INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 15, Number 1-2, Pages 277–287

PRECONDITIONING TECHNIQUES IN CHEBYSHEV COLLOCATION METHOD FOR ELLIPTIC EQUATIONS

ZHI-WEI FANG, JIE SHEN, AND HAI-WEI SUN

This paper is dedicated to memory of late Professor Benyu Guo

Abstract. When one approximates elliptic equations by the spectral collocation method on the Chebyshev-Gauss-Lobatto (CGL) grid, the resulting coefficient matrix is dense and ill-conditioned. It is known that a good preconditioner, in the sense that the preconditioned system becomes well conditioned, can be constructed with finite difference on the CGL grid. However, there is a lack of an efficient solver for this preconditioner in multi-dimension. A modified preconditioner based on the approximate inverse technique is constructed in this paper. The computational cost of each iteration in solving the preconditioned system is $\mathcal{O}(\ell N_x N_y \log N_x)$, where N_x, N_y are the grid sizes in each direction and ℓ is a small integer. Numerical examples are given to demonstrate the efficiency of the proposed preconditioner.

Key words. Chebyshev collocation method, elliptic equation, finite-difference preconditioner, approximate inverse.

1. Introduction

We consider a two-dimensional separable elliptic equation

$$(1)\quad -\frac{\partial}{\partial x}\left(a(x)\frac{\partial u}{\partial x}\right)-\frac{\partial}{\partial y}\left(b(y)\frac{\partial u}{\partial y}\right)+c(x)d(y)u(x,y)=f(x,y)\quad \text{in } \Omega=(-1,1)^2$$

with homogeneous Dirichlet boundary conditions

$$u = 0 \text{ on } \partial\Omega,$$

where the coefficient functions $a(x)$, $b(y)$, $c(x)$, $d(y)$ and $f(x,y)$ are continuous, and $0 < \alpha \leq a(x), b(y) \leq \beta$ in Ω for some positive constants α and β, and $c(x)d(y) \geq 0$.

A very efficient accurate method for obtaining approximate solution of the above boundary-value problem is the Chebyshev collocation method [3, 4, 7, 9, 11, 12], which uses the Lagrange nodal basis functions based on the Chebyshev collocation points. However, due to the global nature of the Lagrange basis polynomials, the associated linear systems are dense and ill-conditioned. Thus it becomes prohibitive to use a direct inversion method or an iterative method without preconditioning in the multi-dimensional case, so it is imperative to use an iterative method with a good preconditioner.

Finite element/finite difference preconditioners have been widely used since the original work by Orszag [9]. Haldenwang et al. [5] proved that the finite difference method based on the Chebyshev collocation points in the one-dimensional case leads to a good preconditioner. The properties of the finite element/finite difference preconditioners in the two-dimensional case were rigorously established by Kim and Parter [6, 7]. Thus, the application of the Krylov subspace methods [1], such as generalized minimal residual method (GMRES), leads to an iterative solver converging to the algebraic solution within a constant number of steps that depends on the required accuracy, but not on the number of unknowns.

Received by the editors February 17, 2017 and, in revised form, April 10, 2017.
2000 *Mathematics Subject Classification.* 35J25, 65F10, 65N22, 65N35.

However, such a preconditioned method requires solving the preconditioner system, i.e., solving the finite element/finite difference system on the spectral collocation points. How to efficiently apply the preconditioners is a challenging problem since the grid formed by spectral collocation points, containing long-thin elements, is not shape-regular. We note that Shen et al. [13] developed a finite element multigrid preconditioner for the second-order elliptic equations. In this paper, we seek to develop an approximate preconditioner by exploring the algebraic properties of the finite difference preconditioner.

It is obvious that the two-dimensional finite difference preconditioner is a nonsymmetric block tridiagonal matrix. Approximating this matrix to construct a new efficient preconditioner is a natural idea. In [8], Ng and Pan proposed an approximate inverse method to modify circulant-plus-diagonal preconditioners for solving Toeplitz-plus-diagonal systems. Their idea is to use circulant matrices to approximate the inversion of Toeplitz matrices and then combine the rows of these matrices together. As the resulting preconditioner is already of the inverted form, only matrix-vector multiplications are required in the preconditioning step. Recently, Pan et al. [10] also proposed approximate inverse preconditioners for diagonal-times-Toeplitz matrices.

The main purpose of this paper is to propose and develop approximate inverse preconditioners for two-dimensional elliptic operators, based on the modification of the finite-difference operator discretized on the CGL grid. First, we use a scaling strategy to approximate the finite-difference operator. Then we construct an approximate inverse preconditioner to approximate the inverse of scaled Laplacian-plus-diagonal matrices and combine them together row-by-row. In order to reduce the influence of the various coefficients, an interpolation method with the eigenvalues of Laplacian is utilized. Special interpolation nodes are chosen to improve the accuracy of approximation. By use of the discrete sine transform (DST), the resulting preconditioner can be efficiently implemented with $\mathcal{O}(\ell N_x N_y \log N_x)$ operations, where the small integer ℓ is independent of N_x and N_y. Numerical examples are given to demonstrate the effectiveness of the proposed preconditioner.

