https://doi.org/10.1016/j.cam.2017.06.008

A New Multigrid Method for Unconstrained Parabolic Optimal Control Problems[☆]

Buyang Li[a], Jun Liu[b], Mingqing Xiao[c,*]

[a] *Department of Applied Mathematics, The Hong Kong Polytechnic University, Kowloon, Hong Kong.*
[b] *Department of Mathematics and Statistical Sciences, Jackson State University, Jackson, MS 39217, USA.*
[c] *Department of Mathematics, Southern Illinois University, Carbondale, IL 62901-4408, USA*

Abstract

A second-order leapfrog finite difference scheme in time is proposed and developed for solving the first-order necessary optimality system of the distributed parabolic optimal control problems. Different from available approaches, the proposed leapfrog scheme for the two-point boundary optimality system is shown to be unconditionally stable and provides a second-order accuracy, though the classical leapfrog scheme usually is unstable. Moreover the proposed leapfrog scheme provides a feasible structure that leads to an effective implementation of a fast solver under the multigrid framework. A detailed mathematical proof for the stability of the proposed scheme is provided in terms of a new norm that is more suitable and stronger to characterize the convergence than the L^2 norm often used in literature. Numerical experiments show that the proposed scheme significantly outperforms the widely used second-order backward time differentiation approach and the resultant fast solver demonstrates a mesh-independent convergence as well as a linear time complexity.

Keywords: Parabolic optimal control, Leapfrog scheme, Finite difference, Multigrid method

1. Introduction

Optimal control problems governed by evolutionary partial differential equations (PDEs) [1, 2, 3, 4] have recently gained dramatically increasing attention from the scientific computing community. This trend is driven not only by its broader applications in different fields but also by the computational challenges it brings on the stage. Among those applications, one of representative examples is the real-time optimal control [5] of reaction-diffusion systems in cardiac electrophysiology [6]. The performance of these systems usually relies on the efficiency and accuracy of numerical algorithms in order to achieve desirable control response, which requires substantial efforts across related disciplines, in particular, from the community of numerical optimization.

We exemplify our proposed approach through discussing a prototype parabolic distributed optimal control problem. Let $\Omega = (0,1)^d$ $(1 \leq d \leq 3)$ be the spatial domain with boundary $\Gamma := \partial\Omega$. Given a finite period of time $T > 0$, define $Q = \Omega \times (0,T)$ and $\Sigma = \Gamma \times (0,T)$. We consider the following optimal control problem [1, 3] of minimizing a tracking-type quadratic cost functional

$$J(y,u) = \tfrac{1}{2}\|y-g\|^2_{L^2(Q)} + \tfrac{\gamma}{2}\|u\|^2_{L^2(Q)} \tag{1.1}$$

subject to a linear parabolic PDE system

$$\begin{cases} -\partial_t y + \Delta y = & f + u \ \text{ in } \ Q, \\ y = & 0 \text{ on } \ \Sigma, \\ y(\cdot,0) = & y_0 \text{ in } \ \Omega, \end{cases} \tag{1.2}$$

[☆] This project was supported in part by NSFC (Grant No. 11301262) and in part by NSF 1021203, 1419028 of the United States.

[*] Corresponding author
Email addresses: buyang.li@polyu.edu.hk (Buyang Li), jun.liu@jsums.edu (Jun Liu), mxiao@siu.edu (Mingqing Xiao)

where $u \in U := L^2(Q)$ is the distributed control function, $g \in L^2(Q)$ is the desired tracking trajectory, $\gamma > 0$ represents either the weight of the cost of control or the Tikhonov regularization parameter, $f \in L^2(Q)$, and the initial condition $y_0 \in L^2(\Omega)$. The existence and uniqueness of solution to the above optimal control problem (1.1-1.2) is well understood (see, for example, [1, 3]).

Current state-of-art numerical methods [7, 8, 4] for solving this class of optimal control problems (1.1–1.2) generally fall into either *discretize-then-optimize* approach or *optimize-then-discretize* approach. In this paper, we will focus on the second category by making use of its first-order optimality system. However, our proposed approach is also suitable for the first category when the discretization has a similar structure. More specifically, our proposed discretization scheme and multigrid solver are numerically shown to work very well under the framework of *discretize-then-optimize* approach.

By defining an appropriate Lagrange functional, making use of the strong convexity of the original optimization problem, the optimal solution pair (y, u) to (1.1-1.2) is shown to be completely characterized by the unique solution triplet (y, p, u) to the following optimality system

$$\left\{\begin{array}{rll} -\partial_t y + \Delta y - u = & f & \text{in } Q, \\ y = & 0 & \text{on } \Sigma, \quad y(\cdot, 0) = y_0 \text{ in } \ \Omega, \\ \partial_t p + \Delta p + y = & g & \text{in } Q, \\ p = & 0 & \text{on } \Sigma, \quad p(\cdot, T) = 0 \text{ in } \ \Omega, \\ \gamma u - p = & 0 & \text{in } Q, \end{array}\right. \tag{1.3}$$

where the state y evolves forward in time and the adjoint state p marches backward in time. According to [9], suitable regularity for y and p can hold under appropriate assumptions on y_0, f, and g. The special relation $\gamma u - p = 0$ implies that u has the same regularity as p. For the purpose of simplified analysis and practical implementation, the control $u = p/\gamma$ can be eliminated from the optimality system as following

$$\left\{\begin{array}{rll} -\partial_t y + \Delta y - p/\gamma = & f & \text{in } Q, \\ y = & 0 & \text{on } \Sigma, \quad y(\cdot, 0) = y_0 \text{ in } \ \Omega, \\ \partial_t p + \Delta p + y = & g & \text{in } Q, \\ p = & 0 & \text{on } \Sigma, \quad p(\cdot, T) = 0 \text{ in } \ \Omega. \end{array}\right. \tag{1.4}$$

This is a standard two-point boundary-value problem (with respect to t) that appears often in optimal control of parabolic PDEs. The main challenge for solving (1.4) results from the fact that the state y and the adjoint state p are marching in opposite directions. Its numerical discretizations will create a large-scale sparse system of algebraic equations as we have to resolve all time steps simultaneously [10].

In terms of finite difference discretization for time variable of (1.4), the backward Euler discretization with respect to time t is favorable choice due to its unconditional stability (see, e.g.,[11]). The drawback is the sloppy first-order accuracy in time t, compared with standard second-order spatial discretizations. Constructing higher order finite difference schemes for the time variable t is a natural development to improve the overall efficiency (for both time and spatial variables) since it allows us to attain the required accuracy with much coarser mesh size that results in a smaller dimension of discretized linear system. Thus much of efforts are devoted by many researchers to explore various second or higher-order numerical schemes for (1.4) or similar-type systems. In [12], the authors introduced a family of second-order Crank–Nicolson (CN) based time discretizations for state and adjoint state in unconstrained optimal control problems with evolution equations, where a second-order accuracy in both time and space is proved under L^2 norm setting. In [13], the authors developed a second-order backward time differentiation formula (BDF2) in time with CN scheme as an initialization step, which is also shown to be second-order accurate with discrete L^2 norm in the case where the constraints on the control are not active. Their BDF2 scheme requires a second-order accurate approximation, such as the CN scheme, to the initial time step of the state equation as well as the final time step of the adjoint equation, respectively. Under the framework of finite element discretizations, similar efforts were also made to develop better convergent schemes. For instance, it was demonstrated in [14, 15] under suitable conditions the optimality system is actually equivalent to a V-elliptic problem on the space-time cylinder that leads to some rigorous error estimates [16, 17, 18, 19]. In addition to above mentioned schemes, many other discretization strategies in time and space have been extensively studied [20, 21].

Although several second-order schemes are available, they are not necessarily suitable for fast solver development due to the complexity of discretization structures. For example, as the authors pointed out in [13, 22], the pure CN scheme is not a good choice for implementing a space-time multigrid algorithm due to the lack of certain symmetric structures of discretization. In fact, past numerical experiences show that some standard multigrid solvers, including the one we present in this paper, may not even converge with CN scheme, and thus this stimulated us to seek more suitable schemes that can successfully facilitate the multigrid solver development. Moreover, in order to improve the overall efficiency it is important and necessary to equip a discretization scheme with some fast direct/iterative linear solvers [23, 24, 25, 26] so that it can deal with large-scale degrees of freedom and higher dimension efficiently. Beginning with a few early numerical endeavors [27, 28, 29], multigrid methods have started to play a more and more irreplaceable character in the field of PDE optimization [30, 31, 32, 4, 33, 34] since the seminal introduction of space-time multigrid for linear parabolic PDEs [35], where the semi-coarsening was shown to give better convergence compared to standard coarsening. In the framework of finite difference discretization, some recent papers [11, 36, 37, 38, 39, 40, 41, 13] are devoted to apply the idea of space-time multigrid to those forward-and-backward coupled linear/nonlinear parabolic PDE systems similar to (1.4). But few research is seen between the connection of numerical scheme design and fast solver implementation in current literature.

In this paper we propose the leapfrog central difference scheme for time discretization. In classical theory, it is well-known that leapfrog scheme is not stable for a single evolutionary equation although it is second-order accurate [42, 43]. However, in this paper, we prove that leapfrog scheme in terms of time discretization for the two point boundary-value problem (1.4) is unconditionally stable and delivers the second-order accuracy of time variable, which has not been seen in current literature. The essential observation is that the conventional instability of leapfrog scheme comes from errors propagation in each time step with an amplification factor being strictly greater than one. In contrast, our approach solves for all time steps in one shot by treating time as a new spatial variable, which will not amplify the temporal errors as there are no explicit time-iteration operations. More importantly, our approach by using leapfrog scheme leads to the implementation of a very efficient multigrid iterative solver. More specifically, this scheme provides a feasible and practical approach in constructing the effective collective Jacobi smoother [39], as it was shown in [44, 45] for the case of elliptic optimal control problems by using finite element discretization. This advantage will become even more valuable when handling the problems with nonlinear parabolic PDEs associated with higher dimensional domains. From the viewpoint of numerical development, it is important for us to discretize the system in an 'optimal' way so that it not only achieves the desired order of accuracy but also is able to accommodate the later effective implementation of a fast solver with a predictable guaranteed converging property.

