8-1-2015

NEW COMPUTATIONAL METHODS FOR OPTIMAL CONTROL OF PARTIAL DIFFERENTIAL EQUATIONS

Jun Liu
Southern Illinois University Carbondale, gdctor@gmail.com

NEW COMPUTATIONAL METHODS FOR

OPTIMAL CONTROL OF PARTIAL DIFFERENTIAL EQUATIONS

by

Jun Liu

M.S., South China Normal University, China, 2010

B.S., Guangdong University of Technology, China, 2004

A Dissertation
Submitted in Partial Fulfillment of the Requirements for the
Doctor of Philosophy Degree

Department of Mathematics
in the Graduate School
Southern Illinois University Carbondale
August, 2015

DISSERTATION APPROVAL

NEW COMPUTATIONAL METHODS FOR OPTIMAL CONTROL OF PARTIAL DIFFERENTIAL EQUATIONS

By

Jun Liu

A Dissertation Submitted in Partial

Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in the field of Computational Mathematics

Approved by:

Dr. Mingqing Xiao, Chair

Dr. Jianhong Xu

Dr. Kathleen Pericak-Spector

Dr. David Olive

Dr. Ying Chen

Graduate School
Southern Illinois University Carbondale
May 7, 2015

AN ABSTRACT OF THE DISSERTATION OF

JUN LIU, for the Doctor of Philosophy degree in COMPUTATIONAL MATHEMATICS, presented on May 7, 2015, at Southern Illinois University Carbondale.

TITLE: NEW COMPUTATIONAL METHODS FOR OPTIMAL CONTROL OF PARTIAL DIFFERENTIAL EQUATIONS

MAJOR PROFESSOR: Dr. Mingqing Xiao

Partial differential equations are the chief means of providing mathematical models in science, engineering and other fields. Optimal control of partial differential equations (PDEs) has tremendous applications in engineering and science, such as shape optimization, image processing, fluid dynamics, and chemical processes. In this thesis, we develop and analyze several efficient numerical methods for the optimal control problems governed by elliptic PDE, parabolic PDE, and wave PDE, respectively.

The thesis consists of six chapters. In Chapter 1, we briefly introduce a few motivating applications and summarize some theoretical and computational foundations of our following developed approaches.

In Chapter 2, we establish a new multigrid algorithm to accelerate the semi-smooth Newton method that is applied to the first-order necessary optimality system arising from semi-linear control-constrained elliptic optimal control problems. Under suitable assumptions, the discretized Jacobian matrix is proved to have a uniformly bounded inverse with respect to mesh size. Different from current available approaches, a new strategy that leads to a robust multigrid solver is employed to define the coarse grid operator. Numerical simulations are provided to illustrate the efficiency of the proposed method, which shows to be computationally more efficient than the popular full approximation storage (FAS) multigrid method. In particular, our proposed approach achieves a mesh-independent convergence and its performance is highly robust with respect to

the regularization parameter.

In Chaper 3, we present a new second-order leapfrog finite difference scheme in time for solving the first-order necessary optimality system of the linear parabolic optimal control problems. The new leapfrog scheme is shown to be unconditionally stable and it provides a second-order accuracy, while the classical leapfrog scheme usually is well-known to be unstable. A mathematical proof for the convergence of the proposed scheme is provided under a suitable norm. Moreover, the proposed leapfrog scheme gives a favorable structure that leads to an effective implementation of a fast solver under the multigrid framework. Numerical examples 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.

In Chapter 4, we develop a new semi-smooth Newton multigrid algorithm for solving the discretized first-order necessary optimality system that characterizes the optimal solution of semi-linear parabolic PDE optimal control problems with control constraints. A new leapfrog discretization scheme in time associated with the standard five-point stencil in space is established to achieve a second-order accuracy. The convergence (or unconditional stability) of the proposed scheme is proved when time-periodic solutions are considered. Moreover, the derived well-structured discretized Jacobian matrices greatly facilitate the development of an effective smoother in our multigrid algorithm. Numerical simulations are provided to illustrate the effectiveness of the proposed method, which validates the second-order accuracy in solution approximations as well as the optimal linear complexity of computing time.

In Chapter 5, we offer a new implicit finite difference scheme in time for solving the first-order necessary optimality system arising in optimal control of wave equations. With a five-point central finite difference scheme in space, the full discretization is proved to be unconditionally convergent with a second-order accuracy, which is not restricted by the classical Courant-Friedrichs-Lewy (CFL) stability condition on the spatial and temporal step sizes. Moreover, based on its advantageous developed structure, an efficient preconditioned Krylov subspace method is provided

and analyzed for solving the discretized sparse linear system. Numerical examples are presented to confirm our theoretical conclusions and demonstrate the promising performance of proposed preconditioned iterative solver.

Finally, brief summaries and future research perspectives are given in Chapter 6.

ACKNOWLEDGMENTS

First of all, I would like to express my deep gratitude to my dissertation advisor, Professor Mingqing Xiao, for his strong support in both academic activities and personal life during the past five years of my Ph.D. study. Without his guidance and efforts, this dissertation as well as all of my academic achievements would not have been possible. It is his inspiration and many help that brought me to this level of research. His vision and wisdom in research and teaching will always positively influence my future career.

I also would like to express my sincere thanks for Professor Buyang Li of Nanjing University who provided many helpful suggestions on Chapter 3 and 5, and for Professor William W. Hager of University of Florida who carefully read Chapter 4 and gave valuable comments/suggestions, that led to the significant improvement of this dissertation.

I also sincerely thank all members of my dissertation committee for their time and efforts in reading my dissertation. I have gained many inspiring ideas from research seminars and personal discussions with Professor Jianhong Xu. I also thank Professor Kathleen, Professor David, and Professor Ying Chen for their many supports during my Ph.D. study at Southern Illinois University.

I also want to thank the student travel award committees of the Society for Industrial and Applied Mathematics (SIAM) and IEEE, respectively, for the generous travel awards. These awards supported me to present some results of this dissertation in SIAM Conference on Optimization (May, 2014) and IEEE Conference on Decision and Control (December, 2014), respectively.

Moreover, this dissertation have benefited a lot from the courses taught by Professor Philip Feinsilver, Professor Robert Fitzgerald, Professor Sakthivel Jeyaratnam, Professor Salah-eldin Mohammed, Professor Edward Neuman, Professor Scott Spector, and Professor Dashun Xu. Besides, I also want to appreciate the support and assistance from Ms. Diane Fritcher and Professor Gregory Budzban.

Especially, I am grateful to Professor Haiwei Sun of University of Macau, who has guided

me the educational path to the U.S. during my bachelor's degree and taught me how to conduct research in computational mathematics during my master's degree.

Last, but certainly not least, I would like to thank my family. Without the constant support from my parents, I would certainly not be where I am today. I also want to thank my wife, Yurino, and our lovely daughter, Emory, for being a forever source of enjoyment, strength, and encouragement. This dissertation is my best reward of their love and encouragement.

This dissertation has been supported in part by Doctoral Fellowship and Dissertation Research Assistantship from Southern Illinois University, and in part by National Science Foundation under Grant 1021203 of United States.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

BACKGROUNDS AND PRELIMINARIES

Partial differential equations (PDEs) have broad applications in almost every area of our modern society, from airplanes in the sky to submarines under the sea, from biological movements to chemical processes, from medical imaging to drug development, etc. As the foundation of applied mathematics, PDEs have been extensively used to model the reality in every disciplines in order to better understand our world. The simulation, optimization, and control of these PDE models in natural sciences, engineering, and economics often lead to control problems governed by PDEs associated with certain control constraints due to physical restrictions [Lions, 1971, Hinze et al., 2009, Tröltzsch, 2010, Borzì and Schulz, 2012, Leugering et al., 2012, Bredies et al., 2013, Leugering et al., 2014]. Such problems arise in a wide range of applications such as flow control design [Gunzburger, 2003], gas dynamics, aerodynamic shape optimization [Jameson, 1988], and photo-acoustic tomography [Bergounioux et al., 2014]. Most of these governing PDEs are nonlinear [Neittaanmaki and Tiba, 1994, Aubert and Kornprobst, 2006, Debnath, 2012], whose analytic solutions are nearly impossible to obtain through purely theoretical investigation. Therefore, numerical approach with the help of computers becomes the most realistic approach to provide the approximated solutions, and thus effective numerical methods for the study of control and optimization of various PDE models are not only desirable but also necessary.

In particular, numerical methods for optimal control problems governed by time-dependent partial differential equations (PDEs) have recently gained dramatically increasing attention from the scientific computing community. This trend is motivated not only by its broader applications in different fields but also by the computational challenges that require new methodology. For example, the real-time optimal control [Biegler et al., 2007] of reaction-diffusion systems in cardiac electrophysiology [Nagaiah et al., 2011] demonstrates the inherent difficulties in computations. Moreover, these applications usually have a very high demand in both efficiency

and accuracy for the chosen numerical algorithms in order to achieve various purposes, which presents many dreadful challenges across related disciplines, including numerical optimization [Nocedal and Wright, 2006], numerical PDEs [Thomas, 1995, Thomas, 1999], and numerical linear algebra [Trefethen and Bau, 1997]. It requires a comprehensive understanding of the subtle interplay among these areas to develop effective numerical methods that excel at both efficiency and accuracy.

1.1 MOTIVATING APPLICATIONS

Our first application is the optimal control of stationary heating or cooling process. Let $\Omega \subset \mathbb{R}^3$ be a bounded domain with boundary $\Gamma := \partial\Omega$, which represents an object to be heated by electromagnetic induction or by microwaves. The temperature distribution or state $y(x)$ inside Ω is controlled by the enforcing heating source $u(x)$, as shown in Fig. 1.1. For some practical purposes (such as for treatment requirement), we would like to choose the optimal control which minimizes the difference between the desired stationary temperature distribution $z(x)$ and the achievable temperature distribution $y(x)$. Mathematically, by assuming the boundary temperature vanishes,

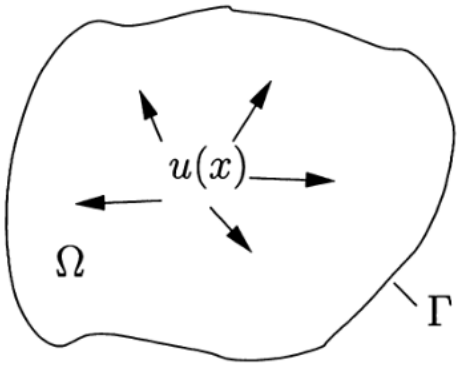

Figure 1.1. The distributed control u as heating source

we can model this process as a constrained optimization problem as follows:

$$\begin{cases} \min \quad J(y,u) := \dfrac{1}{2}\|y-z\|^2_{L^2(\Omega)} + \dfrac{\alpha}{2}\|u\|^2_{L^2(\Omega)} \\ \text{subject to:} \\ \qquad -\Delta y(x) = u(x) \quad \text{in} \quad \Omega, \\ \qquad y(x) = 0 \quad \text{on} \quad \Gamma := \partial\Omega, \\ \qquad u_a(x) \le u(x) \le u_b(x) \quad \text{in} \quad \Omega, \end{cases} \tag{1.1}$$

where Δ is the Laplacian operator, the constant $\alpha \geq 0$ can be understood as either energy cost weight associated with the control u or a regularization parameter to improve the regularity of the problem. Also, the point-wise control constraints $u_a(x) \leq u(x) \leq u_b(x)$ arises naturally from physical restrictions of the heating or cooling capacities. Here u is called distributed control since it acts in the whole domain Ω. On the contrary, we call u boundary control if it only operates on the boundary $\partial\Omega$.

Our second application is the optimal control of time-dependent heating or cooling process, which is a natural extension of previous stationary model by describing the process using a time-dependent PDE (e.g., parabolic heat equation). Let $T > 0$ be the final time of the process, $Q := \Omega \times (0,T)$ and $\Sigma := \Gamma \times (0,T)$. Assume the initial temperature is given by $y_0 = y_0(x)$. Denote the temperature on $x \in \Omega$ at time $t \in (0,T]$ by $y = y(t,x)$. Similarly, by assuming

vanishing boundary temperature, we derive a time-dependent model as follows:

$$\begin{cases} \min \quad J(y,u) := \dfrac{1}{2}\|y-z\|^2_{L^2(Q)} + \dfrac{\alpha}{2}\|u\|^2_{L^2(Q)} \\ \text{subject to:} \\ \qquad y_t(t,x) - \Delta y(t,x) = u(t,x) \quad \text{in} \quad Q, \\ \qquad y(t,x) = 0 \quad \text{on} \quad \Sigma := \Gamma \times (0,T), \\ \qquad y(0,x) = y_0(x) \quad \text{in} \quad \Omega, \\ \qquad u_a(t,x) \le u(t,x) \le u_b(t,x) \quad \text{in} \quad Q, \end{cases} \tag{1.2}$$

where $z = z(t,x)$ is the desired temperature distribution over time and y_t $(= \partial_t y)$ denotes the partial derivative of y with respect to the time variable. This parabolic control problem is computationally more challenging than the previous elliptic control problem (1.1) due the additional time variable.

Our third application is the optimal control of vibrations described by a linear wave equation. Optimal control of hyperbolic equations has many applications, such as noise reduction, focusing of ultrasound waves in medical imaging, as well as elastodynamics. A simple scenario is to use a group of pedestrians to excite oscillations in a bridge by walking across it. Denote the bridge surface domain by $\Omega \subset \mathbb{R}^2$. Let the control $u = u(t,x)$ be the force density acting in the vertical

direction. We then get the corresponding optimization problem as follows:

$$\begin{cases} \min \quad J(y,u) := \dfrac{1}{2}\|y-z\|^2_{L^2(Q)} + \dfrac{\alpha}{2}\|u\|^2_{L^2(Q)} \\ \text{subject to:} \\ \qquad y_{tt}(t,x) - \Delta y(t,x) = u(t,x) \quad \text{in} \quad Q, \\ \qquad y(t,x) = 0 \quad \text{on} \quad \Sigma := \Gamma \times (0,T), \\ \qquad y(0,x) = y_0(x) \quad \text{in} \quad \Omega, \\ \qquad y_t(0,x) = y_1(x) \quad \text{in} \quad \Omega, \\ \qquad u_a(t,x) \le u(t,x) \le u_b(t,x) \quad \text{in} \quad Q, \end{cases} \tag{1.3}$$

where $z = z(t,x)$ is the desired evolution of transverse vibrations over time, y stands for displacement, y_{tt} denotes the second-order partial derivatives with respect to time t, y_0 is the initial displacement, and y_1 is the initial velocity. This hyperbolic control problem is often deemed to be more difficult than aforementioned both elliptic and parabolic control problems due to lower regularity of the solution state variables.

1.2 GENERAL FRAMEWORK OF PDE-CONSTRAINED OPTIMIZATION

Let $\mathcal{Y}$, $\mathcal{U}$, and $\mathcal{W}$ be reflexive Banach spaces (such as Hilbert spaces), and $\mathcal{U}^{ad} \subset \mathcal{U}$ be a closed, bounded and convex set. Consider the general constrained optimization problem

$$\begin{cases} \min_{u \in \mathcal{U}^{ad}} \quad J(y,u) \\ \text{subject to:} \quad e(y,u) = 0, \end{cases} \tag{1.4}$$

with an objective functional $J : \mathcal{Y} \times \mathcal{U} \to \mathbb{R}$, state equality constraint $e : \mathcal{Y} \times \mathcal{U} \to \mathcal{W}$, as well as the control constraints $u \in \mathcal{U}^{ad}$, the set of admissible controls. We are especially interested in those applications where the state equation e is given by a well-posed PDE with suitable boundary and/or initial conditions, such as the three applications introduced in the previous section.

Current numerical methods [Hinze et al., 2009, Ulbrich, 2011, Borzì and Schulz, 2012] for solving this class of optimal control problems (1.4) generally fall into either *discretize-then-optimize* approach or *optimize-then-discretize* approach. In this thesis, we will focus on the second category by making use of its first-order optimality system. However, our proposed methods are also suitable for the first category when the discretization has a similar structure.

Before discussing the existence of optimal solution to the above optimization problem (1.4), we first need to introduce the following standard assumptions [De los Reyes, 2015] on the state equation:

(A) for each $u \in \mathcal{U}^{ad}$, there exists a unique solution $y(u) \in \mathcal{Y}$ to the state equation $e(y,u) = 0$;

(B) the set of solutions $\{y(u)\}$ is bounded in $\mathcal{Y}$ for $u \in \mathcal{U}^{ad}$;

(C) if $u_k \rightharpoonup \hat{u}$ weakly in $\mathcal{U}$, then the corresponding states $y(u_k) \rightharpoonup y(\hat{u})$ weakly in $\mathcal{Y}$.

Under the above assumptions, by defining the solution (or control-to-state) operator

$$\mathbb{S} : \quad \mathcal{U} \to \mathcal{Y}$$

$$u \mapsto y(u) = \mathbb{S}(u),$$

we can reformulate the optimization problem (1.4) in its reduced form as

$$\min_{u \in \mathcal{U}^{ad}} \quad J(u) := J(y(u), u). \tag{1.5}$$

Let $\mathcal{U}$, $\mathcal{V}$ be two Banach spaces and $F : \mathcal{U} \to \mathcal{V}$ a mapping from $\mathcal{U}$ to $\mathcal{V}$. Denote $\mathcal{L}(\mathcal{U}, \mathcal{V})$ the normed space of all bounded linear operators from $\mathcal{U}$ to $\mathcal{V}$, endowed with the operator norm $\|\cdot\|_{\mathcal{L}(\mathcal{U},\mathcal{V})}$, given by

$$\|A\|_{\mathcal{L}(\mathcal{U},\mathcal{V})} = \sup_{\|u\|_{\mathcal{U}}=1} \|Au\|_{\mathcal{V}}.$$

If $\mathcal{V} = \mathbb{R}$, we write $\mathcal{U}^* := \mathcal{L}(\mathcal{U}, \mathbb{R})$, which is also called the dual space of $\mathcal{U}$.

Definition 1 ([De los Reyes, 2015])**.** A functional $J : \mathcal{U} \to \mathbb{R}$ is called weakly lower semi-continuous if for every weakly convergent sequence $u_k \rightharpoonup \hat{u}$ as $k \to \infty$ in $\mathcal{U}$, i.e.,

$$\lim_{k\to\infty} f(u_k) = f(\hat{u}) \quad \forall f \in \mathcal{U}^*,$$

it follows that $J(\hat{u}) \leq \liminf_{k\to\infty} J(u_k)$.

In particular, if J is convex and continuous, then it is also weakly lower semi-continuous.

Definition 2 ([De los Reyes, 2015])**.** An element $\bar{u} \in \mathcal{U}^{ad}$ is called a global optimal solution (minimizer) to (1.5) if $J(\bar{u}) \leq J(u), \forall u \in \mathcal{U}^{ad}$. Further, $\bar{u}$ is called a local optimal solution (minimizer) if there exists a neighborhood $O(\bar{u})$ of $\bar{u}$ in $\mathcal{U}^{ad}$ such that $J(\bar{u}) \leq J(u), \forall u \in O(\bar{u})$.

To derive the optimality conditions that characterizing optimal solutions, it is necessary to introduce some notions of differentiability for operators between Banach spaces.

Definition 3 ([Tröltzsch, 2010]). F is called *directionally differentiable* at $u \in \mathcal{U}$ if the limit

$$\mathcal{D}F(u)(s) := \lim_{t \to 0^+} \frac{F(u+ts) - F(u)}{t} \in \mathcal{V}$$

exists for all $s \in \mathcal{U}$. In this case, $\mathcal{D}F(u) : \mathcal{U} \to \mathcal{V}$ is called directional derivative of F at u.

Definition 4 ([Tröltzsch, 2010]). F is called *Gâteaux differentiable* at $u \in \mathcal{U}$ if F is directionally differentiable at u and the corresponding directional derivative $\mathcal{D}F(u) : \mathcal{U} \to \mathcal{V}$ is bounded and linear, i.e., $\mathcal{D}F(u) \in \mathcal{L}(\mathcal{U}, \mathcal{V})$. In this case, $\mathcal{D}F(u)$ is denoted by $F'(u)$, which is called the Gâteaux derivative of F at u,

Definition 5 ([Tröltzsch, 2010]). F is called *Fréchet differentiable* at $u \in \mathcal{U}$ if F is Gâteaux differentiable at u and it satisfies

$$\lim_{\|s\|_{\mathcal{U}} \to 0} \frac{\|F(u+s) - F(u) - F'(u)s\|_{\mathcal{V}}}{\|s\|_{\mathcal{U}}} = 0.$$

In this case, $F'(u)$ is called the Fréchet derivative of F at u.

The following theorem gives the existence result of the above optimization problem (1.5)

Theorem 1.2.1 ([De los Reyes, 2015]). *Let $J : \mathcal{Y} \times \mathcal{U} \to \mathbb{R}$ be bounded from below and weakly lower semi-continuous. Then there exists a global optimal solution (minimizer) for problem* (1.5).

Hereafter we assume that $J : \mathcal{Y} \times \mathcal{U} \to \mathbb{R}$ and $e : \mathcal{Y} \times \mathcal{U} \to \mathcal{W}$ are continuously Fréchet differentiable. Denote $\bar{y} = y(\bar{u})$. We further assume that $e_y(\bar{y}, \bar{u}) \in \mathcal{L}(\mathcal{Y}, \mathcal{W})$ is a bijection, which, by the implicit function theorem, implies the existence of a (locally) unique solution $y(u)$ to the state equation $e(y, u) = 0$, in a neighborhood of $(\bar{y}, \bar{u})$, and the continuously Fréchet differentiability of the solution operator $\mathbb{S}$.

Theorem 1.2.2 ([De los Reyes, 2015]). *Suppose that $\bar{u} \in \mathcal{U}^{ad}$ is a local minimizer of* (1.5)*, then it satisfies the variational inequality $J'(\bar{u})(v - \bar{u}) \geq 0$ for all $v \in \mathcal{U}^{ad}$. In particular, if $\mathcal{U}^{ad} = \mathcal{U}$, then it implies $J'(\bar{u}) = 0$.*

We now can derive the first-order necessary optimality conditions for (1.4) by using the standard Lagrangian approach [Ito and Kunisch, 2008]. Define the Lagrangian functional corresponding to (1.4) as

$$\mathcal{L} : \mathcal{Y} \times \mathcal{U} \times \mathcal{W}^* \to \mathbb{R}$$

$$(y, u, p) \mapsto \mathcal{L}(y, u, p) := J(y, u) - \langle p, e(y, u) \rangle_{\mathcal{W}^*, \mathcal{W}},$$

where $p \in \mathcal{W}^*$ is called Lagrange multiplier or adjoint state. The first-order necessary optimality system for determining optimal control $\bar{u}$ and optimal state $y(\bar{u})$ is given by

$$\mathcal{L}_p(y, u, p) = 0 \Rightarrow e(y, u) = 0, \tag{1.6a}$$

$$\mathcal{L}_y(y, u, p) = 0 \Rightarrow e_y^*(y, u)p = J_y(y, u), \tag{1.6b}$$

$$\mathcal{L}_u(y, u, p)(v - u) \geq 0 \Rightarrow \langle J_u(y, u) - e_u^*(y, u)p, v - u \rangle_{\mathcal{U}^*, \mathcal{U}} \geq 0 \quad \forall v \in \mathcal{U}^{ad}, \tag{1.6c}$$

where $e_y^*(y, u)$ denotes the adjoint operator of $e_y(y, u)$ and the last variational inequality follows from Theorem (1.2.2).

Throughout this thesis, we will focus on the widely used case of $\mathcal{U} = L^2(\Omega)$ and box constraints on the control, i.e.,

$$\mathcal{U}^{ad} = \{u \in L^2(\Omega) : u_a \leq u(x) \leq u_b \quad a.e. \quad \text{in} \quad \Omega\}$$

with $u_a, u_b \in \mathbb{R}$ such that $u_a \leq u_b$. In this case (notice $\mathcal{U}^* = \mathcal{U}$), the above variational inequality (1.6c) holds if and only if, for almost every $x \in \Omega$,

$$(J_u(y, u)(x) - e_u^*(y, u)p(x))(v - u(x)) \geq 0 \quad \forall v \in \mathbb{R} : u_a \leq v \leq u_b. \tag{1.7}$$

Also, our considered objective functional J is quadratic and of tracking type (with $\alpha > 0$), i.e.,

$$J(y,u) = \frac{1}{2}\|y-z\|^2_{L^2(\Omega)} + \frac{\alpha}{2}\|u\|_{L^2(\Omega)},$$

which gives $J_u(y,u)(x) = \alpha u(x)$. In the following study of distributed control problems, we always have $e^*_u(y,u) = -1$, since the control term u appears alone in the right-hand-side. Under this setting, the above scalar inequality (1.7) is equivalent to the projection formula

$$u(x) = \Phi(-\frac{1}{\alpha}p(x)) := \mathsf{P}_{[u_a,u_b]}\left(-\frac{1}{\alpha}p(x)\right) = \min\{u_a, \max\{u_b, -\frac{1}{\alpha}p(x)\}\}, \tag{1.8}$$

where $\mathsf{P}_{[u_a,u_b]} : \mathbb{R} \to \mathbb{R}$ denotes the projection onto the interval $[u_a, u_b]$. The special case with $\alpha = 0$ usually gives bang-bang control with the last inequality can be converted to

$$p = \min\{0, p+u-u_a\} + \max\{0, p+u-u_b\},$$

which is not discussed in current work since it requires very different numerical treatments. However, it can be numerically approximated using our algorithms by letting the regularization parameter $\alpha \to 0$. Taking the first application (1.1) as an example, according to (1.6), we can formulate its first-order necessary optimality system as

$$\begin{aligned}
&-\Delta y - u = 0 \quad \text{in} \quad \Omega \qquad \text{and} \qquad y = 0 \quad \text{on} \quad \partial\Omega,\\
&-\Delta p = y - z \quad \text{in} \quad \Omega \qquad \text{and} \qquad p = 0 \quad \text{on} \quad \partial\Omega,\\
&u = \Phi(-\tfrac{1}{\alpha}p) \quad \text{in} \quad \Omega,
\end{aligned}$$

where we have replaced the variational inequality (1.6c) by the projection formula (1.8). Notice that u can be easily eliminated by plugging the third equation into the first one, which is computationally more economical as we now only need to solve for y and p from a coupled nonsmooth

PDEs (due to $\Phi(\cdot)$)

$$\begin{cases} -\Delta y + \Phi(\frac{1}{\alpha}p) = 0 \quad \text{in} \quad \Omega \qquad \text{and} \qquad y = 0 \quad \text{on} \quad \partial\Omega, \\ -\Delta p - y = -z \quad \text{in} \quad \Omega \qquad \text{and} \qquad p = 0 \quad \text{on} \quad \partial\Omega. \end{cases} \tag{1.9}$$

The solutions to the optimality system (1.6) are called stationary or critical points. In general, a local optimal solution $(\bar{u}, y(\bar{u}))$ to problem (1.4) is also a stationary point, but not vice versa. Usually, second-order sufficient optimality conditions are required to determine whether a stationary point is also a local optimal solution or not. The general theory of second-order optimality conditions in the control of nonlinear PDEs is still an active research topic with many recent contributions [Casas and Tröltzsch, 2012, Casas and Tröltzsch, 2015, Ali et al., 2015]. Given $\alpha > 0$, noticing that $J(y, u)$ is strictly convex and $\mathcal{U}^{ad}$ is convex, the optimization problem (1.4) is also convex if the constraint $e(y, u)$ is a linear PDE. With convexity assumption, the first-order necessary optimality conditions become also sufficient and hence second-order sufficient optimality conditions are not needed. We will not explicitly verify the second-order optimality conditions in our discussions, since the process is standard and our major interest will focus on developing efficient numerical methods for solving the optimality system (1.6) within the framework of one-shot approach [Gunzburger, 2003], i.e., to determine the optimal state, adjoint state, and optimal control simultaneously by solving the optimality system (1.6) once. The one-shot approach is very attractive since it does not involve any intermediate iterations, compared to the optimization-based approaches using the gradient information of the objective functional. However, this approach requires a good structure of discretization for the implementation of fast solvers.

1.3 FINITE DIFFERENCE DISCRETIZATION

When the state constraint $e(y, u)$ is given by a PDE, the corresponding optimality system (1.6) becomes a coupled system of PDEs, which is usually very difficult to obtain an analytic

solution. Hence it is more practical to seek its approximated solutions in a discrete form using numerical methods on modern computers. Finite difference discretization is a well recognized method of discretizing the continuous PDEs into a discrete structure suitable for numerical implementation on computers with finite precision arithmetic. In this thesis, we will focus on using the finite difference method since we only study the problems with regular rectangular domains. Moreover, it usually takes less effort to develop computationally more efficient algorithms for solving the resultant large-scale discretized systems. Nevertheless, our developed approaches are suitable for other discretizations (such as finite element method [Brenner and Scott, 2008]) as well. In particular, it may be more convenient to use the finite element method in space if the considered problem has a general domain, such as a convex polygon.

The basic idea of a finite difference discretization scheme consists of approximating the derivatives involved in the PDE with corresponding discrete difference quotients using the solution at nearby grid points. The standard approach of deriving a finite difference scheme is to expand the Taylor series of the sufficiently smooth function $f(x)$ at the concerning point $\xi \in (a, b)$ with a small step size $h > 0$, e.g.,

$$f(\xi + h) = f(\xi) + hf'(\xi) + \frac{h^2}{2}f''(\xi) + \frac{h^3}{6}f'''(\xi) + \frac{h^4}{24}f^{(4)}(\xi) + \mathcal{O}(h^5), \tag{1.10}$$

and

$$f(\xi - h) = f(\xi) - hf'(\xi) + \frac{h^2}{2}f''(\xi) - \frac{h^3}{6}f'''(\xi) + \frac{h^4}{24}f^{(4)}(\xi) + \mathcal{O}(h^5). \tag{1.11}$$

Subtracting (1.11) from (1.10) gives a central difference approximation for the first derivative

$$\frac{f(\xi + h) - f(\xi - h)}{2h} = f'(\xi) + \frac{h^2}{6}f'''(\xi) + \mathcal{O}(h^4)$$

with a second-order accuracy. The addition of (1.11) and (1.10) leads to a central difference

approximation for the second derivative

$$\frac{f(\xi+h)-2f(\xi)+f(\xi-h)}{h^2}=f''(\xi)+\frac{h^2}{12}f^{(4)}(\xi)+\mathcal{O}(h^4)$$

with a second-order accuracy. Under the assumption that $f^{(4)}(x)$ is uniformly bounded in $[a,b]$, one can truncate the high-order error terms with h^2 and h^4 to derive the second-order accurate central finite difference scheme

$$f'(\xi)\approx\frac{f(\xi+h)-f(\xi-h)}{2h}$$

and

$$f''(\xi)\approx\frac{f(\xi+h)-2f(\xi)+f(\xi-h)}{h^2},$$

for approximating the first derivative and second derivative, respectively. If the point ξ lies on the boundaries of the interval $[a,b]$, one can also obtain so-called one-sided finite difference schemes. For instance, again by Taylor series expansion, we can derive a second-order accurate finite difference approximation

$$f'(\xi)=\frac{3f(\xi)-4f(\xi-h)+f(\xi-2h)}{2h}$$

for the first derivative on the right boundary $\xi=b$. Such one-sided finite difference schemes are useful when we handle the time derivative at the initial time with a given initial condition.

As an introductory example, we illustrate how to find the numerical solution to the Poisson equation on a two-dimensional bounded domain $\Omega=(0,1)^2\subset\mathbb{R}^2$, i.e.,

$$\begin{cases} -\Delta y=f \quad \text{in} \quad \Omega \\ \\ \quad y=0 \quad \text{on} \quad \partial\Omega, \end{cases} \tag{1.12}$$

by using the above finite difference discretizations. Discretize Ω by a uniform Cartesian grid

$$\bar{\Omega}_h = \Big\{(x_1^i, x_2^j) = (ih, jh) | \quad i = 0, 1, 2, \cdots, n, n+1; j = 0, 1, 2, \cdots, n, n+1.\Big\}$$

with a mesh size step $h = 1/(n+1)$ and denote $y_{i,j} = y(x_1^i, x_2^j)$ and $f_{i,j} = f(x_1^i, x_2^j)$. By applying the above second-order central finite difference scheme to approximate the second-order partial derivatives $y_{x_1x_1}$ and $y_{x_2x_2}$ in each spatial variable, the Poisson equation (1.12) can be approximated (with truncation error $\mathcal{O}(h^2)$) by

$$-\frac{y_{i+1,j} - 2y_{i,j} + y_{i-1,j}}{h^2} - \frac{y_{i,j+1} - 2y_{i,j} + y_{i,j-1}}{h^2} = f_{i,j}$$

for $i = 1, 2, \cdots, n; j = 1, 2, \cdots, n$. This in fact gives a large system of linear equations that can be solved with different numerical algorithms, from which we can get the discrete approximation $y_{i,j}$ of the continuous solution y to the original Poisson equation. Let y_h and f_h be the corresponding horizontal-vertical lexicographic ordering (vectorization) of $y_{i,j}$ and $f_{i,j}$ over all interior grid points, respectively. By incorporating the homogeneous Dirichlet boundary condition, we can further rewrite the above system as

$$A_h y_h := -\frac{1}{h^2}\begin{bmatrix} K & I & & & \\ I & K & I & & \\ & \ddots & \ddots & \ddots & \\ & & I & K & I \\ & & & I & K \end{bmatrix} y_h = f_h, \tag{1.13}$$

where $I \in \mathbb{R}^{n\times n}$ stands for the identity matrix and

$$K = \begin{bmatrix} -4 & 1 & & & \\ 1 & -4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -4 & 1 \\ & & & 1 & -4 \end{bmatrix}_{n\times n}$$

Due to the elliptic differential operator, the resulting coefficient matrix A_h is symmetric positive definite and hence is uniquely solvable. Upon solving the discretized system (1.13) accurately (or up to the machine accuracy) for y_h, it can be proved [Hackbusch, 2003] that the obtained discrete numerical solution y_h has an approximation error of order two, i.e.,

$$\|y_h - y\|_\infty := \max_{i,j} |y_{i,j} - y(x_1^i, x_2^j)| = \mathcal{O}(h^2).$$

Therefore, to get more accurate numerical approximations, we need to choose a smaller mesh step size $h = 1/(n+1)$, which in return gives rise to a larger linear system to be solved. For example, taking $n = 10^3$ in each variable for a 3D Poisson equation, we easily reach a billion (10^9) unknowns. Unfortunately, even with the latest computers, it is still a forbidding task to efficiently solve a large linear system with possible billions of unknowns. Compared to (sparse) direct methods, iterative methods are usually preferred since they require less time and memory complexity by exploiting the sparsity of the coefficient matrix. However, simple iterative methods (such as Gauss-Seidel method) demonstrate a dramatically worsening convergence rate as the mesh size refines, which render them impractical for large-scale applications. Hence, we are especially interested in developing those iterative methods having the potential of achieving a mesh-independent convergence, as those ones to be introduced in the next section.

1.4 ITERATIVE METHODS FOR SOLVING LINEAR SYSTEM

Contrary to direct methods [Davis, 2006], which theoretically produce the exact solution after a finite number of algorithmic steps (in exact arithmetic), iterative methods construct a sequence of solution approximations such that it converges to the unique exact solution of a linear system. Broadly speaking, we can roughly classify most iterative methods into three groups [Barrett et al., 1994, Golub and Van Loan, 2013]:

- Stationary methods: Jacobi, Gauss-Seidel, etc.;
- Multigrid methods: **Geometric multigrid**, Algebraic multigrid.
- Non-stationary methods: **Krylov subspace methods**, etc.;

We will briefly introduce two representative methods from the last two groups, which constitute the foundation of our developed iterative solvers in the following chapters.

1.4.1 Multigrid method

Multigrid methods are in fact built on stationary iterative methods by performing a few such iterations on a hierarchy of multilevel discretizations (e.g., see Figure 1.2). They can often achieve an optimal time and space complexity in solving for the numerical solution of elliptic PDEs. Furthermore, they have been successfully used in a time-stepping scheme of parabolic PDEs, or directly applied to time-dependent PDEs under the space-time multigrid framework.

For a given linear system such as (1.13) that can be discretized with the finest mesh-size h

$$A_h w_h = b_h,$$

one linear multigrid V-cycle iteration [Briggs et al., 2000, Saad, 2003] is delineated in Fig. 1.4.1 , where we need to provide the coarsest mesh size $h_0 \gg h$, the smoothing algorithm SMOOTH, the restriction operator I_h^H, the prolongation operator I_H^h, as well as the coarse grid operator A_H. It is usually suggested to choose $H = 2h$ to get the best overall performance. For 2D domains, as

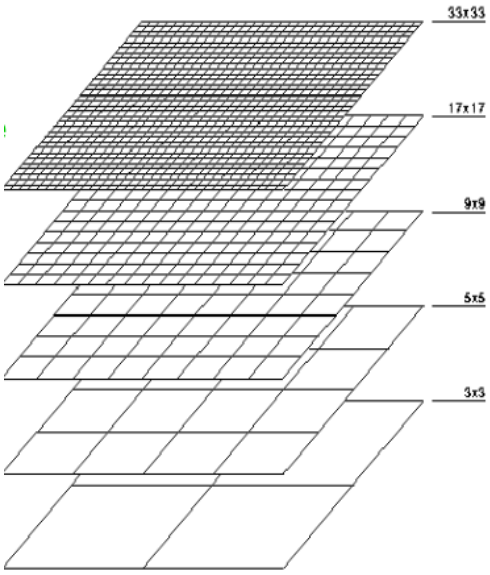

Figure 1.2. A typical hierarchy of multilevel meshes on a 2D domain

suggested in [Borzì, 2008], 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 linear interpolation with a corresponding stencil form

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

Other restriction (half weighting or injection) and prolongation (cubic interpolation) operators can also be used [Briggs et al., 2000], depending on the applications.