The paper is organized as follows. In Section 2, we introduce the Chebyshev collocation method for the elliptic operator and the associated finite-difference operator. In Section 3, we construct the proposed preconditioners. Numerical examples are given to demonstrate the performance of the proposed preconditioner in Section 4. In the final section, concluding remarks are given.

2. The Chebyshev-collocation and the finite-difference operator

In this section we recall the Chebyshev-collocation method for the elliptic operator and the associated finite-difference operator. Let $\mathcal{P}_N$ be the space of polynomials of degree less than or equal to N. Let

$$x_j = -\cos\left(\frac{j\pi}{N}\right), \qquad j = 0, 1, \ldots, N,$$

which are the CGL points.

2.1. The one-dimensional case. Consider the one-dimensional elliptic problems

$$-(a(x)u'(x))' + c(x)u(x) = f(x), \qquad x \in (-1,1); \quad u(\pm 1) = 0. \tag{2}$$

The Chebyshev-collocation method for (2) is to find $u_N \in \mathcal{X}_N := \{v \in \mathcal{P}_N : v(\pm 1) = 0\}$ such that

$$-(au_N')'|_{x=x_k} + c(x_k)u_N(x_k) = f(x_k), \quad k = 1, 2, \ldots, N-1. \tag{3}$$

Let $\{p_j(x)\}_{j=0}^N$ be the Lagrange basis polynomials associated with $\{x_j\}_{j=0}^N$. Then, we can express $u_N(x) = \sum\limits_{j=0}^{N} u_N(x_j)p_j(x)$. Denoting the Chebyshev differentiation matrix by $G_N = (p_j'(x_k))_{k,j=0,1,\dots,N}$, we have

$$u_N'(x_k) = \sum_{j=0}^{N} u_N(x_j)p_j'(x_k) = \sum_{j=0}^{N} (G_N)_{kj}u_N(x_j),$$

and

$$u_N''(x_k) = \sum_{j=0}^{N} u_N(x_j)p_j''(x_k) = \sum_{j=0}^{N} (G_N^2)_{kj}u_N(x_j).$$

We list below the formulas for the entries of G_N for arbitrary N (cf. [2, 14]):

Lemma 1. *For each $N \geq 1$, let the rows and columns of the $(N+1)\times(N+1)$ Chebyshev spectral differentiation matrix G_N be indexed from 0 to N. The entries of this matrix are*

$$(G_N)_{00} = -\frac{2N^2+1}{6}, \qquad (G_N)_{NN} = \frac{2N^2+1}{6},$$

$$(G_N)_{jj} = \frac{-x_j}{2(1-x_j^2)}, \qquad j = 1,\dots,N-1,$$

$$(G_N)_{kj} = \frac{\gamma_k}{\gamma_j}\frac{(-1)^{k+j}}{(x_k - x_j)}, \qquad k \neq j, \quad k,j = 0,\dots,N,$$

where $\gamma_0 = \gamma_N = 2$ and $\gamma_k = 1$ for $k = 1,\dots,N-1$.

Explicit formulae for the entries of G_N^2 is also available in [3, 11]. By using the differentiation matrix G_N, it costs $\mathcal{O}(N^2)$ to compute the derivative of u_N at all CGL points. However, this process can be accelerated to $\mathcal{O}(N\log N)$ by expressing u_N in Chebyshev polynomials and using the fast cosine transform [4, 12].

Set $D_N^a = \operatorname{diag}(a(x_0), a(x_1), \dots, a(x_N))$, $\bar{D}_N^c = \operatorname{diag}(c(x_1), c(x_2), \dots, c(x_{N-1}))$,

$$A = \{(-G_N D_N^a G_N)_{k,j}\}_{1\leq k,j\leq N-1}, \quad \text{and} \quad A_1 = A + \bar{D}_N^c.$$

The Chebyshev-collocation scheme (3) reduces to the following linear system

$$A_1\bar{\mathbf{u}} = \bar{\mathbf{f}},$$

where $\bar{\mathbf{u}} = (u_N(x_1),\dots,u_N(x_{N-1}))^\intercal$ and $\bar{\mathbf{f}} = (f(x_1),\dots,f(x_{N-1}))^\intercal$.

The matrix A is full and ill-conditioned. As proposed in [9], a good preconditioner for A_1 is to use a finite-difference operator on the CGL grid. Denote $h_j = x_j - x_{j-1}$ $(j = 1,\dots,N)$, $\tilde{h}_j = (x_{j+1} - x_{j-1})/2$ $(j = 1,\dots,N-1)$, and $a_{k+1/2} = a((x_k + x_{k+1})/2)$ $(k = 0,1,\dots,N-1)$. We consider the following finite difference approximation:

$$\begin{aligned}(au')'|_{x=x_i} \approx & \frac{a_{i-1/2}}{\tilde{h}_i h_i}u(x_{i-1}) - \left(\frac{a_{i-1/2}}{\tilde{h}_i h_i} + \frac{a_{i+1/2}}{\tilde{h}_i h_{i+1}}\right)u(x_i) \\ & + \frac{a_{i+1/2}}{\tilde{h}_i h_{i+1}}u(x_{i+1}), \quad i = 1,2,\dots,N-1.\end{aligned}$$

