x_{j+1}, h^- = x_j - \alpha$, 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 $\alpha$. Then the following approximations hold to $O(h^4)$ when $M = 5$ :

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

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

where $[u^{(m)}] = [u^{(m)}]_\alpha = \lim_{x\to\alpha^+} u^{(m)}(x) - \lim_{x\to\alpha^-} u^{(m)}(x)$. In practice, $M = 4$ is sufficient to ensure fourth order accuracy for elliptic problem globally despite third order local truncation error $O(h^3)$ [8]. As the derivative jumps $[u^{(m)}]$ are in general not easily determined from PDE and given boundary conditions, we may need numerical approximation. Theoretically, if we can numerically approximate $[u^{(m)}]$ with error of $O(h^{5-m})$, the local third order truncation error can then be guaranteed. On the other hand, due to the fact that high order central differences approximate Laplacian dimension by dimension, we can apply the corrected differences where there exist irregular points.

### 3.2.1 Reconstructing the Cartesian Derivative Jumps

As discussed, the derivative jumps $[u^{(m)}]$ are expected to be numerically approximated with error of $O(h^{5-m})$. Based on the definition of $[u^{(m)}]$, it is natural to use polynomials of certain degree to approximate $\lim_{x\to\alpha^+} u^{(m)}(x)$ or $\lim_{x\to\alpha^-} u^{(m)}(x)$. As the parameter $M$ is set as 4 in the fourth order corrected differences in practical use, the $m$th order derivative $u^{(m)}(x)$ can be up to fourth order. It is sufficient to use polynomials of degree 4 for the approximation with respect to the third order truncation error. Detailed analysis will be shown in the following subsections. For accuracy concern, it is best to use central differences or pseudo-central differences to approximate these one-sided limits. We then may have to generate two layers of fictitious values outside the interface $\Gamma$ as shown in Fig. 5. A systematic MIB boundary closure [32–35] has been developed in the literature for any combination of Dirichlet, Neumann, and Robin boundary conditions. In the present study, this MIB method will be employed to enforce boundary conditions and correspondingly generate the needed fictitious values. Then such fictitious values will be utilized for calculating the derivative jumps $[u^{(m)}]$, as in the AMIB method [7–9]. In our practical numerical approximation, the two fictitious values outside $\Gamma$ will be combined with three real function values on the other side of the interface to give desired polynomials respectively. The right one in Fig. 4 shows two layers of fictitious values needed near the interface $\Gamma$ for the fourth order corrected differences.

### 3.2.2 Fictitious Values Formulation

We adopt the MIB formulation [35] of fictitious values near the staggered boundary condition before the reconstruction of the Cartesian derivative jumps. It is sufficient to illustrate the process in the case of Robin boundary condition in 1D,

$$ku + \frac{\partial u}{\partial n} = ku - \frac{\partial u}{\partial x} = \phi \quad \text{on} \quad \Gamma_1. \tag{12}$$

This procedure can be easily modified for the cases of Dirichlet and Neumann boundary conditions, as Robin boundary condition is essentially comprised of these two conditions. We generate fictitious values iteratively by repeatedly matching the boundary conditions across the boundary. Referring to Fig. 5, to generate $M$ fictitious values $\hat{u}_i$ for $i = 1, 2, \ldots, M$ outside the domain, $L + 1$ function values $u_j$ for $j = 0, 1, \ldots, L$ inside the domain are to be used.

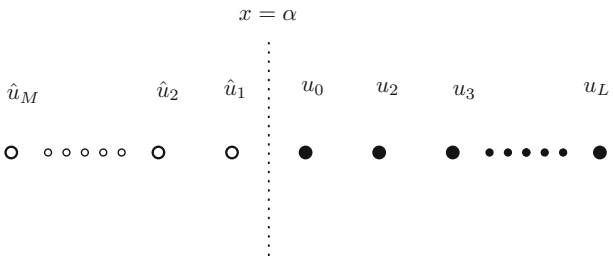


**Fig. 5** An illustration of fictitious values near the staggered left boundary

We can generate the first layer of fictitious value $\hat{u}_1$ according to approximation to boundary (12):

$$k(C_{2,1}^{(0)}\hat{u}_1 + \sum_{i=2}^{L+2} C_{2,i}^{(0)} u_{i-2}) + (C_{2,1}^{(1)}\hat{u}_1 + \sum_{i=2}^{L+2} C_{2,i}^{(1)} u_{i-2}) = \phi, \tag{13}$$

where $C_{2,i}^{(0)}$ and $C_{2,i}^{(1)}$ are finite difference (FD) weights in approximating zeroth and first derivative at $x = a$ by using a stencil $\{\hat{u}_1, u_0, \ldots, u_L\}$. Such weights can be determined by calling a Fortran subroutine *weights* from [10]. The notation 2 in first subscript of $C_{2,i}^{(1)}$ and $C_{2,i}^{(1)}$ is because $u_0$ is the second point in the FD stencils. As a result, unknown $\hat{u}_1$ can be represented as the combination of $u_i$ for $i = 0, 1, \ldots, L$ and $\phi$. Taking $\hat{u}_1$ as known, we can determine another fictitious value $\hat{u}_2$ by one more iteration

$$k(C_{3,1}^{(0)}\hat{u}_2 + C_{3,2}^{(0)}\hat{u}_1 + \sum_{i=3}^{L+3} C_{3,i}^{(0)} u_{i-3}) + (C_{3,1}^{(1)}\hat{u}_2 + C_{3,2}^{(1)}\hat{u}_1 + \sum_{i=3}^{L+3} C_{3,i}^{(1)} u_{i-3}) = \phi, \tag{14}$$

where $C_{3,i}^{(0)}$, $C_{3,i}^{(1)}$ are FD weights to approximate zeroth and first order derivatives at $x = a$ using a stencil $\{\hat{u}_2, \hat{u}_1, u_0, u_1, \ldots, u_L\}$. The first subscript 3 in $C_{3,i}^{(0)}$ and $C_{3,i}^{(1)}$ is because $u_0$ is the third point in the FD stencils. In such iterative approach, we can obtain representation of fictitious values $\hat{u}_3, \hat{u}_4, \ldots, \hat{u}_M$ in terms of $u_i$ for $i = 0, 1, \ldots, L$. We need properly choose the value of $L$ to have high-order accuracy for fictitious values. Generally, $L$ is related to $M$ with $L + 1 = 2M$. As shown below for the fourth order study, two layers of fictitious values are needed for approximation of jump quantities, which means $M$ and $L$ is set to be 2 and 3 respectively.

