ESAIM: COCV 26 (2020) 88
https://doi.org/10.1051/cocv/2020012

ESAIM: Control, Optimisation and Calculus of Variations
www.esaim-cocv.org

DIAGONALIZATION-BASED PARALLEL-IN-TIME ALGORITHMS FOR PARABOLIC PDE-CONSTRAINED OPTIMIZATION PROBLEMS

Shu-Lin Wu[1] and Tao Zhou[2,*]

Abstract. Solving parabolic PDE-constrained optimization problems requires to take into account the discrete time points all-at-once, which means that the computation procedure is often time-consuming. It is thus desirable to design robust and analyzable parallel-in-time (PinT) algorithms to handle this kind of coupled PDE systems with opposite evolution directions. To this end, for two representative model problems which are, respectively, the time-periodic PDEs and the initial-value PDEs, we propose a diagonalization-based approach that can reduce dramatically the computational time. The main idea lies in carefully handling the associated time discretization matrices that are denoted by $\mathbf{B}_{\text{per}}$ and $\mathbf{B}_{\text{ini}}$ for the two problems. For the first problem, we diagonalize $\mathbf{B}_{\text{per}}$ directly and this results in a direct PinT algorithm (*i.e.*, non-iterative). For the second problem, the main idea is to design a suitable approximation $\widehat{\mathbf{B}}_{\text{per}}$ of $\mathbf{B}_{\text{ini}}$, which naturally results in a preconditioner of the discrete KKT system. This preconditioner can be used in a PinT pattern, and for both the Backward-Euler method and the trapezoidal rule the clustering of the eigenvalues and singular values of the preconditioned matrix is justified. Compared to existing preconditioners that are designed by approximating the Schur complement of the discrete KKT system, we show that the new preconditioner leads to much faster convergence for certain Krylov subspace solvers, *e.g.*, the GMRES and BiCGStab methods. Numerical results are presented to illustrate the advantages of the proposed PinT algorithm.

Mathematics Subject Classification. 65M55, 65M12, 65M15, 65Y05.

Received March 14, 2019. Accepted March 10, 2020.

1. Introduction

We are interested in minimizing the tracking-type functionals with distributed control:

$$\min \mathcal{J}(y,u) := \frac{1}{2}\int_0^T \int_\Omega (y(\mathbf{x},t)-\bar{y}(\mathbf{x},t))^2 \mathrm{d}\mathbf{x}\mathrm{d}t + \frac{\alpha}{2}\int_0^T \int_\Omega u^2(\mathbf{x},t)\mathrm{d}\mathbf{x}\mathrm{d}t, \tag{1.1a}$$

Keywords and phrases: Parabolic PDE-constrained optimization, PinT algorithm, diagonalization technique, preconditioner, GMRES, BiCGStab.

[1] School of Mathematics and Statistics, Northeast Normal University, Changchun 130024, P.R. China.
[2] NCMIS & LSEC, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, P.R. China.

[*] Corresponding author: tzhou@lsec.cc.ac.cn

subject to a time-dependent PDE with second-order spatial operator $\mathbf{L}$:

$$\partial_t y - \mathbf{L}y = u, (\mathbf{x}, t) \in \Omega \times (0, T),\ \mathbf{L}y := \sum_{j,l=1}^{d} \frac{\partial}{\partial x_l}\left(a_{jl}(\mathbf{x})\frac{\partial}{\partial x_j} y(\mathbf{x}, t)\right), \tag{1.1b}$$

with boundary condition $y(\mathbf{x}, t)_{(\mathbf{x},t)\in\partial\Omega\times(0,T)} = y_{\rm b}(\mathbf{x}, t)$. We assume that $\partial\Omega$ is smooth and that the model parameters a_{jl} are continuously differentiable on $\Omega \cup \partial\Omega$ and satisfy $a_{jl} = a_{lj}$ and $\sum_{j,l=1}^{d} a_{jl}(\mathbf{x})\xi_j\xi_l \geq \nu|\xi|^2 (\forall\xi \in \mathbb{R}^d)$ for some $\nu > 0$. In (1.1a), y is the state variable, $\bar{y}$ is the desired state that we want to approach and u is the control variable. For PDE (1.1b), we consider two different conditions:

$$\begin{cases} y(\mathbf{x}, 0) = y(\mathbf{x}, T), & \text{time-periodic condition,} \\ y(\mathbf{x}, 0) = y_{\rm ini}(\mathbf{x}), & \text{initial-value condition,} \end{cases} \tag{1.1c}$$

where $y_{\rm ini}(\mathbf{x})$ is known. Applications and numerical methods of optimization problems with initial-value condition have been extensively studied in the past years; see, *e.g.*, [4–6, 30]. The study of time-periodic optimization problems is relatively rare, but this kind of problems indeed have some important applications as well; see [8, 15, 28, 36] for the optimization of the airborne wind energy systems which is an important technique to harvest renewable energy from wind, [16] for the design of cyclically steered (bio-) reactors and [33] for the optimization of simulated moving bed processes, etc. Numerical methods for time-periodic optimization problems were studied by many authors; see, *e.g.*, [1, 13] for the multi-grid approach, [2, 17–21] for the spectral truncation approach (also called multiharmonic approach in the frequency domain) and [14, 31, 32] for the design of efficient preconditioner.

Following an *optimize-then-discretize* approach and then by applying the first-order optimality condition, we get the following KKT system (see [1, 4, 20] for more details):

$$\begin{bmatrix} \mathcal{I} & 0 & \mathcal{L}_2 \\ 0 & \alpha\mathcal{I} & -\mathcal{I} \\ \mathcal{L}_1 & -\mathcal{I} & 0 \end{bmatrix} \begin{bmatrix} y \\ u \\ p \end{bmatrix} = \begin{bmatrix} \bar{y} \\ 0 \\ 0 \end{bmatrix}, \tag{1.2a}$$

where p is the Lagrange multiplier and $\mathcal{L}_{1,2}$ are defined by

$$\begin{aligned} &\mathcal{L}_1 y := (\partial_t - \mathbf{L})y, \quad \text{with } \begin{cases} y(\mathbf{x}, 0) = y(\mathbf{x}, T), & \text{time-periodic condition,} \\ y(\mathbf{x}, 0) = y_{\rm ini}(\mathbf{x}), & \text{initial-value condition,} \end{cases} \\ &\mathcal{L}_2 p := (-\partial_t - \mathbf{L})p, \text{ with } \begin{cases} p(\mathbf{x}, T) = p(\mathbf{x}, 0), & \text{time-periodic condition,} \\ p(\mathbf{x}, T) = 0, & \text{initial-value condition,} \end{cases} \end{aligned} \tag{1.2b}$$

where y evolves forward from $t = 0$ to $t = T$ and p evolutes backward from $t = T$ to $t = 0$. By eliminating the control variable u, the saddle point system (1.2a) is reduced to

$$\begin{bmatrix} \mathcal{L}_1 & -\alpha^{-1}\mathcal{I} \\ \mathcal{I} & \mathcal{L}_2 \end{bmatrix} \begin{bmatrix} y \\ p \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{y} \end{bmatrix}. \tag{1.3}$$

Let $\{t_n\}_{n=0}^{N_t}$ be the equally spaced time points with step-size Δt and $t_0 = 0$ and $t_{N_t} = T$. Then, after space and time discretizations, *e.g.*, by the linear θ-method described in Section 2.1, we get the following discrete version of (1.3):

$$\left(\overbrace{\underbrace{\begin{bmatrix} A_1 & -\Delta t\alpha^{-1}\widetilde{A}_1 \\ \Delta t\widetilde{A}_2 & A_2 \end{bmatrix}}_{:=\mathbf{B}} \otimes I_x - \Delta t \underbrace{\begin{bmatrix} I_t & \\ & I_t \end{bmatrix}}_{:=\mathbf{I}_t} \otimes L_h}^{:=\mathbf{K}}\right) \begin{bmatrix} Y \\ P \end{bmatrix} = \begin{bmatrix} F_y \\ F_p \end{bmatrix}, \tag{1.4a}$$

where $L_h \in \mathbb{R}^{N_x \times N_x}$ denotes the space discretization of the operator $\mathbf{L}$ in (1.1b)[1], F_y, $F_p \in \mathbb{R}^{N_t N_x}$ are input vectors, $I_t \in \mathbb{R}^{N_t \times N_t}$ and $I_x \in \mathbb{R}^{N_x \times N_x}$ are identity matrices, $Y, P \in \mathbb{R}^{N_t N_x}$ are vectors collecting the approximations of y and p at the discrete time points. The matrices $A_{1,2}, \widetilde{A}_{1,2} \in \mathbb{R}^{N_t \times N_t}$ represents the time discretizations at the N_t time points. We distinguish the matrices $\mathbf{B}$ and $\mathbf{K}$ in two cases:

$$\mathbf{B} = \begin{cases} \mathbf{B}_{\text{per}}, \\ \mathbf{B}_{\text{ini}}, \end{cases} \quad \mathbf{K} = \begin{cases} \mathbf{K}_{\text{per}} := \mathbf{B}_{\text{per}} \otimes I_x - \Delta t \mathbf{I}_t \otimes L_h, & \text{(time-periodic condition)}, \\ \mathbf{K}_{\text{ini}} := \mathbf{B}_{\text{ini}} \otimes I_x - \Delta t \mathbf{I}_t \otimes L_h, & \text{(initial-value condition)}, \end{cases} \tag{1.4b}$$

where the matrices $\mathbf{B}_{\text{per}}$ and $\mathbf{B}_{\text{ini}}$ will be given in Sections 2 and 3, respectively. Now, if the matrix $\mathbf{B}$ in (1.4a) is diagonalizable, such as

$$\mathbf{B} = \mathbf{V}^{-1}\mathbf{D}\mathbf{V}, \ \mathbf{D} = \text{diag}(d_1, d_2, \ldots, d_{2N_t}), \tag{1.5}$$

we can factorize the coefficient matrix $\mathbf{K}$ in (1.4a) as follows

$$\mathbf{K} = \mathbf{B} \otimes I_x - \Delta t \mathbf{I}_t \otimes L_h = \left(\mathbf{V}^{-1} \otimes I_x\right)\left(\mathbf{D} \otimes I_x - \Delta t \mathbf{I}_t \otimes L_h\right)\left(\mathbf{V} \otimes I_x\right).$$

This implies that we can solve (1.4a) in three steps:

$$\begin{aligned} &\text{(a)} \qquad && \left(\mathbf{V}^{-1} \otimes I_x\right)\mathbf{X}_1 = \begin{bmatrix} F_y \\ F_p \end{bmatrix}; \\ &\text{(b)} && (d_n I_x - \Delta t L_h)\mathbf{X}_{2,n} = \mathbf{X}_{1,n}, n = 1, 2, \ldots, 2N_t; \\ &\text{(c)} && (\mathbf{V} \otimes I_x)\begin{bmatrix} Y \\ P \end{bmatrix} = \mathbf{X}_2. \end{aligned} \tag{1.6}$$

As will be explained in Section 2, the matrix $\mathbf{V}$ satisfies

$$\mathbf{V} = \begin{bmatrix} W_1 & W_2 \\ W_3 & W_4 \end{bmatrix}\begin{bmatrix} V & \\ & V \end{bmatrix}, \tag{1.7}$$

where V is the discrete Fourier matrix and $W_{1,2,3,4}$ are diagonal matrices. Hence, the parallel fast Fourier transform (PFFT) can be employed to make the implementation of Step-(a) and Step-(c) efficient[2]; see Remark 2.7 for more explanations. For Step-(b), it is clear that these $2N_t$ elliptic PDEs can be solved simultaneously, *i.e.*, in a direct parallel pattern. The above strategy is the so-called *diagonalization* technique, which is proposed by Maday and Rønquist [24] and is further justified by Gander *et al.* [9]. The diagonalization idea is also used in [34] to design a PinT coarse grid correction for the parareal algorithm [10, 22] and in [3, 27] to design efficient preconditioners for nonsymmetric block Toeplitz matrices.

[1]In practice, the matrix L_h can be obtained by several discretizations, *e.g.*, the finite element method or the finite difference method.

[2]See http://www.fftw.org/parallel/parallel-fftw.html for PFFT, which is free for download.

In the case of time-periodic condition, we show that the matrix $\mathbf{B}_{\rm per}$ can be diagonalized with explicit formulas for $\mathbf{V}$, $\mathbf{V}^{-1}$ and $\mathbf{D}$. Thus, problem (1.1a)–(1.1c) can be solved by efficient PinT computation. However, if we follow a direct diagonalization of $\mathbf{B}_{\rm per}$, the condition number ${\rm Cond}_2(\mathbf{V})$ is large and is sensitive to the change of the problem/discretization parameters. As a result, the roundoff error arising from Step-(a) and Step-(c) in (1.6) seriously pollutes the accuracy of the numerical solution (see [9] for more discussions about the influence of a large condition number of $\mathbf{V}$). To overcome this, we propose a *scaling* strategy with a free parameter β and this transforms $\mathbf{B}_{\rm per}$ to $\mathbf{B}_{\rm per}(\beta)$. By choosing $\beta=\frac{1}{\sqrt{\alpha}}$ we show that the condition number of $\mathbf{B}_{\rm per}(\beta)$ is a moderate quantity and is robust with respect to the problem/discretization parameters.

For the initial-value condition, explicit formulas for $\mathbf{V}$, $\mathbf{V}^{-1}$ and $\mathbf{D}$ of the corresponding matrix $\mathbf{B}_{\rm ini}$ are no longer available. If we use an existing algorithm to directly diagonalize $\mathbf{B}_{\rm ini}$, *e.g.*, the `eig` command in Matlab, the eigenvector matrix $\mathbf{V}$ is not of the form (1.7) and therefore FFT is not applicable. Motivated by a recent work [27], we solve the discrete KKT system (1.4a) with $\mathbf{B}=\mathbf{B}_{\rm ini}$ by Krylov subspace solvers, by using $\widehat{\mathbf{K}}_{\rm per}(\beta):=\widehat{\mathbf{B}}_{\rm per}(\beta)\otimes I_x-\Delta\mathbf{I}_t\otimes L_h$ as the preconditioner, where $\widehat{\mathbf{B}}_{\rm per}(\beta)$ is a matrix closely related to the matrix $\mathbf{B}_{\rm per}(\beta)$. The matrix $\widehat{\mathbf{B}}_{\rm per}(\beta)$ can be diagonalized with closed formulas and the eigenvector matrix is still of the form (1.7). Hence, for any input vector $\mathbf{r}$ we can compute $\widehat{\mathbf{K}}_{\rm per}^{-1}(\beta)\mathbf{r}$, which is the expensive part of the Krylov subspace solvers, in an efficient PinT pattern as well. Clustering of the eigenvalues and the singular values of the preconditioned matrix is justified. Numerical results indicate interesting convergence properties of the Krylov solvers for different time-integrators. In particular, for the GMRES method the new preconditioner results in faster (and superlinear) convergence than the classical preconditioner [6, 18, 19, 30, 32], which is designed by approximating the Schur complement of the discrete KKT system. (This advantage holds for both the Backward-Euler method and the trapezoidal rule.) For the BiCGStab method, if the Backward-Euler method is used the new preconditioner results in faster convergence rate than the classical preconditioner, while for the trapezoidal rule the latter is better than the former. For the two time-integrators, the convergence rates of the GMRES and BiCGStab methods using the new preconditioner are robust with respect to the regularization/discretization parameters.

We mention that this is not the first effort toward PinT computation for time-dependent PDE-constrained optimization problems and several relevant work can be found in the literature; see, *e.g.*, [4, 7, 11, 12, 23, 25, 26]. Precisely, in [4, 7, 12, 25, 26] the authors studied the parareal algorithm [10, 22] for this kind of problems. The parareal algorithm is a two-grid in time method using two different time step-sizes. In [11, 23] the authors studied the overlapping and non-overlapping time domain decomposition methods, which is based on decomposing the whole time interval into several subintervals and then solving the time-dependent PDE-constrained optimization problem on these subintervals in parallel. The algorithms studied in these previous work are completely different from the algorithm proposed in this paper, because for our algorithm the key intelligence lies in diagonalizing a class of KKT discrete system, which never appears in these previous work. For our algorithm we use a uniform step-size for time discretization and we do not need a decomposition of the time interval. The rest of this paper is organized as follows. In Section 2 we consider the PDE-constrained optimization problem (1.1a)–(1.1b) with time-periodic condition. Then, in Section 3 we consider the initial-value case. Our numerical results are given in Section 4 and we conclude this paper in Section 5.

