Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **18**, No. 1, pp. 170-188

DOI: 10.4208/aamm.OA-2023-0319
February 2026

A Fast Cascadic Multigrid Method for Direct Finite Difference Discretizations of 3D Biharmonic Equations on Rectangular Domains

Kejia Pan, Pengde Wang, Jinxuan Wang* and Xiaoxin Wu

School of Mathematics and Statistics, Central South University, Changsha, Hunan 410083, China

Received 12 December 2023; Accepted (in revised version) 15 October 2024

Abstract. A new extrapolation cascadic multigrid (EXCMG) method is developed to solve large sparse symmetric positive definite systems resulting from the classical 25-point finite-difference discretizations of the three-dimensional (3D) biharmonic equation on rectangular domains. We accomplish this by designing a quartic interpolation-based prolongation operator and using the symmetric successive over-relaxation (SSOR) preconditioned CG method as the multigrid smoother. For the new prolongation operator, quartic interpolations are used for the finite difference solutions on coarse and fine grids twice and once so that two approximations can be obtained on the next finer grid, and then the completed Richardson extrapolation is used for these two approximations to obtain an excellent initial guess on the next finer grid. The proposed EXCMG method with the new prolongation operator is easier to implement than the original EXCMG method. Numerical experiments demonstrate that the new EXCMG is a highly efficient solver for the 3D biharmonic equation and is considerably faster than the original EXCMG method and the aggregation-based algebraic multigrid (AGMG) method developed by Y. Notay. The proposed EXCMG method can solve discrete 3D biharmonic equations with more than 100 million unknowns in dozens of seconds.

AMS subject classifications: 65M10, 78A48

Key words: Biharmonic equation, cascadic multigrid, quartic interpolation, high efficiency, Richardson extrapolation.

1 Introduction

A class of fourth-order partial differential equations, known as the biharmonic equation, has emerged from fields such as continuum mechanics, fluid mechanics, geohydrodynamics, and geophysics. These equations encompass linear elasticity theory, phase-field

*Corresponding author.
Emails: kejiapan@csu.edu.cn (K. Pan), 212101020@csu.edu.cn (P. Wang), wjx171210@csu.edu.cn (J. Wang), 1130032895@qq.com (X. Wu)

simulations, and Navier-Stokes flows. Due to their significant practical applications, numerous numerical methods have been proposed for solving these equations [1–15]. However, most of these works focus on the two-dimensional cases, with limited attention given to 3D biharmonic equations. This is due to the fact that the ill-conditioning of the discrete biharmonic problems is more severe than that of the discrete problems in the second-order systems. The condition number increases at a rate of $\mathcal{O}(h^{-4})$ rather than $\mathcal{O}(h^{-2})$ for the second-order problems (where h is the meshsize in the discretization). Thus, solving 3D biharmonic problems requires significant computational power and memory storage [8, 12]. This work aims to develop a fast solver that can handle the ill-conditioning of discrete 3D biharmonic problems.

In this paper, we will consider the numerical solution for the following 3D biharmonic equation in a unit cube:

$$\Delta^2 p(x,y,z) = f(x,y,z), \quad (x,y,z) \in \Omega = [0,1]^3, \tag{1.1}$$

with Dirichlet boundary conditions of first kind

$$p(x,y,z) = g_0(x,y,z), \quad \frac{\partial p}{\partial n} = g_1(x,y,z), \quad (x,y,z) \in \partial\Omega, \tag{1.2}$$

or second kind

$$p(x,y,z) = g_0(x,y,z), \quad \frac{\partial^2 p}{\partial n^2} = g_2(x,y,z), \quad (x,y,z) \in \partial\Omega. \tag{1.3}$$

In 3D Cartesian coordinates, the biharmonic operator Δ^2 can be expressed as

$$\Delta^2 p(x,y,z) = \frac{\partial^4 p}{\partial x^4} + \frac{\partial^4 p}{\partial y^4} + \frac{\partial^4 p}{\partial z^4} + 2\frac{\partial^4 p}{\partial x^2 y^2} + 2\frac{\partial^4 p}{\partial x^2 z^2} + 2\frac{\partial^4 p}{\partial y^2 z^2}. \tag{1.4}$$

Numerous methods for numerically solving biharmonic equations have been proposed in the literature. One widely used technique is to split the equation $\Delta^2 p = f$ into two coupled Poisson equations for p and q: $\Delta p = q$ and $\Delta q = f$. This approach has been extensively used in previous work [2, 6, 7]. However, introducing the new variable q creates difficulties in defining its boundary conditions, and the accuracy of the numerical solution strongly depends on the method used to approximate these missing boundary values, which is the most challenging aspect of the coupling method. An alternative approach was proposed by Ribeiro Dos Santos in [5], which involves discretizing the equation on a uniform grid using a 25-point computational stencil with a second-order truncation error. This method requires modification at grid points near the boundaries, and it becomes extremely difficult to solve the resulting linear systems of 3D biharmonic equations by using direct methods when the number of unknowns is large. Iterative methods such as Jacobi or Gauss-Seidel can converge slowly or diverge, and the direct method is only applicable for moderate values of grid width h, according to Dehghan

and Mohebbi [8]. To the best of our knowledge, there are no numerical results for solving 3D biharmonic equations with large-scale meshes using any of these methods.

In this paper, we introduce an improved and effective EXCMG method for solving 3D biharmonic equations (1.1) in a unit cube with both first and second-kind Dirichlet boundary conditions (1.2)-(1.3) by using the conventional finite difference approximation. Our method employs the 25-point finite difference (FD) scheme to approximate the 3D biharmonic equation. To overcome the computational difficulties arising from the resulting linear system, we employ an excellent initial guess for iterative calculations on the next finer grid. This is achieved by combining Richardson extrapolation technique and quartic interpolation. Quartic interpolations are used for the finite difference solutions on coarse and fine grid twice and once so that two approximations can be obtained on the next finer grid, and then the Richardson extrapolation formula is employed for these two approximations to obtain the good initial guess on the next finer grid, which is more flexible and efficient than the previous published work [16, 17]. We find the solution to the large positive definite linear systems by using the SSOR preconditioned CG method. Results show that the performance of proposed method is much better than the original EXCMG method [17] and the aggregation-based algebraic multigrid (AGMG) method developed by Y. Notay [18,19]. Both of these two methods (EXCMG, AGMG) are efficient solvers for large sparse linear systems arising from the discretization of elliptic PDEs.

The remainder of the paper is structured as follows: Section 2 outlines the 25-point difference approximation for the 3D biharmonic equation, along with the modified difference scheme designed for grid points near boundaries. In Section 3, we present the EXCMG method with the SSORCG smoother used to solve the discrete biharmonic equation. Section 4 demonstrates the method's effectiveness and accuracy through several numerical examples, followed by a summary in the last section.

