ITERATIVE-FFT COMBINED METHOD FOR 2D POISSON PROBLEMS WITH IMMERSED INTERFACE METHOD*

XINJIE JI†

Abstract. Immersed Interface Method (IIM) is a novel boundary treatment method for enforcing arbitrary boundaries in a computational domain. A fast iterative-FFT combined method is introduced to solve a reduced linear system derived from the 2D Poisson problems with IIM method. This method solves the reduced system by an iterative method while using a fast Fourier transform (FFT) to accelerate the matrix inverse in every iteration. In this project, we consider the generalized minimal residual method (GMRES) and Biconjugate gradient stabilized (l) method (BICGStab(l)). The GMRES-FFT, BICGStab(1)-FFT, BICGStab(2)-FFT, and BICGStab(3)-FFT are tested and compared for solving the reduced linear system. Moreover, the direct solver, GMRES, BICGStab(1), BICGStab(2), and BICGStab(3) are tested and compared for solving the original large-scale unsymmetric matrix system. Different matrix sizes are considered and results show that the iterative-FFT combined method is more stable and faster than solving the original system, and the BICGStab(l)-FFT method could better balance the accuracy and cost.

Key words. Iterative method, FFT, GMRES, BICGStab(l), Immersed interface method, Poisson equation

1. Introduction.

Computational fluid dynamics (CFD) applications often involve simulations of cases with intricate boundary geometries and dynamic motions, presenting significant challenges. Traditional methods for handling such complex boundaries require frequent adaptations and re-meshing of the underlying mesh, resulting in increased complexity and computational costs. However, the immersed method offers a promising alternative by eliminating the need for mesh modifications, thereby enhancing the simplicity and efficiency of these simulations [7].

While the immersed method provides numerous advantages, it introduces new computational challenges, particularly in solving the Poisson problem, which is the most resource-intensive aspect of a CFD solver. The implementation of the immersed interface method (IIM) results in the generation of a large-scale, unsymmetric matrix for the Poisson problem. Solving this matrix using direct methods becomes prohibitively expensive, necessitating the exploration of alternative strategies for efficient computation. Gillis et al. [9] proposed a reduced system for solving Poisson problems using the IIM method. They employed a GMRES-FFT method to solve the reduced system, which was further validated in subsequent studies such as [4] and [7]. However, no further investigation has been conducted regarding the combination of the GMRES method and the FFT method, nor has there been a comparison between the GMRES method and other iterative methods. These aspects warrant further exploration and analysis to assess their effectiveness and potential improvements for solving the Poisson problem in CFD simulations.

The iterative-FFT combined algorithm has proven to be valuable not only in computational fluid dynamics (CFD) simulations but also in various other research fields. Researchers have explored and utilized this algorithm beyond CFD, applying it to solve problems such as electromagnetic radiation and elastic wave propagation. Sarkar et al. [14] and Abubakar et al. [1] employed the combination of FFT and Conjugate Gradient (CG) method to tackle electromagnetic radiation problems. Pelekanos et al. [11] utilized this algorithm to address the elastic wave propagation problem, while Yang et al. [17] further enhanced the method by introducing a pre-conditioning tech-

*May 2023.

†Department of Mechanical Engineering (xinjie@mit.edu).

nique.

Recently, attention has been drawn to the GMRES-FFT and BICGStab-FFT methods in research studies. These methods combine newly-developed iterative algorithms with the FFT algorithm and have been investigated for their robustness in various applications. For instance, Georgakis et al. [8] utilized the GMRES-FFT method, implementing a regular grid to obtain the impedance matrix with a Toeplitz structure and employing FFT-accelerated matrix-vector multiplication. Similarly, the BICGStab-FFT method was utilized to solve a similar problem in [5], which was further improved through the use of a pre-conditioning method in [6]. Although these studies have demonstrated promising computational efficiency, no direct comparison between the GMRES-FFT and BICGStab-FFT methods has been conducted. Such a comparative analysis would be beneficial for further understanding the strengths and limitations of these methods in different applications.

In terms of the iterative method, GMRES is a Krylov subspace method that iteratively constructs an orthonormal basis for a subspace generated by the iterates [12]. It provides a flexible framework for solving linear systems, particularly when the coefficient matrix is sparse or non-symmetric [18]. The GMRES algorithm has excellent numerical stability and convergence properties, making it a popular choice for tackling large-scale problems arising in scientific simulations, structural mechanics, and fluid dynamics [7]. Many improvements are proposed for the performance of GMRES, such as using the preconditioning method [2], and augmentation technique [10].

On the other hand, BICGStab is another powerful iterative method that addresses the limitations of the Conjugate Gradient (CG) method when applied to non-symmetric systems. The BICGStab(l) algorithm builds upon the BICGStab method by introducing a parameter l, which represents the number of steps used to orthogonalize the residual vector[15]. By adjusting this parameter, the algorithm can achieve enhanced convergence and robustness, particularly when dealing with ill-conditioned or highly non-linear systems [16]. The BICGStab(l) method can also be fastened by a preconditioning teachnique in [3].

In this project, we will implement the GMRES-FFT and BICGStab(l)-FFT methods for solving the reduced linear system. The results are compared to show the performance of these algorithms.

2. Problem set. This project focuses on solving a Poisson problem in computational fluid dynamics with an arbitrary boundary condition on a Cartesian grid.

$$\nabla^2 \psi = -\omega, \tag{2.1}$$

where ψ is the sreamfunction of the flow field, and ω is the vorticity of the flow field. The flow domain is a unit square with dimensions $[0, 1) \times [0, 1)$. A star-shaped body is placed inside the domain at $[0.51, 0.52]$. For a star locates at the origin, its level-set function in a polar coordinate $[r, \theta]$ is,

$$f = r - (r_m + r_d \cos(n\theta)), \tag{2.2}$$

where $r_m = 0.25$ is the mean radius of the star, $r_d = 0.01$ is the deviation radius of the star, $n = 7$ is the number of pointers of the star.