We here conduct some error analysis about the accuracy of the fictitious values by enforcing Robin boundary conditions, which will be used for further analysis in the next subsection. In the situation of $M = 2$ and $L = 3$, we assume the real function values to be $f_1$ and $f_2$ corresponding to the fictitious values $\hat{u}_1$ and $\hat{u}_2$ at the same locations. Due to Lagrange interpolation, we have following equation:

$$k(C_{2,1}^{(0)} f_1 + \sum_{i=2}^{5} C_{2,i}^{(0)} u_{i-2} + O(h^5)) + (C_{2,1}^{(1)} f_1 + \sum_{i=2}^{5} C_{2,i}^{(1)} u_{i-2} + O(h^4)) = \phi, \tag{15}$$

Subtracting Eq. (13) from Eq. (15) gives

$$k[C_{2,1}^{(0)}(\hat{u}_1 - f_1) + O(h^5)] + [C_{2,1}^{(1)}(\hat{u}_1 - f_1) + O(h^4)] = 0.$$

Equivalently,

$$(kC_{2,1}^{(0)} + C_{2,1}^{(1)})(f_1 - \hat{u}_1) = O(h^5) + O(h^4) = O(h^4).$$

Therefore,

$$\epsilon_1 = f_1 - \hat{u}_1 = \frac{O(h^4)}{kC_{2,1}^{(0)} + C_{2,1}^{(1)}} = \frac{O(h^4)}{O(1) + O(\frac{1}{h})} = O(h^5).$$

The last second equality holds because $C_{2,i}^{(0)}$ and $C_{2,i}^{(1)}$ scale as $O(1)$ and $O(\frac{1}{h})$ respectively. From this, we may notice that the error between real function value $f_1$ and the fictitious value $\hat{u}_1$ is of $O(h^5)$.

In the second step with one more fictitious value, it holds that

$$\begin{aligned} &k(C_{3,1}^{(0)} f_2 + C_{3,2}^{(0)} f_1 + \sum_{i=3}^{5} C_{3,i}^{(0)} u_{i-3} + O(h^6)) \\ &+ (C_{3,1}^{(1)} f_2 + C_{3,2}^{(1)} f_1 + \sum_{i=3}^{5} C_{3,i}^{(1)} u_{i-3} + O(h^5)) = \phi, \end{aligned} \tag{16}$$

Subtracting Eq.(14) from Eq.(16) yields to

$$\begin{aligned} &k[C_{3,1}^{(0)}(\hat{u}_2 - f_2) + C_{3,2}^{(0)}(\hat{u}_1 - f_1) + O(h^6)] \\ &+ [C_{3,1}^{(1)}(\hat{u}_2 - f_2) + C_{3,2}^{(1)}(\hat{u}_1 - f_1) + O(h^5)] = 0. \end{aligned}$$

Since $\hat{u}_1 - f_1 = O(h^5)$, $C_{3,i}^{(0)}$ scales as $O(1)$, and $C_{3,i}^{(1)}$ scales $O(\frac{1}{h})$, we arrive at

$$k[C_{3,1}^{(0)}(\hat{u}_2 - f_2) + O(h^5)] + [C_{3,1}^{(1)}(\hat{u}_2 - f_2) + O(h^4)] = 0.$$

Equivalently,

$$(kC_{3,1}^{(0)} + C_{3,1}^{(1)})(f_2 - \hat{u}_2) = O(h^5) + O(h^4) = O(h^4).$$

Hence,

$$\epsilon_2 = f_2 - \hat{u}_2 = \frac{O(h^4)}{kC_{3,1}^{(0)} + C_{3,1}^{(1)}} = \frac{O(h^4)}{O(1) + O(\frac{1}{h})} = O(h^5).$$

In short, the errors of fictitious values $\hat{u}_1$ and $\hat{u}_2$ are of $O(h^5)$ when $L = 3$.

The MIB implementation of Dirichlet or Neumann boundary condition at staggered boundaries can be similarly formulated. Each implementation may just follow the formula inside the first or second parenthesis in (13) and (14).

In comparing with the present boundary treatments with those for non-staggered boundaries [8], Neumann and Robin cases are essentially the same. However, the Dirichlet case is totally different. In [8], with the boundary being a node, the interpolation or zeroth order derivative approximation will have zero weights for fictitious points. To avoid this problem, a numerical boundary condition is typically used

$$\frac{\partial^2 u}{\partial n^2} = \phi, \tag{17}$$

where $n$ denotes the $x$- or $y$-direction. Such boundary condition was manually derived based on the given Poisson equation and Dirichlet boundary conditions [8]. For example, suppose

Dirichlet condition is given as $u(x, y) = \phi_4(y)$ on $\Gamma_4$. Taking second order derivative on $\phi_4(y)$ with respect to $y$ leads to $\frac{d^2\phi_4(y)}{dy^2}$, which is essentially $\frac{\partial^2 u(x,y)}{\partial y^2}$ along boundary $\Gamma_4$. Assuming that the governing equation $\triangle u = f$ is valid on the boundary $\Gamma_4$, we can derive a new boundary condition as in (17):

$$\frac{\partial^2 u}{\partial x^2}\Big|_{\Gamma_4} = \left[f - \frac{\partial^2 u}{\partial y^2}\right]_{\Gamma_4} = \left[f - \frac{d^2\phi_4(y)}{dy^2}\right]_{\Gamma_4} \triangleq \phi(x). \tag{18}$$

The boundary condition (18) can be iteratively enforced to generate desired fictitious values [8]. The MIB scheme for staggered boundaries avoids using numerical boundary conditions (17). Consequently, the present MIB scheme has some advantages over that in [8]. For instance, it is still valid even if the solution loses some high order regularity on the boundary. We will demonstrate this in numerical experiments.

In summary, the fictitious values $\hat{u}_{i,j}$ at $(x_i, y_j)$ calculated from Dirichlet, Neumann, and Robin boundary conditions can be written into a general form:

$$\hat{u}_{i,j} = \sum_{(x_I, y_J)\in\mathbb{S}_{i,j}} W_{I,J} u_{I,J} + W_0\phi, \tag{19}$$

which is a linear combination of functions values on a chosen set $\mathbb{S}_{i,j}$, and known boundary data. The node set $\mathbb{S}_{i,j}$ can be determined by the choice of $M$ and $L$ before calculating the MIB weights $W_{I,J}$.