In each V-cycle iteration, a few smoothing iterations (such as Jacobi iterations) are conducted to improve the fine-grid approximate solution, whose major role is to smooth out the

$w_h := \mathsf{MG}(h, A_h, w_h^0, b_h)$

IF $(h == h_0)$
    Solve exactly: $w_h = A_h^{-1} b_h$
ELSE
    Pre-smooth ν_1 times: $w_h := \text{SMOOTH}^{\nu_1}(A_h, w_h^0, b_h)$
    Restriction: $r_H := I_h^H (b_h - A_h w_h)$
    Recursion: $\delta_H := \mathsf{MG}(H, A_H, 0, r_H)$
    Prolongation: $\delta_h := I_H^h \delta_H$
    Correction: $w_h := w_h + \delta_h$
    Post-smooth ν_2 times: $w_h := \text{SMOOTH}^{\nu_2}(A_h, w_h, b_h)$
ENDIF
RETURN w_h.

Figure 1.3. Algorithm of multigrid V-cycle iteration

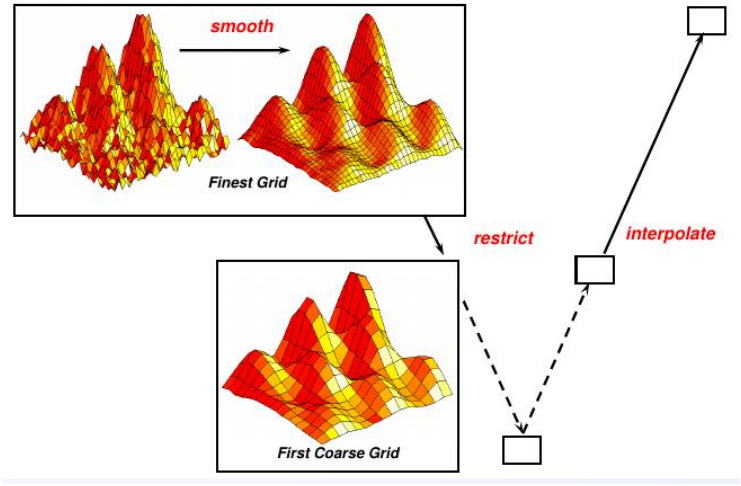

Figure 1.4. A bird's-eye view of one multigrid V-cycle iteration

high-frequency components of the approximation errors. Following this, the residual is restricted to a coarser grid ($H = 2h$) using the restriction operators I_h^H. A new V-cycle iteration is then performed on this coarser level, with this procedure proceeding recursively until the grid reaches the coarsest level $h_0 \gg h$. At the coarsest level, the underlying problem size has become very small so that it can be quickly solved by direct methods. Computing the solution on the coarser level H leads to a coarse approximation of the solution. A prolongation operator (I_H^h) interpolates this coarse grid approximation to the fine grid, which provides a coarse grid correction to the fine grid solution. Finally, the corrected fine grid solution is further enhanced by using a small number

of smoothing iterations. A straightforward definition of the coarse grid operator A_H is possible by simply using the re-discretized equation with a coarser step size H. This is the most common choice for a fully structured mesh as we used. For better illustration, one full such multigrid V-cycle iteration can also be visually depicted as in Figure 1.4[1].

Through applying the multigrid V-cycle iterations (with red-black Gauss-Seidel smoothing) to previous discretized 2D Poisson equation (1.13), we would like to demonstrate its very attractive mesh-independent convergence as well as linear time complexity. Choose $f = 2\pi^2 \sin(\pi x)\sin(\pi y)$ in (1.12) such that the exact solution is given by $y = \sin(\pi x)\sin(\pi y)$. Starting with an initial guess $y_h^0 = 0$, we update the current approximation at the k-th iteration according to

$$y_h^k = \mathsf{MG}(h, A_h, y_h^{k-1}, f_h), \quad k = 1, 2, \cdots$$

until the relative residual norm of y_h^k fulfills the prescribed stopping criterion

$$\|r_k\|_2 / \|r_0\|_2 < 10^{-8},$$

where $r_k = f_h - A_h y_h^k$ is the residual vector at the k-th iteration. The computational results are reported in Table 1.1, from which we can observe that

- When the mesh size h is halved, the infinity norm error (in column '$\|y_h^k - y\|_\infty$') is reduced to about one fourth, which indicates a second-order accuracy of the used central finite difference discretization. The experimental order of accuracy is also estimated in column 'Order' according to
$$\text{Order} = \log_2 \left(\frac{\|y_{2h}^k - y\|_\infty}{\|y_h^k - y\|_\infty} \right).$$
- The required number of V-cycle iterations, as shown in column 'Iter', to attain the stopping condition (verified in column '$\|r_k\|_2/\|r_0\|_2$') is independent of the mesh size h. We refer to this outstanding feature as mesh-independent convergence, which is very desirable in solving

[1]See also `https://computation.llnl.gov/casc/sc2001_fliers/SLS/SLS01.html`.

large-scale linear systems.

- The computational time in seconds (shown in column 'CPU') grows linearly. Notice that it increases by about four times as the mesh size h is halved (which in fact quadruples the dimension of the discretized system). This is very reasonable since it takes a fixed number of total iterations and each iteration costs the same amount of operations in terms of sparse matrix-vector product. We refer to this as $\mathcal{O}(N)$ linear time complexity, which is often called optimal time complexity in the sense that one can not get a better time complexity.

Table 1.1. Results for solving a 2D Poisson equation by multigrid method.

h	$\|r_k\|_2/\|r_0\|_2$	$\|y_h^k - y\|_\infty$	Order	Iter	CPU
1/257	7.9e-09	1.3e-05	–	10	0.37
1/513	8.0e-09	3.1e-06	2.07	10	1.40
1/1025	8.0e-09	7.8e-07	1.99	10	6.03
1/2049	8.0e-09	1.9e-07	2.04	10	25.00
1/4097	8.0e-09	4.4e-08	2.11	10	102.14

1.4.2 Krylov subspace method

In the case of hyperbolic PDEs, multigrid methods turn out to be much less successful. Instead, preconditioned Krylov subspace methods are more favorable in dealing with such linear systems, which may be highly nonsymmetric and indefinite depending on the underlying PDEs as well as discretization schemes.

Consider a general non-singular linear system

$$Av = b \in \mathbb{R}^N. \tag{1.14}$$

Given an initial guess v_0, let $r_0 = b - Av_0$. We can define Krylov subspaces of the form

$$\mathcal{K}_m(A, r_0) := \text{span}\{r_0, Ar_0, \cdots, A^{m-1}r_0\}.$$

Being very different from those stationary iterative methods based on fixed-point iterations,

Krylov subspace methods seek an approximate solution v_m from the generated Krylov subspace $\mathcal{K}_m(A, r_0)$ by imposing certain orthogonality conditions (such as Petrov-Galerkin conditions). The dimension of $\mathcal{K}_m(A, r_0)$ is expected to be much smaller, i.e., $m \ll N$. A standard Arnoldi's algorithm based on the modified Gram-Schimdt procedure can be used for building an orthogonal basis of the Krylov subspace.

The generalized minimal residual (GMRES) method is one of the most popular Krylov subspace method, which is well suitable for general large-scale non-singular sparse linear systems. The standard GMRES algorithm [Saad, 2003] without restarting is described in Fig. 1.4.2. When

$v_m := \mathsf{GMRES}(A, b, v_0, m)$

Initialize $r_0 := b - Av_0$, $\beta := \|r_0\|_2$, and $v_1 := r_0/\beta$;
FOR $k = 1, 2, \ldots, m$,
    $w_j := Av_j$
    FOR $l = 1, 2, \ldots, k$
        $h_{lk} := w_k^T v_l$
        $w_l := w_l - h_{lk} v_l$
    END
    $h_{k+1,k} := \|w_k\|_2$
    IF $(h_{k+1,k} == 0)$ SET $m := k$ BREAK
    $v_{k+1} := w_k / h_{k+1,k}$
END
$\hat{H}_m := \{h_{l,k}\}_{1 \le l \le m+1, 1 \le k \le m}$; $V_m := [v_1, \ldots, v_m]$
$y_m := \mathrm{argmin}_y \, \|\beta[1, 0, \ldots, 0]^T - \hat{H}_m y\|_2$
$v_m := v_0 + V_m y_m$
RETURN v_m.

Figure 1.5. Algorithm of GMRES method

the linear system (1.14) is solved by GMRES method with the initial guess v_0, a theoretical estimate of the residual $r_k = b - Av_k$ at the k-th iteration is given by

$$\|r_k\|_2 = \min_{p \in \mathcal{P}_k} \|p(A) r_0\|_2,$$

where $\mathcal{P}_k$ is the set of monic polynomials of degree $\le k$. However, such an abstract estimate is not practical to predict the convergence behavior of GMRES method. Alternatively, the following theorem provides us a more intuitive understanding on the factors that influence the convergence

rate of GMRES method.

Theorem 1.4.1 ([Saad and Schultz, 1986]). *Suppose that $A \in \mathbb{R}^{n\times n}$ is diagonalizable so that $A = X\Lambda X^{-1}$ with*

$$\Lambda = \mathrm{diag}[\lambda_1, \lambda_2, \cdots, \lambda_m, \lambda_{m+1}, \cdots, \lambda_n],$$

where we assume $\{\lambda_1, \lambda_2, \cdots, \lambda_m\}$ has nonpositive real parts and $\{\lambda_{m+1}, \cdots, \lambda_n\}$ is enclosed in a circle centered at $c > 0$ with radius d satisfying $c > d$, then

$$\|r_{i+1}\|_2 \leq \kappa(X) \left(\frac{d_{\max}}{d_{\min}}\right)^m \left(\frac{d}{c}\right)^{i-m} \|r_0\|_2,$$

where

$$d_{\min} = \min_{1\leq i\leq m} |\lambda_i|, \quad d_{\max} = \max_{1\leq i\leq m<j\leq n} |\lambda_i - \lambda_j|, \quad \text{and} \quad \kappa(X) = \|X\|_2\|X^{-1}\|_2.$$

Based on Theorem 1.4.1, the convergence rate of GMRES method can be improved by altering the eigenvalue distribution of A. The concept of preconditioning emerges when we transform the linear system (1.14) into another mathematically equivalent linear system

$$(AP^{-1})w = b,$$

where $v = P^{-1}w$ with a nonsingular matrix P is called (right) preconditioner. To incorporate such a right preconditioning step, we just need to modify two lines in above GMRES algorithm, i.e.,

$$w_j := Av_j \Rightarrow w_j := AP^{-1}v_j \quad \text{and} \quad v_m := v_0 + V_m y_m \Rightarrow v_m := v_0 + V_m P^{-1} y_m.$$

The idea of preconditioning lies in the fact that the preconditioned coefficient matrix AP^{-1} may have a better eigenvalue distribution than A provided P is chosen appropriately. Meanwhile, the computational cost of the preconditioning step $P^{-1}w$ should be far more less than solving the

original system by $A^{-1}b$. In another words, there are two useful criteria for evaluating a good preconditioner P:

- AP^{-1} has a very clustered eigenvalue distribution (uniformly away from the origin),
- $P^{-1}w$ can be computed or approximated with a low computational cost.

How to find a good preconditioner for more general linear systems is a very active area of current research, which often requires insightful understanding of the original problem and its discretized structures. When the coefficient matrix has better algebraic properties (such as symmetric positive definite), some variants of GMRES (such as Conjugate-Gradient algorithm) may become advantageous. But a good preconditioner is still very necessary to accelerate the convergence.

1.5 ITERATIVE METHODS FOR SOLVING NONLINEAR SYSTEM

The most important iterative method for solving nonlinear systems is Newton's method [Kelley, 2003], which often leads to a super-linear or quadratic local convergence provided the initial guess is sufficiently close to the true solution. However, such a Newton's method can not be directly applied when the system operator is not continuously Fréchet differentiable. In the case of our concerned optimality system (1.6), the variational inequality leads to a projection operator $\Phi(\cdot)$, which is not continuously Fréchet differentiable. On the other hand, as a highly efficient iterative algorithm, the FAS multigrid method is a nonlinear generalization of the linear multigrid method. It provides a powerful approach for handling nonlinear equations without the global linearization required by Newton's method. Unlike with Newton's method, there is typically no need to initialize the solver with a very good initial guess. In general, the practical efficiency of both methods depend on the underlying problems. There is no permanent winner in terms of overall performance. We will briefly describe both methods in the following two subsections.

1.5.1 Semismooth Newton (SSN) method

In our derived infinite dimensional operator equation (1.6), the non-smooth projection operator $\Phi(\cdot)$, which is not Fréchet differentiable, hinders us from directly employing the traditional

Newton method that relies on Fréchet differentiability. However, by generalizing the concept of differentiability, one could derive the semismooth Newton (SSN) method [Ulbrich, 2011] for solving the operator equation having the form as in (1.9). Let $\mathcal{X}, \mathcal{Y}$ be Banach spaces and O be an open subset of $\mathcal{X}$. The mapping $F : O \subset \mathcal{X} \mapsto \mathcal{Y}$ is called *Newton differentiable* [Chen et al., 2000, Hintermüller et al., 2002] in the open subset $V \subset O$ if there exists a family of mappings $\partial F : V \mapsto \mathcal{L}(\mathcal{X}, \mathcal{Y})$ such that

$$\lim_{\|\delta\|_{\mathcal{X}} \to 0} \frac{\|F(v+\delta) - F(v) - \partial F(v+\delta)\delta\|_{\mathcal{Y}}}{\|\delta\|_{\mathcal{X}}} = 0.$$

for every $v \in V$. We refer ∂F as a *generalized derivative* of F in V. Note that ∂F is not necessary to be unique. It is well-known that the projection (mapping) $\Phi : L^q(\Omega) \to L^s(\Omega)$ with $1 \leq s < q \leq \infty$ is Newton differentiable on $L^q(\Omega)$ and its generalized derivative, denoted by $\partial\Phi$, can be chosen as the following form

$$\partial\Phi(v)(x) = \begin{cases} 1, & \text{if} \quad u_a < v(x) < u_b; \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 1.5.1 ([Hintermüller et al., 2002])**.** *Suppose that v^* is a solution of $F(v) = 0$, and F is Newton differentiable in an open neighborhood U containing v^* with a generalized derivative $\partial F(v)$. If $\partial F(v)$ is non-singular and $\|\partial F(v)^{-1}\|$ is bounded (in a suitable operator norm) for all $v \in U$, then the semismooth Newton iteration*

$$v^{k+1} = v^k - \partial F(v^k)^{-1} F(v^k), \quad k = 0, 1, 2, \cdots \tag{1.15}$$

converges super-linearly to v^*, provided that the initial guess v^0 is sufficiently close to v^*.

For computer implementation, we have to discretize the considered optimality system (1.6) as well as its generalized derivative, which consequently leads to the discretized version of above

semismooth Newton iteration

$$v_h^{k+1} = v_h^k - \partial F_h(v_h^k)^{-1} F_h(v_h^k), \quad k = 0, 1, 2, \cdots \tag{1.16}$$

depending on the mesh-size h. Under certain mild assumptions, it was shown that the discretized SSN method has a mesh-independent convergence. This indicates the number of SSN iterations will not change as we refine the mesh. Nevertheless, the computational costs of solving the large-scale discretized Jacobian linear systems given by

$$\partial F_h(v_h^k) w_h = F_h(v_h^k) \tag{1.17}$$

are still very high if we simply use direct methods. Therefore, it is very appealing to employ those iterative methods introduced in previous section for efficiently solving the Jacobian linear systems. We mention that when the Jacobian linear system is only approximately solved, it in fact falls into the inexact Newton method [Dembo et al., 1982, Brown et al., 2003, Ortega and Rheinboldt, 2000]. We do not explicitly distinguish between those slightly different names for the simplicity of exposition. We will actually use a more instructive name, i.e., semismooth Newton-multigrid (SSN-MG) method, since the linear multigrid method is utilized for approximately solving the Jacobian linear systems. We choose to use the multigrid method because of its excellent computational efficiency compared to other types of iterative methods.

1.5.2 Full approximation scheme (FAS) multigrid method

In this section, we introduce the nonlinear full approximation scheme (FAS) multigrid method for solving the discretized nonlinear problem. For a general nonlinear system that is discretized by a finest mesh-size h

$$S_h(w_h) = b_h,$$

one FAS multigrid V-cycle iteration [Briggs et al., 2000, Trottenberg et al., 2001, Saad, 2003, Brandt and Livne, 2011] is shown in Fig. 1.5.2.

$w_h := \mathsf{FAS}(h, S_h, w_h^0, b_h)$

IF $(h == h_0)$	
Approximately solve:	$S_{h_0}(w_{h_0}) = b_{h_0}$
ELSE	
Pre-smooth ν_1 times:	$w_h := \text{SMOOTH}^{\nu_1}(S_h, w_h^0, b_h)$
Restriction residual:	$r_H := I_h^H(b_h - S_h(w_h))$
Initialize coarse guess:	$u_H := I_h^H w_h$, $w_H := \tilde{I}_h^H w_h$
Define coarse r.h.s.:	$b_H := S_H(w_H) + r_H$
Recursion:	$u_H := \mathsf{FAS}(H, S_H, u_H, b_H)$
Prolongation:	$\delta_h := I_H^h(u_H - w_H)$
Correction:	$w_h := w_h + \delta_h$
Post-smooth ν_2 times:	$w_h := \text{SMOOTH}^{\nu_2}(S_h, w_h, b_h)$
ENDIF	
RETURN w_h.	

Figure 1.6. Algorithm of FAS multigrid V-cycle iteration

In each FAS V-cycle iteration, the fine-grid solution first undergoes a few nonlinear smoothing iterations. Following this, both the solution and residual are restricted to a coarser grid ($H = 2h$) using two (possibly different) restriction operators (I_h^H, $\tilde{I}_h^H$). A new V-cycle iteration is then performed on this coarser level, with this procedure proceeding recursively until the grid reaches the coarsest level $h_0 \gg h$. At the coarsest level, the underlying problem size has become so small that it can be (approximately) solved easily using a few smoothing iterations. Computing the solution on the coarser level H leads to a coarse approximation of the solution. A prolongation operator (I_H^h) transfers this approximation to the fine grid, which provides a coarse grid correction in the fine grid solution. The fine grid solution is further improved using a few more smoothing iterations. As suggested in [Trottenberg et al., 2001], the approximation restriction operator $\tilde{I}_h^H$ is often chosen as straight injection.

The last but most crucial component is an effective smoother SMOOTH, which can effectively smooth out high-frequency components of the approximation errors. As a standalone solver, the smoothing iteration may converge very slowly as the mesh refines. This is the case for a standard nonlinear Gauss-Seidel iteration. However, because it wipes out the high-frequency components

of the approximation errors, it will serve as an ideal smoother SMOOTH. This is unsurprising given that the classical linear Gauss-Seidel iteration has been widely employed as a benchmark smoother in the linear multigrid method.

As a quick demonstration, we also apply the above FAS multigrid method to a nonlinear elliptic PDE on a two dimensional domain, i.e.,

$$-\Delta y + 10ye^y = f(x_1, x_2) \quad \text{in} \quad \Omega = (0,1)^2$$

with a given f such that the exact solution is $y = (x_1^2 - x_1^3)\sin(3\pi x_2)$. We will also use the previous introduced second-order central finite difference discretization. As in the linear case, starting with an initial guess $y_h^0 = 0$, we update the current approximation at the k-th iteration according to

$$y_h^k = \mathsf{FAS}(h, S_h, y_h^{k-1}, b_h), \quad k = 1, 2, \cdots$$

until the relative residual norm of y_h^k fulfills the prescribed stopping criterion

$$\|r_k\|_2 / \|r_0\|_2 < 10^{-8},$$

where $r_k = b_h - S_h(y_h^k)$ is the residual vector at the k-th iteration. The corresponding computational results are reported in Table 1.2, which shows a very similar excellent performance like that observed for the linear multigrid method.

Table 1.2. Results for solving a 2D nonlinear elliptic equation by FAS multigrid method.

h	$\|r_k\|_2/\|r_0\|_2$	$\|y_h^k - y\|_\infty$	Order	Iter	CPU
129	3.00e-09	5.54e-05	–	11	0.17
257	3.69e-09	1.39e-05	2.00	11	0.55
513	4.26e-09	3.46e-06	2.00	11	2.26
1025	4.69e-09	8.66e-07	2.00	11	9.62
2049	4.99e-09	2.16e-07	2.00	11	39.62
4097	5.18e-09	5.41e-08	2.00	11	166.80

CHAPTER 2

A NEW SSN-MULTIGRID METHOD FOR SEMILINEAR ELLIPTIC CONTROL PROBLEMS WITH CONTROL CONSTRAINTS

2.1 INTRODUCTION

In this chapter, we consider the following distributed optimal control problem of minimizing the tracking type cost functional

$$J(u) = \frac{1}{2}\|y - z\|^2_{L^2(\Omega)} + \frac{\gamma}{2}\|u\|^2_{L^2(\Omega)} \tag{2.1}$$

over the set U^{ad} of admissible controls given by

$$U^{ad} = \{u \in L^2(\Omega) \mid u_a \leq u \leq u_b \quad a.e. \quad \text{in} \quad \Omega\},$$

subject to a semi-linear elliptic PDE boundary value problem

$$\begin{cases} -\Delta y + S(y) = f + u \quad \text{in} \quad \Omega \\ y = 0 \quad \text{on} \quad \partial\Omega, \end{cases} \tag{2.2}$$

where $\Omega = (0,1)^2$, u is the control, $z \in L^2(\Omega)$ is the target state, $\gamma > 0$ represents either the weight of the cost of control or the Tikhonov regularization parameter, $S : \mathbb{R} \to \mathbb{R}$ is a given nonlinear function, $f \in L^2(\Omega)$, and $\{u_a, u_b\} \subset L^\infty(\Omega)$. The existence and uniqueness of the solution to the state equation (2.2) for every given $u \in L^2(\Omega)$ requires suitable assumptions on S [Arada et al., 2002, Casas, 2007]. Under some appropriate assumptions, the convexity of the cost functional J guarantees the existence of at least one solution (it may not be unique) to the above control problem (2.1–2.2).

During the last two decades, many efficient numerical methods for solving this type of semi-

linear elliptic optimal control problems (2.1–2.2) have been proposed [Ito and Kunisch, 2008, Ulbrich, 2011, Borzì and Schulz, 2012]. The full-approximation-storage multigrid (FAS-MG) method was first introduced in [Borzí and Kunisch, 2005, Borzì and Schulz, 2009] for the linear case (S is linear), where the point-wise projected collective Gauss-Seidel iteration acts as an effective smoother. In [Borzì, 2007a, Borzì, 2008] the author further generalizes their FAS-MG method to nonlinear case ($S(y) = y^4$) by employing local Newton iterations as a smoother. The reported numerical results demonstrate that FAS-MG method does achieve a typical 'textbook' multigrid convergence, however, its overall numerical efficiency in the nonlinear case is not quite satisfactory due to the high computational cost of the point-wise local Newton iteration smoother, in particular, for the more complicated nonlinear term S. It seems that there is little room for the further improvement of the smoother within the FAS-MG framework in order to reduce the computational cost.

In [Schöberl et al., 2011], the authors theoretically proved, for an unconstrained linear case, their proposed W-cycle based multigrid method has an γ-independent convergence provided $\gamma \geq ch^4$ for some constant c. A similar condition was also derived in [Engel and Griebel, 2011], where the authors developed a multigrid method with a block preconditioned Richardson iteration as a smoother for solving the Karush-Kuhn-Tucker (KKT) system arising in each iteration via primal-dual active-set method [Bergounioux et al., 1999]. More recently, in [Takacs and Zulehner, 2011, Takacs and Zulehner, 2013] the authors develop and analyze a class of multigrid methods with a so-called collective point smoother for the linear case without control constraints. Its mesh-independent convergence is shown to be quite robust with respect to the regularization (or cost) parameter. Nevertheless, these results generally do not directly apply to the nonlinear case with control constraints.

In [Hintermüller and Ulbrich, 2004], the authors establish the mesh-independence convergence of the semi-smooth Newton (SSN) method [Chen et al., 2000] applied to (2.1–2.2) under certain assumptions. For linear case, the primal-dual active-set method [Bergounioux et al., 1999] is shown to be a special case of the general SSN method [Hintermüller et al., 2002]. It's critical to

observe that the SSN method can simultaneously handle the non-smooth control constraints and the nonlinear term S, which will be linearized at the same time during each Newton iteration. This motivates us to investigate the SSN method through applying the linear multigrid solver with an appropriate smoother to each linearized Newton system in order to achieve a better computational efficiency. Related to this direction, a mesh-independence convergence result is proved in [Brown et al., 2003], where the authors apply multigrid method to solve the linear Jacobian system of smooth Newton method.

Our main contributions in this chapter are: (i) to derive a new formulation of the SSN method that can be applied to the optimality system of (2.1–2.2) so that the computational efficiency can be greatly improved by incorporating a standard linear multigrid method for solving the linearized Newton systems; (ii) to design an efficient implementation of the preceding multigrid method so that it can achieve a robust mesh-independent convergence with respect to the regularization parameter γ. It's worthwhile to mention that we apply the SSN method to solve both state and adjoint variables in the sense of 'all-at-once' method [Takacs and Zulehner, 2013], which is different from the strategy of reducing the coupled system to one equation depending on only state variable [Hintermüller and Ulbrich, 2004] or control variable [Hackbusch, 1980, Hintermüller et al., 2008, Hinze and Vierling, 2012] before applying the SSN method. The FAS multigrid method treats the nonlinear term through projected local Newton iterations as smoother, while our proposed SSN multigrid (SSN-MG) method linearizes both the semi-smoothness and nonlinear term by SSN prior to using the linear multigrid method. The proposed approach, demonstrated by theoretical discussions as well as numerical simulations, presents a significant improvement in the computational efficiency.

The rest of the chapter is organized as follows. In next section, a new formulation of semi-smooth Newton method is presented for solving the first-order necessary optimality system and its corresponding finite difference discretization. In Section 2.3, we provide a detailed implementation of the linear multigrid method for approximately solving the saddle-point linear system in each semi-smooth Newton iteration. The FAS multigrid method is summarized in detail in Section 2.4

for readiness and a handy comparison with our proposed approach. Numerical experiments are carried out in Section 2.5 to demonstrate the effectiveness of the proposed method. Finally, the chapter ends with concluding remarks in Section 2.6.

2.2 SSN METHOD FOR OPTIMALITY SYSTEM

In this section we introduce the semi-smooth Newton method for the optimality system of the optimal control problem (2.1–2.2). To characterize the possible optimal solutions of (2.1–2.2), the first-order necessary optimality conditions can be stated as [Lions, 1971, Tröltzsch, 2010]

$$\begin{cases} -\Delta y + S(y) - u = f \quad \text{in} \quad \Omega \qquad \text{and} \qquad y = 0 \quad \text{on} \quad \partial\Omega, \\ -\Delta p + S'(y)p + y = z \quad \text{in} \quad \Omega \qquad \text{and} \qquad p = 0 \quad \text{on} \quad \partial\Omega, \\ (\gamma u - p, v - u) \geq 0 \quad \text{for all} \quad v \in U^{ad}, \end{cases} \tag{2.3}$$

where p is called adjoint state. By making use of the principle of variational inequality, one can obtain the following equivalent characterization of the optimal control

$$u = \Phi(p/\gamma) := \min\{u_a, \max\{u_b, p/\gamma\}\}, \tag{2.4}$$

where $\Phi(\cdot)$ denotes the element-wise projection onto U^{ad}. By substituting (2.4) into the optimality conditions (2.3) so that u can be eliminated, we thus obtain the following non-smooth nonlinear optimality system in terms of (y, p)

$$\begin{cases} -\Delta y + S(y) - \Phi(p/\gamma) = f \quad \text{in} \quad \Omega \qquad \text{and} \qquad y = 0 \quad \text{on} \quad \partial\Omega, \\ -\Delta p + S'(y)p + y = z \quad \text{in} \quad \Omega \qquad \text{and} \qquad p = 0 \quad \text{on} \quad \partial\Omega. \end{cases} \tag{2.5}$$

We employ the second order five-point finite difference scheme [Borzí and Kunisch, 2005,

Borzì, 2008] for the discretization of (2.5). Discretize Ω using a uniform Cartesian grid

$$\Omega_h = \Big\{(x_1^i, x_2^j) = (ih, jh) | \quad i = 1, 2, \cdots, n; j = 1, 2, \cdots, n.\Big\}$$

with mesh size $h = 1/(n+1)$ and then let $y_{i,j}$, $p_{i,j}$, $f_{i,j}$, and $z_{i,j}$ represents an approximation to $y(x_1^i, x_2^j)$, $p(x_1^i, x_2^j)$, $f(x_1^i, x_2^j)$, and $z(x_1^i, x_2^j)$, respectively. Also let y_h, p_h, f_h, and z_h be the corresponding lexicographic ordering (vectorization) of those approximations over all interior grid points. Denote the corresponding discretization of Laplacian Δ by Δ_h, where the homogeneous Dirichlet boundary conditions are also included. More specifically, $\Delta_h = (I \otimes J_h) + (J_h \otimes I)$ with $J_h = \quad \text{tridiag} \quad (1, -2, 1)/h^2$, and I being an identity matrix with appropriate dimension. After discretizing (2.5), we thus obtain the discrete optimality system in the form of

$$F_h(y_h, p_h) := \begin{bmatrix} -\Delta_h y_h + S(y_h) - \Phi(p_h/\gamma) - f_h \\ -\Delta_h p_h + S'(y_h)p_h + y_h - z_h \end{bmatrix} = 0 \tag{2.6}$$

where $S(\cdot)$, $S'(\cdot)$, and $\Phi(\cdot)$ are element-wisely defined and so is the multiplication $S'(y_h)p_h$. It is straightforward to verify that F_h in (2.6) has a generalized derivative

$$G_h(y_h, p_h) = \begin{bmatrix} -\Delta_h + \mathcal{D}(S'(y_h)) & -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}p_h)) \\ I + \mathcal{D}(S''(y_h)p_h) & -\Delta_h + \mathcal{D}(S'(y_h)) \end{bmatrix}$$

where $\mathcal{D}(\cdot)$ denotes a diagonal matrix with the input vector as the diagonal elements. Analogously, we could treat the discrete optimality system (2.6) by the discrete version of SSN method given by (1.16). However, to achieve a mesh-independent convergence, we do require the Jacobian matrix G_h has a uniformly bounded inverse in some open neighborhood containing the optimal control and state with respect to h, as stated in Theorem 1.5.1. For proving our following theoretical result, we need to confine the nonlinear function $S : \mathbb{R} \to \mathbb{R}$ by introducing the following two assumptions:

(A1) $S \in C^3$ and S' is non-negative, which are the same conditions given in [Hintermüller and Ulbrich, 2004];

(A2) $S''(y(x_1, x_2))p(x_1, x_2) + 1 \geq \kappa_1$ for some $\kappa_1 > 0$ in some neighborhood of the optimal y and p, which is similar to the second-order necessary conditions in [Borzì and Kunisch, 2006, Borzì, 2007a].

We remark here that the assumption (A2) is a sufficient condition to guarantee the optimal solution, which may not be necessary to the proposed algorithm in some cases. Clearly, (A2) holds when S is linear. The (A2) assumption may be replaced by a more transparent one if bounds for optimal y and p can be estimated for a specific S given at hand.

Theorem 2.2.1. *Under the assumptions* (A1) *and* (A2), $\|G_h(y_h, p_h)^{-1}\|_2$ *is uniformly bounded for all* $h > 0$, *where* $\|\cdot\|_2$ *is the operator (spectral) norm associated with the discrete* L^2 *norm.*

Proof. Let

$$\begin{bmatrix} B_h & D_h \\ C_h & B_h \end{bmatrix} := G_h(y_h, p_h) = \begin{bmatrix} -\Delta_h + \mathcal{D}(S'(y_h)) & -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}p_h)) \\ I + \mathcal{D}(S''(y_h)p_h) & -\Delta_h + \mathcal{D}(S'(y_h)) \end{bmatrix}. \tag{2.7}$$

We first symmetrize the system by reordering the rows

$$\begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \begin{bmatrix} B_h & D_h \\ C_h & B_h \end{bmatrix} = \begin{bmatrix} C_h & B_h \\ B_h & D_h \end{bmatrix} =: T_h,$$

which gives

$$\left\| \begin{bmatrix} B_h & D_h \\ C_h & B_h \end{bmatrix}^{-1} \right\|_2 = \left\| \begin{bmatrix} C_h & B_h \\ B_h & D_h \end{bmatrix}^{-1} \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} C_h & B_h \\ B_h & D_h \end{bmatrix}^{-1} \right\|_2 = \left\| T_h^{-1} \right\|_2$$

since the spectral norm is unitarily invariant. Let the eigenvalues of a square matrix A be arranged

so that $|\lambda_{\max}(A)| \geq \cdots \geq |\lambda_{\min}(A)|$, and the singular values of A be ordered as $\sigma_{\max}(A) \geq \cdots \geq \sigma_{\min}(A)$. It is well-known [Horn and Johnson, 2013] that the singular values of a symmetric square matrix are merely the absolute value of its eigenvalues. Therefore, for a symmetric A, there holds $\sigma_{\min}(A) = |\lambda_{\min}(A)|$.

We first show that T_h is indeed invertible and then prove $\|T_h^{-1}\|_2$ is uniformly bounded. Let (λ, ξ) be any eigenpair of T_h with a normalized eigenvector $\|\xi\|_2^2 = \xi^*\xi = 1$. Partition $\xi = \begin{bmatrix} \xi_1 \\ \\ \xi_2 \end{bmatrix}$ according to the block structure of T_h. Then $T_h\xi = \lambda\xi$ gives

$$\begin{cases} C_h\xi_1 + B_h\xi_2 = \lambda\xi_1 \\ \\ B_h\xi_1 + D_h\xi_2 = \lambda\xi_2. \end{cases} \tag{2.8}$$

Since $B_h = -\Delta_h + \mathcal{D}(S'(y_h))$ is positive definite, it's obvious $\xi_1 = 0$ if and only if $\xi_2 = 0$. Thus we have both $\xi_1 \neq 0$ and $\xi_2 \neq 0$. By multiplying the first equation by ξ_1^* and the second one by ξ_2^* we get

$$\begin{cases} \xi_1^*C_h\xi_1 + \xi_1^*B_h\xi_2 = \lambda\xi_1^*\xi_1 \\ \\ \xi_2^*B_h\xi_1 + \xi_2^*D_h\xi_2 = \lambda\xi_2^*\xi_2. \end{cases} \tag{2.9}$$

Notice $(\xi_1^*B_h\xi_2)^* = \xi_2^*B_h\xi_1$ and $\lambda^* = \lambda$. On subtracting the second equation from the conjugate of first equation we obtain

$$\xi_1^*C_h\xi_1 - \xi_2^*D_h\xi_2 = \lambda(\xi_1^*\xi_1 - \xi_2^*\xi_2).$$

By Assumption (A2), the diagonal matrix $C_h = I + \mathcal{D}(S''(y_h)p_h)$ is positive definite with $\lambda_{\min}(C_h) \geq \kappa_1$. By continuity we further have $\lambda_{\max}(C_h) \leq \kappa_2$ for some $\kappa_2 \geq \kappa_1 > 0$. By the definition of $\partial\Phi$, the diagonal matrix $D_h = -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}p_h))$ is negative semidefinite since

those entries with active control constraints are zeros. Based on these facts, we have

$$0 < \kappa_1 \xi_1^* \xi_1 \leq \xi_1^* C_h \xi_1 \leq \xi_1^* C_h \xi_1 - \xi_2^* D_h \xi_2 = \lambda(\xi_1^* \xi_1 - \xi_2^* \xi_2), \tag{2.10}$$

which implies $\lambda \neq 0$ and thus T_h is invertible.

Next, we will estimate the bounds of all eigenvalues of T_h. For $\lambda > 0$, from the above inequality we get

$$(\lambda - \kappa_1)\xi_1^* \xi_1 \geq \lambda \xi_2^* \xi_2 > 0,$$

which implies $\lambda > \kappa_1 > 0$ since $\xi_1^* \xi_1 > 0$.