This paper is organized as follows. In the next section, we propose leapfrog scheme (with a backward Euler step) in time and a second-order five-point finite difference scheme in space for discretizing the optimality system. The second-order accuracy of our proposed leapfrog scheme is proved under the discrete $L^2(L^\infty)$ norm. In Section 3, a multigrid algorithm is designed for solving the discretized optimality system with some favorable structures. In Section 4, results of numerical simulations are reported to demonstrate the second-order accuracy of our leapfrog finite difference approximations and the mesh independent convergence of the corresponding multigrid solver with linear time complexity. Numerical comparisons are performed among the BDF2 scheme, the CN scheme, and our leapfrog scheme. Finally, the paper ends with concluding remarks in Section 5.

2. Leapfrog scheme and its error estimate

In this section, we exemplify our analysis in the two dimensional case, the conclusions of which can be easily generalized to one and three dimensions. We partition the time interval $[0, T]$ uniformly into $0 = t_0 < t_1 < \cdots < t_N = T$ with $t_k - t_{k-1} = \tau = T/N$, and discretize the space domain $\Omega = [0, 1]^2$ uniformly into $0 = \xi_0 < \xi_1 < \cdots < \xi_{M_1} = 1$ and $0 = \zeta_0 < \zeta_1 < \cdots < \zeta_{M_2} = 1$, with $h_1 = \xi_i - \xi_{i-1}$, $h_2 = \zeta_j - \zeta_{j-1}$. Let $h = \max(h_1, h_2)$. For a given $0 \le n \le N$, we define the discrete inner product $(\varphi^n, \phi^n) = \sum_{i,j=1}^{M_1-1, M_2-1} \varphi_{ij}^n \phi_{ij}^n h_1 h_2$ and the corresponding discrete $L^2(\Omega)$ norm $\|\phi^n\| = \sqrt{(\phi^n, \phi^n)}$. We also define the discrete gradient

$$\nabla_h \varphi^n = \left(\frac{\varphi_{i,j}^n - \varphi_{i-1,j}^n}{h_1}, \frac{\varphi_{i,j}^n - \varphi_{i,j-1}^n}{h_2} \right)_{i=1,j=1}^{M_1,M_2},$$

and the discrete Laplacian

$$(\Delta_h Y^n)_{ij} = \frac{Y^n_{i-1,j} - 2Y^n_{i,j} + Y^n_{i+1,j}}{h_1^2} + \frac{Y^n_{i,j-1} - 2Y^n_{i,j} + Y^n_{i,j+1}}{h_2^2} .$$

We shall use the discrete version of Poincaré inequality and Sobolev embedding inequality [46, 47], i.e. there exists a positive constant C_0, independent of h, such that if $y = (y_{ij})$ satisfies the boundary condition $y_{0,j} = y_{M_1,j} = y_{i,0} = y_{i,M_2} = 0$ for $i = 1, \cdots, M_1 - 1$ and $j = 1, \cdots, M_2 - 1$, then

$$\|y\| \le C_0 \|\nabla_h y\| \qquad \text{and} \qquad \max_{i,j} |y^n_{ij}| \le C_0 \|\Delta_h y^n\|.$$

We shall also use the discrete version of integration by parts:

$$(-\Delta_h z, w) = (\nabla_h z, \nabla_h w)$$

when functions z, w are defined on the mesh points and vanish on the boundary $\partial\Omega$.

We discretize the equations (1.4) by the leap-frog finite difference scheme

$$\frac{Y^{n+1} - Y^{n-1}}{2\tau} - \Delta_h Y^n + P^n/\gamma = -f^n, \quad n = 1, 2, \cdots, N-1 \tag{2.5}$$

$$\frac{P^{n+1} - P^{n-1}}{2\tau} + \Delta_h P^n + Y^n = g^n, \quad n = 1, 2, \cdots, N-1 \tag{2.6}$$

where $Y^n = (Y^n_{ij})_{i=1,j=1}^{M_1-1,M_2-1}$ and $P^n = (P^n_{ij})_{i=1,j=1}^{M_1-1,M_2-1}$ with Y^n_{ij} and P^n_{ij} being the discrete approximation of $y(\xi_i, \zeta_j, t_n)$ and $p(\xi_i, \zeta_j, t_n)$, respectively. Similar notations are used for f^n and g^n. Here Y^0 and P^N are from given initial conditions. At the last time step, we close the linear system by imposing two additional equations by using the backward Euler scheme

$$\frac{Y^N - Y^{N-1}}{\tau} - \Delta_h Y^N + P^N/\gamma = -f^N, \tag{2.7}$$

$$\frac{P^1 - P^0}{\tau} + \Delta_h P^0 + Y^0 = g^0. \tag{2.8}$$

Such a treatment is significantly different from the traditional unstable leapfrog scheme which often uses a backward Euler step for initializing the temporal advancing. Although we only use a first-order backward Euler scheme in the final time step, we shall see that the finite difference approximations $\{Y^n, P^n\}_{n=0}^N$ has a second-order accuracy in both time and space as shown in the following theorem. This extra flexibility of our leapfrog scheme compared to the BDF2 scheme comes from our following more direct proof arguments. In practical implementations, those second-order accurate BDF2 or CN schemes are also applicable to replace the above first-order accurate backward Euler scheme for better accuracy, which is illustrated by the following Example 2 in Section 4,

Let $C^{4,3}(\overline{Q})$ denote the space of functions with bounded continuous spatial partial derivatives up to order 4 and bounded continuous temporal partial derivatives up to order 3, i.e.,

$$C^{4,3}(\overline{Q}) = \{w : \partial^{4-k}_{x_1} \partial^k_{x_2} w \text{ and } \partial^3_t w \text{ are bounded and continuous, } k = 0, 1, 2, 3, 4\}.$$

Theorem 2.1. *Let the dimension $d = 2$. Assuming that $y, p \in C^{4,3}(\overline{Q})$, the linear system defined by* (2.5)-(2.8) *is invertible and the scheme has second-order accuracy in discrete $L^2(L^\infty)$ norm, i.e.,*

$$\left(\|e\|^2_{L^2_\tau(L^\infty_h)} + \|\eta\|^2_{L^2_\tau(L^\infty_h)} \right)^{\frac{1}{2}} := \left(\sum_{n=0}^{N} (\max_{i,j} |e^n_{ij}|^2 + \max_{i,j} |\eta^n_{ij}|^2)\tau \right)^{\frac{1}{2}} \le C(\tau^2 + h^2)$$

for some positive constant C which does not depend on τ and h, where $e^n_{i,j} = Y^n_{i,j} - y(\xi_i, \zeta_j, t_n)$ and $\eta^n_{i,j} = P^n_{i,j} - p(\xi_i, \zeta_j, t_n)$.

Proof. Note that the exact solutions $y^n(x) = y(x, t_n)$ and $p^n(x) = p(x, t_n)$ satisfy the equations

$$\frac{y^{n+1} - y^{n-1}}{2\tau} - \Delta_h y^n + p^n/\gamma = -f^n - F^n, \quad n = 1, 2, \cdots, N-1 \tag{2.9}$$

$$\frac{p^{n+1} - p^{n-1}}{2\tau} + \Delta_h p^n + y^n = g^n - G^n, \quad n = 1, 2, \cdots, N-1 \tag{2.10}$$

and

$$\frac{y^N - y^{N-1}}{\tau} - \Delta_h y^N + p^N/\gamma = -f^N - F^N, \tag{2.11}$$

$$\frac{p^1 - p^0}{\tau} + \Delta_h p^0 + y^0 = g^0 - G^0, \tag{2.12}$$

where F^n and G^n denote the truncation errors [42, 43], which satisfy (by assuming $y, p \in C^{4,3}(\overline{Q})$)

$$\|F^n\| + \|G^n\| \le C_1(\tau^2 + h^2) \quad \text{for} \quad n = 1, 2, \cdots, N-1$$

and

$$\begin{aligned}
F_{ij}^N &= \partial_t y(\xi_i, \zeta_j, t_N) - \frac{y(\xi_i, \zeta_j, t_N) - y(\xi_i, \zeta_j, t_{N-1})}{\tau} - (\Delta y(\xi_i, \zeta_j, t_N) - (\Delta_h y^N)_{ij}) \\
&= \left(\frac{1}{\tau} \int_{t_{N-1}}^{t_N} \int_s^{t_N} \partial_{tt} y(\xi_i, \zeta_j, s') \mathrm{d}s' \mathrm{d}s \right) - (\Delta y(\xi_i, \zeta_j, t_N) - (\Delta_h y^N)_{ij}) =: \overline{F}_{ij}^N - \widetilde{F}_{ij}^N, \\
G_{ij}^0 &= \partial_t p(\xi_i, \zeta_j, 0) - \frac{p(\xi_i, \zeta_j, \tau) - y(\xi_i, \zeta_j, 0)}{\tau} + (\Delta p(\xi_i, \zeta_j, 0) - (\Delta_h p^0)_{ij}) \\
&= \left(\frac{1}{\tau} \int_0^{\tau} \int_s^{\tau} \partial_{tt} p(\xi_i, \zeta_j, s') \mathrm{d}s' \mathrm{d}s \right) + (\Delta p(\xi_i, \zeta_j, 0) - (\Delta_h p^0)_{ij}) =: \overline{G}_{ij}^0 + \widetilde{G}_{ij}^0,
\end{aligned}$$

where

$$\begin{aligned}
&\|\nabla_h \overline{F}^N\| + \|\nabla_h \overline{G}^0\| \le C_1 \tau, \\
&\|\widetilde{F}^N\| + \|\widetilde{G}^0\| \le C_1 h^2,
\end{aligned}$$

for some positive constant C_1, independent of τ and h. Here we define $\overline{F}_{ij}^N$ and $\overline{G}_{ij}^0$ to be zero on the boundary ($i \in \{0, M_1\}$ or $j \in \{0, M_2\}$) so that their discrete gradients are well-defined. The two terms $\|\nabla_h \overline{F}^N\|$ and $\|\nabla_h \overline{G}^0\|$ above are estimated in the following way. Let

$$\nabla_h \partial_{tt} y = \left(\frac{\partial_{tt} y(\xi_i, \zeta_j, s') - \partial_{tt} y(\xi_{i-1}, \zeta_j, s')}{h_1}, \frac{\partial_{tt} y(\xi_i, \zeta_j, s') - \partial_{tt} y(\xi_i, \zeta_{j-1}, s')}{h_2} \right).$$