2 Second-order finite difference discretization

Under the cube region $\Omega=[0,1]\times[0,1]\times[0,1]$, the domain is discretized with equal meshsizes $h=1/N$ and N represents the numbers of uniform intervals along all three coordinate directions (x, y and z). Each grid points can be denoted as (x_i,y_j,z_k), where $x_i=ih$, $y_j=jh$ and $z_k=kh$, $i,j,k=0,1,\cdots,N$ are symbols in three directions respectively. The numerical solution of each point (x_i,y_j,z_k) is recorded as $p_{i,j,k}$.

For internal grid points (i=2,$\cdots$,$N-2$, j=2,$\cdots$,$N-2$, k=2,$\cdots$,$N-2$), as shown in Fig. 1, the 25-point second-order difference scheme for 3D biharmonic equation was derived [5, 12]

$$\begin{aligned}&42p_{i,j,k}-12(p_{i-1,j,k}+p_{i+1,j,k}+p_{i,j-1,k}+p_{i,j+1,k}+p_{i,j,k-1}+p_{i,j,k+1})\\&+p_{i-2,j,k}+p_{i+2,j,k}+p_{i,j-2,k}+p_{i,j+2,k}+p_{i,j,k-2}+p_{i,j,k+2}\end{aligned}$$

$$+2(p_{i-1,j-1,k}+p_{i-1,j+1,k}+p_{i+1,j-1,k}+p_{i+1,j+1,k}+p_{i-1,j,k-1}+p_{i+1,j,k-1}$$
$$+p_{i,j-1,k-1}+p_{i,j+1,k-1}+p_{i-1,j,k+1}+p_{i+1,j,k+1}+p_{i,j-1,k+1}+p_{i,j+1,k+1})=h^4 f_{i,j,k}. \quad (2.1)$$

Note that $p_{i,j,k}$ is connected to mesh points, extending two grids outward from the point (x_i,y_j,z_k) in each direction. Thus, the difference formulation (2.1) for the mesh points near the domain boundary $\partial\Omega$ involves at least one value of point outside the domain, and these points outside the domain are fictitious points which need to be replaced by the internal points through the boundary condition. These could be done for both first and second kind of boundary conditions.

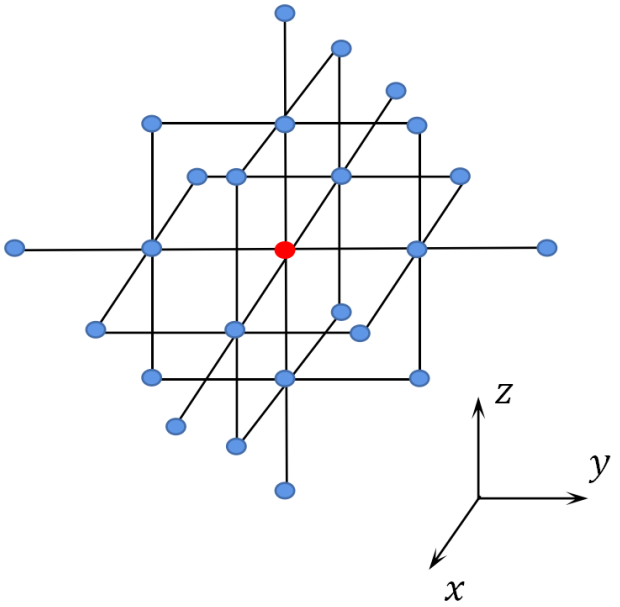


Figure 1: 25-point finite difference stencil.

First kind of boundary condition

For example, for $i=1$, the point (x_{-1},y_j,z_k) lies outside the computational domain, and the value on the fictitious point (x_{-1},y_j,z_k) can be obtained through the following central difference formula called the reflection formulas [3,4]

$$\frac{p_{1,j,k}-p_{-1,j,k}}{2h}=\left(\frac{\partial p}{\partial x}\right)_{0,j,k}, \quad (2.2)$$

where $\left(\frac{\partial p}{\partial x}\right)_{0,j,k}$ can be obtained from the boundary condition (1.2) and $p_{-1,j,k}$ is given by

$$p_{-1,j,k}=p_{1,j,k}-2h\left(\frac{\partial p}{\partial x}\right)_{0,j,k}=p_{1,j,k}+2hg_1(0,y_j,z_k). \tag{2.3}$$

Second kind of boundary condition

The value on the fictitious point (x_{-1},y_j,z_k) can be obtained through the following central difference formula for second-order derivatives

$$\frac{p_{1,j,k}-2p_{0,j,k}+p_{-1,j,k}}{h^2}=\left(\frac{\partial^2 p}{\partial x^2}\right)_{0,j,k}, \tag{2.4}$$

where $p_{0,j,k}$ and $\left(\frac{\partial^2 p}{\partial x^2}\right)_{0,j,k}$ can be obtained from the boundary condition (1.3) and $p_{-1,j,k}$ is given by

$$p_{-1,j,k}=-p_{1,j,k}+2p_{0,j,k}+h^2\left(\frac{\partial^2 p}{\partial x^2}\right)_{0,j,k}=-p_{1,j,k}+2p_{0,j,k}-h^2g_2(0,y_j,z_k). \tag{2.5}$$

The values on the fictitious points outside computational domain on other boundaries can be obtained in a way similar to Eq. (2.3) or Eq. (2.5).

Remark 2.1. The second kind of boundary condition can be addressed the splitting method where the biharmonic equation is replaced by two Possion equations

$$\Delta p(x,y,z)=q(x,y,z), \qquad (x,y,z)\in\Omega=[0,1]^3, \tag{2.6a}$$

$$p(x,y,z)=g_0(x,y,z), \quad \frac{\partial^2 p}{\partial n^2}=g_2(x,y,z), \qquad (x,y,z)\in\partial\Omega, \tag{2.6b}$$

and

$$\Delta q(x,y,z)=f(x,y,z), \qquad (x,y,z)\in\Omega=[0,1]^3 \tag{2.7a}$$

$$q(x,y,z)=\tilde{g}_0(x,y,z), \quad \frac{\partial^2 q}{\partial n^2}=\tilde{g}_2(x,y,z), \qquad (x,y,z)\in\partial\Omega, \tag{2.7b}$$

where Δ represents the Laplace operator. The boundary information $p_{xx}(x,y,z)$, $p_{yy}(x,y,z)$ and $p_{zz}(x,y,z)$ can be derived by incorporating the boundary conditions of Eq. (2.6), which provides the necessary conditions for establishing the boundary value problem for the coupled Eq. (2.7). The above coupled equation can be solved using the fast Poisson solvers that are available even for non-rectangular domains. The coupled equation is not suitable for handling the first kind of boundary condition due to its significantly higher complexity compared to the second type, whereas we present the examples focus on the first kind of the boundary condition.

Let p_h and $p_{h/2}$ be the finite difference solutions of Eq. (1.1) under mesh sizes h and $\frac{h}{2}$, respectively. Afterward, a matrix form, which express the finite difference scheme (2.1) and an equation set including formulas of the grid points near the boundary, can be obtained as below

$$A_h p_h = f_h, \tag{2.8}$$

where A_h is the 25-point finite difference matrix, and f_h is the known right hand-side vector.

