East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.240222.210722

Vol. **13**, No. 1, pp. 119-139
February 2023

An Efficient EXCMG-Newton Method Combined with Fourth-Order Compact Schemes for Semilinear Poisson Equations

Pinxia Wu[1], Kejia Pan[1,*], Weiwei Ling[1] and Dongdong He[2]

[1]*School of Mathematics and Statistics, HNP-LAMA, Central South University, Changsha, Hunan 410083, China.*
[2]*School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, Guangdong 518172, China.*

Received 24 February 2022; Accepted (in revised version) 21 July 2022.

Abstract. A fast solver for nonlinear systems arising from fourth-order compact finite difference schemes for two-dimensional semilinear Poisson equations is constructed. Applying the extrapolation and bi-quartic interpolation to two numerical solutions from the previous two levels of grids, we determine a suitable initial guess for the Newton iterations on the next finer grid. It is fifth-order accurate, which substantially reduces the number of Newton iterations required. Moreover, an extrapolated solution of sixth-order accuracy can be easily constructed on the whole fine grid. Numerical results suggest that the method is much more efficient than the existing multigrid methods for semilinear problems.

AMS subject classifications: 65N06, 65N55

Key words: Semilinear Poisson equation, fourth-order compact scheme, EXCMG-Newton method, high efficiency, bi-quartic interpolation.

1. Introduction

Semilinear Poisson equations appear in various fields, including fluid mechanics and geophysics. In this work, we consider a fast solver, which allows us to efficiently obtain numerical solutions of two-dimensional (2D) Poisson equations with nonlinear forcing terms. These equations have the form

$$\begin{aligned} &u_{xx}+u_{yy}=f(u,x,y), \quad (x,y)\in\Omega,\\ &u(x,y)=g(x,y), \qquad\quad (x,y)\in\partial\Omega, \end{aligned} \tag{1.1}$$

where Ω is a rectangle domain with the boundary $\partial\Omega$. The Dirichlet boundary condition is imposed on $\partial\Omega$. Besides, the nonlinear forcing function $f(u,x,y)$, the boundary function

*Corresponding author. *Email address:* kejiapan@csu.edu.cn (K.J. Pan)

$g(x,y)$ and the true solution $u(x,y)$ are supposed to be continuously differentiable and have indispensable partial derivatives. Following [19, 33], we assume that the problem (1.1) has a unique solution.

High-order compact finite difference (FD) schemes for Poisson equation have been widely studied [17, 19, 30, 34, 35]. The methods are called compact because the corresponding discretization formulas use a minimal number of mesh points in order to achieve the fourth-order accuracy. However, for large-scale problems, general iteration solvers are quite time-consuming. The multigrid method [3] is a very efficient strategy for solving large sparse systems of linear equations arising in these discretizations. Therefore, the combinations of multigrid methods and high-order compact FD schemes have been also developed [11, 13, 30, 35]. The classical multigrid technique has been applied to other equations — e.g. to convection-diffusion equation [12, 14, 31], the biharmonic equation [1] and the Helmholtz equation [10, 28].

The CMG technique developed by Deuflhard and Bornemann [2], represents an one-way multigrid method without any correlation between fine and coarse grids. The method initially used a conjugate gradient (CG) solver as the relax operator on the embedded grids, whereas the initial values of the smoother on the current grids have been approximated by the linear interpolation on the previous grids. For the energy norm, Bornemann and Deuflhard [2] showed that this is an optimal iterative method. We note that the Richardson extrapolation often used to increase the accuracy of numerical solutions, have been initially employed the coarse grids only. In 1993, Roache and Knupp [25] generalized the strategy and obtained extrapolated solutions at the middle of the fine grid points. This is similar to the mid-point extrapolation formula proposed by Chen and Lin [5].

In the past decade, a CMG method has been combined with extrapolation strategies. Thus Chen *et al.* [4] developed an EXCMG method for fast solution of second-order elliptic problems. Besides, an EXCMG method employing high-order compact FD schemes for 2D Poisson equations have been studied in [6, 15, 21]. Pan *et al.* [22] applied the EXCMG method to 3D elliptic boundary value problems. In order to reduce computational time, Dai *et al.* [7] developed an approximation method, where EXCMG has been used for finding a better initial guess in the MSMG method. Recently, Pan *et al.* [23, 24] applied a fast cell-centered EXCMG (CEXCMG) algorithm based on finite volume (FV) discretization to 2D/3D anisotropic diffusion equations with discontinuous coefficients.

Although for linear problems the EXCMG methods are thoroughly studied, there are only a few works devoted to nonlinear problems. As far as the MG method is concerned, the investigations are mainly focused on Newton-MG methods [3], adaptive MG methods [32] and CMG methods [27, 29, 33]. In particular, for solving large nonlinear systems arising in the fourth-order compact FD discretization of the 2D semilinear Poisson equation, Li and Pan [16] proposed a Newton-MSMG method such that the corresponding extrapolation solutions can achieve the sixth-order approximation accuracy, which is much greater than the discretization-level.

The main purpose of this work is to construct and analyze a fast EXCMG-Newton method for the nonlinear system arising in fourth-order compact FD schemes for the 2D semilinear Poisson equation. Using the extrapolation and bi-quartic polynomial interpolation on

previous two grids, we determine an initial guess on the fine grid, which accelerates the convergence of the Newton solver. After that, we use the midpoint extrapolation formula and construct cheaply a sixth-order accurate extrapolated solution from two fourth-order FD solutions. Besides, at each Newton iteration the corresponding large-scale Jacobian system is solved by the BiCGStab solver [20,26] combined with the symmetric successive over relaxation preconditioner (SSOR) [26]. The preconditioning matrix of SSOR-BiCGStab can be immediately obtained from the coefficient matrix with no extra computational cost. It is also important for solving large computational problems. Moreover, tolerances related to Newton and SSOR-BiCGStab iterations are applied to get numerical solutions with the required accuracy. Numerical experiments show that in terms of the convergence order this method is about six times more efficient than the existing Newton-MG method and about three times more efficient than the Newton-MSMG method.

This paper is organized as follows. Section 2 presents high-order compact FD schemes for 2D semilinear Poisson equations. A detailed description of the EXCMG-Newton method is given in Section 3. Section 4 presents the results of numerical simulations, which shows the accuracy and efficiency of the method. Retrospective comments are given in the final section.

2. High-Order Compact Schemes

Consider the rectangular solution domain $\Omega = [L_{lx}, L_{rx}] \times [L_{ly}, L_{ry}]$ and the uniform grid points (x_i, y_j), $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$, where

$$x_i = L_{lx} + ih_x, \quad y_j = L_{ly} + jh_y,$$

h_x and h_y are respective mesh sizes in x- and y-directions, and

$$h_x = \frac{L_{rx} - L_{lx}}{N_x}, \quad h_y = \frac{L_{ry} - L_{ly}}{N_y}.$$

By $U_{i,j}$ we denote an approximate value of the solution u at the point (x_i, y_j). Setting $F_{i,j} = f(U_{i,j}, x_i, y_j)$ and $\gamma = h_x/h_y$, we consider the following 9-point fourth-order compact schemes for the model (1.1):

(A) (See Zhai *et al.* [34]).

$$\begin{aligned}
&\left(5-\gamma^2\right)\left(U_{i+1,j}+U_{i-1,j}\right)+\left(5\gamma^2-1\right)\left(U_{i,j+1}+U_{i,j-1}\right)-10\left(1+\gamma^2\right)U_{i,j}\\
&\quad+\frac{1+\gamma^2}{2}\left(U_{i+1,j+1}+U_{i+1,j-1}+U_{i-1,j+1}+U_{i-1,j-1}\right)\\
&=\frac{h_x^2}{2}\left[\frac{1}{12}\left(F_{i+1,j+1}+F_{i-1,j+1}+F_{i+1,j-1}+F_{i-1,j-1}\right)\right.\\
&\qquad\left.+\frac{5}{6}\left(F_{i+1,j}+F_{i-1,j}+F_{i,j+1}+F_{i,j-1}\right)+\frac{25}{3}F_{i,j}\right].
\end{aligned}$$

(B) (See Manohar & Stephenson [17], Zhang [35]).

