AN ALL-AT-ONCE PRECONDITIONER FOR EVOLUTIONARY PARTIAL DIFFERENTIAL EQUATIONS

XUE-LEI LIN† AND MICHAEL K. NG‡

Abstract. In [McDonald, Pestana and Wathen, SIAM J. Sci. Comput., 40 (2018), pp. A1012–A1033], a block circulant preconditioner is proposed for all-at-once linear systems arising from evolutionary partial differential equations, in which the preconditioned matrix is proven to be diagonalizable and to have identity-plus-low-rank decomposition under the setting of heat equation. In this paper, we generalize the block circulant preconditioner by introducing a small parameter \( \epsilon > 0 \) into the top-right block of the block circulant preconditioner. The implementation of the generalized preconditioner requires the same computational complexity as that of the block circulant one. Theoretically, we prove that (i) the generalization preserves the diagonalizability and the identity-plus-low-rank decomposition; (ii) all eigenvalues of the new preconditioned matrix are clustered at 1 with the clustering radius positively related to \( \epsilon \); (iii) GMRES method for the preconditioned system has a linear convergence rate independent of size of the linear system when \( \epsilon \) is taken to be smaller or comparable with square root of time-step size. Numerical results are reported to confirm the efficiency of the proposed preconditioner and to show that the generalization improves the performance of block circulant preconditioner.

Key words. Evolutionary equations; all-at-once discretization; convergence of GMRES; block Toeplitz matrices; preconditioning technique

AMS subject classifications. 65F08; 65F10; 15B05; 65M22

1. Introduction. In this paper, we are particularly interested in evolutionary partial differential equations (PDEs) with first order temporal derivative. The classical time-stepping method solves the evolutionary PDEs one time level by one time level (i.e., in a fully sequential manner), which is time-consuming if the number of time levels are large. This motivates the development of parallel methods for evolutionary PDEs; see, e.g., [3, 6, 9, 14, 15, 18]. As one type of parallel methods for evolutionary PDEs, the space-time method put the linear equations at all time levels into a large linear system (all-at-once discretization) and then solve all unknowns of the large system simultaneously; see, e.g., [2, 6, 8, 10, 11, 13].

Recently, McDonald, Pestana and Wathen in [14] proposed a block circulant preconditioner to accelerate the convergence of Krylov subspace methods for solving the all-at-once linear system arising from backward-difference-time-discretization of evolutionary PDEs. It is remarkable that the preconditioned system in [14] is diagonalizable in the case of heat equation although the original all-at-once system is not diagonalizable. Such diagonalizability may not be useful in the actual implementation as the eigenvector-matrix is complicated. But it is rare to see that a preconditioned matrix has such a nice diagonalizable property that the original matrix does not possess, which may be useful in aspects of theoretical analysis. Moreover, the preconditioned matrix in [14] has an identity-plus-low-rank decomposition, which is usually related to fast convergence of GMRES method. However, in [14], the convergence of GMRES for the preconditioned system has not been proven to be independent of spatial discretization step-size yet.

In this paper, we generalize the block circulant preconditioner proposed in [14] by introducing a parameter \( \epsilon > 0 \) into the top-right block of the block circulant preconditioner.

*This research was supported by research Grants, 12306616, 12200317, 12300519, 12300218 from HKRGC GRF and 11801479 from NSFC
†Department of Mathematics, Hong Kong Baptist University
‡Department of Mathematics, The University of Hong Kong
preconditioner. We call the generalized preconditioner by block \( \epsilon \)-circulant (BEC) preconditioner (when \( \epsilon = 1 \), the BEC preconditioner is identical to block circulant preconditioner). Theoretically, we show that (i) the generalization preserves the diagonalizability and the identity-plus-low-rank decomposition; (ii) all eigenvalues of the preconditioned matrix by BEC preconditioner is clustered at 1 with the clustering radius positively related to \( \epsilon \); (iii) GMRES method (restarted or non-restarted) for the preconditioned system has a linear convergence rate independent of both temporal and spatial step-sizes when \( \epsilon \) is taken to be smaller or comparable with square root of the temporal-step size. When using Krylov subspace methods to solve the preconditioned linear system, it requires to compute the inverse of block \( \epsilon \)-circulant preconditioner multiplying some given vectors. To compute the matrix-vector multiplication efficiently, we resort to the fact that the block \( \epsilon \)-circulant preconditioner is diagonalizable by means of fast Fourier transform (FFT) with each eigen-block having the same size as that of spatial discretization matrix. That means, to compute the inverse of BEC preconditioner multiplying a given vector is equivalent to solving a block diagonal linear system in Fourier domain. If the spatial term is Laplacian operator and the uniform spatial grid is employed, then the diagonal blocks of the block diagonal linear system are further diagonalizable by fast sine transform (FST), due to which computation of inverse BEC preconditioner times a vector is fast and exact. If the spatial term consists of some more general differential operators, then we resort to some efficient iterative solvers (e.g., multigrid method) as inner spatial solver. The details of implementation of preconditioned matrix times a vector are given in Section 4, which shows that the total storage of the proposed implementation is proportional to number of unknowns and the total computational cost of the proposed implementation is proportional to number of unknowns multiplying a logarithm. GMRES method is employed to solve the preconditioned linear system. Numerical results on heat equation, convection-dominated convection diffusion are reported to show that the BEC preconditioner is efficient and it improves the performance of block circulant preconditioner.

The outline of this paper is organized as follows. In Section 2, the all-at-once linear system arising from an evolutionary PDE is presented. In Section 3, the BEC preconditioner is proposed, the properties of the preconditioned system and convergence of GMRES for the preconditioned system are analyzed. In Section 4, the implementation of preconditioned matrix-vector multiplication and the complexity of GMRES method are discussed. In Section 5, Numerical results are reported. Finally, concluding remarks are given in Section 6.

2. The All-at-Once System for Evolutionary PDEs. As in [14], we start with the following heat equation to describe our method clearly:

\[
\begin{align*}
\partial_t u(x, t) &= \nabla (a(x) \nabla u(x, t)) + f(x, t), \quad (x, t) \in \Omega \times (0, T], \quad \Omega \subset \mathbb{R}^2 \text{ or } \mathbb{R}^3, \quad (2.1) \\
u(x, t) &= g(x, t), \quad (x, t) \in \partial \Omega, \quad (2.2) \\
u(x, 0) &= u_0(x), \quad x \in \bar{\Omega}, \quad (2.3)
\end{align*}
\]

where \( \Omega \) is open, \( \partial \Omega \) denotes boundary of \( \Omega \), \( f \), \( g \) and \( u_0 \) are all given functions, \( a(x) \) is a given positive function.

For positive integer \( N \), denote \( \tau = \frac{T}{N} \) and \( t_n = n\tau \) for \( n = 0, 1, \ldots, N \). The backward difference is employed to discretize \( \partial_t \), i.e., we adopt the discretization:

\[
\partial_t u(x, t_n) \approx \frac{u(x, t_n) - u(x, t_{n-1})}{\tau}, \quad n = 1, 2, \ldots, N \quad (2.4)
\]
Let $J$ be a positive integer. Denote the mass matrix by $M \in \mathbb{R}^{J \times J}$ and denote the discretization of $-\nabla (a(x) \nabla \cdot)$ by $K \in \mathbb{R}^{J \times J}$.