It can be discovered that the coefficient matrix A_h is symmetric but not diagonally dominant. We stored the matrix A_h in the symmetric compressed sparse row (CSR) format, and used DFEAST_SCSREV routine (via the Intel MKL library) to compute all the eigenvalues of the symmetric matrix A_h with different meshsize h, and found that the matrix is positive definite (the eigenvalues of A_h with $h=1/8$ and $h=1/10$ are plotted in Fig. 2, results shows that all eigenvalues are positive, therefore, numerically A_h is positive definite). However, it is difficult to prove theoretically that A_h is positive definite, and we leave it for the future work. Since the coefficient matrix A_h is positive definite, the preconditioned CG methods such as Incomplete Cholesky Conjugate-Gradient method (ICCG) and SSORCG, are good choices to seek the solution of large sparse linear system (2.8). On the basis of these above methods, we will design an EXCMG with SSORCG

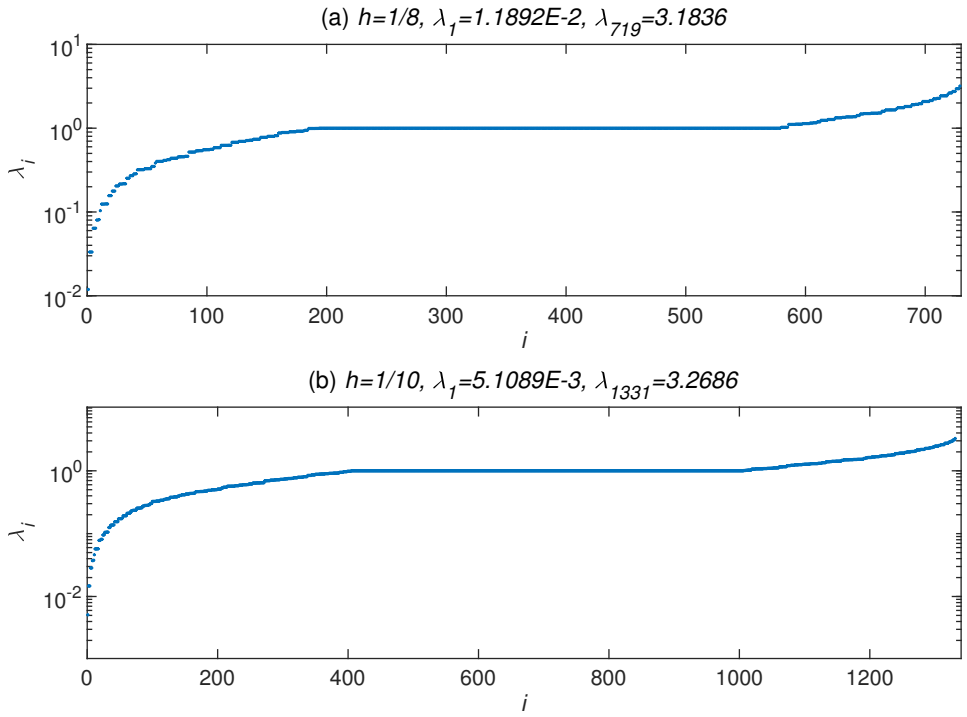


Figure 2: Eigenvalues of the 25-point finite difference matrix A_h.

smoother to solve (2.8) resulting from 25-point discretization of the 3D biharmonic equation with two kinds of Dirichlet boundary conditions, and compare its performance with the original EXCMG method and the AGMG method.

3 Extrapolation cascadic multigrid method

Finding appropriate approaches to solve large linear systems resulting from discretizations of partial differential equations is crucial. Consequently, numerous authors have devoted significant attention to this area and have presented various multigrid methods (MG), for example, the classical MG method, the cascadic multigrid (CMG) method, and the extrapolation cascadic multigrid (EXCMG) method. The classical MG method has been highly successful in many applied fields for several decades. About three decades ago, Deuflhard and Bornemann introduced the CMG method [20]. It is a one-way multigrid method and easy to implement because it does not require coarse grid correction. The EXCMG method was proposed by Chen et al. [16] in 2008, which was inspired by the CMG method. The objective of the EXCMG method was to solve linear systems using linear finite element discretizations of 2D elliptic equations. For the latest developments on the EXCMG method, readers can refer to [22,23] for further details.

The core advantage of the EXCMG method lies in providing a quite good initial guess for the next finer grids, which accelerates the convergence of the smoothing iteration. The linear interpolation-based prolongation operator can only provide a second-order accurate initial guess, consistent with the order of the FD scheme (2.1), thus its acceleration effect is quite limited. In contrast, the original EXCMG method employs the quadratic interpolation-based prolongation operator to construct initial guess, which is a third-order approximation to the discrete solution. This approximation is one order higher in accuracy than the FD scheme itself, effectively reducing the number of iterations during the computation process and greatly enhancing the efficiency. This study mainly focus on how to develop efficient multigrid method to solve the large linear system (2.8) resulting from FD discretization of the 3D biharmonic equation.

3.1 New prolongation operator

Without loss of generality, we consider three nested grids Z_i as shown in Fig. 3, where the mesh sizes are $h_i=h_0/2^i$ $(i=0,1,2)$. Define p^i as the FD solution on the grid Z_i. In this subsection, given second-order FD solutions p^0 and p^1, we will make a lucid explanation to construct an approximation value w^2 to the FD solution p^2 by utilizing extrapolation technique and high-order interpolation technique in detail. The approximation w^2 will be used as a superior initial guesses for the smoother SSORCG on Z_2.

Define $e^i=p^i-p$ as the error of the second-order FD solution p^i. Let's first consider one-dimensional case. We presume that each grid point x_k has the following asymptotic

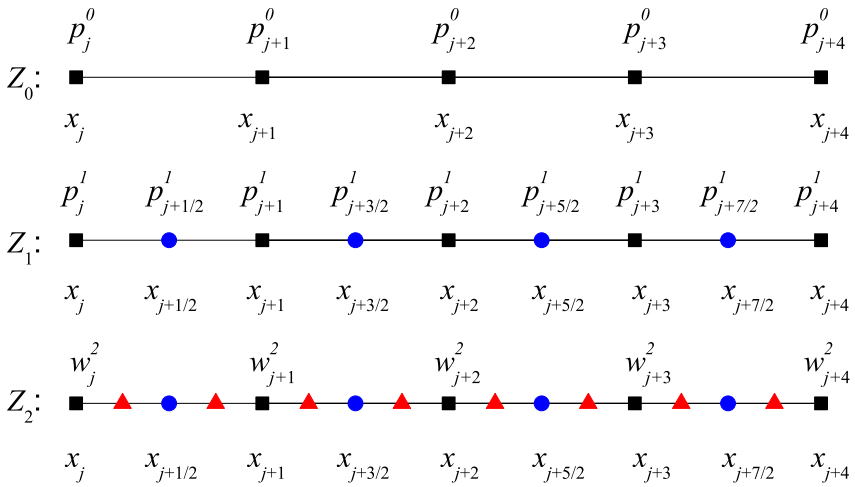


Figure 3: Three nested meshes in 1D.

expansion of error form

$$e^i(x_k)=p^i(x_k)-p(x_k)=a(x_k)h_i^2+\mathcal{O}(h_i^4), \tag{3.1}$$

where $a(x)$ is a suitable smooth function that is independent of h_i.