$$\begin{aligned}&\left(5-\gamma^2\right)\left(U_{i+1,j}+U_{i-1,j}\right)+\left(5\gamma^2-1\right)\left(U_{i,j+1}+U_{i,j-1}\right)-10\left(1+\gamma^2\right)U_{i,j}\\&\quad+\frac{1+\gamma^2}{2}\left(U_{i+1,j+1}+U_{i+1,j-1}+U_{i-1,j+1}+U_{i-1,j-1}\right)\\&=\frac{h_x^2}{2}\left[F_{i+1,j}+F_{i-1,j}+F_{i,j+1}+F_{i,j-1}+8F_{i,j}\right].\end{aligned}$$

(C) (See Manohar & Stephenson [17]).

$$\begin{aligned}&\left(5-\gamma^2\right)\left(U_{i+1,j}+U_{i-1,j}\right)+\left(5\gamma^2-1\right)\left(U_{i,j+1}+U_{i,j-1}\right)-10\left(1+\gamma^2\right)U_{i,j}\\&\quad+\frac{1+\gamma^2}{2}\left(U_{i+1,j+1}+U_{i+1,j-1}+U_{i-1,j+1}+U_{i-1,j-1}\right)\\&=\frac{h_x^2}{2}\left[\frac{1}{2}F_{i+1,j+1}+\frac{1}{2}F_{i-1,j+1}+\frac{1}{2}F_{i+1,j-1}+\frac{1}{2}F_{i-1,j-1}+10F_{i,j}\right].\end{aligned}$$

(D) (See Zhai *et al.* [34]).

$$\begin{aligned}&\left(5-\gamma^2\right)\left(U_{i+1,j}+U_{i-1,j}\right)+\left(5\gamma^2-1\right)\left(U_{i,j+1}+U_{i,j-1}\right)-10\left(1+\gamma^2\right)U_{i,j}\\&\quad+\frac{1+\gamma^2}{2}\left(U_{i+1,j+1}+U_{i+1,j-1}+U_{i-1,j+1}+U_{i-1,j-1}\right)\\&=\frac{h_x^2}{6}\left[F_{i+1,j+1}+F_{i-1,j+1}+F_{i+1,j-1}+F_{i-1,j-1}+F_{i+1,j}+F_{i-1,j}+F_{i,j+1}+F_{i,j-1}+28F_{i,j}\right].\end{aligned}$$

Letting $h=\max\{h_x,h_y\}$, we denote by u_h the approximate solutions $U_{i,j}$ of the model (1.1), which are obtained by a FD method on the grid Ω_h with the mesh sizes h_x and h_y. Correspondingly, $u_{h/2}$ is the FD solution vector on the grid $\Omega_{h/2}$ with mesh sizes $h_x/2$ and $h_y/2$. Adding the Dirichlet boundary conditions, we write the above difference schemes as

$$N_h(u_h)=0, \tag{2.1}$$

where $N_h(u_h)$ is the corresponding nonlinear operator.

3. EXCMG-Newton Method

Since at each iteration step the Newton method uses the solutions of the Jacobian systems, this is a very time consuming process, especially for large systems. On the other hand, the extrapolation cascadic multigrid (EXCMG) technique is a powerful and efficient solver. Therefore, employing the EXCMG technique and a nonlinear solver — viz. the Newton method, we introduce a faster and more efficient algorithm for solving problem (2.1). Let us start with recalling the Newton method.

3.1. Classical Newton method

There is no doubt that the Newton's method is one of the most important techniques in nonlinear problems [8,9]. For high-order compact schemes, the nonlinear operator $N(\cdot)$ and the residual on the grid h can be respectively written as

$$N_h(u_h) = Au_h - f(u_h),$$

and

$$r_h(u_h) = 0 - N_h(u_h).$$

In order to find the solution of the Eq. (2.1), one can use the k-th Newton iterations

$$u_h^{k+1} = u_h^k + e_h^k, \quad k = 0, 1, 2, \ldots$$

with e_h^k obtained from the linear system

$$\mathbb{J}\left(u_n^k\right) e_h^k = r_h\left(u_h^k\right),$$

where $\mathbb{J}(u_n^k) = N_h'(u_n^k)$ is the Jacobian matrix and $r_h(u_h^k) = -N_h(u_n^k)$. Note that the linear Jacobian systems have to be accurately calculated at each Newton iteration. The Newton method is presented by Algorithm 3.1.

Algorithm 3.1 Newton Method for Equation $N(u) = 0$.

1: Set ϵ, $iter$ and u_0.
2: **for** $k = 0, 1, \ldots, iter$ **do**
3: Calculate Jacobian matrix $N'(u_k)$.
4: Solve Jacobian system $N'(u_k)\tilde{s}_k = -N(u_k)$ to obtain $\tilde{s}_k$.
5: **if** $\|\tilde{s}_k\| \geq \varepsilon_{non}$ **then**
6: update $u_{k+1} \Leftarrow u_k + \tilde{s}_k$
7: **else**
8: stop
9: **end if**
10: **end for**

After completing the k-th iteration, iterative solution u_k of the method satisfies the estimate

$$\|u_k - u_*\| \leq \vartheta^k \|u_0 - u_*\|,$$

where u_* is the true solution, u_0 the initial guess, and $\vartheta \in (0, 1)$ a convergence factor. It is well known that the Newton method converges if and only if the initial guess is near to the problem solution. Therefore, if the initial error $\|u_0 - u_*\|$ is small, the Newton method converges very fast. We follow the idea of the EXCMG method in order to determine a suitable initial guess for Newton iterations at every grid level.

3.2. EXCMG-Newton method

Here we present an EXCMG-Newton method for large nonlinear systems arising in the approximate solution of 2D semilinear Poisson equations. It mainly consists of three ingredients — viz. the Newton solver, the extrapolation and quadratic interpolation. These techniques are employed to find an accurate initial guess for the Newton solver on each grid. Besides, the Richardson extrapolation is used to construct approximate solutions with sixth-order accuracy from two fourth-order accuracy solutions. For more detail see Algorithm 3.2.

Algorithm 3.2 EXCMG-Newton Method: $(u_h, w_h) \Leftarrow$ EXCMG-Newton $(A_h, f_h, L, \epsilon_{lin}, \epsilon_{non})$.

1: $u_H \Leftarrow \text{Newton}(A_H u_H = f_H)$;
2: $u_{H/2} \Leftarrow \text{Newton}(A_{H/2} u_{H/2} = f_{H/2})$;
3: $h = H/2$;
4: **for** $j = 1$ to L **do**
5: $h = h/2$;
6: $\epsilon_{lin}^j = \epsilon_{lin} \cdot 10^{j-L}$ % ϵ_{lin} is the relative residual tolerance of SSOR-BiCGStab on the finest grid;
7: $\epsilon_{non}^j = \epsilon_{non} \cdot 10^{j-L}$ % ϵ_{non} is the nonlinear residual tolerance of Newton iteration on the finest grid;
8: $\bar{u}_h = \text{EXP}_{finite}(u_{2h}, u_{4h})$ % $\bar{u}_h$ is used as the initial guess for Newton method;
9: $u_h \Leftarrow \text{Newton}(A_h, \bar{u}_h, f_h, \epsilon_{lin}^j, \epsilon_{non}^j)$ % Jacobian systems are solved by SSOR-BiCGStab;
10: **end for**
11: $w_h = \text{EXP}_{true}(u_h, u_{2h})$ % w_h is a higher-order extrapolated solution.

We note that since the first two coarse levels of the grid are so small that the computation cost is negligible, the Newton's method can be employed in order to accurately solve the problem — cf. lines 1 and 2. The initial guess $\text{EXP}_{finite}(u_{2h}, u_{4h})$ is a fifth-order accuracy approximation for the FD solution u_h obtained by the extrapolation and quartic interpolation from the numerical solutions on the adjacent grids u_{2h} and u_{4h}. It provides a more accurate initial guess for the Newton smoother on the current grid. The term $\text{EXP}_{true}(u_h, u_{2h})$ represents the sixth-order extrapolated solution obtained from the fourth-order numerical solutions u_h and u_{2h}. A detailed description of the extrapolation and quartic interpolation on the nested quadrilateral mesh is presented in Section 3.3. The positive integer L denotes the total level number of the grid except the first two nested grids, and $H/2^{L+1}$ is the size of the finest mesh.