Then, (2.1)–(2.3) is discretized follows

$$M \left( \frac{u^n - u^{n-1}}{\tau} \right) + Ku^n = f^n, \quad n = 1, 2, ..., N,$$

(2.5)

where $f^n (n = 1, 2, ..., N)$ consists of discretization of $f$ and $g$. $u^0$ is discretization of $u_0$ on the spatial mesh, the unknowns $u^n (n = 1, 2, ..., N)$ are approximation of $u(\cdot, t_n)$ on the spatial mesh.

Putting the $N$ many linear systems into a large linear system, we obtain

$$Lu = f,$$

(2.6)

where

$$u = (u^1; u^2; \cdots; u^N), \quad f = (\tau f^1 + Mu^0; \tau f^2; \tau f^3; \cdots; \tau f^N),$$

$$L = \left[ \begin{array}{ccc} A_0 & \cdots & 0 \\ -M & A_0 & \cdots \\ \vdots & \ddots & \ddots \\ 0 & \cdots & -M \end{array} \right] \in \mathbb{R}^{NJ \times NJ}, \quad A_0 = M + \tau K.$$
The BEC preconditioner for the all-at-once system (2.6) is defined as

\[
P_\epsilon = \begin{bmatrix} A_0 & -\epsilon M \\ -M & A_0 \\ \vdots & \vdots \\ -M & A_0 \end{bmatrix} \in \mathbb{R}^{NJ \times NJ},
\]

where \( \epsilon > 0 \) is a parameter. When \( \epsilon = 1 \), then \( P_\epsilon \) is exactly the block circulant preconditioner proposed in [14].

It is clear that \( L \) is invertible. Moreover, since \( L \) is a block lower triangular Toeplitz matrix, \( L^{-1} \) is also a block lower triangular Toeplitz matrix, which can be rewritten as follows [14]

\[
L^{-1} = \begin{bmatrix} (L^{-1})_0 & (L^{-1})_0 & \cdots & (L^{-1})_0 \\ (L^{-1})_1 & (L^{-1})_0 & \cdots & (L^{-1})_0 \\ \vdots & \ddots & \ddots & \vdots \\ (L^{-1})_{N-1} & \cdots & (L^{-1})_1 & (L^{-1})_0 \end{bmatrix}, \quad (L^{-1})_k := (A_0^{-1}M)^kA_0^{-1}
\]

Denote by \( I_k \), the \( k \times k \) identity matrix. Let \( e_i \) be ith column of \( I_N \). Denote \( E_i = e_i \otimes I_j \).

For any Hermitian positive semi-definite matrix \( H \in \mathbb{C}^{m \times m} \), denote

\[
H^{\frac{1}{2}} := U^* \text{diag}(d_1^{\frac{1}{2}}, d_2^{\frac{1}{2}}, \ldots, d_m^{\frac{1}{2}}) U,
\]

where \( U^* \text{diag}(d_1, d_2, \ldots, d_m) U \) is unitary diagonalization of \( H \). In particular, if \( H \) is Hermitian positive definite, then we rewrite \( (H^{\frac{1}{2}})^{-1} \) as \( H^{-\frac{1}{2}} \) for notation simplification.

For any square matrix \( C \), denote by \( \sigma(C) \) the spectrum of \( C \).

**Theorem 3.1.** Let \( \epsilon \in (0, 1] \). Then, both \( P_\epsilon \) and \( Z_\epsilon := \epsilon^{-1}[I_J - \epsilon(A_0^{-1}M)^N]M^{-1} \) are invertible with \( P_\epsilon^{-1} = L^{-1} + L^{-1}E_iZ_\epsilon^{-1}E_i^T L^{-1} \).

**Proof.** By matrix similarity, we have

\[
\sigma(M^{-1}A_0) = \sigma(M^{-\frac{1}{2}}A_0M^{-\frac{1}{2}}) = \sigma(M^{-\frac{1}{2}}(M + \tau K)M^{-\frac{1}{2}}) = \sigma(I_J + \tau M^{-\frac{1}{2}}KM^{-\frac{1}{2}}),
\]

which implies that \( \sigma(M^{-1}A_0) \in (1, +\infty) \). Thus, \( \sigma(A_0^{-1}M) = \sigma((M^{-1}A_0)^{-1}) \in (0, 1) \). By \( \epsilon \in (0, 1] \), we know that \( \sigma(\epsilon(A_0^{-1}M)^N) \in (0, 1) \). That means \( 0 \notin \sigma(I_J - \epsilon(A_0^{-1}M)^N) \), which proves that \( Z_\epsilon \) is invertible.

It is clear that \( P_\epsilon \) can be rewritten as \( P_\epsilon = L - \epsilon E_iME_i^T \). Using this expression of \( P_\epsilon \), it is straightforward to verify that \( P_\epsilon(L^{-1} + L^{-1}E_iZ_\epsilon^{-1}E_i^T L^{-1}) = I_{NJ} \), which shows that \( P_\epsilon \) is invertible and \( P_\epsilon^{-1} = L^{-1} + L^{-1}E_iZ_\epsilon^{-1}E_i^T L^{-1} \).

**Remark 2.** As shown in Theorem [3.1], \( \epsilon \in (0, 1] \) guarantees the invertibility of \( P_\epsilon \). Hence, throughout this paper, we choose \( \epsilon \in (0, 1] \).

With BEC preconditioner, instead of solving (2.6), we employ Krylov subspace methods to solve the preconditioned system as follows

\[
P_\epsilon^{-1}Lu = P_\epsilon^{-1}f. \quad (3.2)
\]

**Theorem 3.2.**

(i) The preconditioned matrix \( P_\epsilon^{-1}L \) has a identity-plus-low-rank decomposition, i.e.,

\[
\text{rank}(P_\epsilon^{-1}L - I_{NJ}) = J. \quad \text{Hence, } P_\epsilon^{-1}L \text{ has exactly } (N-1)J \text{ many eigenvalues equal to } 1.
\]
(ii) Given any constant \( \eta \in (0,1) \), take \( \epsilon \in (0, \eta] \). Then, \( \max_{\lambda \in \sigma(P^{-1}L)} |\lambda - 1| \leq \frac{\epsilon}{1-\eta} \).

Proof. By Theorem 3.1, \( P^{-1}L - I_{NJ} = L^{-1}E_1Z^{-1}_eE^T_N \). Then,
\[
\text{rank}(L^{-1}E_1Z^{-1}_eE^T_N) = \text{rank}(E_1Z^{-1}_eE^T_N) = J,
\]
which proves (i).