### 3.2.3 Approximation to Derivative Jumps

To approximate jump quantities in the fourth order corrected differences, we consider using polynomials of degree 4 from inside of the interface $\Gamma$ together with the external function $u^+ = 0$ to calculate limiting derivatives. The jump quantity at $x = \alpha$ is defined as

$$\left[\frac{\partial^m u}{\partial x^m}\right]|_{x=\alpha} = \lim_{x\to\alpha^+} \frac{\partial^m u}{\partial x^m} - \lim_{x\to\alpha^-} \frac{\partial^m u}{\partial x^m} \tag{20}$$

Figure 6 graphically gives an explanation of fourth order numerical approximation to jumps with aid of polynomial near the interface. Essentially, The derivative jumps are calculated by taking derivatives of one-sided polynomials at intersecting point $(\alpha, y_j)$. Here certain sign is to be determined for the left or right end [8].

We take the case of immersed boundary located to the left boundary for instance. The jump quantities are approximated by (refer to Fig. 6)

$$\begin{aligned}
\left[\frac{\partial^m u}{\partial x^m}\right] &= \frac{\partial^m u^-}{\partial x^m} - 0 \\
&= w^m_{i-1,j} f_{i-1,j} + w^m_{i,j} f_{i,j} + \sum_{k=1}^{3} w^m_{i+k,j} u_{i+k,j} + O(h^{5-m}) \\
&= w^m_{i-1,j}(\hat{u}_{i-1,j} + O(h^5)) + w^m_{i,j}(\hat{u}_{i,j} + O(h^5)) + \sum_{k=1}^{3} w^m_{i+k,j} u_{i+k,j} + O(h^{5-m}) \\
&= w^m_{i-1,j}\hat{u}_{i-1,j} + O(h^{5-m}) + w^m_{i,j}\hat{u}_{i,j} + \sum_{k=1}^{3} w^m_{i+k,j} u_{i+k,j} + O(h^{5-m})
\end{aligned}$$

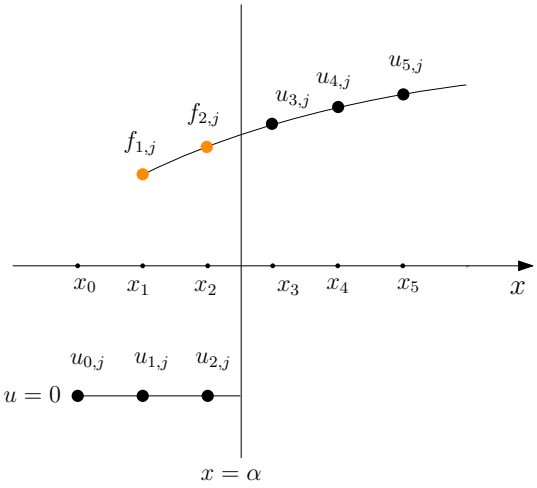


**Fig. 6** An illustration of jumps approximation via a Lagrange polynomial from inside of the interface $x = \alpha$ and zeros solution from the outside for the fourth order approximation. The black dots stand for the real values while the yellow dots denote the fictitious values

$$= w^m_{i-1,j}\hat{u}_{i-1,j} + w^m_{i,j}\hat{u}_{i,j} + \sum_{k=1}^{3} w^m_{i+k,j}u_{i+k,j} + O(h^{5-m}) \tag{21}$$

where $u^- \in \Omega^-$, and $w^m_{p,q}$ stands for the weights in finite difference approximation at $x = \alpha$ and scales as $O(h^{-m})$. This leads to the approximation error of $O(h^{5-m})$ in the last second equality. The approximation in the first equality is due to the Lagrange interpolation error. In $w^m_{p,q}$, subscript $(p, q)$ denotes the location $(x_p, y_q)$, and $m = 0, 1, \ldots, 4$ indicates the $m$th derivative. Consequently, the local truncation error in the fourth order corrected differences will keep third order of $O(h^3)$.

Dropping the error terms and plugging fictitious values representation (19) into (21), we have an equivalent form:

$$\sum_{(x_I, y_J)\in\mathbb{S}_{i,j}} C_{I,J}u_{I,J} + [\frac{\partial^m u}{\partial x^m}] = C_0\phi, \tag{22}$$

where $C_{I,J}$ are the corresponding weights of function value $u_{i,j}$ in approximation to jump quantity $[\frac{\partial^m u}{\partial x^m}]$. Taking $[\frac{\partial^m u}{\partial x^m}], m = 0, 1, \ldots, 4$ as auxiliary variables, system of equation (22) can be written in a matrix-vector form:

$$CU + IQ = \Phi, \tag{23}$$

where $C$ stands for the weights coefficients matrix, $U$ represents the function variables, $I \in \mathbb{R}^{5,5}$ is the identity matrix, $Q$ is the introduced auxiliary variables on all boundary nodes, and $\Phi$ is from the known boundary quantities $\phi$. Without loss of generality, the form of (23) can be applied for derivative jump approximation on the other side of the domain or in the $y-$direction.

## 4 Augmented System

### 4.1 Formulation of the Augmented System

For fourth order study on 2D Poisson's problem (11), let $U_{i,j}$ indicate the discrete solution while $u(x_i, y_j)$ is analytical solution at $(x_i, y_j)$. Based on the corrected difference analysis, the problem (7) is discretized as

$$L_h U_{i,j} + C_{i,j} = f_{i,j}, \quad 1 \le i \le M+4, \quad 1 \le j \le N+4, \tag{24}$$

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 $N1 = (M+4) \times (N+4)$. Note that correction term only exists at irregular points but vanishes at regular points. Following the introduced variables $Q$ in Eq.(23), Eq.(24) can be rewritten into

$$AU + BQ = F, \tag{25}$$

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 $f_{ij}$. The auxiliary variables involve terms as $[u]_i, [\frac{\partial u}{\partial x}]_i, [\frac{\partial^2 u}{\partial x^2}]_i, [\frac{\partial^3 u}{\partial x^3}]_i, [\frac{\partial^4 u}{\partial x^4}]_i$ for $i = 1, 2, \ldots$, 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 $j = 1, 2, \ldots$, 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. So $N2$ 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.