The linear Jacobian systems — cf. line 4 in Algorithm 3.1 in Newton method — cf. line 9 in Algorithm 3.2, are solved by the symmetric successive overrelaxation preconditioned BiCGStab (SSOR-BiCGStab) solver. The SSOR preconditioner can be acquired immediately from the coefficient matrix without any extra computing time. Assuming that the linear Jacobian system has the form of $Bx = l$, we can use the preconditioner M defined by

$$M = (D - \omega E) D^{-1} (D - \omega F),$$

where $-E$, $-F$, and D are respectively the strict lower part, the strict upper part, and the diagonal of B. The preconditioned BiCGStab algorithm has the following form:

Algorithm 3.3 Preconditioned BiCGStab Method: $x \Leftarrow \text{BiCGSTAB}(B, l, x_0, iter_{\max}, \varepsilon_{lin}, M)$.

1: $r_0 = l - Bx_0$;
2: $\hat{r_0} = r_0$;
3: $\rho_0 = \alpha = \omega_0 = 1$;
4: $v_0 = p_0 = 0$;
5: **for** $i = 1$ to $iter_{\max}$ **do**
6: $\quad \rho_i = (\hat{r_0}, r_{i-1})$;
7: $\quad \beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1})$;
8: $\quad p_i = r_{i-1} + \beta(p_{i-1} - \omega_{i-1}v_{i-1})$;
9: $\quad$ Solve $My = p_i$;
10: $\quad v_i = By$;
11: $\quad \alpha = \rho_i/(\hat{r_0}, v_i)$;
12: $\quad s = r_{i-1} - \alpha v_i$;
13: $\quad$ Solve $Mz = s$;
14: $\quad t = Bz$;
15: $\quad \omega_i = (t, s)/(t, t)$;
16: $\quad x_i = x_{i-1} + \alpha y + \omega_i z$;
17: $\quad r_i = s - \omega_i t$;
18: $\quad$ **if** $\|r_i\|_2 < \varepsilon_{lin} \cdot \|l\|_2$ **then**
19: $\quad\quad$ break
20: $\quad$ **end if**
21: **end for**
22: $x = x_i$.

At any j-th level of the grid, the SSOR-BiCGStab solver in Algorithm 3.2 has the relative residual tolerance ϵ_{lin}^j, cf. line 6 in Algorithm 3.2. It is defined by

$$\epsilon_{lin}^j = \epsilon_{lin} \cdot 10^{j-L}, \quad j = 1, 2, \ldots, L,$$

where $\epsilon_{lin} = \epsilon_{lin}^L$ refers to the tolerance of the SSOR-BiCGStab solver on the finest grid. Besides,

$$\epsilon_{non}^j = \epsilon_{non} \cdot 10^{j-L}, \quad j = 1, 2, \ldots, L,$$

where $\epsilon_{non} = \epsilon_{non}^L$ is the nonlinear tolerance on the finest grid, is also used in the Newton method at the j-th level of the grid — cf. line 7 in Algorithm 3.2. These stopping criteria help to avoid unnecessary iterations on each level of grids, and obtain numerical solutions of a desired accuracy.

3.3. Extrapolation and bi-quartic interpolation

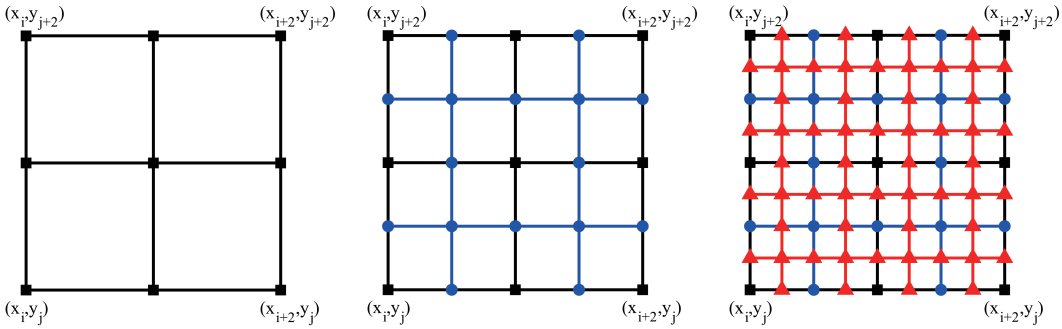
It is well known that the Richardson extrapolation is an efficient algorithm for improving the accuracy of numerical solutions. As far as FD schemes are concerned, it was systematically studied by Marchuk and Shaidurov [18].

3.3.1. Extrapolation of FD solutions

Consider the two dimensional three-level nested grids Z_i, $i = 0, 1, 2$ with mesh sizes $h_i = h_x = h_y = h/2^i$. The coarse grid block $[x_i, x_{i+2}] \times [y_j, y_{j+2}]$ is uniformly divided into eight elements in the x and y directions — cf. Fig. 1(c). It follows that one can allocate 81 points on the finest grid. Using the FD solutions u^0 and u^1 on respective grids Z_0 and Z_1, we can obtain approximate values at 81 points of the finest grid by the extrapolation and quartic Lagrange interpolation. Let us first define coarse mesh blocks $[x_i, x_{i+2}] \times [y_j, y_{j+2}]$ on the coarse grid Ω_h, cf. Fig. 1(a), which can acquire the set of 25 grid points

$$
\begin{aligned}
&(x_m, y_n), (x_{i+1/2}, y_n), (x_{i+3/2}, y_n), (x_m, y_{j+1/2}), (x_m, y_{j+3/2}),\\
&(x_{i+1/2}, y_{j+1/2}), (x_{i+1/2}, y_{j+3/2}), (x_{i+3/2}, y_{j+1/2}), (x_{i+3/2}, y_{j+3/2}),\\
&m = i, i+1, i+2, \quad n = j, j+1, j+2,
\end{aligned}
$$

on the finer grid $\Omega_{h/2}$, cf. Fig. 1(b).



(a) Coarsest grid Z_0 (b) Finer grid Z_1 (c) Finest grid Z_2

Figure 1: Three-level nested rectangular grids.

Suppose that the values

$$
\begin{aligned}
&u^h_{m,n}, \quad m = i, i+1, i+2, \quad n = j, j+1, j+2 \quad \text{on } \Omega_h,\\
&u^{h/2}_{m,n}, u^{h/2}_{i+1/2,n}, u^{h/2}_{i+3/2,n}, u^{h/2}_{m,j+1/2}, u^{h/2}_{m,j+3/2}, u^{h/2}_{i+1/2,j+1/2}, u^{h/2}_{i+1/2,j+3/2}, u^{h/2}_{i+3/2,j+1/2}, u^{h/2}_{i+3/2,j+3/2},\\
&m = i, i+1, i+2, \quad n = j, j+1, j+2 \qquad \text{on } \Omega_{h/2}
\end{aligned}
$$

have already been determined. Then the approximations of $\bar{u}^{h/4}$ on the finest grid are determined as follows:

1. **At corner grid points ■.** Approximate values at the 9 corner grid points are calculated by applying the extrapolation formula

$$
\bar{u}^{h/4}_{m,n} = \frac{17}{16} u^{h/2}_{m,n} - \frac{1}{16} u^h_{m,n}, \quad m = i, i+1, i+2, \quad n = j, j+1, j+2.
$$

2. **At midpoints of edges** •. Approximate values at the 12 midpoints of the edges can be calculated using the midpoint extrapolation formula

$$\begin{aligned}
\bar{u}^{h/4}_{i+1/2,n} &= u^{h/2}_{i+1/2,n} + \frac{1}{32}\Big[u^{h/2}_{i,n} - u^{h}_{i,n} + u^{h/2}_{i+1,n} - u^{h}_{i+1,n}\Big], && n = j, j+1, j+2,\\
\bar{u}^{h/4}_{i+3/2,n} &= u^{h/2}_{i+3/2,n} + \frac{1}{32}\Big[u^{h/2}_{i+1,n} - u^{h}_{i+1,n} + u^{h/2}_{i+2,n} - u^{h}_{i+2,n}\Big], && n = j, j+1, j+2,\\
\bar{u}^{h/4}_{m,j+1/2} &= u^{h/2}_{m,j+1/2} + \frac{1}{32}\Big[u^{h/2}_{m,j} - u^{h}_{m,j} + u^{h/2}_{m,j+1} - u^{h}_{m,j+1}\Big], && m = i, i+1, i+2,\\
\bar{u}^{h/4}_{m,j+3/2} &= u^{h/2}_{m,j+3/2} + \frac{1}{32}\Big[u^{h/2}_{m,j+1} - u^{h}_{m,j+1} + u^{h/2}_{m,j+2} - u^{h}_{m,j+2}\Big], && m = i, i+1, i+2.
\end{aligned}$$