2. Time-periodic PDE-optimization problems

The goal of this section is to present the details of the diagonalization of the matrix $\mathbf{B}$ in (1.4a). These are necessary preparations for Section 3, where we will consider the initial-value case.

2.1. Time discretization

We start by time discretization using the linear θ-method, where $\theta=1$ corresponds to the Backward-Euler method and $\theta=\frac{1}{2}$ corresponds to the trapezoidal rule. By applying the linear θ-method to the two equations in (1.3) and by using the time-periodic condition, we have

$$\begin{cases} \frac{y_1-y_{N_t}}{\Delta t} - L_h[\theta y_1 + (1-\theta)y_{N_t}] - \alpha^{-1}[\theta p_1 + (1-\theta)p_{N_t}] = 0, \\ \frac{y_2-y_1}{\Delta t} - L_h[\theta y_2 + (1-\theta)y_1] - \alpha^{-1}[\theta p_2 + (1-\theta)p_1] = 0, \\ \vdots \\ \frac{y_{N_t}-y_{N_t-1}}{\Delta t} - L_h[\theta y_{N_t} + (1-\theta)y_{N_t-1}] - \alpha^{-1}[\theta p_{N_t} + (1-\theta)p_{N_t-1}] = 0, \end{cases} \tag{2.1a}$$

$$\begin{cases} -\frac{p_1-p_{N_t}}{\Delta t} - L_h[\theta p_{N_t} + (1-\theta)p_1] + \theta y_{N_t} + (1-\theta)y_1 = \bar{y}_{\theta,0} \text{ (move to bottom)}, \\ -\frac{p_2-p_1}{\Delta t} - L_h[\theta p_1 + (1-\theta)p_2] + \theta y_1 + (1-\theta)y_2 = \bar{y}_{\theta,1}, \\ \vdots \\ -\frac{p_{N_t}-p_{N_t-1}}{\Delta t} - L_h[\theta p_{N_t-1} + (1-\theta)p_{N_t}] + \theta y_{N_t-1} + (1-\theta)y_{N_t} = \bar{y}_{\theta,N_t-1}, \end{cases} \tag{2.1b}$$

where $\theta = 1$ or $\theta = \frac{1}{2}$ and $\bar{y}_{\theta,n} = \theta\bar{y}_n + (1-\theta)\bar{y}_{n+1}$. Let

$$\begin{gathered} Y = \begin{bmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_{N_t}(x) \end{bmatrix}, \ P = \begin{bmatrix} p_1(x) \\ p_2(x) \\ \vdots \\ p_{N_t}(x) \end{bmatrix}, F_2 = \Delta t \begin{bmatrix} \bar{y}_{\theta,1}(x) \\ \cdots \\ \bar{y}_{\theta,N_t-1}(x) \\ \bar{y}_{\theta,0}(x) \end{bmatrix}, \\ C_{\text{per},1} = \begin{bmatrix} 1 & & & -1 \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \ C_{\text{per},2} = \begin{bmatrix} \theta & & & 1-\theta \\ 1-\theta & \theta & & \\ & \ddots & \ddots & \\ & & 1-\theta & \theta \end{bmatrix} \in \mathbb{R}^{N_t \times N_t}. \end{gathered} \tag{2.2}$$

Then, as marked in (2.1b), by moving the first equation to the bottom as the last equation it is clear that the equations in (2.1b) can be represented as

$$\begin{aligned} &(C_{\text{per},1} \otimes I_x)Y - \Delta t(C_{\text{per},2} \otimes L_h)Y - \Delta t\alpha^{-1}(C_{\text{per},2} \otimes I_x)P = 0, \\ &(C_{\text{per},1}^\top \otimes I_x)P - \Delta t(C_{\text{per},2}^\top \otimes L_h)P + \Delta t(C_{\text{per},2}^\top \otimes I_x)Y = F_2. \end{aligned} \tag{2.3}$$

By letting $F_y = 0$ and $F_p = ((C_{\text{per},2}^{-1})^\top \otimes I_x)F_2$ and

$$A_1 = C_{\text{per},2}^{-1}C_{\text{per},1}, \ A_2 = (C_{\text{per},1}C_{\text{per},2}^{-1})^\top, \widetilde{A}_{1,2} = I_t, \tag{2.4a}$$

we can rewrite (2.3) as (1.4a), provided $C_{\text{per},2}$ is invertible, where

$$\mathbf{B} = \mathbf{B}_{\text{per}} := \begin{bmatrix} C_{\text{per},2}^{-1}C_{\text{per},1} & -\Delta t\alpha^{-1}I_t \\ \Delta t I_t & (C_{\text{per},1}C_{\text{per},2}^{-1})^\top \end{bmatrix}. \tag{2.4b}$$

Remark 2.1. For $\theta = \frac{1}{2}$, *i.e.*, the trapezoidal rule, the quantity N_t (the number of time points) should be an odd integer, because otherwise the matrix $C_{\text{per},2}$ is singular.

2.2. Diagonalization of $\mathbf{B}_{\text{per}}$

To make a diagonalization of the matrix $\mathbf{B}_{\text{per}}$ in (2.4b), the following lemma is necessary.

Lemma 2.2. *Let* $\omega = e^{-\frac{2\pi i}{N_t}}$, $\delta \in \mathbb{C}$ *and*

$$V = \frac{1}{\sqrt{N_t}} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega & \cdots & \omega^{N_t-1} \\ \vdots & \vdots & \vdots & \ddots \\ 1 & \omega^{N_t-1} & \cdots & \omega^{(N_t-1)(N_t-1)} \end{bmatrix}, \quad C(\delta) = \begin{bmatrix} 1 & & & & \delta \\ \delta & 1 & & & \\ & \ddots & \ddots & & \\ & & \delta & 1 & \\ & & & \delta & 1 \end{bmatrix}.$$

Then, it holds that

$$C(\delta) = V^{-1} D(\delta) V, \ C^{\top}(\delta) = V^{-1} D^*(\delta) V, \tag{2.5}$$

where $D(\delta) = \mathrm{diag}(\lambda_1, \ldots, \lambda_{N_t})$ *with* $\lambda_n = 1 + \delta\omega^{n-1}$ *and* $D^*(\delta)$ *denotes the conjugate matrix of* $D(\delta)$*.*

Proof. The matrix $C(\delta)$ is a circulant matrix and the spectral decomposition of such a matrix can be found in many places; see, *e.g.*, [27]. □

It is clear that $C_{\mathrm{per},1} = C(-1)$ and $C_{\mathrm{per},2} = \theta C(\frac{1-\theta}{\theta})$. Hence, from Lemma 2.2 we have

$$A_1 = V^{-1}\Lambda(\theta)V, \ A_2 = V^{-1}\Lambda^*(\theta)V, \ \text{with } \Lambda(\theta) := \frac{1}{\theta} D(-1) D^{-1}\left(\frac{1-\theta}{\theta}\right). \tag{2.6}$$

Hence, we can use similarity transformation for the matrix $\mathbf{B}_{\mathrm{per}}$ given by (2.4b):

$$\mathbf{B}_{\mathrm{per}} = \begin{bmatrix} V^{-1} & \\ & V^{-1} \end{bmatrix} \underbrace{\begin{bmatrix} \Lambda(\theta) & -\Delta t \alpha^{-1} I_t \\ \Delta t I_t & \Lambda^*(\theta) \end{bmatrix}}_{:=\widetilde{\mathbf{B}}_{\mathrm{per}}} \begin{bmatrix} V & \\ & V \end{bmatrix}. \tag{2.7}$$

2.2.1. A direct diagonalization

From (2.7), it is clear that the diagonalization of $\mathbf{B}_{\mathrm{per}}$ is equivalent to diagonalizing $\widetilde{\mathbf{B}}_{\mathrm{per}}$. To this end, we need the following auxiliary lemma.

Lemma 2.3. *Let* $G_{1,2,3,4} \in \mathbb{C}^{N_t \times N_t}$ *be four diagonal matrices and* $\mathbf{G} = \begin{bmatrix} G_1 & G_2 \\ G_3 & G_4 \end{bmatrix}$*. Suppose* G_2 *and* G_3 *are invertible. Then, it holds that*

$$\mathbf{G} = \mathbf{W} \begin{bmatrix} G_1 + G_2 S_1 & \\ & G_4 + G_3 S_2 \end{bmatrix} \mathbf{W}^{-1}, \ \mathbf{W} = \begin{bmatrix} I_t & S_2 \\ S_1 & I_t \end{bmatrix},$$

provided $\mathbf{W}$ *is invertible (i.e.,* $S_1 S_2 \neq I_t$*), where* I_t *is the identity matrix and*

$$S_1 = \frac{1}{2} G_2^{-1} \left(G_4 - G_1 + \sqrt{(G_4 - G_1)^2 + 4 G_2 G_3} \right),$$
$$S_2 = -\frac{1}{2} G_3^{-1} \left(G_4 - G_1 + \sqrt{(G_4 - G_1)^2 + 4 G_2 G_3} \right).$$

Proof. It is sufficient to verify $\mathbf{GW} = \mathbf{W}\begin{bmatrix} G_1 + G_2S_1 & \\ & G_4 + G_3S_2 \end{bmatrix}$. To this end, a routine calculation yields

$$\mathbf{GW} = \begin{bmatrix} G_1 + G_2S_1 & G_2 + G_1S_2 \\ G_3 + G_4S_1 & G_4 + G_3S_2 \end{bmatrix},$$
$$\mathbf{W}\begin{bmatrix} G_1 + G_2S_1 & \\ & G_4 + G_3S_2 \end{bmatrix} = \begin{bmatrix} G_1 + G_2S_1 & G_4S_2 + G_3S_2^2 \\ G_1S_1 + G_2S_1^2 & G_4 + G_3S_2 \end{bmatrix}.$$

Hence, it suffices to show $G_2S_1^2 + (G_1 - G_4)S_1 - G_3 = 0$ and $G_3S_2^2 + (G_4 - G_1)S_2 - G_2 = 0$. This is true because S_1 and S_2 are respectively the root of these two matrix equations. ☐

Applying Lemma 2.3 to the matrix $\widetilde{\mathbf{B}}_{\rm per}$ given by (2.7) results in $\widetilde{\mathbf{B}}_{\rm per} = \mathbf{WDW}^{-1}$ with

$$\begin{aligned} \mathbf{D} &= \begin{bmatrix} \mathrm{Re}(\Lambda(\theta)) + i\sqrt{\mathrm{Im}^2(\Lambda(\theta)) + \alpha^{-1}\Delta t^2 I_t} & \\ & \mathrm{Re}(\Lambda(\theta)) - i\sqrt{\mathrm{Im}^2(\Lambda(\theta)) + \alpha^{-1}\Delta t^2 I_t} \end{bmatrix}, \\ \mathbf{W} &= \begin{bmatrix} I_t & i\frac{\mathrm{Im}(\Lambda(\theta)) - \sqrt{\mathrm{Im}^2(\Lambda(\theta)) + \alpha^{-1}\Delta t^2 I_t}}{\Delta t} \\ i\frac{\mathrm{Im}(\Lambda(\theta)) - \sqrt{\mathrm{Im}^2(\Lambda(\theta)) + \alpha^{-1}\Delta t^2 I_t}}{\Delta t\alpha^{-1}} & I_t \end{bmatrix}. \end{aligned} \tag{2.8}$$

Substituting this into (2.7), we get the following diagonalization of the matrix $\mathbf{B}_{\rm per}$.

Theorem 2.4. *The matrix $\mathbf{B}_{\rm per}$ given by (2.4b) with A_1 and A_2 being given by (2.4a) can be diagonalized as $\mathbf{B}_{\rm per} = \mathbf{V}^{-1}\mathbf{D}\mathbf{V}$, where*

$$\mathbf{V} = \mathbf{W}^{-1}\begin{bmatrix} V & \\ & V \end{bmatrix},$$

and $\mathbf{D}$ and $\mathbf{W}$ are given by (2.8).

As we mentioned in Section 1, the roundoff error arising from the first and third steps of the diagonalization technique has a serious effect on the accuracy of the obtained numerical solution and in particular such a roundoff error is proportional to the condition number of $\mathbf{V}$; see [9] for more details. In Figure 1, for four values of N_t we plot the condition number of $\mathbf{V}$ when the regularization parameter α varies. We see that for both the Backward-Euler method (*i.e.*, $\theta = 1$) and the trapezoidal rule (*i.e.*, $\theta = \frac{1}{2}$), the condition number $\mathrm{Cond}_2(\mathbf{V})$ is very large when $\alpha \to \infty$ or $\alpha \to 0$.

It would be interesting to show numerical results to illustrate how the condition number of the eigenvector matrix $\mathbf{V}$ effects the accuracy of the numerical solution. To this end, we consider the following simple ODE-constrained optimization problem:

$$\begin{cases} \min \mathcal{J}(y,u) := \frac{1}{2}\int_0^1 (y(t) - \bar{y}(t))^2 \mathrm{d}t + \frac{\alpha}{2}\int_0^1 u^2(t)\mathrm{d}t, \\ y'(t) - \lambda y = u, \ y(0) = y(1), \\ \text{with } \lambda = -5, \ \bar{y}(t) = \min\{\frac{1}{2}, \max\{-\frac{1}{2}, \sin(2\pi t)\}\}. \end{cases} \tag{2.9}$$

The quantity λ can be regarded as an eigenvalue of the operator L_h in (1.1a). For $\alpha = 10^{-5}$ and three values of N_t: $N_t = 33, 65, 129$, we show in Figure 2 the solution computed via diagonalization of the matrix $\mathbf{B}_{\rm per}$ (dash-dot line), where the eigenvector matrix $\mathbf{V}$ is given by Theorem 2.4; for comparison, we also show the numerical solution by directly solving (1.4a) using the `inv` command in Matlab as the reference solution (solid line). For $N_t = 33$, we see from the top row of Figure 2 that, for both the state variable and the control variable, the two numerical solutions are very close, while when we increase N_t to 65 we see obvious difference between

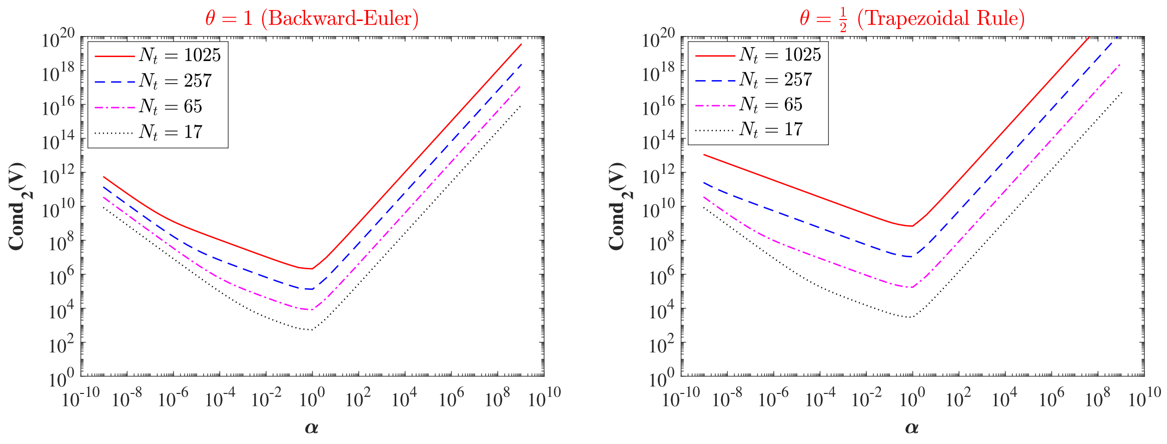


FIGURE 1. For four values of N_t, the condition number of the eigenvector matrix $\mathbf{V}$ given by Theorem 2.4 as a function of α. *Left*: $\theta = \frac{1}{2}$; *Right*: $\theta = 1$. Here, for all the plotings $\Delta t = \frac{1}{N_t - 1}$.

the two numerical solutions. When $N_t = 129$, from the bottom row we see that the numerical solution obtained by diagonalization is completely wrong. These observations confirm the condition number $\mathrm{Cond}_2(\mathbf{V})$ shown in Figure 1 very well, since for fixed α the condition number becomes large as N_t increases.