Then, the preconditioner for A_1 based on the above approximation can be written as follows:

$$B_1 := F_N + \bar{D}_N^c,$$

where

$$(F_N)_{ij} := \begin{cases} -\frac{a_{i-1/2}}{\tilde{h}_i h_i}, & j = i-1, \\ \frac{a_{i-1/2}}{\tilde{h}_i h_i} + \frac{a_{i+1/2}}{\tilde{h}_i h_{i+1}}, & j = i, \\ -\frac{a_{i+1/2}}{\tilde{h}_i h_{i+1}}, & j = i+1. \end{cases} \tag{4}$$

Note that B_1 is a nonsymmetric tridiagonal matrix. Inverting B_1 or solving a linear system with B_1 as the coefficient matrix requires about $\mathcal{O}(N)$ operations.

2.2. The two-dimensional case. For the two-dimensional elliptic equations (1), the collocation points are the tensor product of univariate CGL nodes. Assume that N_x and N_y are the number of the CGL points in each direction respectively. Then, the Chebyshev-collocation method will lead to a linear system with the matrix

$$A_2 := I_{N_x} \otimes A^b_{N_y} + A^a_{N_x} \otimes I_{N_y} + \bar{D}^c_{N_x} \otimes \bar{D}^d_{N_y},$$

where $\otimes$ denotes the Kronecker product, $A^a_{N_x}$ is the matrix A defined in the last sub-section, $A^b_{N_y}$ is similar to $A^a_{N_x}$ with $b(y)$ replacing $a(x)$, $\bar{D}^c_{N_x}$ is the diagonal matrix $\bar{D}^c_N$ defined in the last sub-section, $\bar{D}^d_{N_y}$ is similar to $\bar{D}^c_N$ where $d(y)$ is used instead of $c(x)$, and both I_{N_x} and I_{N_y} are identity matrices. The finite-difference operator associated with the two-dimensional elliptic operator is defined as follows:

$$B_2 := I_{N_x} \otimes F^b_{N_y} + F^a_{N_x} \otimes I_{N_y} + \bar{D}^c_{N_x} \otimes \bar{D}^d_{N_y}, \tag{5}$$

where $F^a_{N_x}$ is defined by the formula (4), $F^b_{N_y}$ is defined analogously with $a(x)$ replaced by $b(y)$.

We remark that B_2 is a non-symmetric block tridiagonal with tridiagonal blocks matrix. Therefore, unlike B_1 in the one-dimensional case, it is not an easy task to invert B_2. In the following section, we shall construct a preconditioner based on the approximate inverse strategy for B_2.

3. Construction of the preconditioner

For the interest of simplicity, we first discuss the basic techniques in the one-dimensional case, and then these techniques are utilized to approximate B_2.

3.1. Construction in the one-dimensional case. We shall construct an effective preconditioner in the one-dimensional case through a sequence of approximations. Taking the structure of the matrix F_N into consideration, we firstly propose a scaled matrix as an approximation. Define

$$t_i = \sqrt{\frac{1}{2}\left(\frac{a_{i-1/2}}{h_i} + \frac{a_{i+1/2}}{h_{i+1}}\right)}, \quad i = 1, 2, \ldots, N-1,$$

and

$$T_N = \operatorname{diag}(t_1, t_2, \ldots, t_{N-1}). \tag{6}$$

The first approximation is as follows:

$$F_N \approx H_N T_N L_N T_N, \tag{7}$$

where

$$H_N = \operatorname{diag}\left(\frac{1}{\tilde{h}_1}, \frac{1}{\tilde{h}_2}, \ldots, \frac{1}{\tilde{h}_{N-1}}\right), \tag{8}$$

and

$$L_N = \begin{bmatrix} 2 & -1 & & 0 \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{(N-1)\times(N-1)}. \tag{9}$$

Then, we construct the first preconditioner

$$P_1 = H_N T_N L_N T_N + \bar{D}_N^c = H_N T_N (L_N + \bar{D}_N^c H_N^{-1} T_N^{-2}) T_N.$$

Denote

$$M = L_N + \bar{D}_N^c H_N^{-1} T_N^{-2}. \tag{10}$$

Then $P_1 = H_N T_N M T_N$. We note that the matrices H_N, T_N are diagonal and can be easily to handle. Therefore, we only consider how to invert the matrix M in (10).

Define

$$K_i = L_N + \frac{\tilde{h}_i c_i}{t_i^2} I_N, \quad i = 1, 2, \ldots, N-1, \tag{11}$$

where $c_i = c(x_i)$. Let e_i be the i-th column of the identity matrix. According to the fact that

$$e_i^{\intercal} M = e_i^{\intercal} K_i,$$

we construct our preconditioner based on the following approximation [8, 10]

$$e_i^{\intercal} M^{-1} \approx e_i^{\intercal} K_i^{-1}.$$

This means that the i-th row of the inverse of M is approximated by the i-th row of the inverse of K_i. Therefore, we propose our second preconditioner P_2 whose inverse is defined by

$$P_2^{-1} = T_N^{-1} \left(\sum_{i=1}^{N-1} e_i e_i^{\intercal} K_i^{-1} \right) H_N^{-1} T_N^{-1}. \tag{12}$$