3. **At center points of faces** •. Since each center point on the uniform grid can be regarded as the intersection of two diagonals, we can find two values at the center point by the midpoint extrapolation formula. The final extrapolation value can be chosen as their arithmetic average. For the grid point $(i+1/2, j+1/2)$, the two different extrapolation values can be acquired by the midpoint extrapolation formula

$$\begin{aligned}
\bar{u}^{h/4,1}_{i+1/2,j+1/2} &= u^{h/2}_{i+1/2,j+1/2} + \frac{1}{32}\Big[u^{h/2}_{i,j} - u^{h}_{i,j} + u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1}\Big],\\
\bar{u}^{h/4,2}_{i+1/2,j+1/2} &= u^{h/2}_{i+1/2,j+1/2} + \frac{1}{32}\Big[u^{h/2}_{i,j+1} - u^{h}_{i,j+1} + u^{h/2}_{i+1,j} - u^{h}_{i+1,j}\Big].
\end{aligned}$$

Thus the final extrapolation value at $(i+1/2, j+1/2)$ can be chosen as follows:

$$\begin{aligned}
\bar{u}^{h/4}_{i+1/2,j+1/2} &= \frac{\bar{u}^{h/4,1}_{i+1/2,j+1/2} + \bar{u}^{h/4,2}_{i+1/2,j+1/2}}{2}\\
&= u^{h/2}_{i+1/2,j+1/2} + \frac{1}{64}\Big[u^{h/2}_{i,j} - u^{h}_{i,j} + u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1}\Big]\\
&\quad + \frac{1}{64}\Big[u^{h/2}_{i,j+1} - u^{h}_{i,j+1} + u^{h/2}_{i+1,j} - u^{h}_{i+1,j}\Big].
\end{aligned}$$

Analogously, the corresponding approximations at the other center points of faces can be determined by the formulas

$$\begin{aligned}
\bar{u}^{h/4}_{i+3/2,j+1/2} &= u^{h/4}_{i+3/2,j+1/2} + \frac{1}{64}\Big[u^{h/2}_{i+1,j} - u^{h}_{i+1,j} + u^{h/2}_{i+2,j+1} - u^{h}_{i+2,j+1}\Big]\\
&\quad + \frac{1}{64}\Big[u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1} + u^{h/2}_{i+2,j} - u^{h}_{i+2,j}\Big],\\
\bar{u}^{h/4}_{i+1/2,j+3/2} &= u^{h/4}_{i+1/2,j+3/2} + \frac{1}{64}\Big[u^{h/2}_{i,j+1} - u^{h}_{i,j+1} + u^{h/2}_{i+1,j+2} - u^{h}_{i+1,j+2}\Big]\\
&\quad + \frac{1}{64}\Big[u^{h/2}_{i,j+2} - u^{h}_{i,j+2} + u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1}\Big],\\
\bar{u}^{h/4}_{i+3/2,j+3/2} &= u^{h/4}_{i+3/2,j+3/2} + \frac{1}{64}\Big[u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1} + u^{h/2}_{i+2,j+2} - u^{h}_{i+2,j+2}\Big]\\
&\quad + \frac{1}{64}\Big[u^{h/2}_{i+1,j+2} - u^{h}_{i+1,j+2} + u^{h/2}_{i+2,j+1} - u^{h}_{i+2,j+1}\Big].
\end{aligned}$$

4. **At four of equal division points ▲**. Once the extrapolation values at the points ■ • on the finest grid $\Omega_{h/4}$ are obtained, then the approximate values at the remaining points (9^2-5^2) can be determined by bi-quartic Lagrange interpolation — e.g.

$$\bar{u}^{h/4}_{i+1/4,j}=\frac{1}{128}\Big[35\bar{u}^{h/4}_{i,j}+140\bar{u}^{h/4}_{i+1/2,j}-70\bar{u}^{h/4}_{i+1,j}+28\bar{u}^{h/4}_{i+3/2,j}-5\bar{u}^{h/4}_{i+2,j}\Big].$$

This process of finding the initial values $\bar{u}^{h/4}$ on the grid $\Omega_{h/4}$ from the respective values on the grids Ω_h and $\Omega_{h/2}$ can be shortly written as

$$\bar{u}^{h/4}=\mathrm{EXP}_{finite}(u^h,u^{h/2}).$$

3.3.2. Extrapolation for true solution

The extrapolation algorithm is a reliable tool for enhancing the accuracy of FD solutions at each level of the grid. In this subsection, the Richardson extrapolation algorithm is used in order to improve numerical accuracy of a compact FD scheme from fourth- to sixth-order. The results are verified by the numerical simulations in the Section 4.

Suppose that the grid point values

$$\begin{aligned}
&u^h_{m,n},\quad m=i,i+1,i+2,\quad n=j,j+1,j+2\quad \text{on } \Omega_h,\\
&u^{h/2}_{m,n},\ u^{h/2}_{i+1/2,n},\ u^{h/2}_{i+3/2,n},\ u^{h/2}_{m,j+1/2},\ u^{h/2}_{m,j+3/2},\ u^{h/2}_{i+1/2,j+1/2},\ u^{h/2}_{i+1/2,j+3/2},\ u^{h/2}_{i+3/2,j+1/2},\ u^{h/2}_{i+3/2,j+3/2}\\
&m=i,i+1,i+2,\quad n=j,j+1,j+2\qquad\qquad \text{on } \Omega_{h/2}
\end{aligned}$$

at the two levels of the nested grids have already been obtained. Then the approximations of $w^{h/2}$ at the fine grid are determined as follows:

1. **At corner grid points**. High-order extrapolated solutions at the corner grid points are obtained by the Richardson extrapolation formula

$$w^{h/2}_{m,n}=\frac{16}{15}u^{h/2}_{m,n}-\frac{1}{15}u^h_{m,n},\quad m=i,i+1,i+2,\quad n=j,j+1,j+2. \tag{3.1}$$

2. **At edges midpoints of the fine grid**. The extrapolated solutions at the edges midpoints of the fine grid are determined by the following midpoint extrapolation formula:

$$\begin{aligned}
&w^{h/2}_{i+1/2,n}=u^{h/2}_{i+1/2,n}+\frac{1}{30}\Big[u^{h/2}_{i,n}-u^h_{i,n}+u^{h/2}_{i+1,n}-u^h_{i+1,n}\Big],\\
&n=j,j+1,j+2, &\text{(3.2a)}\\
&w^{h/2}_{i+3/2,n}=u^{h/2}_{i+3/2,n}+\frac{1}{30}\Big[u^{h/2}_{i+1,n}-u^h_{i+1,n}+u^{h/2}_{i+2,n}-u^h_{i+2,n}\Big],\\
&n=j,j+1,j+2, &\text{(3.2b)}\\
&w^{h/2}_{m,j+1/2}=u^{h/2}_{m,j+c1/2}+\frac{1}{30}\Big[u^{h/2}_{m,j}-u^h_{m,j}+u^{h/2}_{m,j+1}-u^h_{m,j+1}\Big],
\end{aligned}$$

$$m = i, i+1, i+2, \tag{3.2c}$$

$$w^{h/2}_{m,j+3/2} = u^{h/2}_{m,j+3/2} + \frac{1}{30}\Big[u^{h/2}_{m,j+1} - u^{h}_{m,j+1} + u^{h/2}_{m,j+2} - u^{h}_{m,j+2}\Big],$$

$$m = i, i+1, i+2. \tag{3.2d}$$

3. **At faces center points of the fine grid**. The extrapolated solutions at the faces center points of the fine grid are determined by the following extrapolation formula:

$$\begin{aligned}
w^{h/2}_{i+1/2,j+1/2} &= u^{h/2}_{i+1/2,j+1/2} + \frac{1}{60}\Big[u^{h/2}_{i,j} - u^{h}_{i,j} + u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1}\Big] \\
&\quad + \frac{1}{60}\Big[u^{h/2}_{i,j+1} - u^{h}_{i,j+1} + u^{h/2}_{i+1,j} - u^{h}_{i+1,j}\Big], \\
w^{h/2}_{i+3/2,j+1/2} &= u^{h/2}_{i+3/2,j+c1/2} + \frac{1}{60}\Big[u^{h/2}_{i+1,j} - u^{h}_{i+1,j} + u^{h/2}_{i+2,j+1} - u^{h}_{i+2,j+1}\Big] \\
&\quad + \frac{1}{60}\Big[u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1} + u^{h/2}_{i+2,j} - u^{h}_{i+2,j}\Big], \\
w^{h/2}_{i+1/2,j+3/2} &= u^{h/2}_{i+1/2,j+3/2} + \frac{1}{60}\Big[u^{h/2}_{i,j+1} - u^{h}_{i,j+1} + u^{h/2}_{i+1,j+2} - u^{h}_{i+1,j+2}\Big] \\
&\quad + \frac{1}{60}\Big[u^{h/2}_{i,j+2} - u^{h}_{i,j+2} + u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1}\Big], \\
w^{h/2}_{i+3/2,j+3/2} &= u^{h/2}_{i+\frac{3}{2},j+3/2} + \frac{1}{60}\Big[u^{h/2}_{i+1,j+1} - u^{h}_{i+1,j+1} + u^{h/2}_{i+2,j+2} - u^{h}_{i+2,j+2}\Big] \\
&\quad + \frac{1}{60}\Big[u^{h/2}_{i+1,j+2} - u^{h}_{i+1,j+2} + u^{h/2}_{i+2,j+1} - u^{h}_{i+2,j+1}\Big].
\end{aligned} \tag{3.3}$$

High-order extrapolated solution $w^{h/2}$ on the grid $\Omega_{h/2}$ can be easily determined by the extrapolation formulas (3.1)-(3.3). This procedure is shortly written as

$$w^{h/2} = \mathrm{EXP}_{true}(u^h, u^{h/2}).$$

4. Numerical Experiments

Now we want to demonstrate the robustness and efficiency of the above EXCMG-Newton method for the 2D nonlinear Poisson equations by comparing it with classical Newton-MG and MSMG methods. Note that our method is quite straightforward and can be easily implemented in programming language. Moreover, nonlinear forcing terms and Dirichlet boundary conditions are given to fulfill corresponding analytical solutions. All codes are run on a laptop with Intel(R) Core(TM) i7-1065G7 and 16GB RAM by using Matlab R2021a.

Note that the maximum number of iterations in SSOR-BiCGStab is set to 40 and the relaxation factor ω is 1.9. Besides, for the EXCMG-Newton method the convergence order is defined by

$$\mathrm{Order} = \log_2 \frac{\|\mathrm{u_h} - \mathrm{u}\|}{\|\mathrm{u_{h/2}} - \mathrm{u}\|},$$

where $\|\cdot\|$ is the L^2 or L^∞ norm, u denotes the exact solution and u_h and $u_{h/2}$ are the numerical solutions on the mesh sizes h and $h/2$, respectively.

Example 4.1 (cf. Li *et al.* [16])**.** Consider the nonlinear Poisson-Boltzmann model

$$\begin{aligned} &u_{xx}(x,y)+u_{yy}(x,y)-\lambda^2\sinh\big(u(x,y)\big)=g(x,y),\\ &(x,y)\in\Omega=[0,1]\times[0,1]. \end{aligned} \tag{4.1}$$

For the source term

$$\begin{aligned} g(x,y)=&\,2000\exp\left(-1000\left(x-\frac{1}{2}\right)^2\right)+(1000-2000x)^2\exp\left(-1000\left(x-\frac{1}{2}\right)^2\right)\\ &-2+\lambda^2\sinh\left(-\exp\left(-1000\left(x-\frac{1}{2}\right)^2\right)+y^2\right) \end{aligned}$$

the exact solution of (4.1) is

$$u(x,y)=\exp\left(-1000\left(x-\frac{1}{2}\right)^2\right)-y^2. \tag{4.2}$$

We use 6 levels of nested grids with the coarsest grid 48×48 and the finest grid 1536×1536. The numerical results obtained by the EXCMG-Newton method with $\epsilon_{lin}=10^{-9}$ are shown in Tables 1,3,5,7. Note that for $\lambda=0.01$, the Newton's method on the finest grid is terminated when the Euclid norm of the residual gets smaller than 10^{-2}, and for $\lambda=1$ we set $\epsilon_{non}=10^{-6}$. The number of Newton iterations is abbreviated as Iter, and we also show the errors of the methods tested in various norms.

Tables 1,3,5,7 show that the numerical solutions u_h obtained by four compact FD schemes have fourth-order accuracy and the initial values $\bar{u}_h$ are of fifth-order approximations

Table 1: Example 4.1. HOC (a) based discretization. Errors and convergence orders.

λ	Mesh	Iter	$\|u_h-u\|_2$		$\|u_h-u\|_\infty$		$\|\bar{u}_h-u_h\|_2$		$\|w_h-u\|_2$	
			Error	Order	Error	Order	Error	Order	Error	Order
0.01	192×192	2	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	1	3.84e-07	4.01	2.31e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	1	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.80e-12	6.00
	Work unit	1.34 WU [a]								
1	192×192	3	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	2	3.84e-07	4.01	2.30e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	2	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.79e-12	6.00
	Work unit	1.67 WU [a]								

[a] WU (work unit) is the computational cost of one Newton iteration on the finest grid.

Table 2: Example 4.1. HOC (a) based discretization. Computational time.

λ	Mesh	EXCMG-Newton	Newton-MG [16]			Newton-MSMG [16]		
		Time(s)	$\|u_h-u\|_2$	Order	Time(s)	$\|u_h-u\|_2$	Order	Time(s)
0.01	192×192	1.77	6.17e-06	–	5.23	1.05e-06	–	4.77
	384×384	3.71	3.84e-07	4.01	20.67	1.59e-08	6.05	14.65
	768×768	15.09	2.40e-08	4.00	83.95	2.47e-10	6.01	47.79
	1536×1536	53.25	1.50e-09	4.00	329.67	3.84e-12	6.01	161.37
	Total time(s) [b]	73.82			439.52			228.58
1	192×192	2.61	6.17e-06	–	5.01	1.05e-06	–	4.19
	384×384	7.48	3.84e-07	4.01	20.51	1.59e-08	6.05	14.48
	768×768	30.37	2.40e-08	4.00	85.31	2.47e-10	6.01	47.87
	1536×1536	49.93	1.50e-09	4.00	337.39	3.84e-12	6.01	161.96
	Total time(s) [b]	90.39			448.22			228.50

[b] Total CPU time including the solution of algebraic systems.

Table 3: Example 4.1. HOC (b) based discretization. Errors and convergence orders.

λ	Mesh	Iter	$\|u_h-u\|_2$		$\|u_h-u\|_\infty$		$\|\bar{u}_h-u_h\|_2$		$\|w_h-u\|_2$	
			Error	Order	Error	Order	Error	Order	Error	Order
0.01	192×192	2	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	1	3.84e-07	4.01	2.31e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	1	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.80e-12	5.99
	Work unit	1.34 WU								
1	192×192	3	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	2	3.84e-07	4.01	2.30e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	2	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.83e-12	5.97
	Work unit	1.67 WU								

Table 4: Example 4.1. HOC (b) based discretization. Computational time.