We divide the coarse element $(x_j,\ x_{j+1})$ into four uniform elements by adding one midpoint (blue circle $x_{j+1/2}$) as well as two points of quadrisection (red triangle $x_{j+1/4}$, $x_{j+3/4}$), as shown in Fig. 3. The same procedure is applied to other elements as well. Given p^0 and p^1, we assume that p^2 can be approximated by their linear combination as follows,

$$dp^1+(1-d)p^0=p^2+\mathcal{O}(h_0^4). \tag{3.2}$$

Substituting Eq. (3.1) into Eq. (3.2) yields $d=5/4$. And we obtain the following nodal extrapolation formulas

$$w_k^2:=\frac{5p_k^1-p_k^0}{4}=p_k^2+\mathcal{O}(h_0^4),\quad k=j,j+1. \tag{3.3}$$

Eq. (3.3) shows the extrapolation method in the 1D case, where the nodal initial guesses of grid Z_2 are obtained by extrapolating the nodal values of grids Z_0 and Z_1. In order to maintain a fourth-order approximation to the FD solutions on Z_2, quartic interpolation on Z_0, Z_1 should be implemented. The FD solution p^0 will be interpolated twice on grid Z_0 (yielding blue circles and red triangles, respectively), whereas the FD solution p^1 requires one time interpolation to obtain the values at red triangles on grid Z_1. After that, we can apply the extrapolation formula Eq. (3.3) to obtain an approximated FD solution of all nodes on Z_2. Hence, the initial guess on Z_2 can be written as

$$w_h=\frac{5\mathcal{L}_{2h}^h(p_{2h})-\mathcal{L}_{2h}^h(\mathcal{L}_{4h}^{2h}(p_{4h}))}{4}, \tag{3.4}$$

where $\mathcal{L}_{4h}^{2h}$ is the quartic interpolation operator defined on Z_0 as follows

$$\mathcal{L}_{4h}^{2h}(p_{4h})(x)=\sum_{k=0}^{4}p_{j+k}^{0}l_{4,4h}^{k}(x), \tag{3.5}$$

and

$$l_{4,4h}^{k}(x)=\prod_{s=0,s\neq k}^{4}\frac{x-x_{j+s}}{x_{j+k}-x_{j+s}}.$$

As for grid Z_1, the quartic interpolation are carried out on both the regions $[x_j,x_{j+2}]$ and $[x_{j+2},x_{j+4}]$. The quartic interpolation on $[x_j,x_{j+2}]$ is given as follows,

$$\mathcal{L}_{2h}^{h}(p_h)(x)=\sum_{k=0}^{4}p_{j+k/2}^{1}l_{4,2h}^{k}(x), \tag{3.6}$$

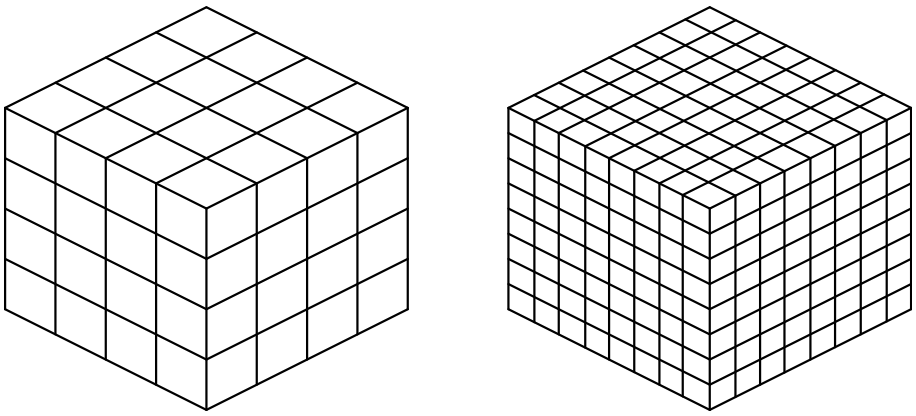
where

$$l_{4,2h}^{k}(x)=\prod_{s=0,s\neq k}^{4}\frac{x-x_{j+s/2}}{x_{j+k/2}-x_{j+s/2}}.$$

The quartic interpolation on $[x_{j+2},x_{j+4}]$ can be obtained in similar way.

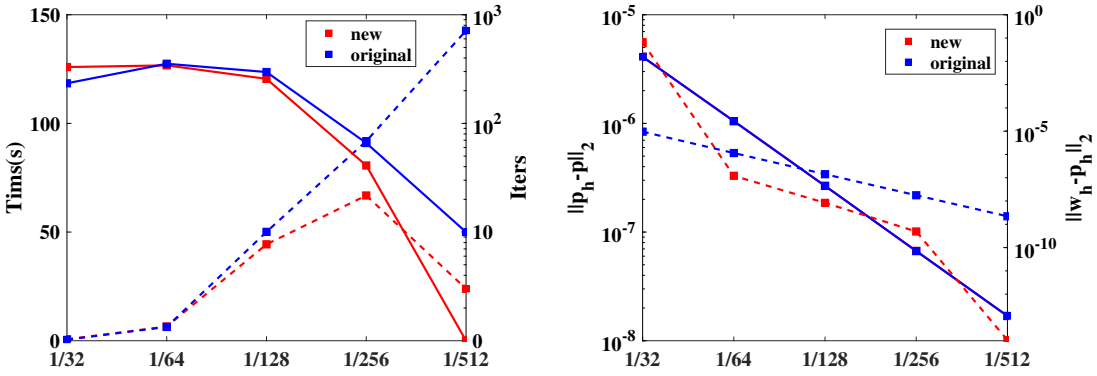
The above method can be easily extended into 3D case. For 3D case shown in Fig. 4, the tri-quartic interpolation will be conducted for the coarse and fine grids twice and once, respectively. After that, the extrapolation formula (3.4) can be used to obtain an approximated FD solution on the next finer grid, which is served as the initial guess for the SSORCG smoother.

As Example 3 in Section 4.1, we conducted a numerical comparison of the prolongation operators between the new EXCMG and original EXCMG. Fig. 5(a) depicts the comparison of the two algorithms in terms of time (dashed line) and iterations (solid line), while Fig. 5(b) shows the L_2 norm error $||p_h-p||_2$ of the numerical solution p_h with respect to the true value p (solid line), as well as the L_2 norm error $||w_h-p_h||_2$ of the numerical solution p_h with respect to the initial guess w_h (dashed line). According to Fig. 5(b), the error $||p_h-p||_2$ of the two algorithms are identical, which indicates that the computational results are accurate. However, it can be clearly seen that with the smaller mesh size, the error $||w_h-p_h||_2$ obtained by the new prolongation operator is obviously better than that obtained by the quadratic-based prolongation operator, indicting that the initial guess constructed based on quartic interpolation prolongation operator has a higher accuracy, which directly reflects the significant differences in the number of required iterations and the computational time. At the finest grid resolution of $512\times512\times512$, the new EXCMG needs only 1 iteration, while the original EXCMG needs 10 iterations. Effective initial guess are the fundamental reason for the high efficiency of EXCMG algorithm.