On the other hand, inspired by formula (23), approximation to all auxiliary variables leads to

$$CU + IQ = \Phi. \tag{26}$$

Matrix $C$ is a sparse matrix of dimension $N2 \times N1$, and $I$ is a $N2$ by $N2$ identity matrix. An augmented matrix is therefore obtained by combining (25) and (26),

$$KW = R, \tag{27}$$

where

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

### 4.2 Schur Complement

As we discussed in the beginning, the matrix form of $A$ enables FFT solution. Eliminating $U$ in (27) gives a Schur complement system for $Q$,

$$(I - CA^{-1}B)Q = \Phi - CA^{-1}F, \tag{28}$$

with variable $Q$ solved by using biconjugate gradient method combined with FFT. The reason for obtained $Q$ first is that system (28) has a smaller dimension, and FFT can accelerate the overall calculation. Finally $U$ is solved by applying FFT one more time for the system

$$AU = (F - BQ). \tag{29}$$

The details of implementation on Schur complement system (28) in biconjugate gradient subroutines are formulated as below:

1. One determines right side $\hat{F} = \Phi - CA^{-1}F$ with one FFT for $A^{-1}F$ and some 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 $IQ - CA^{-1}BQ$, a FFT is carried out on the product of $BQ$, that's to say, $A^{-1}(BQ)$. Then it is followed with simple multiplication and summation for $IQ - CA^{-1}(BQ)$. The transpose of $(I - CA^{-1}B)^T Q$ is done on $IQ - B^T A^{-1} C^T Q$ in a similar fashion.
3. A initial guess $Q = (0, 0, \ldots, 0)^T$ starts the biconjugate gradient iteration, and either the maximal iteration number 5000 or error tolerance equal to $10^{-15}$ terminates the iteration. The termination criteria are adjustable according to actual computational needs.

## 5 Numerical Experiments

In this section, we examine the performance of the proposed AMIB method in solving two or three dimensional Poisson's equation with an arbitrary combination of Dirichlet, Neumann, and Robin boundary conditions on staggered boundaries. The numerical performance of AMIB4, AMIB6, and AMIB8 methods will be tested, with the number after AMIB standing for the designed order of accuracy. Moreover, we will compare the present AMIB with the one in [8] in some examples. To denote the difference, we use S to stand for the study with staggered boundary conditions. For example, AMIB4-S indicates the fourth order AMIB method on staggered boundary conditions, while AMIB4-NS means the non-staggered case.

For simplicity, a square or cubic domain with uniform number of grids in each direction is studied, and we follow the boundary notations in Fig. 2. The numerical accuracy and convergence will be tested by comparing the numerical solutions and exact solutions under the standard maximum norm and $L_2$ norm defined as

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

and

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

where $u_h(x_i, y_j)$, $u(x_i, y_j)$ are numerical and analytical solution, respectively. The set $S_{\Omega^-}$ 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 $S_{\Omega^-}$. In each table below, $N_x$, $N_y$ and $N_z$ stand for grid numbers on the domain $\Omega^-$ in each direction.

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

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

where $||E_i||$ is the 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 [19] 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 the experiments were carried out on MacBook Pro with 8.00 RAM and Intel 2.3 GHz Intel Core i5.

### 5.1 The AMIB Method for Staggered Boundaries

In this subsection, we examine the AMIB method for Poisson problems in 2D and 3D with Dirichlet conditions.

***Example 1*** This example focuses on validating the ultra high orders of our new schemes. 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

$$u_{xx} + u_{yy} = -2k^2 \sin(kx)\cos(ky) \tag{30}$$

with Dirichlet boundary condition determined by analytical solution $u = \sin(kx)\cos(ky)$, in which the wave number $k$ could be chosen as a large number. Basically, only for highly oscillating solutions, ultra high order of convergence can be detected numerically. In this example, AMIB4-S, AMIB6-S, AMIB8-S are examined under above Dirichlet boundary conditions with the parameter $k$ set to be 5, 10 and 20, respectively.

The high order convergence of the AMIB-S methods is validated in Table 1. We note that because different wave numbers are used, the error comparison among different orders does not make much sense. In particular, the errors on the coarsest mesh for the AMIB8-S are much larger than those of AMIB6-S, simply because the numerical problem becomes much challenging for $k = 20$. It can be seen that the fourth and sixth orders are achieved. The eighth order accuracy of AMIB8-S is not reached first, but is attained eventually when the mesh resolution is fine enough.

As the computing time is related to the iteration number for solving the auxiliary variables in Schur complement, we report the iteration number in the second last column in Table 1. 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, we may say that the iteration number weakly depends the mesh size in this case, and the AMIB-S method is very efficient.

***Example 2*** 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 performance of AMIB4-S is tested by the following 3D Poisson problem:

$$u_{xx} + u_{yy} + u_{zz} = \sin(kx)e^{y+z}(2 - k^2) \tag{31}$$

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 this 3D test. The fourth order convergence of the AMIB4-S for 3D Poisson problem is numerically verified through Table 2. Moreover, the iteration number increases very little when the mesh size is doubled in each dimension.

### 5.2 Comparison of AMIB-S and AMIB-NS

Because only a single variable Poisson's equation and a uniform mesh are considered in this paper, the users actually have the freedom to partition the domain into either staggered or non-staggered boundaries. This means for many problems studied in this work, both AMIB-S and AMIB-NS methods can be applied. It is thus interested to compare the performance of two methods for solving the same problem. In particular, two special examples are designed to demonstrate the advantages of AMIB-S over AMIB-NS.