λ	Mesh	EXCMG-Newton	Newton-MG [16]			Newton-MSMG [16]		
		Time(s)	$\|u_h-u\|_2$	Order	Time(s)	$\|u_h-u\|_2$	Order	Time(s)
0.01	192×192	1.83	6.17e-06	–	4.86	1.05e-06	–	3.90
	384×384	3.45	3.84e-07	4.01	19.51	1.59e-08	6.05	13.54
	768×768	14.66	2.40e-08	4.00	79.04	2.47e-10	6.01	47.40
	1536×1536	53.37	1.50e-09	4.00	328.77	3.84e-12	6.01	161.08
	Total time(s)	73.31			432.18			225.92
1	192×192	2.53	6.17e-06	–	4.93	1.05e-06	–	3.95
	384×384	7.48	3.84e-07	4.01	20.47	1.59e-08	6.05	14.08
	768×768	28.55	2.40e-08	4.00	80.82	2.47e-10	6.01	47.89
	1536×1536	51.51	1.50e-09	4.00	346.11	3.84e-12	6.01	162.81
	Total time(s)	90.07			452.33			228.73

Table 5: Example 4.1. HOC (c) based discretization. Errors and convergence orders.

λ	Mesh	Iter	$\|u_h-u\|_2$		$\|u_h-u\|_\infty$		$\|\bar{u}_h-u_h\|_2$		$\|w_h-u\|_2$	
			Error	Order	Error	Order	Error	Order	Error	Order
0.01	192×192	2	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	1	3.84e-07	4.01	2.31e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	1	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.80e-12	5.99
	Work unit	1.34 WU								
1	192×192	3	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	2	3.84e-07	4.01	2.30e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	2	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.80e-12	5.99
	Work unit	1.67 WU								

Table 6: Example 4.1. HOC (c) based discretization. Computational time.

λ	Mesh	EXCMG-Newton	Newton-MG [16]			Newton-MSMG [16]		
		Time(s)	$\|u_h-u\|_2$	Order	Time(s)	$\|u_h-u\|_2$	Order	Time(s)
0.01	192×192	1.84	6.17e-06	–	4.83	1.05e-06	–	4.27
	384×384	3.71	3.84e-07	4.01	19.86	1.59e-08	6.05	14.04
	768×768	14.08	2.40e-08	4.00	81.29	2.47e-10	6.01	47.34
	1536×1536	54.27	1.50e-09	4.00	324.84	3.84e-12	6.01	174.81
	Total time(s)	73.90			430.82			240.46
1	192×192	2.68	6.17e-06	–	4.87	1.05e-06	–	4.03
	384×384	7.46	3.84e-07	4.01	19.36	1.59e-08	6.05	15.94
	768×768	28.70	2.40e-08	4.00	79.74	2.47e-10	6.01	56.94
	1536×1536	49.69	1.50e-09	4.00	335.39	3.84e-12	6.01	190.30
	Total time(s)	88.53			439.36			267.21

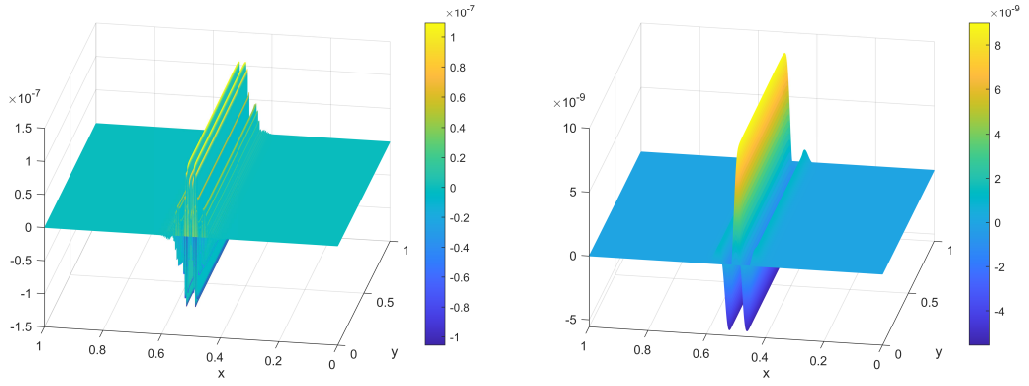
Table 7: Example 4.1. HOC (d) based on discretization. Errors and convergence orders.

λ	Mesh	Iter	$\|u_h-u\|_2$		$\|u_h-u\|_\infty$		$\|\bar{u}_h-u_h\|_2$		$\|w_h-u\|_2$	
			Error	Order	Error	Order	Error	Order	Error	Order
0.01	192×192	2	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	1	3.84e-07	4.01	2.31e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	1	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.80e-12	5.99
	Work unit	1.34 WU								
1	192×192	3	6.17e-06	4.04	3.72e-05	4.05	3.59e-04	–	4.77e-07	–
	384×384	2	3.84e-07	4.01	2.30e-06	4.01	1.12e-05	5.01	7.35e-09	6.02
	768×768	2	2.40e-08	4.00	1.44e-07	4.00	3.41e-07	5.03	1.15e-10	6.00
	1536×1536	1	1.50e-09	4.00	8.98e-09	4.00	1.07e-08	4.99	1.81e-12	5.98
	Work unit	1.67 WU								

Table 8: Example 4.1. HOC (d) based discretization. Computational time.

λ	Mesh	EXCMG-Newton	Newton-MG [16]			Newton-MSMG [16]		
		Time(s)	$\|u_h-u\|_2$	Order	Time(s)	$\|u_h-u\|_2$	Order	Time(s)
0.01	192×192	1.81	6.17e-06	–	5.77	1.05e-06	–	3.97
	384×384	3.63	3.84e-07	4.01	22.32	1.59e-08	6.05	14.28
	768×768	14.68	2.40e-08	4.00	83.92	2.47e-10	6.01	48.98
	1536×1536	56.25	1.50e-09	4.00	340.19	3.84e-12	6.01	167.54
	Total time(s)	76.37			452.20			234.77
1	192×192	2.67	6.17e-06	–	5.07	1.05e-06	–	4.05
	384×384	7.43	3.84e-07	4.01	20.48	1.59e-08	6.05	14.36
	768×768	28.97	2.40e-08	4.00	85.08	2.47e-10	6.01	49.90
	1536×1536	51.43	1.50e-09	4.00	343.80	3.84e-12	6.01	167.32
	Total time(s)	90.50			454.43			235.63

for the numerical solutions u_h, consistent with the theoretical results of Section 3.3. The extrapolated solution w_h almost achieves sixth-order approximation on all grids. This is due to the extrapolated solution w_h is derived from two fourth-order FD solutions u_h and u_{2h} on the first two levels of grids. They must be extremely accurate in order to get extrapolated solution w_h of sixth-order accuracy. As the grid becomes finer, the number of nonlinear iterations needed remains the same — viz. 1. This is essential for efficient solving of large nonlinear systems. Fig. 2 displays the difference between the true solution $u_{1/1536}$ and the iterative solutions $u^k_{1/1536}$, $k=0,1$ in the case of HOC (a) discretization with $\lambda=0.01$. Note that the high frequency error can be markedly smoothed out only after one Newton iteration, with the error reduced by two orders in magnitude.



(a) $u^0_{1/1536}-u_{1/1536}$ (b) $u^1_{1/1536}-u_{1/1536}$

Figure 2: Example 4.1. Difference between $u^k_{1/1536}$, $k=0,1$ and $u_{1/1536}$ when HOC (a) discretization is used, $\lambda=0.01$.

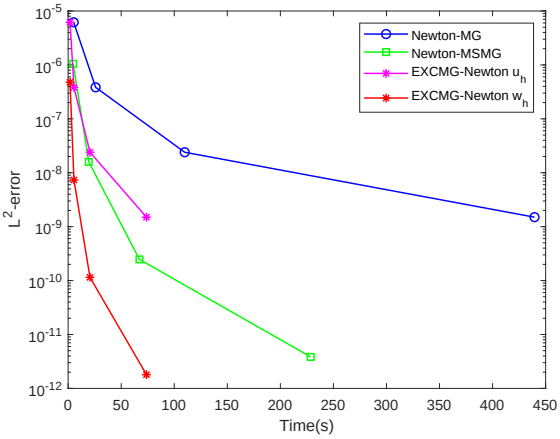


Figure 3: Example 4.1. MG algorithms based on HOC (a) discretization, $\lambda = 0.01$. Computational time versus L^2-error.