To validate our results, the following exact solutions are used,

$$\psi = -\cos(2\pi x)\cos(2\pi y) \tag{2.3}$$

$$\omega = -8\pi^2 \cos(2\pi x)\cos(2\pi y). \tag{2.4}$$

Note that ψ and ω are both zero inside the body. The field of ω in the problem is shown in Figure 1.

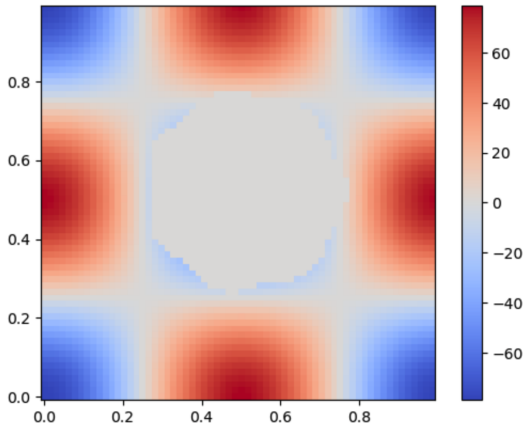

FIG. 1. *ω field of the setting. A star-shaped body is embedded in the field.*

To numerically solve the problem, we set an $N \times N$ Cartesian grid and use the second-order centered finite difference method for the discretization. The Dirichlet boundary condition is enforced on the domain boundary for simplicity. The IIM method [7] is used to enforce the arbitrary body boundary. In this case, the discretized Poisson problem can still be written into a simple linear system form:

$$\hat{L}\psi = f, \tag{2.5}$$

but here $\hat{L} \in \mathbb{C}^{N^2 \times N^2}$ is an arbitrary sparse matrix whose structure depends on the shape of the arbitrary body boundary. As a result, we could only solve it with a direct solver or an iterative solver. But usually, such a method has a high computation cost $\approx \mathcal{O}(N^4)$. To solve the problem faster, [9] introduces a much smaller system to solve the Poisson equation: given N_a the number of points near the boundary, N_c the number of points containing the boundary conditions, we have

$$(I_A + AL^{-1}E_A)\gamma = AL^{-1}f + B\psi_b, \tag{2.6}$$

$$\psi = L^{-1}(f - E_A\gamma), \tag{2.7}$$

where L is a finite difference discretization matrix for free space, which can be fastly solved by FFT method. $A \in \mathbb{C}^{N_a \times N^2}, E_A \in \mathbb{C}^{N^2 \times N_a}, \gamma \in \mathbb{C}^{N_a}, B\psi_b \in \mathbb{C}^{N_a}$. Then to solve it, from [4] we need several FFT solvers for L^{-1} with cost $\mathcal{O}(N^2 \log N^2)$, and an iterative solver for the first equation with cost $\approx \mathcal{O}(N_a^2)$. The general cost will be much smaller. Note here for reducing the cost, in the iterative method, the algorithm first uses FFT to solve $L^{-1}E_A\gamma$, then calculates the left-hand side. In the following tests, the matrices A, L, E_A, f, B, ψ_b are all calculated from another software which is seen as a black box in this project.

3. Algorithm. In this project, the FFT-based method is implemented for solving the multiplication with L^{-1}, and the GMRES and the BICGStab(l) methods are also implemented for solving the reduced linear system.

3.1. Fast FFT method. In this case, when a second-order finite difference method is applied to a free space domain, and the Dirichlet boundary condition is enforced on the domain boundary. We can use an FFT-based method [13] to accelerate the computation. Consider the 2D Possion problem with Dirichlet boundary condition:

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega \qquad \text{in } \Omega \tag{3.1}$$

$$\psi = \psi_b \qquad \text{on } \Gamma \tag{3.2}$$

On an $N \times N$ uniform Cartesian grid, define every grid point $\mathbf{x}_{i,j} = (x_i, y_j)$, $i = 1, 2, \dots N$, and $j = 1, 2, \dots N$. We can use the second-order finite difference method to discretize the equation:

$$\frac{1}{h^2}(\psi_{i-1,j} - 2\psi_{i,j} + \psi_{i+1,j}) + \frac{1}{h^2}(\psi_{i,j-1} - 2\psi_{i,j} + \psi_{i,j+1}) + \mathcal{O}(h^2) = -\omega \tag{3.3}$$

where $\psi_{i,j}$ is the value of ψ on $\mathbf{x}_{i,j}$, and h is the grid spacing equals to $1/N$. The discretization can be transformed into linear systems as $L\psi = -\omega$, where $L \in \mathbb{C}^{N^2, N^2}$ are given by

$$L = \frac{1}{h^2}\begin{bmatrix} B & I & 0 & \cdots & 0 \\ I & B & I & \cdots & 0 \\ & \ddots & \ddots & \ddots & \\ 0 & \cdots & I & B & I \\ 0 & \cdots & 0 & I & B \end{bmatrix}, B = \begin{bmatrix} -4 & 1 & 0 & \cdots & 0 \\ 1 & -4 & 1 & \cdots & 0 \\ & \ddots & \ddots & \ddots & \\ 0 & \cdots & 1 & -4 & 1 \\ 0 & \cdots & 0 & 1 & -4 \end{bmatrix}, \tag{3.4}$$

B is a Toeplitz matrix whose eigenvalues and eigenvectors of B can be calculated by Fourier methods:

$$\lambda_i = -4 + 2\cos(\frac{i\pi}{N+1}), i = 1, \dots N \tag{3.5}$$

Define $\theta_i = (i\pi)/(N+1)$, the corresponding eigenvectors are:

$$v_i = \sqrt{\frac{2}{N+1}}[\sin\theta_i, \sin(2\theta_i), \dots, \sin(N\theta_i)]^T. \tag{3.6}$$