Then $\|\nabla_h \partial_{tt} y\| \le C \|\nabla \partial_{tt} y\|_{C(\overline{\Omega})} \le C$, due to the regularity assumption $y \in C^{4,3}(\overline{Q})$. Consequently, we have

$$\begin{aligned}
\|\nabla_h \overline{F}^N\| &= \left\| \frac{1}{\tau} \int_{t_{N-1}}^{t_N} \int_s^{t_N} \nabla_h \partial_{tt} y \, ds' ds \right\| \\
&\le \frac{1}{\tau} \int_{t_{N-1}}^{t_N} \int_s^{t_N} \|\nabla_h \partial_{tt} y\| \, ds' ds \\
&\le \frac{1}{\tau} \int_{t_{N-1}}^{t_N} \int_s^{t_N} C \|\nabla \partial_{tt} y\|_{C(\overline{\Omega})} \, ds' ds \\
&\le C \|\nabla \partial_{tt} y\|_{C(\overline{\Omega})} \tau.
\end{aligned}$$

$\|\nabla_h \overline{G}^0\|$ can be estimated similarly.

Let $e^n = Y^n - y^n$ and $\eta^n = P^n - p^n$. Then the difference between (2.5)-(2.8) and (2.9)-(2.12) gives

$$\frac{e^{n+1} - e^{n-1}}{2\tau} - \Delta_h e^n + \eta^n/\gamma = F^n, \tag{2.13}$$

$$\frac{\eta^{n+1} - \eta^{n-1}}{2\tau} + \Delta_h \eta^n + e^n = G^n, \tag{2.14}$$

and

$$\frac{e^N - e^{N-1}}{\tau} - \Delta_h e^N + \eta^N/\gamma = F^N, \tag{2.15}$$

$$\frac{\eta^1 - \eta^0}{\tau} + \Delta_h \eta^0 + e^0 = G^0, \tag{2.16}$$

with the initial conditions $e^0 = \eta^N = 0$.

The discrete inner product of (2.13) and $-\tau\,\Delta_h e^n$ is

$$\frac{(\nabla_h e^{n+1}, \nabla_h e^n) - (\nabla_h e^n, \nabla_h e^{n-1})}{2} + \tau\|\Delta_h e^n\|^2 + \tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma = -\tau(F^n, \Delta_h e^n), \tag{2.17}$$

and by summing up above equations for $n = 1, \cdots, N-1$, we get (note that $(\nabla_h e^1, \nabla_h e^0) = 0$)

$$\frac{(\nabla_h e^N, \nabla_h e^{N-1})}{2} + \sum_{n=1}^{N-1} \tau\|\Delta_h e^n\|^2 + \sum_{n=1}^{N-1} \tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma = -\sum_{n=1}^{N-1} \tau(F^n, \Delta_h e^n). \tag{2.18}$$

The discrete inner product of (2.15) and $-\tau\,\Delta_h e^N/2$ is (note that $\eta^N = 0$)

$$\begin{aligned}\frac{\|\nabla_h e^N\|^2 - (\nabla_h e^N, \nabla_h e^{N-1})}{2} + \frac{\tau}{2}\|\Delta_h e^N\|^2 &= -\frac{\tau}{2}(F^N, \Delta_h e^N) \\ &= \frac{\tau}{2}(\nabla_h \overline{F}^N, \nabla_h e^N) + \frac{\tau}{2}(\widetilde{F}^N, \Delta_h e^N).\end{aligned} \tag{2.19}$$

The sum of the last two equations gives
(by Cauchy's inequality with ϵ [9]: $ab \le a^2/(4\epsilon) + \epsilon b^2$ for $a > 0, b > 0, \epsilon > 0$)

$$\begin{aligned}&\frac{\|\nabla_h e^N\|^2}{2} + \sum_{n=1}^{N-1} \tau\|\Delta_h e^n\|^2 + \frac{\tau}{2}\|\Delta_h e^N\|^2 + \sum_{n=1}^{N-1} \tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma \\ &= -\sum_{n=1}^{N-1} \tau(F^n, \Delta_h e^n) + \frac{\tau}{2}(\nabla_h \overline{F}^N, \nabla_h e^N) + \frac{\tau}{2}(\widetilde{F}^N, \Delta_h e^N) \\ &\le \sum_{n=1}^{N-1} \tau\|F^n\|\|\Delta_h e^n\| + \frac{\tau}{2}(\|\nabla_h \overline{F}^N\|\|\nabla_h e^N\| + \|\widetilde{F}^N\|\|\Delta_h e^N\|) \\ &\le C_1 \sum_{n=1}^{N-1} \tau(\tau^2 + h^2)\|\Delta_h e^n\| + \frac{C_1}{2}\tau(\tau\|\nabla_h e^N\| + h^2\|\Delta_h e^N\|) \\ &\le C_1^2(\tau^2 + h^2)^2/(4\epsilon) \sum_{n=1}^{N-1} \tau + \epsilon \sum_{n=1}^{N-1} \tau\|\Delta_h e^n\|^2 + \frac{C_1^2}{4}(\tau^4 + \tau h^4)/(4\epsilon) + \epsilon\|\nabla_h e^N\|^2 + \epsilon\tau\|\Delta_h e^N\|^2 \\ &\le 2TC_1^2(\tau^4 + h^4)/(4\epsilon) + \epsilon \sum_{n=1}^{N-1} \tau\|\Delta_h e^n\|^2 + \frac{C_1^2}{4}(\tau^4 + \tau h^4)/(4\epsilon) + \epsilon\|\nabla_h e^N\|^2 + \epsilon\tau\|\Delta_h e^N\|^2 \\ &= C_1^2((T + \frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4)/(2\epsilon) + \epsilon \sum_{n=1}^{N-1} \tau\|\Delta_h e^n\|^2 + \epsilon\|\nabla_h e^N\|^2 + \epsilon\tau\|\Delta_h e^N\|^2,\end{aligned}$$

that is

$$(\frac{1}{2}-\epsilon)\|\nabla_h e^N\|^2 + (1-\epsilon)\sum_{n=1}^{N-1}\tau\|\Delta_h e^n\|^2 + (\frac{1}{2}-\epsilon)\tau\|\Delta_h e^N\|^2 + \sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma \tag{2.20}$$

$$\le C_1^2((T+\frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4)/(2\epsilon) \tag{2.21}$$

for arbitrary $\epsilon > 0$. By choosing $\epsilon = 1/4$ so that $(1-\epsilon) \ge 1/4$ and $(1/2-\epsilon) \ge 1/4$, the last inequality is reduced to

$$\begin{aligned}&\frac{1}{4}\|\nabla_h e^N\|^2 + \frac{1}{4}\sum_{n=1}^{N-1}\tau\|\Delta_h e^n\|^2 + \frac{1}{4}\tau\|\Delta_h e^N\|^2 + \frac{1}{\gamma}\sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h \eta^n)\\ &\le 2C_1^2((T+\frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4),\end{aligned}$$

which implies (by dropping $\frac{1}{4}\|\nabla_h e^N\|^2$ and absorbing the higher-order term τh^4 into C_2)

$$\sum_{n=1}^{N}\tau\|\Delta_h e^n\|^2 + \frac{4}{\gamma}\sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h \eta^n) \le C_2(\tau^4 + h^4). \tag{2.22}$$

Following above analogous arguments, the discrete inner product of (2.14) and $\tau\,\Delta_h \eta^n$ is

$$-\frac{(\nabla_h \eta^{n+1}, \nabla_h \eta^n) - (\nabla_h \eta^{n-1}, \nabla_h \eta^n)}{2} + \tau\|\Delta_h \eta^n\|^2 - \tau(\nabla_h e^n, \nabla_h \eta^n) = \tau(G^n, \Delta_h \eta^n), \tag{2.23}$$

and by summing up above equations for $n = 1, \cdots, N-1$, we get (note that $(\nabla_h \eta^N, \nabla_h \eta^{N-1}) = 0$)

$$\frac{(\nabla_h \eta^0, \nabla_h \eta^1)}{2} + \sum_{n=1}^{N-1}\tau\|\Delta_h \eta^n\|^2 - \sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h \eta^n) = \sum_{n=1}^{N-1}\tau(G^n, \Delta_h \eta^n). \tag{2.24}$$