Substituting (3.1) into \( P^{-1}L = I_{NJ} + L^{-1}E_1Z^{-1}_eE^T_N \), we obtain
\[
P^{-1}L = \begin{bmatrix} I_J & (L^{-1})_0Z^{-1}_e & \cdots & (L^{-1})_{J-1}Z^{-1}_e \\ I_J & \vdots & \ddots & \vdots \\ I_J & \vdots & \cdots & I_J + (L^{-1})_{N-1}Z^{-1}_e \end{bmatrix} \tag{3.3}
\]
Therefore, \( \sigma(P^{-1}L) = \{1\} \cup \sigma(I_J + (L^{-1})_{N-1}Z^{-1}_e) \). And then,
\[
\max_{\lambda \in \sigma(P^{-1}L)} |\lambda - 1| = \max_{\lambda \in \sigma(I_J + (L^{-1})_{N-1}Z^{-1}_e)} |\lambda - 1|.
\]
It thus remains to investigate \( \sigma(I_J + (L^{-1})_{N-1}Z^{-1}_e) \). By (3.1) and definition of \( Z_e \) given in Theorem 3.1,
\[
I_J + (L^{-1})_{N-1}Z^{-1}_e = I_J + \epsilon[(M^{-1}A_0)^N - \epsilon I_J]^{-1}, \tag{3.4}
\]
which implies that
\[
\sigma(I_J + (L^{-1})_{N-1}Z^{-1}_e) = \left\{1 + \epsilon(\lambda^N - \epsilon)^{-1}|\lambda \in \sigma(M^{-1}A_0)\right\} = \left\{\frac{\lambda^N}{\lambda^N - \epsilon} |\lambda \in \sigma(I_J + \tau M^{-\frac{1}{2}}KM^{-\frac{1}{2}})\right\} \subset \left\{\frac{\lambda^N}{\lambda^N - \epsilon} |\lambda \in (1, +\infty)\right\}.
\]
Hence,
\[
\max_{\lambda \in \sigma(I_J + (L^{-1})_{N-1}Z^{-1}_e)} |\lambda - 1| \leq \sup_{\lambda \in (1, +\infty)} \left|\frac{\lambda^N}{\lambda^N - \epsilon} - 1\right| = \sup_{\lambda \in (1, +\infty)} \left|\frac{\epsilon}{\lambda^N - \epsilon}\right| \leq \frac{\epsilon}{1-\eta},
\]
which completes the proof. \( \square \)

Theorem 3.2 (i) implies that by using GMRES method, the exact solution of the preconditioned system (3.2) can be found within at most \( J + 1 \) iterations. But this is not a sharp estimation of convergence rate of GMRES method when \( J \) is not small. In Theorem 3, we will show that GMRES method for the system (3.2) has a linear convergence rate independent of \( N \) and \( J \) when \( \epsilon \lesssim \sqrt{\tau} \). Theorem 3.2 (ii) shows that all the eigenvalues of the preconditioned matrix are clustered at 1 with clustering radius of \( O(\epsilon) \).

**Lemma 3.3.** There exists an invertible matrix \( V \in \mathbb{R}^{J \times J} \) and a diagonal matrix \( D \in \mathbb{R}^{J \times J} \) such that \( I_J + (L^{-1})_{N-1}Z^{-1}_e = VDV^{-1} \) and \( 1 \notin \sigma(D) \).

**Proof.** Denote \( H_0 := I_J + \epsilon[(M^{-\frac{1}{2}}A_0M^{-\frac{1}{2}})^N - \epsilon I_J]^{-1} \). From (3.4), we know that
\[
I_J + (L^{-1})_{N-1}Z^{-1}_e = I_J + \epsilon[(M^{-1}A_0)^N - \epsilon I_J]^{-1} = M^{-\frac{1}{2}}H_0M^{\frac{1}{2}}.
\]
Since $M^{-\frac{1}{2}}A_0M^{-\frac{1}{2}}$ is real symmetric, so is $H_0$. Thus, $H_0$ is orthogonally diagonalizable, i.e., there exists an orthogonal matrix $Q \in \mathbb{R}^{J \times J}$ and a diagonal matrix $D \in \mathbb{R}^{J \times J}$ such that $H_0 = QDQ^T$. Letting $V = M^{-\frac{1}{2}}Q$, we then obtain $I_J + (L^{-1})_{N-1}Z_{e}^{-1} = VDV^{-1} = VD^{-1}$.

By $H_0 = QDQ^T$, definition of $H_0$ and $\epsilon(0, 1]$, we know that

$$
\sigma(D) = \sigma(H_0) = \left\{ \frac{\lambda^N}{\lambda^N - \epsilon} \mid \lambda \in \sigma(M^{-\frac{1}{2}}A_0M^{-\frac{1}{2}}) \right\} = \left\{ \frac{\lambda^N}{\lambda^N - \epsilon} \mid \lambda \in \sigma(I_J + \tau M^{-\frac{1}{2}}KM^{-\frac{1}{2}}) \right\}
\subset (1, +\infty),
$$

which means $1 \notin \sigma(D)$. □

**Theorem 3.4.** The preconditioned matrix $P^{-1}L$ is diagonalizable, i.e.,

$$
P^{-1}_eL = \hat{V}DV^{-1},
$$

where

$$
\hat{V} = \begin{bmatrix} I_J & & & \end{bmatrix}, \quad \hat{D} = \begin{bmatrix} I_J & & & \end{bmatrix},
$$

$$
V_i = (L^{-1})_iZ_{e}^{-1}V(I_J - D)^{-1}, \quad i = 0, 1, ..., N - 2.
$$

with $V$ and $D$ given by Lemma 3.3.

**Proof.** By Lemma 3.3 $1 \notin \sigma(D)$, i.e., $I_J - D$ is invertible. Thus, $V_i$ ($i = 0, 1, ..., N - 2$) are well-defined. Then, it is straightforward to verify that $L\hat{V} = P_eVD$. Moreover, invertibility of $V$ guarantees the invertibility of $\hat{V}$. That means $P^{-1}_eL = \hat{V}DV^{-1}$. The proof is complete. □

For any Hermitian matrices $H_1, H_2 \in \mathbb{C}^{m \times m}$, denote $H_2 \succ (\succeq) H_1$ if $H_2 - H_1$ is Hermitian positive definite (or Hermitian positive semi-definite). Also, $H_1 \prec (\preceq) H_2$ has the same meaning as that of $H_2 \succ (\succeq) H_1$.

Let $O$ denote zero matrix with proper size.

**Lemma 3.5.** Given any $\eta \in (0, 1)$, choose $\epsilon \in (0, \eta]$. Then,

$$
\|P^{-1}_eL - I_{N,J}\|_2 \leq c_0 \sqrt{\frac{N}{1 - \eta}},
$$

where $c_0 := \sup_{J \in \mathbb{N}^+} \kappa_2(M^2) = \sqrt{\sup_{J \in \mathbb{N}^+} \kappa_2(M)} < +\infty$ is independent of $J$ and $N$.

**Proof.** As $M^2A_0^{-1}M^2$ is real symmetric, $M^2A_0^{-1}M^2$ is orthogonally diagonalizable, i.e., there exists an orthogonal matrix $Q \in \mathbb{R}^{J \times J}$ and a diagonal matrix $A \in \mathbb{R}^{J \times J}$ such that $M^2A_0^{-1}M^2 = QAQ^T$. Since $\sigma(D) = \sigma(M^2A_0^{-1}M^2)$, $M^2A_0^{-1}M^2 = [I_J + \tau M^{-\frac{1}{2}}KM^{-\frac{1}{2}}]^{-1}$ implies that $O \preceq A \preceq I_J$. Then, by (3.1) and definition of $Z_{e}$, we have

$$
(L^{-1})_kZ_{e}^{-1} = \epsilon(A_0^{-1}M)^{k+1}[I_J - \epsilon(A_0^{-1}M)^N]^{-1}
\supseteq \epsilon M^{-\frac{1}{2}}(M^2A_0^{-1}M^2)^{k+1}[I_J - \epsilon(M^2A_0^{-1}M^2)^N]^{-1}M^2
\supseteq \epsilon M^{-\frac{1}{2}}QA^{k+1}[I_J - \epsilon A^N]^{-1}QM^2, \quad k = 0, 1, ..., N - 1,
$$
which together with \(3.3\) implies that

\[
P^{-1}_e L - I_{N,J} = \begin{bmatrix}
(L^{-1})_0 Z_e^{-1} \\
(L^{-1})_1 Z_e^{-1} \\
\vdots \\
(L^{-1})_{N-1} Z_e^{-1}
\end{bmatrix}
\]

\[
= \epsilon [I_N \otimes (M^{-\frac{1}{2}} Q)] \begin{bmatrix}
\Lambda^1 [I_J - \epsilon \Lambda^N]^{-1} \\
\Lambda^2 [I_J - \epsilon \Lambda^N]^{-1} \\
\vdots \\
\Lambda^N [I_J - \epsilon \Lambda^N]^{-1}
\end{bmatrix} [I_N \otimes (Q^T M^\frac{1}{2})].
\]