We define $V = [v_i, v_2, \dots, v_N]$. It is obvious that $V^T B V = \Lambda = diag(\lambda_j)$. Then for the block-Toeplitz matrix L, we have $(V \otimes V)L(V^T \otimes V^T) = \Lambda_L$. Consider the discretized Poisson problem, we can multiply $(V \otimes V)$ on both sides:

$$(V \otimes V)L\psi = (V \otimes V)(-\omega). \tag{3.7}$$

Substituting Λ_L to it, We have:

$$\Lambda_L(V \otimes V)\psi = (V \otimes V)(-\omega). \tag{3.8}$$

The multiplication with $(V \otimes V)$ and its inverse can be fastly applied by the FFT-based method: 2D DST-I transform and the inverse of Λ_L is simply another diagonal matrix $diag(1/\lambda_{L,i})$. So the whole process can be summarized as

1. Compute Sine transform for $-\omega$ using 2D DST-I, then we get $\tilde{\omega}$.
2. Calculate $\tilde{\psi}_i = \tilde{\omega}_i/\lambda_{L,i}$, where the value of $\lambda_{L,i}$ depends on the FD scheme we use.
3. Compute ψ via the inverse 2D DST-I transform of $\tilde{\psi}$.

Then the computational cost of the whole process will be $\mathcal{O}(N^2 \log N^2)$ because of the use of the fast Fourier transforms.

3.2. Standard GMRES. GMRES (Generalized Minimal Residual) is used to solve linear systems like $\hat{L}\psi = f$, where $\hat{L}$ is a large, sparse and non-singular matrix. It is a Krylov subspace method that generates a sequence of approximate solutions, and it is particularly useful when A is too large to be stored in memory [2]. Consider a Krylov subspace $\mathcal{K}_m$ of $\hat{L} \in \mathbb{C}^{N^2 \times N^2}$, which is a subspace spanned by the vectors $K_m = \begin{bmatrix} f & \hat{L}f & \hat{L}^2 f & \dots & \hat{L}^{m-1} f \end{bmatrix}$. From the Arnoldi algorithm, suppose we had a QR factorization of K_m:

$$K_m = Q_m R_m = \begin{bmatrix} q_1 & q_2 & \dots & q_m \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & \dots & R_{1m} \\ 0 & R_{22} & \dots & R_{2m} \\ \vdots & & \ddots & \\ 0 & 0 & \dots & R_{mm} \end{bmatrix}. \tag{3.9}$$

Then the vectors $q_1, \dots, q_m$ are the orthonormal basis for $\mathcal{K}_m$. From Theorem 3.1 we know that $\hat{L}q_m \in \mathcal{K}_{m+1}$, so

$$\hat{L}q_m = H_{1m}q_1 + H_{2m}q_2 + \cdots + H_{m+1,m}q_{m+1} \tag{3.10}$$

for some choice of the H_{ij}. By using orthonormality, we have

$$q_i^*(\hat{L}q_m) = H_{im}, i = 1, 2, \dots, m. \tag{3.11}$$

Then (3.10) can be written into

$$\hat{L}Q_m = \begin{bmatrix} q_1 & q_2 & \dots & q_{m+1} \end{bmatrix} \begin{bmatrix} H_{11} & H_{12} & \dots & H_{1m} \\ H_{21} & H_{22} & \dots & H_{2m} \\ 0 & H_{32} & \dots & H_{3m} \\ \vdots & & \ddots & \\ 0 & 0 & \dots & H_{mm} \\ 0 & 0 & \dots & H_{m+1,m} \end{bmatrix} = Q_{m+1}H_m, \tag{3.12}$$

where H_m has a particular "triangular plus one" structure.

Based on the above method, we can reduce the linear system $\hat{L}\psi = f$ by the lower dimension approximation

$$\min_{\psi \in \mathcal{K}_m} \|\hat{L}\psi - f\| = \min_{z \in \mathbb{C}^m} \|\hat{L}Q_m z - f\| = \min_{z \in \mathbb{C}^m} \|Q_{m+1}H_m z - f\|, \tag{3.13}$$

where we set $\psi = Q_m z$. Note that q_1 is a unit multiple of f, so $f = \|f\|Q_{m+1}e_1$, thus the least square problem becomes

$$\min_{z \in \mathbb{C}^m} \|Q_{m+1}(H_m z - \|f\|e_1)\|. \tag{3.14}$$

The problem is $N^2 \times m$, but we know that for any $w \in \mathbb{C}^{m+1}$,

$$\|Q_{m+1}w\|^2 = w^* Q_{m+1}^* Q_{m+1} w = w^* w = \|w\|. \tag{3.15}$$

It reduces a norm on $\mathbb{C}^{N^2}$ to $\mathbb{C}^{m+1}$. Hence the least square problem can be further reduced to

$$\min_{z \in \mathbb{C}^m} \|H_m z - \|f\|e_1\|. \tag{3.16}$$

Then the problem has a size $(m+1) \times m$. When writing it into an iteration, every iteration we get the solution of this minimization z_m, then $\psi_m = Q_m z_m$ is the m-th approximation to the solution of $\hat{L}\psi = f$. The detailed algorithm implementation is described in Appendix A. The computational cost of the GMRES depends on the number of iterations in practice. If it converges after m iterations, the general cost of GMRES for $\hat{L} \in \mathbb{C}^{N^2 \times N^2}$ will become $\mathcal{O}(mN^4)$.