(a) Coarse grid (b) Fine grid

Figure 4: Illustration of nested cubic meshes.



(a) (b)

Figure 5: The Comparison between the new EXCMG and the original EXCMG: (a) Iters are shown as solid line, and time as dashed line; (b) $||p_h-p||_2$ is represented by solid lines, and $||w_h-p_h||_2$ by a dashed line.

Remark 3.1. It should be mentioned that extrapolation formula (3.4) is easier to implement than the method reported in the previous work [17], since the extrapolated values at mid-points and quater-points can be obtained by using the uniform formula while they are obtained separately by using different formulas in previous work [17].

3.2 Smooth operator

SSOR is a well-known preconditioner, which can be easily obtained from the coefficient matrix. For symmetric matrix A, its SSOR preconditioner is defined as follows

$$M_{SSOR}=(D+\omega L)D^{-1}(D+\omega L^T), \tag{3.7}$$

where D is the diagonal of matrix A, L and its transpose L^T are the strict lower part and upper part of matrix A, respectively and ω is the relaxation factor.

In order to implement the SSOR preconditioned CG method, we need to compute $w=M_{SSOR}^{-1}x$. We can easily obtain w by sequentially solving the following two tridiagonal systems

$$(D+\omega L)D^{-1}z=x, \tag{3.8a}$$

$$(D+\omega L^T)w=z. \tag{3.8b}$$

3.3 Description of the EXCMG algorithm

In Algorithm 3.1, H denotes the mesh size of the coarsest grid. The number of grids is denoted as L ($L>0$) excluding the original two coarsest girds. On the two coarsest grids, a sparse direct solver is used (see line 1-2 in the following Algorithm 3.1). In line 10, w_h represents an approximation to the FD solution p_h, which is attained from the prolongation operator by using numerical solutions p_{2h} and p_{4h}.

Algorithm 3.1 EXCMG algorithm.

Require:

Relative residual tolerance ε, maximum iterations m on the finest mesh, level of grids L

Ensure:

Approximate solution p_h with mesh size h

1: Solve the coarsest grid equation $A_H p_H=f_H$ for p_H by using a sparse direct solver
2: Solve $A_{H/2}p_{H/2}=f_{H/2}$ for $p_{H/2}$ by using a sparse direct solver
3: $h=H/2$
4: **for** $i=1$ to L **do**
5: $h=h/2$
6: $tol=\varepsilon\cdot 10^{i-L}$ % tol is the relative residual tolerance
7: $maxit=m\cdot(2^d)^{L-i}$ % $maxit$ is the maximum number of iterations
8: $\tilde{p}_h=\mathcal{L}_{2h}^h(p_{2h})$ % $\mathcal{L}_{2h}^h$ denotes the quartic interpolation operator
9: $\bar{p}_h=\mathcal{L}_{2h}^h(\mathcal{L}_{4h}^{2h}(p_{4h}))$ % $\mathcal{L}_{4h}^{2h}$ denotes the quartic interpolation operator
10: $w_h=(5\tilde{p}_h-\bar{p}_h)/4$ % w_h is used as the initial guess for SSORCG smoother
11: $p_h \Leftarrow$ SSORCG$(A_h,f_h,tol,maxit,w_h)$ % SSORCG denotes the SSOR preconditioned CG smoother
12: **end for**

4 Numerical experiments

In this section, we will present numerical results for several examples to illustrate the validity of the proposed EXCMG method by comparing with the original EXCMG

method [17] and AGMG method [19]. In the presented results, "Iters" means that the relative residual decreased to a level less than the setting tolerance, namely the number of SSORCG iterations. "Rate" is associated with the grid size of convergence order. Each table lists the L_2-norm error and the infinite-norm error (L_∞) of the numerical solution p_h, L_2-norm of the difference between the initial guess w_h and p_h, and corresponding rate-of-convergence. The computation time of solving linear systems with EXCMG and the total computational cost in terms of work unit (WU) on the finest grid, which is marked as the total computation involved to perform one relaxation sweep on the finest grid, are also given. We wrote the code under the Fortran 90 environment and compiled it through Intel Visual Fortran Compiler XE 12.1. All simulations were worked on a super-computing server with Intel(R) Xeon(R) Gold 6248R CPU @ 3.0GHz CPU and 192 GB RAM.

4.1 Validation of accuracy and efficiency

Problem 1. We take the exact solution of the numerical experiment as follows

$$p(x,y,z)=e^{xyz}. \tag{4.1}$$

The forcing term $f(x,y,z)$ can be yielded from the exact solution, which is given as follows

$$\begin{aligned} f(x,y,z)=&e^{xyz}(x^4y^4+2x^4y^2z^2+x^4z^4+8x^3yz+2x^2y^4z^2+2x^2y^2z^4+4x^2\\ &+8xy^3z+8xyz^3+y^4z^4+4y^2+4z^2). \end{aligned} \tag{4.2}$$

The inclusion of relevant information such as boundary conditions can yield from the exact solution. Starting from the coarsest grid $8\times8\times8$, the finest grid $512\times512\times512$ is attained after seven levels of encryption. As the linear systems on the two coarsest grids are solved directly, the presented results begin from the third level of mesh $32\times32\times32$. First, we conduct an experiment with different choices of ω, where the computation is done on a grid of size $64\times64\times64$. As Fig. 6 shows, the CPU time decreases as ω approaches 2, thus we set $\omega=1.95$ for all cases. The numerical results of Problem 1 via the new EXCMG method with SSORCG smoother (ω=1.95) and ε=10^{-10} are displayed in Table 1. As one can see that the number of iterations becomes smaller when the meshsize becomes smaller. For smallest meshsize $h=1/512$, almost no iteration is required. The L_2-norm and L_∞ of the numerical solutions p_h reach second-order convergence, which confirms the theoretical results.

Moreover, the existing EXCMG method [16,17] is also performed. Errors and convergence rates with relative residual tolerance 10^{-10} are also listed in Table 2. As one can see, the computational cost and final iterations of the existing EXCMG method are larger than the newly proposed method, which indicates that a better initial guess is provided via the new prolongation operator.