Rewrite \( \Lambda = \text{diag}(\lambda_i)_{i=1}^J \). Then,

\[
||P^{-1}_e L - I_{N,J}||_2 \leq \epsilon ||I_N \otimes (M^{-\frac{1}{2}} Q)||_2 ||I_N \otimes (Q^T M^\frac{1}{2})||_2 \sqrt{\sum_{k=1}^N \Lambda^2 k (I_J - \epsilon \Lambda^N)^{-2}}
\]

\[
= \epsilon \kappa_2 (M^\frac{1}{2}) \left[ \max_{1 \leq i \leq J} \sum_{k=1}^N \left( \frac{\lambda_i^k}{1 - \epsilon \lambda_i^N} \right)^2 \right] \leq \epsilon c_0 \sqrt{\sum_{k=1}^N \lambda_i^k} \cdot \max_{1 \leq i \leq J} \sum_{k=1}^N \left( \frac{\lambda_i^k}{1 - \epsilon \lambda_i^N} \right)^2.
\]

Moreover, it is easy to check that the functions \(g_k(x) := \frac{x^k}{1 - \epsilon x^N}\) is monotonically increasing on \(x \in [0, 1]\) for each \(k = 1, 2, ..., N\). Since \(O < A \leq I_N\), \(\{\lambda_i|1 \leq i \leq J\} \subseteq [0, 1]\). Hence,

\[
||P^{-1}_e L - I_{N,J}||_2 \leq \epsilon c_0 \sqrt{\sum_{k=1}^N \frac{1}{(1 - \epsilon)^2}} \leq \frac{\epsilon c_0 \sqrt{N}}{1 - \epsilon} \leq \frac{\epsilon c_0 \sqrt{N}}{1 - \eta},
\]

which completes the proof \[\square\]

For any matrix \(Z \in \mathbb{R}^{m \times m}\), denote

\[
H(Z) := \frac{Z + Z^T}{2}, \quad S(Z) := \frac{Z - Z^T}{2}.
\]

Let \(\lambda_{\text{min}}(\cdot)\) and \(\lambda_{\text{max}}(\cdot)\) denote the minimal and maximal eigenvalue of a Hermitian matrix, respectively. Let \(\rho(\cdot)\) denotes the spectral radius of a square matrix.

**Lemma 3.6.** (see [1, (1.1)]) Let \(ZQ = w\) be a real square linear system with \(Z > O\). Then, the residuals of the iterates generated by applying (restarted or non-restarted) GMRES to solving \(Zv = w\) satisfy

\[
||r_k||_2 \leq \left(1 - \frac{\lambda_{\text{min}}(H(Z))^2}{||Z||_2^2}\right)^{k/2} ||r_0||_2,
\]

where \(r_k = w - Zq_k\) with \(q_k\) (\(k \geq 1\)) being the iterate solution at \(k\)th GMRES iteration and \(q_0\) being an arbitrary initial guess.

**Theorem 3.7.** For any given constants \(\delta \in (0, 1)\), choose \(\epsilon \in (0, b_\epsilon)\), where

\[
b_\epsilon := \frac{\lambda_{\text{max}}}{\sqrt{\lambda_{\text{max}} + \epsilon}}
\]

and \(c_0\) is given by Lemma \[3.3\]. Then, the residuals of the iterates generated by applying (restarted or non-restarted) GMRES to solving the preconditioned system \(3.2\) satisfy

\[
||r_k||_2 \leq \left(\frac{2\sqrt{\delta}}{1 + \delta}\right)^k ||r_0||_2,
\]
where \( r_k = P^{-1}\mathbf{f} - P^{-1}\mathbf{L}u_k \) with \( u_k \) \((k \geq 1)\) being the iterative solution at \( k \)th GMRES iteration and \( u_0 \) denoting an arbitrary initial guess.

**Proof.** Denote \( \Xi = P^{-1}\mathbf{L} - \mathbf{I}_{NJ} \). Since \( b_\tau \in (0, 1) \), Lemma 3.5 is applicable. By Lemma 3.5, we have

\[
||\Xi||_2 \leq \epsilon_c 0 \sqrt{\frac{N}{1 - b_\tau}} = \delta.
\]

Then,

\[
\mathcal{H}(P^{-1}\mathbf{L}) = \mathbf{I}_{NJ} + \mathcal{H}(\Xi) \succeq (1 - \delta)\mathbf{I}_{NJ} \succ 0,
\]

implies that Lemma 3.6 is applicable to the preconditioned system (3.2). It remains to estimate \( \lambda_{\min}(\mathcal{H}(P^{-1}\mathbf{L}))^2 \) and \( ||P^{-1}\mathbf{L}||_2^2 \).

Clearly, (3.5) implies that \( \lambda_{\min}(\mathcal{H}(P^{-1}\mathbf{L}))^2 \geq (1 - \delta)^2 \).

Moreover,

\[
||P^{-1}\mathbf{L}||_2^2 = ||(P^{-1}\mathbf{L})^T P^{-1}\mathbf{L}||_2 = ||\mathbf{I}_{NJ} + \Xi + \Xi^T + \Xi^T \Xi||_2 \leq (1 + 2\delta + \delta^2) = (1 + \delta)^2.
\]

Then, Lemma 3.6 implies that

\[
||r_k||_2 \leq \left(1 - \frac{\lambda_{\min}(\mathcal{H}(P^{-1}\mathbf{L}))^2}{||P^{-1}\mathbf{L}||_2^2}\right)||r_0||_2
\]

\[
\leq \left(1 - \frac{(1 - \delta)^2}{(1 + \delta)^2}\right)^{k/2} ||r_0||_2 = \left(\frac{2\sqrt{\delta}}{1 + \delta}\right)^k ||r_0||_2,
\]

which completes the proof. \( \square \)

**Remark 3.** Theorem 3.7 shows that GMRES for the preconditioned system (3.2) has a linear convergence rate independent of system size whenever \( \epsilon \ll \sqrt{\tau} \). Nevertheless, it will be presented in Section 4 that the actual implementation of GMRES involves \( \epsilon^{-1} \) times some numbers (referring to \( \mathbf{D}_\epsilon^{-1} \) in Section 4). Hence, \( \epsilon \) cannot be arbitrarily close to 0 or otherwise it will lead to numerical instability that brings large round error to the iterative solution. Actually, as illustrated by numerical results in Section 5, taking \( \epsilon = \mathcal{O}(\tau) \) already leads to a fast convergence of GMRES.

**4. Implementation.** In this section, we discuss on how to efficiently implement the GMRES method for the preconditioned system (3.2). In GMRES iteration, it requires to compute the matrix-vector product, \( P^{-1}(\mathbf{L}v) \) for some given vector \( v \). In this section, we present a fast implementation for computing the matrix-vector product. Since our presented fast implementation also works when \( \partial_t \) is discretized by multi-step backward difference, we start with multi-step-backward-difference discretization of \( \partial_t \) to describe the fast implementation.