The results presented in Tables 2,4,6,8 show the efficiency of the proposed method. We note that the Newton-MG method needs about 450 seconds to reach fourth-order accuracy on the finest grid, the Newton-MSMG almost 230 seconds to reach the sixth-order accuracy, but the extrapolated solution w_h achieves the sixth-order accuracy, and the numerical solution u_h needs only about 80 seconds to achieves the fourth-order accuracy — i.e. the calculation efficiency of our method is about 5-6 times that of Newton-MG method, and about 2-3 times that of Newton-MSMG method. We also note that L^2-errors of the extrapolated solutions are about one-half of the ones obtained by the Newton-MSMG method, so that this method is much more efficient than the Newton-MG and Newton-MSMG methods.

Besides, Fig. 3 shows the L^2-errors of three methods combined with HOC (a) discretization on the each grid versus their corresponding computational times. Comparing the rates of the L^2-errors, we observe that the errors of our method rapidly decline in magnitude just after a few Newton iterations, and the achieved relative residual errors are smaller than the given tolerance.

Example 4.2 (cf. Li *et al.* [16])**.** Consider another nonlinear Poisson-Boltzmann model

$$\begin{aligned} &u_{xx}(x,y)+u_{yy}(x,y)+100^2\exp\big(-u(x,y)\big)=g(x,y),\\ &(x,y)\in\Omega=[0,1]\times[0,1] \end{aligned} \tag{4.3}$$

with the source term

$$\begin{aligned} g(x,y)=2\exp(y-x)+\frac{101^2(1+y)^{101}}{2^{100}(1+y)^2}-\frac{101(1+y)^{101}}{2^{100}(1+y)^2}\\ +100^2\exp\left(-\exp(y-x)-\frac{(1+y)^{101}}{2^{100}}\right). \end{aligned}$$

The solution of this problem is

$$u(x,y)=\exp(y-x)+\frac{(1+y)^{100}}{2^{100}}. \tag{4.4}$$

Once again, we apply 6 levels of embedded grids with the 1536×1536 finest grid. Numerical results obtained by the EXCMG-Newton method with $\epsilon_{lin} = 10^{-10}$ are shown in Table 9. Note that on the finest grid, the Newton method is terminated when the Euclid norm of the residual is smaller than 10^{-2}. The numerical solution u_h in two compact FD discretizations reaches fourth-order accuracy, the initial value $\bar{u}_h$ is of the fifth-order approximation for numerical solution u_h, and on all grids the extrapolated solution w_h almost achieves sixth-order approximation. When grids get finer, the number of Newton iterations needed is stable at 1. It is critical to reducing computational cost.

Table 10 shows computational time for our and two other methods on each grid and the total CPU time. Note that the Newton-MG method requires about 470 seconds to reach the fourth-order accuracy on the finest grid, the Newton-MSMG method needs about 230 seconds to achieve the sixth-order accuracy, whereas the extrapolated solution w_h and numer-

Table 9: Example 4.2. Two HOC based discretizations. Errors and convergence orders.

HOC	Mesh	Iter	$\|u_h-u\|_2$		$\|u_h-u\|_\infty$		$\|\bar{u}_h-u_h\|_2$		$\|w_h-u\|_2$	
			Error	Order	Error	Order	Error	Order	Error	Order
HOC (a)	192×192	2	1.86e-06	3.98	1.19e-05	3.96	8.44e-05	–	8.06e-08	–
	384×384	1	1.17e-07	3.99	7.43e-07	4.00	3.20e-06	4.72	1.29e-09	5.97
	768×768	1	7.31e-09	4.00	4.65e-08	4.00	1.06e-07	4.92	2.01e-11	6.00
	1536×1536	1	4.57e-10	4.00	2.90e-09	4.00	3.35e-09	4.98	3.17e-13	5.99
	Work unit	1.34 WU								
HOC (d)	192×192	2	1.86e-06	3.98	1.19e-05	3.96	8.44e-05	–	8.06e-08	–
	384×384	1	1.17e-07	4.00	7.43e-07	4.00	3.20e-06	4.72	1.29e-09	5.97
	768×768	1	7.31e-09	4.00	4.65e-08	4.00	1.06e-07	4.92	2.01e-11	6.00
	1536×1536	1	4.57e-10	4.00	2.90e-09	4.00	3.35e-09	4.98	3.18e-13	5.98
	Work unit	1.34 WU								

Table 10: Example 4.2. Two HOC based discretizations. Computational time.

HOC	Mesh	EXCMG-Newton	Newton-MG [16]			Newton-MSMG [16]		
		Time(s)	$\|u_h-u\|_2$	Order	Time(s)	$\|u_h-u\|_2$	Order	Time(s)
HOC (a)	192×192	1.60	1.86e-06	–	5.19	1.86e-07	–	3.11
	384×384	3.15	1.17e-07	4.00	21.06	3.04e-09	5.94	12.61
	768×768	13.48	7.31e-09	4.00	85.71	4.80e-11	5.98	51.08
	1536×1536	60.45	4.57e-10	4.00	360.93	7.51e-13	6.00	163.87
	Total time(s)	78.68			472.89			230.67
HOC (d)	192×192	1.56	1.86e-06	–	5.32	1.86e-07	–	3.05
	384×384	3.21	1.17e-07	4.00	21.41	3.04e-09	5.94	13.42
	768×768	13.52	7.31e-09	4.00	87.06	4.80e-11	5.98	50.82
	1536×1536	61.29	4.57e-10	4.00	360.14	7.51e-13	6.00	160.60
	Total time(s)	79.58			473.93			227.89

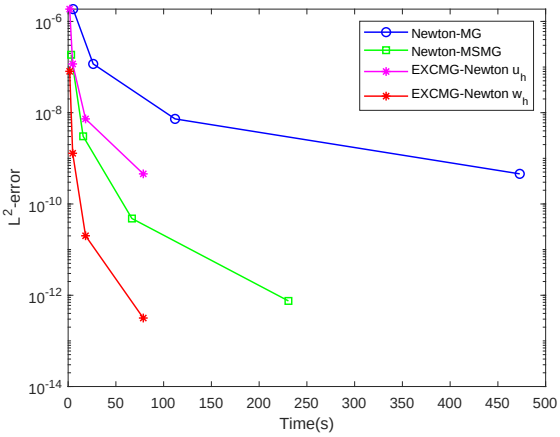


Figure 4: Example 4.2. MG algorithms based on HOC (a) discretization. Computational time versus L^2-error.

ical solution u_h in our method reach the corresponding accuracy only in about 80 seconds. In addition, the L^2-errors of the extrapolated solutions are about half of the Newton-MSMG method. Hence, the EXCMG-Newton method is more cost-effective than Newton-MG and Newton-MSMG methods. Computational times versus L^2-errors is presented in Fig. 4. We observe that errors of the proposed method rapidly diminish just after one Newton iteration.

Example 4.3. Consider the low regularity problem

$$u_{xx}(x,y)+u_{yy}(x,y)-u(x,y)^3=g(x,y),$$
$$(x,y)\in\Omega=[0,1]\times[0,1],$$

whose solution

$$u(x,y)=-\frac{(x^2+y^2)^{5/3}}{0.01^2} \tag{4.5}$$

has a singularity at the origin. The source term and boundary conditions are determined from the analytical solution (4.5).

We use 6 nested grids with the 1536×1536 finest grid. The corresponding numerical results for the EXCMG-Newton method with $\epsilon_{non}=10^{-2}$ and $\epsilon_{lin}=10^{-10}$ are shown in Table 11.

Table 11: Example 4.3. Errors and convergence orders of the method based on HOC (b) discretization.

Mesh	Iter	Time(s)	$\|u_h-u\|_2$		$\|u_h-u\|_\infty$		$\|\bar{u}_h-u_h\|_2$		$\|w_h-u\|_2$	
			Error	Order	Error	Order	Error	Order	Error	Order
192 × 192	1	0.84	1.42e-07	3.93	8.60e-06	3.33	3.80e-06	–	4.08e-08	–
384 × 384	1	3.81	9.13e-09	3.96	8.54e-07	3.33	1.89e-07	4.33	2.03e-09	4.33
768 × 768	1	15.42	5.80e-10	3.98	8.47e-08	3.33	9.41e-09	4.33	1.01e-10	4.33
1536 × 1536	1	69.13	3.66e-11	3.99	8.40e-09	3.33	4.67e-10	4.33	5.07e-12	4.31
Work unit	1.33 WU									

Since the exact solution is not smooth, the numerical solution u_h still reaches the fourth-order accuracy in L^2-norm, but only 3.33-order in L^∞-norm. The initial guess $\bar{u}_h$ is of 4.33-order approximation to the numerical solution u_h, and the extrapolated solution w_h is also of 4.33-order accuracy. Note that the EXCMG-Newton method achieves the desired accuracy with only one Newton iteration on each grid. Thus it is still efficient for low regularity problems.