We see from above that, to construct P_2^{-1}, we need to compute the inverse of K_i $(i = 1, 2, \ldots, N-1)$. Since the matrix L_N can be diagonalized in $\mathcal{O}(N \log N)$ operations by the DST, the product $K_i^{-1} v$ for any vector v can be computed in $\mathcal{O}(N \log N)$ operations. Let S_N be the $(N-1) \times (N-1)$ DST matrix. Note that S_N is symmetric, orthogonal and its (i, j)-th entry is given by

$$\sqrt{\frac{2}{N}} \sin\left(\frac{\pi i j}{N}\right), \qquad 1 \le i, j \le N-1.$$

Thus, the inverse of K_i can be computed by

$$K_i^{-1} = S_N \left(\Lambda_N + \frac{\tilde{h}_i c_i}{t_i^2} I_N \right)^{-1} S_N,$$

where Λ_N is a diagonal matrix whose entries are $2 - 2\cos(\frac{j\pi}{N}), j = 1, 2, \ldots, N-1$, the eigenvalues of L_N. Hence, implementing a preconditioner based on P_2 requires $\mathcal{O}(N)$ DST per iteration, which is still too expensive.

In order to reduce the computational cost, we propose to use the interpolation method to construct a more efficient preconditioner. We choose a small number

$\ell(\ell \ll N)$ of values $\{\theta_j\}_{j=1}^{\ell} \subset \{\xi_i = \frac{i\pi}{N}\}_{i=1}^{N-1}$, which covers (most of) the range of values of $\{\xi_i\}_{i=1}^{N-1}$. Define

$$q_i(\theta) = \frac{1}{\lambda_\Lambda(\theta) + w_i}, \qquad \theta \in (0, \pi),$$

where $\lambda_\Lambda(\theta) = 2 - 2\cos\theta$ and $w_i = \frac{\tilde{h}_i c_i}{t_i^2}$. Let

$$p_i(\theta) = \phi_1(\theta) q_i(\theta_1) + \phi_2(\theta) q_i(\theta_2) + \cdots + \phi_\ell(\theta) q_i(\theta_\ell) \tag{13}$$

be the piecewise linear interpolation for $q_i(\theta)$ based on the ℓ points $\{\theta_j, q_i(\theta_j)\}_{j=1}^{\ell}$.

We apply interpolation formula (13) to approximate K_i^{-1}:

$$K_i^{-1} \approx S_N \left(\sum_{j=1}^{\ell} \Phi_j q_i(\theta_j) \right) S_N, \quad i = 1, 2, \ldots, N-1, \tag{14}$$

where $\Phi_j = \operatorname{diag}(\phi_j(\xi_1), \phi_j(\xi_2), \ldots, \phi_j(\xi_{N-1}))$ are the interpolation coefficient matrices.

Finally,combining the above consideration, we define our final preconditioner P_3 by

$$\begin{aligned}
P_3^{-1} =& T_N^{-1} \sum_{i=1}^{N-1} e_i e_i^\intercal S_N \left(\sum_{j=1}^{\ell} \Phi_j q_i(\theta_j) \right) S_N H_N^{-1} T_N^{-1} \\
=& T_N^{-1} \sum_{i=1}^{N-1} \sum_{j=1}^{\ell} e_i e_i^\intercal q_i(\theta_j) S_N \Phi_j S_N H_N^{-1} T_N^{-1} \\
=& T_N^{-1} \sum_{j=1}^{\ell} \left(\sum_{i=1}^{N-1} e_i e_i^\intercal q_i(\theta_j) \right) S_N \Phi_j S_N H_N^{-1} T_N^{-1} \\
=& T_N^{-1} \left(\sum_{j=1}^{\ell} W_j S_N \Phi_j \right) S_N H_N^{-1} T_N^{-1},
\end{aligned}$$

where $W_j = \operatorname{diag}(q_1(\theta_j), q_2(\theta_j), \ldots, q_{N-1}(\theta_j))$ are diagonal matrices. Now applying P_3^{-1} to any vector requires about $\mathcal{O}(\ell N \log N)$ operations which is acceptable for a small number ℓ. Since the original function $q_i(\theta)$ has weak singularities near $\theta = 0$, the interpolation nodes should be slightly dense near ξ_1.

3.2. Construction in the two-dimensional case. In the following, we apply similar techniques to construct a sequence of approximate preconditioner for B_2 defined in (5).