Discretizing \( \partial_t \) by a \( p \)-step backward difference scheme, then the corresponding \( \mathbf{L} \) is as follows (14)

\[
\mathbf{L} = \mathbf{R} \otimes \mathbf{M} + \tau \mathbf{I}_N \otimes \mathbf{K},
\]

(4.1)
where ‘⊗’ denotes the Kronecker product,

\[
R := \begin{bmatrix}
  r_0 &   &   &   &   &   \\
  r_1 & r_0 &   &   &   &   \\
  \vdots & \ddots & \ddots &   &   &   \\
  r_p & \ddots & \ddots & \ddots &   &   \\
  \vdots & \ddots & \ddots & \ddots & \ddots &   \\
  r_0 &   & \cdots & r_1 & r_0 &   \\
  r_0 & r_1 & \cdots & \cdots & \cdots & r_0 \\
\end{bmatrix} \in \mathbb{R}^{N \times N}
\]

Note that if \( p = 1, r_0 = 1, r_1 = -1 \), then the \( p \)-step scheme reduces to the backward difference scheme presented in Section 2. For \( p \)-step scheme, the corresponding BEC preconditioner \( P_\epsilon \) is defined as follows

\[
P_\epsilon = R_\epsilon \otimes M + \tau I_N \otimes K,
\]

where

\[
R_\epsilon = \begin{bmatrix}
  r_0 & \epsilon r_p & \cdots & \epsilon r_2 & \epsilon r_1 \\
  r_1 & r_0 & \cdots & \cdots & \epsilon r_2 \\
  \vdots & \ddots & \ddots & \ddots & \ddots \\
  r_p & \cdots & r_0 & \cdots & \epsilon r_p \\
  \vdots & \ddots & \ddots & \ddots & \ddots \\
  r_0 & \cdots & r_1 & r_0 & \cdots \\
  r_0 & \cdots & r_1 & r_0 & \cdots \\
\end{bmatrix} \in \mathbb{R}^{N \times N}.
\]

For a given vector \( v \in \mathbb{R}^{NJ \times 1} \), to compute \( P_\epsilon^{-1}Lv \) is equivalent to compute \( \tilde{v} = Lv \) and \( P_\epsilon^{-1}\tilde{v} \). Hence, to compute the preconditioned-matrix-vector product efficiently, it suffices to compute both \( P_\epsilon^{-1}v \) and \( Lv \) efficiently for a given vector \( v \).

We firstly discuss the fast computation of \( \quadLv \) for a given vector \( v \). Notice that when \( p \) is small, \( L \) is a sparse matrix, in the case of which the computation of \( \quadLv \) requires \( O(NJ) \) storage and operations. When \( p \) is large, one can exploit the fact that \( R \) is a Toeplitz matrix whose matrix-vector product can be fast computed using fast Fourier transform (FFT); see, e.g., [16]. Using FFTs and properties of Kronecker product, the computation of \( \quadLv \) requires \( O(JN \log N) \) operations and \( O(JN) \) storage no matter how big \( p \) is.

Now, we focus on the fast computation of \( P_\epsilon^{-1}y \) for a given vector \( y \in \mathbb{R}^{NJ \times 1} \). To this end, we exploit a remarkable property of the matrix \( R_\epsilon \), i.e., its diagonalizable property. From [2, Theorem 2.10], we know that \( R_\epsilon \) can be diagonalized as follows

\[
R_\epsilon = D_\epsilon^{-1}F_N^*A_\epsilon F_N D_\epsilon,
\]

where

\[
F_N = \frac{1}{\sqrt{N}} \left[ \theta^{(i-1)(j-1)} \right]_{i,j=1}^N, \quad \theta := \exp \left( \frac{2\pi i}{N} \right), \quad i = \sqrt{-1}.
\]

\[
D_\epsilon = \text{diag} \left( \epsilon^{\frac{i}{N}}, \epsilon^{\frac{2i}{N}}, \ldots, \epsilon^{\frac{(N-1)i}{N}} \right), \quad A_\epsilon = \text{diag}(\lambda_0^{(\epsilon)}, \lambda_1^{(\epsilon)}, \ldots, \lambda_{N-1}^{(\epsilon)}),
\]
\[ \lambda_k^{(e)} = \sum_{j=0}^{p} r_j \epsilon \hat{\theta}^{kj}, \quad k = 0, 1, \ldots, N - 1. \]

\( F_N \) is called Fourier transform matrix which is unitary. \( F_N \) times a vector and \( F_N \)' times a vector can be fast computed using FFT and inverse FFT (IFFT), respectively, which requires \( O(N \log N) \) operations and \( O(N) \) storage.

When \( p \) is small, then it is clear that the computation of \( \{ \lambda_k^{(e)} \}_{k=0}^{N-1} \) requires \( O(N) \) operations and storage. When \( p \) is large, one can exploit the fact that
\[ \left( \lambda_0^{(e)}, \lambda_1^{(e)}, \ldots, \lambda_{N-1}^{(e)} \right)^T = \sqrt{N} \mathbf{F} (r_0 \epsilon \hat{\theta}, r_1 \epsilon \hat{\theta}, \ldots, r_p \epsilon \hat{\theta}, 0, 0, \ldots, 0)^T. \]

Hence, using IFFTs, FFTs and properties of Kronecker product, it is easy to see that (4.5) and (4.7) requires \( O(\log N) \) operations and \( O(N) \) storage, no matter how big \( p \) is.

By (4.3), \( \mathbf{P}_r \) can be rewritten as the following block diagonalization form
\[ \mathbf{P}_r = [D_\epsilon^{-1} F_N] \otimes \mathbf{I}_J \text{blockdiag}(\mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_{N-1})[(F_N \mathbf{D}_r) \otimes \mathbf{I}_J], \quad (4.4) \]

where
\[ \mathbf{B}_k = \lambda_k^{(e)} \mathbf{M} + \tau \mathbf{K}, \quad k = 0, 1, \ldots, N - 1. \]

Let \( \mathbf{y} = (y_1; y_2; \cdots; y_N) \in \mathbb{R}^{N \times J} \) with \( y_k \in \mathbb{R}^{J \times 1} \) \((k = 1, 2, \ldots, N)\) be a given vector. Then, the computation of \( \mathbf{z} = \mathbf{P}_r^{-1} \mathbf{y} \) can be equivalently rewritten as the following 3 steps:

**Step 1:** Compute \( \tilde{\mathbf{y}} = [(F_N \mathbf{D}_r) \otimes \mathbf{I}_J] \mathbf{y}, \quad (4.5) \)

**Step 2:** Solve \( \mathbf{B}_{k-1} \tilde{\mathbf{z}}^k = \tilde{\mathbf{y}}^k \) for \( \tilde{\mathbf{z}}^k \), \( k = 1, 2, \ldots, N \), where \( \tilde{\mathbf{y}} = (\tilde{\mathbf{y}}^1; \tilde{\mathbf{y}}^2; \cdots; \tilde{\mathbf{y}}^N) = \tilde{\mathbf{y}}, \quad (4.6) \)

**Step 3:** Compute \( \mathbf{z} = [(D_\epsilon^{-1} F_N) \otimes \mathbf{I}_J] \tilde{\mathbf{z}} \), where \( \tilde{\mathbf{z}} = (\tilde{\mathbf{z}}^1; \tilde{\mathbf{z}}^2; \cdots; \tilde{\mathbf{z}}^N). \quad (4.7) \)

Using IFFTs, FFTs and properties of Kronecker product, it is easy to see that (4.5) and (4.7) requires \( O(J \log N) \) operations and \( O(JN) \) storage. If the spatial discretization is finite difference method or finite element method with uniform square grid and the diffusion coefficient function \( a \) is a constant, then \( \mathbf{B}_k \) are all diagonalizable by means of fast sine transform (see [4]), in the case of which the \( N \) many linear systems in (4.6) can be fast and directly solved with \( O(NJ \log J) \) operations and \( O(NJ) \) storage. In a more general situation that \( \mathbf{B}_k \) are not diagonalizable, one can use some efficient spatial solvers, such as multigrid method to solve the linear systems in (4.6), for which only a few iteration is required since \( \mathbf{P}_r \) serves as a preconditioner. Actually, as illustrated by numerical results in Section 5, one iteration of V-cycle geometric multigrid method for each linear system in (4.6) already leads to a fast convergence of GMRES for the preconditioned system. Solving the linear systems in (4.6) by V-cycle multigrid method with a fixed iteration number, it requires \( O(NJ) \) operations and storage.