2.2.2. The modified diagonalization

We now propose a new diagonalization for the matrix $\mathbf{B}_{\text{per}}$ in (2.4b), which results in a much better condition number of the eigenvector matrix. The idea lies in *parametrization* and *scaling*. With $\beta \neq 0$ being a free parameter, we let $\widehat{Y} = \beta^{-1} Y$ and $\widehat{F}_y = \beta^{-1} F_y$; then we can rewrite (1.4a) with $\mathbf{B} = \mathbf{B}_{\text{per}}$ as follows

$$\left(\begin{bmatrix} \beta^{-1} I_t & \\ & I_t \end{bmatrix} \otimes I_x\right) (\mathbf{B}_{\text{per}} \otimes I_x - \Delta t \mathbf{I}_t \otimes L_h) \left(\begin{bmatrix} \beta I_t & \\ & I_t \end{bmatrix} \otimes I_x\right) \begin{bmatrix} \widehat{Y} \\ P \end{bmatrix} = \begin{bmatrix} \widehat{F}_y \\ F_p \end{bmatrix},$$

i.e.,

$$\left(\underbrace{\begin{bmatrix} A_1 & -\Delta t (\alpha\beta)^{-1} I_t \\ \Delta t \beta I_t & A_2 \end{bmatrix}}_{:=\mathbf{B}_{\text{per}}(\beta)} \otimes I_x - \Delta t \begin{bmatrix} I_t & \\ & I_t \end{bmatrix} \otimes L_h \right) \begin{bmatrix} \widehat{Y} \\ P \end{bmatrix} = \begin{bmatrix} \widehat{F}_y \\ F_p \end{bmatrix}, \tag{2.10}$$

where, as before, A_1 and A_2 are the matrices given by (2.4a). By using (2.6) we have

$$\mathbf{B}_{\text{per}}(\beta) = \begin{bmatrix} V^{-1} & \\ & V^{-1} \end{bmatrix} \begin{bmatrix} \Lambda(\theta) & -\Delta t (\alpha\beta)^{-1} I_t \\ \Delta t \beta I_t & \Lambda^*(\theta) \end{bmatrix} \begin{bmatrix} V & \\ & V \end{bmatrix}. \tag{2.11}$$

With $\Lambda(\theta)$ being the diagonal matrix given by (2.6), we define

$$S_1(\beta) = i \frac{\mathrm{Im}(\Lambda(\theta)) - \sqrt{\mathrm{Im}^2(\Lambda(\theta)) + \alpha^{-1} \Delta t^2 I_t}}{\Delta t (\alpha\beta)^{-1}}, \quad S_2(\beta) = \frac{S_1(\beta)}{\alpha}$$

$$\widetilde{\mathbf{D}}(\beta) = \begin{bmatrix} \sqrt{\Lambda(\theta)} + S_1(\beta) & \\ & \sqrt{\Lambda^*(\theta)} + S_2(\beta) \end{bmatrix}.$$

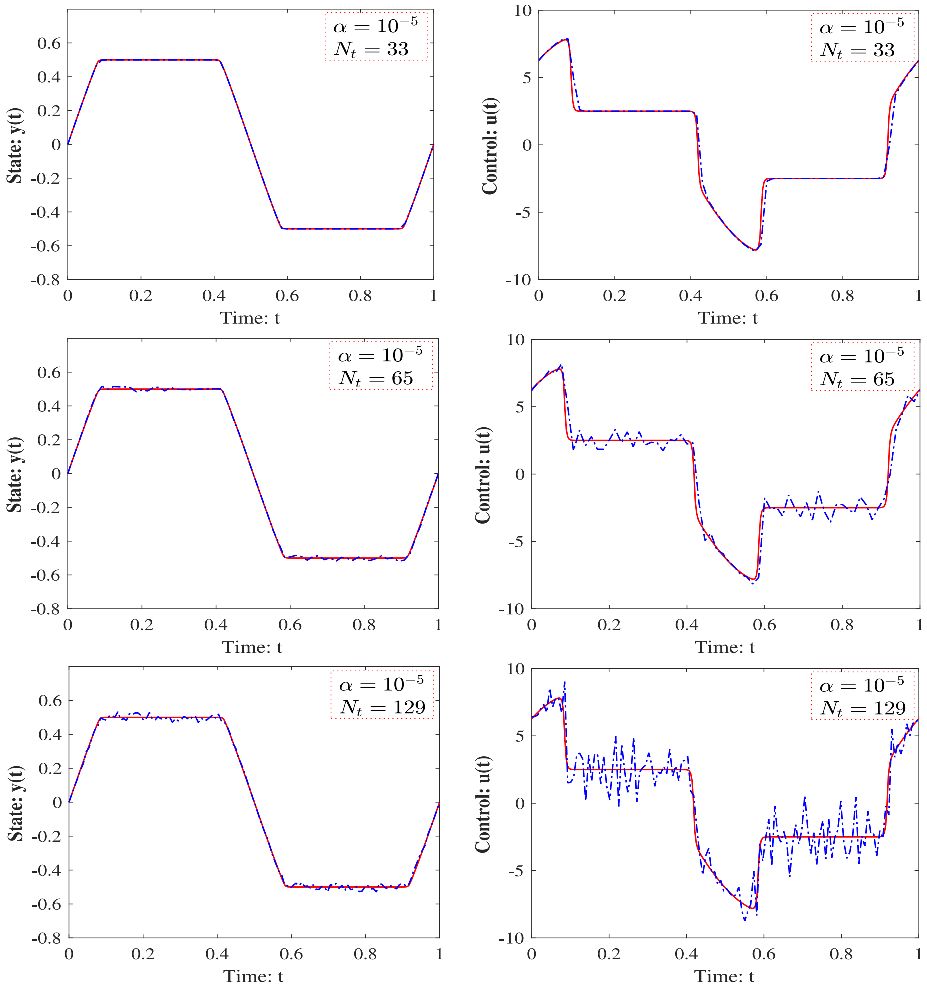


FIGURE 2. Comparison between the numerical solutions computed by two ways: diagonalization technique (1.6) (*dash-dot line*) and directly solving (1.4a) by the `inv` command in Matlab (*solid line*). *From top to bottom*: $N_t = 33$, 65 and 129. *Left column*: the numerical solution for $y(t)$; *Right column*: the numerical solution for $u(t)$. Here, $\alpha = 10^{-5}$ and $\Delta t = \frac{1}{Nt-1}$ and we use the Backward-Euler method, *i.e.*, $\theta = 1$ in (2.1a)–(2.1b), as the time-integrator. For the trapezoidal rule, *i.e.*, $\theta = \frac{1}{2}$, the results are the same.

Theorem 2.5. *With the matrices* $\mathbf{D}$ *given by* (2.8)*, the matrix* $\mathbf{B}_{\rm per}(\beta)$ *with* $\beta \neq 0$ *can be diagonalized as* $\mathbf{B}_{\rm per}(\beta) = \mathbf{V}(\beta)^{-1}\mathbf{D}\mathbf{V}(\beta)$*, where*

$$\mathbf{V}(\beta) = \mathbf{W}^{-1}(\beta)\begin{bmatrix} V & \\ & V \end{bmatrix}, \ \mathbf{W}(\beta) = \begin{bmatrix} I_t & S_2(\beta) \\ S_1(\beta) & I_t \end{bmatrix} \widetilde{\mathbf{D}}^{-1}(\beta). \tag{2.12}$$

Proof. Let $\widetilde{\mathbf{B}}_{\rm per}(\beta) = \begin{bmatrix} \Lambda(\theta) & -\Delta t(\alpha\beta)^{-1} I_t \\ \Delta t \beta I_t & \Lambda^*(\theta) \end{bmatrix}$. Then, by applying Lemma 2.3 and by some routine calculations we have

$$\widetilde{\mathbf{B}}_{\rm per}(\beta) = \mathbf{W}(\beta)\mathbf{D}\mathbf{W}^{-1}(\beta) = \mathbf{W}(\beta)\widetilde{\mathbf{D}}(\beta)\mathbf{D}\widetilde{\mathbf{D}}^{-1}(\beta)\mathbf{W}^{-1}(\beta). \tag{2.13}$$

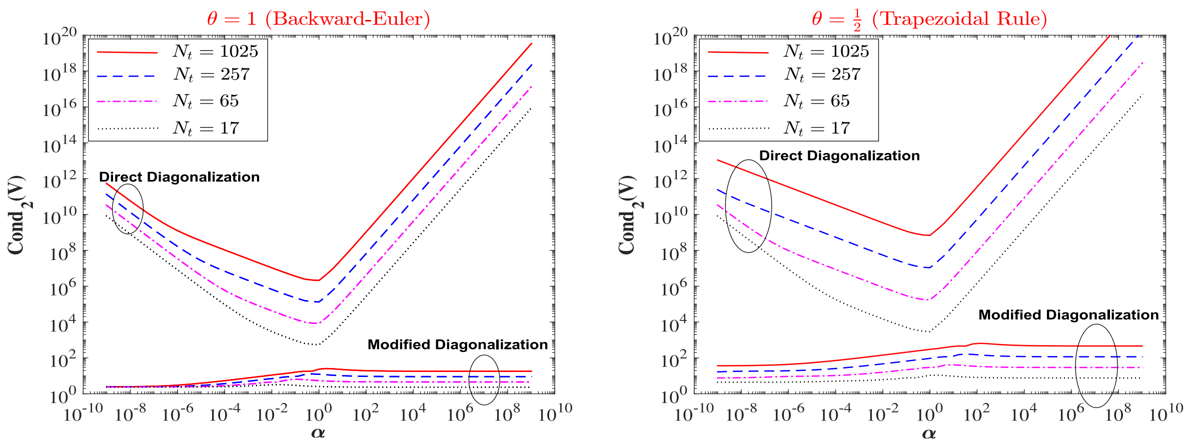


FIGURE 3. Comparisons of $\mathrm{Cond}_2(\mathbf{V})$ when α varies from 2^{-30} to 2^{30}. The direct diagonalization is presented by Theorem 2.4 and the modified diagonalization based on parametrization and scaling is given by Theorem 2.5 (with $\beta = \frac{1}{\sqrt{\alpha}}$). For all the plotings here, $\Delta t = \frac{1}{N_t - 1}$.

(Because both $\mathbf{D}$ and $\widetilde{\mathbf{D}}$ are diagonal matrices, the second equality holds.) Substituting this into (2.11) gives the desired diagonalization of $\mathbf{B}_{\mathrm{per}}$. $\square$

It is difficult to get the best choice of the parameter β in theory, such that $\mathrm{Cond}_2(\mathbf{V}(\beta))$ is minimal. Our suggestion for such a parameter is $\beta = \frac{1}{\sqrt{\alpha}}$, because, as we can see Figure 3, with this choice the quantity $\mathrm{Cond}_2(\mathbf{V}(\beta))$ is of order $\mathcal{O}(10^2)$ and this is a dramatic reduction compared to the direct diagonalization. Moreover, we see that the modified diagonalization gives robust condition number with respect to N_t and α.

Remark 2.6. In (2.13) it is important to bring in the diagonal matrix $\widetilde{\mathbf{D}}(\beta)$. This matrix has no effect on the spectrum of $\mathbf{B}_{\mathrm{per}}(\beta)$, but it effects the eigenvector matrix $\mathbf{V}(\beta)$ given by (2.12). Without $\widetilde{\mathbf{D}}(\beta)$ the condition number $\mathrm{Cond}_2(\mathbf{V}(\beta))$ still grows rapidly as the regularization parameter α approaches to 0 or ∞.

We now illustrate that with $\beta = \frac{1}{\sqrt{\alpha}}$ the effect of the roundoff error arising from the diagonalization technique is negligible. To this end, we continue to consider the ODE-constrained optimization problem (2.9) and we show in Figure 4 the numerical solution computed via the modified diagonalization (dash-dot line) and the reference solution (solid line). We fix $N_t = 2049$ and consider two values of α: $\alpha = 10^{-3}$ (top row) and $\alpha = 10^{-6}$ (bottom row). We see that there is no difference between the two numerical solutions. The results shown in Figure 4 imply that the modified diagonalization formula really removes the effect of the roundoff error on the accuracy of the numerical solution.

Remark 2.7 (Implementation of steps (a) and (c) in (1.6))**.** For the parameterized system (2.10), according to Theorem 2.5 we can first solve $[\widehat{Y}, P]^\top$ *via* (1.6) and then get $Y = \beta\widehat{Y}$ (with $\beta = \alpha^{-\frac{1}{2}}$). For the first step (and similarly for the third step) in (1.6), we can solve $\mathbf{X}_1$ in two steps:

① compute $\mathbf{X}_1^{(1)} = \begin{bmatrix} (V \otimes I_x)\widehat{F}_y \\ (V \otimes I_x)F_p \end{bmatrix}$;

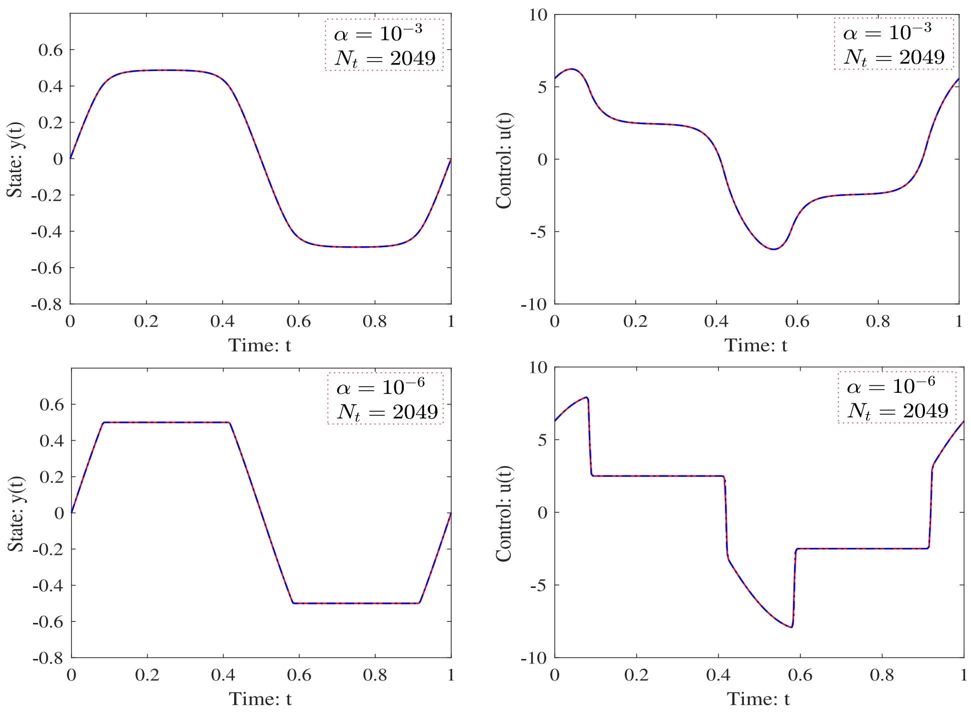


FIGURE 4. Similar to Figure 2, comparison between the numerical solutions computed by using the modified diagonalization of $\mathbf{B}_{\text{per}}$ given by Theorem 2.5 with $\beta = \frac{1}{\sqrt{\alpha}}$ and by directly solving (1.4a) using the `inv` command in Matlab. Here, $\Delta t = \frac{1}{N_t - 1}$ and we choose for α two values: $\alpha = 10^{-3}$ (*top row*) and $\alpha = 10^{-6}$ (*bottom row*). For the trapezoidal rule, *i.e.*, $\theta = \frac{1}{2}$, the results are the same.