THEOREM 3.1 (Krylov subspace). *Suppose A is $n \times n$, $0 < m < n$, and a vector u is used to genrate Krylov subspaces K_m. If $x \in \mathcal{K}_m$, then the following hold:*

1. *$x = K_m z$ for some $z \in \mathbb{C}^m$.*
2. *$x \in \mathcal{K}_{m+1}$.*
3. *$Ax \in \mathcal{K}_{m+1}$*

3.3. BICGStab(l). The Bi-Conjugate Gradient Stabilized method with l [16] is also an iterative algorithm for solving linear equations systems. Same to the GMRES method, it is suitable for solving large, sparse, and non-singular matrices. This method is a variant of the Bi-Conjugate Gradient (BiCG) method that is designed to improve the stability and convergence of the algorithm.

To illustrate the idea of BICGStab(l), first, we introduce the BICG. Its concept is quite different from Arnoldi's method because it relies on a biorthogonal basis instead of an orthogonal basis. It uses coupled two-term recurrence

$$u_m = r_m - \beta_{m-1} u_{m-1} \tag{3.17}$$

$$r_{m+1} = r_m - \alpha_m \hat{L} u_m \tag{3.18}$$

to produce a residual r_m at every step m which is orthogonal to the m-th shadow Krylov subspace $\mathcal{K}_m(\hat{L}^*, \tilde{r}_0)$. With $\tilde{r}_0, \tilde{r}_1, \ldots, \tilde{r}_i$ spanning $\mathcal{K}_i(\hat{L}^*, \tilde{r}_0)$ for all $i \leq m$, the coeffiecients β_{m-1} and α_m are such that $\hat{L}u_m \perp \tilde{r}_{m-1}$ and $r_{m+1} \perp \tilde{r}_m$. Then the approximation solution x_{m+1} is updated from $x_m + \alpha_m u_m$. The recurrence can also be written as

$$\begin{aligned} r_{m+1} &= r_m - \alpha_m \hat{L} u_m && \perp \tilde{r}_m \\ \hat{L}u_{m+1} &= \hat{L} r_{m+1} - \beta_m \hat{L} u_m && \perp \tilde{r}_m \\ u_{m+1} &= r_{m+1} - \beta_m u_m. \end{aligned} \tag{3.19}$$

Then since $\tilde{r}_m \in \mathcal{K}_m(\hat{L}^*, \tilde{r}_0)$, there are polynomials P_m of degree m such that $\tilde{r}_m = P_m(\hat{L}^*)\tilde{r}_0$. For convenience, we write $P_m(\hat{L}^*)$ as P_m in the following illustration. P_m is a matrix. Multiply with (3.19) we get

$$\begin{aligned} P_m r_{m+1} &= P_m r_m - \alpha_m \hat{L} P_m u_m && \perp \tilde{r}_0 \\ \hat{L} P_m u_{m+1} &= \hat{L} P_m r_{m+1} - \beta_m \hat{L} P_m u_m && \perp \tilde{r}_0 \\ P_m u_{m+1} &= P_m r_{m+1} - \beta_m P_m u_m. \end{aligned} \tag{3.20}$$

In BICGStab method, we require that P_m satisfies the relationship $P_{m+1} = (1 - \omega_{m+1}\hat{L})P_m$ and

$$P_{m+1} r_{m+1} = P_m r_{m+1} - \omega_{m+1} \hat{L} P_m r_{m+1} \tag{3.21}$$

$$P_{m+1} u_{m+1} = P_m u_{m+1} - \omega_{m+1} \hat{L} P_m u_{m+1}. \tag{3.22}$$

Define the primary residual $\hat{r}_m = P_m r_m$, then updating (3.21) and (3.22) with (3.20) means that we update $\hat{r}_m$ in $\mathcal{S}(P_m, \hat{L}, \tilde{r}_0)$ to $\hat{r}_{m+1}$ in $\mathcal{S}(P_{m+1}, \hat{L}, \tilde{r}_0)$, where

$\mathcal{S}(P_m, \hat{L}, \tilde{r}_0) \equiv \{P_m(\hat{L})v | v \perp \mathcal{K}_m(\hat{L}^*, \tilde{r}_0)\}$ is called the (P_m) Sonneveld subspace. The BICGStab(l) [15] selects ω_i in $\mathbb{C}$ such that at the end of each loop of l of the iterations, the following expression gets the minimum value (Note here the notation γ is independent of the γ in our Poisson problem):

$$\|P_m r_{m+l} - \gamma_{1,m}\hat{L}P_m r_{m+l} - \cdots - \gamma_{l,m}\hat{L}^l P_m r_{m+l}\|. \tag{3.23}$$

Here $1-\gamma_{1,m}\lambda-\cdots-\gamma_{l,m}\lambda^l = (1-\omega_{m+1}\lambda)\cdots(1-\omega_{m+l}\lambda)$ relates γ and ω. In practice, the l loop needs some rearrangement to calculate the minimization result of (3.23). In general, the cost of BICGStab(l) for $\hat{L} \in \mathbb{C}^{N^2 \times N^2}$ also depends on the number of iteration m, and is approximately $\mathcal{O}(mN^4)$. The detailed algorithm is included in Appendix B.

3.4. Iterative and FFT combination. Now we have the FFT-based fast method and the iterative methods in place. To solve the linear systems (2.6) with the FFT-based method for L^{-1}, its left-hand side should be applied through several operations instead of a matrix. So in the iterative methods GMRES and BICGStab(l), we first calculate $L^{-1}E_A\gamma$ using the FFT-based solver, then calculate the whole left-hand side. In practice, since here we use a second-order finite difference method for the discretization, the accuracy of our solver for the Poisson problem is $\approx \mathcal{O}(h^2) = \mathcal{O}(N^{-2})$. As a result, we set the tolerance of the iterative methods as $tol = N^{-2}$, so the solver will always obtain its expected accuracy. Define (2.6) as a classic linear system

$$\hat{A}\gamma = \hat{b}. \tag{3.24}$$

The whole process for the iterative - FFT solver is:

1. Calculate $\hat{b} = AL^{-1}f + B\psi_b$ by FFT-based method.
2. Use the iterative method to solve γ. In every iteration, first, calculate $AL^{-1}(E_A * \hat{x})$ by the FFT-based method, then get the left-hand-side $\hat{A}\hat{x}$, where $\hat{x}$ is an arbitrary vector used in the iterative method.
3. Finally, calculate the solution ψ from the (2.7) by an FFT-based method.