For $\lambda < 0$, $(C_h - \lambda I)$ is positive definite, according to the first equation in (2.8) we obtain $\xi_1 = -(C_h - \lambda I)^{-1} B_h \xi_2$, which can be substituted into the second equation and leads to

$$B_h (C_h - \lambda I)^{-1} B_h \xi_2 - D_h \xi_2 = -\lambda \xi_2.$$

Multiplying from the left by ξ_2^* and noticing $(-D_h)$ is positive semidefinite we obtain

$$\xi_2^* B_h (C_h - \lambda I)^{-1} B_h \xi_2 \leq \xi_2^* B_h (C_h - \lambda I)^{-1} B_h \xi_2 - \xi_2^* D_h \xi_2 = -\lambda \xi_2^* \xi_2.$$

This further gives

$$(\lambda_{\max}(C_h) - \lambda)^{-1} \sigma_{\min}^2(B_h) \leq \frac{\xi_2^* B_h (C_h - \lambda I)^{-1} B_h \xi_2}{\xi_2^* B_h B_h \xi_2} \frac{\xi_2^* B_h B_h \xi_2}{\xi_2^* \xi_2} = \frac{\xi_2^* B_h (C_h - \lambda I)^{-1} B_h \xi_2}{\xi_2^* \xi_2} \leq -\lambda,$$

which is

$$\lambda^2 - \lambda_{\max}(C_h)\lambda - \sigma_{\min}^2(B_h) \geq 0.$$

Under the condition $\lambda < 0$, we derive

$$\lambda \leq \tfrac{1}{2}\left(\lambda_{\max}(C_h) - \sqrt{\lambda^2_{\max}(C_h) + 4\sigma^2_{\min}(B_h)}\right).$$

It follows from $B_h = -\Delta_h + \mathcal{D}(S'(y_h))$ is symmetric positive definite with $S'(y_h) \geq 0$ and the monotonicity theorem [Horn and Johnson, 2013] that

$$\sigma_{\min}(B_h) = \lambda_{\min}(-\Delta_h + \mathcal{D}(S'(y_h))) \geq \lambda_{\min}(-\Delta_h) = 2\pi^2 - O(h^2) > \pi^2$$

for any $h < 1$, where the estimation of $\lambda_{\min}(-\Delta_h)$ is a classical result [Hackbusch, 2003]. Thus

$$\lambda < \tfrac{1}{2}\left(\lambda_{\max}(C_h) - \sqrt{\lambda^2_{\max}(C_h) + 4\pi^4}\right) = \tfrac{-2\pi^4}{\lambda_{\max}(C_h)+\sqrt{\lambda^2_{\max}(C_h)+4\pi^4}} \leq \tfrac{-2\pi^4}{\kappa_2+\sqrt{\kappa_2^2+4\pi^4}}.$$

To this end, we have shown that either $\lambda > \kappa_1 > 0$ or $\lambda < \frac{-2\pi^4}{\kappa_2+\sqrt{\kappa_2^2+4\pi^4}} < 0$, which gives

$$\|T_h^{-1}\|_2 = \tfrac{1}{\sigma_{\min}(T_h)} = \tfrac{1}{|\lambda_{\min}(T_h)|} \leq \tfrac{1}{\min(\kappa_1, \frac{2\pi^4}{\kappa_2+\sqrt{\kappa_2^2+4\pi^4}})} = \max\{\tfrac{1}{\kappa_1}, \tfrac{\kappa_2+\sqrt{\kappa_2^2+4\pi^4}}{2\pi^4}\},$$

where $0 < \kappa_1 \leq \kappa_2$ are independent of mesh size h. This completes the proof. □

Our above proof mainly follows the arguments in [Silvester and Wathen, 1994] for estimating the eigenvalue bounds of preconditioned saddle point systems arising from stabilized Stokes systems. The major difference from Stokes systems is that the (1,2) and (2,1) blocks in our case, B_h, have nice algebraic properties with $\sigma_{\min}(B_h) \neq 0$, which allows us to derive the uniformly boundedness of the inverse T_h^{-1}. Additionally, the invertibility of the saddle point system T_h was also discussed in the review paper [Benzi et al., 2005]. The authors showed that T_h is invertible if $\ker(C_h) \cap \ker(B_h) = \{0\}$ or B_h has full rank. However, the uniform boundedness of $\|T_h^{-1}\|_2$, which is essential to our approach, was not investigated there.

Based on the above discussions, the semi-smooth Newton method for solving (2.6) can be

iterated as

$$\begin{bmatrix} y_h^{k+1} \\ p_h^{k+1} \end{bmatrix} = \begin{bmatrix} y_h^k \\ p_h^k \end{bmatrix} - G_h(y_h^k, p_h^k)^{-1} F_h(y_h^k, p_h^k), \quad k = 0, 1, 2, \cdots$$

where the initials (y_h^0, p_h^0) will be specified accordingly. In each semi-smooth Newton iteration, we need to solve the linearized Newton system

$$\begin{bmatrix} -\Delta_h + \mathcal{D}(S'(y_h^k)) & -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}p_h^k)) \\ I + \mathcal{D}(S''(y_h^k)p_h^k) & -\Delta_h + \mathcal{D}(S'(y_h^k)) \end{bmatrix} \begin{bmatrix} \delta y^k \\ \delta p^k \end{bmatrix} = F_h(y_h^k, p_h^k), \tag{2.11}$$

and to then update the k-th approximation by

$$\begin{bmatrix} y_h^{k+1} \\ p_h^{k+1} \end{bmatrix} = \begin{bmatrix} y_h^k \\ p_h^k \end{bmatrix} - \begin{bmatrix} \delta y^k \\ \delta p^k \end{bmatrix}.$$

The summary of current numerical methods for solving saddle-point systems such as (2.11) can be found in the review paper [Benzi et al., 2005]. However, it is not difficult to see that the numerical computation of (2.11) becomes more challenging as $\gamma \to 0$ since the system tends to be more ill-conditioned. The simple fixed-point iterative method usually deteriorates or fails to converge when γ becomes small (about $< 10^{-3}$). The multigrid method has been successfully employed to solve ill-conditioned Toeplitz systems [Chan et al., 1998]. Unfortunately, here the underlying system (2.11) is not Toeplitz and thus does not have those nice features that Toeplitz matrices offer . Therefore, to develop an efficient multigrid scheme, it becomes necessary to improve the SSN method, which will be shown in next section.

2.3 MULTIGRID METHOD FOR JACOBIAN SYSTEM

This section is devoted to developing a multigrid algorithm for approximately solving (2.11). We now carry out a specific multigrid implementation for our previous saddle point linear

system (2.11), which can be simplified (by omitting subscript k) as

$$\begin{bmatrix} B_h & D_h \\ C_h & B_h \end{bmatrix} \begin{bmatrix} \delta y \\ \delta p \end{bmatrix} = F(y_h, p_h) \tag{2.12}$$

where

$$A_h := \begin{bmatrix} B_h & D_h \\ C_h & B_h \end{bmatrix} := \begin{bmatrix} -\Delta_h + \mathcal{D}(S'(y_h)) & -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}p_h)) \\ I + \mathcal{D}(S''(y_h)p_h) & -\Delta_h + \mathcal{D}(S'(y_h)) \end{bmatrix}. \tag{2.13}$$

Next, we discuss how to construct the coarse grid operator

$$A_H := \begin{bmatrix} B_H & D_H \\ C_H & B_H \end{bmatrix}.$$

Unlike the algebraic multigrid, we perform coarsening in a geometric way. The blocks B_H and C_H are derived from the finite difference discretization with a coarse step-size H, that is,

$$B_H = -\Delta_H + \mathcal{D}(S'(I_h^H y_h)) \qquad \text{and} \qquad C_H = I + \mathcal{D}(S''(I_h^H y_h) I_h^H p_h).$$

However, there are two possible approaches to coarse the non-smooth operator

$$D_h = -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}p_h)).$$

The first strategy directly applies I_h^H to the adjoint variable p_h to obtain

$$\tilde{D}_H := -\frac{1}{\gamma}\mathcal{D}(\partial\Phi(\frac{1}{\gamma}I_h^H p_h)) \tag{2.14}$$

which fails to achieve favorable convergence for small γ ($< 10^{-3}$) in our simulations. It seems that

this type of deteriorated performance results from those extra high frequency errors introduced by the non-smooth operator $\partial\Phi$, and these errors are supposed to be smoothed out by the smoother on fine grid. As an alternative, we place the restriction operator I_h^H on the non-smooth operator $\partial\Phi$, that is,

$$D_H := -\frac{1}{\gamma}\mathcal{D}(I_h^H \partial\Phi(\frac{1}{\gamma}p_h)), \tag{2.15}$$

which provides more information to the coarse operator compared with $\tilde{D}_H$. Postponing the restriction operator I_h^H after the non-smooth operator $\partial\Phi$ results in the iterations being able to capture more nonlinear structure of the finer system.

For the smoother SMOOTH, we tested both the standard Gauss-Seidel (G-S) smoother and the more recently damped collective Jacobi (C-JAC) smoother given in [Takacs and Zulehner, 2011]. In particular, these iteration schemes can be represented in one compact formula (with damping factor $\omega \in (0, 1]$ for C-JAC smoother and $\omega = 1$ for G-S smoother)

$$\begin{bmatrix} \delta y^{new} \\ \delta p^{new} \end{bmatrix} = \begin{bmatrix} \delta y \\ \delta p \end{bmatrix} + \omega P_h^{-1} \left(F_h(y_h, p_h) - \begin{bmatrix} B_h & D_h \\ C_h & B_h \end{bmatrix} \begin{bmatrix} \delta y \\ \delta p \end{bmatrix} \right), \tag{2.16}$$

where

$$P_h = \begin{bmatrix} \mathrm{diag}(B_h) & D_h \\ C_h & \mathrm{diag}(B_h) \end{bmatrix} \quad \text{or} \quad P_h = \begin{bmatrix} \mathrm{tril}(B_h) & 0 \\ C_h & \mathrm{tril}(B_h), \end{bmatrix} \tag{2.17}$$

where $\mathrm{diag}(B_h)$ and $\mathrm{tril}(B_h)$ stand for the diagonal and left-lower triangular part of B_h, respectively. Here, the matrix-vector multiplication $P_h^{-1}r$ can be computed very efficiently based on the

well-known partitioned inverse formula

$$\begin{bmatrix} B & D \\ C & B \end{bmatrix}^{-1} = \begin{bmatrix} I & -B^{-1}D \\ 0 & I \end{bmatrix} \begin{bmatrix} B^{-1} & 0 \\ 0 & (B-CB^{-1}D)^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -CB^{-1} & I \end{bmatrix},$$

since these blocks are all diagonal or lower triangular. Although both collective Richardson and collective Gauss-Seidel methods [Takacs and Zulehner, 2011] can be used as well, our numerical experiments show that the C-JAC smoother is the fastest one among all of these approaches. In particular, the C-JAC smoother shows stronger robustness than G-S smoother in handling very small γ, as reported in the following Example 3.

2.4 FAS MULTIGRID METHOD FOR OPTIMALITY SYSTEM

In this section, for the sake of completeness and comparison, we give a brief description of the FAS multigrid method [Borzí and Kunisch, 2005, Borzì, 2007a, Borzì, 2008] for a direct finite difference discretization of the optimality system (2.3) in the form of

$$\begin{cases} -\Delta_h y_h + S(y_h) - u_h = f_h, \\ -\Delta_h p_h + S'(y_h)p_h + y_h = z_h, \\ (\gamma u_h - p_h)\cdot(v_h - u_h) \geq 0 \quad \text{for all} \quad v_h \in U_h^{ad}, \end{cases} \tag{2.18}$$

where the control variable and constraints are also explicitly discretized in a straightforward manner with

$$U_h^{ad} = \{u \in L_h^2(\Omega_h) \mid u_a \leq u \leq u_b \quad \text{in} \quad \Omega_h\}.$$

To illustrate the (projected) collective Gauss-Seidel smoothing scheme [Borzì, 2007a, Borzì, 2008], we rewrite the discretized optimality system (2.18) in its coordinate form at each

$(x_1^i, x_2^j) \in \Omega_h$

$$\frac{1}{h^2}(4y_{i,j} - y_{i-1,j} - y_{i+1,j} - y_{i,j-1} - y_{i,j+1}) + S(y_{i,j}) - u_{i,j} = f_{i.j}, \tag{2.19}$$

$$\frac{1}{h^2}(4p_{i,j} - p_{i-1,j} - p_{i+1,j} - p_{i,j-1} - p_{i,j+1}) + S'(y_{i,j})p_{i,j} + y_{i,j} = z_{i,j}, \tag{2.20}$$

$$(\gamma u_{i,j} - p_{i,j}) \cdot (v_{i,j} - u_{i,j}) \geq 0 \quad \text{for all} \quad v_h \in U_h^{ad}. \tag{2.21}$$

To simplify notation we set

$$Y_{i,j} = f_{i,j} + \tfrac{1}{h^2}(y_{i-1,j} + y_{i+1,j} + y_{i,j-1} + y_{i,j+1})$$

and

$$P_{i,j} = z_{i,j} + \tfrac{1}{h^2}(p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1}).$$

Notice that both $Y_{i,j}$ and $P_{i,j}$ are considered as constants during the update at (x_1^i, x_2^j). The first step is to derive the local Newton update formula for $y_{i,j}$ and $p_{i,j}$ only based on the first two equations

$$\tfrac{4}{h^2}y_{i,j} + S(y_{i,j}) = Y_{i.j} + u_{i,j},$$

$$\tfrac{4}{h^2}p_{i,j} + S'(y_{i,j})p_{i,j} + y_{i,j} = P_{i,j},$$

where $u_{i,j}$ is set to be fixed during the update. It's easy to find the inverse of the Jacobian $J_{i,j}$ of the above two equations as

$$J_{i,j}^{-1} = \begin{bmatrix} \frac{4}{h^2} + S'(y_{i,j}) & 0 \\ S''(y_{i,j})p_{i,j} + 1 & \frac{4}{h^2} + S'(y_{i,j}) \end{bmatrix}^{-1} = \frac{1}{\det(J_{i,j})} \begin{bmatrix} \frac{4}{h^2} + S'(y_{i,j}) & 0 \\ -(S''(y_{i,j})p_{i,j} + 1) & \frac{4}{h^2} + S'(y_{i,j}) \end{bmatrix},$$

where $\det(J_{i,j}) = (\frac{4}{h^2} + S'(y_{i,j}))^2 > 0$ for sufficiently small h. By checking the reduced Hessian [Borzì and Kunisch, 2006], one finds that second-order necessary conditions for a minimum require that $(S''(y_{i,j})p_{i,j} + 1) \geq 0$ (see also assumption (A2)). Hence, the local smooth Newton update

for $y_{i,j}$ and $p_{i,j}$ is given by

$$\begin{bmatrix} \hat{y}_{i,j} \\ \hat{p}_{i,j} \end{bmatrix} = \begin{bmatrix} y_{i,j} \\ p_{i,j} \end{bmatrix} - J_{i,j}^{-1} \begin{bmatrix} e_{i,j}^{y} \\ e_{i,j}^{p} \end{bmatrix},$$

where $e_{i,j}^{y} = \frac{4}{h^2} y_{i,j} + S(y_{i,j}) - Y_{i.j} - u_{i,j}$ and $e_{i,j}^{p} = \frac{4}{h^2} p_{i,j} + S'(y_{i,j}) p_{i,j} + y_{i,j} - P_{i,j}$ denote the corresponding residual. In particular, we notice that $\hat{p}_{i,j}$ depends on $u_{i,j}$ as follows:

$$\begin{aligned} \hat{p}_{i,j}(u_{i,j}) = \; & p_{i,j} - \tfrac{1}{\det(J_{i,j})} \left((\tfrac{4}{h^2} + S'(y_{i,j}))(\tfrac{4}{h^2} p_{i,j} + S'(y_{i,j}) p_{i,j} + y_{i,j} - P_{i,j}) \right) \\ & + \tfrac{1}{\det(J_{i,j})} \left((S''(y_{i,j}) p_{i,j} + 1)(\tfrac{4}{h^2} y_{i,j} + S(y_{i,j}) - Y_{i.j} - u_{i,j}) \right), \end{aligned}$$

which in together with the unconstrained variational inequality ($\gamma \bar{u}_{i,j} = \hat{p}_{i,j}(\bar{u}_{i,j})$) gives the intermediate control update as

$$\begin{aligned} \bar{u}_{i,j} = \; & \left(\gamma + \tfrac{(S''(y_{i,j}) p_{i,j} + 1)}{\det(J_{i,j})} \right)^{-1} \\ & \times \Big[p_{i,j} - \tfrac{1}{\det(J_{i,j})} \left((\tfrac{4}{h^2} + S'(y_{i,j}))(\tfrac{4}{h^2} p_{i,j} + S'(y_{i,j}) p_{i,j} + y_{i,j} - P_{i,j}) \right) \\ & + \tfrac{1}{\det(J_{i,j})} \left((S''(y_{i,j}) p_{i,j} + 1)(\tfrac{4}{h^2} y_{i,j} + S(y_{i,j}) - Y_{i.j}) \right) \Big]. \end{aligned}$$

Finally, the new value of control $u_{i,j}$ is obtained by enforcing the control constraint

$$\hat{u}_{i,j} = \min\{u_a(x_1^i, x_2^j), \max\{u_b(x_1^i, x_2^j), \bar{u}_{i,j}\}\}. \tag{2.22}$$

With this updated $\hat{u}_{i,j}$ we now can update $y_{i,j}$ and $p_{i,j}$ collectively according to (2.4).

To summarize, one local Newton update step at (x_1^i, x_2^j) includes

(1) obtain an auxiliary control variable $\bar{u}_{i,j}$ based on the update formula (2.4);

(2) project $\bar{u}_{i,j}$ onto U_h^{ad} as in (2.22) to get the updated control $\hat{u}_{i,j}$;

(3) perform a local Newton update (2.4) for $y_{i,j}$ and $p_{i,j}$ with new $\hat{u}_{i,j}$.

This whole process implicitly treats the variational inequality through the projection (2.22). Finally, one complete Gauss-Seidel smoothing iteration consists of sweeping (in certain ordering) n^2 local Newton update steps over all grid points in Ω_h. It is called Gauss-Seidel method because the updated nodes will be used once they are computed.

2.5 NUMERICAL EXAMPLES

In this section we numerically verify the mesh-independence convergence of our multigrid accelerated SSN method and compare its computational performance with that of the well-accepted FAS multigrid method. All numerical simulations are implemented using MATLAB on a laptop PC with Intel(R) Core(TM) i3-3120M CPU@2.50GHz and 6GB RAM. The CPU time is estimated by MATLAB's built-in timing functions *tic/toc*, which may be slightly different from other programming languages, but it gives a reliable reference. Let r_y and r_p be the residual of the state and adjoint equation, respectively, i.e.,

$$r_y = -\Delta_h y_h + S(y_h) - \Phi(p_h/\gamma) - f_h \qquad \text{and} \qquad r_p = -\Delta_h p_h + S'(y_h)p_h + y_h - z_h.$$

All unknowns are initialized to be zero and the stopping criterion is chosen as

$$\frac{\|r_y^k\|_2 + \|r_p^k\|_2}{\max(1, \|r_y^0\|_2 + \|r_p^0\|_2)} \leq 10^{-8},$$

where r_y^k and r_p^k denote the residuals at k-th iteration. Here $\|\cdot\|_2$ denotes the standard discrete L^2 norm. To approximately solve the inner semi-smooth Newton system (2.12), we perform only two V-cycle multigrid iterations with zero initial and two pre- and post- smoothing iterations. If the nonlinear term S is very complicated, we may increase the number of inner iterations to recover the mesh-independence convergence. The damping factor is set as $\omega = 2/3$ for C-JAC smoother. The coarsest mesh size is chosen as $h_0 = 1/8$.

Mesh-independence

Example 1 [Hintermüller and Ulbrich, 2004].

Let $S(y) = y^3 + y$, $z = \sin(2\pi x_1)\sin(2\pi x_2)e^{2x_1}/6$, $u_a = -4$, $u_b = 0$, and $f = 0$.

Fig. 2.1 shows the computed optimal control u_h and the corresponding state y_h for $\gamma = 10^{-3}$ with $h = 1/256$, which is indistinguishable from the plot in [Hintermüller and Ulbrich, 2004] to the human eye.

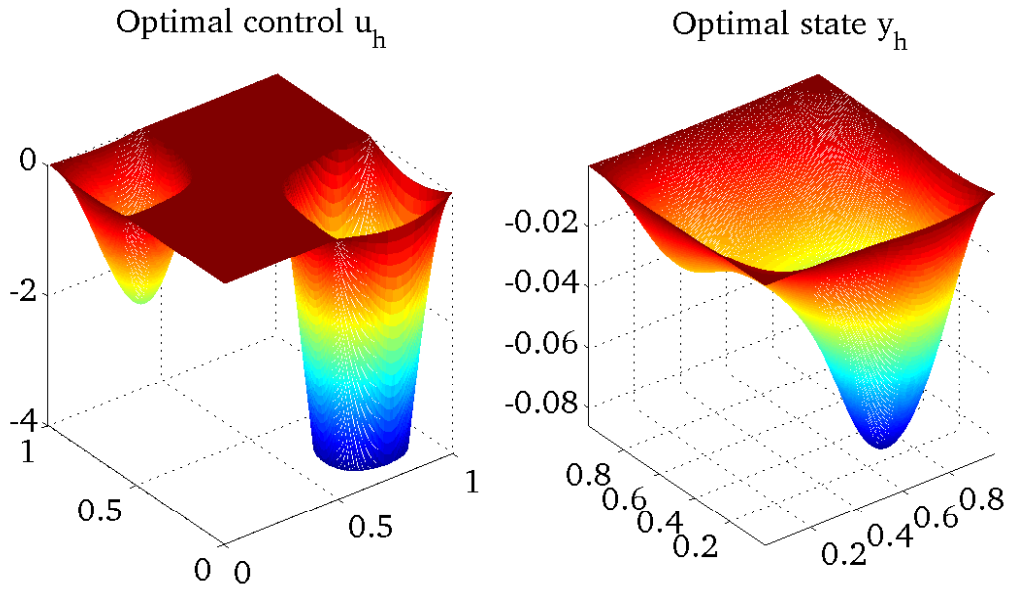

Figure 2.1. Computed optimal control and optimal state of Ex. 1 with $\gamma = 10^{-3}$ for $h = 1/256$

In Table 2.1, we provide the required numbers (column 'Iter') and the corresponding CPU time in seconds (column 'CPU') of the SSN multigrid iterations (using G-S smoother) for different levels of mesh size h with different γ. We clearly observe the mesh-independent convergence of our proposed SSN-MG method. More importantly, the SSN-MG algorithm numerically achieves the optimal $O(N)$ linear complexity since its CPU time does present a roughly fourfold increase as the number of unknowns quadruples from one level to the next. As what we have anticipated, if we just solve the semi-smooth Newton system (2.11) by MATLAB's backslash('\') sparse direct solver, the corresponding CPU time, as stated in Table 2.2, turns out to be much slower than SSN-MG method although it does achieve slightly better mesh-independent convergence rate and accuracy. The cruciality of the coarsen strategy (2.15) is illustrated in Table 2.3, where we

implement our SSN-MG method using (2.14) instead of (2.15). It's convergence rate is obviously deteriorated for $\gamma = 10^{-6}$.

Moreover, the required numbers of SSN-MG iterations are very robust with respect to the regularization parameter γ (only 2 more iterations for $\gamma = 10^{-6}$ compared with $\gamma = 10^{-3}$). A slight increase in 'Iter' is reasonable since the problem becomes more ill-conditioned as γ becomes smaller. However, even for extremely small $\gamma(< 10^{-6})$, we suggest to handle it via an extrapolation based continuation technique proposed in [Hintermüller et al., 2008]. Its basic idea is, for each fixed h, to derive a better initial guess for smaller γ by performing a linear extrapolation using already computed solutions for larger γ's.

Table 2.1. Results of SSN-MG method (V-cycles) for Ex. 1.

	$\gamma = 10^{-3}$				$\gamma = 10^{-6}$			
h	$\|r_y\|_2$	$\|r_p\|_2$	Iter	CPU	$\|r_y\|_2$	$\|r_p\|_2$	Iter	CPU
1/64	1.42e-09	1.22e-11	6	0.15	2.83e-09	7.89e-12	14	0.31
1/128	2.27e-09	1.63e-11	6	0.38	1.28e-09	1.72e-12	9	0.56
1/256	2.81e-09	1.68e-11	6	1.20	8.47e-10	1.21e-12	9	1.87
1/512	3.42e-09	1.67e-11	6	5.24	2.96e-09	3.74e-12	8	6.40
1/1024	4.02e-09	1.70e-11	6	21.43	5.21e-10	1.16e-12	8	27.47
1/2048	4.41e-09	1.75e-11	6	82.40	5.92e-10	4.52e-12	8	124.22

Table 2.2. Results of MATLAB's backslash direct solver for Ex. 1.

	$\gamma = 10^{-3}$				$\gamma = 10^{-6}$			
h	$\|r_y\|_2$	$\|r_p\|_2$	Iter	CPU	$\|r_y\|_2$	$\|r_p\|_2$	Iter	CPU
1/64	2.40e-10	2.93e-11	4	0.22	3.66e-11	1.12e-12	13	0.77
1/128	1.41e-13	1.65e-14	5	1.27	1.99e-13	1.77e-14	10	2.82
1/256	5.67e-13	6.60e-14	5	6.39	7.65e-13	7.04e-14	7	9.51
1/512	2.25e-12	2.67e-13	5	33.61	3.08e-12	2.83e-13	7	49.07
1/1024	9.02e-12	1.06e-12	5	246.26	1.23e-11	1.13e-12	7	350.64

Table 2.3. Results of SSN-MG method (V-cycles with $\tilde{D}_H$) for Ex. 1.

	$\gamma = 10^{-3}$				$\gamma = 10^{-6}$			
h	$\|r_y\|_2$	$\|r_p\|_2$	Iter	CPU	$\|r_y\|_2$	$\|r_p\|_2$	Iter	CPU
1/64	1.34e-09	1.30e-11	6	0.14			>50	
1/128	1.91e-09	1.35e-11	6	0.38			>50	
1/256	2.46e-09	1.39e-11	6	1.57			>50	
1/512	3.15e-09	1.40e-11	6	5.45	5.73e-09	3.56e-13	36	29.79
1/1024	3.78e-09	1.41e-11	6	21.57	8.62e-09	1.17e-12	36	135.23
1/2048	4.21e-09	1.47e-11	6	87.45	8.02e-09	4.52e-12	35	517.37

Comparison with FAS multigrid method

In this subsection, we compare our proposed SSN-MG method with the FAS multigrid method given in [Borzí and Kunisch, 2005, Borzì, 2008, Borzì and Schulz, 2009], where the authors suggested to use the W-cycle with two pre- and post-smoothing iterations. We will use 10 Gauss-Seidel smoothing iterations as the direct solver on the coarsest level. By comparing both methods (with the same stop criterion), we continue using zero initials since it's more clear to see a mesh-independent convergence rate and hence judge their computational efficiency more fairly.

Example 2 [Borzì, 2008].

Let $S(y) = y^4$, $z = \sin(2\pi x_1)\sin(3\pi x_2)$, $u_a = -1/2$, $u_b = 1/2$, and $f = 0$.

In Table 2.4 and 2.5, we compare the convergence results and computational time of both our proposed SSN-MG method (C-JAC and G-S smoother) with the FAS-MG method (V-cycles and W-cycles) for Ex.1 and Ex. 2 with $\gamma = 10^{-4}$, respectively. It evidently shows that the FAS-MG method with W-cycles is faster and more robust than with V-cycles. However, there is no advantage over V-cycles to choose more expensive W-cycles in our SSN-MG method, since the robust mesh-independence convergence mainly comes from SSN outer iterations, which will not be obviously influenced by how the inner multigrid solver is implemented. Although both of them achieve a very favorable mesh-independent convergence, our proposed SSN-MG method costs significantly less CPU time than the FAS-MG method does. For example, in Table 2.5 with $h = 1/1024$, our proposed SSN-MG method (for both smoothers) only need less than half of the CPU time taken by the FAS-MG method (with W-cycles). We also find that the standard G-S smoother is obviously faster than the C-JAC smoother for $\gamma = 10^{-4}$.

Table 2.4. Results of FAS-MG and SSN-MG method for Ex. 1 ($\gamma = 10^{-4}$).

	FAS-MG Method				SSN-MG Method (V-cycles)			
	V-cycle		W-cycle		C-JAC smoother		G-S smoother	
h	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
1/64	13	0.32	8	0.31	9	0.22	7	0.16
1/128	13	1.10	7	0.97	9	0.57	7	0.42
1/256	13	4.40	6	3.09	9	1.96	7	1.43
1/512	13	17.98	6	12.67	9	7.32	7	5.56
1/1024	13	73.34	6	52.30	9	28.89	7	23.25

Table 2.5. Results of FAS-MG and SSN-MG method for Ex. 2 ($\gamma = 10^{-4}$).

	FAS-MG Method				SSN-MG Method (V-cycles)			
	V-cycle		W-cycle		C-JAC smoother		G-S smoother	
h	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
1/64	35	0.90	11	0.53	8	0.20	6	0.15
1/128	43	3.97	8	1.37	8	0.52	6	0.41
1/256	46	16.92	6	3.73	8	1.87	6	1.32
1/512	45	69.91	6	15.04	8	7.14	6	5.30
1/1024	45	333.03	6	59.34	8	26.62	6	20.68

Example 3 [Hintermüller et al., 2008].

Let $S(y) = y^3 + y + e^{10y}$, $z = \cos(\pi x_1)\cos(\pi x_2)e^{x_1}/2$, $u_a = -8$, $u_b = 4$, and $f = 0$.

Our last Example 3 is slightly modified from the Example 1 in [Hintermüller et al., 2008], where the original problem is mixed control-state constrained. Fig. 2.2 shows the computed optimal control u_h and the corresponding optimal state y_h for $\gamma = 10^{-4}$ with $h = 1/1024$, which displays some similar characteristics as the plots reported in [Hintermüller et al., 2008]. Again, the corresponding computational results as given in Table 2.6 and 2.7 demonstrate the better performance for our proposed SSN-MG method. Interestingly, the FAS-MG method fails to converge within 50 iterations and the C-JAC smoother becomes faster than the G-S smoother when $\gamma = 10^{-6}$. However, in the case of $\gamma = 10^{-6}$, the extrapolation technique [Hintermüller et al., 2008] should be adopted.

Table 2.6. Results of FAS-MG and SSN-MG method for Ex. 3 ($\gamma = 10^{-4}$).

	FAS-MG Method				SSN-MG Method (V-cycles)			
	V-cycle		W-cycle		C-JAC smoother		G-S smoother	
h	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
1/64	24	0.63	8	0.36	10	0.24	8	0.19
1/128	25	2.41	7	1.10	10	0.63	8	0.52
1/256	25	9.18	6	3.51	10	2.15	8	1.87
1/512	25	38.31	6	13.60	10	7.99	8	6.51
1/1024	25	158.45	6	56.32	10	32.24	8	27.20

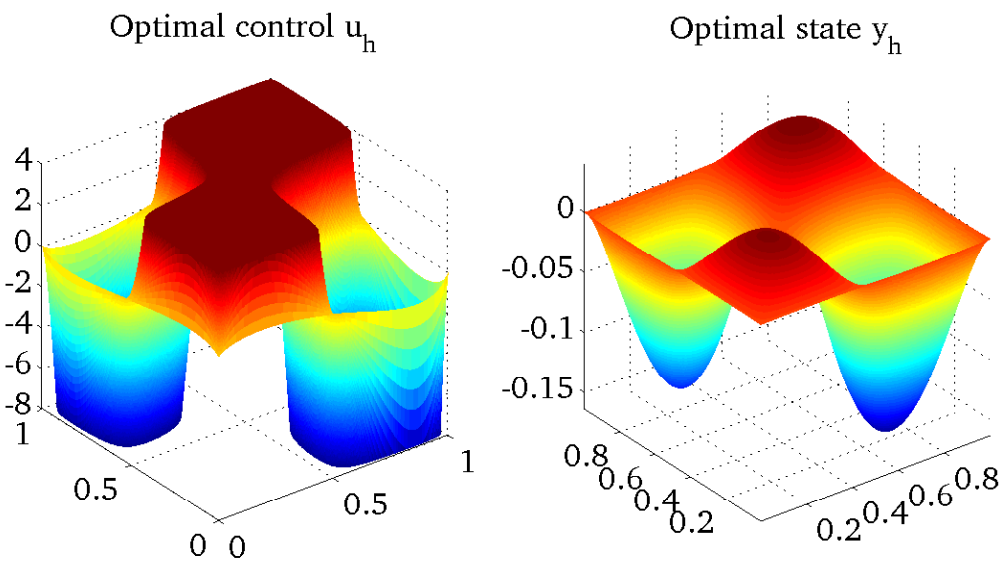

Figure 2.2. Computed optimal control and optimal state of Ex. 3 with $\gamma = 10^{-4}$ for $h = 1/1024$

Table 2.7. Results of FAS-MG and SSN-MG method for Ex. 3 ($\gamma = 10^{-6}$).

	FAS-MG Method				SSN-MG Method (V-cycles)			
	V-cycle		W-cycle		C-JAC smoother		G-S smoother	
h	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
1/64	>50	1.20	>50	2.00	20	0.73	27	0.59
1/128	>50	4.70	>50	7.24	16	1.45	31	2.68
1/256	>50	18.74	>50	27.54	16	4.09	32	8.44
1/512	>50	76.64	>50	112.96	16	13.70	32	29.26
1/1024	>50	321.70	>50	468.32	16	57.71	32	102.87

2.6 CONCLUSIONS

We propose a new multigrid approach to implement the semi-smooth Newton method, which works very well for a class of semi-linear elliptic optimal control problems with control-constraints comparing with current available approaches in literature in terms of computational efficiency. Numerical results show that our proposed semi-smooth Newton multigrid method outperforms the currently widely used FAS multigrid method, attributed to the proposed new coarsening strategy and computationally more efficient smoothers.

CHAPTER 3

A NEW LEAPFROG MULTIGRID METHOD FOR LINEAR PARABOLIC CONTROL PROBLEMS WITHOUT CONTROL CONSTRAINTS

3.1 INTRODUCTION

In this chapter, we exemplify our proposed approach through discussing a typical 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 of minimizing a tracking-type quadratic cost functional

$$J(y,u) = \frac{1}{2}\|y-g\|^2_{L^2(Q)} + \frac{\gamma}{2}\|u\|^2_{L^2(Q)} \tag{3.1}$$

subject to a linear parabolic PDE system

$$\begin{cases} -\partial_t y + \Delta y = f + u \quad \text{in} \quad Q, \\ y = 0 \quad \text{on} \quad \Sigma, \\ y(\cdot,0) = y_0 \quad \text{in} \quad \Omega, \end{cases} \tag{3.2}$$

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 $y_0 \in H^1_0(\Omega)$. The existence and uniqueness of solution to the above optimal control problem (3.1-3.2) is well understood (see, e.g., [Lions, 1971, Tröltzsch, 2010]).

By defining an appropriate Lagrange functional, making use of the strong convexity of the original optimization problem, the optimal solution pair (y,u) to (3.1-3.2) is shown to be

completely characterized by the unique solution triplet (y, p, u) to the following optimality system

$$\begin{cases} -\partial_t y + \Delta y - u = f \quad \text{in} \quad Q, \\ y = 0 \quad \text{on} \quad \Sigma, \quad y(\cdot, 0) = y_0 \quad \text{in} \quad \Omega, \\ \partial_t p + \Delta p + y = g \quad \text{in} \quad Q, \\ p = 0 \quad \text{on} \quad \Sigma, \quad p(\cdot, T) = 0 \quad \text{in} \quad \Omega, \\ \gamma u - p = 0 \quad \text{in} \quad Q, \end{cases} \tag{3.3}$$

where the state y evolves forward in time and the adjoint state p marches backward in time. According to [Evans, 2010], 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. This would not be the case if there are some boxed constraints on the control u, which will be discussed in the next chapter. For the purpose of simplified analysis and practical implementation, the control $u = p/\gamma$ can be eliminated from the optimality system as following

$$\begin{cases} -\partial_t y + \Delta y - p/\gamma = f \quad \text{in} \quad Q, \\ y = 0 \quad \text{on} \quad \Sigma, \quad y(\cdot, 0) = y_0 \quad \text{in} \quad \Omega, \\ \partial_t p + \Delta p + y = g \quad \text{in} \quad Q, \\ p = 0 \quad \text{on} \quad \Sigma, \quad p(\cdot, T) = 0 \quad \text{in} \quad \Omega. \end{cases} \tag{3.4}$$

This is a standard two-point boundary-value problem (with respect to t) appeared in optimal control of parabolic PDEs. It is well-known that the main challenge for solving (3.4) results from the fact that the state y and the adjoint state p are marching in opposite orientations. Its numerical discretizations will create an enormously huge system of algebraic equations as we have to resolve all time steps simultaneously [Heinkenschloss, 2005].