② compute $\mathbf{X}_1^{(2)} = \mathbf{W}^{-1}(\beta)\mathbf{X}_1^{(1)}$, where

$$\begin{aligned}
\mathbf{W}^{-1}(\beta) &= \begin{bmatrix} \widetilde{S}_2(\theta) & \\ & \widetilde{S}_1(\beta) \end{bmatrix}^{-1} \begin{bmatrix} -I_t & S_2(\beta) \\ S_1(\beta) & -I_t \end{bmatrix}, \\
\widetilde{S}_1(\theta) &:= (\sqrt{\Lambda^*(\theta)} + S_2(\beta))(S_1(\beta)S_2(\beta) - I_t), \\
\widetilde{S}_2(\theta) &:= (\sqrt{\Lambda(\theta)} + S_1(\beta))(S_1(\beta)S_2(\beta) - I_t).
\end{aligned} \tag{2.14}$$

Since V is a Fourier matrix, we can compute $\mathbf{X}_1^{(1)}$ in step-① *via* FFT. The computation of $\mathbf{X}_1^{(2)}$ in Step-② is trivial, because $S_{1,2}(\beta)$ and $\widetilde{S}_{1,2}(\beta)$ are diagonal matrices.

Remark 2.8 (Implementation of step (b) in (1.6))**.** For the second step in (1.6), for each time point t_n we have to solve a discrete elliptic PDE with a complex 'shift' d_n: $(d_n I_x - \Delta t L_h)\mathbf{X}_{2,n} = \mathbf{X}_{1,n}$. If the discretization matrix L_h is obtained by the finite element method, *i.e.*, $L_h = M_h^{-1}K_h$, it is more convenient to solve the following linear system:

$$(d_n M_h - \Delta t K_h)\mathbf{X}_{2,n} = M_h \mathbf{X}_{1,n}, \tag{2.15}$$

where M_h and K_h denote the mass and stiffness matrices. From (2.8) and (2.6) we have

$$\mathrm{Re}(d_n)=\frac{1}{\theta}\mathrm{Re}\left(\frac{1-\omega^{n-1}}{1+\frac{1-\theta}{\theta}\omega^{n-1}}\right)=\begin{cases}1-\cos(\frac{2(n-1)\pi}{N_t}), & \text{if } \theta=1,\\ 0, & \text{if } \theta=\frac{1}{2},\end{cases}$$

where $\omega=e^{-\frac{2\pi i}{N_t}}$. Hence, it holds that $\mathrm{Re}(d_n)\geq 0$ for all n. This is a good property to design efficient solvers for (2.15). For example, in the case of $\mathbf{L}=\Delta$ (*i.e.*, the PDE in (1.1a) is a heat equation with constant coefficient), the multi-grid method using the Richardson iteration with optimized damping parameter as the smoother works well; see [35].

At the end of this section, we summarize the diagonalization technique in the algorithmic style for clear and better understanding. In the following we denote the time-sequential and time-parallel parts of the algorithm by the symbols $\ominus$ and $\oplus$, respectively.

Algorithm 2.1 Diagonalization technique for PDE-constrained optimization problems with time-periodic condition

```
⊖ Initialization: Input the problem and discretization parameters for the PDE-constrained optimization problems, e.g., $N_t$, $\alpha$ and $\theta$, etc. Form vectors $F_y$ and $F_p$. Let $\beta=\frac{1}{\sqrt{\alpha}}$ and $\widehat{F}_y=\beta^{-1}F_y$.
⊖ 1 Compute $\mathbf{X}_1^{(1)}=\begin{bmatrix}V\otimes I_x & \\ & V\otimes I_x\end{bmatrix}\begin{bmatrix}\widehat{F}_y\\ F_p\end{bmatrix}$ via FFT, where $V$ is the discrete Fourier matrix (cf. Lem. 2.2).
⊕ 2 Compute $\mathbf{X}_1^{(2)}=\mathbf{W}^{-1}(\beta)\mathbf{X}_1^{(1)}$ with $\mathbf{W}(\beta)$ being the matrix given by (2.14).
⊕ 3 Solve linear system $(d_nI_x-\Delta tL_h)\mathbf{X}_{2,n}=\mathbf{X}_{1,n}^{(2)}$ for $n=1,2,\dots,2N_t$, where $L_h$ denotes the space discrete matrix and $d_n$ is the $n$-th diagonal element of the matrix $\mathbf{D}$ in (2.8).
⊕ 4 Compute $\mathbf{X}_3=\mathbf{W}(\beta)\mathbf{X}_2$.
⊖ 5 Compute $\begin{bmatrix}\widehat{Y}\\ P\end{bmatrix}=\begin{bmatrix}V^{-1}\otimes I_x & \\ & V^{-1}\otimes I_x\end{bmatrix}\mathbf{X}_3$ via inverse FFT and then let $Y=\beta\widehat{Y}$.
```

3. Initial-value PDE optimization problems

For the initial-value case, we need to compute discrete solution values $\{p_0,\dots,p_{N_t-1}\}$, instead of $\{p_1,\dots,p_{N_t}\}$, because in this case $p_{N_t}=0$. Define

$$\begin{gathered}P=\begin{bmatrix}p_0(x)\\ p_1(x)\\ \vdots\\ p_{N_t-1}(x)\end{bmatrix}, F_1=\begin{bmatrix}(I_x+(1-\theta)\Delta tL_h)y_{\mathrm{ini}}(x)\\ 0\\ \vdots\\ 0\end{bmatrix}, F_2=\Delta t\begin{bmatrix}\bar{y}_{\theta,0}(x)-(1-\theta)y_{\mathrm{ini}}(x)\\ \bar{y}_{\theta,1}(x)\\ \vdots\\ \bar{y}_{\theta,N_t-1}(x)\end{bmatrix},\\ C_{\mathrm{ini},1}=\begin{bmatrix}1 & & & & \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ & & & -1 & 1\end{bmatrix}, C_{\mathrm{ini},2}=\begin{bmatrix}\theta & & & & \\ 1-\theta & \theta & & & \\ & \ddots & \ddots & & \\ & & 1-\theta & \theta & \\ & & & 1-\theta & \theta\end{bmatrix}\in\mathbb{R}^{N_t\times N_t},\end{gathered}\tag{3.1}$$

where $\theta=1$ or $\theta=\frac{1}{2}$ and $\bar{y}_{\theta,n}=\theta\bar{y}_n+(1-\theta)\bar{y}_{n+1}$. Similar to (2.3), we get the discrete version of (1.3) as

$$\begin{aligned}&(C_{\mathrm{ini},1}\otimes I_x)Y-\Delta t(C_{\mathrm{ini},2}\otimes L_h)Y-\Delta t\alpha^{-1}(C_{\mathrm{ini},2}^{\top}\otimes I_x)P=F_1,\\ &(C_{\mathrm{ini},1}^{\top}\otimes I_x)P-\Delta t(C_{\mathrm{ini},2}^{\top}\otimes L_h)P+\Delta t(C_{\mathrm{ini},2}\otimes I_x)Y=F_2.\end{aligned}\tag{3.2}$$

3.1. Preconditioner design: the classical approach

We can represent (3.2) as

$$\underbrace{\begin{bmatrix} \Delta t\mathcal{I} & 0 & \mathcal{K}_2 \\ 0 & \Delta t\alpha\mathcal{I} & -\Delta t\mathcal{I} \\ \mathcal{K}_1 & -\Delta t\mathcal{I} & 0 \end{bmatrix}}_{:=\mathcal{Q}} \begin{bmatrix} Y \\ U \\ P \end{bmatrix} = \begin{bmatrix} (C_{\text{ini},2}\otimes I_x)^{-1}F_2 \\ 0 \\ (C_{\text{ini},2}^{\top}\otimes I_x)^{-1}F_1 \end{bmatrix}, \tag{3.3a}$$

where $\mathcal{I} = I_t \otimes I_x$ and

$$\begin{aligned} \mathcal{K}_1 &= (C_{\text{ini},2}^{\top}\otimes I_x)^{-1}\left(C_{\text{ini},1}\otimes I_x - \Delta t C_{\text{ini},2}\otimes L_h\right), \\ \mathcal{K}_2 &= (C_{\text{ini},2}\otimes I_x)^{-1}\left(C_{\text{ini},1}^{\top}\otimes I_x - \Delta t C_{\text{ini},2}^{\top}\otimes L_h\right). \end{aligned} \tag{3.3b}$$

For system (3.3a), the study of the Krylov subspace methods, *e.g.*, the restarted GMRES method, is popular in recent years. The main point of this kind of methods is to design a good preconditioner $\mathcal{P}$, such as the following block lower triangular matrix:

$$\mathcal{P} = \begin{bmatrix} \Delta t\mathcal{I} & 0 & 0 \\ 0 & \Delta t\alpha\mathcal{I} & 0 \\ \mathcal{K}_1 & -\Delta t\mathcal{I} & \widehat{\mathbf{S}} \end{bmatrix}, \tag{3.4}$$

where $\widehat{\mathbf{S}}$ denotes an approximation of the (negative) Schur complement of $\mathcal{Q}$:

$$\mathbf{S}_{\text{exact}} = \Delta t^{-1}\mathcal{K}_1\mathcal{K}_2 + \Delta t\alpha^{-1}\mathcal{I}. \tag{3.5}$$

With $\widehat{\mathbf{S}} = \mathbf{S}_{\text{exact}}$, the GMRES method converges in 1 iteration [29]. It is however unpractical to use $\mathbf{S}_{\text{exact}}$, because it is expensive to compute $\mathbf{S}_{\text{exact}}^{-1}$ in each iteration[3].

A practical choice of $\widehat{\mathbf{S}}$, which was studied extensively in the literature [6, 18, 19, 30, 32], is of the following form

$$\widehat{\mathbf{S}} = \frac{1}{\Delta t}\left(\mathcal{K}_1 + \frac{\Delta t}{\sqrt{\alpha}}\mathcal{I}\right)\left(\mathcal{K}_2 + \frac{\Delta t}{\sqrt{\alpha}}\mathcal{I}\right). \tag{3.6}$$

We denote the preconditioned matrix by

$$\mathbf{\Gamma}_{\text{cla}} := \widehat{\mathbf{S}}^{-1}\mathbf{S}_{\text{exact}}. \tag{3.7}$$

3.2. Preconditioner design: a new approach

We propose in this section a new preconditioner which can be used in a PinT pattern. To this end, similar to Section 2.2.2 we first equivalently transform (3.2) to the following form (with $\widehat{Y} = \beta^{-1}Y$):

$$\begin{aligned} &(C_{\text{ini},1}\otimes I_x)\widehat{Y} - \Delta t(C_{\text{ini},2}\otimes L_h)\widehat{Y} - \Delta t(\alpha\beta)^{-1}(C_{\text{ini},2}^{\top}\otimes I_x)P = \frac{1}{\beta}F_1, \\ &(C_{\text{ini},1}^{\top}\otimes I_x)P - \Delta t(C_{\text{ini},2}^{\top}\otimes L_h)P + \Delta t\beta(C_{\text{ini},2}\otimes I_x)\widehat{Y} = F_2, \end{aligned} \tag{3.8}$$

[3]For any input vector $\mathbf{b}$, computing $\mathbf{S}_{\text{exact}}^{-1}\mathbf{b}$ is equal to solve a linear problem $\mathbf{S}_{\text{exact}}\mathbf{r} = \mathbf{b}$. This procedure is time-consuming, because it requires to solve a forward evolution PDE and then a backward evolution PDE.

and this can be represented as follows

$$\left(\underbrace{\mathbf{B}_{\text{ini}}(\beta)\otimes I_x-\Delta t\mathbf{I}_t\otimes L_h}_{:=\mathbf{K}_{\text{ini}}(\beta)}\right)\begin{bmatrix}\widehat{Y}\\ P\end{bmatrix}=\begin{bmatrix}F_y\\ F_p\end{bmatrix}, \tag{3.9a}$$

where $F_y=\frac{1}{\beta}(C_{\text{ini},2}^{-1}\otimes I_x)F_1$, $F_p=((C_{\text{ini},2}^{\top})^{-1}\otimes I_x)F_2$ and

$$\mathbf{B}_{\text{ini}}(\beta):=\begin{bmatrix}C_{\text{ini},2}^{-1}C_{\text{ini},1} & -\Delta t(\alpha\beta)^{-1}C_{\text{ini},2}^{-1}C_{\text{ini},2}^{\top}\\ \Delta t\beta(C_{\text{ini},2}^{\top})^{-1}C_{\text{ini},2} & (C_{\text{ini},1}C_{\text{ini},2}^{-1})^{\top}\end{bmatrix}. \tag{3.9b}$$

Now, with the matrices $C_{\text{per},1}$ and $C_{\text{per},2}$ given by (2.2), we define

$$\begin{aligned}\widehat{\mathbf{B}}_{\text{per}}(\beta)&:=\begin{bmatrix}C_{\text{per},2}^{-1}C_{\text{per},1} & -\Delta t(\alpha\beta)^{-1}C_{\text{per},2}^{-1}C_{\text{per},2}^{\top}\\ \Delta t\beta(C_{\text{per},2}^{\top})^{-1}C_{\text{per},2} & (C_{\text{per},1}C_{\text{per},2}^{-1})^{\top}\end{bmatrix},\\ \widehat{\mathbf{K}}_{\text{per}}(\beta)&=\widehat{\mathbf{B}}_{\text{per}}(\beta)\otimes I_x-\Delta t\begin{bmatrix}I_t & \\ & I_t\end{bmatrix}\otimes L_h.\end{aligned} \tag{3.10}$$

We use $\widehat{\mathbf{K}}_{\text{per}}(\beta)$ as a preconditioner for $\mathbf{K}_{\text{ini}}(\beta)$ in (3.9a). For the Backward-Euler method, we have $C_{\text{per},2}=I_t$ and therefore $\widehat{\mathbf{B}}_{\text{per}}(\beta)=\mathbf{B}_{\text{per}}(\beta)$, where $\mathbf{B}_{\text{per}}(\beta)$ is given by (2.4b). However, for the trapezoidal rule these two matrices are not equal. Similar to (3.7), here we denote the preconditioned matrix by

$$\mathbf{\Gamma}_{\text{new}}:=\left(\widehat{\mathbf{B}}_{\text{per}}(\beta)\otimes I_x-\Delta t\begin{bmatrix}I_t & \\ & I_t\end{bmatrix}\otimes L_h\right)^{-1}\left(\mathbf{B}_{\text{ini}}(\beta)\otimes I_x-\Delta t\begin{bmatrix}I_t & \\ & I_t\end{bmatrix}\otimes L_h\right). \tag{3.11}$$

We note that the sizes of $\mathbf{\Gamma}_{\text{cla}}$ is $N_tN_x\times N_tN_x$, while the size of $\mathbf{\Gamma}_{\text{new}}$ is $2N_tN_x\times 2N_tN_x$.

To solve (3.9a) by some preconditioned iterative method using the preconditioner $\widehat{\mathbf{K}}_{\text{per}}(\beta)$, the major computation cost for each iteration is to compute $\widehat{\mathbf{K}}_{\text{per}}^{-1}\mathbf{r}$ with some input $\mathbf{r}$. Similar to the diagonalization of the matrix $\mathbf{B}_{\text{per}}(\beta)$ in Section 2.2, we can also diagonalize the matrix $\widehat{\mathbf{B}}_{\text{per}}(\beta)$ with closed formulas and thus we can solve this linear problem *via* the diagonalization technique as well. The algorithmic formula for computing $\widehat{\mathbf{K}}_{\text{per}}^{-1}\mathbf{r}$ is similar to Algorithm 2.1 and details should be omitted.

Theorem 3.1 (Clustering of the eigenvalues and singular values)**.** *For the matrix $\mathbf{\Gamma}_{\text{new}}$ defined by (3.11), it holds that*
1. if $\theta=1$, i.e., the Backward-Euler method, $\mathbf{\Gamma}_{\text{new}}$ has at most $2N_x$ eigenvalues that do not equal to 1 and at most $6N_x$ singular values that do not equal to 1;
2. if $\theta=\frac{1}{2}$, i.e., the trapezoidal rule, $\mathbf{\Gamma}_{\text{new}}$ has at most $4N_x$ eigenvalues that do not equal to 1 and at most $12N_x$ singular values that do not equal to 1.