First, using the approximation (7) in x-direction, we define our first preconditioner by

$$\begin{aligned}
\hat{P}_1 =& I_{N_x} \otimes F_{N_y}^b + H_{N_x} T_{N_x} L_{N_x} T_{N_x} \otimes I_{N_y} + \bar{D}_{N_x}^c \otimes \bar{D}_{N_y}^d \\
=& (H_{N_x} T_{N_x} \otimes I_{N_y}) \Big(H_{N_x}^{-1} T_{N_x}^{-2} \otimes F_{N_y}^b + L_{N_x} \otimes I_{N_y} \\
& + H_{N_x}^{-1} T_{N_x}^{-2} \bar{D}_{N_x}^c \otimes \bar{D}_{N_y}^d \Big) (T_{N_x} \otimes I_{N_y}) \\
\triangleq& (H_{N_x} T_{N_x} \otimes I_{N_y}) \hat{M} (T_{N_x} \otimes I_{N_y}),
\end{aligned} \tag{15}$$

where H_{N_x}, T_{N_x} and L_{N_x} are defined as formulas (8), (6) and (9) respectively, and $\hat{M}$ denotes the middle term in the above decomposition. Note that the first and the last terms in the decomposition of $\hat{P}_1$ in (15) are diagonal. Therefore, we only need to construct a preconditioner for the middle term $\hat{M}$. Define

$$\hat{K}_i = \frac{\tilde{h}_i}{t_i^2} I_{N_x} \otimes F_{N_y}^b + L_{N_x} \otimes I_{N_y} + \frac{\tilde{h}_i c_i}{t_i^2} I_{N_x} \otimes \bar{D}_{N_y}^d, \quad i = 1, 2, \ldots, N_x - 1,$$

where t_i, $\tilde{h}_i$ and c_i are as in (11). Let e_i be the i-th column of the identity matrix I_{N_x}. Note the fact that

$$(e_i^{\intercal} \otimes I_{N_y})\hat{M} = (e_i^{\intercal} \otimes I_{N_y})\hat{K}_i.$$

As in one-dimensional case, we use the following approximation

$$(e_i^{\intercal} \otimes I_{N_y})\hat{M}^{-1} \approx (e_i^{\intercal} \otimes I_{N_y})\hat{K}_i^{-1},$$

which means that the $((i-1)(N_y-1)+1)$-th to $i(N_y-1)$-th rows of the inverse of $\hat{M}$ are approximated by the $((i-1)(N_y-1)+1)$-th to $i(N_y-1)$-th rows of the inverse of $\hat{K}_i$. Therefore, similarly to P_2 in (12) for the one-dimensional case, we propose the preconditioner $\hat{P}_2$ whose inverse is defined by

$$\hat{P}_2^{-1} = (T_{N_x}^{-1} \otimes I_{N_y}) \left(\sum_{i=1}^{N_x-1} (e_i e_i^{\intercal} \otimes I_{N_y}) \hat{K}_i^{-1} \right) (T_{N_x}^{-1} H_{N_x}^{-1} \otimes I_{N_y}).$$

We note that $\hat{K}_i$ can be factored as

(16)
$$\begin{aligned}\hat{K}_i =& (S_{N_x} \otimes I_{N_y}) \left(\frac{\tilde{h}_i}{t_i^2} I_{N_x} \otimes F_{N_y}^b + \Lambda_{N_x} \otimes I_{N_y} + \frac{\tilde{h}_i c_i}{t_i^2} I_{N_x} \otimes \bar{D}_{N_y}^d \right) (S_{N_x} \otimes I_{N_y}) \\ \triangleq & (S_{N_x} \otimes I_{N_y}) C_i (S_{N_x} \otimes I_{N_y}),\end{aligned}$$

where $\Lambda_{N_x} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_{N_x-1})$ is a diagonal matrix whose diagonals are the eigenvalues of L_{N_x}, and C_i, which is a block diagonal with tridiagonal blocks matrix, denotes the middle factor in the factorization of $\hat{K}_i$. We remark that the inverse of the first and the last term in (16), multiplying any vector, can be implemented by the DST in $\mathcal{O}(N_x N_y \log N_x)$ operations, while the middle term C_i can be inverted in $\mathcal{O}(N_x N_y)$ operations; i.e.,

$$\hat{K}_i^{-1} = (S_{N_x} \otimes I_{N_y}) C_i^{-1} (S_{N_x} \otimes I_{N_y}).$$

Nevertheless, it is too expensive to calculate $\hat{P}_2^{-1}$ since we need to compute about N_x inverses of $\hat{K}_i$ $(i = 1, 2, \ldots, N_x - 1)$. In order to reduce the computational workload, as in the one-dimensional case, we propose to exploit the interpolation method to construct the practical preconditioner.

Denote $C_i = \mathrm{diag}(C_{i1}, C_{i2}, \ldots, C_{i(N_x-1)})$ where C_{ik} is an $(N_y-1) \times (N_y-1)$ tridiagonal matrix as follows,

$$C_{ik} = \frac{\tilde{h}_i}{t_i^2} F_{N_y}^b + \frac{\tilde{h}_i c_i}{t_i^2} \bar{D}_{N_y}^d + \lambda_\Lambda(\tilde{\xi}_k) I_{N_y},$$

in which $\lambda_\Lambda(\theta) = 2 - 2\cos\theta$ and $\tilde{\xi}_k = \frac{k\pi}{N_x}$. We investigate its inverse using the interpolation method.

Let

$$\tilde{Q}_i(\theta) \triangleq \left(\frac{\tilde{h}_i}{t_i^2} F_{N_y}^b + \frac{\tilde{h}_i c_i}{t_i^2} \bar{D}_{N_y}^d + \lambda_\Lambda(\theta) I_{N_y} \right)^{-1}, \qquad \theta \in (0, \pi),$$

be an $(N_y-1) \times (N_y-1)$ matrix function. Then we have $C_{ik}^{-1} = \tilde{Q}_i(\tilde{\xi}_k)$.