In terms of finite difference discretization for time variable of (3.4), the backward Euler discretization with respect to time t is a favorable choice due to its unconditional stability (see,

e.g.,[Borzì, 2003]). 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 effort is devoted by many researchers to explore various second or higher-order numerical schemes for (3.4) or similar-type systems. In [Apel and Flaig, 2012], the authors introduced a family of second-order Crank-Nicolson 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 [González Andrade and Borzì, 2012], the authors developed a second-order backward time differentiation formula (BDF2) in time with Crank-Nicolson 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 Crank-Nicolson 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 [Neitzel and Tröltzsch, 2009, Neitzel et al., 2011] 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 [Meidner and Vexler, 2008a, Meidner and Vexler, 2008b, Meidner and Vexler, 2011, Gong et al., 2012]. In addition to the above mentioned schemes, many other discretization strategies in time and space have been extensively studied [Liu et al., 2004, Chrysafinos, 2010].

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 [González Andrade and Borzì, 2012, Borzì and González Andrade, 2014], the pure Crank-Nicolson 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, numerical

experiences show that some multigrid solvers, including the one we present in this work, may not even converge with the Crank-Nicolson scheme, and thus this simulates us to seek more suitable schemes for 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 [Rees et al., 2010, Herzog and Sachs, 2010, Pearson et al., 2012, Pearson and Stoll, 2013] so that it can deal with large-scale degrees of freedom and higher dimension efficiently. Beginning with a few early numerical endeavors [Hackbusch, 1978, Hackbusch, 1979, Hackbusch, 1981], multigrid methods have started to play a more and more irreplaceable character in the field of PDE optimization [Briggs et al., 2000, Trottenberg et al., 2001, Saad, 2003, Borzì and Schulz, 2012, Hinze et al., 2012, Liu and Xiao, 2014a] since the seminal introduction of space-time multigrid for linear parabolic PDEs [Horton and Vandewalle, 1995], where the semi-coarsening was shown to give better convergence compared to standard coarsening. In the framework of finite difference discretization, some recent papers [Borzì, 2003, Borzì and Griesse, 2005, Borzì and Griesse, 2006, Borzì, 2007b, Borzì and Schulz, 2009, Borzì and von Winckel, 2009, Borzì and González Andrade, 2012, González Andrade and Borzì, 2012] are devoted to apply the idea of space-time multigrid to those forward-and-backward coupled linear/nonlinear parabolic PDE systems similar to (3.4). But little research is seen between the connection of numerical scheme design and fast solver implementation.

In this chapter we propose a new leapfrog central difference scheme for time discretization. In classical theory, it is well-known that the leapfrog scheme is not stable for a single evolutionary equation although it is second-order accurate [Strikwerda, 2004, LeVeque, 2007]. However, in this work, we prove that our new leapfrog scheme in terms of time discretization for the two point boundary-value problem (3.4) is unconditionally stable and delivers the second-order accuracy of time variable, which has not been seen in current literature. Our method for treating parabolic PDEs can be regarded as a generalization of the boundary value methods [Axelsson and Verwer, 1985, Brugnano and Trigiante, 1998], which are orignally developed only for solving ordinary differential equations (ODEs). The essential observation is that the

conventional instability of the 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 of using the new 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 [Borzì and Schulz, 2009], as was shown in [Lass et al., 2009, Takacs and Zulehner, 2011] 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.

This chapter is organized as follows. In the next section, we propose the 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 3.3, a multigrid algorithm is designed for solving the discretized optimality system with some favorable structures. In Section 3.4, 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 Crank-Nicolson scheme, and our leapfrog scheme. Finally, the chapter ends with concluding remarks in Section 3.5.

3.2 A LEAPFROG SCHEME AND ITS ERROR ESTIMATE

In this section, we conduct 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 $h = \max(h_1, h_2)$. 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

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

We shall use the discrete version of Poincare inequality and Sobolev embedding inequality [Knabner and Angermann, 2003, Jovanović and Süli, 2014], 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

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

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 (3.4) by the leap-frog finite difference scheme

$$\frac{Y^{n+1} - Y^{n-1}}{2\tau} - \Delta_h Y^n + P^n/\gamma = -f^n, \quad n = 1, 2, \cdots, N-1 \tag{3.5}$$

$$\frac{P^{n+1} - P^{n-1}}{2\tau} + \Delta_h P^n + Y^n = g^n, \quad n = 1, 2, \cdots, N-1 \tag{3.6}$$

where $Y^n = (Y_{ij}^n)_{i=1,j=1}^{M_1-1,M_2-1}$ and $P^n = (P_{ij}^n)_{i=1,j=1}^{M_1-1,M_2-1}$ with Y_{ij}^n and P_{ij}^n being the discrete ap-

proximation 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

$$\frac{Y^N - Y^{N-1}}{\tau} - \Delta_h Y^N + P^N/\gamma = -f^N, \tag{3.7}$$

$$\frac{P^1 - P^0}{\tau} + \Delta_h P^0 + Y^0 = g^0. \tag{3.8}$$

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 $\{Y^n, P^n\}_{n=0}^N$ have 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 Crank-Nicolson schemes are also applicable to replace the above backward Euler scheme, and numerical comparisons will be provided in last section.

Theorem 3.2.1. *Let the dimension $d = 2$. Assume $f, g \in C^{4,3}(\overline{Q})$ and the solution $y, p \in C^{4,3}(\overline{Q})$, then the linear system defined by* (3.5)-(3.8) *is invertible and the scheme has a second-order accuracy in discrete $L^2(L^\infty)$ norm, i.e.,*

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

for some positive constant $C(T, \gamma)$ 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

$$\frac{y^{n+1} - y^{n-1}}{2\tau} - \Delta_h y^n + p^n/\gamma = -f^n - F^n, \quad n = 1, 2, \cdots, N-1 \tag{3.9}$$

$$\frac{p^{n+1}-p^{n-1}}{2\tau}+\Delta_h p^n+y^n=g^n-G^n,\quad n=1,2,\cdots,N-1 \tag{3.10}$$

and

$$\frac{y^N-y^{N-1}}{\tau}-\Delta_h y^N+p^N/\gamma=-f^N-F^N, \tag{3.11}$$

$$\frac{p^1-p^0}{\tau}+\Delta_h p^0+y^0=g^0-G^0, \tag{3.12}$$

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

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

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

$$\|\nabla_h\overline{F}^N\|+\|\nabla_h\overline{G}^0\|\le C_1\tau,$$

$$\|\widetilde{F}^N\|+\|\widetilde{G}^0\|\le C_1h^2,$$

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. Let

$e^n = Y^n - y^n$ and $\eta^n = P^n - p^n$. Then the difference between (3.5)-(3.8) and (3.9)-(3.12) gives

$$\frac{e^{n+1} - e^{n-1}}{2\tau} - \Delta_h e^n + \eta^n/\gamma = F^n, \tag{3.13}$$

$$\frac{\eta^{n+1} - \eta^{n-1}}{2\tau} + \Delta_h \eta^n + e^n = G^n, \tag{3.14}$$

and

$$\frac{e^N - e^{N-1}}{\tau} - \Delta_h e^N + \eta^N/\gamma = F^N, \tag{3.15}$$

$$\frac{\eta^1 - \eta^0}{\tau} + \Delta_h \eta^0 + e^0 = G^0, \tag{3.16}$$

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

The discrete inner product of (3.13) and $-\tau\,\Delta_h e^n$ is

$$\frac{(\nabla_h e^{n+1}, \nabla_h e^n) - (\nabla_h e^n, \nabla_h e^{n-1})}{2} + \tau\|\Delta_h e^n\|^2 + \tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma = -\tau(F^n, \Delta_h e^n), \tag{3.17}$$

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

$$\frac{(\nabla_h e^N, \nabla_h e^{N-1})}{2} + \sum_{n=1}^{N-1} \tau\|\Delta_h e^n\|^2 + \sum_{n=1}^{N-1} \tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma = -\sum_{n=1}^{N-1} \tau(F^n, \Delta_h e^n). \tag{3.18}$$

The discrete inner product of (3.15) and $-\tau\,\Delta_h e^N/2$ is (note that $\eta^N = 0$)

$$\begin{aligned}\frac{\|\nabla_h e^N\|^2 - (\nabla_h e^N, \nabla_h e^{N-1})}{2} + \frac{\tau}{2}\|\Delta_h e^N\|^2 &= -\frac{\tau}{2}(F^N, \Delta_h e^N) \\ &= \frac{\tau}{2}(\nabla_h \overline{F}^N, \nabla_h e^N) + \frac{\tau}{2}(\widetilde{F}^N, \Delta_h e^N).\end{aligned} \tag{3.19}$$

The sum of the last two equations gives

(by Cauchy's inequality with ϵ [Evans, 2010]: $ab \le a^2/(4\epsilon) + \epsilon b^2$ for $a > 0, b > 0, \epsilon > 0$)

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

which leads to (after moving the last three terms to the left-hand-side)

$$
\begin{aligned}
&(\frac{1}{2} - \epsilon)\|\nabla_h e^N\|^2 + (1 - \epsilon) \sum_{n=1}^{N-1} \tau \|\Delta_h e^n\|^2 + (\frac{1}{2} - \epsilon)\tau\|\Delta_h e^N\|^2 + \sum_{n=1}^{N-1} \tau(\nabla_h e^n, \nabla_h \eta^n)/\gamma \qquad (3.20) \\
&\le C_1^2((T + \frac{1}{8})\tau^4 + Th^4 + \frac{1}{8}\tau h^4)/(2\epsilon) \qquad (3.21)
\end{aligned}
$$

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)

$$\sum_{n=1}^{N} \tau \|\Delta_h e^n\|^2 + \frac{4}{\gamma} \sum_{n=1}^{N-1} \tau (\nabla_h e^n, \nabla_h \eta^n) \le C_2(\tau^4 + h^4). \tag{3.22}$$

Following the above analogous arguments, the discrete inner product of (3.14) and $\tau\, \Delta_h \eta^n$ is

$$-\frac{(\nabla_h \eta^{n+1}, \nabla_h \eta^n) - (\nabla_h \eta^{n-1}, \nabla_h \eta^n)}{2} + \tau \|\Delta_h \eta^n\|^2 - \tau (\nabla_h e^n, \nabla_h \eta^n) = \tau (G^n, \Delta_h \eta^n), \tag{3.23}$$

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

$$\frac{(\nabla_h \eta^0, \nabla_h \eta^1)}{2} + \sum_{n=1}^{N-1} \tau \|\Delta_h \eta^n\|^2 - \sum_{n=1}^{N-1} \tau (\nabla_h e^n, \nabla_h \eta^n) = \sum_{n=1}^{N-1} \tau (G^n, \Delta_h \eta^n). \tag{3.24}$$

The discrete inner product of (3.16) and $\tau\, \Delta_h \eta^0/2$ is (note that $e^0 = 0$)

$$\begin{aligned} -\frac{(\nabla_h \eta^1, \nabla_h \eta^0) - (\nabla_h \eta^0, \nabla_h \eta^0)}{2} + \frac{\tau}{2} \|\Delta_h \eta^0\|^2 &= \frac{\tau}{2} (G^0, \Delta_h \eta^0) \\ &= -\frac{\tau}{2} (\nabla_h \overline{G}^0, \nabla_h \eta^0) + \frac{\tau}{2} (\widetilde{G}^0, \Delta_h \eta^0). \end{aligned} \tag{3.25}$$

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

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

$$\leq \frac{C_1^2}{4}(\tau^4+\tau h^4)/(4\epsilon)+\epsilon\|\nabla_h\eta^0\|^2+\epsilon\tau\|\Delta_h\eta^0\|^2+C_1^2(\tau^2+h^2)^2/(4\epsilon)\sum_{n=1}^{N-1}\tau+\epsilon\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2$$
$$\leq \frac{C_1^2}{4}(\tau^4+\tau h^4)/(4\epsilon)+\epsilon\|\nabla_h\eta^0\|^2+\epsilon\tau\|\Delta_h\eta^0\|^2+2TC_1^2(\tau^4+h^4)/(4\epsilon)+\epsilon\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2$$
$$= C_1^2((T+\frac{1}{8})\tau^4+Th^4+\frac{1}{8}\tau h^4)/(2\epsilon)+\epsilon\|\nabla_h\eta^0\|^2+\epsilon\tau\|\Delta_h\eta^0\|^2+\epsilon\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2,$$

that is

$$(\frac{1}{2}-\epsilon)\|\nabla_h\eta^0\|^2+(\frac{1}{2}-\epsilon)\tau\|\Delta_h\eta^0\|^2+(1-\epsilon)\sum_{n=1}^{N-1}\tau\|\Delta_h\eta^n\|^2-\sum_{n=1}^{N-1}\tau(\nabla_h e^n,\nabla_h\eta^n) \tag{3.26}$$
$$\leq C_1^2((T+\frac{1}{8})\tau^4+Th^4+\frac{1}{8}\tau h^4)/(2\epsilon) \tag{3.27}$$

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

$$\sum_{n=0}^{N-1}\tau\|\Delta_h\eta^n\|^2-4\sum_{n=1}^{N-1}\tau(\nabla_h e^n,\nabla_h\eta^n)\leq C_3(\tau^4+h^4). \tag{3.28}$$

Adding $\gamma\times$(3.22) with (3.28) gives

$$\gamma\sum_{n=1}^{N}\tau\|\Delta_h e^n\|^2+\sum_{n=0}^{N-1}\tau\|\Delta_h\eta^n\|^2\leq(\gamma C_2+C_3)(\tau^4+h^4), \tag{3.29}$$

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

$$\sum_{n=0}^{N} \tau \|\Delta_h e^n\|^2 + \sum_{n=0}^{N} \tau \|\Delta_h \eta^n\|^2 \leq C_4(\tau^4 + h^4). \tag{3.30}$$

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

$$\sum_{n=0}^{N} (\max_{i,j} |e_{ij}^n|^2 + \max_{i,j} |\eta_{ij}^n|^2)\tau \leq C_5(\tau^4 + h^4), \tag{3.31}$$

which completes the proof.

From the proof we can also see that, if we set $F^n = G^n = 0$ in (3.11)-(3.14), then (3.11)-(3.14) imply that $e^n = \eta^n = 0$. This substantiates the invertibility of the discretized linear system (3.5)-(3.8) in our conclusions. □

Remark 1. 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 [Borzì, 2003, González Andrade and Borzì, 2012]. The approach technique for the L^∞ norm in space also holds for dimensions $d = 1$ and $d = 3$. With $d > 3$ we are not able to reach (3.31) from (3.30) due to the failure of the discrete Sobolev embedding inequality. However, we still can obtain a similar error estimate in discrete $L^2(Q)$ norm directly from (3.30).

3.3 MULTIGRID METHOD FOR LINEAR SYSTEM

To illustrate our multigrid linear solver for the discretized system, we reformulate our leapfrog scheme (3.5,3.6,3.7,3.8) 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

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

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

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

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

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 [Liu and Xiao, 2014b] for solving (3.32). 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 [Takacs and Zulehner, 2011]. 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 diag$(B_h) = B_h$ and 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 [Horn and Johnson, 2013] 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 the above smoother is the invertibility of J_h, which trivially holds for our proposed leapfrog scheme.

3.4 NUMERICAL EXAMPLES

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 Crank-Nicolson 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 is chosen to be

$$\frac{\sqrt{\|r_y^{(k)}\|^2 + \|r_p^{(k)}\|^2}}{\sqrt{\|r_y^{(0)}\|^2 + \|r_p^{(0)}\|^2}} \leq 10^{-7},$$

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

Example 4.

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 (3.4) 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 3.1 and 3.2 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 shows that our multigrid solver is quite robust with respect to the regularization parameter γ, which is very attractive to those practical applications with a possible large range of regularization parameters.

Table 3.1. Results for Ex. 4 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 3.2. Results for Ex. 4 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

As the first comparison, we report in Table 3.3 and Table 3.4 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 fair comparison between them using the same multigrid solver. Comparing Tables 3.1 and 3.2 with Tables 3.3 and 3.4, 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.

Table 3.3. Results for Ex. 4 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 3.4. Results for Ex. 4 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 3.5. Results for Ex. 4 with the Crank-Nicolson 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

For a further comparison, we also report in Table 3.5 and Table 3.6 the corresponding results of the Crank-Nicolson scheme, which is anticipated to be problematic for our multigrid solver

Table 3.6. Results for Ex. 4 with the Crank-Nicolson 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

framework. Although the Crank-Nicolson 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 Crank-Nicolson scheme. Therefore, as also mentioned in [González Andrade and Borzì, 2012], the Crank-Nicolson scheme is not recommended when solving the underlying problem with the standard multigrid algorithm implementations. In summary, our proposed leapfrog scheme demonstrates the desired advantage in both provable second-order accuracy and fast iterative linear solver.

Example 5.

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

$$f = \big(\pi \sin(\pi t) - 2\pi^2 \cos(\pi t) - \gamma^{-1} \sin(\pi t)\big) \sin(\pi x_1) \sin(\pi x_2)$$

and

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

in (3.4) such that the exact solution is

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

and

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

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 $\gamma = 10^{-4}$.

Table 3.7. Results for Ex. 5 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)	9.25e-02	–	1.56e-02	–	7	0.02
(16,16,16)	2.42e-02	1.9	3.87e-03	2.0	8	0.04
(32,32,32)	6.17e-03	2.0	9.63e-04	2.0	8	0.16
(64,64,64)	1.55e-03	2.0	2.40e-04	2.0	8	1.10
(128,128,128)	3.86e-04	2.0	5.99e-05	2.0	8	8.91
(256,256,256)	9.64e-05	2.0	1.50e-05	2.0	8	90.46

Table 3.8. Results for Ex. 5 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)	3.57e-01	–	6.70e-03	–	7	0.02
(16,16,16)	9.40e-02	1.9	1.05e-03	2.7	7	0.03
(32,32,32)	2.40e-02	2.0	1.52e-04	2.8	7	0.15
(64,64,64)	6.08e-03	2.0	2.19e-05	2.8	7	0.97
(128,128,128)	1.53e-03	2.0	3.74e-06	2.5	7	7.79
(256,256,256)	3.79e-04	2.0	8.01e-07	2.2	7	80.16

Table 3.9. Results for Ex. 5 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)	9.41e-02	–	1.89e-02	–	7	0.03
(16,16,16)	2.82e-02	1.7	5.81e-03	1.7	8	0.04
(32,32,32)	7.25e-03	2.0	1.60e-03	1.9	8	0.17
(64,64,64)	1.83e-03	2.0	4.18e-04	1.9	8	1.12
(128,128,128)	4.61e-04	2.0	1.07e-04	2.0	9	9.96
(256,256,256)	1.16e-04	2.0	2.69e-05	2.0	10	114.02

Table 3.10. Results for Ex. 5 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)	4.87e-01	–	1.06e-02	–	7	0.02
(16,16,16)	1.47e-01	1.7	1.56e-03	2.8	7	0.05
(32,32,32)	3.88e-02	1.9	1.98e-04	3.0	7	0.15
(64,64,64)	9.98e-03	2.0	2.96e-05	2.7	8	1.14
(128,128,128)	2.53e-03	2.0	6.12e-06	2.3	8	9.14
(256,256,256)	6.39e-04	2.0	1.46e-06	2.1	9	104.21

We give in Tables 3.7-3.8 and Tables 3.9-3.10 the corresponding numerical results of our proposed leapfrog scheme and the BDF2 scheme with different parameters, respectively. Similar

to Example 4, we observe a satisfactory second-order accuracy for our leapfrog scheme as the mesh is refined. Also, our leapfrog scheme produces more accurate approximations than the BDF2 scheme with the same mesh size. Moreover, our multigrid solver achieves a desired mesh-independent convergence for our leapfrog scheme. However, the performance of our multigrid solver with the BDF2 scheme is getting a little bit worse when the mesh is refined, as shown in columns 'Iter' of Tables 3.9-3.10. Thus one can also see from Example 5 that our leapfrog scheme exceeds the BDF2 scheme in providing better efficiency and higher accuracy.

Finally, we choose not to present the corresponding numerical results of the Crank-Nicolson scheme since the our multigrid solver is not suitable for this example by using the Crank-Nicolson scheme. In this case, the Crank-Nicolson scheme could still be realized by the backslash sparse direct solver within MATLAB, of which the CPU time will soar up very fast as the mesh is refined. For example, it takes about 35 seconds for just a $32 \times 32 \times 32$ mesh. Hence it would be misleading to compare its computational CPU time with our multigrid iterative solver since the sparse direct solver is based on a completely different philosophy. How to come up with an efficient iterative solver for the Crank-Nicolson scheme is a quiet open problem deserving further investigations, which is beyond the scope of our current work.

3.5 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 an undesirable structure, such as the classical Crank–Nicolson scheme. Thus 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 chapter, 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 solver under the multigrid framework. According to the proof presented in this chapter, the proposed approach is also applicable to the semi-linear parabolic cases, whose results will be reported in the next chapter.

APPENDIX: A BDF2 SCHEME WITH A CRANK–NICOLSON INITIALIZATION STEP.

For the purpose of completeness, the BDF2 scheme with a Crank–Nicolson initialization step [González Andrade and Borzì, 2012] for the system (3.4) using the same notation is included below. As we discussed in the introduction, the Crank-Nicolson 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 (3.38,3.39,3.40,3.41) has more complicated structure 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 $2 \leq n \leq N$, the state equation is discretized as

$$\frac{3Y^n - 4Y^{n-1} + Y^{n-2}}{2\tau} - \Delta_h Y^n + \frac{P^n}{\gamma} = -f^n. \tag{3.38}$$

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

$$-\frac{3P^n - 4P^{n+1} + P^{n+2}}{2\tau} + \Delta_h P^n + Y^n = g^n. \tag{3.39}$$

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

$$\frac{Y^n - Y^{n-1}}{\tau} - \frac{\Delta_h Y^n + \Delta_h Y^{n-1}}{2} + \frac{P^n + P^{n-1}}{2\gamma} = -\frac{f^n + f^{n-1}}{2} \tag{3.40}$$

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

$$\frac{P^n - P^{n-1}}{\tau} + \frac{\Delta_h P^n + \Delta_h P^{n-1}}{2} + \frac{Y^n + Y^{n-1}}{2} = \frac{g^n + g^{n-1}}{2} \tag{3.41}$$

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

CHAPTER 4

A LEAPFROG SSN-MULTIGRID METHOD FOR SEMILINEAR PARABOLIC CONTROL PROBLEMS WITH CONTROL CONSTRAINTS

4.1 INTRODUCTION

In this chapter, we consider the following standard distributed optimal control problem [Borzì and González Andrade, 2012] of minimizing a quadratic cost functional

$$\begin{cases} J(y,u) = \frac{\alpha}{2}\|y-z\|^2_{L^2_Q} + \frac{\beta}{2}\|y(\cdot,T)-z_T\|^2_{L^2_\Omega} + \frac{\gamma}{2}\|u\|^2_{L^2_Q} \end{cases} \tag{4.1}$$

over the set U^{ad} of admissible controls given by

$$U^{ad} = \{u \in L^2_Q := L^2(Q) \mid u_a \le u \le u_b \quad a.e. \quad \text{in} \quad Q = \Omega \times (0,T)\},$$

subject to a semi-linear parabolic PDE system

$$\begin{cases} -\partial_t y + \sigma\Delta y + S(y) = f + u \quad \text{in} \quad Q = \Omega \times (0,T), \\ y = g \quad \text{on} \quad \Omega \times \{t=0\}, \\ y = 0 \quad \text{on} \quad \Sigma = \partial\Omega \times (0,T), \end{cases} \tag{4.2}$$

where $\Omega = (0,1)^2$, $u \in U^{ad}$ is the control function, $z \in L^2(Q)$ is the desired tracking trajectory, $z_T \in L^2_\Omega := L^2(\Omega)$ is the target terminal state at the terminal time $T > 0$, α and $\beta \ge 0$ ($\alpha + \beta > 0$) are optimization parameters, $\gamma > 0$ represents either the weight of the cost of control or the Tikhonov regularization parameter, σ denotes the diffusion coefficient, $S : \mathbb{R} \to \mathbb{R}$ is a given nonlinear smooth function, $f \in L^2(Q)$, and $\{u_a, u_b\} \subset L^\infty(Q)$. To focus on the reaction-diffusion equation as in [Borzì and González Andrade, 2012], we further assume $\sigma > 0$ and the initial condition $g \in H^1_0(\Omega)$. Notice that the case $(\alpha,\beta) = (1,0)$ corresponds to tracking without

a terminal observation, while another case $(\alpha, \beta) = (0, 1)$ only concerns the final target state without specifying any reference trajectory. The existence of solutions to the above optimal control problem (4.1-4.2) can be obtained under suitable assumptions for the non-linearity of S; see [Lions, 1971, Neittaanmaki and Tiba, 1994, Tröltzsch, 2010].

Several recent papers [Borzì, 2003, Borzì and Griesse, 2005, Borzì and Griesse, 2006, Borzì, 2007b, Borzì and Schulz, 2009, Borzì and von Winckel, 2009, Borzì and González Andrade, 2012, González Andrade and Borzì, 2012] are devoted to apply the idea of space-time multigrid to the forward-and-backward coupled parabolic optimality system derived from the first-order necessary condition of parabolic optimal control problems. In particular, the full-approximation-storage (FAS) multigrid method [Brandt and Livne, 2011] is extensively used to treat the non-linearity as well as the variational inequality due to control constraints. The developed smoother in the FAS multigrid method is shown to be equivalent to a local semi-smooth Newton (SSN) method [Borzì and González Andrade, 2012, González Andrade and Borzì, 2012], which mainly focuses on the non-linearity of the system and lets the non-smooth control constraints be implicitly treated by projection in the process of smoothing iterations.

As a new attempt, in this chapter, we are trying to lift up the SSN implementation to the most top level so that both non-linearity and non-smoothness can be simultaneously handled in a single procedure. More precisely, the nonlinear term and the non-smoothness are 'linearized' at the same time during the approximation of each Newton iteration to gain a better computational efficiency. This approach is different from the current FAS multigrid method, in which the nonlinear term and the non-smoothness are handled in separate procedures that requires more computational effort. More details in terms of computational advantages of the Newton-multigrid method over the FAS multigrid method have been discussed in a recent paper [Brabazon et al., 2014], in which Newton-multigrid method is applied to solve the second order differential operators of elliptic and parabolic type with nonlinear coefficients. Moreover, it is well-known that the SSN method has the property of mesh-independence convergence as discussed in [Hintermüller and Ulbrich, 2004, Bergounioux et al., 1999, Hintermüller et al., 2002,

Ito and Kunisch, 2008, Hinze et al., 2009, Ulbrich, 2011], and in [Chen et al., 2000] where a class of semi-linear elliptic PDEs is considered. Thus it is attractive to develop a new SSN method for solving the optimization of parabolic PDE problem in order to achieve a better computational efficiency and maintain its favorable super-linear convergence [Ortega and Rheinboldt, 2000]. It is worthy of mentioning here that the FAS multigrid method has its own merits, in particular, when the available memory becomes a main concern [Brabazon et al., 2014], though it seems not an issue for the problem discussed in this chapter.

Under the same motivation explained in the previous chapter, we provide a new second-order leapfrog scheme in time, which is more advantageous in the design of the collective Jacobi smoother under our multigrid solver setting. Besides finite difference methods, many other discretization strategies in both time and space have been extensively studied [Liu et al., 2004, Chrysafinos, 2010, Meidner and Vexler, 2011]. Nevertheless, the enormously increasing dimension (4 million unknowns with a mesh size 1/128) of the discretized nonlinear and non-smooth system as the mesh size refines requires us to scrutinize the possibility of designing a high efficient solver during (not after) the process of discretization, since those standard/general direct or iterative (sparse) solvers may become very inefficient in dealing with such high dimensional systems.

This chapter is organized as follows. In the next section, we formulate the corresponding first-order necessary optimality conditions to characterize the optimal solutions of our posed control-constrained semi-linear parabolic optimal control problem. A continuous SSN method is used to solve the non-smooth optimality system associated with control constraints. We also propose a second-order five-point finite difference scheme in space and leapfrog scheme (with BDF2) in time for the discretization of the optimality system. The convergence of our proposed leapfrog scheme is proved for both linear and nonlinear cases with time-periodic solutions in Section 4.3. In Section 4.4, the discretized optimality system is solved by the discrete SSN method, in which a linear multigrid algorithm is presented to approximately solve the Jacobian system in each outer Newton iteration. In Section 4.5, results of numerical simulations are reported to demonstrate

the second-order accuracy of approximations and mesh independent convergence with linear time complexity of our integrated SSN multigrid approach. Finally, the chapter ends with concluding remarks in Section 4.6.

4.2 OPTIMALITY SYSTEM WITH A LEAPFROG SCHEME

Based on the Lagrange functional, the optimal solution to (4.1-4.2) is characterized by the following first-order optimality system

$$\begin{cases} -\partial_t y + \sigma\Delta y + S(y) - u = f \quad \text{in} \quad Q, y = 0 \quad \text{on} \quad \Sigma, \\ \partial_t p + \sigma\Delta p + S'(y)p + \alpha y = \alpha z \quad \text{in} \quad Q, p = 0 \quad \text{on} \quad \Sigma, \\ (\gamma u - p, v - u) \geq 0 \quad \text{for all} \quad v \in U^{ad}, \end{cases} \tag{4.3}$$

where the state variable y evolves forward with the initial condition

$$y(x_1, x_2, 0) = g(x_1, x_2) \tag{4.4}$$

and the adjoint variable p marches backward with a terminal condition

$$p(x_1, x_2, T) = \beta(y(x_1, x_2, T) - z_T(x_1, x_2)). \tag{4.5}$$

By making use of the principle of variational inequality (last inequality in (4.3)), one can obtain the following equivalent characterization of the optimal control

$$u = \Phi(p/\gamma) := \max\{u_a, \min\{u_b, p/\gamma\}\}, \tag{4.6}$$

where $\Phi(\cdot)$ denotes the element-wise projection onto U^{ad}. By substituting (4.6) into the optimality conditions (4.3), we thus obtain the following non-smooth nonlinear optimality system in terms

of only (y, p)

$$\begin{cases} -\partial_t y + \sigma\Delta y + S(y) - \Phi(p/\gamma) = f \quad \text{in} \quad Q, y = 0 \quad \text{on} \quad \Sigma, \\ \partial_t p + \sigma\Delta p + S'(y)p + \alpha y = \alpha z \quad \text{in} \quad Q, p = 0 \quad \text{on} \quad \Sigma. \end{cases} \tag{4.7}$$

It is well-known [Borzì and Schulz, 2009] that solving the above type of coupled time-dependent nonlinear PDE equations marching in opposite time orientation poses a major challenge (even in linear case) in scientific computing, partially because it is required to store those dependent variables for all time steps (especially for a large time interval).

In order to implement the SSN algorithm in practice, one has to first discretize the continuous optimality system (4.7). We use finite difference schemes since it makes the algorithm formulation simpler. Let $h = 1/(n+1)$ be the space mesh size and $\tau = T/n_t$ be the time-step size. Then we discretize the space domain Ω using a uniform Cartesian grid

$$\Omega_h = \left\{ (x_1^i, x_2^j) = (ih, jh) : \quad i, j = 1, 2, \cdots, n \right\}$$

and define the space-time mesh

$$Q_{h,\tau} = \{(x, t_m) : x \in \Omega_h, t_m = m\tau, 0 \le m \le n_t\}.$$

Let $y_{i,j}^m$, $p_{i,j}^m$, $f_{i,j}^m$, and $z_{i,j}^m$ represent approximations to $y(x_1^i, x_2^j, t_m)$, $p(x_1^i, x_2^j, t_m)$, $f(x_1^i, x_2^j, t_m)$, and $z(x_1^i, x_2^j, t_m)$, respectively. We denote y_h^m, p_h^m, f_h^m, and z_h^m the corresponding lexicographic ordering (vectorization) of those approximations over $\Omega_h \times \{t = t_m\}$. In discretization of (4.7), we employ the second order five-point finite difference scheme [Borzí and Kunisch, 2005, Borzì, 2008] in space. Denote the corresponding discretization of the Laplacian operator Δ on Ω_h by Δ_h, where the homogeneous Dirichlet boundary conditions are already included.

The time discretization is more involved since y evolves forward and p marches backward. Furthermore, the terminal condition $p(x_1, x_2, T)$ may be unknown when $\beta > 0$, which is different

from the case of the given initial condition g. Thus, a robust scheme should treat $p(x_1, x_2, T)$ as an unknown. In the following schemes, we always incorporate the discretized initial and terminal conditions

$$y_h^0 = g_h \quad \text{and} \quad p_h^{n_t} = \beta(y_h^{n_t} - z_h^{n_t}). \tag{4.8}$$

In [Borzì, 2003], appling the backward Euler scheme to the state equation in (4.7) which gives

$$-\frac{y_h^m - y_h^{m-1}}{\tau} + \sigma\Delta_h y_h^m + S(y_h^m) - \Phi(p_h^m/\gamma) = f_h^m \tag{4.9}$$

for $1 \leq m \leq n_t$ and the forward Euler scheme to the adjoint equation generates

$$\frac{p_h^{m+1} - p_h^m}{\tau} + \sigma\Delta_h p_h^m + S'(y_h^m)p_h^m + \alpha y_h^m = \alpha z_h^m \tag{4.10}$$

for $0 \leq m \leq n_t - 1$. Without any surprise, the above Euler scheme (4.8,4.9,4.10) gives only first-order accuracy in time but second-order accuracy in space. To achieve second-order accuracy in time, the authors in [González Andrade and Borzì, 2012] suggest to use the second-order backward differentiation formula (BDF2) together with the Crank-Nicolson scheme at the initial time step. Though the BDF2 with Crank-Nicolson scheme is proved to have a second-order accuracy in the case where control constraints are inactive, the averaging treatment in the Crank-Nicolson scheme complicates the structure of the discretized optimality system. This will consequently make it more difficult to develop an effective multigrid solver for the Jacobian systems arising from the SSN method applied to the discretized optimality system.

It is shown in [Neitzel et al., 2011] under suitable conditions the optimality system is actually equivalent to a V-elliptic problem on the space-time cylinder, and this motivates us to employ the second-order leapfrog scheme (in time variable), which is known to have poor stability in terms

of classical stability definition for a single parabolic equation. However, we will show the stability of the two-point boundary problem resulted from leapfrog scheme in next section. Our proposed second-order scheme is set to be

$$-\frac{y_h^{m+1}-y_h^{m-1}}{2\tau}+\sigma\Delta_h y_h^m+S(y_h^m)-\Phi(p_h^m/\gamma)=f_h^m \tag{4.11}$$

for $1\le m\le n_t-1$ and

$$\frac{p_h^{m+1}-p_h^{m-1}}{2\tau}+\sigma\Delta_h p_h^m+S'(y_h^m)p_h^m+\alpha y_h^m=\alpha z_h^m \tag{4.12}$$

for $1\le m\le n_t-1$. In addition, the above leapfrog scheme at the final time step ($y_h^{n_t}$ and p_h^0) need to be replaced by the BDF2 scheme since we could only use the one-sided finite difference formula to approximate the time derivative at $t=0$ and $t=T$, that is,

$$-\frac{y_h^{n_t-2}-4y_h^{n_t-1}+3y_h^{n_t}}{2\tau}+\sigma\Delta_h y_h^{n_t}+S(y_h^{n_t})-\Phi(p_h^{n_t}/\gamma)=f_h^{n_t}, \tag{4.13}$$

$$\frac{-3p_h^0+4p_h^1-p_h^2}{2\tau}+\sigma\Delta_h p_h^0+S'(y_h^0)p_h^0+\alpha y_h^0=\alpha z_h^0, \tag{4.14}$$

We remark that our proposed scheme (4.8,4.11,4.12,4.13,4.14) does achieve a second-order accuracy in numerical simulations, and, more importantly, the simple structure of its corresponding Jacobian matrices allows us to develop a very effective multigrid solver for SSN iterations as illustrated in the next section. The Jacobian matrices produced by the BDF2 with the Crank-Nicolson scheme [González Andrade and Borzì, 2012] will have much more complicated structure, which may not guarantee an effective multigrid algorithm as our proposed approach. Alternatively, as we already showed in the previous chapter, it is also possible to use a simpler backward Euler scheme at the final time step without sacrificing the second-order accuracy. But its approximation

error is slightly larger than that of BDF2 even though it has the same order of accuracy. For the case without control constraints, the error estimates by using energy norm is provided in Theorem 3.2.1 when the backward Euler scheme at the final time step is used. We mainly focus on the difficulties resulting from non-linearity as well as control constraints, which requires more subtle techniques for the implementation of a fast solver.