The discrete inner product of (2.16) and $\tau\,\Delta_h \eta^0/2$ is (note that $e^0 = 0$)

$$\begin{aligned}-\frac{(\nabla_h \eta^1, \nabla_h \eta^0) - (\nabla_h \eta^0, \nabla_h \eta^0)}{2} + \frac{\tau}{2}\|\Delta_h \eta^0\|^2 &= \frac{\tau}{2}(G^0, \Delta_h \eta^0)\\ &= -\frac{\tau}{2}(\nabla_h \overline{G}^0, \nabla_h \eta^0) + \frac{\tau}{2}(\widetilde{G}^0, \Delta_h \eta^0).\end{aligned} \tag{2.25}$$

Similarly, the sum of the last two equations gives (by Cauchy's inequality with ϵ)

$$\begin{aligned}&\frac{\|\nabla_h \eta^0\|^2}{2} + \frac{\tau}{2}\|\Delta_h \eta^0\|^2 + \sum_{n=1}^{N-1}\tau\|\Delta_h \eta^n\|^2 - \sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h \eta^n)\\ &= -\frac{\tau}{2}(\nabla_h \overline{G}^0, \nabla_h \eta^0) + \frac{\tau}{2}(\widetilde{G}^0, \Delta_h \eta^0) + \sum_{n=1}^{N-1}\tau(G^n, \Delta_h \eta^n)\\ &\le \frac{\tau}{2}(\|\nabla_h \overline{G}^0\|\|\nabla_h \eta^0\| + \|\widetilde{G}^0\|\|\Delta_h \eta^0\|) + \sum_{n=1}^{N-1}\tau\|G^n\|\|\Delta_h \eta^n\|\\ &\le \frac{C_1}{2}\tau(\tau\|\nabla_h \eta^0\| + h^2\|\Delta_h \eta^0\|) + C_1\sum_{n=1}^{N-1}\tau(\tau^2 + h^2)\|\Delta_h \eta^n\|\\ &\le \frac{C_1^2}{4}(\tau^4 + \tau h^4)/(4\epsilon) + \epsilon\|\nabla_h \eta^0\|^2 + \epsilon\tau\|\Delta_h \eta^0\|^2 + C_1^2(\tau^2 + h^2)^2/(4\epsilon)\sum_{n=1}^{N-1}\tau + \epsilon\sum_{n=1}^{N-1}\tau\|\Delta_h \eta^n\|^2\\ &\le \frac{C_1^2}{4}(\tau^4 + \tau h^4)/(4\epsilon) + \epsilon\|\nabla_h \eta^0\|^2 + \epsilon\tau\|\Delta_h \eta^0\|^2 + 2TC_1^2(\tau^4 + h^4)/(4\epsilon) + \epsilon\sum_{n=1}^{N-1}\tau\|\Delta_h \eta^n\|^2\end{aligned}$$

$$= C_1^2((T+\frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4)/(2\epsilon) + \epsilon\|\nabla_h\eta^0\|^2 + \epsilon\tau\|\Delta_h\eta^0\|^2 + \epsilon\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2,$$

that is

$$(\frac{1}{2}-\epsilon)\|\nabla_h\eta^0\|^2 + (\frac{1}{2}-\epsilon)\tau\|\Delta_h\eta^0\|^2 + (1-\epsilon)\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2 - \sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h\eta^n) \tag{2.26}$$

$$\leq C_1^2((T+\frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4)/(2\epsilon) \tag{2.27}$$

for arbitrary $\epsilon > 0$. By choosing $\epsilon = 1/4$ so that $(1-\epsilon) \geq 1/4$ and $(1/2-\epsilon) \geq 1/4$, then the last inequality becomes

$$\frac{\|\nabla_h\eta^0\|^2}{4} + \frac{1}{4}\tau\|\Delta_h\eta^0\|^2 + \frac{1}{4}\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2 - \sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h\eta^n)$$
$$\leq 2C_1^2((T+\frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4),$$

which further indicates that

$$\sum_{n=0}^{N-1}\tau\|\Delta_h\eta^n\|^2 - 4\sum_{n=1}^{N-1}\tau(\nabla_h e^n, \nabla_h\eta^n) \leq C_3(\tau^4 + h^4). \tag{2.28}$$

Adding $\gamma\times$(2.22) with (2.28) gives

$$\gamma\sum_{n=1}^{N}\tau\|\Delta_h e^n\|^2 + \sum_{n=0}^{N-1}\tau\|\Delta_h\eta^n\|^2 \leq (\gamma C_2 + C_3)(\tau^4 + h^4), \tag{2.29}$$

which also shows (recall that $\|\Delta_h e^0\| = 0$ and $\|\Delta_h\eta^N\| = 0$)

$$\sum_{n=0}^{N}\tau\|\Delta_h e^n\|^2 + \sum_{n=0}^{N}\tau\|\Delta_h\eta^n\|^2 \leq C_4(\tau^4 + h^4). \tag{2.30}$$

Since $\max_{i,j}|e_{ij}^n| \leq C_0\|\Delta_h e^n\|$ for some positive constant C_0, the last inequality implies that

$$\sum_{n=0}^{N}(\max_{i,j}|e_{ij}^n|^2 + \max_{i,j}|\eta_{ij}^n|^2)\tau \leq C_5(\tau^4 + h^4), \tag{2.31}$$

which completes the proof.

From the proof we can also see that, if we set $F^n = G^n = 0$ in (2.11)-(2.14), then (2.11)-(2.14) imply that $e^n = \eta^n = 0$. This substantiates the invertibility of the discretized linear system (2.5)-(2.8) in our conclusions. □

Remark 2.1. In our approach, we focus on the establishment of an effective numerical scheme. Thus for the purpose of transparency, we directly assume the regularity $C^{4,3}(\overline{Q})$. Since the system is parobolic-type, one can always assume that the initial condition x_0 and the source f to be sufficiently smooth, assuming the compatible condition at the boundaries $\partial\Omega \times \{0\}$ and $\partial\Omega \times \{T\}$ is met ([9, Page 388]), to achieve this regularity requirement. Here our estimation is based on retangular domain. Some revision or additinal assumptions may be needed for other types of domain to maintain the regularity requirement.

Remark 2.2. From the proof of Theorem 2.1 we see that the proposed scheme for the KKT system is unconditional stable. In particular, Theorem 2.1 holds with h and τ arbitrary (i.e., not assumed to be sufficiently small, or dependent of each other). Moreover, by letting $p \equiv 0$ and $g \equiv y$, the above proof also implies that our leapfrog scheme achieves a second-order accuracy for solving the heat equation.

Remark 2.3. It is worthy of pointing out that our proved error estimate in terms of discrete $L^2(L^\infty)$ norm is stronger than the often used discrete $L^2(Q)$ norm estimate in literature [11, 13]. The approach technique for L^∞ norm in space also holds for dimensions $d = 1$ and $d = 3$ with minor modification of the proof. With $d > 3$ the estimate (2.31) from (2.30) may not hold due to the failure of the discrete Sobolev embedding inequality. However, a similar error estimate in discrete $L^2(Q)$ norm is still valid due to (2.30).

3. Multigrid solver for the linear system

To illustrate our multigrid linear solver for the discretized system, we reformulate our leapfrog scheme (2.5,2.6,2.7,2.8) in a two-by-two block structured linear system

$$L_h w_h := \begin{bmatrix} A_h & B_h \\ C_h & D_h \end{bmatrix} \begin{bmatrix} y_h \\ p_h \end{bmatrix} = \begin{bmatrix} f_h \\ g_h \end{bmatrix} =: b_h, \tag{3.32}$$

where

$$A_h = \begin{bmatrix} I & 0 & 0 & \cdots & 0 & 0 \\ -I/2\tau & -\Delta_h & I/2\tau & \cdots & 0 & 0 \\ 0 & -I/2\tau & -\Delta_h & I/2\tau & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & -I/2\tau & -\Delta_h & I/2\tau \\ 0 & 0 & \cdots & 0 & -I/\tau & (-\Delta_h + I/\tau) \end{bmatrix}, \tag{3.33}$$

$$B_h = \begin{bmatrix} 0 & & 0 & \cdots & 0 & 0 \\ 0 & \frac{I}{\gamma} & 0 & \cdots & 0 & 0 \\ 0 & 0 & \frac{I}{\gamma} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & \frac{I}{\gamma} & 0 \\ 0 & 0 & \cdots & 0 & 0 & \frac{I}{\gamma} \end{bmatrix}, C_h = \begin{bmatrix} I & 0 & 0 & \cdots & 0 & 0 \\ 0 & I & 0 & \cdots & 0 & 0 \\ 0 & 0 & I & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & I & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}, \tag{3.34}$$

$$D_h = \begin{bmatrix} (-I/\tau + \Delta_h) & I/\tau & 0 & \cdots & 0 & 0 \\ -I/2\tau & \Delta_h & I/2\tau & \cdots & 0 & 0 \\ 0 & -I/2\tau & \Delta_h & I/2\tau & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & -I/2\tau & \Delta_h & I/2\tau \\ 0 & 0 & \cdots & 0 & 0 & I \end{bmatrix}, \tag{3.35}$$

$$f_h = \begin{bmatrix} y_0 \\ -f^1 \\ -f^2 \\ \vdots \\ -f^{N-1} \\ -f^N \end{bmatrix}, g_h = \begin{bmatrix} g^0 \\ g^1 \\ g^2 \\ \vdots \\ g^{N-1} \\ 0 \end{bmatrix}, y_h = \begin{bmatrix} y^0 \\ y^1 \\ y^2 \\ \vdots \\ y^{N-1} \\ y^N \end{bmatrix}, \text{and } p_h = \begin{bmatrix} p^0 \\ p^1 \\ p^2 \\ \vdots \\ p^{N-1} \\ p^N \end{bmatrix}. \tag{3.36}$$

Here I is an identity matrix of appropriate size and the vectors y_0, f^n, g^n, and p^n are the lexicographic ordering (vectorization) of the corresponding approximations over mesh grids. Also notice that we include the given initial conditions $y^0 = y_0$ and $p^N = 0$ as unknowns for the unified formulation purpose.