Proof. We denote by

$$\mathbf{E}=\mathbf{K}_{\text{ini}}(\beta)-\widehat{\mathbf{K}}_{\text{per}}(\beta). \tag{3.12}$$

For the case $\theta=1$, we have $C_{\text{per},2}=C_{\text{ini},2}=I_t$. Hence,

$$\mathbf{E}=\begin{bmatrix}uv^{\top} & 0\\ 0 & vu^{\top}\end{bmatrix}\otimes I_x,\ u=\begin{bmatrix}1\\0\\ \vdots\\0\end{bmatrix},\ v=\begin{bmatrix}0\\ \vdots\\0\\1\end{bmatrix}.$$

It is clear that $\mathrm{rank}(\mathbf{E}) = 2N_x$. Therefore, we have

$$\mathrm{rank}(\mathbf{\Gamma}_{\mathrm{new}} - \mathbf{I}_t \otimes I_x) = \mathrm{rank}\left(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)(\widehat{\mathbf{K}}_{\mathrm{per}}(\beta) + \mathbf{E}) - \mathbf{I}_t \otimes I_x\right) = \mathrm{rank}\left(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1}\mathbf{E}\right) = 2N_x.$$

This implies that the matrix $\mathbf{\Gamma}_{\mathrm{new}}$ has at most $2N_x$ eigenvalues that do not equal to 1. For the singular values, we need to estimate the rank of $\mathbf{\Gamma}_{\mathrm{new}}\mathbf{\Gamma}_{\mathrm{new}}^\top - \mathbf{I}_t \otimes I_x$.[4] We have

$$\begin{aligned}\mathbf{\Gamma}_{\mathrm{new}}\mathbf{\Gamma}_{\mathrm{new}}^\top &= \widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)(\widehat{\mathbf{K}}_{\mathrm{per}}(\beta) + \mathbf{E})(\widehat{\mathbf{K}}_{\mathrm{per}}(\beta) + \mathbf{E})^\top(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta))^\top \\ &= \mathbf{I}_t \otimes I_x + \widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)\mathbf{E} + \left(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)\mathbf{E}\right)^\top + \widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)\mathbf{E}\mathbf{E}^\top(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1})^\top.\end{aligned}$$

From this we have

$$\begin{aligned}\mathrm{rank}\left(\mathbf{\Gamma}_{\mathrm{new}}\mathbf{\Gamma}_{\mathrm{new}}^\top - \mathbf{I}_t \otimes I_x\right) &\le 2\mathrm{rank}(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)\mathbf{E}) + \mathrm{rank}(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1}(\beta)\mathbf{E}\mathbf{E}^\top(\widehat{\mathbf{K}}_{\mathrm{per}}^{-1})^\top) \\ &\le 2\mathrm{rank}(\mathbf{E}) + \mathrm{rank}(\mathbf{E}\mathbf{E}^\top) = 3\mathrm{rank}(\mathbf{E}) = 6N_x.\end{aligned}$$

Hence, there are at most $6N_x$ singular values of $\mathbf{\Gamma}_{\mathrm{new}}\mathbf{\Gamma}_{\mathrm{new}}^\top$ that do not equal to 1.

We next consider the case $\theta = \frac{1}{2}$, *i.e.*, the trapezoidal rule. We claim that in this case the matrix $\mathbf{E}$ defined by (3.12) satisfies

$$\mathrm{rank}(\mathbf{E}) = 4N_x. \tag{3.13}$$

If this is true, we can prove the second result in Theorem 3.1 by following the same analysis for the case $\theta = 1$. Since $\mathbf{E} = (\mathbf{B}_{\mathrm{ini}}(\beta) - \widehat{\mathbf{B}}_{\mathrm{per}}(\beta)) \otimes I_x$, it is sufficient to estimate $\mathrm{rank}(\mathbf{B}_{\mathrm{ini}}(\beta) - \widehat{\mathbf{B}}_{\mathrm{per}}(\beta))$. From (2.2) and (3.1), we have

$$\begin{aligned}&\mathbf{B}_{\mathrm{ini}}(\beta) - \widehat{\mathbf{B}}_{\mathrm{per}}(\beta) \\ &= \begin{bmatrix} C_{\mathrm{ini},2}^{-1}C_{\mathrm{ini},1} - C_{\mathrm{per},2}^{-1}C_{\mathrm{per},1} & -\Delta t(\alpha\beta)^{-1}\left[C_{\mathrm{ini},2}^{-1}C_{\mathrm{ini},2}^\top - C_{\mathrm{per},2}^{-1}C_{\mathrm{per},2}^\top\right] \\ \Delta t\beta\left[(C_{\mathrm{ini},2}^\top)^{-1}C_{\mathrm{ini},2} - (C_{\mathrm{per},2}^\top)^{-1}C_{\mathrm{per},2}\right] & (C_{\mathrm{ini},1}C_{\mathrm{ini},2}^{-1})^\top - (C_{\mathrm{per},1}C_{\mathrm{per},2}^{-1})^\top \end{bmatrix},\end{aligned}$$

and a routine calculation yields

$$\begin{aligned}&C_{\mathrm{ini},2}^{-1}C_{\mathrm{ini},1} - C_{\mathrm{per},2}^{-1}C_{\mathrm{per},1} = 2[\mathbf{e}, -\mathbf{e}, \dots, \mathbf{e}] \in \mathbb{R}^{N_t \times N_t}, \\ &C_{\mathrm{ini},2}^{-1}C_{\mathrm{ini},2}^\top - C_{\mathrm{per},2}^{-1}C_{\mathrm{per},2}^\top = [\tilde{\mathbf{e}}, 0, \dots, 0] \in \mathbb{R}^{N_t \times N_t}, \\ &(C_{\mathrm{ini},2}^\top)^{-1}C_{\mathrm{ini},2} - (C_{\mathrm{per},2}^\top)^{-1}C_{\mathrm{per},2} = [0, \dots, 0, \bar{\mathbf{e}}] \in \mathbb{R}^{N_t \times N_t}, \\ &(C_{\mathrm{ini},1}C_{\mathrm{ini},2}^{-1})^\top - (C_{\mathrm{per},1}C_{\mathrm{per},2}^{-1})^\top = 2[\mathbf{e}, -\mathbf{e}, \dots, \mathbf{e}] \in \mathbb{R}^{N_t \times N_t},\end{aligned}$$

where N_t is an odd integer[5], $\mathbf{e} = (-1, 1, \dots, -1)^\top \in \mathbb{R}^{N_t}$, $\tilde{\mathbf{e}} = (-1, 1, \dots, -1, 1, 0)^\top$ and $\bar{\mathbf{e}} = (0, 1, -1, \dots, 1, -1)$. Clearly, the matrix $\mathbf{B}_{\mathrm{ini}}(\beta) - \widehat{\mathbf{B}}_{\mathrm{per}}(\beta)$ consists of 4 *independent* vectors:

$$\begin{bmatrix} 2\mathbf{e} \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 2\mathbf{e} \end{bmatrix}, \begin{bmatrix} 2\mathbf{e} \\ \Delta t\beta\bar{\mathbf{e}} \end{bmatrix}, \begin{bmatrix} -\Delta t(\alpha\beta)^{-1}\tilde{\mathbf{e}} \\ 2\mathbf{e} \end{bmatrix}.$$

Hence, $\mathrm{rank}\,(\mathbf{B}_{\mathrm{ini}}(\beta) - \widehat{\mathbf{B}}_{\mathrm{per}}(\beta)) = 4$ and this gives (3.13). □

[4]This is because of the fact that for any square matrix G a non-zero singular value of G is equal to the square root of some eigenvalue of $GG^\top$.

[5]The reason why N_t should be odd is explained in Remark 2.1.

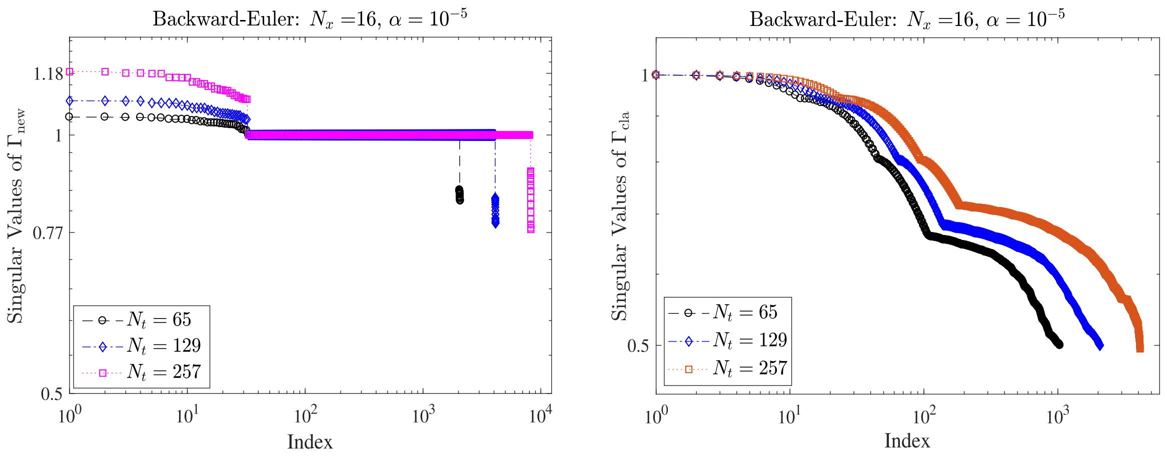


FIGURE 5. For the Backward-Euler method, the distributions of the singular values of the matrices $\mathbf{\Gamma}_{\rm new}$ (*left*) and $\mathbf{\Gamma}_{\rm cla}$ (*right*) for three values of N_t.

Remark 3.2. From Theorem 3.1, we see that for the preconditioned matrix $\mathbf{\Gamma}_{\rm new}$ the number of both the eigenvalues and singular values not equal to 1 is independent of N_t. This implies that, for fixed N_x increasing the number of time points does not effect the convergence rate of the Krylov subspace solvers.

We now make some comparisons for the clustering of the eigenvalues and singular values of $\mathbf{\Gamma}_{\rm cla}$ and $\mathbf{\Gamma}_{\rm new}$ by using the following representative model:

$$\partial_{xx}v|_{\partial_x v(0)=\partial_x v(1)=0} \approx L_h V, \text{with } L_h = \frac{1}{\Delta x^2}\begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix}_{N_x\times N_x} \tag{3.14}$$

where $N_x = \frac{1}{\Delta x}$. It is clear that $\sigma(L_h) \subseteq [0, \infty)$. (For $\mathbf{\Gamma}_{\rm new}$, we use $\beta = \frac{1}{\sqrt{\alpha}}$ for all plottings in the following.) We plot in Figures 5 and 6 the distributions of the singular values of the two preconditioned matrices $\mathbf{\Gamma}_{\rm new}$ and $\mathbf{\Gamma}_{\rm cla}$ when the Backward-Euler method is used as the time-integrator. Here, we consider four values of N_t: 65, 129 and 257 and the step-size Δt is fixed by $\Delta t = \frac{1}{N_t - 1}$. For $\mathbf{\Gamma}_{\rm cla}$, we see that both the singular values and the eigenvalues distribute in the interval $[0.5, 1]$ and this confirms the previous study in [30]. The matrix $\mathbf{\Gamma}_{\rm new}$ does not satisfy this property, but the clustering of the eigenvalues and singular values seems better.

We next consider the trapezoidal rule, for which we show similar results in Figures 7 and 8. We see that for $\mathbf{\Gamma}_{\rm cla}$ it still holds that $\lambda(\mathbf{\Gamma}_{\rm cla}) \in [\frac{1}{2}, 1]$, but this does not hold for the singular values. Moreover, we see that for both $\mathbf{\Gamma}_{\rm new}$ and $\mathbf{\Gamma}_{\rm cla}$, compared to the case of Backward-Euler method, the singular values distribute in a larger interval. Different from the case of Backward-Euler method, the clustering of the singular values and eigenvalues of $\mathbf{\Gamma}_{\rm new}$ is worse than that of $\mathbf{\Gamma}_{\rm cla}$.

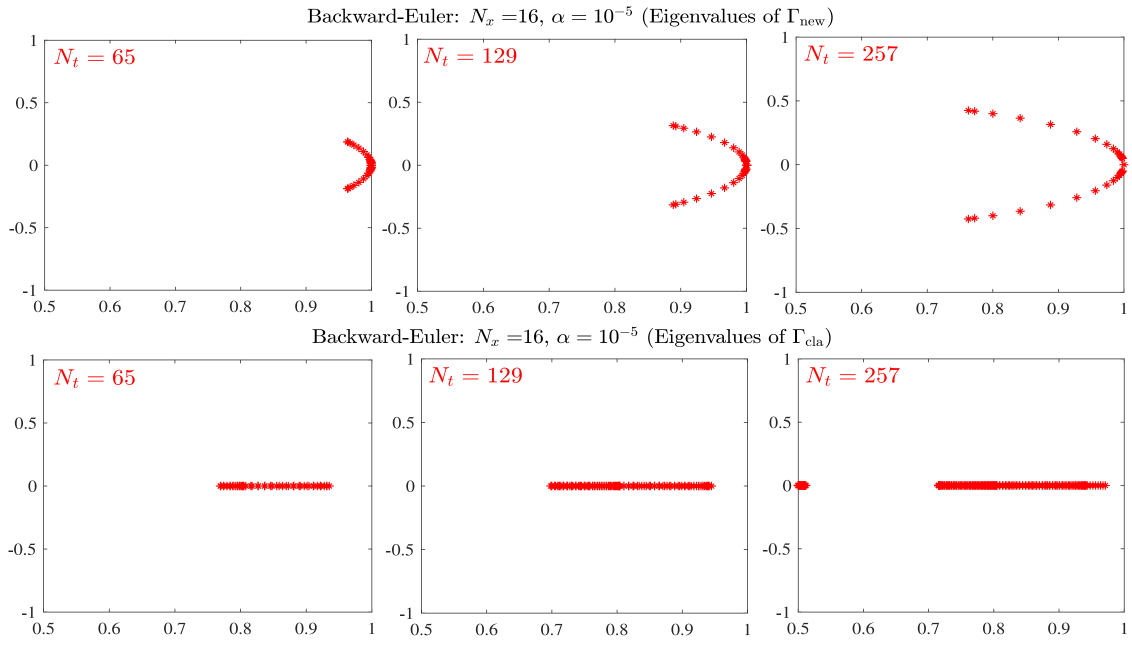


FIGURE 6. For the Backward-Euler method, the distributions of the eigenvalues of the matrices $\mathbf{\Gamma}_{\rm new}$ (*top row*) and $\mathbf{\Gamma}_{\rm cla}$ (*bottom row*) on the complex plane.

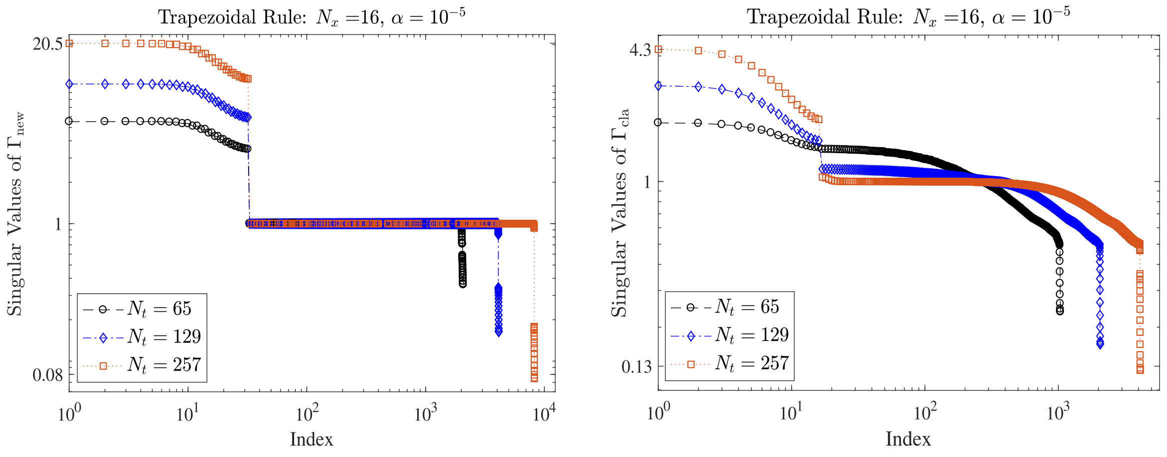


FIGURE 7. For the trapezoidal rule, distributions of the singular values of the matrices $\mathbf{\Gamma}_{\rm new}$ (*left*) and $\mathbf{\Gamma}_{\rm cla}$ (*right*) for three values of N_t.