**Table 1** Example 1—high order numerical error analysis

| $[N_x, N_y]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4-S and $k = 5$ | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [27, 27] | 2.917E−4 | – | 1.131E−4 | – | 11 | 4.912E−3 |
| [59, 59] | 1.133E−5 | 4.16 | 4.655E−6 | 4.08 | 11 | 1.168E−2 |
| [123, 123] | 5.956E−7 | 4.01 | 2.430E−7 | 4.02 | 10 | 3.261E−2 |
| [251, 251] | 3.563E−8 | 3.95 | 1.431E−8 | 3.97 | 10 | 0.125 |
| [507, 507] | 2.187E−9 | 3.97 | 8.795E−10 | 3.97 | 10 | 0.468 |
| $[N_x, N_y]$ | AMIB6-S and $k = 10$ | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [25, 25] | 9.183E−4 | – | 3.241E−4 | – | 30 | 1.135E−2 |
| [57, 57] | 9.968E−6 | 5.48 | 2.885E−6 | 5.73 | 38 | 3.671E−2 |
| [121, 121] | 1.160E−7 | 5.92 | 3.479E−8 | 5.87 | 44 | 1.121 |
| [249, 249] | 1.762E−9 | 5.80 | 4.823E−10 | 5.93 | 48 | 0.447 |
| [505, 505] | 2.920E−11 | 5.80 | 7.158E−12 | 5.95 | 52 | 2.029 |
| $[N_x, N_y]$ | AMIB8-S and $k = 20$ | | | | | |
| | $L_\infty$ | | $L_2$ | | | |
| [23, 23] | 4.932E−2 | – | 1.343E−2 | – | 35 | 1.451E−2 |
| [55, 55] | 1.439E−4 | 6.70 | 2.625E−5 | 7.15 | 46 | 5.823E−2 |
| [119, 119] | 4.458E−7 | 7.49 | 6.451E−8 | 7.79 | 53 | 0.177 |
| [247, 247] | 1.830E−9 | 7.53 | 2.013E−10 | 7.90 | 60 | 0.603 |
| [503, 503] | 6.625E−12 | 7.90 | 7.006E−13 | 7.96 | 63 | 2.482 |

**Table 2** Example 2—fourth order numerical error analysis of the AMIB4-S on 3D Poisson's equation

| $[N_x, N_y, N_z]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4-S | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [11, 11, 11] | 2.291E−2 | – | 5.127E−3 | – | 31 | 4.098E−2 |
| [27, 27, 27] | 6.986E−4 | 3.89 | 1.780E−4 | 3.74 | 39 | 0.301 |
| [59, 59, 59] | 3.095E−5 | 3.99 | 7.852E−6 | 3.99 | 42 | 2.689 |
| [123, 123, 123] | 1.639E−6 | 4.00 | 4.161E−7 | 4.00 | 44 | 24.28 |
| [251, 251, 251] | 9.445E−8 | 4.00 | 2.400E−8 | 4.00 | 45 | 246.57 |

***Example 3*** This example shows some advantage of AMIB4-S over AMIB4-NS on Poisson problem with analytical solution:

$$u = \sin(kx)\left(y + \frac{\pi}{3}\right)^{\frac{5}{2}} \tag{32}$$

on domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$ with $k = 5$. On four sides, Dirichlet boundary conditions are given by the analytical solution. Note that the analytical solution has good smoothness

**Table 3** Example 3—fourth order numerical error analysis of the AMIB4-S and AMIB4-NS

| $[N_x, N_y]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4-S | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [59, 59] | 3.739E−5 | – | 1.270E−5 | – | 42 | 3.556E−2 |
| [123, 123] | 1.977E−6 | 4.00 | 6.737E−7 | 4.00 | 41 | 9.587E−2 |
| [251, 251] | 1.139E−7 | 4.00 | 3.946E−8 | 3.97 | 43 | 0.371 |
| [507, 507] | 6.847E−9 | 4.00 | 2.636E−9 | 3.85 | 44 | 1.690 |
| $[N_x, N_y]$ | AMIB4-NS | | | | | |
| | $L_\infty$ | | $L_2$ | | | |
| [59, 59] | 4.107E−5 | – | 1.341E−5 | – | 20 | 1.780E−2 |
| [123, 123] | 3.426E−6 | 3.34 | 8.878E−7 | 3.65 | 20 | 6.472E−2 |
| [251, 251] | 6.210E−7 | 2.38 | 1.062E−7 | 2.96 | 20 | 0.189 |
| [507, 507] | 1.116E−7 | 2.43 | 1.746E−8 | 2.56 | 21 | 0.845 |

except at $y = -\frac{\pi}{3}$, on which the partial derivatives with respect to $y$ are defined only up to second order.

Recall that the implementations of AMIB-S and AMIB-NS are different for Dirichlet conditions. For AMIB4-S, the boundary enforcement is basically through interpolation. Nevertheless, for AMIB4-NS, numerical boundary conditions like (17) need to be constructed, which essentially involve approximations of Laplacian along $y = -\frac{\pi}{3}$. For the fourth order central difference approximations, this requires at least sixth order regularity, which is obviously not true for the present example. Hence, order reduction is inevitable for the AMIB4-NS scheme. On the other hand, since the boundary $y = -\frac{\pi}{3}$ is staggered with respect to the grid, the approximation of Laplacian is simply avoided at this low-regularity boundary in the AMIB4-S discretization. Consequently, the order of AMIB4-S is not affected by the low regularity solution.

In the Table 3, we can observe that AMIB4-S is superior to AMIB4-NS in preserving fourth order accuracy on the numerical results.

***Example 4*** In this example, we consider a problem in which the PDE is not satisfied at the Dirichlet boundaries. Such a problem is frequently encountered in practice, due to some randomness or perturbation in experimental measurements. On domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$, the Poisson problem is given as

$$u_{xx} + u_{yy} = -2k^2 \sin(kx)\cos(ky) \tag{33}$$

subject to a perturbed Dirichlet boundary condition defined by the function $v(x, y) = \sin(k_0 x)\cos(k_0 y)$. A small perturbation exists in parameters with $k = k_0 + \epsilon$. Here we set $k_0 = 10$ and $\epsilon = 0.01$.

The AMIB4-NS scheme is not applicable to this problem, because the PDE solution will not satisfy the Dirichlet condition. Hence, the numerical boundary condition (17) based on the PDE and Dirichlet condition is simply erroneous. This leaves the AMIB4-S scheme a natural choice for the present problem, which can still maintain the fourth order convergence. As there is no analytical solution for this problem, we generate reference solutions on sufficiently dense mesh to carry out numerical analysis, which means we have to compare the solutions

**Table 4** Example 4—fourth order numerical error analysis of the AMIB4-S

| $[N_x, N_y]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4-S | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [10, 10] | 1.486E−1 | – | 7.269E−2 | – | 16 | 3.812E−3 |
| [20, 20] | 1.129E−2 | 3.72 | 5.447E−3 | 3.74 | 26 | 4.205E−3 |
| [40, 40] | 8.478E−3 | 3.74 | 3.775E−4 | 3.99 | 31 | 1.404E−2 |
| [80, 80] | 5.703E−4 | 3.89 | 2.447E−5 | 4.00 | 36 | 4.107E−2 |
| [160, 160] | 3.617E−5 | 3.98 | 1.546E−6 | 4.00 | 39 | 0.238 |

on coarse meshes to those on fine meshes. Suppose that there are $N$ grids on the coarse mesh of the given domain and $M$ grids on the fine mesh in each direction. Here $M$ has to be $(1 - 2k_2)$ times of $N$ with $k_2$ being negative integers such the each grid on coarse mesh can agree with the grid on fine mesh at the same location. The index $i$ on the coarse mesh should correspond to $j$ on fine mesh with the relation $j = (1 - 2k_2)i + k_2$. In this way, we can use the solutions on sufficiently fine mesh as reference solutions for numerical analysis. In our numerical experiments, we choose fine meshes with $k_2$ as $-39$, $-19$, $-9$, $-4$ and $-2$ for grid number $N_x$ equal to 10, 20, 40, 80, and 160 on the coarse meshes. In this example, the FFT subroutine used is from FFTPACK.