In order to illustrate the novelty and difference of our above leapfrog scheme from the conventional unstable leapfrog scheme, we consider the following initial-boundary value problem with heat equation

$$\begin{cases} \partial_t y(x,t) = \partial_{xx} y(x,t), \quad (x,t) \in (0,1) \times (0,T], \\ y(x,0) = \sin(\pi x), \quad x \in (0,1), \\ y(0,t) = y(1,t) = 0, \quad t \in [0,T], \end{cases} \tag{4.15}$$

with the known analytic solution $y(x,t) = e^{-\pi^2 t}\sin(\pi x)$. Applying central finite difference in space and the leapfrog discretization in time, the resulting coupled scheme defined on each grid node x^i reads (using previous notations)

$$\begin{cases} \frac{y_h^{m+1} - y_h^{m-1}}{2\tau} = \Delta_h y_h^m, \quad 1 \le m \le n_t - 1, \\ \frac{y_h^{n_t-2} - 4y_h^{n_t-1} + 3y_h^{n_t}}{2\tau} = \Delta_h y_h^{n_t}, \end{cases} \tag{4.16}$$

with $\Delta_h y_h^m(x^i) = \frac{y_h^m(x^{i-1}) - 2y_h^m(x^i) + y_h^m(x^{i+1})}{h^2}$ and the initial $y_h^0(x^i) = \sin(\pi x^i)$. The above leapfrog scheme (4.16) has to be solved in one-shot due to the coupling. This is different from the classical unstable explicit three-level time-marching leapfrog scheme [Morton and Mayers, 2005, Strikwerda, 2004, LeVeque, 2007], that is

$$\begin{cases} \frac{y_h^1 - y_h^0}{\tau} = \Delta_h y_h^1, \\ \frac{y_h^{m+1} - y_h^{m-1}}{2\tau} = \Delta_h y_h^m, \quad 1 \le m \le n_t - 1. \end{cases} \tag{4.17}$$

One may solve (4.17) in one-shot by stacking all time steps, but the instability persists in the highly ill-conditioned coefficient matrix. Hence the intrinsic difference is the scheme itself instead of the one-shot solving. In Table 4.1 we compare the maximum errors of solving the above heat equation using both schemes (4.16) and (4.17). Our modified leapfrog scheme (4.16) demonstrates an evident second-order accuracy, while the unstable leapfrog scheme (4.17) diverges rapidly as the mesh is refined. Here the notation 'Inf' implies the computed solution already blows up to be greater that the largest finite floating-point number ($\approx 1.7977e308$ in MATLAB) in IEEE double precision. Although our modified leapfrog scheme may not be necessary in this case (a single PDE) since it requires to solve a larger linear system, it does show a better efficiency for solving our interested forward-backward PDEs (4.3). The instability of the classical leapfrog scheme for a parabolic PDE is so recognized and thus is never used for optimal control of parabolic PDE. In terms of the effort shown in this chapter, our modified leapfrog scheme fills in this gap.

Table 4.1. Maximum norm errors for solving the heat equation with different T.

	Unstable leapfrog scheme (4.17)			Our leapfrog scheme (4.16)		
(n, n_t)	$T = 0.01$	$T = 0.1$	$T = 1$	$T = 0.01$	$T = 0.1$	$T = 1$
(16,16)	2.90e-04	2.33e-01	3.30e14	2.88e-04	1.42e-03	2.13e-02
(32,32)	4.03e-03	7.73e26	5.37e57	7.19e-05	3.55e-04	5.90e-03
(64,64)	7.43e28	7.18e90	7.01e153	1.80e-05	8.86e-05	1.51e-03
(128,128)	1.30e112	2.45e238	Inf	4.50e-06	2.21e-05	3.82e-04
(256,256)	Inf	Inf	Inf	1.12e-06	5.53e-06	9.57e-05
(512,512)	Inf	Inf	Inf	2.81e-07	1.38e-06	2.39e-05
(1024,1024)	Inf	Inf	Inf	7.03e-08	3.46e-07	5.98e-06

4.3 STABILITY ANALYSIS FOR PERIODIC CASE

In this section, we conduct the stability analysis of the proposed scheme. For our proposed scheme (4.8,4.11,4.12,4.13,4.14), those one-sided finite difference formulas greatly complicate the theoretical analysis of the resulting discretized system. Hence, we further assume the solutions are periodic in time [Abbeloos et al., 2011] with period T, that is $y(\cdot, 0) = y(\cdot, T)$ and therefore we have

$$y_h^{n_t} = y_h^0,\ y_h^{n_t-1} = y_h^{-1},\ p_h^0 = p_h^{n_t}, \quad \text{and} \quad p_h^{-1} = p_h^{n_t-1}.$$

With the above assumptions, we can avoid the one-sided finite difference formulas (4.13,4.14) and initial conditions (4.8). It allows us to formulate the discretized system from (4.11,4.12) with unknowns y_h^m and p_h^m for $0 \le m \le n_t - 1$ as

$$T_h := \begin{bmatrix} C_h & B_h^\intercal \\ B_h & D_h \end{bmatrix} \begin{bmatrix} y_h \\ p_h \end{bmatrix} = \begin{bmatrix} c_h \\ d_h \end{bmatrix}, \tag{4.18}$$

where $C_h = \alpha I$, $D_h = -I/\gamma$,

$$B_h = \begin{bmatrix} \sigma\Delta_h & -I/(2\tau) & 0 & \cdots & I/(2\tau) \\ I/(2\tau) & \sigma\Delta_h & -I/(2\tau) & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I/(2\tau) & \sigma\Delta_h & -I/(2\tau) \\ -I/(2\tau) & 0 & \cdots & I/(2\tau) & \sigma\Delta_h \end{bmatrix},$$

and

$$c_h = \alpha \begin{bmatrix} z_h^0 \\ z_h^1 \\ \vdots \\ z_h^{n_t-2} \\ z_h^{n_t-1} \end{bmatrix}, \; d_h = \begin{bmatrix} f_h^0 \\ f_h^1 \\ \vdots \\ f_h^{n_t-2} \\ f_h^{n_t-1} \end{bmatrix}.$$

Here $B_h^\intercal$ denotes the transpose of B_h. Under the above framework, we actually do not need any time forward iteration procedures as the classical approach. Thus, the traditional approach of proving convergence by showing the scheme is consistent and stable for parabolic equations does not fit in our framework. Instead, we turn to consider its stability from the perspective of elliptic equations, that is to validate that T_h has a uniformly bounded inverse as given in the following theorem.

Theorem 4.3.1. *In system (4.18), $\|T_h^{-1}\|_2$ is uniformly bounded for all $h > 0$ and $\tau > 0$, where $\|\cdot\|_2$ is the operator (spectral) norm associated with the discrete L^2 norm.*

Proof. We first show that T_h is invertible and then prove $\|T_h^{-1}\|_2$ is uniformly bounded. Let the eigenvalues of a square matrix A be arranged so that $|\lambda_{\max}(A)| \geq \cdots \geq |\lambda_{\min}(A)|$, and the singular values of A be ordered as $s_{\max}(A) \geq \cdots \geq s_{\min}(A)$. Let (λ, ξ) be any eigenpair of T_h with a normalized eigenvector $\|\xi\|_2^2 = \xi^*\xi = 1$. Partition $\xi = \begin{bmatrix} \xi_1 \\ \\ \xi_2 \end{bmatrix}$ according to the block structure of T_h. Then $T_h\xi = \lambda\xi$ gives

$$\begin{cases} C_h\xi_1 + B_h^{\mathsf{T}}\xi_2 = \lambda\xi_1 \\ \\ B_h\xi_1 + D_h\xi_2 = \lambda\xi_2. \end{cases} \tag{4.19}$$

By multiplying the first equation by ξ_1^* and the second one by ξ_2^* we get

$$\begin{cases} \xi_1^*C_h\xi_1 + \xi_1^*B_h^{\mathsf{T}}\xi_2 = \lambda\xi_1^*\xi_1 \\ \\ \xi_2^*B_h\xi_1 + \xi_2^*D_h\xi_2 = \lambda\xi_2^*\xi_2. \end{cases} \tag{4.20}$$

Notice $\left(\xi_1^*B_h^{\mathsf{T}}\xi_2\right)^* = \xi_2^*B_h\xi_1$ and $\lambda^* = \lambda$. By subtracting the second equation from the conjugate of first one, we obtain

$$0 < \alpha\xi_1^*\xi_1 + \frac{1}{\gamma}\xi_2^*\xi_2 = \xi_1^*C_h\xi_1 - \xi_2^*D_h\xi_2 = \lambda(\xi_1^*\xi_1 - \xi_2^*\xi_2), \tag{4.21}$$

which implies $\lambda \neq 0$ and thus T_h is invertible.

Next, we will estimate the bounds of eigenvalues of T_h. If $\lambda > 0$ then $\xi_1 \neq 0$, since otherwise (4.19) implies $\xi_2 = 0$, which contradicts to $\xi_1^*\xi_1 + \xi_2^*\xi_2 = 1$. Thus we get

$$(\lambda - \alpha)\xi_1^*\xi_1 > (\lambda + \tfrac{1}{\gamma})\xi_2^*\xi_2 \geq 0,$$

which implies $\lambda > \alpha > 0$ since $\xi_1^*\xi_1 > 0$.

Similarly, for $\lambda < 0$, $(C_h - \lambda I)$ is positive definite, from the first equation in (4.19) we obtain $\xi_1 = -(C_h - \lambda I)^{-1} B_h^{\intercal} \xi_2$, which can be substituted into the second equation to get

$$B_h (C_h - \lambda I)^{-1} B_h^{\intercal} \xi_2 - D_h \xi_2 = -\lambda \xi_2.$$

Multiplying from the left by ξ_2^* and noticing $(-D_h)$ is positive semidefinite we obtain

$$\xi_2^* B_h (C_h - \lambda I)^{-1} B_h^{\intercal} \xi_2 \leq -\lambda \xi_2^* \xi_2.$$

This further gives

$$\begin{aligned} (\lambda_{\max}(C_h) - \lambda)^{-1} s_{\min}^2(B_h) &\leq \tfrac{\xi_2^* B_h (C_h - \lambda I)^{-1} B_h^{\intercal} \xi_2}{\xi_2^* B_h B_h^{\intercal} \xi_2} \tfrac{\xi_2^* B_h B_h^{\intercal} \xi_2}{\xi_2^* \xi_2} \\ &= \tfrac{\xi_2^* B_h (C_h - \lambda I)^{-1} B_h^{\intercal} \xi_2}{\xi_2^* \xi_2} \leq -\lambda, \end{aligned}$$

which is

$$\lambda^2 - \lambda_{\max}(C_h)\lambda - s_{\min}^2(B_h) \geq 0.$$

Under the condition $\lambda < 0$, we derive

$$\lambda \leq \tfrac{1}{2} \left(\lambda_{\max}(C_h) - \sqrt{\lambda_{\max}^2(C_h) + 4 s_{\min}^2(B_h)} \right).$$

Denote the Hermitian part of B_h by $\mathcal{H}(B_h)$, then

$$\mathcal{H}(B_h) = (B_h + B_h^{\intercal})/2 = I_{n_t} \otimes (\sigma \Delta_h).$$

From matrix theory, there holds [Horn and Johnson, 2013]

$$\begin{aligned} s_{\min}(B_h) &= s_{\min}(-B_h) \geq \lambda_{\min}(\mathcal{H}(-B_h)) \\ &= \sigma\lambda_{\min}(-\Delta_h) = \sigma(2\pi^2 - O(h^2)) > \sigma\pi^2 \end{aligned}$$

for any $h < 1$, where the estimation of $\lambda_{\min}(-\Delta_h)$ is a classical result from [Hackbusch, 2003]. Thus

$$\begin{aligned} \lambda &< \tfrac{1}{2}\left(\lambda_{\max}(C_h) - \sqrt{\lambda_{\max}^2(C_h) + 4\sigma^2\pi^4}\right) \\ &= \tfrac{-2\sigma^2\pi^4}{\lambda_{\max}(C_h)+\sqrt{\lambda_{\max}^2(C_h)+4\sigma^2\pi^4}} \leq \tfrac{-2\sigma^2\pi^4}{\alpha+\sqrt{\alpha^2+4\sigma^2\pi^4}}. \end{aligned}$$

To this end, we have shown that either $\lambda > \alpha > 0$ or $\lambda < \frac{-2\sigma^2\pi^4}{\alpha+\sqrt{\alpha^2+4\sigma^2\pi^4}}$, which gives

$$\|T_h^{-1}\|_2 = \tfrac{1}{\sigma_{\min}(T_h)} = \tfrac{1}{|\lambda_{\min}(T_h)|} \leq \max\{\tfrac{1}{\alpha}, \tfrac{\alpha+\sqrt{\alpha^2+4\pi^4}}{2\sigma^2\pi^4}\},$$

where $\alpha > 0$ and $\sigma > 0$ are independent of h and τ. This completes the proof. □

In the above theorem, we only discussed the time-periodic case without control constraints and nonlinear term $S(\cdot)$. The time-periodicity allows us to derive a well-structured coefficient matrix by neglecting sophisticated boundary schemes. The technique assumption on periodic solutions is only for the purpose of theoretical analysis, our scheme also works for general non-periodic cases as illustrated in the numerical examples section. As to be shown the following section, the general nonlinear case with control constraints will be linearized at each SSN iteration. According to Theorem 1.5.1, we need to validate the Jacobian systems (4.24) have a uniformly bounded inverse. Those Jacobian matrices have a very similar structure as in the above linear unconstrained time-periodic case. We briefly outline how the above proof can be generalized to the nonlinear constrained time-periodic case under certain technical assumptions on the nonlinear term $S(\cdot)$.

In the case with nonlinear term as well as control constraints, in terms of discretization shown in the next section, those corresponding blocks B_h, C_h, and D_h appearing in (4.18) now

become

$$B_h = \begin{bmatrix} \sigma\Delta_h & -I/(2\tau) & 0 & \cdots & I/(2\tau) \\ I/(2\tau) & \sigma\Delta_h & -I/(2\tau) & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I/(2\tau) & \sigma\Delta_h & -I/(2\tau) \\ -I/(2\tau) & 0 & \cdots & I/(2\tau) & \sigma\Delta_h \end{bmatrix} + \mathcal{D}(S'(y_h)),$$

$$C_h = \alpha I + \mathcal{D}(S''(y_h)p_h), \quad D_h = -\mathcal{D}(\partial\Phi(p_h/\gamma)/\gamma).$$

Theorem 4.3.2. *For the nonlinear case, we assume that there exists two positive constants κ_1 and κ_2, respectively, such that*

(1) $s_{\min}(B_h) \geq \kappa_1$, *and*

(2) $\alpha + S''(y_h)p_h \geq \kappa_2$

hold pointwisely. Then the inverse of T_h is uniformly bounded for all $h > 0$ and $\tau > 0$.

Although the above assumptions seem to be cumbersome, it is a sufficient condition for the existence of a minimizer at least at the discrete level.

Proof. The assumption $s_{\min}(B_h) \geq \kappa_1$ gives $\xi_1 \neq 0$ and $\xi_2 \neq 0$ according to (4.19) since otherwise (4.19) would lead to a contradiction $s_{\min}(B_h) = 0$. By the second assumption $\alpha + S''(y_h)p_h \geq \kappa_2$, the inequality (4.21) now reads

$$0 < \kappa_2\xi_1^*\xi_1 \leq \xi_1^* C_h\xi_1 \leq \xi_1^* C_h\xi_1 - \xi_2^* D_h\xi_2 = \lambda(\xi_1^*\xi_1 - \xi_2^*\xi_2), \tag{4.22}$$

which shows that T_h is invertible. Notice that here $D_h = -\mathcal{D}(\partial\Phi(p_h/\gamma)/\gamma)$ is negative semidefinite. The rest of the proof is omitted since it follows the arguments in Theorem 4.3.1 in a straightforward manner, except for using the constants κ_1 and κ_2 in the derived bounds. □

4.4 SSN-MULTIGRID METHOD FOR OPTIMALITY SYSTEM

In this section, we first reformulate the proposed scheme (4.8,4.11,4.12,4.13,4.14) in an organized way such that the resulting nonlinear systems of equations have well-structured Jacobian matrices. Then we carefully construct all key components of a linear multigrid algorithm for solving the linearized Jacobian systems in each SSN iteration. The critical technique here is to separate the linear and nonlinear part of the discretized optimality system.

By using the above notation, we define the following vectors by vertically concatenation over all time steps

$$y_h = [y_h^0; y_h^1; \cdots; y_h^{n_t}], \quad p_h = [p_h^0; p_h^1; \cdots; p_h^{n_t}],$$

$$\hat{y}_h = [0_h; y_h^1; \cdots; y_h^{n_t}], \quad \hat{p}_h = [0_h; p_h^1; \cdots; p_h^{n_t}],$$

$$\check{y}_h = [y_h^0; y_h^1; \cdots; y_h^{n_t-1}; 0_h], \quad \check{p}_h = [p_h^0; p_h^1; \cdots; p_h^{n_t-1}; 0_h],$$

where 0_h denotes the vectorization of a zero function on Ω_h. After discretizing (4.7) with scheme (4.8,4.11,4.14,4.13,4.12), we obtain the discrete optimality system in the form of

$$F_h(y_h, p_h) = 0 \tag{4.23}$$

with

$$F_h(y_h, p_h) :=$$

$$\begin{bmatrix}
y_h^0 - g_h \\
-\frac{y_h^2 - y_h^0}{2\tau} + \sigma\Delta_h y_h^1 + S(y_h^1) - \Phi(p_h^1/\gamma) - f_h^1 \\
\vdots \\
-\frac{y_h^{n_t} - y_h^{n_t-2}}{2\tau} + \sigma\Delta_h y_h^{n_t-1} + S(y_h^{n_t-1}) - \Phi(p_h^{n_t-1}/\gamma) - f_h^{n_t-1} \\
-\frac{y_h^{n_t-2} - 4y_h^{n_t-1} + 3y_h^{n_t}}{2\tau} + \sigma\Delta_h y_h^{n_t} + S(y_h^{n_t}) - \Phi(p_h^{n_t}/\gamma) - f_h^{n_t} \\
\frac{-3p_h^0 + 4p_h^1 - p_h^2}{2\tau} + \sigma\Delta_h p_h^0 + S'(y_h^0)p_h^0 + \alpha y_h^0 - \alpha z_h^0 \\
\frac{p_h^2 - p_h^0}{2\tau} + \sigma\Delta_h p_h^1 + S'(y_h^1)p_h^1 + \alpha y_h^1 - \alpha z_h^1 \\
\vdots \\
\frac{p_h^{n_t} - p_h^{n_t-2}}{2\tau} + \sigma\Delta_h p_h^{n_t-1} + S'(y_h^{n_t-1})p_h^{n_t-1} + \alpha y_h^{n_t-1} - \alpha z_h^{n_t-1} \\
p_h^{n_t} - \beta(y_h^{n_t} - z_h^{n_t})
\end{bmatrix},$$

where $S(\cdot)$, $S'(\cdot)$, and $\Phi(\cdot)$ are element-wise defined and so is the multiplication $S'(y_h^m)p_h^m$. A further employment of some matrix notation we can separate the above system into

$$F_h(y_h, p_h) = L_h \begin{bmatrix} y_h \\ p_h \end{bmatrix} + N_h(y_h, p_h) = 0,$$

where the linear part

$$L_h = \begin{bmatrix} X_h & 0 \\ Y_h & Z_h \end{bmatrix}$$

with

$$X_h = \begin{bmatrix} I & 0 & 0 & \cdots & 0 & 0 \\ \frac{I}{2\tau} & \sigma\Delta_h & -\frac{I}{2\tau} & \cdots & 0 & 0 \\ 0 & \frac{I}{2\tau} & \sigma\Delta_h & -\frac{I}{2\tau} & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \frac{I}{2\tau} & \sigma\Delta_h & -\frac{I}{2\tau} \\ 0 & 0 & \cdots & -\frac{I}{2\tau} & \frac{4I}{2\tau} & (\sigma\Delta_h - \frac{3I}{2\tau}) \end{bmatrix},$$

$$Y_h = \mathrm{diag}(\alpha I, \alpha I, \cdots, \alpha I, -\beta I),$$

and

$$Z_h = \begin{bmatrix} (\sigma\Delta_h - \frac{3I}{2\tau}) & \frac{4I}{2\tau} & -\frac{I}{2\tau} & 0 & \cdots & 0 \\ -\frac{I}{2\tau} & \sigma\Delta_h & \frac{I}{2\tau} & \cdots & 0 & 0 \\ 0 & -\frac{I}{2\tau} & \sigma\Delta_h & \frac{I}{2\tau} & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & -\frac{I}{2\tau} & \sigma\Delta_h & \frac{I}{2\tau} \\ 0 & 0 & 0 & \cdots & 0 & I \end{bmatrix},$$

and the nonlinear part

$$N_h(y_h, p_h) = \begin{bmatrix} -g_h \\ S(y_h^1) - \Phi(p_h^1/\gamma) - f_h^1 \\ \vdots \\ S(y_h^{n_t-1}) - \Phi(p_h^{n_t-1}/\gamma) - f_h^{n_t-1} \\ S(y_h^{n_t}) - \Phi(p_h^{n_t}/\gamma) - f_h^{n_t} \\ S'(y_h^0)p_h^0 - \alpha z_h^0 \\ S'(y_h^1)p_h^1 - \alpha z_h^1 \\ \vdots \\ S'(y_h^{n_t-1})p_h^{n_t-1} - \alpha z_h^{n_t-1} \\ \beta z_h^{n_t} \end{bmatrix}.$$

It is easy to verify that F_h in (4.23) has a generalized derivative

$$G_h(y_h, p_h) = L_h + N_h'(y_h, p_h),$$

where

$$N_h'(y_h, p_h) = \begin{bmatrix} \mathcal{D}(S'(\hat{y}_h)) & -\mathcal{D}(\partial\Phi(\hat{p}_h/\gamma)/\gamma) \\ \mathcal{D}(S''(\check{y}_h)\check{p}_h) & \mathcal{D}(S'(\check{y}_h)) \end{bmatrix}$$

where $\mathcal{D}(\cdot)$ denotes a diagonal matrix with the input vector as the diagonal elements. Notice that L_h is not dependent on y_h and p_h, which needs to be constructed only once during the SSN iterations.

Analogously, we can iteratively solve the discrete optimality system (4.23) by the discrete

SSN method. However, to achieve a mesh-independent convergence, we do require that the Jacobian matrix G_h has a uniformly bounded inverse in some open neighborhood containing the optimal control and state with respect to h, as stated in Theorem 1.5.1. This requirement can be tediously verified under certain assumptions on the system. Based on above discussions, the discrete SSN iteration for solving (4.23) is

$$\begin{bmatrix} y_h^{(k+1)} \\ p_h^{(k+1)} \end{bmatrix} = \begin{bmatrix} y_h^{(k)} \\ p_h^{(k)} \end{bmatrix} - G_h(y_h^{(k)}, p_h^{(k)})^{-1} F_h(y_h^{(k)}, p_h^{(k)})$$

with $k = 0, 1, 2, \cdots$, where the initials $(y_h^{(0)}, p_h^{(0)})$ will be specified appropriately. In each SSN iteration, we need to first (approximately) solve the linearized Jacobian system

$$\Big(L_h + N_h'(y_h^{(k)}, p_h^{(k)})\Big) \begin{bmatrix} \delta y^{(k)} \\ \delta p^{(k)} \end{bmatrix} = F_h(y_h^{(k)}, p_h^{(k)}), \tag{4.24}$$

and then update the current k-th approximation according to

$$\begin{bmatrix} y_h^{(k+1)} \\ p_h^{(k+1)} \end{bmatrix} = \begin{bmatrix} y_h^{(k)} \\ p_h^{(k)} \end{bmatrix} - \begin{bmatrix} \delta y^{(k)} \\ \delta p^{(k)} \end{bmatrix}.$$

The desired super-linear or mesh-independent convergence could be fulfilled by the above SSN iterations provided that the assumptions in Theorem 1.5.1 hold [Hintermüller and Ulbrich, 2004]. Since the algebraic system (4.24) is required to be exactly solved up to machine error, such an approach becomes very inefficient as the mesh refines. Another difficulty is how to efficiently solve the above linearized Jacobian system (4.24) by taking advantage of its high sparsity. When (4.24) is only approximately solved, such as with only two multigrid V-cycles to be described in the following, it gives the inexact Newton method [Ortega and Rheinboldt, 2000, Dembo et al., 1982, Brown et al., 2003]. The review paper [Benzi et al., 2005] summarizes many numerical methods

for solving two-by-two block saddle-point systems, but our current system (4.24) has less algebraic properties such as symmetry and definiteness. To the best of our knowledge, no iterative solvers are available for handling the system (4.24). However, our modified leapfrog scheme provides an effective path to handle these challenges for solving parabolic optimal control problems. More specifically, our following multigrid iterative V-cycle aims at efficiently solving (4.24) with a sufficient level of accuracy, which can assure the mesh-independence convergence of the above SSN iterations. During the approach we also try to balance the accuracy of solving (4.24) and the corresponding computational costs, in order to achieve the best overall performance.

We now carry out a specific multigrid implementation for our previous Jacobian system (4.24), which can be simplified (by omitting subscript (k)) as

$$A_h w_h := \big(L_h + N_h'(y_h, p_h)\big) \begin{bmatrix} \delta y \\ \\ \delta p \end{bmatrix} = F_h(y_h, p_h), \tag{4.25}$$

where

$$A_h := \begin{bmatrix} B_h & D_h \\ \\ C_h & E_h \end{bmatrix} := L_h + N_h'(y_h, p_h) = \tag{4.26}$$

$$\begin{bmatrix} X_h + \mathcal{D}(S'(\hat{y}_h)) & -\mathcal{D}(\partial\Phi(\hat{p}_h/\gamma)/\gamma) \\ \\ Y_h + \mathcal{D}(S''(\check{y}_h)\check{p}_h) & Z_h + \mathcal{D}(S'(\check{y}_h)) \end{bmatrix}. \tag{4.27}$$

Next, we discuss how to construct the coarse grid operator

$$A_H := \begin{bmatrix} B_H & D_H \\ \\ C_H & E_H \end{bmatrix}.$$

Here we only use semi-coarsening in space (no coarsening in time) for its faster convergence. The blocks B_H, C_H, and E_H are derived from the finite difference discretization with a coarse step-size

H in space and full-weighted restriction of those smooth nonlinear parts

$$B_H = X_H + \mathcal{D}(S'(I_h^H \hat{y}_h)),$$
$$C_H = Y_H + \mathcal{D}(S''(I_h^H \check{y}_h) I_h^H \check{p}_h),$$
$$E_H = Z_H + \mathcal{D}(S'(I_h^H \check{y}_h)).$$

However, there are two approaches [Liu and Xiao, 2014a] to coarse the non-smooth D_h. The first strategy directly applies I_h^H to the adjoint p_h

$$\tilde{D}_H := -\mathcal{D}(\partial\Phi(I_h^H \hat{p}_h/\gamma))/\gamma, \tag{4.28}$$

which fails to achieve favorable convergence for small γ in our simulations. This type of deteriorated performance may results from those extra high frequency errors introduced by the non-smooth operator $\partial\Phi$, and these errors are supposed to be smoothed out by the smoother on the fine grid. As an alternative, we place the restriction operator I_h^H on the non-smooth operator $\partial\Phi$, that is,

$$D_H := -\mathcal{D}(I_h^H \partial\Phi(\hat{p}_h/\gamma))/\gamma, \tag{4.29}$$

which provides more information to the coarse operator compared with $\tilde{D}_H$. Postponing the restriction operator I_h^H after the non-smooth operator $\partial\Phi$ results in the iterations being able to capture more nonlinear structure of the finer discrete system.

For the smoother SMOOTH, considering its lower computational costs, we make use of a damped collective Jacobi (C-JAC) smoother given in [Takacs and Zulehner, 2011]. Numerical results 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, these iteration schemes can be represented in one compact formula (with damping factor $\omega \in (0, 1]$ for C-JAC

smoother)

$$\begin{bmatrix} \delta y \\ \delta p \end{bmatrix}^{\text{new}} = \begin{bmatrix} \delta y \\ \delta p \end{bmatrix} + \omega P_h^{-1} \left(F_h(y_h, p_h) - A_h \begin{bmatrix} \delta y \\ \delta p \end{bmatrix} \right),$$

with

$$P_h := \begin{bmatrix} \text{diag}(B_h) & \text{diag}(D_h) \\ \text{diag}(C_h) & \text{diag}(E_h) \end{bmatrix} \tag{4.30}$$

where $\text{diag}(\cdot)$ stands for the diagonal part of the input matrix, respectively. Notice that we have $\text{diag}(C_h) = C_h$ and $\text{diag}(D_h) = D_h$ since they are diagonal matrices. Here, the preconditioning step $P_h^{-1}v$ can be computed efficiently based on the partitioned inverse formula [Horn and Johnson, 2013]. Indeed, the time complexity of calculating $P_h^{-1}v$ is of linear $\mathcal{O}(N)$.

4.5 NUMERICAL EXAMPLES

In this section we demonstrate the second-order accuracy and mesh-independence convergence of our developed SSN multigrid method and evaluate its computational performance with respect to parameters. All simulations are implemented using MATLAB 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*. Let r_y and r_p be the residual of the state and adjoint equation, respectively. The SSN algorithm is usually only locally convergent, thus we need to carefully choose the initial values. In outer SSN iterations, the state unknowns are initialized to be the desired tracking trajectory, the adjoint unknowns are set as zero, and the stopping criterion is chosen as

$$\frac{\|r_y^{(k)}\| + \|r_p^{(k)}\|}{\max(1, \|r_y^{(0)}\| + \|r_p^{(0)}\|)} \leq 10^{-8}, \tag{4.31}$$

where $r_y^{(k)}$ and $r_p^{(k)}$ denote the residuals at k-th SSN iteration. For a grid function defined on Q, let $\|\cdot\|$ denotes the standard discrete L^2 norm on Q. Similarly, for a grid function on Ω,

$\|\cdot\|$ means the standard discrete L^2 norm on Ω. The specific definition should be clear from the context. When the exact optimal state y^* and adjoint p^* are known, we measure the order of accuracy by using the norms of their approximation errors $e_y = \|y_h - y^*\|$ and $e_p = \|p_h - p^*\|$. We also compute the tracking error $e_z = \|y_h - z\|$ (when $\alpha > 0$) and terminal observation error $e_T = \|y_h(\cdot, T) - z_T\|$ (when $\beta > 0$). To approximately solve the inner Jacobian system (4.25), we perform only 2 V-cycle multigrid iterations with zero initial and two pre- and post- smoothing iterations. In case of very small γ, numerical tests show that using a W-cycle multigrid gives better convergence. If the nonlinear term S is very complicated, we may increase the number of inner iterations to recover the mesh-independence convergence. The damping factor is set as $\omega = 2/3$ for the C-JAC smoother. The coarsest mesh size is chosen as $h_0 = 1/8$ and spatial coarsening mesh size $H = 2h$.

Example 6.

Our first example is slightly modified from [González Andrade and Borzì, 2012] such that it includes both a nonlinear term and active control constraints. We let $T = 1, \alpha = 1, \beta = 0, \sigma = 1, u_a = -1/2, u_b = 1/2$, and choose the following state, adjoint, and control functions

$$\begin{aligned} y^* &= t^2(1-t)^2 \sin(\pi x_1) \sin(\pi x_2), \\ p^* &= 2\gamma t(1-t)(\pi^2 t^2 - (\pi^2 - 2)t - 1) \sin(\pi x_1) \sin(\pi x_2), \\ u^* &= \max\{u_a, \min\{u_b, p/\gamma\}\}. \end{aligned}$$

The other corresponding functions are given by

$$\begin{aligned} f &= -\partial_t y^* + \sigma \Delta y^* + S(y^*) - u^*, \\ z &= \tfrac{1}{\alpha}(\partial_t p^* + \sigma \Delta p^* + S'(y^*)p^* + \alpha y^*), \end{aligned}$$

where we choose $S(y) = \exp(y)$. Here we assume $\alpha > 0$, for otherwise we needn't to specify the tracking trajectory z and instead only the terminal state z_T should be provided. The above

constructed exact solutions allows us to validate the order of accuracy of our method with a comparison. Notice that we did not approximate the control variable u in our method directly, thus its accuracy is completely determined by the accuracy of adjoint variable p. In Fig. 4.1,

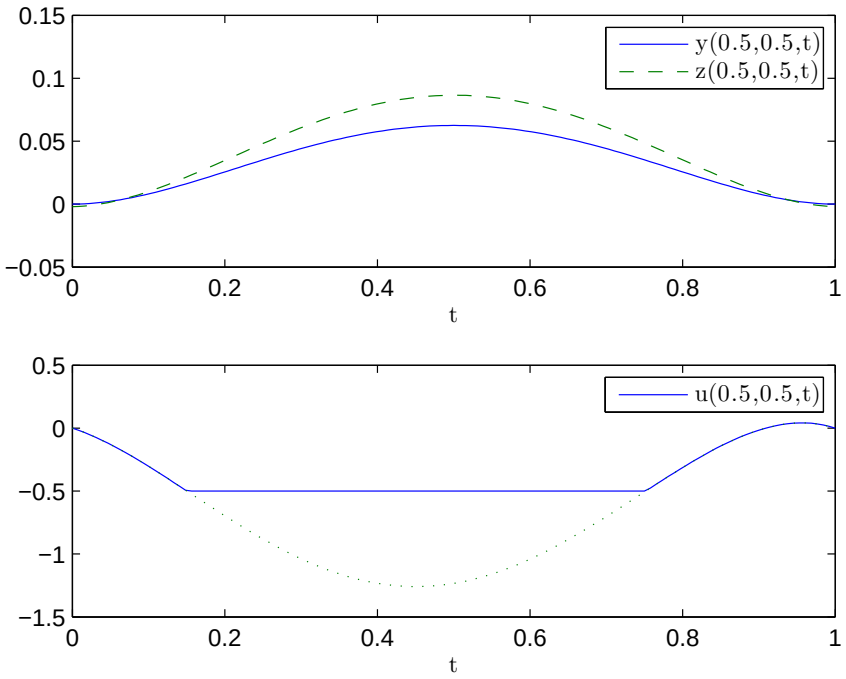

Figure 4.1. The evolution of y, z, and u at $(x_1, x_2) = (0.5, 0.5)$ for Ex. 6.

the time evolution (with $n = 128$) of the state variable p compared to the desired trajectory z at a fix point $(x_1, x_2) = (0.5, 0.5)$ is depicted, where the corresponding control function u is also shown to attain the lower bound u_a in a large portion of the time interval. Here the dotted line in the bottom is p/γ. We also plot (with $n = 128$) in Fig. 4.2 the evolution of the approximation error $\|e_y(\cdot, t)\|_{L^2(\Omega)}$ to inspect any possible instability of the leapfrog scheme in time. The peak occuring in the middle of the time interval indicates that the backward marching adjoint equation suppresses the possible error propagation incurred by the leapfrog scheme. This is a very interesting difference from the case of a single parabolic equation.

In Tables 4.2–4.3, we provide the approximation errors, the required SSN iteration numbers (column 'Iter') and the corresponding CPU time in seconds (column 'CPU') of the our proposed

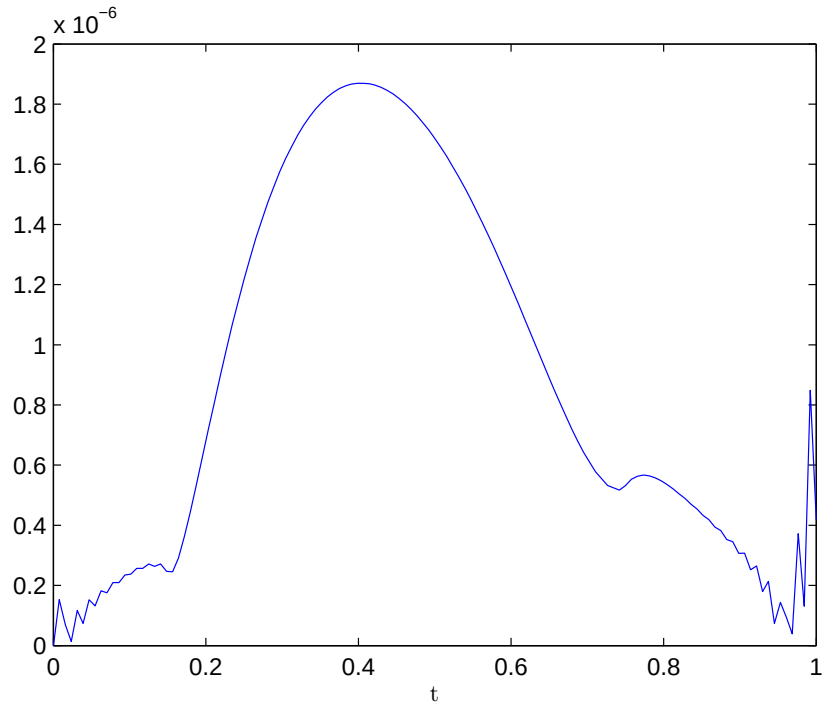

Figure 4.2. The evolution of $e_y(\cdot, t)$ for Ex. 6 with $S(y) = \exp(y)$ and $\gamma = 10^{-3}$.

SSN multigrid (SSN-MG) algorithm (using C-JAC smoother) for different levels of mesh sizes with different regularization parameter γ. They show that halving the space and time mesh sizes, the approximation errors (e_y, e_p) reduce approximately by a factor of four, thus demonstrating a second-order accuracy. We clearly observe the mesh-independent convergence of our proposed SSN-MG method. More importantly, the SSN-MG algorithm numerically achieves the optimal $O(N)$ linear complexity since its CPU time increases roughly eight-fold as the number of unknowns also increase by eight times from one level to the next. Also, the tracking errors e_z becomes smaller as γ decreases, which is as expected since the cost functional becomes more prone to tracking trajectory. Our numerical tests show that the performance of our SSN-MG method is insensitive to the choice of different nonlinear terms $S(y)$.