We choose a small number $\ell(\ell \ll N_x)$ of values $\{\theta_j\}_{j=1}^{\ell} \subset \{\tilde{\xi}_k = \frac{k\pi}{N_x}\}_{k=1}^{N_x-1}$. Let

$$\tilde{P}_i(\theta) = \phi_1(\theta)\tilde{Q}_i(\theta_1) + \phi_2(\theta)\tilde{Q}_i(\theta_2) + \cdots + \phi_\ell(\theta)\tilde{Q}_i(\theta_\ell) \tag{17}$$

be the piecewise linear interpolation for $\tilde{Q}_i(\theta)$ based on the ℓ matrices $\{\theta_j, \tilde{Q}_i(\theta_j)\}_{j=1}^{\ell}$. Thus, by interpolation formula (17) to approximate C_i^{-1}, we have

$$C_{ik}^{-1} \approx \tilde{P}_i(\tilde{\xi}_k) = \sum_{j=1}^{\ell} \phi_j(\tilde{\xi}_k)\tilde{Q}_i(\theta_j), \qquad i = 1, 2, \ldots, N_x - 1,$$

and

$$C_i^{-1} \approx \sum_{j=1}^{\ell} \operatorname{diag}\left(\phi_j(\tilde{\xi}_1), \phi_j(\tilde{\xi}_2), \ldots, \phi_j(\tilde{\xi}_{N_x-1})\right) \otimes \tilde{Q}_i(\theta_j) = \sum_{j=1}^{\ell} \Phi_j \otimes \tilde{Q}_i(\theta_j),$$

where the diagonal matrix Φ_j is as in (14). Finally, we define the practical preconditioner $\hat{P}_3$ whose inverse is defined as follows,

$$\begin{aligned}
\hat{P}_3^{-1} =& (T_{N_x}^{-1} \otimes I_{N_y}) \left[\sum_{i=1}^{N_x-1} (e_i e_i^\intercal S_{N_x} \otimes I_{N_y}) \left(\sum_{j=1}^{\ell} \Phi_j \otimes \tilde{Q}_i(\theta_j) \right) (S_{N_x} \otimes I_{N_y}) \right] \\
& \times (T_{N_x}^{-1} H_{N_x}^{-1} \otimes I_{N_y}) \\
=& (T_{N_x}^{-1} \otimes I_{N_y}) \left[\sum_{i=1}^{N_x-1} \sum_{j=1}^{\ell} \left(e_i e_i^\intercal \otimes \tilde{Q}_i(\theta_j) \right) \left(S_{N_x} \Phi_j S_{N_x} \otimes I_{N_y} \right) \right] \\
& \times (T_{N_x}^{-1} H_{N_x}^{-1} \otimes I_{N_y}) \\
=& (T_{N_x}^{-1} \otimes I_{N_y}) \left[\sum_{j=1}^{\ell} \left(\sum_{i=1}^{N_x-1} e_i e_i^\intercal \otimes \tilde{Q}_i(\theta_j) \right) \left(S_{N_x} \Phi_j S_{N_x} \otimes I_{N_y} \right) \right] \\
& \times (T_{N_x}^{-1} H_{N_x}^{-1} \otimes I_{N_y}) \\
=& (T_{N_x}^{-1} \otimes I_{N_y}) \left[\sum_{j=1}^{\ell} \tilde{W}_j (S_{N_x} \Phi_j S_{N_x} \otimes I_{N_y}) \right] \left(T_{N_x}^{-1} H_{N_x}^{-1} \otimes I_{N_y} \right),
\end{aligned} \tag{18}$$

where $\tilde{W}_j = \operatorname{diag}(\tilde{Q}_1(\theta_j), \tilde{Q}_2(\theta_j), \ldots, \tilde{Q}_{N_x-1}(\theta_j)), j = 1, 2, \ldots, \ell$, are block diagonal matrices in which each block is the inverse of a tridiagonal matrix. Therefore, $\tilde{W}_j$ can be inverted with only $\mathcal{O}(N_x N_y)$ operations. Applying $\hat{P}_3^{-1}$ to multiply any vector requires $\mathcal{O}(\ell N_x N_y \log N_x)$ operations which is acceptable for a small number ℓ. It is expected that as ℓ, the number of interpolation nodes, increases, the number of iterations required for convergence decreases. However, the cost of forming and applying the preconditioner grows proportionally to ℓ. Hence there is a trade-off to determine a suitable number of interpolation points.

4. Numerical experiments

In this section, we carry out numerical experiments to study the performance of the proposed preconditioner $\hat{P}_3$ in (18). We employ the preconditioned GMRES method to solve the collocation system. In all numerical experiments, the stopping criterion is $\frac{||r_k||_2}{||r_0||_2} < 10^{-10}$, where r_k is the residual vector after k iterations and r_0 is the initial residual vector. All numerical experiments are implemented using Matlab on a Dell Optiplex 3020 with the configuration: Intel(R) Core(TM) CPU i5-4590 3.30 GHz and 8.00 GB of memory.