The numerical results of the AMIB4-S scheme are reported in Table 4. It can be seen that the AMIB4-S scheme is accurate and efficient for problems without analytical solutions.

***Example 5*** This example compares the results of AMIB4-S and AMIB4-NS for solving Poisson's equation

$$u_{xx} + u_{yy} = e^{-x^2 - \frac{y^2}{2}}(4x^2 + y^2 - 3) \tag{34}$$

subject to Robin boundary conditions:

$$\begin{aligned}
u + \frac{\partial u}{\partial n} &= \left(1 - \frac{\pi}{3}\right) e^{-x^2 - \frac{\pi^2}{18}} \quad \text{on} \quad \Gamma_1, \\
u + \frac{\partial u}{\partial n} &= \left(1 + \frac{2\pi}{3}\right) e^{-\frac{\pi^2}{9} - \frac{y^2}{2}} \quad \text{on} \quad \Gamma_2, \\
u + \frac{\partial u}{\partial n} &= \left(1 + \frac{\pi}{3}\right) e^{-x^2 - \frac{\pi^2}{18}} \quad \text{on} \quad \Gamma_3, \\
u + \frac{\partial u}{\partial n} &= \left(1 - \frac{2\pi}{3}\right) e^{-\frac{\pi^2}{9} - \frac{y^2}{2}} \quad \text{on} \quad \Gamma_4,
\end{aligned}$$

which are determined by analytical solution $u = e^{-x^2 - \frac{y^2}{2}}$ on domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$. Table 5 shows that the both approaches give good accuracy in solving the Poisson equation with Robin boundary conditions, but the iteration number of AMIB4-S is higher than that of AMIB4-NS.

***Example 6*** Consider Poisson's equation

$$u_{xx} + u_{yy} = -2e^x \sin(x + y) - e^x \cos(x + y) \tag{35}$$

**Table 5** Example 5—fourth order numerical error analysis

| $[N_x, N_y]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4-S | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [11, 11] | 4.182E−4 | – | 3.017E−4 | – | 15 | 5.820E−3 |
| [27, 27] | 3.893E−6 | 5.21 | 2.161E−6 | 5.50 | 20 | 6.565E−3 |
| [59, 59] | 2.839E−7 | 3.35 | 1.177E−7 | 3.72 | 26 | 2.135E−2 |
| [123, 123] | 1.744E−8 | 3.80 | 8.023E−9 | 3.66 | 34 | 8.622E−2 |
| [251, 251] | 1.071E−9 | 3.92 | 5.162E−10 | 3.85 | 45 | 0.389 |
| $[N_x, N_y]$ | AMIB4-NS | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [11, 11] | 6.064E−4 | – | 3.669E−4 | – | 16 | 2.919E−3 |
| [27, 27] | 8.515E−6 | 4.46 | 4.075E−6 | 4.71 | 21 | 6.771E−3 |
| [59, 59] | 2.791E−7 | 4.26 | 1.322E−7 | 4.27 | 22 | 1.952E−2 |
| [123, 123] | 1.291E−8 | 4.13 | 6.417E−9 | 4.07 | 26 | 6.725E−2 |
| [251, 251] | 6.967E−10 | 4.07 | 3.594E−10 | 4.02 | 33 | 0.302 |

on domain $[-\frac{\pi}{3}, \frac{\pi}{3}] \times [-\frac{\pi}{3}, \frac{\pi}{3}]$, subject to mixed boundary conditions determined by the exact solution $u = e^x \cos(x + y)$.

$$
\begin{aligned}
u &= e^x \cos\left(x + \frac{\pi}{3}\right) && \text{on } \Gamma_1,\\
u + \frac{\partial u}{\partial n} &= e^{\frac{\pi}{3}} \left(2\cos\left(\frac{\pi}{3} + y\right) - \sin\left(\frac{\pi}{3} + y\right)\right) && \text{on } \Gamma_2,\\
u + \frac{\partial u}{\partial n} &= e^x \left(\cos\left(x + \frac{\pi}{3}\right) - \sin\left(x + \frac{\pi}{3}\right)\right) && \text{on } \Gamma_3,\\
u_x &= e^{-\frac{\pi}{3}} \left(\cos\left(-\frac{\pi}{3} + y\right) - \sin\left(-\frac{\pi}{3} + y\right)\right) && \text{on } \Gamma_4.
\end{aligned}
$$

Table 6 indicates that the AMIB4-S produces a fourth order of convergence. However, the iterative number increases dramatically as $N_x = N_y$ increases. So does the CPU time. Consequently, the AMIB4-S scheme becomes inefficient comparing with the AMIB4-NS scheme. For the AMIB4-NS scheme, the overall complexity is still on the order of $O(N \log N)$ for this 2D problem with $N = N_x \times N_y$. In general, it is found that the AMIB4-S scheme fails to reach the desired efficiency for problems with mixed boundary conditions as in this example. Therefore, whenever the regular AMIB4-NS scheme is applicable, we recommend to use the regular AMIB method for both accuracy and efficiency.