To compare the performance of our SSN-MG method with the BDF2-based FAS multigrid (FAS-MG) approach in [González Andrade and Borzì, 2012], we also present the corresponding numerical results of Ex. A solved by the FAS-MG method in Tables 4.4–4.5. Here the column 'Iter'

Table 4.2. Results for Ex. 6 using SSN-MG method, with $S(y) = \exp(y)$, $\gamma = 10^{-3}$.

(n, n, n_t)	e_z	e_y	e_p	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	7.36e-03	3.41e-04	1.21e-05	4.03e-16	1.03e-17	4	0.18
(16,16,16)	7.54e-03	7.32e-05	3.01e-06	4.80e-10	8.58e-12	7	0.30
(32,32,32)	7.59e-03	1.74e-05	7.58e-07	2.11e-09	1.76e-11	7	1.13
(64,64,64)	7.60e-03	4.33e-06	1.90e-07	3.87e-09	2.38e-11	7	7.12
(128,128,128)	7.60e-03	1.08e-06	4.74e-08	5.14e-09	2.75e-11	7	55.55

Table 4.3. Results for Ex. 6 using SSN-MG method, with $S(y) = \exp(y)$, $\gamma = 10^{-5}$.

(n, n, n_t)	e_z	e_y	e_p	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	2.25e-04	2.18e-04	1.45e-06	7.22e-11	2.49e-12	3	0.16
(16,16,16)	7.74e-05	3.75e-05	5.21e-07	2.14e-09	5.44e-12	9	0.38
(32,32,32)	7.42e-05	6.77e-06	1.31e-07	1.72e-09	3.92e-12	10	1.71
(64,64,64)	7.55e-05	1.51e-06	3.32e-08	2.18e-09	4.56e-12	10	9.93
(128,128,128)	7.59e-05	3.67e-07	8.29e-09	2.44e-09	5.00e-12	10	76.95

Table 4.4. Results for Ex. 6 using FAS-MG method, with $S(y) = \exp(y)$, $\gamma = 10^{-3}$.

(n, n, n_t)	e_z	e_y	e_p	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	7.70e-03	2.90e-04	1.73e-05	1.70e-09	2.15e-10	10	0.67
(16,16,16)	7.60e-03	6.20e-05	5.28e-06	1.30e-09	2.34e-11	6	2.25
(32,32,32)	7.60e-03	1.43e-05	1.40e-06	1.97e-09	7.01e-12	5	12.47
(64,64,64)	7.60e-03	3.52e-06	3.60e-07	2.14e-09	2.00e-12	4	83.74
(128,128,128)	7.60e-03	8.78e-07	9.12e-08	4.31e-09	3.81e-12	3	528.74

Table 4.5. Results for Ex. 6 using FAS-MG method, with $S(y) = \exp(y)$, $\gamma = 10^{-5}$.

(n, n, n_t)	e_z	e_y	e_p	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	2.79e-04	2.57e-04	2.19e-06	3.96e-09	3.01e-11	5	0.42
(16,16,16)	8.42e-05	3.61e-05	4.31e-07	4.97e-09	1.53e-11	9	2.99
(32,32,32)	7.57e-05	6.30e-06	1.12e-07	5.96e-09	9.35e-12	6	15.37
(64,64,64)	7.58e-05	1.45e-06	2.91e-08	1.68e-09	9.64e-13	5	105.31
(128,128,128)	7.60e-05	3.60e-07	7.41e-09	9.18e-09	1.50e-12	3	514.96

denotes the required number of FAS multigrid iterations. In [González Andrade and Borzì, 2012] two different smoothing schemes (point-wise and line-wise) were discussed. Considering their very similar performance, here we use the point-wise Gauss-Seidel smoothing scheme. Following [González Andrade and Borzì, 2012], for implementing the FAS-MG algorithm we choose W-cycles, two pre- and two post-smoothing steps, and $h_0 = 1/8$ as the coarsest space-mesh size. The nonlinear system on the coarsest grid is approximately solved using 10 smoothing iterations. We

also set the same initial guess and stopping criterion as in our SSN-MG method. We remark that the column 'Iter' in our SSN-MG method is completely different from the column 'Iter' in the FAS-MG method, hence it is not meaningful to compare these values literally.

As also demonstrated in [González Andrade and Borzì, 2012], the FAS-MG algorithm with W-cycles shows a very decent convergence rate. However, our SSN-MG method is significantly faster than the FAS-MG method (comparing with the CPU times) while both of them achieve a comparable second-order accuracy. Notice that our simulations are programmed using MATLAB. The ratios of CPU times may be slightly different if some other programming languages are used, but the conclusion remains the same. The point-wise treatment of the non-linearity and non-smoothness (due to control constraints) in the FAS-MG method becomes much less efficient when the mesh refines. Therefore, our new leapfrog scheme delivers a comparable second-order accuracy as the BDF2 scheme and, more attractively, the proposed SSN-MG method (based on our leapfrog scheme) shows a much better computational efficiency than the one using the FAS-MG method (based on the BDF2 scheme).

Example 7.

The second example is modified from [Borzì, 2003] by adding a nonlinear term $S(y) = \frac{1}{1+y^2}$. We let $T = 5, \gamma = 10^{-6}, u_a = -10, u_b = 10,\ f = 0$, and

$$z = \sin(2\pi t)\sin(\pi x_1)\sin(\pi x_2).$$

Here the desired target trajectory is an oscillating function over time. For simplicity, we set the desired terminal state $z_T = z(x_1, x_2, T)$ whenever $\beta > 0$. For this example, no exact solutions are known. We will report the norm of corresponding residuals to validate the convergence. For a long-time (large T) trajectory tracking, an alternative efficient strategy is to combine the receding-horizon techniques developed in [Borzì, 2007b], which is not implemented here.

In Tables 4.6–4.8, we report the tracking errors e_z, terminal observation error e_T, and resid-

Table 4.6. Results for Ex. 7 using SSN-MG method ($\alpha = 1, \beta = 0, \sigma = 1$).

(n, n, n_t)	e_z	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	1.36e-01	6.12e-12	3.77e-13	6	0.20
(16,16,16)	1.30e-01	9.53e-08	1.96e-10	22	0.84
(32,32,32)	1.38e-01	8.73e-08	1.42e-10	24	3.53
(64,64,64)	1.35e-01	1.24e-07	1.60e-10	26	24.52
(128,128,128)	1.31e-01	1.27e-07	1.55e-10	26	214.46

Table 4.7. Results for Ex. 7 using SSN-MG method ($\alpha = 0, \beta = 1, \sigma = 1$).

(n, n, n_t)	e_T	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	9.66e-07	7.75e-12	4.51e-19	3	0.16
(16,16,16)	1.11e-06	3.45e-08	2.44e-14	5	0.24
(32,32,32)	1.30e-06	8.03e-09	6.26e-15	6	0.94
(64,64,64)	1.53e-06	1.78e-08	1.81e-14	6	5.99
(128,128,128)	1.71e-06	2.71e-08	4.20e-14	6	53.81

Table 4.8. Results for Ex. 7 using SSN-MG method ($\alpha = 1, \beta = 1, \sigma = 10$).

(n, n, n_t)	e_z	e_T	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	7.13e-01	1.04e-06	1.87e-07	8.88e-09	3	0.16
(16,16,16)	7.10e-01	6.73e-07	6.06e-07	1.03e-09	7	0.31
(32,32,32)	7.09e-01	6.71e-07	2.94e-07	2.99e-10	8	1.23
(64,64,64)	7.09e-01	5.04e-07	8.29e-07	5.43e-10	8	8.02
(128,128,128)	7.09e-01	8.64e-07	1.13e-07	4.79e-11	9	80.19

uals of approximated solutions with different parameters. Notice that the residuals r_y and r_p are only required to fulfill the stopping criterion after the last iteration, which may not necessarily decrease as the mesh refines as in Tables 4.6–4.8. However, the residuals should be small enough in order to recover the discretization error of approximated solutions. Surprisingly, although the tracking error e_z may be large due to the restricted control constraints, the terminal observation error e_T could be very small as in Tables 4.7 an 4.8. Our SSN multigrid algorithm demonstrates a very robust high performance with respect to those parameters. In Fig. 4.3–4.4, we plot (with $n = 128$) the time evolution of the state variable p compared to the desired trajectory z at a fixed point $(x_1, x_2) = (0.5, 0.5)$ and the corresponding control u with different β and σ. It shows the capacity of the proposed method to track the desired trajectory over long-time simulation under the control constraints.

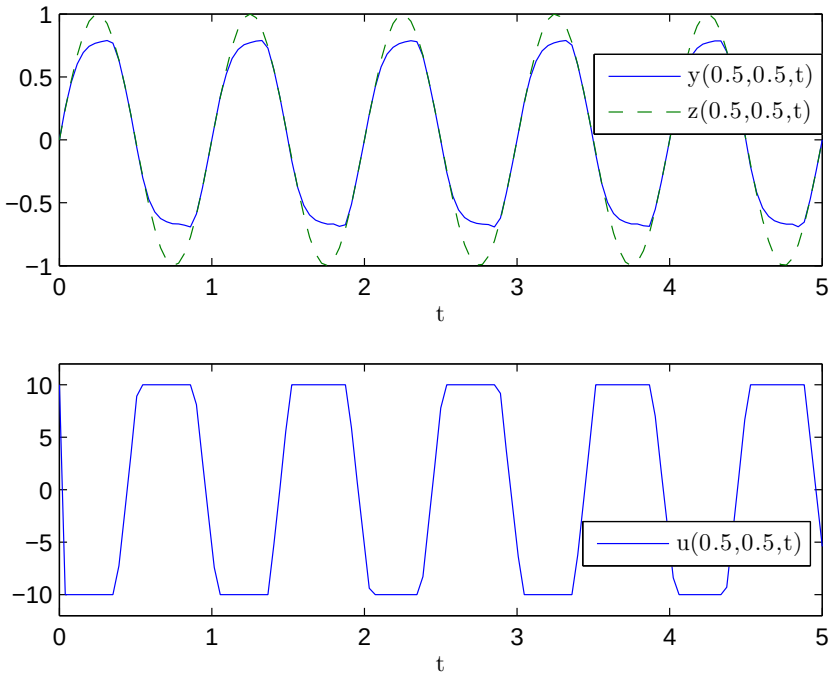

Figure 4.3. The trajectory of y, z, and u at $(0.5, 0.5)$ for Ex. 7 ($\alpha = 1, \beta = 0, \sigma = 1$).

Table 4.9. Results for Ex. 7 using FAS-MG method ($\alpha = 1, \beta = 0, \sigma = 1$).

(n, n, n_t)	e_z	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	2.46e-01	1.84e-08	1.88e-09	11	0.75
(16,16,16)	2.70e-01	5.71e-08	1.91e-10	56	19.43
(32,32,32)	1.50e-01	2.16e-08	1.62e-11	23	64.25
(64,64,64)				>100	
(128,128,128)				>100	

Table 4.10. Results for Ex. 7 using FAS-MG method ($\alpha = 0, \beta = 1, \sigma = 1$).

(n, n, n_t)	e_T	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	1.06e-06	4.52e-08	3.76e-12	11	0.75
(16,16,16)	1.26e-06	5.84e-08	2.37e-12	6	2.17
(32,32,32)	1.48e-06	1.26e-08	1.42e-12	6	16.73
(64,64,64)	1.66e-06	1.27e-08	8.26e-13	5	116.40
(128,128,128)	1.77e-06	4.24e-09	4.63e-13	5	933.39

The corresponding numerical results of Ex. 7 solved by the FAS-MG method are given in Tables 4.9–4.11. Similar as in Ex. 6, our leapfrog-based SSN-MG method shows a much better computational efficiency in terms of CPU time while achieving almost the same level of accuracy

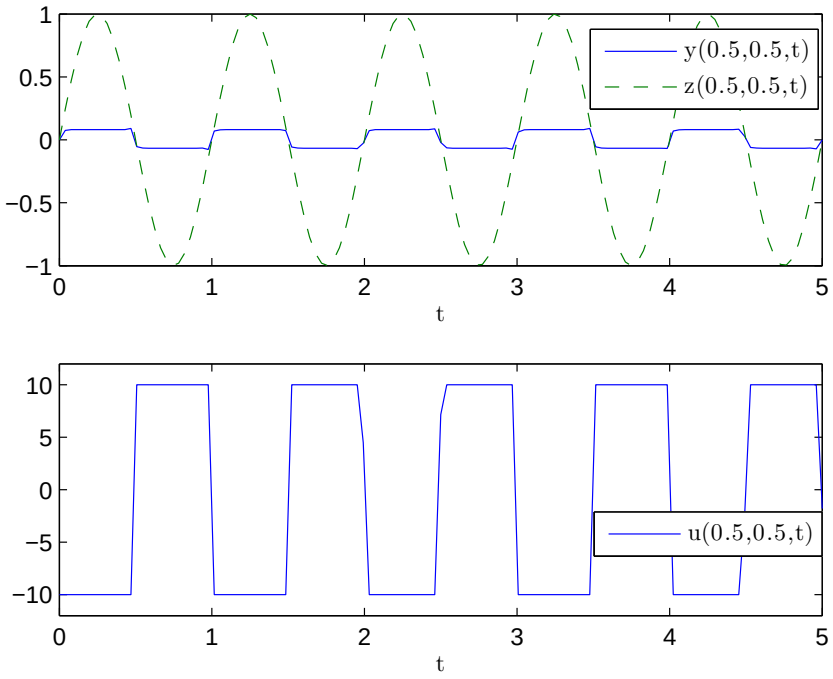

Figure 4.4. The evolution of y, z, and u at $(0.5, 0.5)$ for Ex. 7 ($\alpha = 1, \beta = 1, \sigma = 10$.

Table 4.11. Results for Ex. 7 using FAS-MG method ($\alpha = 1, \beta = 1, \sigma = 10$).

(n, n, n_t)	e_z	e_T	$\|r_y\|$	$\|r_p\|$	Iter	CPU
(8,8,8)	7.06e-01	1.03e-06	3.81e-07	3.15e-08	12	0.81
(16,16,16)	7.05e-01	6.81e-07	1.01e-07	5.19e-09	7	2.58
(32,32,32)	7.06e-01	6.78e-07	4.58e-08	4.60e-11	6	16.89
(64,64,64)	7.08e-01	5.09e-07	1.04e-08	1.51e-12	6	136.63
(128,128,128)	7.09e-01	7.63e-07	4.64e-08	2.36e-12	5	933.37

as the BDF2-based FAS-MG method. In particular, the FAS-MG method in Table 4.9 fails to reach convergence within 100 iterations for a mesh size $h \leq 1/64$. The worsening convergence rate of FAS-MG method was also observed and discussed in [González Andrade and Borzì, 2012] when handling a very small $\gamma = 10^{-6}$. The authors suggested that using finer meshes are necessary to restore a decent convergence rate, but further refinements have become impractical due to very high computational costs. In contrast, our SSN-MG method displays a very robust mesh-independent convergence without restrictions on mesh sizes, as seen in Table 4.6.

4.6 CONCLUSIONS

A new second-order discretization scheme of control-constrained semilinear parabolic optimal control problems was developed by a modified leapfrog scheme in the time variable. A semi-smooth Newton method with a space-time multigrid algorithm as the inner solver was studied in order to efficiently solve the resulting discretized nonlinear and non-smooth optimality system. Numerical experiments were conducted to demonstrate the second-order accuracy, linear time complexity, and robustness of the proposed SSN multigrid method.

CHAPTER 5

AN IMPLICIT PRECONDITIONED ITERATIVE METHOD FOR WAVE CONTROL PROBLEMS WITHOUT CONTROL CONSTRAINTS

5.1 INTRODUCTION

In this chapter, we present an alternative fast iterative solver for solving the optimality PDE system, arising from wave control problems, when the standard multigrid method fails to work. 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 standard optimal control problem [Lions, 1971] of minimizing a tracking-type quadratic cost functional

$$\left\{ \quad J(y,u) = \tfrac{1}{2}\|y-g\|^2_{L^2(Q)} + \tfrac{\gamma}{2}\|u\|^2_{L^2(Q)} \right. \tag{5.1}$$

subject to the linear wave equation:

$$\begin{cases} y_{tt} - \Delta y = & f+u & \text{in} & Q, \\ y = & 0 & \text{on} & \Sigma, \\ y(\cdot,0) = & y_0 & \text{in} & \Omega, \\ y_t(\cdot,0) = & y_1 & \text{in} & \Omega, \end{cases} \tag{5.2}$$

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 conditions $y_0 \in H_0^1(\Omega)$ and $y_1 \in L^2(\Omega)$. The existence, uniqueness and regularity of the solution to the above optimal control problem (5.1)-(5.2) are well established [Lions, 1971]. By defining an appropriate Lagrange functional and making use of the strict convexity, the optimal solution pair (y,u) to (5.1)-(5.2) is shown to be completely

characterized by the unique solution triplet (y, p, u) to the following optimality system

$$\begin{cases} y_{tt} - \Delta y - u &= f \quad \text{in} \quad Q, \quad y = 0 \quad \text{on} \quad \Sigma, \\ y(\cdot, 0) &= y_0 \quad \text{in} \quad \Omega, \quad y_t(\cdot, 0) = y_1 \quad \text{in} \quad \Omega, \\ p_{tt} - \Delta p + y &= g \quad \text{in} \quad Q, \quad p = 0 \quad \text{on} \quad \Sigma, \\ p(\cdot, T) &= 0 \quad \text{in} \quad \Omega, \quad p_t(\cdot, T) = 0 \quad \text{in} \quad \Omega, \\ \gamma u - p &= 0 \quad \text{in} \quad Q, \end{cases} \tag{5.3}$$

where the state y evolves forward in time and the adjoint state p marches backward in time. The control $u = p/\gamma$ can be eliminated from the above optimality system, giving

$$\begin{cases} y_{tt} - \Delta y - p/\gamma &= f \quad \text{in} \quad Q, \quad y = 0 \quad \text{on} \quad \Sigma, \\ y(\cdot, 0) &= y_0 \quad \text{in} \quad \Omega, \quad y_t(\cdot, 0) = y_1 \quad \text{in} \quad \Omega, \\ p_{tt} - \Delta p + y &= g \quad \text{in} \quad Q, \quad p = 0 \quad \text{on} \quad \Sigma, \\ p(\cdot, T) &= 0 \quad \text{in} \quad \Omega, \quad p_t(\cdot, T) = 0 \quad \text{in} \quad \Omega. \end{cases} \tag{5.4}$$

It is well-known that the main challenge for solving (5.4) results from the fact that the state y and the adjoint state p are marching in opposite orientations. Its numerical discretizations will create an enormously huge system of equations as we have to resolve all time steps simultaneously [Heinkenschloss, 2005].

Different from elliptic and parabolic cases, there are few available results on fast computing of optimal control of wave equations. There are some developments of numerical algorithms for the optimal control of hyperbolic or wave equations [Kröner et al., 2011, Kröner, 2011a, Kröner, 2013, Luo et al., 2013, Kröner, 2011b, Kunisch and Wachsmuth, 2013b, Kunisch and Wachsmuth, 2013a, Kröner and Kunisch, 2014, Bucci, 1992, Zuazua, 2005, Gerdts et al., 2008, Gugat et al., 2009, Gugat and Grimm, 2011]. Some comprehensive and interesting results are given, for example, in [Kröner et al., 2011]. In their work,

the authors analyzed the superlinear convergence of the semismooth Newton method that is employed to treat the inequality control constraints in optimal control problems governed by the wave equation. The discretization is through finite element approach for distributed control, Neumann boundary control, and Dirichlet boundary control, respectively. The original second-order wave equation is formulated as a first-order system for their discretizations, in which the time variable is discretized by the Crank-Nicolson scheme based on the trapezoidal rule. However, there is no discussion on the implementation of proposal algorithms for solving the resultant discretized systems. Furthermore, the reformulated first-order system introduces two extra dependent variables, which increases the computational burdens. Another notable work can be found in [Luo et al., 2013]. In their approach, the authors applied the finite volume element method to the distributed control problems governed by second-order hyperbolic equations, where the optimal error estimates in certain norm were proved for the spatially semi-discrete optimality system, but the convergence of the full-discrete scheme is not seen. In the given numerical experiments, the spatial and temporal step sizes are chosen to satisfy the CFL condition that is not desirable for an efficient algorithm. For solving the discretized system, a nice fixed-point iterative algorithm is provided in [Rincon and Liu, 2003]. However, when the regularization (or penalization) parameter γ in the cost functional becomes very small, the approach for our underlying problem may suffer from slow convergence or even divergence. More recently, numerical methods were developed for the optimal control of nonlinear hyperbolic system with possible discontinuous solutions [Chertock et al., 2014, Herty et al., 2015]. However, the implementation of fast computing for solving the discretized optimality system has not been discussed. Although these second-order schemes are available in literature, to the best of our knowledge, the study of an efficient numerical implementation (fast solver) for the optimal control problem of wave equations has been seen yet.

Generally speaking, an efficient numerical implementation includes two steps: the first step is to seek a numerical algorithm that is not only convergent but also can provide a well-structured discretization, and the second step is to develop an efficient iteration for the obtained large algebraic systems. These two steps are inevitably correlated. If a high-order numerical scheme

is developed with a poor structure, then an efficient implementation will be very difficult, if it is not impossible. Therefore, in order to have an efficient computing, it is essential to develop the numerical schemes that can be easily adapted to the later construction of iterative linear solvers[Rees et al., 2010, Herzog and Sachs, 2010, Pearson et al., 2012, Pearson and Stoll, 2013, Saad, 2003] so that it can handle large-scale degrees of freedom and high dimension.

In this chapter we develop a new implicit central difference scheme for both time and spatial variables. The proposed numerical scheme for solving (5.4) is not only shown to be unconditionally stable but also to have a nice discretized structure. It is not required to satisfy the CFL condition that usually is necessary in classical theory for solving hyperbolic equations by standard explicit scheme[Strikwerda, 2004, LeVeque, 2007]. Based on our setting, we construct an effective preconditioned iterative solver for solving the resultant discretized linear system.

The chapter is organized as follows. In next section we give a standard explicit scheme in time with a central finite difference scheme in space for discretizing the optimality system (5.4) and illustrate the drawback for this approach. As a development, we present a new implicit scheme in time and the error estimate of the resulting full-discrete scheme in Section 5.3. Section 5.4 discusses the construction of an effective block upper triangular preconditioner by the well-known GMRES method that is suitable for solving the fully discretized linear system. Numerical experiments are performed in Section 5.5 to validate our theoretical outcome and to demonstrate the effectiveness of the proposed preconditioner. Finally, the chapter ends with concluding remarks in Section 5.6.

5.2 A STANDARD EXPLICIT CENTRAL DIFFERENCE SCHEME

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 Ω 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 $h = \max(h_1, h_2)$. 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

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

and the discrete Laplacian (in 2D)

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

We discretize the equations (5.4) by the explicit scheme in time with a standard five-point second order central difference discretization in space

$$\frac{Y^{n+1} - 2Y^n + Y^{n-1}}{\tau^2} - \Delta_h Y^n - P^n/\gamma = f^n, \quad n = 0, 1, 2, \cdots, N-1 \tag{5.5}$$

$$\frac{P^{n+1} - 2P^n + P^{n-1}}{\tau^2} - \Delta_h P^n + Y^n = g^n, \quad n = 1, 2, \cdots, N-1, N \tag{5.6}$$

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. Similarly notations are used for f^n and g^n. The initial conditions are derived based on Taylor expansions using (5.4) to represent $y_t t$ and p_{tt}, that is,

$$Y^0_{i,j} = y_0(\xi_i, \zeta_j), \quad Y^1_{i,j} = y_0(\xi_i, \zeta_j) + y_1(\xi_i, \zeta_j)\tau + \frac{\tau^2}{2}(\Delta y_0(\xi_i, \zeta_j) + f^0_{i,j} + \frac{1}{\gamma} P^0_{i,j}), \tag{5.7}$$

$$P^N_{i,j} = 0, \quad P^{N-1}_{i,j} = \frac{\tau^2}{2}(-Y^N_{i,j} + g^N_{i,j}), \tag{5.8}$$

where we have used the final conditions $p(\cdot, T) = 0$ and $p_t(\cdot, T) = 0$.

To illustrate the structure of the discretized system, we formulate the above explicit scheme

into a two-by-two block structured symmetric indefinite linear system

$$S_h \begin{bmatrix} y_h \\ p_h \end{bmatrix} := \begin{bmatrix} \check{I} & F_h^{\intercal} \\ F_h & -\hat{I}/\gamma \end{bmatrix} \begin{bmatrix} y_h \\ p_h \end{bmatrix} = \begin{bmatrix} g_h \\ f_h \end{bmatrix}, \tag{5.9}$$

where

$$F_h = \frac{1}{\tau^2} \begin{bmatrix} I & 0 & \cdots & 0 & 0 & 0 \\ -2I - \tau^2 \Delta_h & I & 0 & \cdots & 0 & 0 \\ I & -2I - \tau^2 \Delta_h & I & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & I & -2I - \tau^2 \Delta_h & I & 0 \\ 0 & 0 & \cdots & I & -2I - \tau^2 \Delta_h & I \end{bmatrix}, \tag{5.10}$$

$$\hat{I}_h = \begin{bmatrix} I/2 & 0 & 0 & 0 & \cdots & 0 \\ 0 & I & 0 & 0 & \cdots & 0 \\ 0 & 0 & I & 0 & \ddots & \vdots \\ 0 & 0 & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & I & 0 \\ 0 & 0 & \cdots & 0 & 0 & I \end{bmatrix}, \quad \check{I}_h = \begin{bmatrix} I & 0 & 0 & 0 & \cdots & 0 \\ 0 & I & 0 & 0 & \cdots & 0 \\ 0 & 0 & I & 0 & \ddots & \vdots \\ 0 & 0 & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & I & 0 \\ 0 & 0 & \cdots & 0 & 0 & I/2 \end{bmatrix}, \tag{5.11}$$

$$f_h = \begin{bmatrix} f^0/2 + y_1/\tau + (I/\tau^2 + \Delta_h/2)y_0 \\ f^1 - y_0/\tau^2 \\ \vdots \\ f^{N-2} \\ f^{N-1} \end{bmatrix}, \tag{5.12}$$

$$g_h = \begin{bmatrix} g^1 \\ g^2 \\ g^3 \\ \vdots \\ g^N/2 \end{bmatrix}, y_h = \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^{N-1} \\ y^N \end{bmatrix}, \text{and } p_h = \begin{bmatrix} p^0 \\ p^1 \\ p^2 \\ \vdots \\ p^{N-1} \end{bmatrix}. \tag{5.13}$$

Here I is an identity matrix of appropriate size and the vectors y_0, y_1, f^n, g^n, y^n, and p^n are the lexicographic ordering (vectorization) of the corresponding function approximations over spatial grid points. Notice that the submatrix F_h becomes very ill-conditioned when the CFL condition ($\tau \leq h$) does not hold, which is expected since it corresponds to the instability of explicit scheme applied to a single wave equation. Although, due to the coupling effects, the whole system seems to be well-conditioned regardless of the CFL condition, it is difficult to design efficient iterative solvers by existing method, such as by precoditioned GMRES [Saad, 2003], with the ill-conditioned F_h. We shall modify the above scheme in next section to remove the CFL condition and to overcome the ill-conditioned problem caused by the standard discretization.

5.3 A NEW IMPLICIT SCHEME AND ITS ERROR ESTIMATE

One critical observation is that the above explicit scheme in time does not help us to efficiently solve the coupled system by any time-marching algorithms as usually found in dealing

with a single wave equation. This motivates us to change the explicit scheme that relies on CFL condition to an implicit scheme so that restrictions on mesh size ratios can be eliminated. It is expected that implicit schemes are more suitable for developing robust iterative solvers. In this section, we introduce an implicit central difference scheme for optimality system, which not only allows us to show the convergence but also to construct an effective preconditioner of the discretized system.

We propose the following scheme (averaging the Laplacian term)

$$\frac{Y^{n+1}-2Y^n+Y^{n-1}}{\tau^2}-\frac{\Delta_h Y^{n+1}+\Delta_h Y^{n-1}}{2}-P^n/\gamma=f^n,\quad n=1,2,\cdots,N-1 \tag{5.14}$$

$$\frac{P^{n+1}-2P^n+P^{n-1}}{\tau^2}-\frac{\Delta_h P^{n+1}+\Delta_h P^{n-1}}{2}+Y^n=g^n,\quad n=1,2,\cdots,N-1 \tag{5.15}$$

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} are the discrete approximations of $y(\xi_i,\zeta_j,t_n)$ and $p(\xi_i,\zeta_j,t_n)$, respectively. Compared to the standard central difference scheme (5.5)-(5.8), we artificially introduce a second-order approximation over three consecutive time steps, i.e.,

$$\Delta_h Y^n=\frac{\Delta_h Y^{n+1}+\Delta_h Y^{n-1}}{2}+\frac{\tau^2\Delta_h}{2}\frac{\partial^2 y}{\partial t^2}(t_n)+O(\tau^4), \tag{5.16}$$

$$\Delta_h P^n=\frac{\Delta_h P^{n+1}+\Delta_h P^{n-1}}{2}+\frac{\tau^2\Delta_h}{2}\frac{\partial^2 p}{\partial t^2}(t_n)+O(\tau^4), \tag{5.17}$$

which maintains the same second order of accuracy as the standard central difference scheme does. The initial conditions are derived based on Taylor expansions up to the order $O(\tau^3)$, by using (5.4) to represent y_{tt} and p_{tt}, i.e.

$$Y^0_{i,j}=y_0(\xi_i,\zeta_j),\quad \left(1-\frac{\tau^2}{2}\Delta_h\right)Y^1_{i,j}=y_0(\xi_i,\zeta_j)+y_1(\xi_i,\zeta_j)\tau+\frac{\tau^2}{2}(f^0_{i,j}+\frac{1}{\gamma}P^0_{i,j}), \tag{5.18}$$

$$P^N_{i,j}=0,\qquad \left(1-\frac{\tau^2}{2}\Delta_h\right)P^{N-1}_{i,j}=\frac{\tau^2}{2}(-Y^N_{i,j}+g^N_{i,j}). \tag{5.19}$$

where we have used implicit schemes in approximating $Y^1_{i,j}$ and $P^{N-1}_{i,j}$. In Section 5.4 we will see that the implicit schemes used in (5.18)-(5.19) guarantees an effective preconditioner.

By denoting $D_h = I - \frac{\tau^2}{2}\Delta_h$, the above scheme can be formulated as a symmetric indefinite linear system

$$M_h \begin{bmatrix} y_h \\ p_h \end{bmatrix} := \begin{bmatrix} \check{I}_h & L_h^\intercal \\ L_h & -\hat{I}_h/\gamma \end{bmatrix} \begin{bmatrix} y_h \\ p_h \end{bmatrix} = \begin{bmatrix} g_h \\ f_h \end{bmatrix}, \tag{5.20}$$

where

$$L_h = \frac{1}{\tau^2} \begin{bmatrix} D_h & 0 & 0 & 0 & \cdots & 0 \\ -2I & D_h & 0 & 0 & \cdots & 0 \\ D_h & -2I & D_h & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & D_h & -2I & D_h & 0 \\ 0 & 0 & \cdots & D_h & -2I & D_h \end{bmatrix}, \tag{5.21}$$

$$f_h = \begin{bmatrix} f^0/2 + y_1/\tau + y_0/\tau^2 \\ f^1 - D_h y_0/\tau^2 \\ f^2 \\ \vdots \\ f^{N-2} \\ f^{N-1} \end{bmatrix}, \quad g_h = \begin{bmatrix} g^1 \\ g^2 \\ \vdots \\ g^{N-1} \\ g^N/2 \end{bmatrix}, \tag{5.22}$$

where the first two components of f_h are different from previous standard scheme. The matrix L_h

has a much smaller condition number than F_h in (5.9), which will be crucial to the development of efficient iterative solvers. In the following Table 5.1, we report the numerically estimated condition numbers of the explicit and implicit central difference scheme for the following Ex. 8 with $T = 2$ using MATLAB's build-in function condest. Notice here we have $\tau = 2h$, which of course violates the CFL condition ($\tau \leq h$). We see that F_h is highly ill-conditioned compared with L_h, which will inevitably incapacitate any numerical methods that rely on computing or approximating $F_h^{-1}v$ for solving the linear system (5.9). Our implicit scheme does not suffer from this drawback since the corresponding L_h is even much more well-conditioned than the whole system M_h. For our approach, we do need to compute $L_h^{-1}v$ for the implementation of a desirable preconditioner.

Table 5.1. The condition numbers of the explicit and implicit scheme for Ex. 8 ($T = 2, \gamma = 10^{-2}$).

	Explicit Scheme		Implicit Scheme	
(M, N)	cond(F_h)	cond(S_h)	cond(L_h)	cond(M_h)
(8,8)	1.59e9	1.53e3	1.54e2	8.23e2
(16,16)	2.47e18	6.57e3	6.97e2	2.34e3
(32,32)	5.00e36	2.85e4	1.55e3	8.05e3
(64,64)	2.01e73	1.26e5	9.96e3	3.33e4
(128,128)	3.26e146	5.56e5	3.71e4	1.29e5
(256,256)	8.55e292	2.44e6	8.74e4	5.49e5

We next present error estimates for the numerical solution given by the scheme (5.14)-(5.19). The discrete version of Poincare inequality [Jovanović and Süli, 2014] will be used, 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

$$\|y\| \leq C_0 \|\nabla_h y\| . \tag{5.23}$$

The following discrete version of integration by parts will also be used:

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

where the functions z, w are defined on the mesh points and vanish on the boundary $\partial\Omega$. We also need the following two lemmas in the proof of Theorem 5.3.3.

Lemma 5.3.1 (Lemma 5.1 of [Heywood and Rannacher, 1990], Discrete Gronwall's inequality)**.** *Let $\tau = T/N$. If $E^n \geq 0$ for $n = 0, 1, \cdots, N-1$ and $E^k \leq \alpha + \beta \sum_{n=0}^{k-1} \tau E^n$ for $0 \leq k \leq N-1$, then*

$$\max_{0 \leq n \leq N-1} E^n \leq C_{\beta,T}\, \alpha,$$

where the constant $C_{\beta,T}$ only depends on β and T.

Lemma 5.3.2. *For any function w defined on the mesh points of Ω vanishing on the boundary $\partial\Omega$, we have*

$$\|D_h^{-1} w\| \leq \|w\|, \tag{5.24}$$

$$\|D_h^{-1} w\| \leq \|D_h^{-1/2} w\|, \tag{5.25}$$

$$\|\nabla_h D_h^{-1} w\| \leq \|\nabla_h w\|, \tag{5.26}$$

$$\tau^2 \|\nabla_h D_h^{-1} w\|^2 \leq \frac{1}{2} \|w\|^2 \tag{5.27}$$

Proof. Since $-\Delta_h$ is symmetric and positive definite, we denote by ξ_j, $j = 1, \cdots, (M_1 - 1)(M_2 - 1)$, the orthonormal eigenfunctions of $-\Delta_h$ corresponding to the positive eigenvalues λ_j, $j = 1, \cdots, (M_1 - 1)(M_2 - 1)$, respectively. For any function $u = \sum_{j=1}^{(M_1-1)(M_2-1)} \alpha_j \xi_j$ we have

$$\begin{aligned}
\|\nabla_h D_h^{-1} u\|^2 &= (\nabla_h (1 - \tau^2 \Delta_h/2)^{-1} u, \nabla_h (1 - \tau^2 \Delta_h/2)^{-1} u) \\
&= (-\Delta_h (1 - \tau^2 \Delta_h/2)^{-1} u, (1 - \tau^2 \Delta_h/2)^{-1} u) \\
&= \left(\sum_{j=1}^{(M_1-1)(M_2-1)} \frac{\alpha_j \lambda_j}{1 + \tau^2 \lambda_j/2} \xi_j, \sum_{j=1}^{(M_1-1)(M_2-1)} \frac{\alpha_j}{1 + \tau^2 \lambda_j/2} \xi_j \right) \\
&= \sum_{j=1}^{(M_1-1)(M_2-1)} \frac{|\alpha_j|^2 \lambda_j}{(1 + \tau^2 \lambda_j/2)^2}
\end{aligned}$$

$$\leq \sum_{j=1}^{(M_1-1)(M_2-1)} |\alpha_j|^2 \lambda_j$$

$$= (-\Delta_h u, u) = \|\nabla_h u\|^2.$$

Similarly, we have

$$\|D_h^{-1} u\|^2 = \sum_{j=1}^{(M_1-1)(M_2-1)} \frac{|\alpha_j|^2}{(1+\tau^2\lambda_j/2)^2} \leq \sum_{j=1}^{(M_1-1)(M_2-1)} |\alpha_j|^2 = \|u\|^2,$$

$$\|D_h^{-1/2} u\|^2 = \sum_{j=1}^{(M_1-1)(M_2-1)} \frac{|\alpha_j|^2}{1+\tau^2\lambda_j/2} \leq \sum_{j=1}^{(M_1-1)(M_2-1)} |\alpha_j|^2 = \|u\|^2,$$

and

$$\tau^2 \|\nabla_h D_h^{-1} u\|^2 = \sum_{j=1}^{(M_1-1)(M_2-1)} \frac{|\alpha_j|^2 \tau^2 \lambda_j}{(1+\tau^2\lambda_j/2)^2} \leq \frac{1}{2} \sum_{j=1}^{(M_1-1)(M_2-1)} |\alpha_j|^2 = \frac{1}{2}\|u\|^2.$$

Therefore, $\|D_h^{-1} w\| = \|D_h^{-1/2} D_h^{-1/2} w\| \leq \|D_h^{-1/2} w\|$. □

Theorem 5.3.3. *Let $1 \leq d \leq 3$. Assume the solution $y, p \in C^{4,4}(\overline{Q})$. Then there exists a positive constant $C_* := C_*(\gamma, T)$ independent of h and τ such that*

$$\max_{1\leq n\leq N} \{\|Y^n - y^n\| + \|P^n - p^n\|\} \leq C_*(\tau^2 + h^2) \tag{5.28}$$

provided $\tau \leq \gamma^{\frac{1}{4}}$.