4. NUMERICAL RESULTS

In this section, we show numerical results to illustrate the advantages of the proposed algorithm. We consider the following 2D problem with time-periodic condition or initial-value condition:

$$\begin{cases} \min \mathcal{J}(y,u) := \frac{1}{2}\int_0^2\int_\Omega (y(\mathbf{x},t)-\bar{y}(\mathbf{x},t))^2 \mathrm{d}\mathbf{x}\mathrm{d}t + \frac{\alpha}{2}\int_0^2\int_\Omega u^2(\mathbf{x},t)\mathrm{d}\mathbf{x}\mathrm{d}t, \\ \partial_t y - \partial_{x_1}(e^{-5x_1}\partial_{x_1} y) - \partial_{x_2x_2} y = u, \ (x_1,x_2,t)\in\Omega\times(0,2), \\ \text{with } \bar{y}(\mathbf{x},t) = \frac{(x_1^2-\frac{1}{4})(x_2^2-\frac{1}{4})}{x_1^2+x_2^2+\frac{1}{10}} \sin\left(5\pi x_1^2\sqrt{\frac{1}{2}-x_2}+2\pi t\right), \ (x_1,x_2,t)\in\Omega\times(0,2), \end{cases} \tag{4.1}$$

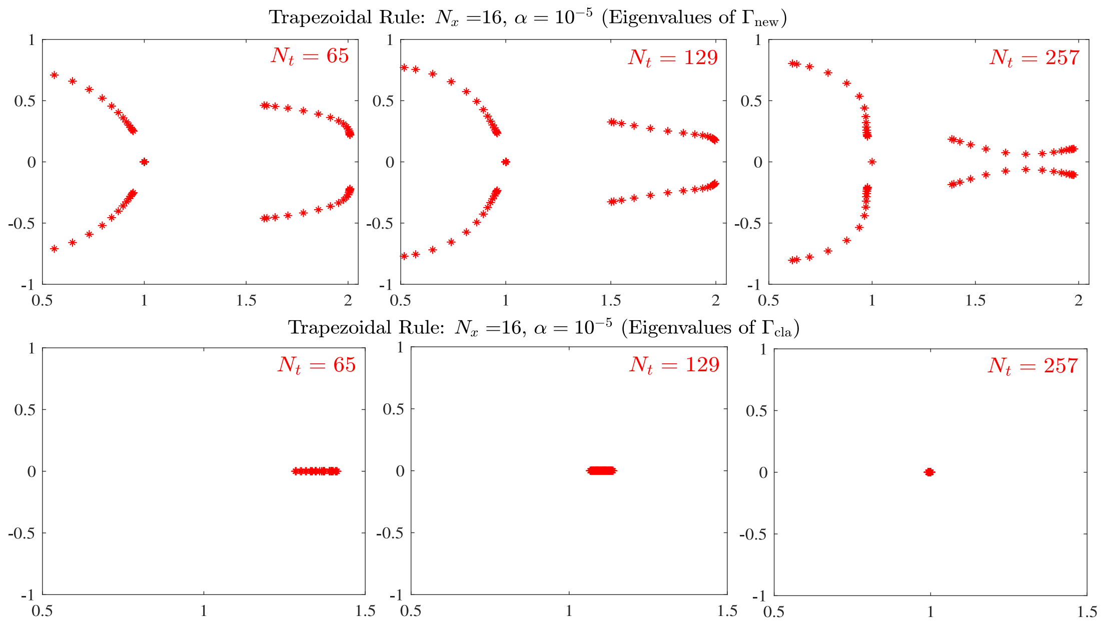


FIGURE 8. For the trapezoidal rule, distributions of the eigenvalues of the matrices $\mathbf{\Gamma}_{\rm new}$ (*top row*) and $\mathbf{\Gamma}_{\rm cla}$ (*bottom row*) on the complex plane.

where $\mathbf{x} := (x_1, x_2)$ and $\Omega = (-\frac{1}{2}, \frac{1}{2}) \times (-\frac{1}{2}, \frac{1}{2})$. We discretize the Laplacian by FEM with linear basis functions on triangles and this gives the discrete matrix L_h in (1.4a) as $L_h = M_h^{-1} K_h$, where M_h and K_h are respectively the mass matrix and the stiffness matrix of the FEM discretization. As mentioned in Remark 2.8, in practice the second step of the diagonalization technique (1.6) is implemented as

$$(d_n M_h - \Delta t K_h)\mathbf{X}_{2,n} = M_h \mathbf{X}_{1,n}, \ \ n = 1, 2, \dots, 2N_t.$$

We solve this algebraic system by the Trilinos ML AMG package. All the numerical results are obtained with the following software/hardware configurations:

- **CPU**: Intel Core i7-3770K 3.5 GHz and 32 GB RAM using gcc 4.8.1. A single CPU was used for the sequential computation and the codes were tested with gcc's fast math option (`ffast_math`).
- **GPU**: NVIDIA GeForce GTX 660 installed in a system with the above described CPU. The GPU operates at 1.10 GHz clock speed and consists of 8 multiprocessors (each contains 192 CUDA cores). We compiled the code using CUDA version 5.5 in combination with the gcc 4.8.1 compiler with fast math option (`use_fast_math`).

4.1. The time-periodic condition

We first consider the time-periodic condition $y(\mathbf{x}, 0) = y(\mathbf{x}, 2)$ for $\mathbf{x} \in \Omega$. In Figure 9, we show the filled contour plots of the computed state $y(\mathbf{x}, t)$ and the desired state $\bar{y}(\mathbf{x}, t)$ at $t = 0, 2$ (the initial and final time points), $t = 0.4$, $t = 0.92$ and $t = 1.8$. We see that the computed state is close to the desired state.

We next show the computation time of solving (4.1) *via* two strategies: the PinT computation by the diagonalization technique and the sequential-in-time computation by two different methods proposed in [32] and [14]. The first sequential-in-time method is based on solving the discrete KKT system by the preconditioned GMRES method, where the preconditioner is constructed *via* a novel approximation of the Schur complement of the KKT system. Let $\mathcal{K}_{1,\rm per} = (C_{{\rm per},2}^\top \otimes I_x)^{-1} (C_{{\rm per},1} \otimes I_x - \Delta t C_{{\rm per},2} \otimes L_h)$ and

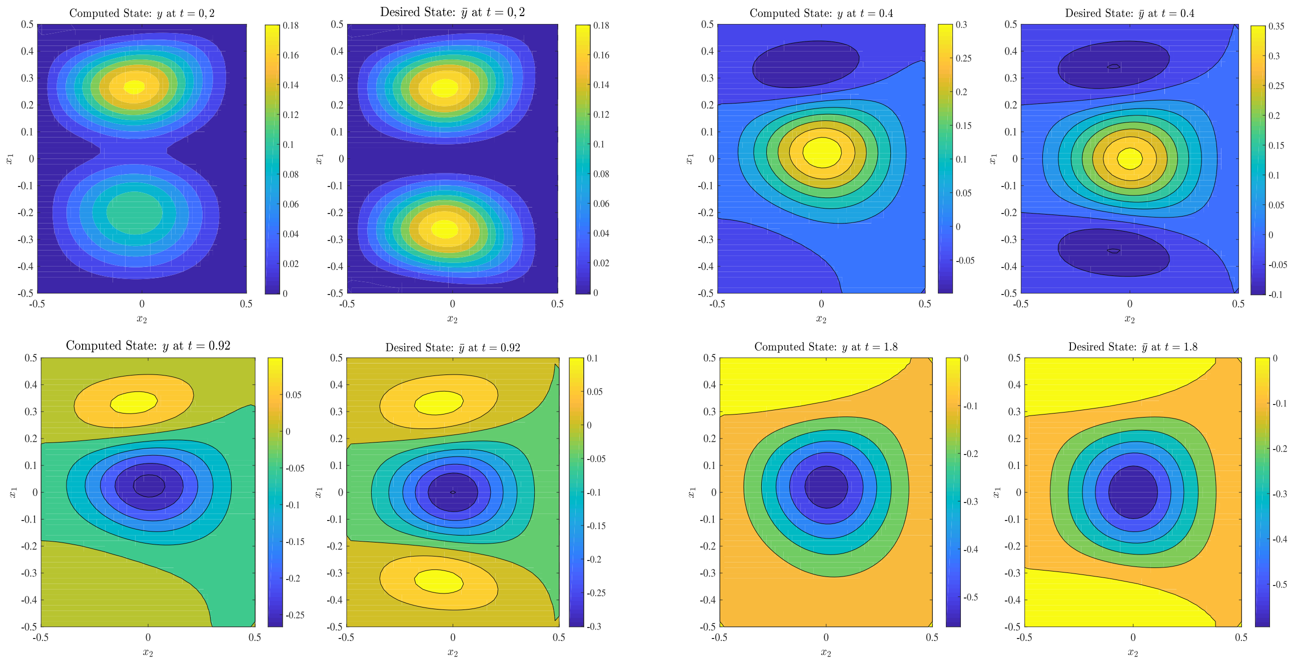


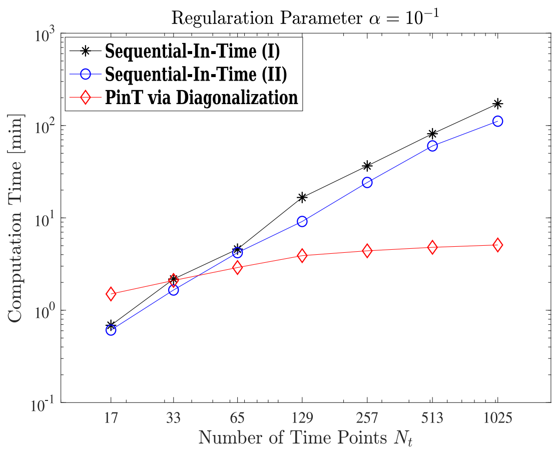
FIGURE 9. For the optimal control problem (4.1) with time-periodic condition $y(\mathbf{x}, 0) = y(\mathbf{x}, 2)$ and $\alpha = 10^{-5}$, the filled contour plots of the computed state $y(x_1, x_2, t)$ and the desired state $\bar{y}(x_1, x_2, t)$ at $t = 0, 2$ (*top left*), $t = 0.4$ (*top right*), $t = 0.92$ (*bottom left*) and $t = 1.8$ (*bottom right*).

$\mathcal{K}_{2,\text{per}} = (C_{\text{per},2} \otimes I_x)^{-1} \left(C_{\text{per},1}^\top \otimes I_x - \Delta t C_{\text{per},2}^\top \otimes L_h \right)$ be respectively the space-time discretization matrices of the state equation and the adjoint equation of problem (4.1). Then, the major computation cost for getting the approximate Schur complement lies in approximating $\mathcal{K}_{1,\text{per}}^{-1}$ and $\mathcal{K}_{2,\text{per}}^{-1}$, for which we use the stationary iteration approach together with a V-cycle multi-grid method as the inner solver (as in [19, 30]); see ([32], Sect. 5.2) for more details. For the second sequential-in-time method used for comparison, the main idea is as follows. First, by using a suitable shooting operators associated with the spatial second-order operator $\partial_{x_1}(e^{-5x_1}\partial_{x_1}) + \partial_{x_2x_2}$, we obtain a reduced version of problem (4.1) for which we can derive a KKT system. By an eigenfunction expansion of the spatial second-order operator, we can represent the shooting operators and their adjoints by series expansions. Then, we solve such a KKT system *via* a preconditioned fixed-point iteration, where the preconditioner is constructed by truncating the series expansions of the shooting operators and their adjoints. For practical computation, we only needs to handle the dominant part of the spectrum of the spatial second-order operator to obtain fast convergence rates and thus the preconditioner can be realized well by coarse grid discretizations (the details for this aspect can be found in [14], Sect. 5).

Let the space mesh size be $h = 0.01$. Then, we show in Figure 10 the measured computation time of the PinT computation and the aforementioned sequential-in-time computation, when the number of time points N_t varies. Here $\Delta t = \frac{1}{N_t - 1}$ and we consider two regularization parameters $\alpha = 10^{-1}$ (left subfigure) and $\alpha = 10^{-3}$ (right subfigure). Similarly, for the case that N_t is fixed and α varies, we show in Figure 11 the comparison of the computation time of these three methods.[6] It is clear that, compared to the two sequential-in-time methods, the PinT computation *via* the diagonalization technique dramatically reduces the computation time.

We next illustrate in Figure 12 the evolution of the numerical control variable $u(x_1, 0, t)$ at $x_2 = 0$ generated by the PinT computation and the method 'Sequential-In-Time (I)' [32], after 2, 4 and 6 minutes. Here, we consider $h = 0.01$, $N_t = 257$ and $\alpha = 10^{-5}$. (As we mentioned above, the sequential-in-time method in [14] does

[6]The method 'Sequential-In-Time (II)' , *i.e.*, the method proposed in [14], diverges for $\alpha \leq 10^{-4}$.



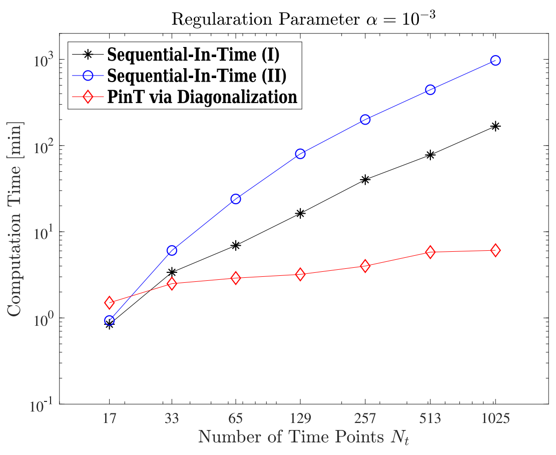
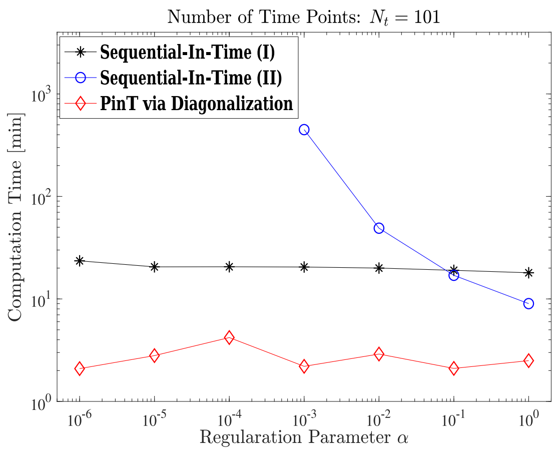


FIGURE 10. For $h = 0.01$ and N_t varies, the measured computation time for the sequential-in-time computation and the PinT computation *via* the diagonalization technique. The 'Sequential-In-Time (I)' denotes the method in [32] and the 'Sequential-In-Time (II)' denotes the method in [14].