TABLE 1. Numerical results for $a(x) = b(y) = 1$.

$N_x = N_y$	$\hat{P}_3(\ell = 8)$		$\hat{P}_3(\ell = 10)$		$\hat{P}_3(\ell = 12)$		$\hat{P}_3(\ell = 14)$		Ave-GMRES	
	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
64	16	0.34	15	0.26	15	0.29	15	0.33	300	2.66
128	17	0.90	16	0.96	16	1.09	15	1.15	685	15.89
256	18	3.66	17	4.02	16	4.38	16	4.96	1532	123.65
512	19	16.20	18	18.33	17	20.20	17	22.95	3494	1621.91
1024	20	63.95	19	73.39	18	81.29	17	88.26	-	-

TABLE 2. Numerical results for $a(x) = e^x,\ b(y) = 1$.

$N_x = N_y$	$\hat{P}_3(\ell = 8)$		$\hat{P}_3(\ell = 10)$		$\hat{P}_3(\ell = 12)$		$\hat{P}_3(\ell = 14)$		Ave-GMRES	
	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
64	16	0.33	16	0.29	16	0.32	16	0.35	549	4.98
128	18	0.94	17	1.04	17	1.17	16	1.23	1362	33.40
256	20	4.06	18	4.31	17	4.70	17	5.28	3627	304.40
512	21	18.15	19	19.60	18	21.57	18	24.50	-	-
1024	22	70.91	20	78.06	19	86.49	18	93.99	-	-

We consider the two-dimensional elliptic equation (1) with the source term $f(x, y) = 1$. Note that no approximation is used in the y-direction in deriving the preconditioner $\hat{P}_3$ which indicates that the coefficient function $b(y)$ will not affect the convergence rate of the preconditioner. Therefore we set $b(y) = 1$ in all the experiments. On the other hand, the coefficient function $c(x)d(y)$ is to add information to the main diagonal of the system matrix. $c(x)d(y) = 0$ is chosen to demonstrate the bad conditional cases. In consequence of the weak singularities of the original function, the interpolation nodes are selected as

$$\theta_j = \frac{\pi}{N_x}\left\lceil (N_x - 1)^{\frac{j-1}{\ell-1}} \right\rceil, \quad j = 1, 2, \ldots, \ell,$$

where $\lceil x \rceil$ denotes the ceiling of x. We remark the every θ_j should be different, and we set $\theta_r = \theta_{r-1} + \pi/N_x$ once $\theta_r \equiv \theta_{r-1}$.

For the purpose of comparisons, we take the average of each diagonal of the finite difference preconditioner, resulting in a block tridiagonal Toeplitz with tridiagonal Toeplitz blocks structured preconditioner for the collocation system which is denoted by "Ave-GMRES".

The numerical results are listed in Table 1-4, where "$\hat{P}_3(\ell = 8, 10, 12, 14)$" denotes the GMRES method with the preconditioner $\hat{P}_3$ with ℓ being the number of interpolation nodes, "Iter" denotes the number of iterations required to solve (1), "CPU" denotes the CPU time in seconds for solving the discretized system, and "-" means that the methods do not converge within 6000 iterations.

In Table 1, the numerical results are reported with constant coefficient $a(x) = 1$. We see that the preconditioned GMRES methods exhibit excellent performance both in terms of iteration steps and CPU time, and the iteration number only increases slightly as the number of grid points increases. The number of iterations decreases as expected while the number of interpolation nodes increases.

Table 2 and 3 list the numerical results for non-constant coefficients. The results with coefficient $a(x) = e^x$ is tested in Table 2. We observe that $\ell = 8$ provides better results in terms of the iteration number and CPU time. In Table 3, we list the results

TABLE 3. Numerical results for $a(x)$ in large variation case and $b(y) = 1$ with $\ell = 8$.

$a(x)$	e^{6x}		$10^3 \cos(x)$		$\cos(8\pi x) + 10$	
$N_x = N_y$	Iter	CPU	Iter	CPU	Iter	CPU
64	21	0.36	16	0.29	18	0.33
128	22	1.15	16	0.85	19	0.99
256	23	4.63	17	3.44	20	4.03
512	24	20.58	19	16.36	20	17.23
1024	25	80.18	19	61.19	22	70.57

TABLE 4. Numerical results for $a(x, y) = b(x, y) = e^{(x+y)} + 1$.

$N_x = N_y$	$\hat{P}_3(\ell = 5)$		$\hat{P}_3(\ell = 6)$		$\hat{P}_3(\ell = 7)$	
	Iter	CPU	Iter	CPU	Iter	CPU
64	39	0.44	40	0.47	40	0.49
128	42	1.37	42	1.44	42	1.52
256	44	5.42	41	5.43	42	6.01
512	45	26.39	43	27.24	43	29.47

with $a(x)$ being functions with large variations. We observe that the preconditioners is not sensitive for problems with coefficients having large variations.