Proof. To simplify the notations, we denote by C_1, C_2, $\cdots$, positive constants which do not depend on τ, h, n or k in the following arguments.

Note that the exact solution $y_{i,j}^n = y(\xi_i, \zeta_j, t_n)$ and $p_{i,j}^n = p(\xi_i, \zeta_j, t_n)$ satisfy the equations

$$\frac{y^{n+1} - 2y^n + y^{n-1}}{\tau^2} - \frac{\Delta_h y^{n+1} + \Delta_h y^{n-1}}{2} - p^n/\gamma = f^n + F^n, \quad n = 1, 2, \cdots, N-1 \tag{5.29}$$

$$\frac{p^{n+1} - 2p^n + p^{n-1}}{\tau^2} - \frac{\Delta_h p^{n+1} + \Delta_h p^{n-1}}{2} + y^n = g^n + G^n, \quad n = 1, 2, \cdots, N-1 \tag{5.30}$$

and

$$\begin{aligned}
&y_{i,j}^0 = y_0(\xi_i, \zeta_j), \\
&\left(1 - \frac{\tau^2}{2}\Delta_h\right) y_{i,j}^1 = y_0(\xi_i, \zeta_j) + y_1(\xi_i, \zeta_j)\tau + \frac{\tau^2}{2}\left(f_{i,j}^0 + \frac{1}{\gamma} p_{i,j}^0\right) + F_{i,j}^0, \qquad (5.31) \\
&p_{i,j}^N = 0, \\
&\left(1 - \frac{\tau^2}{2}\Delta_h\right) p_{i,j}^{N-1} = -\frac{1}{2} y_{i,j}^N \tau^2 + \frac{1}{2} g_{i,j}^N \tau^2 + G_{i,j}^N, \qquad (5.32)
\end{aligned}$$

where (5.31) and (5.32) are derived via Taylor expansions by using (5.4) to represent y_{tt} and p_{tt}. Also, F^n and G^n denote the truncation errors, which satisfy

$$\begin{aligned}
&\|F^n\| + \|G^n\| \le C_1(\tau^2 + h^2) \quad \text{for} \;\; n = 1, 2, \cdots, N-1, \\
&\|F^0\| + \|G^N\| + \|\nabla_h F^0\| + \|\nabla_h G^N\| \le C_1(\tau^3 + \tau^2 h),
\end{aligned}$$

for some positive constant C_1 when the solution $y, p \in C^{4,4}(\overline{Q})$.

Let $e^n = Y^n - y^n$ and $\eta^n = P^n - p^n$. Then the difference between (5.14)-(5.19) and (5.29)-(5.32) gives

$$\frac{e^{n+1} - 2e^n + e^{n-1}}{\tau^2} - \frac{\Delta_h e^{n+1} + \Delta_h e^{n-1}}{2} - \eta^n/\gamma = -F^n, \tag{5.33}$$

$$\frac{\eta^{n+1} - 2\eta^n + \eta^{n-1}}{\tau^2} - \frac{\Delta_h \eta^{n+1} + \Delta_h \eta^{n-1}}{2} + e^n = -G^n, \tag{5.34}$$

for $n = 1, 2, \cdots, N-1$ and

$$e^0 = 0, \quad \left(1 - \frac{\tau^2}{2}\Delta_h\right) e^1 = \frac{1}{2\gamma}\eta^0\tau^2 - F^0, \tag{5.35}$$

$$\eta^N = 0, \quad \left(1 - \frac{\tau^2}{2}\Delta_h\right) \eta^{N-1} = -\frac{1}{2}e^N\tau^2 - G^N. \tag{5.36}$$

The discrete inner product of (5.33) and $e^{n+1} - e^{n-1}$ yields

$$\begin{aligned} &\frac{\|e^{n+1} - e^n\|^2 - \|e^n - e^{n-1}\|^2}{\tau^2} + \frac{\|\nabla_h e^{n+1}\|^2 - \|\nabla_h e^{n-1}\|^2}{2} - (e^{n+1} - e^{n-1}, \eta^n)/\gamma \\ &= -(F^n, e^{n+1} - e^{n-1}), \end{aligned} \tag{5.37}$$

and by summing up the equations for $n = 1, \cdots, N-1$, one can get

$$\begin{aligned} &\frac{\|e^N - e^{N-1}\|^2 - \|e^1 - e^0\|^2}{\tau^2} + \frac{1}{2}\|\nabla_h e^N\|^2 + \frac{1}{2}\|\nabla_h e^{N-1}\|^2 - \frac{1}{2}\|\nabla_h e^1\|^2 - \frac{1}{2}\|\nabla_h e^0\|^2 \\ &= \sum_{n=1}^{N-1} (e^{n+1} - e^{n-1}, \eta^n)/\gamma - \sum_{n=1}^{N-1} (F^n, e^{n+1} - e^{n-1}) \end{aligned} \tag{5.38}$$

which together with (5.35) implies that

$$\begin{aligned} &\frac{\|e^N - e^{N-1}\|^2}{\tau^2} + \frac{1}{2}\|\nabla_h e^N\|^2 + \frac{1}{2}\|\nabla_h e^{N-1}\|^2 \\ &= \sum_{n=1}^{N-1} (e^{n+1} - e^{n-1}, \eta^n)/\gamma - \sum_{n=1}^{N-1} (F^n, e^{n+1} - e^{n-1}) \\ &\quad + \left\| D_h^{-1}\left(\frac{1}{2\gamma}\eta^0\tau - F^0/\tau\right)\right\|^2 + \frac{\tau^2}{2}\left\|\nabla_h D_h^{-1}\left(\frac{1}{2\gamma}\eta^0\tau - F^0/\tau\right)\right\|^2. \end{aligned} \tag{5.39}$$

Similarly, the discrete inner product of (5.34) and $\eta^{n+1} - \eta^{n-1}$ gives

$$\begin{aligned} &\frac{\|\eta^{n+1} - \eta^n\|^2 - \|\eta^n - \eta^{n-1}\|^2}{\tau^2} + \frac{\|\nabla_h \eta^{n+1}\|^2 - \|\nabla_h \eta^{n-1}\|^2}{2} + (\eta^{n+1} - \eta^{n-1}, e^n) \\ &= -(G^n, \eta^{n+1} - \eta^{n-1}), \end{aligned} \tag{5.40}$$

and by summing up the equations for $n = 1, \cdots, N-1$, we have

$$\begin{aligned}
&\frac{\|\eta^1 - \eta^0\|^2}{\tau^2} + \frac{1}{2}\|\nabla_h \eta^0\|^2 + \frac{1}{2}\|\nabla_h \eta^1\|^2 \\
&= \sum_{n=1}^{N-1} (\eta^{n+1} - \eta^{n-1}, e^n) + \sum_{n=1}^{N-1} (G^n, \eta^{n+1} - \eta^{n-1}) \\
&\quad + \left\| D_h^{-1} \left(\frac{1}{2} e^N \tau + G^N/\tau \right) \right\|^2 + \frac{\tau^2}{2} \left\| \nabla_h D_h^{-1} \left(\frac{1}{2} e^N \tau + G^N/\tau \right) \right\|^2. \qquad (5.41)
\end{aligned}$$

By using Lemma 5.3.2, the sum of (5.41) and $\gamma\times$(5.39) implies

$$\begin{aligned}
&\frac{\gamma\|e^N - e^{N-1}\|^2 + \|\eta^1 - \eta^0\|^2}{\tau^2} + \frac{\gamma}{2}\|\nabla_h e^N\|^2 + \frac{\gamma}{2}\|\nabla_h e^{N-1}\|^2 + \frac{1}{2}\|\nabla_h \eta^0\|^2 + \frac{1}{2}\|\nabla_h \eta^1\|^2 \\
&= \sum_{n=1}^{N-1} (e^{n+1} - e^{n-1}, \eta^n) + \sum_{n=1}^{N-1} (\eta^{n+1} - \eta^{n-1}, e^n) \\
&\quad - \sum_{n=1}^{N-1} (\gamma F^n, e^{n+1} - e^{n-1}) + \sum_{n=1}^{N-1} (G^n, \eta^{n+1} - \eta^{n-1}) \\
&\quad + \gamma \left\| D_h^{-1} \left(\frac{\tau}{2\gamma} \eta^0 - F^0/\tau \right) \right\|^2 + \left\| D_h^{-1} \left(\frac{\tau}{2} e^N + G^N/\tau \right) \right\|^2 \\
&\quad + \frac{\gamma}{2} \left\| \nabla_h D_h^{-1} \left(\frac{\tau^2}{2\gamma} \eta^0 - F^0 \right) \right\|^2 + \frac{1}{2} \left\| \nabla_h D_h^{-1} \left(\frac{\tau^2}{2} e^N + G^N \right) \right\|^2 \\
&= (e^N, \eta^{N-1}) - (e^1, \eta^0) - \sum_{n=1}^{N-1} (\gamma F^n, e^{n+1} - e^{n-1}) + \sum_{n=1}^{N-1} (G^n, \eta^{n+1} - \eta^{n-1}) \\
&\quad + \gamma \left\| D_h^{-1} \left(\frac{\tau}{2\gamma} \eta^0 - F^0/\tau \right) \right\|^2 + \left\| D_h^{-1} \left(\frac{\tau}{2} e^N + G^N/\tau \right) \right\|^2 \\
&\quad + \frac{\gamma}{2} \left\| \nabla_h D_h^{-1} \left(\frac{\tau^2}{2\gamma} \eta^0 - F^0 \right) \right\|^2 + \frac{1}{2} \left\| \nabla_h D_h^{-1} \left(\frac{\tau^2}{2} e^N + G^N \right) \right\|^2 \\
&\le (e^N, -\frac{1}{2} D_h^{-1} e^N \tau^2 - D_h^{-1} G^N) - \left(\frac{1}{2\gamma} D_h^{-1} \eta^0 \tau^2 - D_h^{-1} F^0, \eta^0 \right) \\
&\quad + \left(\sum_{n=1}^{N-1} \tau \|\gamma F^n\|^2 \right)^{\frac{1}{2}} \left(\sum_{n=1}^{N-1} \tau \frac{\|e^{n+1} - e^n\|^2 + \|e^n - e^{n-1}\|^2}{\tau^2} \right)^{\frac{1}{2}} \\
&\quad + \left(\sum_{n=1}^{N-1} \tau \|G^n\|^2 \right)^{\frac{1}{2}} \left(\sum_{n=1}^{N-1} \tau \frac{\|\eta^{n+1} - \eta^n\|^2 + \|\eta^n - \eta^{n-1}\|^2}{\tau^2} \right)^{\frac{1}{2}} \\
&\quad + 2\gamma \left\| \frac{1}{2\gamma} D_h^{-1} \eta^0 \tau \right\|^2 + 2\gamma \|D_h^{-1} F^0/\tau\|^2 + 2 \left\| \frac{\tau}{2} D_h^{-1} e^N \right\|^2 + 2\|D_h^{-1} G^N/\tau\|^2
\end{aligned}$$

$$
\begin{aligned}
&+\frac{\tau^4}{4\gamma}\|\nabla_h\eta^0\|^2+\gamma\|\nabla_h F^0\|^2+\frac{\tau^4}{4}\|\nabla_h e^N\|^2+\|\nabla_h G^N\|^2\\
&\le \|e^N\|\|G^N\|-\frac{\tau^2}{2}\|D_h^{-1/2}e^N\|^2+\|F^0\|\|\eta^0\|-\frac{\tau^2}{2\gamma}\|D_h^{-1/2}\eta^0\|^2\\
&\quad+\Big(\sum_{n=1}^{N-1}\tau\|\gamma F^n\|^2\Big)^{\frac12}\Big(\sum_{n=1}^{N-1}\tau\frac{\|e^{n+1}-e^n\|^2+\|e^n-e^{n-1}\|^2}{\tau^2}\Big)^{\frac12}\\
&\quad+\Big(\sum_{n=1}^{N-1}\tau\|G^n\|^2\Big)^{\frac12}\Big(\sum_{n=1}^{N-1}\tau\frac{\|\eta^{n+1}-\eta^n\|^2+\|\eta^n-\eta^{n-1}\|^2}{\tau^2}\Big)^{\frac12}\\
&\quad+\frac{\tau^2}{2\gamma}\|D_h^{-1}\eta^0\|^2+2\gamma\big\|F^0/\tau\big\|^2+\frac{\tau^2}{2}\|D_h^{-1}e^N\|^2+2\big\|G^N/\tau\big\|^2\\
&\quad+\frac{\tau^4}{4\gamma}\|\nabla_h\eta^0\|^2+\gamma\|\nabla_h F^0\|^2+\frac{\tau^4}{4}\|\nabla_h e^N\|^2+\|\nabla_h G^N\|^2\\
&\le C_1(\tau^3+\tau^2h)(\|e^N\|+\|\eta^0\|)\\
&\quad+2(\sqrt{\gamma}+1)C_1\sqrt{T}(\tau^2+h^2)\Big(\sum_{n=0}^{N-1}\tau\frac{\gamma\|e^{n+1}-e^n\|^2+\|\eta^{n+1}-\eta^n\|^2}{\tau^2}\Big)^{\frac12}\\
&\quad+(2\gamma+2)C_1^2(\tau^2+\tau)^2+(\gamma+1)C_1^2(\tau^3+\tau^2h)^2+\frac{\tau^4}{4\gamma}\|\nabla_h\eta^0\|^2+\frac{\tau^4}{4}\|\nabla_h e^N\|^2\\
&\le 2C_1\max(C_0/\gamma,C_0)(\tau^3+\tau^2h)(\frac{\gamma}{2}\|\nabla_h e^N\|+\frac12\|\nabla_h\eta^0\|)\\
&\quad+2(\sqrt{\gamma}+1)C_1\sqrt{T}(\tau^2+h^2)\Big(\sum_{n=0}^{N-1}\tau\frac{\gamma\|e^{n+1}-e^n\|^2+\|\eta^{n+1}-\eta^n\|^2}{\tau^2}\Big)^{\frac12}\\
&\quad+(2\gamma+2)C_1^2(\tau^2+\tau)^2+(\gamma+1)C_1^2(\tau^3+\tau^2h)^2+\frac{\tau^4}{4\gamma}\|\nabla_h\eta^0\|^2+\frac{\tau^4}{4}\|\nabla_h e^N\|^2\\
&\le [4C_1\max(C_0/\gamma,C_0)\tau+2(\sqrt{\gamma}+1)C_1\sqrt{T}](\tau^2+h^2)\Big\{\frac{\gamma}{2}\|\nabla_h e^N\|+\frac12\|\nabla_h\eta^0\|\\
&\qquad\qquad+\Big(\sum_{n=0}^{N-1}\tau\frac{\gamma\|e^{n+1}-e^n\|^2+\|\eta^{n+1}-\eta^n\|^2}{\tau^2}\Big)^{\frac12}\Big\}\\
&\quad+(2\gamma+2)C_1^2(\tau^2+\tau)^2+(\gamma+1)C_1^2(\tau^3+\tau^2h)^2+\frac{\tau^4}{4\gamma}\|\nabla_h\eta^0\|^2+\frac{\tau^4}{4}\|\nabla_h e^N\|^2\\
&\le C_2(\tau^2+h^2)E_{\tau,h}+C_2(\tau^4+h^4)+\frac{\tau^4}{4\gamma}\|\nabla_h\eta^0\|^2+\frac{\tau^4}{4}\|\nabla_h e^N\|^2,
\end{aligned}
\tag{5.42}
$$

where C_2 is some positive constant and

$$
E_{\tau,h}:=\max_{0\le n\le N-1}E^n_{\tau,h} \tag{5.43}
$$

with

$$E^n_{\tau,h} := \Big(\frac{\gamma\|e^{n+1}-e^n\|^2 + \|\eta^{n+1}-\eta^n\|^2}{\tau^2} + \frac{\gamma}{2}\|\nabla_h e^{n+1}\|^2 + \frac{\gamma}{2}\|\nabla_h e^n\|^2 + \frac{1}{2}\|\nabla_h \eta^{n+1}\|^2 + \frac{1}{2}\|\nabla_h \eta^n\|^2 \Big)^{\frac{1}{2}}. \tag{5.44}$$

When $\tau \le \gamma^{\frac{1}{4}}$, the two terms $\frac{\tau^4}{4\gamma}\|\nabla_h \eta^0\|^2$ and $\frac{\tau^4}{4}\|\nabla_h e^N\|^2$ are eliminated by the left-handed side, and the last inequality thus reduces to

$$\begin{aligned} &\frac{\gamma\|e^N - e^{N-1}\|^2 + \|\eta^1-\eta^0\|^2}{\tau^2} + \frac{\gamma}{4}\|\nabla_h e^N\|^2 + \frac{\gamma}{4}\|\nabla_h e^{N-1}\|^2 + \frac{1}{4}\|\nabla_h \eta^0\|^2 + \frac{1}{4}\|\nabla_h \eta^1\|^2 \\ &\le C_3(\tau^2+h^2)E_{\tau,h} + C_3(\tau^4+h^4). \end{aligned} \tag{5.45}$$

By using the last inequality, one can convert the analysis of the forward and backward boundary value problem to the analysis of an initial-value problem, where the initial errors e^0, e^1, η^0 and η^1 are well controlled.

Let $1 \le k \le N-1$ be fixed and sum up (5.40) for $n = 1, \cdots, k$. Then we obtain that

$$\begin{aligned} &\frac{\|\eta^{k+1}-\eta^k\|^2 - \|\eta^1-\eta^0\|^2}{\tau^2} + \frac{\|\nabla_h \eta^{k+1}\|^2}{2} + \frac{\|\nabla_h \eta^k\|^2}{2} - \frac{\|\nabla_h \eta^1\|^2}{2} - \frac{\|\nabla_h \eta^0\|^2}{2} \\ &= -\sum_{n=1}^{k}(\eta^{n+1}-\eta^{n-1}, e^n) - \sum_{n=1}^{k}(G^n, \eta^{n+1}-\eta^{n-1}). \end{aligned} \tag{5.46}$$

Summing up (5.37) for $n = 1, \cdots, k$, we derive

$$\begin{aligned} &\frac{\|e^{k+1}-e^k\|^2 - \|e^1-e^0\|^2}{\tau^2} + \frac{1}{2}\|\nabla_h e^{k+1}\|^2 + \frac{1}{2}\|\nabla_h e^k\|^2 - \frac{1}{2}\|\nabla_h e^1\|^2 - \frac{1}{2}\|\nabla_h e^0\|^2 \\ &= \sum_{n=1}^{k}(e^{n+1}-e^{n-1}, \eta^n)/\gamma - \sum_{n=1}^{k}(F^n, e^{n+1}-e^{n-1}). \end{aligned} \tag{5.47}$$

Adding (5.46) and $\gamma\times$(5.47) and using (5.45), we obtain (note $e^0=0$ and $\tau^4\le\gamma$)

$$
\begin{aligned}
&\frac{\gamma\|e^{k+1}-e^k\|^2+\|\eta^{k+1}-\eta^k\|^2}{\tau^2}+\frac{\gamma}{2}\|\nabla_h e^{k+1}\|^2+\frac{\gamma}{2}\|\nabla_h e^k\|^2+\frac{1}{2}\|\nabla_h\eta^{k+1}\|^2+\frac{1}{2}\|\nabla_h\eta^k\|^2\\
&=\frac{\gamma\|e^1-e^0\|^2+\|\eta^1-\eta^0\|^2}{\tau^2}+\frac{\gamma}{2}\|\nabla_h e^1\|^2+\frac{\gamma}{2}\|\nabla_h e^0\|^2+\frac{1}{2}\|\nabla_h\eta^1\|^2+\frac{1}{2}\|\nabla_h\eta^0\|^2\\
&\quad+\sum_{n=1}^{k}(e^{n+1}-e^{n-1},\eta^n)-\sum_{n=1}^{k}(\gamma F^n,e^{n+1}-e^{n-1})\\
&\quad-\sum_{n=1}^{k}(\eta^{n+1}-\eta^{n-1},e^n)-\sum_{n=1}^{k}(G^n,\eta^{n+1}-\eta^{n-1})\\
&=\frac{\gamma}{\tau^2}\left\|D_h^{-1}\left(\frac{1}{2\gamma}\eta^0\tau^2-F^0\right)\right\|^2+\frac{\gamma}{2}\left\|\nabla_h D_h^{-1}\left(\frac{1}{2\gamma}\eta^0\tau^2-F^0\right)\right\|^2\\
&\quad+\frac{\|\eta^1-\eta^0\|^2}{\tau^2}+\frac{1}{2}\|\nabla_h\eta^1\|^2+\frac{1}{2}\|\nabla_h\eta^0\|^2\\
&\quad+\sum_{n=1}^{k}(e^{n+1}-e^{n-1},\eta^n)-\sum_{n=1}^{k}(\gamma F^n,e^{n+1}-e^{n-1})\\
&\quad-\sum_{n=1}^{k}(\eta^{n+1}-\eta^{n-1},e^n)-\sum_{n=1}^{k}(G^n,\eta^{n+1}-\eta^{n-1})\\
&\le\frac{\tau^2}{4\gamma}C_0^2\|\nabla_h\eta^0\|^2+\gamma C_1^2(\tau^2+\tau h)^2+\frac{\tau^4}{8\gamma}\|\nabla_h\eta^0\|^2+\frac{C_1^2\gamma}{2}(\tau^3+\tau^2h)^2+\frac{\|\eta^1-\eta^0\|^2}{\tau^2}\\
&\quad+\frac{1}{2}\|\nabla_h\eta^1\|^2+\frac{1}{2}\|\nabla_h\eta^0\|^2+\sum_{n=1}^{k}(e^{n+1}-e^{n-1},\eta^n)-\sum_{n=1}^{k}(\gamma F^n,e^{n+1}-e^{n-1})\\
&\quad-\sum_{n=1}^{k}(\eta^{n+1}-\eta^{n-1},e^n)-\sum_{n=1}^{k}(G^n,\eta^{n+1}-\eta^{n-1})\\
&\le\frac{1}{4\sqrt{\gamma}}C_0^2\|\nabla_h\eta^0\|^2+\gamma C_1^2(\tau^2+\tau h)^2+\frac{1}{8}\|\nabla_h\eta^0\|^2+\frac{C_1^2\gamma^{3/2}}{2}(\tau^2+\tau h)^2\\
&\quad+\frac{\|\eta^1-\eta^0\|^2}{\tau^2}+\frac{1}{2}\|\nabla_h\eta^1\|^2+\frac{1}{2}\|\nabla_h\eta^0\|^2+\sum_{n=1}^{k}(e^{n+1}-e^{n-1},\eta^n)-\sum_{n=1}^{k}(\gamma F^n,e^{n+1}-e^{n-1})\\
&\quad-\sum_{n=1}^{k}(\eta^{n+1}-\eta^{n-1},e^n)-\sum_{n=1}^{k}(G^n,\eta^{n+1}-\eta^{n-1})\\
&\le C_4(\tau^2+h^2)E_{\tau,h}+C_4(\tau^4+h^4)\\
&\quad+\left(\sum_{n=1}^{k}\tau\gamma\|F^n\|^2\right)^{\frac{1}{2}}\left(\sum_{n=1}^{k}\tau\frac{\gamma\|e^{n+1}-e^n\|^2+\gamma\|e^n-e^{n-1}\|^2}{\tau^2}\right)^{\frac{1}{2}}\\
&\quad+\left(\sum_{n=1}^{k}\tau\|G^n\|^2\right)^{\frac{1}{2}}\left(\sum_{n=1}^{k}\tau\frac{\|\eta^{n+1}-\eta^n\|^2+\|\eta^n-\eta^{n-1}\|^2}{\tau^2}\right)^{\frac{1}{2}}
\end{aligned}
$$

$$
\begin{aligned}
&+ T\sum_{n=1}^{k}\tau\big(\|\eta^n\|^2/\gamma + \|e^n\|^2\big)\\
&+ \frac{1}{4T}\sum_{n=1}^{k}\tau\bigg(\frac{\gamma\|e^{n+1}-e^n\|^2+\gamma\|e^n-e^{n-1}\|^2}{\tau^2}+\frac{\|\eta^{n+1}-\eta^n\|^2+\|\eta^n-\eta^{n-1}\|^2}{\tau^2}\bigg)\\
&\le C_4(\tau^2+h^2)E_{\tau,h} + C_4(\tau^4+h^4) + C_4(\tau^2+h^2)E_{\tau,h}\\
&+ \frac{2C_0^2T}{\gamma}\sum_{n=1}^{k}\tau\bigg(\frac{1}{2}\|\nabla_h\eta^n\|^2+\frac{\gamma}{2}\|\nabla_h e^n\|^2\bigg)+\frac{1}{2T}\sum_{n=0}^{k}\tau\bigg(\frac{\gamma\|e^{n+1}-e^n\|^2+\|\eta^{n+1}-\eta^n\|^2}{\tau^2}\bigg)\\
&\le 2C_4(\tau^2+h^2)E_{\tau,h} + C_4(\tau^4+h^4) + \frac{1}{2}\frac{\gamma\|e^{k+1}-e^k\|^2+\|\eta^{k+1}-\eta^k\|^2}{\tau^2}\\
&+ \bigg(\frac{1}{2T}+\frac{2C_0^2T}{\gamma}\bigg)\sum_{n=0}^{k-1}\tau\bigg(\frac{\gamma\|e^{n+1}-e^n\|^2+\|\eta^{n+1}-\eta^n\|^2}{\tau^2}+\frac{\gamma}{2}\|\nabla_h e^{n+1}\|^2+\frac{1}{2}\|\nabla_h\eta^{n+1}\|^2\bigg).
\end{aligned}
\tag{5.48}
$$

By moving the term $\frac{1}{2}\frac{\gamma\|e^{k+1}-e^k\|^2+\|\eta^{k+1}-\eta^k\|^2}{\tau^2}$ to the left-handed side and using the definition of $E^n_{\tau,h}$ we reduce (5.48) to

$$
|E^k_{\tau,h}|^2 \le 4C_4(\tau^2+h^2)E_{\tau,h} + 2C_4(\tau^4+h^4) + \bigg(\frac{1}{T}+\frac{4C_0^2T}{\gamma}\bigg)\sum_{n=0}^{k-1}\tau|E^n_{\tau,h}|^2, \tag{5.49}
$$

which holds for any $1 \le k \le N-1$. By applying discrete Gronwall's inequality, we derive that

$$
\begin{aligned}
|E_{\tau,h}|^2 &\le C_5(\tau^2+h^2)E_{\tau,h} + C_5(\tau^4+h^4)\\
&\le \frac{1}{2}|E_{\tau,h}|^2 + \frac{1}{2}C_5^2(\tau^2+h^2)^2 + C_5(\tau^4+h^4),
\end{aligned}
\tag{5.50}
$$

which further implies

$$
|E_{\tau,h}|^2 \le C_5^2(\tau^2+h^2)^2 + 2C_5(\tau^4+h^4). \tag{5.51}
$$

Finally, by applying the discrete Poincare inequality, the proof of Theorem 5.3.3 is thus completed.

□

5.4 A FAST PRECONDITIONED ITERATIVE SOLVER

Next we are ready to utilize the preconditioned Krylov subspace methods, such as the GMRES method [Saad, 2003], to solve the symmetric indefinite sparse linear system (5.20), i.e.,

$$M_h \begin{bmatrix} y_h \\ p_h \end{bmatrix} := \begin{bmatrix} \check{I}_h & L_h^{\intercal} \\ L_h & -\hat{I}_h/\gamma \end{bmatrix} \begin{bmatrix} y_h \\ p_h \end{bmatrix} = \begin{bmatrix} g_h \\ f_h \end{bmatrix}. \tag{5.52}$$

The review paper [Benzi et al., 2005] summarized many modern numerical methods for the above two-by-two block sparse linear system with a saddle point structure. One crucial task is to find an effective and efficient preconditioner which can speed up the convergence under GMRES approach by altering the spectrum distribution of the original system in a desirable way. Inspired by the framework presented in [Schöberl and Zulehner, 2007, Sun and Liu, 2010], we construct the following symmetric indefinite constrained preconditioner

$$P_h = \begin{bmatrix} 0 & L_h^{\intercal} \\ L_h & -\hat{I}_h/\gamma \end{bmatrix},$$

where L_h has a block upper triangular structure with the same diagonal block D_h. Here $L^{-1}v$ can be quickly computed by applying a block forward substitution as well as the well-known FFT algorithm to solve each diagonal block. In particular, the preconditioning step $P^{-1}v$ can be done with $NM\log(M)$ operations for 1D case and with $NM_1M_2\log(M_1M_2)$ operations for 2D case. From the expressions of L_h we know that P_h is nonsingular. Moreover, the right preconditioned system is given by

$$M_h P_h^{-1} = \begin{bmatrix} I_h + \gamma^{-1}\check{I}_h L_h^{-1}\hat{I}_h L_h^{-T} & \check{I}_h L_h^{-1} \\ 0 & I_h \end{bmatrix}.$$

Clearly, half of the eigenvalues of $M_h P_h^{-1}$ are ones, while the remaining half are determined by

$$R_h := (I_h + \gamma^{-1} \check{I}_h L_h^{-1} \hat{I}_h L_h^{-T}).$$

By exploring the connection between the matrices L_h^{-1} and L_h^{-T} and the underlying discretized linear system (5.14)-(5.15), we are able to show the following theorem, which implies that all eigenvalues of the preconditioned coefficient matrix $M_h P_h^{-1}$ are real numbers and they are uniformly greater than one and less than an upper bound depending only on γ and T.

Theorem 5.4.1. *Let $\lambda(R_h)$ be any eigenvalue of R_h, then $\lambda(R_h) \in \mathbb{R}$ and*

$$1 < \lambda(R_h) < 1 + \kappa/\gamma,$$

where κ is a positive constant independent of τ and h.

Proof. Using the fact that $\lambda(AB) = \lambda(BA)$, we get

$$\lambda(\check{I}_h L_h^{-1} \hat{I}_h L_h^{-T}) = \lambda(\check{I}_h^{1/2} L_h^{-1} \hat{I}_h^{1/2} \hat{I}_h^{1/2} L_h^{-T} \check{I}_h^{1/2}) = \lambda\left((\check{I}_h^{1/2} L_h^{-1} \hat{I}_h^{1/2})(\check{I}_h^{1/2} L_h^{-1} \hat{I}_h^{1/2})^\intercal\right), \tag{5.53}$$

which indicates $\lambda(\check{I}_h L_h^{-1} \hat{I}_h L_h^{-T})$ is a real number and so is $\lambda(R_h)$. To show the lower bound, we need to invoke the fact that $\lambda(AA^\intercal) > 0$ for any nonsingular matrix A. Obviously, the matrix $(\check{I}_h^{1/2} L_h^{-1} \hat{I}_h^{1/2})$ is nonsingular and thus it follows

$$\lambda(\check{I}_h L_h^{-1} \hat{I}_h L_h^{-T}) = \lambda\left((\check{I}_h^{1/2} L_h^{-1} \hat{I}_h^{1/2})(\check{I}_h^{1/2} L_h^{-1} \hat{I}_h^{1/2})^\intercal\right) > 0. \tag{5.54}$$

To prove the upper bound of the eigenvalues, we first prove the boundedness of the matrix norm $\|L_h^{-T}\|_{\infty,2}$ induced by the following vector norm

$$\|\boldsymbol{q}\|_{\infty,2} := \max_{1 \le k \le N} \|q^n\|,$$

where $\boldsymbol{q} = (q^1, q^2, \cdots, q^N)^\intercal$. Let $\boldsymbol{\psi} := (\psi^0, \psi^1, \cdots, \psi^{N-2}, \psi^{N-1})^\intercal$ be a solution the linear system

$$L_h^\intercal \boldsymbol{\psi} = \boldsymbol{q}, \tag{5.55}$$

we need to show

$$\|\boldsymbol{\psi}\|_{\infty,2} = \|L_h^{-T}\boldsymbol{q}\|_{\infty,2} \le C\|\boldsymbol{q}\|_{\infty,2}. \tag{5.56}$$

Recall that for the corresponding finite difference discretizations (5.15) of $L_h^\intercal$, the solution $\boldsymbol{\psi}$ solves

$$\frac{\psi^{n+1} - 2\psi^n + \psi^{n-1}}{\tau^2} - \frac{\Delta_h\psi^{n+1} + \Delta_h\psi^{n-1}}{2} = q^n, \quad n = 1, 2, \cdots, N-1 \tag{5.57}$$

with the initial conditions $\psi^{N-1} = \tau^2 D_h^{-1} q^N/2$ and $\psi^N = 0$.

We consider the discrete inner product of (5.57) with $\psi^{n+1} - \psi^{n-1}$, which gives

$$\frac{\|\psi^{n+1} - \psi^n\|^2 - \|\psi^n - \psi^{n-1}\|^2}{\tau^2} + \frac{\|\nabla_h\psi^{n+1}\|^2 - \|\nabla_h\psi^{n-1}\|^2}{2} = (q^n, \psi^{n+1} - \psi^{n-1}), \tag{5.58}$$

and by summing up the equations for $n = k, \cdots, N-1$, we obtain

$$\begin{aligned}
&\frac{\|\psi^k - \psi^{k-1}\|^2}{\tau^2} + \frac{\|\nabla_h\psi^k\|^2 + \|\nabla_h\psi^{k-1}\|^2}{2} \\
&= \sum_{n=k}^{N-1} (q^n, \psi^{n+1} - \psi^{n-1}) + \frac{\|\psi^N - \psi^{N-1}\|^2}{\tau^2} + \frac{\|\nabla_h\psi^N\|^2 + \|\nabla_h\psi^{N-1}\|^2}{2} \\
&\le \sum_{n=k}^{N-1} \frac{\tau\|q^n\|^2}{2} + \frac{1}{2}\sum_{n=k}^{N} \tau\left(\frac{\|\psi^n - \psi^{n-1}\|^2}{\tau^2} + \frac{\|\nabla_h\psi^n\|^2 + \|\nabla_h\psi^{n-1}\|^2}{2}\right) \\
&\quad + \frac{\tau^2\|D_h^{-1}q^N\|^2}{4} + \frac{\tau^4\|\nabla_h D_h^{-1}q^N\|^2}{4} \\
&\le \frac{\tau^2\|q^N\|^2}{2} + \sum_{n=k}^{N-1} \frac{\tau\|q^n\|^2}{2} + \frac{1}{2}\sum_{n=k}^{N} \tau\left(\frac{\|\psi^n - \psi^{n-1}\|^2}{\tau^2} + \frac{\|\nabla_h\psi^n\|^2 + \|\nabla_h\psi^{n-1}\|^2}{2}\right),
\end{aligned} \tag{5.59}$$

where we have used Lemma 5.3.2 in the last step. By applying Gronwall's inequality, we derive

$$\max_{1\le k\le N}\left(\frac{\|\psi^k-\psi^{k-1}\|^2}{\tau^2}+\frac{\|\nabla_h\psi^k\|^2+\|\nabla_h\psi^{k-1}\|^2}{2}\right)\le C_6\sum_{n=k}^{N}\tau\|q^n\|^2\le C_6\max_{1\le k\le N}\|q^n\|^2, \quad (5.60)$$

where C_6 is some positive constant which is independent of τ and h (but may depend on T). The last inequality above, together with (5.23), yields

$$\|\boldsymbol{\psi}\|_{\infty,2}=\|L^{-T}\boldsymbol{q}\|_{\infty,2}\le C_7\|\boldsymbol{q}\|_{\infty,2},$$

that is $\|L^{-T}\|_{\infty,2}\le C_7$ for some positive constant C_7. Similarly, one can show that $\|L^{-1}\|_{\infty,2}\le C_8$ also holds for some positive constant C_8. Moreover, it is obvious that $\|\hat{I}\|_{\infty,2}=\|\check{I}\|_{\infty,2}\le 1$. Therefore,

$$\lambda(\check{I}_hL_h^{-1}\hat{I}_hL_h^{-T})\le\|\check{I}_hL_h^{-1}\hat{I}_hL_h^{-T}\|_{\infty,2}\le C_7C_8=:\kappa,$$

where κ is a positive constant that is independent of τ and h. The proof is thus completed.