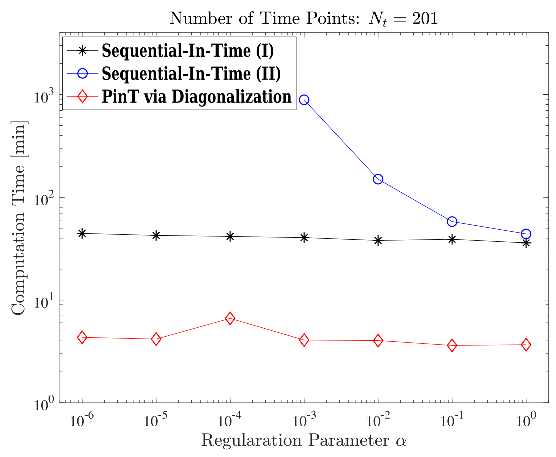


FIGURE 11. Similar to Figure 10, for $h = 0.01$ and α varies the measured computation time for the sequential-in-time computation and the PinT computation *via* the diagonalization technique. The method 'Sequential-In-Time (II)' diverges for $\alpha \leq 10^{-4}$.

not converge for $\alpha \leq 10^{-4}$, so we only consider the method in [32] here.) We see that, after 6 minutes we finish approximately 92 % of the whole computation by the diagonalization technique, while by the sequential-in-time method only 9% is finished.

4.2. The initial-value condition

We next consider the following initial-value condition for the optimal control problem (4.1):

$$\mathbf{y}(\mathbf{x}, 0) = \bar{y}(\mathbf{x}, 0) = \frac{(x_1^2 - \frac{1}{4})(x_2^2 - \frac{1}{4})}{x_1^2 + x_2^2 + \frac{1}{10}} \sin\left(5\pi x_1^2 \sqrt{\frac{1}{2} - x_2}\right). \tag{4.2}$$

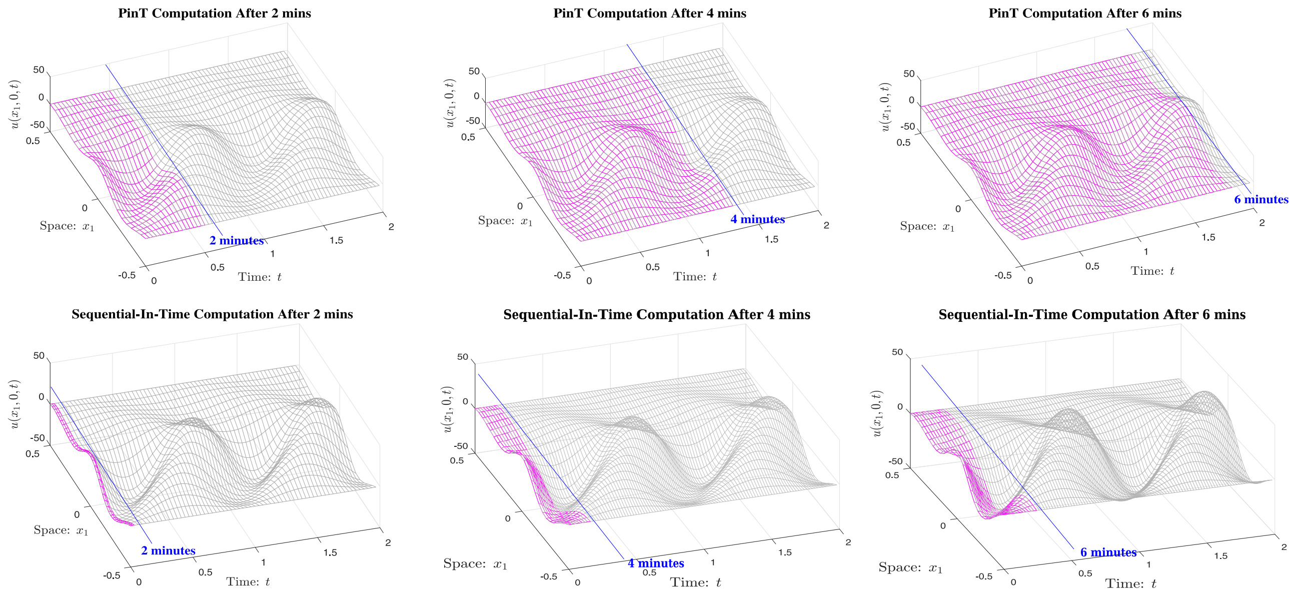


FIGURE 12. For $h = 0.01$, $N_t = 257$ and $\alpha = 10^{-5}$, the evolution of the computed control $u(x_1, 0, t)$ at $x_2 = 0$ after $t = 2$ minutes, $t = 4$ minutes and $t = 6$ minutes. *Top row*: PinT computation *via* the diagonalization technique; *Bottom row*: sequential-in-time computation by the method in [32].

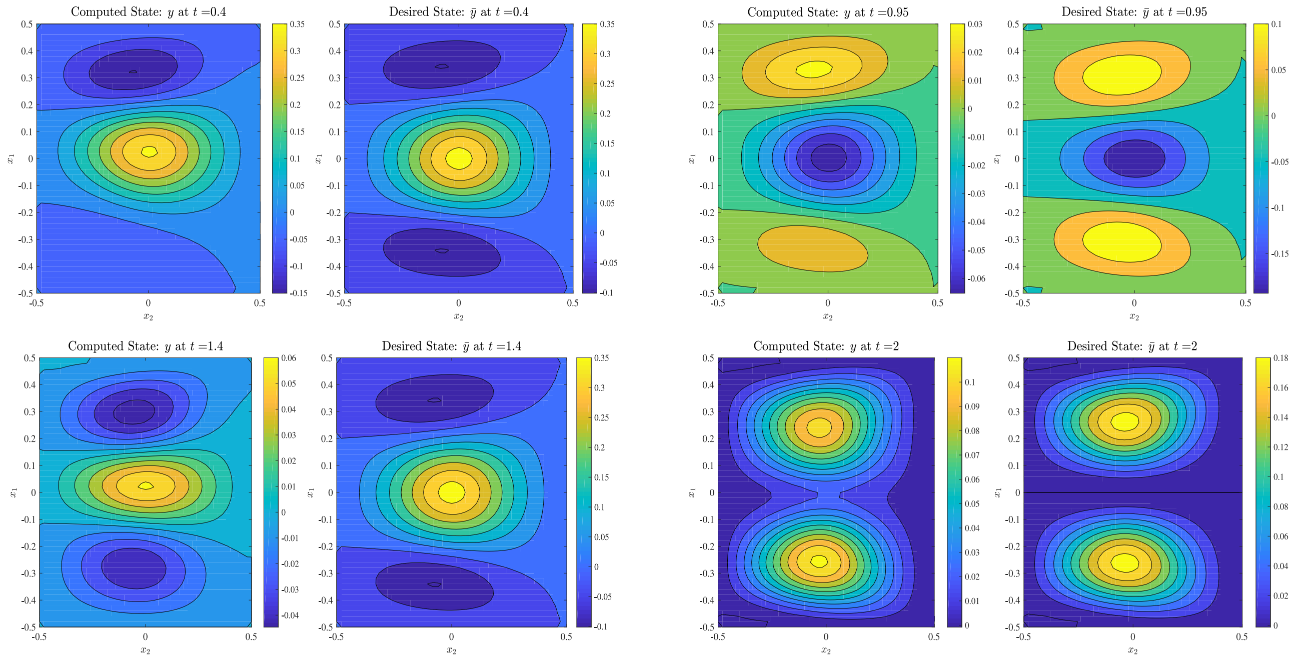


FIGURE 13. For the optimal control problem (4.1) with initial-value condition (4.2) and $\alpha = 10^{-5}$, the filled contour plots of the computed state $y(x_1, x_2, t)$ and the desired state $\bar{y}(x_1, x_2, t)$ at $t = 0.4$ (*top left*), $t = 0.95$ (*top right*), $t = 1.4$ (*bottom left*) and $t = 2$ (*bottom right*).

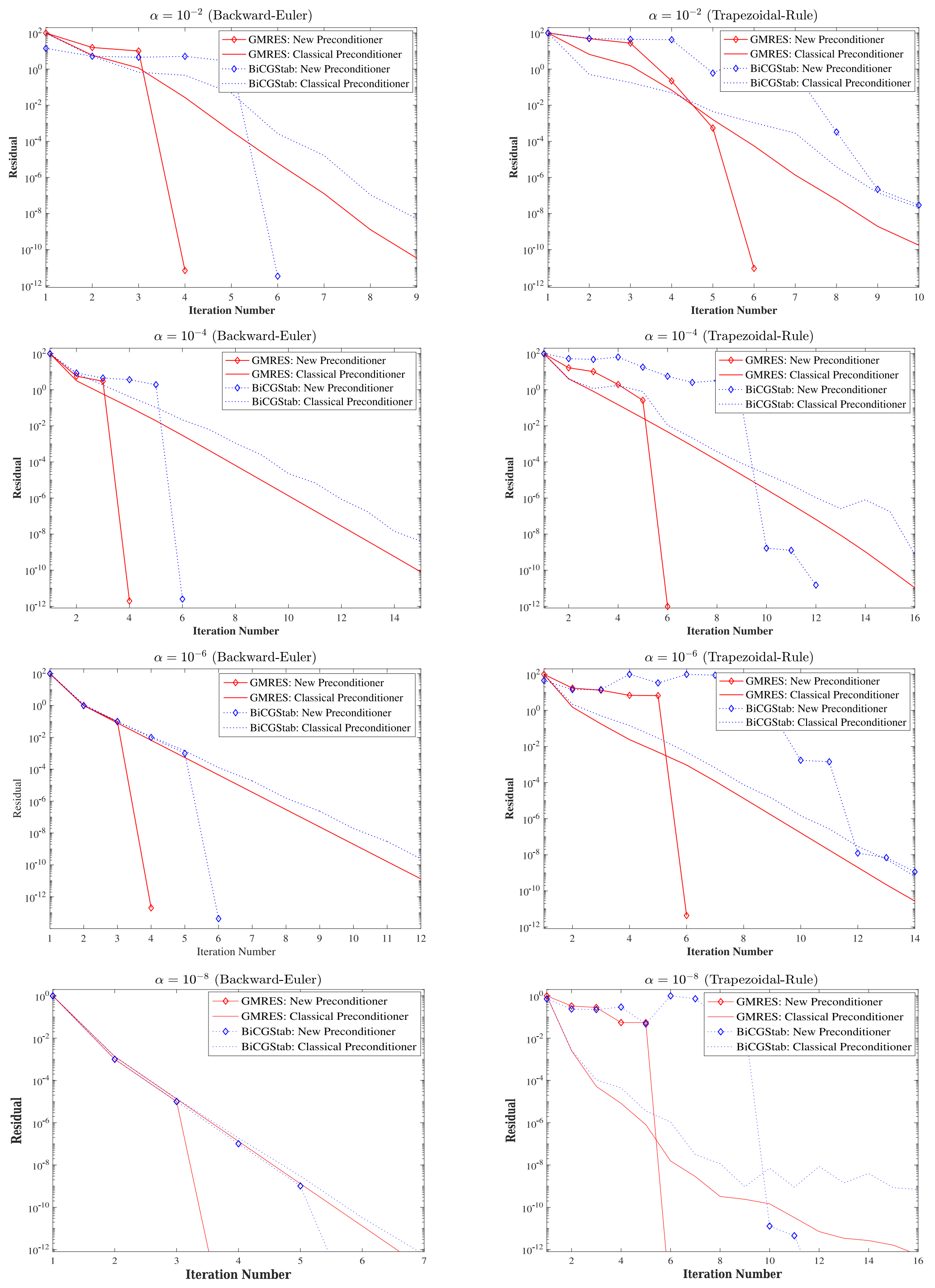


FIGURE 14. For $h = 0.01$, $N_t = 101$, the residuals of the GMRES and BiCGStab methods using the two preconditioners introduced in Section 3. *Left column*: using the Backward-Euler method as the time-integrator; *Right column*: similar results for trapezoidal rule. *From top to bottom*: $\alpha = 10^{-2}$, $\alpha = 10^{-4}$, $\alpha = 10^{-6}$ and $\alpha = 10^{-8}$.

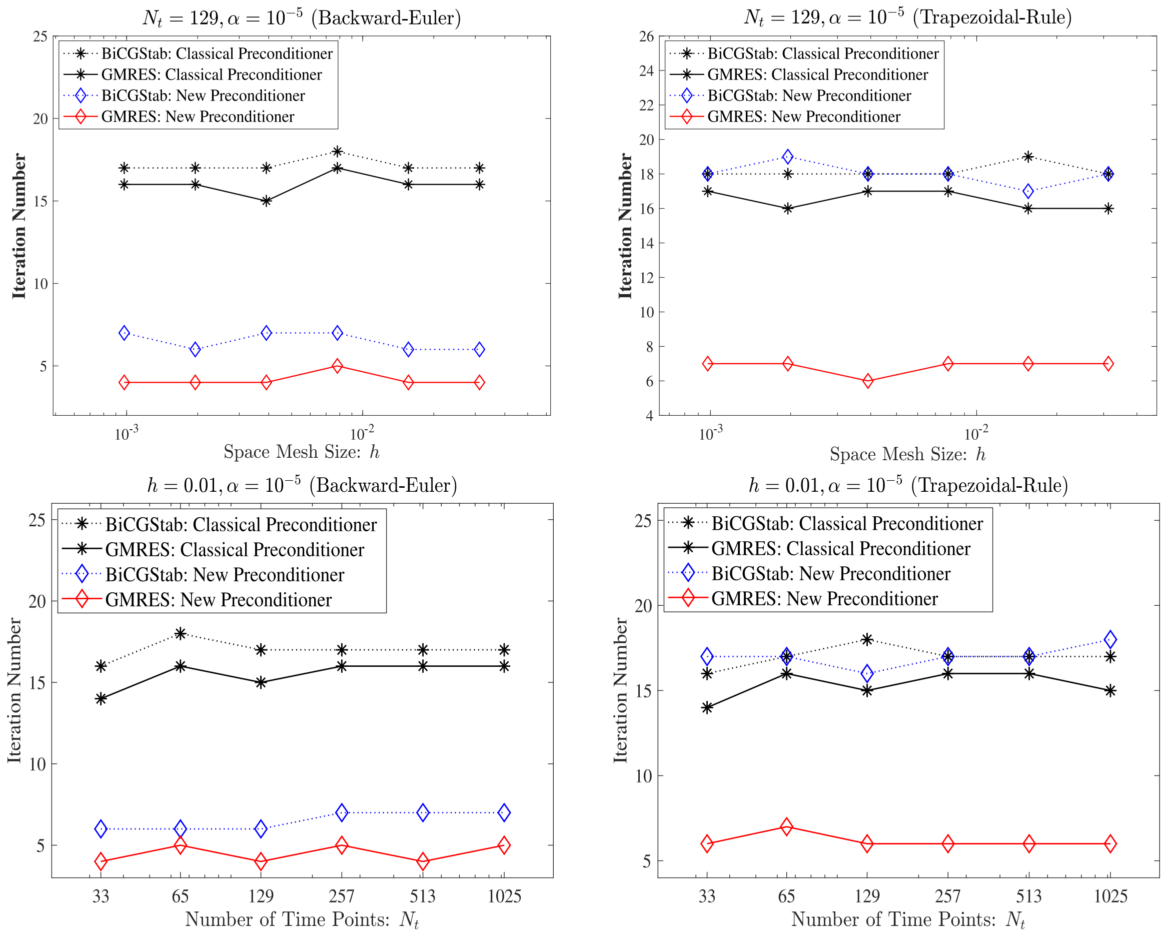


FIGURE 15. Iteration numbers of the GMRES and BiCGStab methods using the two preconditioners introduced in Section 3. *Top row*: $N_t = 129$ is fixed and h varies. *Bottom row*: $h = 0.01$ is fixed and N_t varies. *Left column*: Backward-Euler; *Right column*: trapezoidal rule.

Similar to Figure 9, we show in Figure 13 the filled contour plots of the computed state $y(x_1, x_2, t)$ and the desired state $\bar{y}(x_1, x_2, t)$ at $t = 0.4$, $t = 0.95$, $t = 1.4$ and $t = 2$. We see that in this case the computed state is also close to the desired state.