It is remarkable to note that only half of the \( N \) many systems in (4.6) need to be solved, the reason of which is explained as follows. From (4.5), we know that the right hand sides in (4.6) can be expressed as
\[ \tilde{\mathbf{y}}^{k+1} = \frac{1}{\sqrt{N}} \sum_{j=0}^{p} \epsilon \hat{\theta}^{kj}, \quad k = 0, 1, \ldots, N - 1. \]
Recall that the matrices in (4.6) have the following expressions

\[
B_k = \left( \sum_{j=0}^{p} r_j \hat{\tau}^{-k} \hat{\theta}^{kj} \right) M + \tau K, \quad k = 0, 1, \ldots, N - 1.
\]

Let \( \text{conj}(\cdot) \) denote conjugate of a matrix or a vector. Then,

\[
\text{conj}(\hat{y}^{k+1}) = \frac{1}{\sqrt{N}} \sum_{j=0}^{p} \epsilon \hat{\tau}^{-kj} \hat{\theta}^{N-k} y^{j+1} = \frac{1}{\sqrt{N}} \sum_{j=0}^{p} \epsilon \hat{\tau}^{(N-k)j} y^{j+1} = \hat{y}^{N-k+1}, \quad 1 \leq k \leq N - 1,
\]

\[
\text{conj}(B_k) = \left( \sum_{j=0}^{p} r_j \hat{\tau}^{-k} \hat{\theta}^{-kj} \right) M + \tau K = \left( \sum_{j=0}^{p} r_j \hat{\tau}^{N-k} \hat{\theta}^{(N-k)j} \right) M + \tau K = B_{N-k}, \quad 1 \leq k \leq N - 1.
\]

That means the unknowns in (4.6) hold equalities: \( \hat{z}^{k+1} = \text{conj}(\hat{z}^{N-k+1}) \) for \( k = 1, 2, \ldots, N - 1 \). Hence, only the first \( \lceil \frac{N+1}{2} \rceil \) many linear systems in (4.6) need to be solved.

Hence, when \( M \) and \( K \) are diagonalizable by fast sine transform, then the computation of \( P^{-1}_\epsilon Lv \) for a given vector \( v \) can be fast and exactly implemented, which requires \( O(NJ) \) storage and \( O(NJ \log J) \) operations. In other more general cases, the computation of \( P^{-1}_\epsilon Lv \) for a given vector \( v \) can be approximately implemented by \( Vy \)-cycle multigrid method with a fixed iteration number, which requires \( O(NJ) \) operations and storage. Hence, each iteration of restarted GMRES method for the preconditioned system requires \( O(NJ) \) storage and \( O(NJ) \) operations if the systems in (4.6) are solved by \( V \)-cycle multigrid method while it requires \( O(NJ \log J) \) operations and \( O(NJ) \) storage if the systems in (4.6) are diagonalizable by sine transform and solved by using fast sine transform.

**Remark 4.** As preconditioner, the invertibility of the \( p \)-step BEC matrix (4.2) should be guaranteed. Clearly, a well-defined \( p \)-step discretization matrix \( L \) defined in (4.1) is invertible. Then, from the fact that \( \lim_{\epsilon \to 0^+} ||P_\epsilon - L||_2 = 0 \), we know that for sufficiently small \( \epsilon \), \( P_\epsilon \) is invertible. From (4.3), we see that another way to guarantee the invertibility of \( P_\epsilon \) is to guarantee the invertibility of \( B_k \)'s \( (k = 0, 1, \ldots, N - 1) \). As \( B_k = \lambda_k^{(\epsilon)} M + \tau K = M^\dagger (\lambda_k^{(\epsilon)} I + \tau M^{-\dagger} K M^{-\dagger} \hat{\tau}^k) M^\dagger \) for \( k = 0, 1, \ldots, N - 1 \), it not hard to see from the definition of \( \lambda_k^{(\epsilon)} \) \( (k = 0, 1, \ldots, N - 1) \) that if \( R \) is diagonally dominant with positive \( r_0 \) and \( \epsilon \in (0, 1] \), then \( B_k \)'s are invertible. Hence, for those \( p \)-step backward difference scheme whose corresponding \( R \) is diagonally dominant with positive \( r_0 \), the corresponding BEC preconditioner \( P_\epsilon \) is unconditionally invertible for \( \epsilon \in (0, 1] \).

**5. Numerical Results.** In this section, we test the performance of the proposed BEC preconditioner through examples of heat equation, convection diffusion equation and compare it with block circulant preconditioner proposed in [14]. Finite element discretization with Q1 element and uniform square mesh is used to discretize the spatial terms of all the examples in this section. The backward difference is used as temporal discretization. All numerical experiments are performed via MATLAB R2018a on a workstation equipped with dual Xeon Gold 6146 12-Cores 3.2GHz CPUs, NVIDIA Quadro P2000 GPU, 384GB RAM running CentOS Linux version 7.

Restarted GMRES method is employed to solve the preconditioned systems. The restarting number of GMRES is set as 50. The tolerance of GMRES is set as \( ||r_k||_2 \leq \ldots \)
\[ 10^{-7} \| \mathbf{r}_0 \|_2, \] where \( \mathbf{r}_k \) denotes the residual vector at \( k \)th GMRES iteration. The zero vector is used as initial guess of GMRES method.

For convenience, the block circulant preconditioner is denoted by BC. As BC preconditioner is a special case of BEC preconditioner. Hence, we use the same algorithm for implementation of BC preconditioner as the one used for that of BEC preconditioner. We also denote GMRES with BC and BEC preconditioners by GMRES-BC and GMRES-BEC, respectively.

Denote by 'Iter', the iteration number of restarted GMRES. Denote by ‘DoF’, the degree of freedom, i.e., the number of unknowns.

Denote by ‘CPU’, the computational in unit of seconds.

For all the numerical experiments in this section, we take \( \epsilon = \min \{ 0.5, 0.5 \tau \} \) for the BEC preconditioner.

**Example 1.** The first example is heat equation (2.1)–(2.3) with
\[
\Omega = (0, 1) \times (0, 1), \quad T = 1, \quad f \equiv 0, \quad a \equiv 10^{-5}, \quad g \equiv 0, \quad u_0 = x(x-1)y(y-1).
\]
For Example 1, the corresponding \( B'_k \)s in (4.6) is diagonalizable by sine transform. Hence, we implement the matrix-vector multiplication by fast sine transform for Example 1. The results of GMRES-BEC and GMRES-BC preconditioner for solving Example 1 are listed in Table 5.1.