Now, we proceed to propose a multigrid algorithm [48] for solving (3.32). Given a linear system such as (3.32) that is discretized with finest mesh-size h

$$L_h w_h = b_h,$$

one V-cycle linear multigrid iteration is recursively delineated in Algorithm 1 [30, 32], where we have to provide the coarsest mesh size h_0, the smoothing algorithm SMOOTH, the restriction operator I_h^H, the prolongation operator I_H^h, as well as the coarse grid operator L_H.

Algorithm 1: V-cycle multigrid iteration

```
$w_h := \mathrm{MG}(h, L_h, w_h^0, b_h)$

IF $(h == h_0)$
    Solve exactly: $w_h = L_h^{-1} b_h$
ELSE
    Pre-smooth $\nu_1$ times: $w_h := \textsc{smooth}^{\nu_1}(L_h, w_h^0, b_h)$
    Restriction: $r_H := I_h^H (b_h - L_h w_h)$
    Initialize correction:$\delta_H := 0$
    Recursion : $\delta_H := \mathrm{MG}(H, L_H, \delta_H, r_H)$
    Prolongation: $\delta_h := I_H^h \delta_H$
    Correction: $w_h := w_h + \delta_h$
    Post-smooth $\nu_2$ times: $w_h := \textsc{smooth}^{\nu_2}(L_h, w_h, b_h)$
ENDIF
RETURN $w_h$.
```

For our implementation, we only use semi-coarsening in space (no coarsening in time) for its fast convergence. The coarse grid operator L_H is derived from the finite difference discretization with a coarse step-size H in space. For the smoother SMOOTH, considering its lower computational costs, we make use of a damped collective Jacobi (C-JAC) smoother given in [45]. Numerical simulations indicate that the C-JAC smoother works better than conventional Gauss-Seidel (G-S) smoother, especially when the regularization parameter γ is small. In particular, a single smoothing iteration can be represented in a compact formula (with a damping factor $\omega \in (0, 1]$)

$$\begin{bmatrix} y_h^{(k+1)} \\ p_h^{(k+1)} \end{bmatrix} = \begin{bmatrix} y_h^{(k)} \\ p_h^{(k)} \end{bmatrix} + \omega J_h^{-1} \left(b_h - L_h \begin{bmatrix} y_h^{(k)} \\ p_h^{(k)} \end{bmatrix} \right),$$

with

$$J_h := \begin{bmatrix} \mathrm{diag}(A_h) & \mathrm{diag}(B_h) \\ \mathrm{diag}(C_h) & \mathrm{diag}(D_h) \end{bmatrix}, \tag{3.37}$$

where diag$(\cdot)$ stands for the diagonal part of the input matrix block, respectively. Notice that we have $\mathrm{diag}(B_h) = B_h$ and $\mathrm{diag}(C_h) = C_h$ since they are diagonal matrices. Here, the matrix-vector multiplication $J_h^{-1} v$ can be computed very efficiently based on the partitioned inverse formula [49] since these blocks are all diagonal. Indeed, the time complexity of calculating $J_h^{-1} v$ is of $\mathcal{O}(N)$. A obviously necessary condition for above smoother is the invertibility of J_h, which trivially holds for our proposed leapfrog scheme.

Similar to [50], in 2D case, we define the restriction operator I_h^H from the full-weighting averaging with the following stencil form

$$I_h^H = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

and the prolongation operator I_H^h from bilinear interpolation with a corresponding stencil form

$$I_H^h = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}.$$

For 1D case, we also have similar restriction and prolongation operators [30].

4. Numerical experiments

In this section, we will provide several numerical examples to validate the obtained theoretical results and to demonstrate the high efficiency of our proposed approach. All simulations are implemented using MATLAB R2014a on a laptop PC with Intel(R) Core(TM) i3-3120M CPU@2.50GHz and 12GB RAM. The CPU time is estimated by timing functions *tic/toc*.

For simplicity, we will denote the discrete L^2 norm on Q in short by $\|\cdot\|$, that is $\|\cdot\| := \|\cdot\|_{L_h^2(Q)}$. Based on our error estimates, we also defined the discrete $L^2(L^\infty)$ norm $\|\cdot\|_{L_\tau^2(L_h^\infty)}$. We first compute the discrete $L^2(L^\infty)$ norms of state and adjoint state approximation errors

$$e_y^h = \|y_h - y\|_{L_\tau^2(L_h^\infty)} \quad \text{and} \quad e_p^h = \|p_h - p\|_{L_\tau^2(L_h^\infty)}$$

and then estimate the experimental order of accuracy by computing the logarithmic ratios of the approximation errors between two successive refined meshes, i.e.,

$$\text{Order} = \log_2(e^{2h}/e^h),$$

which should be close to two for a second-order accuracy. Theoretically, our leapfrog scheme, the BDF2 scheme, and the CN scheme should exhibit the same second-order accuracy. However, the absolute approximation errors of our leapfrog scheme are expected to be smaller than that of BDF2 scheme since the leapfrog scheme is based on central finite difference approximations instead of one-sided finite difference formulas as in the BDF2 scheme (see Appendix A). This anticipation is verified by the following numerical simulations.

In our multigrid implementation, we choose the damping factor $\omega = 1/2$ for $d = 1$ and $\omega = 4/5$ for $d = 2$, the coarsest mesh size $h_0 = 4^{d-3}$, and the spatial coarsening mesh size $H = 2h$. In each V-cycle iterations two pre- and post- smoothing steps are performed. For initialization, the state y and the adjoint state p are set to be zero, and the stopping criterion based on relative residuals is chosen to be

$$\text{Rel. Res.} := \frac{\sqrt{\|r_y^{(k)}\|^2 + \|r_p^{(k)}\|^2}}{\sqrt{\|r_y^{(0)}\|^2 + \|r_p^{(0)}\|^2}} \leq 10^{-7},$$

where $r_y^{(k)}$ and $r_p^{(k)}$ denote the residuals after k-th V-cycle iteration.

4.1. Example 1.

Let $\Omega = (0,1)$ and $T = 2$. Let

$$f = \pi \sin(\pi x)\sin(\pi t) - \pi^2 \sin(\pi x)\cos(\pi t) - \sin(\pi x)\sin(\pi t)/\gamma$$

and

$$g = \pi \sin(\pi x)\cos(\pi t) - \pi^2 \sin(\pi x)\sin(\pi t) + \sin(\pi x)\cos(\pi t)$$

in (1.4) such that the exact solution is

$$y(x,t) = \sin(\pi x)\cos(\pi t) \quad \text{and} \quad p(x,t) = \sin(\pi x)\sin(\pi t).$$

Here the initial condition is set as $y_0(x) = \sin(\pi x)$. We test with different parameters $\gamma = 10^{-1}$ and $\gamma = 10^{-3}$.

We report in Tables .1 and .2 the errors, the experimental order of accuracy, the required multigrid iteration numbers, and the CPU time of our proposed leapfrog scheme with different parameters. Clearly, the finite difference approximations achieve a second-order accuracy for both state y and adjoint state p, which validates our theoretical conclusions. The mesh-independent number of iterations in column 'Iter' indicates our proposed multigrid solver has a roughly linear time complexity with respective to the number of degrees of freedom. Notice the CPU time increases about four times as the mesh size is halved. The almost unchanging iteration numbers for different parameters in our numerical experiments indicates that our multigrid solver is very robust with respect to the regularization parameter γ, which is very attractive to those practical applications with a possible large range of regularization parameters. At this point, we left the theoretical justification of this nice robust convergence property as future work.

As the first comparison, we report in Table .3 and Table .4 the corresponding results of the BDF2 scheme. Because the BDF2 scheme shares a similar structure with our leapfrog scheme, as a by-product, numerical experiments show that our multigrid solver also works quite well with the BDF2 scheme. This allows us to conduct an adequate comparison between these different schemes using the same multigrid solver. Comparing Tables .1 and .2 with Tables .3 and .4, our proposed leapfrog scheme delivers more accurate approximations than the BDF2 scheme with less CPU time. In particular, the multigrid solver shows better mesh-independent convergence when applied to our leapfrog scheme. In particular, our leapfrog scheme outperforms the BDF2 scheme in terms of efficiency as well as accuracy.

For a further comparison, we also report in Table .5 and Table .6 the corresponding results of the CN scheme, which is anticipated to be problematic for our multigrid solver framework. Although the CN scheme gives a comparable second-order accuracy, the required multigrid iteration numbers to fulfill the convergence criterion are almost doubled as the mesh size is halved, which will greatly degrade the computational efficiency of the CN scheme. Therefore, based on our numerical experiences and as also mentioned in [13], the CN scheme is not recommended when solving the underlying problem with the standard multigrid algorithm implementations. Nevertheless, it is still possbile to design efficient multigrid algorithms with different smoothers that work well with the CN scheme, which again will be left as our future work. In this regard, our proposed leapfrog scheme demonstrates the desired advantage in both provable second-order accuracy and fast iterative system solver.