Problem 2. The exact solution [12,21] is taken as follows

$$p(x,y,z)=(1-\cos(\alpha\pi x))(1-\cos(\alpha\pi y))(1-\cos(\alpha\pi z)). \tag{4.3}$$

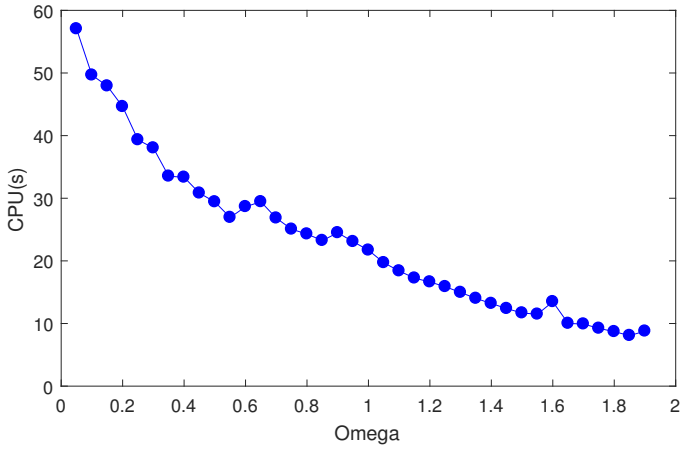


Figure 6: The CPU time of SSORCG for different choices of the factor ω.

Table 1: Rate-of-convergence and errors using new EXCMG with SSORCG smoother ($\omega=1.95$) for Problem 1 with $tol=10^{-10}$. The total computational time is 58.76 seconds, and the computational cost is 4.32 WU.

h	Iters[2]	$\|\|p_h-p\|\|_2$	Rate	$\|\|p_h-p\|\|_\infty$	Rate	$\|\|w_h-p_h\|\|_2$	Rate
1/32	352	8.96(−7)		8.06(−6)		2.87(−1)	
1/64	239	2.30(−7)	1.96	2.06(−6)	1.97	6.19(−8)	22.15
1/128	113	5.77(−8)	1.99	5.15(−7)	2.00	4.67(−9)	3.73
1/256	16	1.41(−8)	2.03	1.28(−7)	2.01	2.93(−10)	3.99
1/512	0	3.14(−9)	2.16	3.14(−8)	2.02	6.12(−15)	15.55

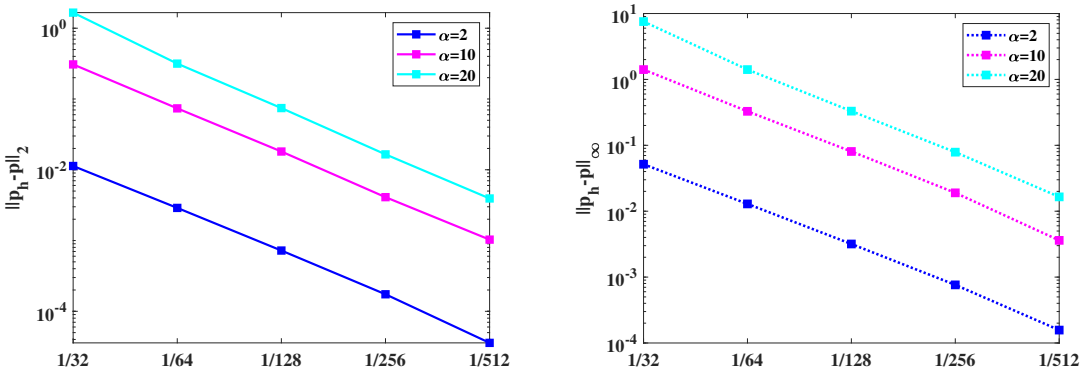
[1] WU (*work unit*) is the total computational cost of performing one SSORCG iteration on the finest grid. Here, the EXCMG computational cost $=0+16\times2^{-3}+113\times2^{-6}+239\times2^{-9}+352\times2^{-12}\approx4.32$. Other examples are calculated in a similar way.
[2] Iters denotes the quantity of smoothing steps of SSORCG on each level of grid.

Table 2: Rate-of-convergence and errors using old EXCMG with SSORCG smoother ($\omega=1.95$) for Problem 1 with $tol=10^{-10}$. The total computational time is 177.27 seconds, and the computational cost is 14.36 WU.

h	Iters	$\|\|p_h-p\|\|_2$	Rate	$\|\|p_h-p\|\|_\infty$	Rate	$\|\|w_h-p_h\|\|_2$	Rate
1/32	209	8.96(−7)		8.06(−6)		4.59(−6)	
1/64	279	2.30(−7)	1.96	2.06(−6)	1.97	5.48(−7)	3.06
1/128	129	5.77(−8)	1.99	5.15(−7)	2.00	6.66(−8)	3.04
1/256	30	1.43(−8)	2.01	1.29(−7)	2.00	8.19(−9)	3.02
1/512	8	3.45(−9)	2.05	3.35(−8)	1.95	1.01(−9)	3.02

Applying the biharmonic operator on the exact solution, we can obtain the forcing term

$$\begin{aligned} f(x,y,z)=&-(\alpha\pi)^4(\cos(\alpha\pi x)+\cos(\alpha\pi y)+\cos(\alpha\pi z)-4\cos(\alpha\pi x)\cos(\alpha\pi z)\\ &-4\cos(\alpha\pi y)\cos(\alpha\pi z)-4\cos(\alpha\pi x)\cos(\alpha\pi y)\\ &+9\cos(\alpha\pi x)\cos(\alpha\pi y)\cos(\alpha\pi z)). \end{aligned} \tag{4.4}$$



(a) (b)

Figure 7: The convergence of L_2-norm and L_∞-norm under different parameters α.

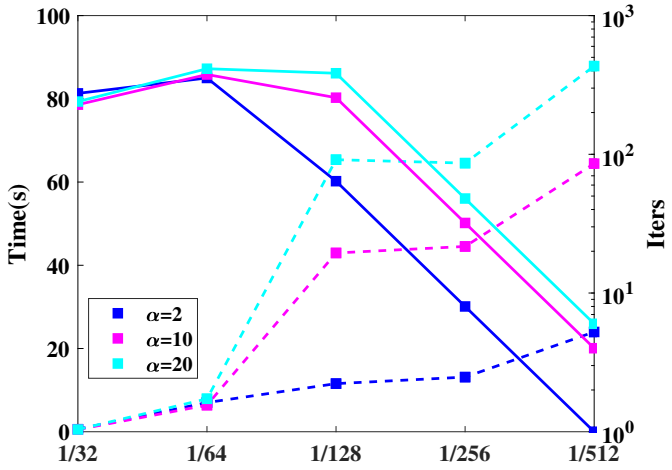


Figure 8: The iteration counts and computation costs under different parameters α: Iters are shown as solid line, and time as dashed line.

The parameter α take values of $\{2, 10, 20\}$. Fig. 7 illustrates the curves of the L_2-norm and L_∞-norm of the numerical solution p_h with different α, demonstrating second-order convergence in low- and high-frequency cases.