5. Conclusions

Using the framework of EXCMG methods, we construct a fast solver — viz. an EXCMG-Newton method combined with fourth-order compact FD schemes for 2D Poisson equations with nonlinear forcing term. Combining extrapolation and bi-quartic interpolation for two numerical solutions from the previous two levels of grids, we derive a suitable initial guess for Newton iterations on the next finer grid. It is of fifth-order accuracy, which substantially reduces the number of Newton iterations required. Moreover, an extrapolated solution of sixth-order accuracy can be easily constructed on the whole fine grid. Numerical results suggest that the proposed EXCMG-Newton method is much more efficient than Newton-MG and Newton-MSMG methods.

Note that the method can be extended to other nonlinear partial differential equations. Besides, fourth-order compact FD schemes can be replaced by other discretization methods, such as finite element and finite volume methods.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments. We are also grateful to the High Performance Computing Center of the Central South University for assistance with the computations.

K. Pan was supported by the National Natural Science Foundation of China (Grant Nos. 42274101, 41874086) and the Shenzhen Science and Technology Program (Grant No. JCYJ20210324125601005). P. Wu was supported by the Fundamental Research Funds for the Central Universities of Central South University (Grant No. 2020zzts035). D. He was supported by the National Natural Science Foundation of China (Grant No. 12172317), the Shenzhen Science and Technology Program (Grant No. JCYJ20210324125601005) and the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2022A1515011784).

References

[1] I. Altas, J. Dym, M.M. Gupta and R.P. Manohar, *Multigrid solution of automatically generated high-order discretizations for the biharmonic equation*, SIAM J. Sci. Comput. **19**, 1575–1585 (1998).

[2] F.A. Bornemann and P. Deuflhard, *The cascadic multigrid method for elliptic problems*, Numer. Math. **75**, 135–152 (1996).

[3] W.L. Briggs, V.E. Henson and S.F. McCormick, *A Multigrid Tutorial*, SIAM (2000).

[4] C. Chen, H. Hu, Z. Xie and C. Li, *Analysis of extrapolation cascadic multigrid method (EXCMG)*, Sci. China. Math. **51**, 1349–1360 (2008).
[5] C. Chen and Q. Lin, *Extrapolation of finite element approximation in a rectangular domain*, J. Comput. Math. **3**, 227–233 (1989).
[6] R. Dai, P. Lin and J. Zhang, *An efficient sixth-order solution for anisotropic Poisson equation with completed Richardson extrapolation and multiscale multigrid method*, Comput. Math. Appl. **73**, 1865–1877 (2017).
[7] R. Dai, P. Lin and J. Zhang, *An EXCMG accelerated multiscale multigrid computation for 3D Poisson equation*, Comput. Math. Appl. **77**, 2051–2060 (2019).
[8] R.S. Dembo, S.C. Eisenstat and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal. **19**, 400–408 (1982).
[9] P. Deuflhard, *Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms*, Springer Science & Business Media, (2005).
[10] H.C. Elman, O.G. Ernst and D.P. O'leary, *A multigrid method enhanced by Krylov subspace iteration for discrete Helmholtz equations*, SIAM J. Sci. Comput. **23**, 1291–1315 (2001).
[11] Y. Ge, *Multigrid method and fourth-order compact difference discretization scheme with unequal meshsizes for 3D Poisson equation*, J. Comput. Phys. **229**, 6381–6391 (2010).
[12] Y. Ge and F. Cao, *Multigrid method based on the transformation-free HOC scheme on nonuniform grids for 2D convection diffusion problems*, J. Comput. Phys. **230**, 4051–4070 (2011).
[13] M.M. Gupta, J. Kouatchou and J. Zhang, *Comparison of second- and fourth-order discretizations for multigrid Poisson solvers*, J. Comput. Phys. **132**, 226–232 (1997).
[14] M.M. Gupta and J. Zhang, *High accuracy multigrid solution of the 3D convection-diffusion equation*, Appl. Math. Comput. **113**, 249–274 (2000).
[15] M. Li, C. Li, X. Cui and J. Zhao, *Cascadic multigrid methods combined with sixth order compact scheme for Poisson equation*, Numer. Algorithms. **71**, 715–727 (2016).
[16] M. Li, Z. Zheng, K. Pan and X. Yue, *An efficient Newton multiscale multigrid method for 2D semilinear Poisson equations*, East Asian J. Appl. Math. **10**, 620–634 (2020).
[17] R. Manohar and J. Stephenson, *High order difference shemes for linear partial differential equations*, SIAM J. Sci. Statist. Comput. **5**, 69–77 (1984).
[18] G.I. Marchuk and V.V. Shaidurov, *Difference Methods and Their Extrapolations*, Springer Science & Business Media (2012).
[19] R.K. Mohanty and S. Singh, *A new fourth order discretization for singularly perturbed two dimensional non-linear elliptic boundary value problems*, Appl. Math. Comput. **175**, 1400–1414 (2006).
[20] S. Molavi-Arabshahi and M. Dehghan, *Preconditioned techniques for solving large sparse linear systems arising from the discretization of the elliptic partial differential equations*, Appl. Math. Comput. **188**, 1371–1388 (2007).
[21] K. Pan, D. He and H. Hu, *An extrapolation cascadic multigrid method combined with a fourth-order compact scheme for 3D Poisson equation*, J. Sci. Comput. **70**, 1180–1203 (2017).
[22] 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**, 499–515 (2017).
[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**, 110794 (2022).
[24] K. Pan, X. Wu, Y. Yu, Z. Sheng and G. Yuan, *Extrapolation cascadic multigrid method for cell-centered FV discretization of diffusion equations with strongly discontinuous and anisotropic coefficients*, Commun. Comput. Phys. **31**, 1561–1584 (2022).
[25] P.J. Roache and P.M. Knupp, *Completed Richardson extrapolation*, Commun. Numer. Methods.

Eng. **9**, 365–374 (1993).
[26] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM (2003).
[27] V. Shaidurov and G. Timmermann, *A cascadic multigrid algorithm for semilinear indefinite elliptic problems*, Computing. **64**, 349–366 (2000).
[28] C.C. Stolk, M. Ahmed and S.K. Bhowmik, *A multigrid method for the Helmholtz equation with optimized coarse grid corrections*, SIAM J. Sci. Comput. **36**, 2841–2819 (2014).
[29] G. Timmermann, *A cascadic multigrid algorithm for semilinear elliptic problems*, Numer. Math. **86**, 717–731 (2000).
[30] Y. Wang and J. Zhang, *Sixth order compact scheme combined with multigrid method and extrapolation technique for 2D Poisson equation*, J. Comput. Phys. **228**, 137–146 (2009).
[31] Y. Wang and J. Zhang, *Fast and robust sixth-order multigrid computation for the three-dimensional convection-diffusion equation*, J. Comput. Appl. Math. **234**, 3496–3506 (2010).
[32] F. Xu, Q. Huang, S. Chen and T. Bing, *An adaptive multigrid method for semilinear elliptic equations*, East Asian J. Appl. Math. **9**, 683–702 (2019).
[33] H. Yu and J. Zeng, *A cascadic multigrid method for a kind of semilinear elliptic problem*, Numer. Algorithms **58**, 143–162 (2011).
[34] S. Zhai, X. Feng and Y. He, *A new method to deduce high-order compact difference schemes for two-dimensional Poisson equation*, Appl. Math. Comput. **230**, 9–26 (2014).
[35] J. Zhang, *Multigrid method and fourth-order compact scheme for 2D Poisson equation with unequal mesh-size discretization*, J. Comput. Phys. **179**, 170–179 (2002).