4.2. Example 2.

Let $\Omega = (0,1)^2$ and $T = 2$. Let

$$f = \big(\pi \sin(\pi t) - 2\pi^2 \cos(\pi t) - \sin(\pi t)\big) \sin(\pi x_1) \sin(\pi x_2)$$

and

$$g = \big(\gamma\pi \cos(\pi t) - 2\gamma\pi^2 \sin(\pi t) + \cos(\pi t)\big) \sin(\pi x_1) \sin(\pi x_2)$$

in (1.4) such that the exact solution is

$$y(x,t) = \cos(\pi t) \sin(\pi x_1) \sin(\pi x_2)$$

and

$$p(x,t) = \gamma \sin(\pi t) \sin(\pi x_1) \sin(\pi x_2).$$

Here the initial condition is set as $y_0(x) = \sin(\pi x_1)\sin(\pi x_2)$. We test with different parameters $\gamma = 10^{-2}$ and $\gamma = 10^{-4}$. This example is inspired by [11], where $p(x,t)$ is chosen to be depend on γ.

We give in Tables .7-.8 and Tables .9-.10 the corresponding numerical results of our proposed leapfrog scheme (using CN scheme at the last time step) and the BDF2 scheme with different parameters, respectively. Similar as in Example 1, we observe a satisfactory second-order accuracy for our leapfrog scheme as mesh refines. Moreover, our multigrid solver achieves a desired mesh-independent convergence for our leapfrog scheme. Again, our proposed leapfrog scheme delivers more accurate approximations than the BDF2 scheme with less CPU time. Notice the performance of our multigrid solver with the BDF2 scheme is slightly getting worse when the mesh refines, as shown in columns 'Iter' of Tables .9-.10.

Finally, we choose not to present the numerical results of the CN scheme since our multigrid solver becomes divergent quickly for this example. In this case, the CN scheme could be solved by the backslash sparse direct solver within MATLAB, of which the CPU time will soar up very fast as the mesh refines. For example, it takes about 35 seconds for just a 32 × 32 × 32 mesh. Hence it would be scientifically unfair to compare its computational CPU time with our multigrid iterative solver since the sparse direct solver is based on a different philosophy. How to come up with an efficient iterative solver for the CN scheme is a quiet open problem deserving further investigations, which is beyond the scope of our current paper.

4.3. Example 3: $\gamma \to 0$ as a regularization parameter.

In this example, we will show how our proposed algorithm performs as the regularization parameter $\gamma \to 0$, which corresponds to those practical applications expecting the best goal without concerning the control costs. Inspired by [51], let $\Omega = (-1,1)^2$, $T = 5$, $f \equiv 0$, and choose a non-attainable discontinuous target function

$$g(x,t) = \begin{cases} \frac{2}{5}t(1-x_1^{12})(1-x_2^{12}), & t \le \frac{5}{2}, \\ (\frac{2}{5}t-2)(1-x_1^{12})(1-x_2^{12}), & t > \frac{5}{2}. \end{cases}$$

Here the initial condition is set as $y_0(x) = 0$ such that it fits the value $g(x,0)$. In this case, since the exact solutions are not known, we will report the relative residuals of computed approximations to verify the convergence of our multigrid solver. We also compute the difference norm $\|y_h - g\|$ to show how the computed optimal state y_h approaches the desired target g, as the regularization parameter γ approaches to zero. With a fixed $\gamma = 10^{-2}$, Table .11 presents a very similar mesh-independent convergence as in previous examples.

With varing γ, the computational results with a fixed mesh $M_1 = M_2 = N = 64$ are given in Table .12. The column "Iter" has a almost constant iteration number, which numerically shows the high robustness of our multigrid solver with respect to the regularization parameter γ. The decreasing numbers in column "$\|y_h - g\|$" indicates that a smaller regularization parameter γ provides a better control for more accurately tracking the target g. Theoretically, the difference $\|y_h - g\|$ has a uniform lower bound since g is non-attainable due to discontinuity, which explains the staggering (or convergence) of the numbers in the column "$\|y_h - g\|$". For practical purpose, one may be satisfied with $\gamma \approx 10^{-9}$, considering the derived control $u_h = p_h/\gamma$ becomes very sensitive if γ is too small. Our algorithm shows to be very stable even when the value of γ reaches the scale of machine round-off error. Moreover, Fig. .1 illustrates the target state g as well as the computed optimal state y_h and optimal control $u_h = p_h/\gamma$ at the fixed point $(x_1, x_2) = (0, 0)$ with $\gamma = 10^{-2}$, $\gamma = 10^{-4}$, and $\gamma = 10^{-6}$, respectively. Clearly, a smaller γ leads to smaller tracking errors with no extra computational costs. It is worthwhile to remark that many standard preconditioned Krylov subspace iterative methods show a clearly deteriorated convergence as the parameter γ goes to zero, which may render those methods inefficient for such type of applications. In fact, a smaller γ leads to a linear system with larger condition number. We refer to [25] for further discussion on regularization-robust preconditioners.

5. Conclusions

Although several second-order temporal schemes are proposed for the optimal control problems in order to improve the efficiency and accuracy of numerical approximations, little attention is paid to the suitability of the underlying discretization structure for the establishment of fast solvers. Due to the high dimensions of discretized data set resulting from solving PDE-constrained optimization problems, the design of a fast solver would be very difficult or even impossible if the given scheme has a poor structure, such as a direct use of the classical Crank–Nicolson scheme. An ideal scheme should be not only designed to achieve high order accuracy but also make the later implementation of fast linear system solvers approachable.

In this paper, we have established the second-order accuracy of a leapfrog central difference scheme in time for a forward-and-backward coupled parabolic PDE systems arising from standard parabolic optimal control problems. The proposed scheme is unconditionally stable and the discretized structure allows us to establish a fast iterative solver under the framework of multigrid method. The proposed scheme can be further used to study semilinear parabolic optimal control problems with control constraints using semi-smooth Newton method without the periodic state assumption in [52]. In a recent following-up paper [53], our proposed algorithm has been successfully applied to optimal control of parabolic PDEs with Robin boundary conditions, where the convergence analysis is accomplished through alternative matrix analysis techniques. Furthermore our leapfrog scheme in time and the developed multigrid algorithm should work seamlessly with finite element discretizations in space, and thus provides an approachable implementation of fast solvers to further study this type of problems. Our future work include the application of our proposed methods to nonlinear PDE models arising in image segmentation [54], image denoising [55], image registration [56, 57], and flow control [58, 59].

Acknowledgments

The authors would like to thank the two anonymous referees for their valuable comments and detailed suggestions that have greatly contributed to improving the presentation of this paper.

Appendix A: A BDF2 scheme with a Crank–Nicolson initialization step.

For the purpose of completeness, the BDF2 scheme with a Crank–Nicolson (CN) initialization step [13] for the system (1.4) using the same notations is included below. As we discussed in the introduction, the CN initialization step here is necessary for the BDF2 scheme to achieve a second-order accuracy. It is worthwhile to notice that the resultant discretized coefficient matrix of the whole BDF2 scheme (.1,.2,.3,.4) has more complicated structures compared to our proposed leapfrog scheme. Especially, we only used the first-order backward Euler scheme at the finalization step.

- **BDF2 scheme for time-stepping**: For $2 \le n \le N$, the state equation is discretized as

$$\frac{3Y^n - 4Y^{n-1} + Y^{n-2}}{2\tau} - \Delta_h Y^n + \frac{P^n}{\gamma} = -f^n. \tag{.1}$$

  Similarly, for $0 \le n \le (N-2)$, the adjoint equation is approximated as

$$-\frac{3P^n - 4P^{n+1} + P^{n+2}}{2\tau} + \Delta_h P^n + Y^n = g^n. \tag{.2}$$

- **Crank–Nicolson scheme for initialization**: For $n = 1$, the state equation is discretized as

$$\frac{Y^n - Y^{n-1}}{\tau} - \frac{\Delta_h Y^n + \Delta_h Y^{n-1}}{2} + \frac{P^n + P^{n-1}}{2\gamma} = -\frac{f^n + f^{n-1}}{2} \tag{.3}$$

  with Y^0 is given by the initial condition y_0. Similarly, for $n = N$, the adjoint equation is approximated as

$$\frac{P^n - P^{n-1}}{\tau} + \frac{\Delta_h P^n + \Delta_h P^{n-1}}{2} + \frac{Y^n + Y^{n-1}}{2} = \frac{g^n + g^{n-1}}{2} \tag{.4}$$

  with P^N is given by the terminal condition $p(\cdot, T) = 0$.

[1] J.-L. Lions, Optimal control of systems governed by partial differential equations., Springer-Verlag, New York, 1971.

[2] M. Hinze, R. Pinnau, M. Ulbrich, S. Ulbrich, Optimization with PDE constraints, Springer, New York, 2009.

[3] F. Tröltzsch, Optimal control of partial differential equations, AMS, Providence, RI, 2010.

[4] A. Borzì, V. Schulz, Computational optimization of systems governed by partial differential equations, SIAM, Philadelphia, PA, 2012.

[5] L. T. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, B. van Bloemen Waanders (Eds.), Real-time PDE-constrained optimization, Vol. 3 of Comput. Sci. Eng., SIAM, Philadelphia, PA, 2007.

[6] C. Nagaiah, K. Kunisch, G. Plank, Numerical solution for optimal control of the reaction-diffusion equations in cardiac electrophysiology, Comput. Optim. Appl. 49 (1) (2011) 149–178.

[7] K. Ito, K. Kunisch, Lagrange multiplier approach to variational problems and applications, SIAM, Philadelphia, PA, 2008.

[8] M. Ulbrich, Semismooth Newton methods for variational inequalities and constrained optimization problems in function spaces, SIAM, Philadelphia, PA, 2011.