Fig. 8 depicts the iterations (solid lines) and computation times (dashed lines) for different α. Due to the coarsest level can not effectively resolve the oscillatory solutions, the number of iterations at each level increases as the parameters α increase in Fig. 8. On the finest grid $512\times512\times512$, only one iteration is required for low-frequency solution ($\alpha=2$), as well as only 6 iterations for $\alpha=20$. This confirms that effective initial guess significantly reduce the need for excessive iterations even in the cases of high oscillation,

Table 3: Rate-of-convergence and Errors using EXCMG with SSORCG smoother ($\omega=1.95$) for Problem 3 with $tol=10^{-10}$. The total solution time is 128.32 seconds, and the computational cost is 9.89 WU.

h	Iters	$\|\|p_h-p\|\|_2$	Rate	$\|\|p_h-p\|\|_\infty$	Rate	$\|\|w_h-p_h\|\|_2$	Rate
1/32	330	4.10(−6)		1.75(−5)		6.58(−2)	
1/64	342	1.05(−6)	1.96	4.36(−6)	2.00	1.21(−7)	19.05
1/128	257	2.66(−7)	1.98	1.09(−6)	2.00	8.33(−9)	3.86
1/256	41	6.70(−8)	1.99	2.73(−7)	2.00	5.03(−10)	4.04
1/512	0	1.69(−8)	1.98	6.80(−8)	2.00	1.06(−14)	15.53

Table 4: Rate-of-convergence and Errors using EXCMG with SSORCG smoother ($\omega=1.95$) for Problem 4 with $tol=10^{-12}$. The total solution time is 166.46 seconds, and the computational cost is 12.99 WU.

h	Iters	$\|\|p_h-p\|\|_2$	Rate	$\|\|p_h-p\|\|_\infty$	Rate	$\|\|w_h-p_h\|\|_2$	Rate
1/32	399	1.35(−6)		3.47(−6)		5.75(−2)	
1/64	457	3.47(−7)	1.96	8.69(−7)	2.00	3.30(−8)	20.73
1/128	256	8.77(−8)	1.98	2.17(−7)	2.00	2.34(−9)	3.81
1/256	32	2.22(−8)	1.99	5.44(−8)	1.99	1.30(−10)	4.16
1/512	4	5.56(−9)	1.99	1.37(−8)	1.99	4.75(−12)	4.77

thereby greatly reducing computational costs.

Two more examples are provided in the following. The rate-of-convergence and errors of EXCMG are listed in Table 3 and Table 4, respectively.

Problem 3. The exact solution of the test 3 can be written as

$$p(x,y,z)=\sinh(x)\sinh(y)\sinh(z). \tag{4.5}$$

And the forcing term $f(x,y,z)$ is given as

$$f(x,y,z)=\sinh(x)\sinh(y)\sinh(z). \tag{4.6}$$

Problem 4. The exact solution of the test 4 can be written as

$$p(x,y,z)=xyz\log(1+x+y+z). \tag{4.7}$$

And the forcing term $f(x,y,z)$ is given as follows

$$f(x,y,z)=\frac{-2(4x^3+8x^2+15xyz+4xy+4xz+4x+4y^3+8y^2+4yz+4y+4z^3+8z^2+4z)}{(x+y+z+1)^4}. \tag{4.8}$$

4.2 Comparisons with other multigrid solvers

Traditional multigrid methods involve three operators: interpolation, restriction and iteration, necessitating cycles between coarse and fine grids, e.g., V-cycle, W-cycle. In the application of classical multigrid methods, Altas [2,12] devised the full weighting restriction

Table 5: Computational time of EXCMG compared with AGMG for Problems 1 to 4 with $h=1/512$.

Problem	*tol*	AGMG		original EXCMG[3]		new EXCMG[4]	
		Iters[1]	CPU/s	Iters[2]	CPU/s	Iters	CPU/s
1	10^{-10}	302	3915.47	8	177.27	0	58.76
2 ($\alpha=2$)	10^{-4}	35	572.43	8	292.79	1	56.28
3	10^{-10}	214	2734.92	10	291.46	0	128.32
4	10^{-12}	277	3670.61	10	368.31	4	166.46

[1] Iters denotes the quantity of AMG preconditioned GCR(10) iterations required to decrease the relative residual less than the pre-set tolerance, and CPU time for AGMG includes the setup time.
[2] Iters denotes the quantity of smoothing steps of SSORCG for EXCMG on the finest grid.
[3] The original EXCMG method [17] developed for second-order elliptic boundary value problems.
[4] The new EXCMG method proposed in this paper.

operator and cubic interpolation operator for solving biharmonic equations. Although convergence is achieved, the computational cost is relatively high. Algebraic multigrid method is independent of the geometric information of specific problems [24], and is efficient for such symmetric positive definite systems (see [19]). Therefore, we conduct a comparison between the proposed EXCMG method, the original EXCMG method and the AGMG method for all examples with the same mesh size $h=1/512$. We can see from Table 5, the new EXCMG method takes fewer steps to converge and less computational time than the original EXCMG method and AGMG method, validating the superiority of the proposed EXCMG method.

The significant advantage of the EXCMG method over the AGMG method is its lower computational complexity. The multigrid method is $\mathcal{O}(nnz)$ (nnz represents the number of non-zero elements in the matrix) [19], whereas for EXCMG it is $\mathcal{O}(n_L)$ (n_L denotes the number of unknowns on level L) [25], with nnz being significantly greater than n_L. An analysis of the existing literature reveals that both multigrid and EXCMG are optimal under the L_2-norm [26]. However, in terms of the energy-norm, multigrid is merely optimal, whereas EXCMG achieves super-optimal. Therefore, the EXCMG method demonstrates higher computational efficiency. Although both the new and original EXCMG have similar computational complexity, the increase in efficiency is largely attributed to the application of higher-order interpolation techniques. This approach offers a more accurate initial guess, effectively reducing the number of iterations and significantly enhancing computational efficiency.

To optimize the performance of our algorithm, we replace the original SSOR-CG algorithm to its CUDA version, which was implemented with cuSPARSE library as well as parallel computing capabilities of GPU. From the Table 6, it is evident that using CUDA for parallel computing obtained about 5 to 6 times overall speedup on a GPU node with Nvidia Tesla V100 among original and new EXCMG. However, due to the limited GPU

Table 6: Computational time of EXCMG for Problems 1 to 4 with $h=1/256$.

Problem	*tol*	original EXCMG		new EXCMG	
		CPU/s	GPU/s	CPU/s	GPU/s
1	10^{-10}	42.71	8.91	13.39	4.52
2 ($\alpha=2$)	10^{-4}	94.56	16.21	34.85	7.96
3	10^{-10}	77.19	13.91	37.38	8.06
4	10^{-12}	121.46	19.67	52.91	10.49

memory (32G) on high-performance computing platform, the SSORCG solving process (3.8a)-(3.8b) consumes a significant amount of VRAM. Therefore, we can only solve problems with a maximum scale of 24 million degrees of freedom.