The cost of the whole iterative-FFT method is dominated by the size of grid N and the number of iterations m. In every iteration, we need to apply the FFT method twice. Note here $\gamma \in \mathbb{C}^{N_a}$, so the FFT method dominates the cost in the iterations. In general, the cost of the fast solver should be $\mathcal{O}((mN^2\log(N^2))$.

4. Experimental results. In this section, we first test the accuracy of these implemented algorithms, then compare their numbers of iterations and computational cost.

4.1. Accuracy test. All the results are compared with the known exact solution ψ_e. Define the solution on every grid point as $\psi_{ij}, i = 1, 2, \ldots N, j = 1, 2, \ldots, N$. The L_∞ error is defined as

$$L_\infty = \max_{ij} |\psi_{ij} - \psi_{e,ij}| \tag{4.1}$$

Since here we use a second-order finite difference discretization method, the order of accuracy should perform as $\mathcal{O}(N^{-2})$ with different grid sizes $N \times N$. Here we consider $N = [32, 64, 128, 256]$ and the maximum iteration is set as 1000. First, we use the direct solver (backslash in Julia), the GMRES, the BICGStab(1), the BICGStab(2), and BICGStab(3) to solve the direct $N \times N$ system in (2.5), results are shown in Figure 2.

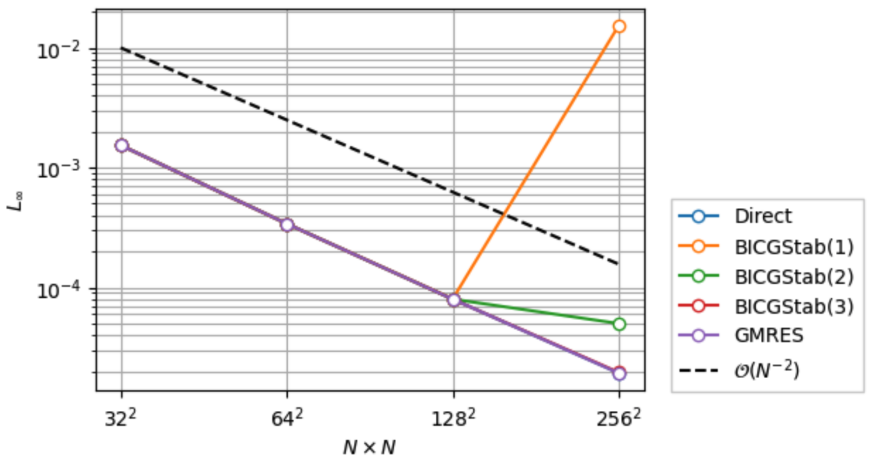

FIG. 2. *Accuracy convergence of different methods for the direct system at different grid sizes N.*

The figure shows that for the direct $N \times N$ system, when $N \leq 128$, all the methods could converge and get the expected accuracy. However, when $N = 256$, all the BICGStab(l) methods could not get the expected solution when the maximum iteration number is 1000. When l is larger, the result will be more accurate. This observation shows that in this case, the BICGStab(l) method is not optimal for solving large-scale problems, especially using a small l. On the other side, the GMRES method could get the expected accuracy for different N.

Then we apply the GMRES-FFT, the BICGStab(1)-FFT, the BICGStab(2)-FFT, and BICGStab(3)-FFT methods to the reduced system (2.6) and (2.7), results are shown in Figure 3. From the figure, we can find that because the reduced system transforms the size of the system from $N^2 \times N^2$ to $N_a \times N_a$, all the iterative methods converge and give us the expected accuracy.

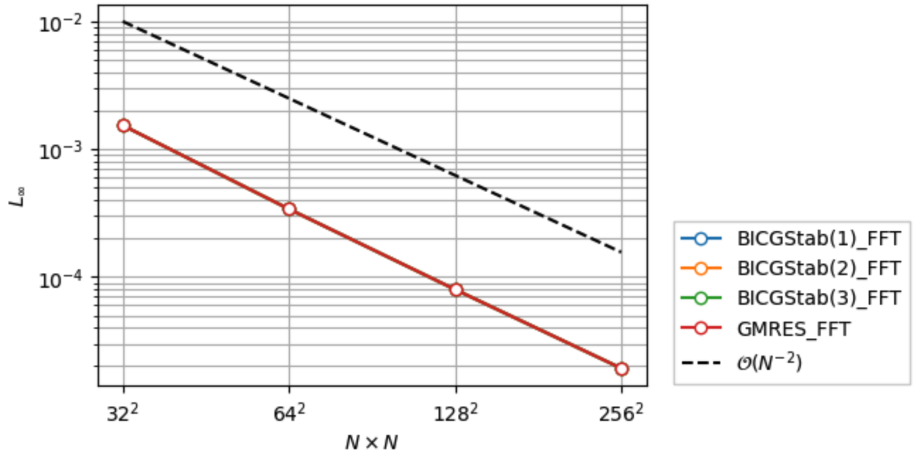

FIG. 3. *Accuracy convergence of different methods for the reduced system at different grid sizes N.*

4.2. Convergence test. From subsection 4.1 we find that some iterative solvers cannot get converged for $N = 256$. As a result, we plot the absolute residual history of these methods for the original $N \times N$ system in Figure 4. It shows that the GMRES method has the fastest convergence speed, but it is flattened when touches some threshold. For the BICGStab(l) method, its convergence speed grows with the increase of l. As a result, if we increase the maximum iteration number, the BICGStab(l) could also get the expected solution, but obviously, its computational cost will be much higher than the GMRES method.