[9] L. C. Evans, Partial differential equations, 2nd Edition, Vol. 19 of Graduate Studies in Mathematics, AMS, Providence, RI, 2010.

[10] M. Heinkenschloss, A time-domain decomposition iterative method for the solution of distributed linear quadratic optimal control problems, J. Comput. Appl. Math. 173 (1) (2005) 169–198.

[11] A. Borzì, Multigrid methods for parabolic distributed optimal control problems, J. Comput. Appl. Math. 157 (2) (2003) 365–382.

[12] T. Apel, T. G. Flaig, Crank-Nicolson schemes for optimal control problems with evolution equations, SIAM J. Numer. Anal. 50 (3) (2012) 1484–1512.

[13] S. González Andrade, A. Borzì, Multigrid second-order accurate solution of parabolic control-constrained problems, Comput. Optim. Appl. 51 (2) (2012) 835–866.

[14] I. Neitzel, F. Tröltzsch, On regularization methods for the numerical solution of parabolic control problems with pointwise state constraints, ESAIM Control Optim. Calc. Var. 15 (2) (2009) 426–453.

[15] I. Neitzel, U. Prüfert, T. Slawig, A smooth regularization of the projection formula for constrained parabolic optimal control problems, Numer. Funct. Anal. Optim. 32 (12) (2011) 1283–1315.

[16] D. Meidner, B. Vexler, A priori error estimates for space-time finite element discretization of parabolic optimal control problems. I. Problems without control constraints, SIAM J. Control Optim. 47 (3) (2008) 1150–1177.

[17] D. Meidner, B. Vexler, A priori error estimates for space-time finite element discretization of parabolic optimal control problems. II. Problems with control constraints, SIAM J. Control Optim. 47 (3) (2008) 1301–1329.

[18] D. Meidner, B. Vexler, A priori error analysis of the Petrov-Galerkin Crank-Nicolson scheme for parabolic optimal control problems, SIAM J. Control Optim. 49 (5) (2011) 2183–2211.

[19] W. Gong, M. Hinze, Z. J. Zhou, Space-time finite element approximation of parabolic optimal control problems, J. Numer. Math. 20 (2) (2012) 111–145.

[20] W. Liu, H. Ma, T. Tang, N. Yan, A posteriori error estimates for discontinuous Galerkin time-stepping method for optimal control problems governed by parabolic equations, SIAM J. Numer. Anal. 42 (3) (2004) 1032–1061.

[21] K. Chrysafinos, Convergence of discontinuous Galerkin approximations of an optimal control problem associated to semilinear parabolic PDE's, M2AN Math. Model. Numer. Anal. 44 (1) (2010) 189–206.

[22] A. Borzì, S. González Andrade, Second-order approximation and fast multigrid solution of parabolic bilinear optimization problems, Advances in Computational MathematicsTo appear. doi:10.1007/s10444-014-9369-9.

[23] T. Rees, H. S. Dollar, A. J. Wathen, Optimal solvers for PDE-constrained optimization, SIAM J. Sci. Comput. 32 (1) (2010) 271–298.

[24] R. Herzog, E. Sachs, Preconditioned conjugate gradient method for optimal control problems with control and state constraints, SIAM J. Matrix Anal. Appl. 31 (5) (2010) 2291–2317.

[25] J. W. Pearson, M. Stoll, A. J. Wathen, Regularization-robust preconditioners for time-dependent PDE-constrained optimization problems, SIAM J. Matrix Anal. Appl. 33 (4) (2012) 1126–1152.

[26] J. W. Pearson, M. Stoll, Fast iterative solution of reaction-diffusion control problems arising from chemical processes, SIAM J. Sci. Comput. 35 (5) (2013) B987–B1009.

[27] W. Hackbusch, A numerical method for solving parabolic equations with opposite orientations, Computing 20 (3) (1978) 229–240.

[28] W. Hackbusch, On the fast solving of parabolic boundary control problems, SIAM J. Control Optim. 17 (2) (1979) 231–244.

[29] W. Hackbusch, Numerical solution of linear and nonlinear parabolic control problems, in: Optimization and optimal control (Proc. Conf., Math. Res. Inst., Oberwolfach, 1980), Vol. 30 of Lecture Notes in Control and Information Sci., Springer, Berlin-New York, 1981, pp. 179–185.

[30] W. L. Briggs, V. E. Henson, S. F. McCormick, A multigrid tutorial, SIAM, Philadelphia, PA, 2000.

[31] U. Trottenberg, C. W. Oosterlee, A. Schüller, Multigrid, Academic Press Inc., San Diego, CA, 2001.

[32] Y. Saad, Iterative methods for sparse linear systems, SIAM, Philadelphia, PA, 2003.

[33] M. Hinze, M. Köster, S. Turek, A space-time multigrid method for optimal flow control, in: Constrained optimization and optimal control for partial differential equations, Vol. 160 of Internat. Ser. Numer. Math., Birkhäuser/Springer Basel AG, Basel, 2012, pp. 147–170.

[34] J. Liu, M. Xiao, A new semi-smooth Newton multigrid method for control-constrained semi-linear elliptic PDE problems, J. Global Optim. 64 (3) (2016) 451–468.

[35] G. Horton, S. Vandewalle, A space-time multigrid method for parabolic partial differential equations, SIAM J. Sci. Comput. 16 (4) (1995) 848–864.

[36] A. Borzi, R. Griesse, Experiences with a space-time multigrid method for the optimal control of a chemical turbulence model, Internat. J. Numer. Methods Fluids 47 (8-9) (2005) 879–885.

[37] A. Borzì, R. Griesse, Distributed optimal control of lambda-omega systems, J. Numer. Math. 14 (1) (2006) 17–40.

[38] A. Borzì, Space-time multigrid methods for solving unsteady optimal control problems, in: Real-time PDE-constrained optimization, Vol. 3 of Comput. Sci. Eng., SIAM, Philadelphia, PA, 2007, pp. 97–113.

[39] A. Borzi, V. Schulz, Multigrid methods for PDE optimization, SIAM Rev. 51 (2) (2009) 361–395.

[40] A. Borzì, G. von Winckel, Multigrid methods and sparse-grid collocation techniques for parabolic optimal control problems with random coefficients, SIAM J. Sci. Comput. 31 (3) (2009) 2172–2192.

[41] A. Borzì, S. González Andrade, Multigrid solution of a Lavrentiev-regularized state-constrained parabolic control problem, Numer. Math. Theory Methods Appl. 5 (1) (2012) 1–18.

[42] J. C. Strikwerda, Finite difference schemes and partial differential equations, 2nd Edition, SIAM, Philadelphia, PA, 2004.

[43] R. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems, SIAM, Philadelphia, PA, USA, 2007.

[44] O. Lass, M. Vallejos, A. Borzi, C. C. Douglas, Implementation and analysis of multigrid schemes with finite elements for elliptic optimal control problems, Computing 84 (1-2) (2009) 27–48.

[45] S. Takacs, W. Zulehner, Convergence analysis of multigrid methods with collective point smoothers for optimal control problems, Comput. Vis. Sci. 14 (3) (2011) 131–141.

[46] P. Knabner, L. Angermann, Numerical methods for elliptic and parabolic partial differential equations, Vol. 44 of Texts in Applied Mathematics, Springer-Verlag, New York, 2003.

[47] B. Jovanović, E. Süli, Analysis of Finite Difference Schemes: For Linear Partial Differential Equations with Generalized Solutions, Springer London, 2013.

[48] J. Liu, M. Xiao, A new semi-smooth newton multigrid method for parabolic pde optimal control problems, Proc. of the 53rd IEEE Conference on Decision and Control, 2014, pp. 5568–5573.

[49] R. A. Horn, C. R. Johnson, Matrix analysis, Cambridge University Press, Cambridge, 2013.

[50] A. Borzì, Smoothers for control- and state-constrained optimal control problems, Comput. Vis. Sci. 11 (1) (2008) 59–66.

[51] T. Carraro, M. Geiger, R. Rannacher, Indirect multiple shooting for nonlinear parabolic optimal control problems with control constraints, SIAM J. Sci. Comput. 36 (2) (2014) A452–A481.

[52] J. Liu, M. Xiao, A leapfrog semi-smooth Newton-multigrid method for semilinear parabolic optimal control problems, Comput. Optim. Appl. 63 (1) (2016) 69–95.

[53] J. Liu, M. Xiao, A leapfrog multigrid algorithm for the optimal control of parabolic PDEs with robin boundary conditions, Journal of Computational and Applied Mathematics 307 (2016) 216–234.

[54] B. Chen, Q.-H. Zou, W.-S. Chen, B.-B. Pan, A novel adaptive partial differential equation model for image segmentation, Applicable Analysis 93 (11) (2014) 2440–2450.

[55] Q. Liu, Z. Guo, C. Wang, Renormalized solutions to a reaction-diffusion system applied to image denoising, Discrete and Continuous Dynamical Systems - Series B 21 (6) (2016) 1839–1858.

[56] A. Mang, G. Biros, An inexact Newton–Krylov algorithm for constrained diffeomorphic image registration, SIAM Journal on Imaging Sciences 8 (2) (2015) 1030–1069.

[57] A. Mang, G. Biros, Constrained H^1-regularization schemes for diffeomorphic image registration, SIAM Journal on Imaging Sciences 9 (3) (2016) 1154–1194.

[58] M. D. Gunzburger, Perspectives in flow control and optimization, SIAM, 2003.

[59] Q. Duan, On the dynamics of Navier-Stokes equations for a spherically symmetric shallow water model, Journal of Mathematical Analysis and Applications 404 (2) (2013) 260–282.

Table .1: Results for Example 1 with our leapfrog scheme ($\gamma = 10^{-1}$).