5 Conclusions and discussions

In this work, a new multigrid method EXCMG is developed to seek the solution of the large linear system resulting from the direct finite difference discritization of 3D biharmonic equations with two kinds of Dirichlet boundary conditions. The novelty of the method is that a new prolongation operator is designed, where quartic interpolations are used for the finite difference solutions on coarse and fine grids twice and once to obtain two approximations on the next finer grid, and the Richardson extrapolation is then used to obtain a good initial guess for the SSORCG smoother. Non-trivial examples are presented to show that the new EXCMG method offers 3 times speedup compared to the original EXCMG method, achieving a 60-fold acceleration over the AGMG method in simple problems, and nearly a 20-fold speedup in more complex cases. The EXCMG method with the new prolongation operator is proved to be efficient in solving 3D biharmonic equation.

Despite its advantages over traditional methods, the new EXCMG method requires solving several linear systems for the intermediate levels. Combining the multigrid variant with EXCMG to achieve intermediate-level solutions holds promise for further speedup our algorithm. This study only considers a relatively easy parallel EXCMG algorithm and show its efficiency and advantages. Developing highly scalable MPI-based parallel EXCMG algorithm is another goal. Additionally, the EXCMG method has been successfully applied to non-rectangular domains and semi-structured grids, such as distorted quadrilateral meshes. This method is also applicable to the biharmonic equation, which constructing new scheme at the boundary conditions presents significant challenges, and finding suitable prolongation operators in unstructured grids remains a considerable topic.

Acknowledgements

Kejia Pan was supported by the National Natural Science Foundation of China (No. 42274101) and the 173 Program of China (No. 2020-JCJQ-ZD-029). The authors are grateful for resources from the High Performance Computing Center of Central South University.

References

[1] M. M. GUPTA, AND R. MANOHAR, *Direct solution of biharmonic equation using noncoupled approach,* J. Comput. Phys., 33 (1979), pp. 236–248.
[2] I. ALTAS, J. DYM, M. M. GUPTA, AND R. MANOHAR, *Multigrid solution of automatically generated high order discretisation for the biharmonic equation,* SIAM J. Sci. Comput., 19 (1998), pp. 1575–1585.
[3] L. BAUER, AND E. L. REISS, *Block five diagonal matrices and the fast numerical solution of the biharmonic equation,* Math. Comput., 26 (1972), pp. 311–326.
[4] B. L. BUZBEE, AND F. W. DORR, *The direct solution of the biharmonic equation on rectangular regions and the Poisson equation on irregular regions,* SIAM J. Numer. Anal., 11 (1974), pp. 753–763.
[5] J. R. DOSSANTOS, *Équations aux différences finies pour l'équation biharmonique dans l'espace à trois dimensions,* C.R. Acad. Sci. Paris Sér. A-B, 264 (1967), pp. A291–A293.
[6] M. M. GUPTA, *Discretization error estimates for certain splitting procedures for solving first biharmonic boundary value problems,* SIAM J. Numer. Anal., 12 (1975), pp. 364–377.
[7] M. M. GUPTA, AND L. W. EHRLICH, *Some difference schemes for the biharmonic equation,* SIAM J. Numer. Anal., 12 (1975), pp. 773–790.
[8] M. DEHGHAN, AND A. MOHEBBI, *Multigrid solution of high order discretisation for three-dimensional biharmonic equation with Dirichlet boundary conditions of second kind,* Appl. Math. Comput., 180 (2006), pp. 575–593.
[9] N. A. GUMEROV, AND R. DURAISWAMI, *Fast multipole method for the biharmonic equation in three dimensions,* J. Comput. Phys., 215 (2006), pp. 363–383.
[10] L. J. T. DOSS, AND N. KOUSALYA, *Finite Pointset Method for biharmonic equations,* Appl. Math. Comput., 75 (2018), pp. 3756–3785.
[11] O. KARAKASHIAN, AND C. COLLINS, *Two-level additive Schwarz methods for discontinuous Galerkin approximations of the Biharmonic equation,* J. Sci. Comput., 74 (2018), pp. 573–604.
[12] I. ALTAS, J. ERHEL, AND M. M. GUPTA, *High accuracy solution of three-dimensional biharmonic equations,* Numer. Algorithm, 29 (2002), pp. 1–19.
[13] B. BIALECKI, *A fast solver for the orthogonal spline collocation solution of the biharmonic Dirichlet problem on rectangles,* J. Comput. Phys., 191 (2003), pp. 601–621.
[14] Q. ZHUANG, AND L. CHEN, *Legendre-Galerkin spectral-element method for the biharmonic equations and its applications,* Appl. Math. Comput., 74 (2017), pp. 2958–2968.
[15] B. P. LAMICHHANE, *A finite element method for a biharmonic equation based on gradient recovery operators,* BIT Numer. Math., 54 (2014), pp. 469–484.
[16] C. M. CHEN, H. L. HU, Z. Q. XIE, AND C. L. LI, *Analysis of extrapolation cascadic multigrid method (EXCMG),* Sci. China Ser. A-Math., 51 (2008), pp. 1349–1360.
[17] K. PAN, D. HE, H. HU, AND Z. REN, *A new extrapolation cascadic multigrid method for three dimensional elliptic boundary value problems,* J. Comput. Phys., 344 (2017), pp. 499–515.

[18] A. NAPOV, AND Y. NOTAY, *An algebraic multigrid method with guaranteed convergence rate,* SIAM J. Sci. Comput., 34(2) (2012), pp. A1079–A1109.
[19] VY. NOTAY, *An aggregation-based algebraic multigrid method,* Electron. T. Numer. Anal., 37(6) (2010), pp. 123–146.
[20] F. A. BORNEMANN, AND P. DEUFLHARD, *The cascadic multigrid method for elliptic problems,* Numer. Math., 75 (1996), pp. 135–152.
[21] G. I. MARCHUK, AND V. V. SHAIDUROV, Difference Methods and Their Extrapolations, Springer, NewYork (1983).
[22] S. HU, K. PAN, X. WU, Y. GE, AND Z. LI, *An efficient extrapolation multigrid method based on a HOC scheme on nonuniform rectilinear grids for solving 3D anisotropic convection-diffusion problems,* Comput. Methods Appl. Mech. Eng., 403 (2023), 115724.
[23] K. PAN, X. WU, H. HU, Y. YU, AND Z. LI, *A new FV scheme and fast cell-centered multigrid solver for 3D anisotropic diffusion equations with discontinuous coefficients,* J. Comput. Phys., 449 (2022), 110794.
[24] W. L. BRIGGS, V. E. HENSON, AND S. F. MCCORMICK, A Multigrid Tutorial, Philadelphia: Society for Industrial and Applied Mathematics, (2000).
[25] H. L. HU, Z. Y. REN, D. D. HE, AND K. J. PAN, *On the convergence of an extrapolation cascadic multigrid method for elliptic problems,* Comput. Math. Appl., 74 (2017), pp. 759–771.
[26] Y. Q. HUANG, Z. C. SHI, T. TANG, AND W. XUE, *A multilevel successive iteration method for nonlinear elliptic problems,* Math. Comput., 73 (2004), pp. 525–539.