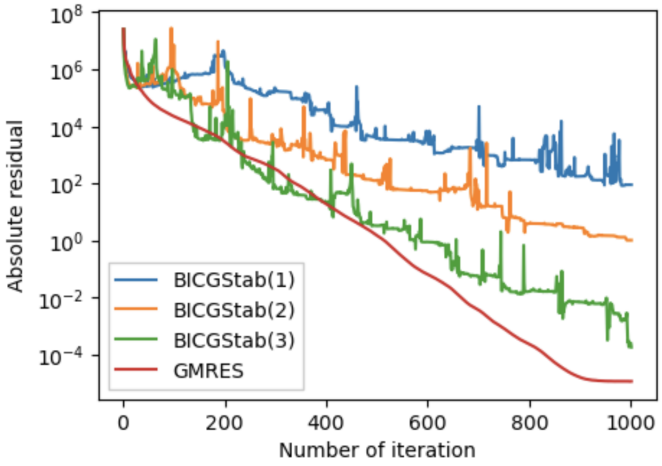

FIG. 4. *Absolute residual history of different methods for the original system at* $N = 256$.

On the other hand, for the reduced linear system, we get the residual history in Figure 5. It illustrates that first, the GMRES method has a relatively fast convergence speed, but it reaches a stagnation point and cannot get the solution with machine precision. This phenomenon may be caused by the FFT method embedded in the GMRES iterations which could be further studied. Meanwhile, the convergence speed of the BICGStab(l) method increases with the increase of l, and when $l = 2$ and $l = 3$, their speeds are faster than the GMRES method. All the tested BICGStab(l) can get the solution with machine precision. To summarize, for the reduced system, If we require high accuracy, the BICGStab(l) with a relatively large l is better.

The number of iterations under different grid sizes using these methods is also shown in Figure 6. It illustrates a similar conclusion with Figure 5. In these algorithms, the BICGStab(3) method could always get the fastest convergence speed. It is interesting to note that the number of iterations of BICGStab(1) grows fast with the increase in grid size.

4.3. Order of magnitude test. Finally, we test the computational cost of these iterative-FFT combined methods by measuring their time cost for different grid sizes. Results are shown in Figure 7. We can find that generally the time cost of all these methods are close to each other and performs as $\mathcal{O}(f(N)N^2)$, where $f(N)$ is related to the size of the grid. In subsection 3.4 we expect that the general cost is $\mathcal{O}(m \log(N^2)N^2)$, where m is the iteration number. Then the computational cost is as expected. Note for the BICGStab(l) method, we add l loops in every iteration that require some additional FFT procedures. As a result, for BICGStab(l) method, its general cost should be $\mathcal{O}(lm \log(N^2)N^2)$. It explains why although different l get

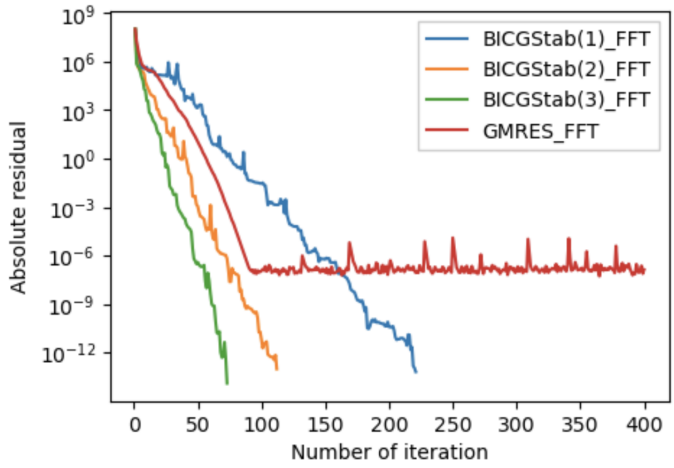

FIG. 5. *Absolute residual history of different methods for the reduced system at* $N = 256$.

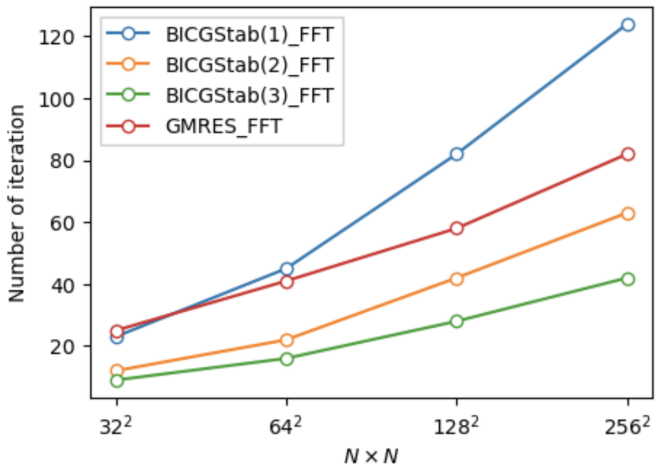

FIG. 6. *Number of iterations of different methods for the reduced system at different grid sizes* N.

different numbers of iterations, they have nearly the same computational cost.