(M,N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	4.22e-03	–	1.66e-03	–	3	0.024
(64,64)	1.04e-03	2.02	4.14e-04	2.00	3	0.034
(128,128)	2.57e-04	2.02	1.03e-04	2.00	4	0.108
(256,256)	6.38e-05	2.01	2.58e-05	2.00	4	0.309
(512,512)	1.59e-05	2.00	6.45e-06	2.00	4	0.987
(1024,1024)	3.96e-06	2.00	1.61e-06	2.00	5	4.676
(2048,2048)	9.90e-07	2.00	4.03e-07	2.00	5	20.491

Table .2: Results for Example 1 with our leapfrog scheme ($\gamma = 10^{-3}$).

(M,N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	1.88e-02	–	2.90e-04	–	4	0.028
(64,64)	4.77e-03	1.98	5.68e-05	2.35	4	0.046
(128,128)	1.20e-03	1.99	1.31e-05	2.12	4	0.112
(256,256)	3.00e-04	2.00	3.19e-06	2.04	4	0.304
(512,512)	7.50e-05	2.00	7.89e-07	2.01	4	0.992
(1024,1024)	1.88e-05	2.00	1.96e-07	2.01	4	3.938
(2048,2048)	4.70e-06	2.00	4.83e-08	2.02	4	15.608

Table .3: Results for Example 1 with the BDF2 scheme ($\gamma = 10^{-1}$).

(M,N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	4.95e-03	–	3.18e-03	–	5	0.047
(64,64)	1.20e-03	2.05	8.29e-04	1.94	5	0.112
(128,128)	2.93e-04	2.03	2.12e-04	1.97	5	0.229
(256,256)	7.22e-05	2.02	5.34e-05	1.99	6	0.605
(512,512)	1.79e-05	2.01	1.34e-05	1.99	6	2.121
(1024,1024)	4.47e-06	2.00	3.36e-06	2.00	7	6.351
(2048,2048)	1.12e-06	2.00	8.41e-07	2.00	8	31.799

Table .4: Results for Example 1 with the BDF2 scheme ($\gamma = 10^{-3}$).

(M, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	3.41e-02	–	4.58e-04	–	4	0.038
(64,64)	8.74e-03	1.96	1.07e-04	2.10	4	0.065
(128,128)	2.22e-03	1.98	2.63e-05	2.02	4	0.158
(256,256)	5.59e-04	1.99	6.56e-06	2.01	4	0.392
(512,512)	1.40e-04	1.99	1.64e-06	2.00	5	1.615
(1024,1024)	3.51e-05	2.00	4.11e-07	2.00	5	4.615
(2048,2048)	8.79e-06	2.00	1.03e-07	2.00	6	22.788

Table .5: Results for Example 1 with the CN scheme ($\gamma = 10^{-1}$).

(M, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	5.60e-04	–	1.20e-03	–	23	0.146
(64,64)	1.38e-04	2.02	2.96e-04	2.01	43	0.587
(128,128)	3.45e-05	2.00	7.37e-05	2.01	79	2.511
(256,256)	8.61e-06	2.00	1.84e-05	2.00	140	12.121
(512,512)	2.15e-06	2.00	4.59e-06	2.00	267	75.268

Table .6: Results for Example 1 with the CN scheme ($\gamma = 10^{-3}$).

(M, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	1.14e-02	–	1.51e-04	–	32	0.208
(64,64)	2.83e-03	2.01	3.58e-05	2.08	54	0.696
(128,128)	7.05e-04	2.00	8.72e-06	2.04	99	3.317
(256,256)	1.76e-04	2.00	2.15e-06	2.02	163	14.016
(512,512)	4.40e-05	2.00	5.35e-07	2.01	254	71.130

Table .7: Results for Example 2 with our leapfrog scheme ($\gamma = 10^{-2}$).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(8,8,8)	1.62e-02	–	6.44e-04	–	8	0.02
(16,16,16)	4.23e-03	1.9	1.61e-04	2.0	9	0.04
(32,32,32)	1.07e-03	2.0	4.02e-05	2.0	9	0.19
(64,64,64)	2.67e-04	2.0	1.01e-05	2.0	9	1.26
(128,128,128)	6.67e-05	2.0	2.51e-06	2.0	9	10.65
(256,256,256)	1.67e-05	2.0	6.28e-07	2.0	9	99.67

Table .8: Results for Example 2 with our leapfrog scheme ($\gamma = 10^{-4}$).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(8,8,8)	1.83e-03	–	3.69e-05	–	8	0.03
(16,16,16)	2.96e-04	2.6	9.56e-06	1.9	8	0.04
(32,32,32)	8.21e-05	1.9	2.43e-06	2.0	8	0.17
(64,64,64)	1.66e-05	2.3	6.11e-07	2.0	8	1.31
(128,128,128)	3.59e-06	2.2	1.53e-07	2.0	8	9.32
(256,256,256)	8.58e-07	2.1	3.83e-08	2.0	8	89.82

Table .9: Results for Example 2 with the BDF2 scheme ($\gamma = 10^{-2}$).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(8,8,8)	2.22e-02	–	1.11e-03	–	7	0.02
(16,16,16)	6.22e-03	1.8	3.25e-04	1.8	8	0.04
(32,32,32)	1.62e-03	1.9	8.91e-05	1.9	8	0.17
(64,64,64)	4.14e-04	2.0	2.32e-05	1.9	9	1.26
(128,128,128)	1.04e-04	2.0	5.93e-06	2.0	9	10.23
(256,256,256)	2.62e-05	2.0	1.50e-06	2.0	10	114.18

Table .10: Results for Example 2 with the BDF2 scheme ($\gamma = 10^{-4}$).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(8,8,8)	1.94e-03	–	5.40e-05	–	8	0.03
(16,16,16)	3.58e-04	2.4	1.50e-05	1.9	8	0.04
(32,32,32)	9.17e-05	2.0	3.97e-06	1.9	8	0.16
(64,64,64)	2.11e-05	2.1	1.02e-06	2.0	9	1.28
(128,128,128)	5.14e-06	2.0	2.58e-07	2.0	9	10.51
(256,256,256)	1.28e-06	2.0	6.48e-08	2.0	10	113.01

Table .11: Results for Example 3 with our leapfrog scheme ($\gamma = 10^{-2}$).

(M_1, M_2, N)	$\|y_h - g\|$	Rel. Res.	Iter	CPU
(8,8,8)	1.86858	9.43e-08	8	0.037
(16,16,16)	1.92293	9.32e-08	9	0.060
(32,32,32)	1.92859	2.86e-08	10	0.225
(64,64,64)	1.93244	3.50e-08	10	1.513
(128,128,128)	1.93743	3.78e-08	10	11.937
(256,256,256)	1.94083	3.91e-08	10	119.122

Table .12: Results for Example 3 with our leapfrog scheme ($M_1 = M_2 = N = 64$).

γ	$\|y_h - g\|$	Rel. Res.	Iter	CPU
1e-01	2.244321766405156	7.85e-08	9	1.298
1e-02	1.932437053618098	3.50e-08	10	1.435
1e-03	1.135176797865582	6.61e-08	11	1.586
1e-04	0.676364539674578	1.86e-08	12	1.749
1e-05	0.325299055954976	1.63e-08	12	1.763
1e-06	0.122678554464732	2.59e-08	12	1.816
1e-07	0.035976113323683	1.74e-08	11	1.607
1e-08	0.008838570306190	7.90e-09	10	1.452
1e-09	0.004034521514918	2.98e-08	8	1.175
1e-10	0.003867444880618	2.99e-08	8	1.174
1e-11	0.003865475149814	2.99e-08	8	1.162
1e-12	0.003865455147979	2.99e-08	8	1.154
1e-13	0.003865454949440	2.99e-08	8	1.157
1e-14	0.003865454947635	2.99e-08	8	1.144
1e-15	0.003865454947635	2.99e-08	8	1.197

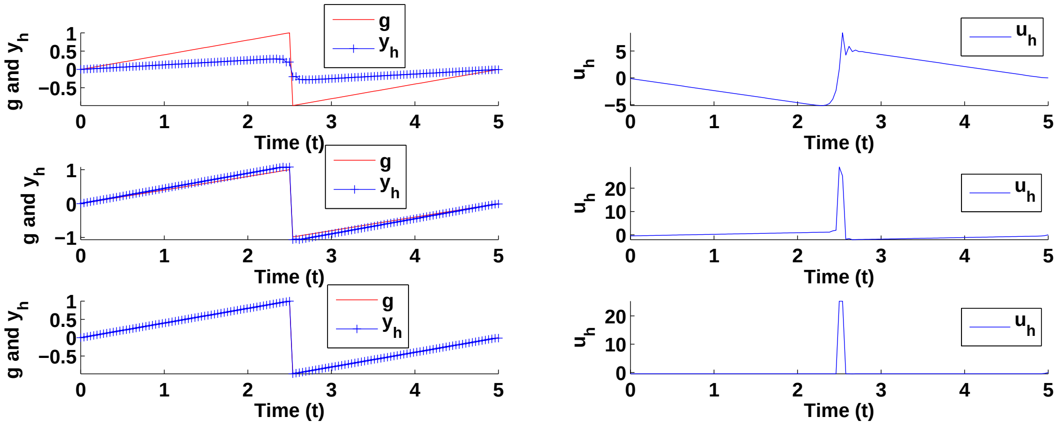

Figure .1: Computed y_h and u_h at $(x_1, x_2) = (0, 0)$ in Example 3 (with $\gamma = 10^{-2}$ (top), 10^{-4} (middle), and 10^{-6} (bottom), $M_1 = M_2 = N = 128$)