We next show the convergence rates of two Krylov subspace solvers, the GMRES method and the BiCGStab method, using the two preconditioners introduced in Section 3, *i.e.*, the classical one given by (3.4) (with the choice of $\widehat{\mathbf{S}}$ given by (3.6)) and the new one given by (3.10). The initial guess for these two Krylov solvers are zero and the iteration stops when the residual is less than 10^{-12}. In Figure 14, for fixed $h = 0.01$ and $N_t = 101$ and three values of the regularization parameter α we show the measured convergence rates of the two Krylov subspace solvers. When the Backward-Euler method is used as the time-integrator, we see that the new preconditioner results in *superlinear* convergence for the two Krylov subspace solvers. For the trapezoidal rule, the GMRES method using the new preconditioner still converges superlinearly, while the BiCGStab method does not. In particular, in this case it seems that the new preconditioner is worse than the classical preconditioner. For both the GMRES and BiCGStab methods, from Figure 14 we see that the convergence rates are robust with respect to the regularization parameter α.

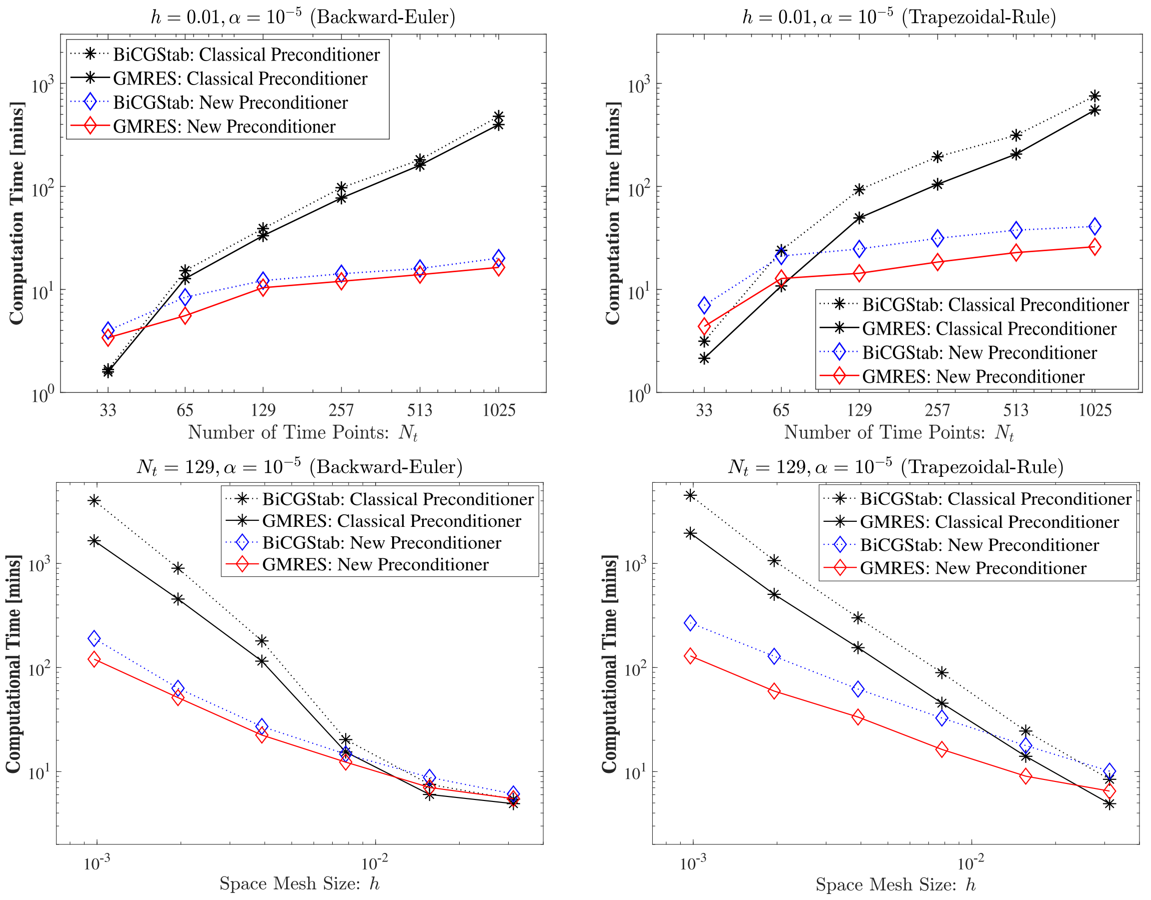


FIGURE 16. *Top row*: for $\alpha = 10^{-5}$, $h = 0.01$ and several values of N_t, the computation time (in minutes) of the the GMRES and BiCGStab methods using the two preconditioners introduced in Section 3. *Bottom row*: similar results for the case $N_t = 129$ and several values of h. *Left column*: Backward-Euler; *Right column*: trapezoidal rule.

It would be also interesting to study how the convergence rates of the two Krylov subspace solvers depend on the space mesh size h and the number of time points N_t. The results for this aspect are shown in Figure 15. We see that for the Backward-Euler method, the GMRES method (resp. the BiCGStab method) using the new preconditioner needs approximately 4∼5 iterations (resp. 6∼7 iterations) to reach a tolerance 10^{-12}, while these two Krylov subspace solvers using the classical preconditioner need approximately $15 \sim 17$ iterations. For the trapezoidal rule, as we already saw in Figure 14 on the bottom row, the convergence rates of the two Krylov subspace solvers differ obviously. In particular, the GMRES method using the new preconditioner needs approximately $6 \sim 7$ iterations, while for the classical preconditioner it needs $16 \sim 18$ iterations. The BiCGStab method using the two preconditioners has similar convergence rate and it needs $16 \sim 18$ iterations as well. Moreover, from Figure 15 we see that the convergence rates of the two Krylov subspace solvers are robust with respect to h and N_t.

At the end of this section, we compare the computation time (in minutes) of the Krylov subspace solvers using the two preconditioners. The classical preconditioner is used in the sequential-in-time pattern and the new one is used in the PinT pattern. The measured computation time is shown in Figure 16. It is clear that, thanks to the potential of PinT computation the new preconditioner significantly reduces the computation time, compared to the classical preconditioner. In particular, for the trapezoidal rule even though the new preconditioner does not ameliorate the convergence rate, but the former can be implemented in the PinT pattern and thus it still reduces the computation time.

5. Conclusions

Based on the diagonalization technique proposed recently [9, 24], we have studied PinT algorithms for the tracking-type PDE-constrained optimization problems with distributed control. For the time-periodic condition, the matrix $\mathbf{B}_{\rm per}$ representing the time-discretization can be diagonalized directly and this yields direct PinT computation for all the N_t time points. A direct diagonalization of $\mathbf{B}_{\rm per}$ results in unacceptable condition number of the eigenvector matrix and our main contribution is a novel modification, which dramatically reduces the condition number. For the initial-value condition, the matrix representing the time-discretization is $\mathbf{B}_{\rm ini}$, which can not be diagonalized with explicit formulas. The matrix $\mathbf{B}_{\rm ini}$ is a 2×2 block matrix and the two diagonal blocks are Toeplitz matrices. Motivated by a recent work [27] we approximate these two blocks by two suitable *circulant* matrices and this naturally gives a preconditioner for the discrete KKT system. The preconditioner has the same structure of the discrete KKT system in the time-periodic case and thus for Krylov subspace methods it can be used in an efficient PinT pattern *via* the diagonalization technique. Clustering of the eigenvalues and singular values of the preconditioned matrix is justified. For both the time-periodic case and the initial-value case, numerical results indicate that the PinT algorithms proposed here can significantly reduce the computation time. In particular, for the initial-value case, compared to the classical preconditioner popular in the literature the new preconditioner results in faster (and superlinear) convergence of the GMRES method. For the BiCGStab method, the convergence rate depends on the time-integrator: for the Backward-Euler method the new preconditioner results in faster (and superlinear) convergence than the classical preconditioner, while for the trapezoidal rule the latter is better than the former.

Optimal control for time-dependent problems often involves control constraints or have the terminal constraint. The mainstream method for solving this kind of problems is to apply the semi-smooth Newton iterations (or equivalently primal-dual active set method) together with the preconditioned Krylov subspace solvers (as inner loop). With some suitable approximation of the Schur complement of the KKT system at each semi-smooth Newton iteration, the preconditioner can be also implemented in a PinT pattern *via* the diagonalization technique studied in this paper. Details on this issue will be given in our forthcoming work.

Acknowledgements. The authors are very grateful to the anonymous referees for their careful reading of a preliminary version of the manuscript and their valuable suggestions, which greatly improved the quality of this paper. The work of the first author was supported by the NSF of China through grant 11771313, the NSF of Sichuan Province through grant 2018JY0469 and the Science Challenge Project (TZ2016002). The work of the second author was supported by the Science Challenge Project through grant TZ2018001, NSF of China through grants 11822111, 11688101, and 11731006, National Key Basic Research Program through grant 2018YFB0704304, and the Youth Innovation Promotion Association (CAS).

References

[1] D. Abbeloos, M. Diehl, M. Hinze and S. Vandewalle, Nested multigrid methods for time-periodic, parabolic optimal control problems. *Comput. Visual Sci.* **14** (2011) 27–38.

[2] O. Axelsson, S. Farouq and M. Neytcheva, A preconditioner for optimal control problems, constrained by Stokes equation with a time-harmonic control. *J. Comput. Appl. Math.* **310** (2017) 5–18.

[3] L. Banjai and D. Peterseim, Parallel multistep methods for linear evolution problems. *IMA J. Numer. Anal.* **32** (2012) 1217–1240.

[4] L.L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes and B. van Bloemen Waanders, Real-time PDE-constrained Optimization. *SIAM* (2007).

[5] A. Borzì and K. Kunisch, A multigrid method for optimal control of time-dependent reaction diffusion processes, Fast Solution of Discretized Optimization Problems. Edited by Hoffmann, R. Hoppe and V. Schulz Vol. 138 of *ISNM International Series of Numerical Mathematics*. Birkhäuser, Basel (2001).

[6] A. Borzì and V. Schulz, m Computational optimization of systems governed by partial differential equations. SIAM, Philadelphia, PA (2012).

[7] X. Du, M. Sarkis, C.E. Schaerer and D.B. Szyld, Inexact and truncated Parareal-in-time Krylov subspace methods for parabolic optimal control problems. *Electronic Trans. Numer. Anal.* **40** (2013) 36–57.

[8] M. Erhard and H. Strauch, Control of towing kites for seagoing vessels. *IEEE Trans. Control Syst. Technol.* **21** (2013) 1629–1640.

[9] M.J. Gander, L. Halpern, J. Rannou and J. Ryan, A direct time parallel solver by diagonalization for the wave equation. *SIAM J. Sci. Comput.* **41** (2019) A220–A245.

[10] M.J. Gander and S. Vandewalle, Analysis of the parareal time-parallel time-integration method. *SIAM J. Sci. Comput.* **29** (2007) 556–578.

[11] M.J. Gander and F. Kwok, Schwarz methods for the time-parallel solution of parabolic control problems. *Lect. Notes Comput. Sci. Eng.* **104** (2016) 207–216.

[12] S.G unther, N.R. Gauger and J.B. Schroder, A non-intrusive parallel-in-time approach for simultaneous optimization with unsteady PDEs. *Optim. Methods Softw.* **34** (2019) 1306–1321.

[13] W. Hackbusch, Fast numerical solution of time-periodic parabolic problems by a multigrid method. *SIAM J. Sci. Stat. Comput.* **2** (1981) 198–206.

[14] F.M. Hante, M.S. Mommer and A. Potschka, Newton–Picard preconditioners for time-periodic parabolic optimal control problems. *SIAM J. Numer. Anal.* **53** (2015) 2206–2225.

[15] B. Houska and M. Diehl, Robustness and stability optimization of power generating kite systems in a periodic pumping mode, in Vol. 58 of *Proceeding of IEEE International Conference on Control Applications (CCA)* (2010) 2172–2177.

[16] B. Houska, F. Logist, J.V. Impe and M. Diehl, Approximate robust optimization of time-periodic stationary states with application to biochemical processes, in *Proceedings of the 48th IEEE Conference on Decision and Control.* Shanghai, China (2009) 6280–6285.

[17] M. Kolmbauer, Efficient solvers for multiharmonic eddy current optimal control problems with various constraints and their analysis. *IMA J. Numer. Anal.* **33** (2013) 1063-1094.

[18] M. Kolmbauer and U. Langer, A robust preconditioned MinRes solver for distributed time-periodic eddy current optimal control problems. *SIAM J. Sci. Comput.* **34** (2012) B785–B809.

[19] M. Kollmann and M. Kolmbauer, A preconditioned MinRes solver for time-periodic parabolic optimal control problems. *Numer. Linear Algebra Appl.* **20** (2014) 761–784.

[20] U. Langer and M. Wolfmayr, Multiharmonic finite element analysis of a time-periodic parabolic optimal control problem. *J. Numer. Math.* **21** (2013) 265–300.

[21] Z. Liang, O. Axelsson and M. Neytcheva, A robust structured preconditioner for time-harmonic parabolic optimal control problems. *Numer. Algor.* **79** (2018) 575–596.

[22] J.L. Lions, Y. Maday and G. Turinici, A "parareal" in time discretization of PDE's. *C.R. Acad. Sci. Paris Sér. I Math.* **332** (2001) 661–668.

[23] J. Liu and Z. Wang, Efficient time domain decomposition algorithms for parabolic PDE-constrained optimization problems. *Comput. Math. Appl.* **75** (2018) 2115–2133.

[24] Y. Maday and E.M. Rønquist, Parallelization in time through tensor product space-time solvers. *C.R. Math. Acad. Sci. Paris* **346** (2008) 113–118.

[25] Y. Maday, M.K. Riahi and J. Salomon, Parareal in time intermediate targets methods for optimal control problems, in Control and Optimization with PDE Constraints. Birkhäuser, Basel (2013) 79–92.

[26] T.P. Mathew, M. Sarkis, and C.E. Schaerer, Analysis of block parareal preconditioners for parabolic optimal control problems. *SIAM J. Sci. Comput.* **32** (2010) 1180–1200.

[27] E. McDonald, J. Pestana and A. Wathen, Preconditioning and iterative solution of all-at-once systems for evolutionary partial differential equations. *SIAM J. Sci. Comput.* **40** (2018) A1012—A1033.

[28] P. Nikpoorparizi, N. Deodhar and C. Vermillion, Modeling, control design and combined plant/controller optimization for an energy-harvesting tethered wing. *IEEE Trans. Control Syst. Technol.* **99** (2017) 1–13.

[29] M.F. Murphy, G.H. Golub and A.J. Wathen, A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.* **21** (2000) 1969–1972.

[30] J.W. Pearson, M. Stoll and A.J. Wathen, Regularization-robust preconditioners for time-dependent PDE-constrained optimization problems. *SIAM J. Matrix Anal. Appl.* **33** (2012) 1126–1152.

[31] A. Potschka, M.S. Mommer, J.P. Der Schl and H.G. Bock, Newton-picard-based preconditioning for linear-quadratic optimization problems with time-periodic parabolic pde constraints. *SIAM J. Sci. Comput.* **34** (2010) 1214–1239.

[32] M. Stoll, One-shot solution of a time-dependent time-periodic PDE-constrained optimization problem. *IMA J. Numer. Anal.* **34** (2014) 1554–1577.

[33] A. Toumi, S. Engell, M. Diehl, H. Bock and J. Schlöder, Efficient optimization of simulated moving bed processes. *Chem. Eng. Process.* **46** (2007) 1067–1084.

[34] S.L. Wu, Toward parallel coarse grid correction for the parareal algorithm. *SIAM J. Sci. Comput.* **40** (2018) A1446–A1472.

[35] S.L. Wu, H. Zhang and T. Zhou, Solving time-periodic fractional diffusion equations *via* diagonalization technique and multigrid. *Numer. Linear Algebra Appl.* **25** (2018) e2178.

[36] A.U. Zgraggen, L. Fagiano and M. Morari, Automatic retraction and full-cycle operation for a class of airborne wind energy generators. *IEEE Trans. Control Syst. Technol.* **24** (2016) 594–608.