5. Conclusions. In this project, we construct a reduced system for solving the 2D Poisson problems with immersed interface method. An iterative-FFT combined method is introduced to solve such problems and the GMRES and BICGStab(l) methods are implemented and compared. From the experimental results, we get the following conclusions for solving the Poisson problem:

1. If solving the linear system directly, the GMRES has the highest convergence speed.
2. If solving the reduced system, the BICGStab(3) method has the fastest convergence speed.
3. If solving the reduced system, the GMRES method will reach a stagnation point around 10^{-7}. Meanwhile, the BICGStab(l) can get the machine's precision.

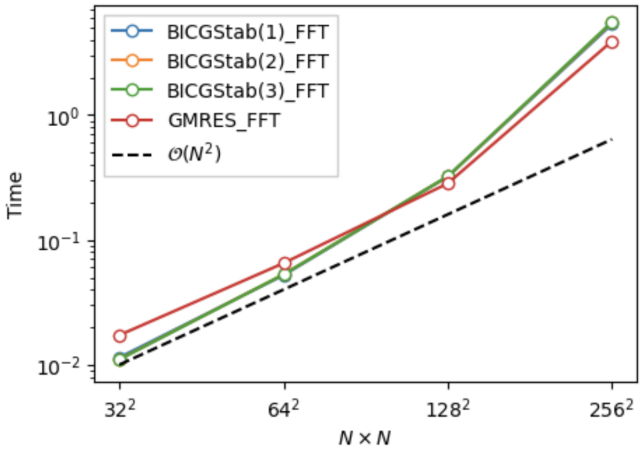

FIG. 7. *Time of different methods for the reduced system at different grid sizes N.*

4. If solving the reduced system, the computational costs of all these iterative-FFT combined methods are close and perform as $\mathcal{O}(mN^2 \log(N^2))$. Here m is the number of iterations. Specifically for BICGStab(l), m is the number of iterations times l. N is the grid size of the 2D $N \times N$ grid.

To summarize, BICGStab(l) is more suitable for solving the reduced system by iterative-FFT combined method because it balances the accuracy and computational cost.

Appendix A. Implementation of GMRES. The implementation of GMRES for a linear system $Ax = b$. Assume $A \in \mathbb{C}^{n\times n}$, the maximum iteration number is m and we have an initial guess $x = 0$, create $Q \in \mathbb{C}^{n\times(m+1)}$, $Q = \begin{bmatrix} q_1 & \cdots & q_{m+1} \end{bmatrix}$, and $H \in \mathbb{C}^{(m+1)\times m}$

Algorithm A.1 Standard GMRES

$q_1 = b/\|b\|$
for j in $1 : m$ **do**
  $v = A * q_j$
  for i in $1 : j$ **do**
    $H_{i,j} = q_j^T * v$
    $v = v - H_{i,j} * q_i$
  end for
  $H_{j+1,j} = \|v\|$
  $q_{j+1} = v/H_{j+1,j}$
  $r = \|b\| * e_1$
  $z = H^{-1}_{1:j+1,1:j} * r$
  $x = \begin{bmatrix} q_1 & \cdots & q_j \end{bmatrix} * z$
  if $\|Ax - b\| < tol$ **then**
    return x
  end if
end for

Appendix B. Implementation of BICGStab(l). The implementation of

BICGStab(l) for a linear system $Ax = b$. Assume $A \in \mathbb{C}^{n\times n}$.

Algorithm B.1 BICGStab(l)

$k = -l + 1$
choose $x_1, \tilde{r}_1$
$r_1 = b - Ax_1$
$u_1 = 0, \rho_1 = 1, \alpha = 0, \omega = 1$
while $\|r_{k+l}\| < tol$ **do**
  $k = k + l$
  $\hat{u}_1 = u_1, \hat{r}_0 = r_k, \hat{x}_1 = x_k, \rho_1 = -\omega\rho_1.$
  for j in $1 : l$ **do**
    $\rho_2 = (\hat{r}_j, \hat{r}_1), \beta = \alpha\rho_2/\rho_1, \rho_1 = \rho_2$ {BI-CG part}
    for i in $1 : j$ **do**
      $\hat{u}_i = \hat{r}_i - \beta\hat{u}_i$
    end for
    $\hat{u}_{j+1} = A\hat{u}_j$
    $\gamma = (\hat{u}_{j+1}, \hat{r}_1), \alpha = \rho_1/\gamma$
    for i in $1 : j$ **do**
      $\hat{r}_i = \hat{r}_i - \alpha\hat{u}_{i+1}$
    end for
    $\hat{r}_{j+1} = A\hat{r}_j, \hat{x}_1 = \hat{x}_1 + \alpha\hat{u}_1$
  end for
  for j in $2 : l + 1$ **do**
    for i in $2 : j - 1$ **do**
      $\tau_{ij} = 1/\sigma_i(\hat{r}_j, \hat{r}_i)$ {modified BI-CG}
      $\hat{r}_j = \hat{r}_j - \tau_{ij}\hat{r}_i$
    end for
    $\sigma_j = (\hat{r}_j, \hat{r}_j), \gamma'_j = 1/\sigma_j(\hat{r}_0, \hat{r}_j)$
  end for
  $\gamma_l = \gamma'_l, \omega = \gamma_l$
  for j in $l : 2$ **do**
    $\gamma_j = \gamma'_j - \sum_{i=j+1}^{l+1} \tau_{ji}\gamma_i$
  end for
  for j in $2 : l$ **do**
    $\gamma''_j = \gamma_{j+1} + \sum_{i=j+1}^{l} \tau_{ji}\gamma_{i+1}$
  end for
  $\hat{x}_1 = \hat{x}_1 + \gamma_2\hat{r}_1, \hat{r}_1 = \hat{r}_1 - \gamma'_{l+1}\hat{r}_{l+1}, \hat{u}_1 = \hat{u}_1 - \gamma_{l+1}\hat{u}_{l+1}$
  for j in $2 : l$ **do**
    $\hat{u}_1 = \hat{u}_1 - \gamma_j\hat{u}_j, \hat{x}_1 = \hat{x}_2 + \gamma''_j\hat{r}_j, \hat{r}_1 = \hat{r}_1 - \gamma'_j\hat{r}_j$
  end for
  $u_{k+l} = \hat{u}_1, \hat{r}_{k+l} = \hat{r}_1, x_{k+l} = \hat{x}_1.$
end while
return x

Acknowledgments. This project is implemented and finished by myself, the code can be found on my github page: https://github.com/xinjieji/IterativeProject.