Table 5.1 shows that (i) GMRES-BEC converges faster than GMRES-BC does in terms of iteration number; (ii) GMRES-BEC is more efficient than GMRES-BC in terms of CPU; (iii) the convergence rate of GMRES-BEC is independent of temporal and spatial stepsizes.

| \( N \) | \( J + 1 \) | DoF       | GMRES-BEC | GMRES-BEC |
|-------|-------|-----------|------------|------------|
|       |       |           | Iter | CPU | Iter | CPU |
| \( 2^4 \) | 2\textsuperscript{7} | 258064    | 2    | 0.44 | 13   | 0.75 |
|       | 2\textsuperscript{8} | 1040400   | 2    | 0.64 | 13   | 2.63 |
|       | 2\textsuperscript{9} | 4177936   | 2    | 2.67 | 13   | 10.73 |
|       | 2\textsuperscript{10} | 16744464 | 2    | 10.67 | 13 | 43.37 |
| \( 2^6 \) | 2\textsuperscript{7} | 1032256   | 2    | 0.61 | 13   | 2.40 |
|       | 2\textsuperscript{8} | 4161600   | 2    | 2.49 | 13   | 11.04 |
|       | 2\textsuperscript{9} | 16711744  | 2    | 10.41 | 13 | 42.04 |
|       | 2\textsuperscript{10} | 66977856  | 2    | 40.02 | 13 | 162.97 |
| \( 2^8 \) | 2\textsuperscript{7} | 4129024   | 2    | 2.51 | 13   | 9.64 |
|       | 2\textsuperscript{8} | 16646400  | 2    | 10.33 | 13 | 41.89 |
|       | 2\textsuperscript{9} | 66846976  | 2    | 40.15 | 13 | 161.67 |
|       | 2\textsuperscript{10} | 267911424 | 2    | 158.00 | 13 | 637.88 |
| \( 2^{10} \) | 2\textsuperscript{7} | 16516096  | 1    | 7.76 | 13   | 42.28 |
|       | 2\textsuperscript{8} | 66585600  | 1    | 29.80 | 13 | 162.95 |
|       | 2\textsuperscript{9} | 267387904 | 1    | 118.27 | 13 | 645.30 |
|       | 2\textsuperscript{10} | 1071645696 | 1    | 470.32 | 13 | 2561.33 |

**Example 2.** The second example is also a heat equation but with variable diffusion coefficient function \( a \), which is defined as follows
\[
\Omega = (0, 1) \times (0, 1), \quad T = 1, \quad a(x, y) = 10^{-5} \times \sin(\pi xy), \quad g \equiv 0, \quad u_0 = x(x-1)y(y-1),
\]
\[ f(x, y, t) = \exp(-t)x(1-x)[2\sin(\pi xy) - y(1-y) - \pi \cos(\pi xy)x(1-2y)] + \exp(-t)y(1-y)[2\sin(\pi xy) - \pi \cos(\pi xy)y(1-2x)]. \]

Example 2 has the closed form analytical solution as follows
\[ u(x, y, t) = \exp(-t)x(1-x)y(1-y). \]

Hence, for Example 2 we can measure the error of its numerical solution. For this purpose, we define the error function as follows
\[ E_{N,J} = ||u_{\text{iter}} - u^*||_\infty, \]

where \( u_{\text{iter}} \) denotes the iterative solution of the linear system (2.6), \( u^* \) denotes the values of exact solution of the heat equation on the mesh. Notice that \( B'_s \) in (4.6) arising from Example 2 is no longer diagonalizable by sine transform. Hence, for Example 2 instead of solving (4.6) exactly, we approximately solve it by one iteration of V-cycle geometric multigrid method, in which ILU smoother is employed with one time of pre-smoothing and one time of post-smoothing; the piecewise linear interpolation and its transpose are used as the interpolation and restriction operators (see [17]). The results of GMRES-BEC and GMRES-BC for solving Example 2 are listed in Table 5.2.

From Table 5.2 shows that GMRES-BEC converges much faster than GMRES-BC does and the accuracy of GMRES-BEC is slightly better than that of GMRES-BC. That means introducing the parameter \( \epsilon \) indeed help improve the performance of BC preconditioner on Example 2. Also, the iteration number of GMRE-BEC changes only slightly as \( N \) and \( J \) changes, which illustrates the matrix-size independent convergence rate of GMRES-BEC.

| \( N \) | \( J + 1 \) | DoF | GMRES-BEC | GMRES-BC |
|-------|---------|-----|-----------|-----------|
| \( N \) | Iter | CPU | \( E_{N,J} \) | Iter | CPU | \( E_{N,J} \) |
| \( 2^6 \) | 2 | 0.60 | 2.95e-4 | 71 | 8.86 | 2.96e-4 |
| \( 2^7 \) | 2 | 1.64 | 3.05e-4 | 79 | 36.33 | 3.05e-4 |
| \( 2^8 \) | 2 | 6.53 | 3.07e-4 | 91 | 171.06 | 3.07e-4 |
| \( 2^9 \) | 2 | 37.82 | 3.08e-4 | 139 | 1118.43 | 3.08e-4 |
| \( 2^6 \) | 2 | 0.79 | 1.41e-4 | 71 | 16.00 | 1.42e-4 |
| \( 2^7 \) | 2 | 2.75 | 1.51e-4 | 79 | 67.62 | 1.51e-4 |
| \( 2^8 \) | 2 | 12.11 | 1.53e-4 | 91 | 333.21 | 1.53e-4 |
| \( 2^9 \) | 2 | 61.86 | 1.54e-4 | 139 | 2181.33 | 1.54e-4 |
| \( 2^6 \) | 2 | 1.38 | 6.43e-5 | 71 | 30.75 | 6.50e-5 |
| \( 2^7 \) | 2 | 5.43 | 7.39e-5 | 79 | 136.69 | 7.39e-5 |
| \( 2^8 \) | 2 | 23.16 | 7.63e-5 | 91 | 671.52 | 1.04e-4 |
| \( 2^9 \) | 2 | 107.16 | 7.69e-5 | 138 | 4376.09 | 7.72e-5 |
| \( 2^6 \) | 2 | 2.65 | 2.57e-5 | 71 | 60.90 | 2.65e-5 |
| \( 2^7 \) | 2 | 10.55 | 3.84e-5 | 79 | 271.01 | 3.75e-5 |
| \( 2^8 \) | 2 | 44.62 | 3.78e-5 | 91 | 1324.53 | 1.04e-4 |
| \( 2^9 \) | 2 | 204.27 | 3.84e-5 | 138 | 8755.48 | 7.60e-5 |
To visualize the numerical solution of Example 2, we present its surface plot and contour plot in Figure 5.1.

![Surface plot and contour plot](image)

Fig. 5.1: Numerical solution of Example 2 at final time $T$ by GMRES-BEC with $N = 20$ and $J = 31$

**Example 3.** (see [14]) The third example is an evolutionary convection diffusion equation with circulating wind and hot wall boundary, which is defined as follows

$$
\partial_t u(x, y, t) = \frac{1}{200} \Delta u - \vec{w} \cdot \nabla u, \quad (x, y) \in \Omega := (-1, 1) \times (-1, 1), \quad t \in (0, T],
$$

$$
u(x, y, t) = (1 - \exp(-10t))\phi(x, y), \quad (x, y) \in \partial \Omega,
$$

$$
u(x, y, 0) = 0, \quad (x, y) \in \bar{\Omega},$$

where $\vec{w} := (2y(1 - x^2), -2x(1 - y^2))$ is the circulating wind, $\phi$ represents the hot wall boundary condition defined as follows

$$\phi(x, y) := \begin{cases} 
1, & x = 1 \text{ and } (x, y) \in \partial \Omega, \\
0, & x \neq 1 \text{ and } (x, y) \in \partial \Omega.
\end{cases}$$

The steady-state version of Example 3 is given by [4, Example 6.1.4]. The Streamline-upwind Petrov-Galerkin (SUPG) stabilization [3] is used to stabilize the discrete spatial terms. Again, we solve (4.6) arising from Example 3 by one iteration of V-cycle geometric multigrid method, in which ILU smoother is employed with one time of pre-smoothing and one time of post-smoothing; the piecewise linear interpolation and its transpose are used as the interpolation and restriction operators (see [17]). The results of GMRES-BEC and GMRES-BC for solving Example 3 are listed in Table 5.3.