### 5.3 Combined AMIB Method

***Example 7*** In this example, we combine AMIB4-S and AMIB4-NS schemes to attack one type of problems for which high order fast Poisson solvers have never been reported. Practical applications sometimes lead to the consideration of a staggered boundary on one end but a non-staggered boundary on the other end. Since in the AMIB method, the boundary treatments at left and right ends are independent, the AMIB4-S and AMIB4-NS schemes can be simply

**Table 6** Example 6—fourth order numerical error analysis of AMIB4-S for mixed boundary conditions

| $[N_x, N_y]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4-S | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [11, 11] | 1.245E−4 | – | 5.487E−5 | – | 3.980E−3 | 31 |
| [27, 27] | 5.075E−6 | 3.56 | 1.595E−6 | 3.94 | 1.209E−2 | 48 |
| [59, 59] | 2.542E−7 | 3.83 | 7.063E−8 | 3.99 | 4.603E−2 | 69 |
| [123, 123] | 1.414E−8 | 3.93 | 3.727E−9 | 4.00 | 0.233 | 108 |
| [251, 251] | 8.333E−10 | 3.97 | 2.147E−10 | 4.00 | 1.323 | 157 |
| [507, 507] | 5.402E−11 | 3.89 | 1.716E−11 | 3.59 | 8.193 | 224 |
| $[N_x, N_y]$ | AMIB4-NS | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [11, 11] | 4.883E−4 | – | 1.469E−4 | – | 3.946E−3 | 28 |
| [27, 27] | 1.480E−5 | 3.66 | 5.404E−6 | 3.46 | 8.879E−3 | 31 |
| [59, 59] | 6.514E−7 | 3.89 | 2.586E−7 | 3.79 | 2.810E−2 | 38 |
| [123, 123] | 3.457E−8 | 3.95 | 1.419E−8 | 3.90 | 0.120 | 50 |
| [251, 251] | 1.987E−9 | 3.98 | 8.276E−10 | 3.96 | 0.578 | 67 |
| [507, 507] | 9.724E−11 | 4.28 | 3.177E−11 | 4.62 | 3.522 | 96 |

employed at either end. The combined AMIB method is flexible and robust in handling any combination of grid setting.

We test our method on the following problem. On domain $[-\frac{\pi}{3}, \frac{\pi}{3}]\times[-\frac{\pi}{3}, \frac{\pi}{3}]$, the Poisson problem has

$$u_{xx} + u_{yy} = \cos(ky)e^y(-k^2 + 1) \tag{36}$$

with analytical solution $u = \cos(kx)e^y$. Suppose it is subject to boundary conditions and grid setting as below:

$$\begin{aligned} u_y &= \cos(kx)e^{-\frac{\pi}{3}} \quad \text{on} \quad \Gamma_1, \\ u &= \cos\left(\frac{k\pi}{3}\right)e^y \quad \text{on} \quad \Gamma_2, \\ u &= \cos(kx)e^{\frac{\pi}{3}} \quad \text{on} \quad \Gamma_3, \\ u_x &= \sin\left(\frac{k\pi}{3}\right)e^y \quad \text{on} \quad \Gamma_4. \end{aligned}$$

In addition, we have staggered boundary condition imposed on the boundaries $\Gamma_1$ and $\Gamma_4$. The results in Table 7 indicate that the combined AMIB method achieves a fourth order convergence, while being efficient.

## 6 Summary

Due to the popularity of staggered grids in scientific computing for solving PDEs with multiple dependent variables, the development of fast Poisson solvers with staggered boundaries

**Table 7** Example 7—fourth order numerical error analysis of the AMIB4

| $[N_x, N_y]$ | Error | Order | Error | Order | | |
|---|---|---|---|---|---|---|
| | AMIB4 | | | | | |
| | $L_\infty$ | | $L_2$ | | iter no. | CPU time |
| [27, 27] | 5.680E−3 | – | 2.427E−3 | – | 30 | 9.736E−3 |
| [59, 59] | 2.356E−4 | 4.07 | 9.742E−5 | 4.64 | 39 | 2.959E−2 |
| [123, 123] | 1.194E−5 | 4.06 | 4.863E−6 | 4.08 | 51 | 0.123 |
| [251, 251] | 6.719E−7 | 4.03 | 2.715E−7 | 4.04 | 69 | 0.572 |
| [507, 507] | 3.984E−8 | 4.01 | 1.604E−8 | 4.02 | 101 | 3.964 |

deserves research attention. However, grand challenges exist for spectral methods and FFT algorithms in handling staggered boundaries. The only known high order fast Poisson solver [1] combines fourth order compact finite difference with multigrid for staggered Neumann conditions. There are many unsolved open problems, such as Robin conditions, problems with both staggered and non-staggered boundaries on both ends, etc.

Based on the AMIB method developed recently in [8], which provides high order fast Poisson solvers for non-staggered boundaries, a systematic approach is introduced in this paper to attack the aforementioned open problems with staggered boundaries. Different from traditional direct FFT solution using Fourier analysis and synthesis, our AMIB methods avoid the boundary treatment difficulty by transforming the original problem into immersed boundary problems, so that the FFT inversion could be utilized. In treating Dirichlet and Robin conditions, the AMIB method is efficient enough for solving Poisson problem with staggered boundaries, for which fourth, sixth, eighth order accuracy have been observed. As it uses high order central differences, multidimensional problems can be naturally solved by means of tensor product.

Comparison between AMIB methods based on staggered and non-staggered boundary settings has been conducted, because this paper only solves a single variable Poisson's equation. Due to the difficulties associated with the numerical boundary condition in the Dirichlet implementation, the non-staggered AMIB scheme is not applicable to problems in which the PDE is not satisfied at the boundaries, while the staggered AMIB scheme works perfectly for such problems. Moreover, for problems with low regularity solutions on the boundaries, the non-staggered AMIB scheme experiences order reduction, while the staggered AMIB scheme keeps the high order convergence. However, the efficiency of the proposed method needs to be improved for problems with mixed types of boundary conditions. For such problems, the non-staggered AMIB method [8] performs robustly and should be employed. In the future, we plan to investigate preconditioning strategies so that the iteration number of the new method could be reduced to speed up the AMIB computation.

**Acknowledgements** This research is partially supported by the National Science Foundation (NSF) Grant DMS-1812930, the Natural Science Foundation of China under Grant 11461011, and the key project of Guangxi Provincial Natural Science Foundation of China under Grants AD20238065, 2017GXNSFDA198014 and 2018GXNSFDA050014.