REFERENCES

[1] A. Abubakar and P. M. van den Berg, *Iterative forward and inverse algorithms based on domain integral equations for three-dimensional electric and magnetic objects*, Journal of Computational Physics, 195 (2004), pp. 236–262, https://doi.org/10.1016/j.jcp.2003.10.009.

[2] A. H. Baker, E. R. Jessup, and T. Manteuffel, *A Technique for Accelerating the Convergence of Restarted GMRES*, SIAM Journal on Matrix Analysis and Applications, 26 (2005), pp. 962–984, https://doi.org/10.1137/S0895479803422014. Publisher: Society for Industrial and Applied Mathematics.

[3] M. Carr, M. Bleszynski, and J. Volakis, *A near-field preconditioner and its performance in conjunction with the BiCGstab(ell) solver*, IEEE Antennas and Propagation Magazine, 46 (2004), pp. 23–30, https://doi.org/10.1109/MAP.2004.1305531. Conference Name: IEEE Antennas and Propagation Magazine.

[4] H. Feng and S. Zhao, *Fft-based high order central difference schemes for three-dimensional poisson's equation with various types of boundary conditions*, Journal of Computational Physics, 410 (2020), p. 109391, https://doi.org/https://doi.org/10.1016/j.jcp.2020.109391.

[5] R. Florencio, Somolinos, I. González, and F. Cátedra, *BICGSTAB-FFT Method of Moments with NURBS for Analysis of Planar Generic Layouts Embedded in Large Multilayer Structures*, Electronics, 9 (2020), p. 1476, https://doi.org/10.3390/electronics9091476. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute.

[6] R. Florencio, Somolinos, I. González, and F. Cátedra, *Fast Preconditioner Computation for BICGSTAB-FFT Method of Moments with NURBS in Large Multilayer Structures*, Electronics, 9 (2020), p. 1938, https://doi.org/10.3390/electronics9111938. Number: 11 Publisher: Multidisciplinary Digital Publishing Institute.

[7] J. Gabbard, T. Gillis, P. Chatelain, and W. M. van Rees, *An immersed interface method for the 2D vorticity-velocity Navier-Stokes equations with multiple bodies*, Journal of Computational Physics, 464 (2022), p. 111339, https://doi.org/10.1016/j.jcp.2022.111339. arXiv:2109.13628 [physics].

[8] I. P. Georgakis, I. I. Giannakopoulos, M. S. Litsarev, and A. G. Polimeridis, *A Fast Volume Integral Equation Solver With Linear Basis Functions for the Accurate Computation of EM Fields in MRI*, IEEE Transactions on Antennas and Propagation, 69 (2021), pp. 4020–4032, https://doi.org/10.1109/TAP.2020.3044685. Conference Name: IEEE Transactions on Antennas and Propagation.

[9] T. Gillis, G. Winckelmans, and P. Chatelain, *Fast immersed interface Poisson solver for 3D unbounded problems around arbitrary geometries*, Journal of Computational Physics, 354 (2018), pp. 403–416, https://doi.org/10.1016/j.jcp.2017.10.042.

[10] Q. Liu, R. B. Morgan, and W. Wilcox, *Polynomial preconditioned gmres and gmres-dr*, SIAM Journal on Scientific Computing, 37 (2015), pp. S407–S428, https://doi.org/10.1137/140968276.

[11] G. Pelekanos, R. E. Kleinman, and P. M. van den Berg, *A Weak Form of the Conjugate Gradient FFT Method for Two-Dimensional Elastodynamics*, Journal of Computational Physics, 160 (2000), pp. 597–611, https://doi.org/10.1006/jcph.2000.6476.

[12] Y. Saad and M. H. Schultz, *Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 856–869, https://doi.org/10.1137/0907058.

[13] Y. Saaf, *Iterative methods for sparse linear systems*.

[14] T. Sarkar, E. Arvas, and S. Rao, *Application of FFT and the conjugate gradient method for the solution of electromagnetic radiation from electrically large and small conducting bodies*, IEEE Transactions on Antennas and Propagation, 34 (1986), pp. 635–640, https://doi.org/10.1109/TAP.1986.1143871. Conference Name: IEEE Transactions on Antennas and Propagation.

[15] G. L. G. Sleijpen and D. R. Fokkema, *BICGStab(l) for linear equations involving unsymmetric matrices with complex spectrum*, Electronic Transactions on Numerical Analysis, 1 (1993), pp. 11–32.

[16] G. L. G. Sleijpen, H. A. van der Vorst, and D. R. Fokkema, *BiCGstab(l) and other hybrid Bi-CG methods*, Numerical Algorithms, 7 (1994), pp. 75–109, https://doi.org/10.1007/BF02141261.

[17] J. Yang, A. Abubakar, P. M. van den Berg, T. M. Habashy, and F. Reitich, *A CG-FFT approach to the solution of a stress-velocity formulation of three-dimensional elastic scattering problems*, Journal of Computational Physics, 227 (2008), pp. 10018–10039, https://doi.org/10.1016/j.jcp.2008.07.027.

[18] Q. Zou, *GMRES algorithms over 35 years*, Applied Mathematics and Computation, 445 (2023), p. 127869, https://doi.org/10.1016/j.amc.2023.127869.