Table 5.3 shows that GMRES-BEC converges much faster than GMRES-BC on Example 3 with a convergence rate independent of temporal and spatial stepsizes, which means the introducing of parameter $\epsilon$ improves the performance of BC preconditioner.
Table 5.3: Iteration number of GMRES-BC and GMRES-BEC on Example 3 with $T = 1$

| $N$ | $J + 1$ | DoF | GMRES-BEC | GMRES-BC |
|-----|---------|-----|-----------|----------|
|     |         |     | Iter | CPU | Iter | CPU |
| $2^6$ | 254016  | 5   | 0.74 | 20 | 2.51 |
|       | 1032256 | 5   | 2.82 | 21 | 9.55 |
|       | 4161600 | 5   | 12.26 | 21 | 41.23 |
|       | 16711744 | 5   | 63.70 | 21 | 186.39 |
| $2^7$ | 254016  | 5   | 1.37 | 21 | 5.05 |
|       | 1032256 | 5   | 5.24 | 21 | 19.43 |
|       | 4161600 | 5   | 22.81 | 22 | 83.93 |
|       | 16711744 | 5   | 110.05 | 22 | 374.18 |
| $2^8$ | 254016  | 5   | 2.58 | 21 | 9.73 |
|       | 1032256 | 5   | 10.05 | 22 | 40.43 |
|       | 4161600 | 5   | 43.15 | 22 | 164.93 |
|       | 16711744 | 5   | 199.60 | 22 | 748.97 |
| $2^9$ | 254016  | 4   | 4.94 | 21 | 19.29 |
|       | 1032256 | 5   | 19.86 | 22 | 81.56 |
|       | 4161600 | 5   | 84.35 | 22 | 352.02 |
|       | 16711744 | 5   | 388.14 | 22 | 1493.90 |

Since the boundary condition of Example 3 converges to the steady state, one can expect that solution of Example 3 will be very close to its steady-state solution for sufficiently large $T$. To observe this, we present the numerical solution of Example 3 at $T = 200$ by GMRES-BEC in Figure 5.2. Indeed, the numerical solution exhibited in Figure 5.2 is very close to the numerical steady-state solution exhibited in [4, FIG. 6.5].

Fig. 5.2: Numerical solution of Example 3 at time $T = 200$ by GMRES-BEC with $N = 200$ and $J = 127$
6. Concluding Remark. In this paper, we have proposed the BEC preconditioner as a generalization of BC preconditioner for all-at-once system arising from evolutionary PDEs by introducing a positive parameter $\epsilon$ into the top-right corner of BC preconditioner. We have shown that such generalization preserves the diagonalizability, identity-plus-low-rank decomposition of the preconditioned matrix. Moreover, when $\epsilon$ is sufficiently small, we have shown that (i) the preconditioned matrix by BEC preconditioner has all eigenvalues clustered at 1; (ii) GMRES for the preconditioned system by BEC preconditioner has a linear convergence rate independent of matrix-size. A fast implementation has been introduced so that the computational complexity required for implementation of BEC preconditioner stays the same as that for BC preconditioner. Numerical results have been reported to show that BEC preconditioner outperforms BC preconditioner.

References.

[1] Bernhard Beckermann, Sergei A Goreinov, and Eugene E Tyrtyshnikov. Some remarks on the elman estimate for gmres. SIAM journal on Matrix Analysis and Applications, 27(3):772–778, 2005.

[2] D. Bini, Latouche G., and Meini B. Numerical Methods for Structured Markov Chains. Oxford University Press: New York, 2005.

[3] Alexander N Brooks and Thomas JR Hughes. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. Computer methods in applied mechanics and engineering, 32(1-3):199–259, 1982.

[4] H. C. Elman, D. J. Silvester, and A. J. Wathen. Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics. Numerical Mathematics and Scie, 2014.

[5] Robert D Falgout, Stephanie Friedhoff, Tz V Kolev, Scott P MacLachlan, and Jacob B Schroder. Parallel time integration with multigrid. SIAM Journal on Scientific Computing, 36(6):C635–C661, 2014.

[6] Martin J Gander. 50 years of time parallel time integration. In Multiple Shooting and Time Domain Decomposition Methods, pages 69–113. Springer, 2015.

[7] Martin J Gander and Martin Neumüller. Analysis of a new space-time parallel multigrid algorithm for parabolic problems. SIAM Journal on Scientific Computing, 38(4):A2173–A2208, 2016.

[8] Martin J Gander and Stefan Vandewalle. Analysis of the parareal time-parallel time-integration method. SIAM Journal on Scientific Computing, 29(2):556–578, 2007.

[9] Martin J Gander, Laurence Halpern, Juliet Ryan, and Thuy Thi Bich Tran. A direct solver for time parallelization. In Domain Decomposition Methods in Science and Engineering XXII, pages 491–499. Springer, 2016.

[10] Wolfgang Hackbusch. Parabolic multi-grid methods. In Proc. of the sixth int’l. symposium on Computing methods in applied sciences and engineering, VI, pages 189–197. North-Holland Publishing Co., 1985.

[11] Graham Horton and Stefan Vandewalle. A space-time multigrid method for parabolic partial differential equations. SIAM Journal on Scientific Computing, 16(4):848–864, 1995.

[12] Lennard Kamenski, Weizhang Huang, and Hongguo Xu. Conditioning of finite element equations with arbitrary anisotropic meshes. Mathematics of computation, 83(289):2187–2211, 2014.

[13] Jacques-Louis Lions, Yvon Maday, and Gabriel Turinici. A parareal in time
discretization of PDEs. *C.R.Acad. Sci. Paris, Serie I*, 332(7):661 – 668, 2001. doi: https://doi.org/10.1016/S0764-4442(00)01793-6.

[14] E. McDonald, J. Pestana, and A. Wathen. Preconditioning and iterative solution of all-at-once systems for evolutionary partial differential equations. *SIAM Journal on Scientific Computing*, 40(2):A1012–A1033, 2018.

[15] Eleanor McDonald, Sean Hon, Jennifer Pestana, and Andy Wathen. Preconditioning for nonsymmetry and time-dependence. In *Domain Decomposition Methods in Science and Engineering XXIII*, pages 81–91. Springer, 2017.

[16] M. K. Ng. *Iterative methods for Toeplitz systems*. Numerical Mathematics and Scie, 2004.

[17] David Silvester, Howard Elman, and Alison Ramage. Incompressible Flow and Iterative Solver Software (IFISS) version 3.5, September 2016. http://www.manchester.ac.uk/ifiss/.

[18] A Wathen and A Goddard. A note on parallel preconditioning for all-at-once evolutionary pdes. *Electronic Transactions on Numerical Analysis*, 2019.