## References


1. Abide, S., Zeghmati, B.: Multigrid defect correction and fourth-order compact scheme for Poisson's equation. Comput. Math. Appl. **73**, 1433–1444 (2017)
2. Averbuch, A., Israeli, M., Vozovoi, L.: A fast Poisson solver of arbitrary order accuracy in rectangular regions. SIAM J. Sci. Comput. **19**, 933–952 (1998)
3. Boisvert, R.F.: A fourth order accurate Fourier method for the Helmholtz equation in three dimensions. ACM Trans. Math. Softw. (TOMS) **13**, 221–234 (1987)
4. Braverman, E., Israeli, M., Averbuch, A., Vozovoi, L.: A fast 3D Poisson solver of arbitrary order accuracy. J. Comput. Phys. **144**, 109–136 (1998)
5. Braverman, E., Israeli, M., Averbuch, A.: A fast spectral solver for a 3D Helmholtz equation. SIAM J. Sci. Comput. **20**, 2237–2260 (1999)
6. Bruger, A., Nilsson, J., Kress, W.: A compact higher order finite difference method for the incompressible Navier–Stokes equations. J. Sci. Comput. **17**, 551–560 (2002)
7. Feng, H., Long, G., Zhao, S.: An augmented matched interface and boundary (MIB) method for solving elliptic interface problem. J. Comput. Appl. Math. **361**, 426–433 (2019)
8. Feng, H., Zhao, S.: FFT-based high order central difference schemes for the three-dimensional Poisson equation with various types of boundary conditions. J. Comput. Phys. **410**, 109391 (2020)
9. Feng, H., Zhao, S.: A fourth order finite difference method for solving elliptic interface problems with the FFT acceleration. J. Comput. Phys. **419**, 109677 (2020)
10. Fornberg, B.: Calculation of weights in finite difference formulas. SIAM Rev. **40**, 685–691 (1998)
11. Golub, G.H., Huang, L.C., Simon, H., Tang, W.: A fast Poisson solver for the finite difference solution of the incompressible Navier–Stokes equations. SIAM J. Comput. **19**, 1606–1624 (1998)
12. Gupta, M.M., Kouatchou, J., Zhang, J.: Comparison of second and fourth order discretization multigrid Poisson solvers. J. Comput. Phys. **132**, 226–232 (1997)
13. Ge, Y.: Multigrid method and fourth-order compact difference discretization scheme with unequal mesh-sizes for 3D Poisson equation. J. Comput. Phys. **229**, 6381–6391 (2010)
14. Haidvoge, D., Zang, T.: The accurate solution of Poisson's equation by expansion in Chebyshev polynomials. J. Comput. Phys. **30**, 167–180 (1979)
15. Kampanis, N.A., Ekaternaris, J.A.: A staggered grid, high-order accurate method for the incompressible Navier–Stokes equations. J. Comput. Phys. **215**, 589–613 (2006)
16. Lai, M.-C.: A simple compact fourth-order Poisson solver on polar geometry. J. Comput. Phys. **182**, 337–345 (2002)
17. Ma, Z.H., Qian, L., Causon, D.M., Gu, H.B., Mingham, C.G.: A cartesian ghost-cell multigrid poisson solver for incompressible flows. Int. J. Numer. Meth. Eng. **85**, 230–246 (2011)
18. Nagel, J.R.: Solving the Generalized Poisson's Equation Using the Finite-Difference Method (FDM). University of Utah, Salt Lake City, Department of Electrical and Computer Engineering (2011)
19. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in Fortran: The Art of Scientific Computing, 2nd edn. Cambridge University Press, Cambridge (1992)
20. Schumann, U., Sweet, R.: A direct method for the solution of Poisson's equation with neumann boundary conditions on a staggered grid of arbitrary size. J. Comput. Phys. **20**, 171–182 (1976)
21. Schumann, U., Sweet, R.: Fast Fourier transforms for direct solution of Poisson's equation with staggered boundary conditions. J. Comput. Phys. **75**, 123–137 (1988)
22. Shen, J., Tang, T., Wang, L.L.: Spectral methods: Algorithm, Analysis and Application. Springer Series in Computational Mathematics. Springer, Berlin (2011)
23. Sun, X.H., Zhuang, Y.: A high-order direct solver for helmholtz equations with neumann boundary conditions. Technical Report. Institute for Computer Applications in Science and Engineering (ICASE) (1997)
24. Swarztrauber, P.N.: Symmetric FFTs. Math. Comput. **47**, 323–346 (1986)
25. Swarztrauber, P., Sweet, R.: Algorithm 541: efficient Fortran subprograms for the solution of separable elliptic partial differential equations. ACM Trans. Math. Softw. (TOMS) **5**, 352–364 (1979)
26. Trottenberg, U., Oosterlee, C.W.: Multigrid. Academic Press, Cambridge (2001)
27. Wang, H., Zhang, Y., Ma, X., Qiu, J., Liang, Y.: An efficient implementation of fourth-order compact finite difference scheme for Poisson's equation with Dirichlet boundary conditions. Comput. Math. Appl. **71**, 1843–1860 (2016)
28. Wang, Y., Zhang, J.: Sixth order compact scheme combined with multigrid method and extrapolation technique for 2D Poisson equation. J. Comput. Phys. **228**, 137–146 (2009)
29. Wesseling, P.: An Introduction to Multigrid Methods. Pure and Applied Mathematics. Wiley, Hoboken (1992)

30. Zhang, K.K.O., Shotorban, B., Minkowycz, W.J., Mashayek, A.: A compact finite difference method on staggered grid for Navier–Stokes flow. Int. J. Numer. Methods Fluids **52**, 867–881 (2006)
31. Zhao, S., Wei, G.W.: High-order FDTD methods via derivative matching for Maxwell's equations with material interfaces. J. Comput. Phys. **200**, 60–103 (2004)
32. Zhao, S., Wei, G.W., Xiang, Y.: DSC analysis of free-edged beams by an iteratively matched boundary method. J. Sound Vib. **284**, 487–493 (2005)
33. Zhao, S.: On the spurious solutions in the high-order finite difference methods. Comput. Methods Appl. Mech. Eng. **196**, 5031–5046 (2007)
34. Zhao, S.: A fourth order finite difference method for waveguides with curved perfectly conducting boundaries. Comput. Methods Appl. Mech. Eng. **199**, 2655–2662 (2010)
35. Zhao, S., Wei, G.W.: Matched interface and boundary (MIB) for the implementation of boundary conditions in high order central finite differences. Int. J. Numer. Methods Eng. **77**, 1690–1730 (2009)
36. Zhou, Y.C., Zhao, S., Feig, M., Wei, G.W.: High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular source. J. Comput. Phys. **213**, 1–30 (2006)
37. Zhuang, Y., Sun, X.: A high-order fast direct solver for singular Poisson equations. J. Comput. Phys. **20**, 79–94 (2001)