In the last example, we examine the effectiveness of the preconditioner to non-separable elliptic equations:

$$-\frac{\partial}{\partial x}\left(a(x,y)\frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(b(x,y)\frac{\partial u}{\partial y}\right) = f(x,y) \quad \text{in } \Omega = (-1,1)^2.$$

For problems with non-separable coefficients, we build the preconditioner by using averages of the coefficients as

$$\bar{a}(x) = \frac{1}{2}\int_{-1}^{1} a(x,y)dy,$$

and $\bar{b}(y)$ in the similar way. Then our preconditioners could be applied to the relative separable coefficient systems. Numerical results are given for $a(x, y) = b(x, y) = e^{(x+y)} + 1$ in Table 4. We observe that the iteration numbers are larger than the separable case, but are still acceptable.

We remark that the current algorithm is based on the approximation in x direction. However, problem (1) is symmetric with respect to x and y. The roles of x and y can be switched to get an alternative algorithm. When $N_y < N_x$, the alternative one would be less computational expensive.

5. Concluding remarks

The main contribution of this paper is to develop a preconditioner based on the approximate inverses in Chebyshev collocation method for two-dimensional elliptic equations. The complexity of the matrix-vector multiplication of $\hat{P}_3^{-1}$ is of $\mathcal{O}(\ell N_x N_y \log N_x)$. It is shown numerically that the preconditioned GMRES method for solving these preconditioned collocation systems converges very quickly.

We only considered two-dimensional case in this paper. But since one direction is approximated in the two-dimensional case, our strategy for constructing preconditioners can be easily extended to three-dimensional cases. Indeed, in the three-dimensional case, we can first apply the approximation (7) to x and y directions of the finite difference operator, resulting in a scaled tensor product. Using the row-by-row approximation, we can obtain the second preconditioner, which is of inverted form. Finally, by defining a two-dimensional tensor function, the interpolation method can be utilized to construct a practical preconditioner which requires only $\mathcal{O}(\ell N_x N_y N_z \log N_x N_y))$ operations.

Nevertheless, even the numerical results show the efficiency and fast convergence of the proposed method, the convergence of our algorithm has not been studied theoretically, but will be tackled in our future work.

Acknowledgments

This research was supported in part by NSF DMS-1620262 and AFOSR FA9550-16-1-0102, and in part by research grants MYRG2016-00063-FST from University of Macau and 054/2015/A2 from FDCT of Macao.

References

[1] M. Benzi, Preconditioning techniques for large linear systems: A survey, J. Comput. Phys., 182(2002), pp. 418–477.

[2] D. Gottlieb, M. Y. Hussaini, and S. A. Orszag, Introduction: Theory and Applications of Spectral Methods, in R. G. Voigt, D. Gottlieb, and M. Y. Hussaini, eds., Spectral Methods for Partial Differential Equations, SIAM, Philadelphia, 1984.

[3] D. Gottlieb and L. Lustman, The Dufort-Frankel Chebyshev method for parabolic initial boundary value problems, Comput. Fluids, 11(1983), pp. 107–120.

[4] D. Gottlieb and S. A. Orszag, Numerical Analysis of Spectral Methods: Theory and Applications, SIAM, 1977.

[5] P. Haldenwang, G. Labrosse, S. Abboudi, and M. DeVille, Chebyshev 3-D spectral and 2-D pseudospectral solvers for the Helmholtz equation, J. Comput. Phys, 55(1984), pp. 115–128.

[6] S. Kim and S. Parter, Preconditioning Chebyshev spectral collocation method for elliptic partial differential equations, SIAM J. Numer. Anal., 33(1996), pp. 2375–2400.

[7] S. Kim and S. Parter, Preconditioning Chebyshev spectral collocation by finite-differences operators, SIAM J. Numer. Anal., 34(1997), pp. 939–958.

[8] M. Ng and J. Pan, Approximate inverse circulant-plus-diagonal preconditioners for Toeplitz-plus-diagonal matrices, SIAM J. Sci. Comput., 32(2010), pp. 1442–1464.

[9] S. A. Orszag, Spectral methods for problems in complex geometries, J. Comput. Phys., 37(1980), pp. 70–92.

[10] J. Pan, R. Ke, M. Ng, and H. Sun, Preconditioning techniques for diagonal-times-Toeplitz matrices in fractional diffusion equations, SIAM J. Sci. Comput., 36(2014), pp. A2698–A2719.

[11] R. Peyret, Introduction to Spectral Methods, Von Karman Institute, Rhode-St.-Genèse, Belgium, 1986.

[12] J. Shen, T. Tang, and L. Wang, Spectral Methods: Algorithms, Analysis and Applications, Springer Series in Computational Mathematics, Springer, 2011.

[13] J. Shen, F. Wang, and J. Xu, A finite element multigrid preconditioner for Chebyshev-collocation methods, Appl. Numer. Math., 33(2000), pp. 471–477.

[14] J. A. C. Weideman and S. C. Reddy, A MATLAB Differentiation Matrix Suite, ACM Trans. Math. Software, 2000.

Department of Mathematics, University of Macau, Macao
E-mail: `fzw913@yeah.net`

Department of Mathematics, Purdue University, West Lafayette, IN 47907-1957, USA
E-mail: `shen7@purdue.edu`

Department of Mathematics, University of Macau, Macao
E-mail: `hsun@umac.mo`