To illustrate the effect of our above theoretical estimates, in the following Fig. 5.1 and Fig. 5.2, we plot the numerically computed eigenvalues of M_h and $M_hP_h^{-1}$ for Ex. 8 with $\gamma=10^{-2}$ using $M=N=16$ and $M=N=32$, respectively. As anticipated, the eigenvalues of preconditioned systems are highly concentrated around one within a uniformly bounded interval, which reasonably envisions a fast convergence of the preconditioned GMRES method. Notice that the eigenvalue distributions of $M_hP_h^{-1}$ perfectly verified our estimated uniformly bounds. Such desirable clustered spectrum distributions after preconditioning are possible only with our implicit scheme, which is not readily achievable for the standard explicit scheme. According to our above estimates, the preconditioned GMRES method may show a slower convergence rate as the regularization parameter γ decreases to zero, but this is an inherent problem due to the very weak convexity of the underlying cost function. How to come up with an effective and regularization parameter robust preconditioner in such a case is another active and widely open research topic, with many recent contributions [Stoll and Wathen, 2012, Porcelli et al., 2014,

Schiela and Ulbrich, 2014] as well as references therein.

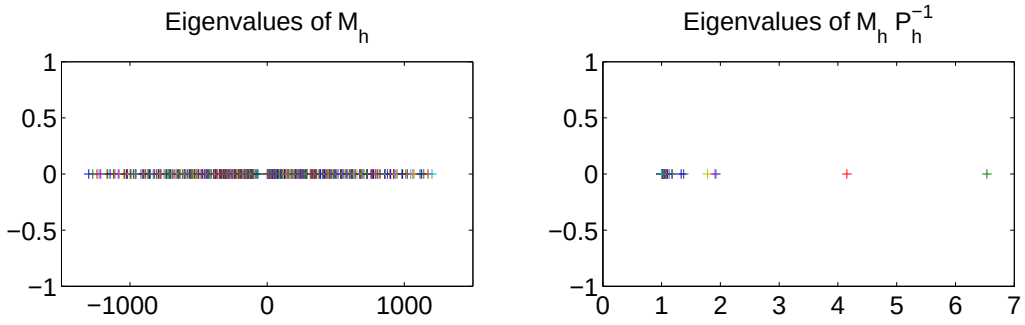

Figure 5.1. Eigenvalue distributions of M_h and $M_h P_h^{-1}$ in Ex. 8 ($M = N = 16$)

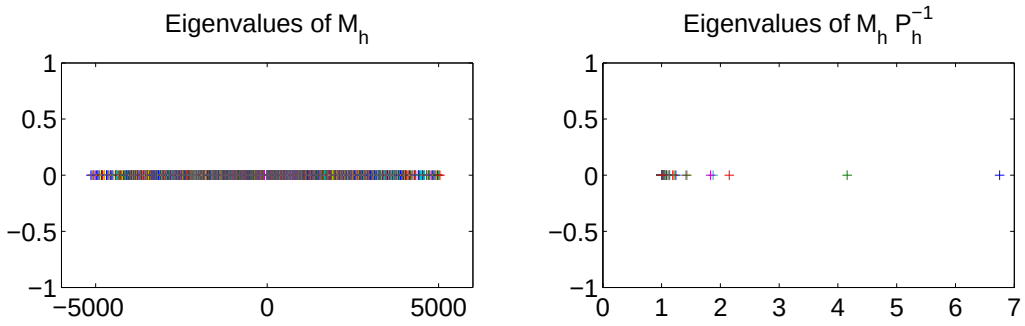

Figure 5.2. Eigenvalue distributions of M_h and $M_h P_h^{-1}$ in Ex. 8 ($M = N = 32$)

5.5 NUMERICAL EXAMPLES

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 (in seconds) 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 $L^\infty(L^2)$ norm $\|\cdot\|_{L_\tau^\infty(L_h^2)}$.

We first compute the discrete $L^\infty(L^2)$ norms of state and adjoint state approximation errors

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

and then estimate the experimental order of accuracy by compute 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. For initialization of the iterative methods, the state y and the adjoint state p are set to be zero, and the stopping criterion is chosen to be

$$\frac{\sqrt{\|r_y^{(k)}\|^2 + \|r_p^{(k)}\|^2}}{\sqrt{\|r_y^{(0)}\|^2 + \|r_p^{(0)}\|^2}} \leq \text{tol},$$

where $r_y^{(k)}$ and $r_p^{(k)}$ denote the residuals after k-th iteration. In our numerical simulations, according to the level of discretization errors as well as the regularization parameter γ, we set $\text{tol} = \gamma \times 10^{-6}$ and $\text{tol} = \gamma \times 10^{-4}$ for 1D and 2D examples, respectively. Note that in the following numerical simulations the CFL condition does not hold since we choose $\tau/h = 2$.

Example 8. Let $\Omega = (0, 1)$ and $T = 2$. Choose $y_0(x) = \sin(\pi x)$, $y_1(x) = 0$,

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

and

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

such that the exact solution is

$$y(x, t) = \sin(\pi x)\cos(\pi t) \quad \text{and} \quad p(x, t) = \sin(\pi x)(t - T)^2.$$

The numerical results of our implicit scheme solving by the preconditioned GMRES method with regularization parameter $\gamma = 10^{-2}$ and $\gamma = 10^{-4}$ are reported in Table 5.2 and 5.3, respectively. The implicit scheme delivers a clear second-order accuracy, which validates our proved error estimates for the implicit scheme. The required number of iterations for achieving convergence criterion is independent of mesh size and the computational CPU time grows roughly as a linearithmic function ($O(m\log(m))$) with respect to the degrees of freedom m, which shows the excellent effectiveness of our proposed preconditioner. However, comparing the column 'Iter' in Table 5.2 and 5.3, it does cost more iterations as γ decreasing, which is reasonable according to our previous discussion.

For comparison, we also give the corresponding results of the standard explicit scheme solving by MATLAB's backslash sparse direct solver in Tables 5.4 and 5.5. Notice that the standard explicit scheme provides slightly better approximations than our implicit scheme providing the same discretization step sizes, which is reasonable since the implicit scheme introduced extra truncation error terms as in (5.16-5.17). However, our proposed implicit scheme with preconditioned GMRES is computationally more efficient than the standard explicit scheme with the sparse direct solver when more accuracy is required. We remark that MATLAB's sparse direct solver is highly optimized and robust. Hence it is reliable to use it as a benchmark when we have no other available iterative solvers for the comparison. For such 1D problems, it seems that sparse direct solver still has certain marginal advantage in CPU time when the mesh size is not very small, but this is not the case when handling 2D or 3D problems. Furthermore, the corresponding preconditioned GMRES method does not work for the standard explicit scheme due to the highly ill-conditioned matrix F_h.

Example 9. Let $\Omega = (0,1)^2$ and $T = 2$. Choose

$$y_0(x_1,x_2) = \sin(\pi x_1)\sin(\pi x_2), \quad y_1(x) = \sin(\pi x_1)\sin(\pi x_2),$$

$$f = (1+2\pi^2)e^t\sin(\pi x_1)\sin(\pi x_2) - (t-T)^2\sin(\pi x_1)\sin(\pi x_2)/\gamma,$$

Table 5.2. Results for Ex. 8 with $\gamma = 10^{-2}$ (Implicit scheme with preconditioned GMRES).

(M, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	2.70e-02	–	1.59e-03	–	8	0.067
(64,64)	6.76e-03	2.00	4.00e-04	1.99	8	0.142
(128,128)	1.69e-03	2.00	1.00e-04	2.00	8	0.332
(256,256)	4.23e-04	2.00	2.50e-05	2.00	8	0.851
(512,512)	1.06e-04	2.00	6.26e-06	2.00	8	2.661
(1024,1024)	2.64e-05	2.00	1.56e-06	2.00	8	7.655
(2048,2048)	6.61e-06	2.00	3.91e-07	2.00	8	31.419

Table 5.3. Results for Ex. 8 with $\gamma = 10^{-4}$ (Implicit scheme with preconditioned GMRES).

(M, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(32,32)	2.76e-02	–	8.38e-05	–	21	0.156
(64,64)	6.92e-03	2.00	2.19e-05	1.94	20	0.344
(128,128)	1.73e-03	2.00	5.53e-06	1.99	21	0.836
(256,256)	4.33e-04	2.00	1.38e-06	2.00	21	2.581
(512,512)	1.08e-04	2.00	3.40e-07	2.02	21	7.385
(1024,1024)	2.71e-05	2.00	7.98e-08	2.09	21	24.218
(2048,2048)	6.77e-06	2.00	1.80e-08	2.15	22	116.744

Table 5.4. Results for Ex. 8 with $\gamma = 10^{-2}$ (Explicit scheme with sparse direct solver).

(M, N)	e_y^h	Order	e_p^h	Order	CPU
(32,32)	6.75e-03	–	1.39e-03	–	0.006
(64,64)	1.69e-03	2.00	3.48e-04	1.99	0.031
(128,128)	4.21e-04	2.00	8.71e-05	2.00	0.270
(256,256)	1.05e-04	2.00	2.18e-05	2.00	1.347
(512,512)	2.63e-05	2.00	5.44e-06	2.00	15.540
(1024,1024)	6.58e-06	2.00	1.36e-06	2.00	156.712

Table 5.5. Results for Ex. 8 with $\gamma = 10^{-4}$ (Explicit scheme with sparse direct solver).

(M, N)	e_y^h	Order	e_p^h	Order	CPU
(32,32)	1.61e-02	–	1.78e-04	–	0.006
(64,64)	4.03e-03	2.00	4.78e-05	1.90	0.029
(128,128)	1.01e-03	2.00	1.22e-05	1.97	0.148
(256,256)	2.52e-04	2.00	3.06e-06	1.99	0.772
(512,512)	6.30e-05	2.00	7.65e-07	2.00	5.149
(1024,1024)	1.58e-05	2.00	1.91e-07	2.00	37.128

and

$$g = (e^t + 2 + 2\pi^2(t-T)^2)\sin(\pi x_1)\sin(\pi x_2),$$

such that the exact solution is

$$y(x,t) = e^t \sin(\pi x_1)\sin(\pi x_2) \quad \text{and} \quad p(x,t) = (t-T)^2 \sin(\pi x_1)\sin(\pi x_2).$$

Table 5.6. Results for Ex. 9 with $\gamma = 10^{-2}$ (Implicit scheme with preconditioned GMRES).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(8,8,8)	2.81e-01	–	3.21e-02	–	6	0.099
(16,16,16)	7.68e-02	1.87	8.10e-03	1.99	6	0.099
(32,32,32)	1.98e-02	1.95	2.02e-03	2.00	7	0.355
(64,64,64)	4.99e-03	1.99	5.04e-04	2.00	7	1.969
(128,128,128)	1.25e-03	2.00	1.26e-04	2.00	7	14.539
(256,256,256)	3.13e-04	2.00	3.15e-05	2.00	7	124.004

Table 5.7. Results for Ex. 9 with $\gamma = 10^{-4}$ (Implicit scheme with preconditioned GMRES).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	Iter	CPU
(8,8,8)	6.03e-01	–	1.01e-03	–	9	0.122
(16,16,16)	1.53e-01	1.98	2.67e-04	1.92	15	0.201
(32,32,32)	3.87e-02	1.98	7.47e-05	1.84	16	0.775
(64,64,64)	9.69e-03	2.00	1.94e-05	1.94	16	5.222
(128,128,128)	2.42e-03	2.00	4.71e-06	2.04	16	35.181
(256,256,256)	6.05e-04	2.00	1.11e-06	2.09	17	374.137

Table 5.8. Results for Ex. 9 with $\gamma = 10^{-2}$ (Explicit scheme with sparse direct solver).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	CPU
(8,8,8)	1.05e-01	–	1.86e-02	–	0.003
(16,16,16)	2.63e-02	2.00	4.90e-03	1.92	0.104
(32,32,32)	6.62e-03	1.99	1.24e-03	1.99	9.221

Table 5.9. Results for Ex. 9 with $\gamma = 10^{-4}$ (Explicit scheme with sparse direct solver).

(M_1, M_2, N)	e_y^h	Order	e_p^h	Order	CPU
(8,8,8)	3.60e-01	–	1.14e-03	–	0.014
(16,16,16)	9.03e-02	1.99	7.54e-04	0.60	0.225
(32,32,32)	2.30e-02	1.97	2.49e-04	1.60	10.146

The numerical results are reported in Table 5.6, 5.7, 5.8, and 5.9 for our implicit scheme and the explicit scheme with different regularization parameter, respectively. Similar conclusions can be drawn as in the previous example. With a laptop PC, the sparse direct solver can not

handle a $64 \times 64 \times 64$ mesh (about 262,144 unknowns) easily due to too high memory costs, while our preconditioned GMRES method can solve a $256 \times 256 \times 256$ mesh (about 16,777,216 unknowns) in about two minutes. The key difference is that our preconditioned GMRES method has a linearithmic time complexity while the sparse direct solver usually does not. This shows a marvelous advantage of iterative methods over (sparse) direct solvers in treating large-scale problems like we are confronting here.

5.6 CONCLUSIONS

In this chaper, we have shown the second-order accuracy of a new implicit scheme in time for a system of forward-and-backward coupled wave equations arising from wave optimal control problems. The proposed scheme is unconditionally stable and the favorable discretized structure allows us to build a fast solver using preconditioned iterative methods with a very effective preconditioner. Numerical tests were conducted to validate our theoretical analysis. Based on our approach, the proposed implicit scheme in time as well as the obtained preconditioner are also expected to work seamlessly with finite element discretizations in space. Our next step is to incorporate the control constraints, which can be effectively treated by utilizing semismooth Newton methods (or equivalently primary-dual active set strategies) as the outer iteration. Some other future work includes constructing higher order finite difference schemes in time, which 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.

CHAPTER 6

SUMMARY AND FUTURE RESEARCH

6.1 SUMMARY

Nowadays, the methodology of mathematical modeling with computer simulations has been applied to nearly all fields of science. To better analyze and optimize these sophisticated models, users often have to utilize effective numerical methods for their approximated solutions whenever the analytic approaches are impossible or impractical. Hence, efficient and reliable computational methods have become more and more irreplaceable, especially for those industries involving large-scale, nonlinear, time-dependent interdisciplinary models. In particular, optimization or optimal control of nonlinear time-dependent PDE systems represents one of the most challenging problems in scientific computing. Serving as two major goals of the current thesis, higher accuracy of approximations and better computational efficiency are the two most fundamental pursuits in theoretic and algorithmic developments of scientific computing.

In chapters 2, 3, and 4, we have successfully applied the multigrid method with the semismooth Newton (SSN) method to optimal control problems governed by semilinear elliptic PDE and semilinear parabolic PDE, respectively. The efficiency of our proposed SSN multigrid method is significantly better than the available full-approximation-storage (FAS) multigrid in current literature. For time discretizations of parabolic PDEs, different from the commonly used Crank-Nicolson and BDF2 scheme, we developed a new provable second-order stable and convergent leapfrog scheme. Its simpler structure allows us to further establish a fast multigrid linear solver. Our new leapfrog multigrid solver is at least two orders of magnitude faster than the Crank-Nicolson scheme that based on MATLAB's sparse direct solver.

In chapter 5, we have also established a second-order accurate implicit scheme in time for wave optimal control problems. Differing from the elliptic and parabolic cases, we suggested to solve the discretized linear system using a preconditioned Krylov subspace method with an efficient preconditioner. Again, our proposed fast implicit solver is significantly faster than the

standard explicit scheme based on MATLAB's sparse direct solver.

From the viewpoint of numerical development, it is important for us to discretize the PDE 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 decent converging property. In short, we have presented a new integrated perspective of designing **fast-solver-oriented** discretizations in the context of PDE-constrained optimization. We expect to apply such an integrated perspective in more applications involving discretizations of differential or integral operators as well as their corresponding large-scale system solving. This assimilated thinking is anticipated to become more and more important as we gradually shift to the parallel computing era with more affordable and powerful computers built on many-core CPU/GPU architectures. Most current sequential discretizations may not be well suitable for implementing parallel algorithms.

6.2 FUTURE RESEARCH

Based on our past experience in numerical methods for PDE-constrained optimization, we would like to highlight a few possible interesting extensions of our current work.

- Generalization of the proposed methods to deal with more complicated boundary conditions as well as other evolution PDE control problems associated with state and/or control and/or gradient constraints, including the far more challenging non-stationary Stokes and Navier-Stokes flow control problems [Kunisch et al., 2009].

- Another practical improvement of our leapfrog scheme is to construct some higher order compact finite difference schemes [Spotz, 1995, Spotz and Carey, 2001, Lin et al., 2009, Lee et al., 2014] in time, which 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 a much coarser mesh size. Such a high-order scheme would be very attractive to those problems with sufficiently smooth solutions, because high-order accuracy usually requires higher regularity of the solutions.

- To reduce the high computational costs for 3D problems, we certainly want to look into some parallel algorithms based on domain decomposition techniques [Smith et al., 1996, Toselli and Widlund, 2005, Mathew, 2008, Heinkenschloss, 2005] for solving the forward-and-backward time-dependent PDE system. Parallel-in-time methods [Nievergelt, 1964] have been investigated for evolutionary PDE problems over the last four decades. Its basic idea is to distribute a tremendous computational task into many small connected parts, which can be executed by many different processors simultaneously. This could become very necessary when the size of the system is getting too large to fit into a single computer's memory, or it takes too much time for a well optimized method to converge.

- Many fast iterative methods [Pang and Sun, 2012, Lei and Sun, 2013, Pan et al., 2014] have been proposed in the last decade for solving fractional PDEs [Podlubny, 1999]. However, there are considerably less contributions devoted to optimal control of fractional PDEs. The most challenging issue brought by the non-local fractional derivatives is the highly computational and memory costs from fully dense systems upon any standard discretizations. One crucial principle is to keep certain computationally favorable structures when discretizing the fractional operators so that one can employ those state of the art fast iterative solvers, such as Toeplitiz solvers [Ng, 2004, Chan and Jin, 2007, Jin, 2010].

REFERENCES

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

[Ali et al., 2015] Ali, A. A., Deckelnick, K., and Hinze, M. (2015). Global minima for semilinear optimal control problems. *arXiv:1503.07086*.

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

[Arada et al., 2002] Arada, N., Casas, E., and Tröltzsch, F. (2002). Error estimates for the numerical approximation of a semilinear elliptic control problem. *Comput. Optim. Appl.*, 23(2):201–229.

[Aubert and Kornprobst, 2006] Aubert, G. and Kornprobst, P. (2006). *Mathematical problems in image processing*. Springer, New York.

[Axelsson and Verwer, 1985] Axelsson, A. O. H. and Verwer, J. G. (1985). Boundary value techniques for initial value problems in ordinary differential equations. *Math. Comp.*, 45(171):153–171.

[Barrett et al., 1994] Barrett, R., Berry, M., Chan, T. F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and der Vorst, H. V. (1994). *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*. SIAM, Philadelphia, PA.

[Benzi et al., 2005] Benzi, M., Golub, G. H., and Liesen, J. (2005). Numerical solution of saddle point problems. *Acta Numer.*, 14:1–137.

[Bergounioux et al., 2014] Bergounioux, M., Bonnefond, X., Haberkorn, T., and Privat, Y. (2014). An optimal control problem in photoacoustic tomography. *Math. Models Methods Appl. Sci.*, 24(12):2525–2548.

[Bergounioux et al., 1999] Bergounioux, M., Ito, K., and Kunisch, K. (1999). Primal-dual strategy for constrained optimal control problems. *SIAM J. Control Optim.*, 37(4):1176–1194.

[Biegler et al., 2007] Biegler, L. T., Ghattas, O., Heinkenschloss, M., Keyes, D., and van Bloemen Waanders, B., editors (2007). *Real-time PDE-constrained optimization*. SIAM, Philadelphia, PA.

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

[Borzì, 2007a] Borzì, A. (2007a). High-order discretization and multigrid solution of elliptic nonlinear constrained optimal control problems. *J. Comput. Appl. Math.*, 200(1):67–85.

[Borzì, 2007b] Borzì, A. (2007b). Space-time multigrid methods for solving unsteady optimal control problems. In *Real-time PDE-constrained optimization*, volume 3 of *Comput. Sci. Eng.*, pages 97–113. SIAM, Philadelphia, PA.

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

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

[Borzì and González Andrade, 2014] Borzì, A. and González Andrade, S. (2014). Second-order approximation and fast multigrid solution of parabolic bilinear optimization problems. *Adv. Comput. Math.* To appear.

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

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

[Borzí and Kunisch, 2005] Borzí, A. and Kunisch, K. (2005). A multigrid scheme for elliptic constrained optimal control problems. *Comput. Optim. Appl.*, 31:309–333.

[Borzì and Kunisch, 2006] Borzì, A. and Kunisch, K. (2006). A globalization strategy for the multigrid solution of elliptic optimal control problems. *Optim. Methods Softw.*, 21(3):445–459.

[Borzì and Schulz, 2009] Borzì, A. and Schulz, V. (2009). Multigrid methods for PDE optimization. *SIAM Rev.*, 51(2):361–395.

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

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

[Brabazon et al., 2014] Brabazon, K., Hubbard, M., and Jimack, P. (to appear, 2014). Nonlinear multigrid methods for second order differential operators with nonlinear diffusion coefficient. *Computers & Mathematics with Applications.*

[Brandt and Livne, 2011] Brandt, A. and Livne, O. (2011). *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics, Revised Edition.* Classics in Applied Mathematics. SIAM.

[Bredies et al., 2013] Bredies, K., Clason, C., Kunisch, K., and von Winckel, G., editors (2013). *Control and Optimization with PDE Constraints.* Birkhäuser Verlag, Basel.

[Brenner and Scott, 2008] Brenner, S. C. and Scott, L. R. (2008). *The mathematical theory of finite element methods.* Springer, New York, third edition.

[Briggs et al., 2000] Briggs, W. L., Henson, V. E., and McCormick, S. F. (2000). *A multigrid tutorial.* SIAM, Philadelphia, PA.

[Brown et al., 2003] Brown, P. N., Vassilevski, P. S., and Woodward, C. S. (2003). On mesh-independent convergence of an inexact Newton-multigrid algorithm. *SIAM J. Sci. Comput.*, 25(2):570–590.

[Brugnano and Trigiante, 1998] Brugnano, L. and Trigiante, D. (1998). *Solving differential problems by multistep initial and boundary value methods.* Gordon and Breach Science Publishers, Amsterdam.

[Bucci, 1992] Bucci, F. (1992). A Dirichlet boundary control problem for the strongly damped wave equation. *SIAM J. Control Optim.*, 30(5):1092–1100.

[Casas, 2007] Casas, E. (2007). Using piecewise linear functions in the numerical approximation

of semilinear elliptic control problems. *Adv. Comput. Math.*, 26(1-3):137–153.

[Casas and Tröltzsch, 2012] Casas, E. and Tröltzsch, F. (2012). Second order analysis for optimal control problems: improving results expected from abstract theory. *SIAM J. Optim.*, 22(1):261–279.

[Casas and Tröltzsch, 2015] Casas, E. and Tröltzsch, F. (2015). Second order optimality conditions and their role in PDE control. *Jahresber. Dtsch. Math.-Ver.*, 117(1):3–44.

[Chan et al., 1998] Chan, R. H., Chang, Q.-S., and Sun, H.-W. (1998). Multigrid method for ill-conditioned symmetric Toeplitz systems. *SIAM J. Sci. Comput.*, 19(2):516–529.

[Chan and Jin, 2007] Chan, R. H.-F. and Jin, X.-Q. (2007). *An introduction to iterative Toeplitz solvers*. SIAM, Philadelphia, PA.

[Chen et al., 2000] Chen, X., Nashed, Z., and Qi, L. (2000). Smoothing methods and semismooth methods for nondifferentiable operator equations. *SIAM J. Numer. Anal.*, 38(4):1200–1216.

[Chertock et al., 2014] Chertock, A., Herty, M., and Kurganov, A. (2014). An eulerian-lagrangian method for optimization problems governed by multidimensional nonlinear hyperbolic pdes. *Comput. Optim. Appl., to appear*.

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

[Davis, 2006] Davis, T. A. (2006). *Direct methods for sparse linear systems*. SIAM, Philadelphia, PA.

[De los Reyes, 2015] De los Reyes, J. C. (2015). *Numerical PDE-Constrained Optimization*. Springer International Publishing.

[Debnath, 2012] Debnath, L. (2012). *Nonlinear partial differential equations for scientists and engineers*. Birkhäuser/Springer, New York.

[Dembo et al., 1982] Dembo, R., Eisenstat, S., and Steihaug, T. (1982). Inexact newton methods. *SIAM J. Numer. Anal.*, 19(2):400–408.

[Engel and Griebel, 2011] Engel, M. and Griebel, M. (2011). A multigrid method for constrained

optimal control problems. *J. Comput. Appl. Math.*, 235(15):4368–4388.

[Evans, 2010] Evans, L. C. (2010). *Partial differential equations*. AMS, Providence, RI, second edition.

[Gerdts et al., 2008] Gerdts, M., Greif, G., and Pesch, H. J. (2008). Numerical optimal control of the wave equation: optimal boundary control of a string to rest in finite time. *Math. Comput. Simulation*, 79(4):1020–1032.

[Golub and Van Loan, 2013] Golub, G. H. and Van Loan, C. F. (2013). *Matrix computations*. Johns Hopkins University Press, Baltimore, MD, fourth edition.

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

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

[Gugat and Grimm, 2011] Gugat, M. and Grimm, V. (2011). Optimal boundary control of the wave equation with pointwise control constraints. *Comput. Optim. Appl.*, 49(1):123–147.

[Gugat et al., 2009] Gugat, M., Keimer, A., and Leugering, G. (2009). Optimal distributed control of the wave equation subject to state constraints. *ZAMM Z. Angew. Math. Mech.*, 89(6):420–444.

[Gunzburger, 2003] Gunzburger, M. D. (2003). *Perspectives in flow control and optimization*. SIAM.

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

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

[Hackbusch, 1980] Hackbusch, W. (1980). Fast solution of elliptic control problems. *J. Optim. Theory Appl.*, 31(4):565–581.

[Hackbusch, 1981] Hackbusch, W. (1981). Numerical solution of linear and nonlinear parabolic

control problems. In *Optimization and optimal control (Proc. Conf., Math. Res. Inst., Oberwolfach, 1980)*, volume 30 of *Lecture Notes in Control and Information Sci.*, pages 179–185. Springer, Berlin-New York.

[Hackbusch, 2003] Hackbusch, W. (2003). *Elliptic Differential Equations: Theory and Numerical Treatment*. Springer.

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

[Herty et al., 2015] Herty, M., Kurganov, A., and Kurochkin, D. (2015). Numerical method for optimal control problems governed by nonlinear hyperbolic systems of PDEs. *Commun. Math. Sci.*, 13(1):15–48.

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

[Heywood and Rannacher, 1990] Heywood, J. G. and Rannacher, R. (1990). Finite-element approximation of the nonstationary Navier-Stokes problem. IV. Error analysis for second-order time discretization. *SIAM J. Numer. Anal.*, 27(2):353–384.

[Hintermüller et al., 2002] Hintermüller, M., Ito, K., and Kunisch, K. (2002). The primal-dual active set strategy as a semismooth newton method. *SIAM J. Optim.*, 13(3):865–888.

[Hintermüller et al., 2008] Hintermüller, M., Tröltzsch, F., and Yousept, I. (2008). Mesh-independence of semismooth Newton methods for Lavrentiev-regularized state constrained nonlinear optimal control problems. *Numer. Math.*, 108(4):571–603.

[Hintermüller and Ulbrich, 2004] Hintermüller, M. and Ulbrich, M. (2004). A mesh-independence result for semismooth Newton methods. *Math. Program.*, 101(1):151–184.

[Hinze et al., 2012] Hinze, M., Köster, M., and Turek, S. (2012). A space-time multigrid method for optimal flow control. In *Constrained optimization and optimal control for partial differential equations*, pages 147–170. Birkhäuser/Springer, Basel.

[Hinze et al., 2009] Hinze, M., Pinnau, R., Ulbrich, M., and Ulbrich, S. (2009). *Optimization with PDE constraints*. Springer, New York.

[Hinze and Vierling, 2012] Hinze, M. and Vierling, M. (2012). The semi-smooth Newton method for variationally discretized control constrained elliptic optimal control problems; implementation, convergence and globalization. *Optim. Methods Softw.*, 27(6):933–950.

[Horn and Johnson, 2013] Horn, R. A. and Johnson, C. R. (2013). *Matrix analysis*. Cambridge University Press, Cambridge.

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

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

[Jameson, 1988] Jameson, A. (1988). Aerodynamic design via control theory. *J. Sci. Comput.*, 3(3):233–260.

[Jin, 2010] Jin, X.-Q. (2010). *Preconditioning Techniques for Toeplitz Systems*. Higher Education Press, Beijing.

[Jovanović and Süli, 2014] Jovanović, B. S. and Süli, E. (2014). *Analysis of finite difference schemes*. Springer, London.

[Kelley, 2003] Kelley, C. T. (2003). *Solving nonlinear equations with Newton's method*. SIAM, Philadelphia, PA.

[Knabner and Angermann, 2003] Knabner, P. and Angermann, L. (2003). *Numerical methods for elliptic and parabolic partial differential equations*. Springer-Verlag, New York.

[Kröner, 2011a] Kröner, A. (2011a). Adaptive finite element methods for optimal control of second order hyperbolic equations. *Comput. Methods Appl. Math.*, 11(2):214–240.

[Kröner, 2011b] Kröner, A. (2011b). *Numerical Methods for Control of Second Order Hyperbolic Equations*. PhD thesis, Fakultät für Mathematik, Technische Universität München.

[Kröner, 2013] Kröner, A. (2013). Semi-smooth Newton methods for optimal control of the dynamical Lamé system with control constraints. *Numer. Funct. Anal. Optim.*, 34(7):741–769.

[Kröner and Kunisch, 2014] Kröner, A. and Kunisch, K. (2014). A minimum effort optimal control problem for the wave equation. *Comput. Optim. Appl.*, 57(1):241–270.

[Kröner et al., 2011] Kröner, A., Kunisch, K., and Vexler, B. (2011). Semismooth Newton methods for optimal control of the wave equation with control constraints. *SIAM J. Control Optim.*, 49(2):830–858.

[Kunisch et al., 2009] Kunisch, K., Leugering, G., Sprekels, J., and Tröltzsch, F., editors (2009). *Optimal control of coupled systems of partial differential equations*. Birkhäuser Verlag, Basel.

[Kunisch and Wachsmuth, 2013a] Kunisch, K. and Wachsmuth, D. (2013a). On time optimal control of the wave equation and its numerical realization as parametric optimization problem. *SIAM J. Control Optim.*, 51(2):1232–1262.

[Kunisch and Wachsmuth, 2013b] Kunisch, K. and Wachsmuth, D. (2013b). On time optimal control of the wave equation, its regularization and optimality system. *ESAIM Control Optim. Calc. Var.*, 19(2):317–336.

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

[Lee et al., 2014] Lee, S. T., Liu, J., and Sun, H.-W. (2014). Combined compact difference scheme for linear second-order partial differential equations with mixed derivative. *J. Comput. Appl. Math.*, 264:23–37.

[Lei and Sun, 2013] Lei, S.-L. and Sun, H.-W. (2013). A circulant preconditioner for fractional diffusion equations. *J. Comput. Phys.*, 242:715–725.

[Leugering et al., 2014] Leugering, G., Benner, P., Engell, S., Griewank, A., Harbrecht, H., Hinze, M., Rannacher, R., and Ulbrich, S., editors (2014). *Trends in PDE Constrained Optimization*. Springer International Publishing, Switzerland.

[Leugering et al., 2012] Leugering, G., Engell, S., Griewank, A., Hinze, M., Rannacher, R., Schulz, V., Ulbrich, M., and Ulbrich, S., editors (2012). *Constrained optimization and optimal control for partial differential equations*. Birkhäuser/Springer, Basel.

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

[Lin et al., 2009] Lin, Y., Gao, X., and Xiao, M. (2009). A high-order finite difference method for 1D nonhomogeneous heat equations. *Numer. Methods Partial Differential Equations*, 25(2):327–346.

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

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

[Liu and Xiao, 2014a] Liu, J. and Xiao, M. (to appear, 2014a). A new semi-smooth newton multigrid method for control-constrained semi-linear elliptic pde problems. *J. Global Optim.*

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

[Luo et al., 2013] Luo, X., Chen, Y., and Huang, Y. (2013). A priori error estimates of finite volume element method for hyperbolic optimal control problems. *Sci. China Math.*, 56(5):901–914.

[Mathew, 2008] Mathew, T. P. A. (2008). *Domain decomposition methods for the numerical solution of partial differential equations.* Springer-Verlag, Berlin.

[Meidner and Vexler, 2008a] Meidner, D. and Vexler, B. (2008a). 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):1150–1177.

[Meidner and Vexler, 2008b] Meidner, D. and Vexler, B. (2008b). 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):1301–1329.

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

[Morton and Mayers, 2005] Morton, K. W. and Mayers, D. F. (2005). *Numerical solution of partial differential equations*. Cambridge University Press, Cambridge, second edition.

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

[Neittaanmaki and Tiba, 1994] Neittaanmaki, P. and Tiba, D. (1994). *Optimal Control of Nonlinear Parabolic Systems: Theory, Algorithms, and Applications*. Taylor & Francis.

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

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

[Ng, 2004] Ng, M. K. (2004). *Iterative methods for Toeplitz systems*. Oxford University Press, New York.

[Nievergelt, 1964] Nievergelt, J. (1964). Parallel methods for integrating ordinary differential equations. *Comm. ACM*, 7:731–733.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. J. (2006). *Numerical optimization*. Springer, New York, second edition.

[Ortega and Rheinboldt, 2000] Ortega, J. M. and Rheinboldt, W. C. (2000). *Iterative solution of nonlinear equations in several variables*. SIAM, Philadelphia, PA.

[Pan et al., 2014] Pan, J., Ke, R., Ng, M. K., and Sun, H.-W. (2014). Preconditioning techniques for diagonal-times-Toeplitz matrices in fractional diffusion equations. *SIAM J. Sci. Comput.*, 36(6):A2698–A2719.

[Pang and Sun, 2012] Pang, H.-K. and Sun, H.-W. (2012). Multigrid method for fractional diffusion equations. *J. Comput. Phys.*, 231(2):693–703.

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

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

[Podlubny, 1999] Podlubny, I. (1999). *Fractional differential equations*. Academic Press, Inc., San Diego, CA.

[Porcelli et al., 2014] Porcelli, M., Simoncini, V., and Tani, M. (2014). Preconditioning of Active-Set Newton Methods for PDE-constrained Optimal Control Problems. *arXiv:1407.1144*.

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

[Rincon and Liu, 2003] Rincon, A. and Liu, I.-S. (2003). On numerical approximation of an optimal control problem in linear elasticity. *Divulg. Mat.*, 11(2):91–107.

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

[Saad and Schultz, 1986] Saad, Y. and Schultz, M. H. (1986). Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3).

[Schiela and Ulbrich, 2014] Schiela, A. and Ulbrich, S. (2014). Operator preconditioning for a class of inequality constrained optimal control problems. *SIAM J. Optim.*, 24(1):435–466.

[Schöberl et al., 2011] Schöberl, J., Simon, R., and Zulehner, W. (2011). A robust multigrid method for elliptic optimal control problems. *SIAM J. Numer. Anal.*, 49(4):1482–1503.

[Schöberl and Zulehner, 2007] Schöberl, J. and Zulehner, W. (2007). Symmetric indefinite preconditioners for saddle point problems with applications to PDE-constrained optimization problems. *SIAM J. Matrix Anal. Appl.*, 29(3):752–773.

[Silvester and Wathen, 1994] Silvester, D. and Wathen, A. (1994). Fast iterative solution of stabilised Stokes systems. II. Using general block preconditioners. *SIAM J. Numer. Anal.*, 31(5):1352–1367.

[Smith et al., 1996] Smith, B. F., Bjorstad, P. E., and Gropp, W. D. (1996). *Domain decomposition*. Cambridge University Press, Cambridge. Parallel multilevel methods for elliptic partial differential equations.

[Spotz, 1995] Spotz, W. F. (1995). *High-Order Compact Finite Difference Schemes for Computational Mechanics*. PhD thesis, The University of Texas at Austin.

[Spotz and Carey, 2001] Spotz, W. F. and Carey, G. F. (2001). Extension of high-order compact schemes to time-dependent problems. *Numer. Methods Partial Differential Equations*, 17(6):657–672.

[Stoll and Wathen, 2012] Stoll, M. and Wathen, A. (2012). Preconditioning for partial differential equation constrained optimization with control constraints. *Numer. Linear Algebra Appl.*, 19(1):53–71.

[Strikwerda, 2004] Strikwerda, J. C. (2004). *Finite difference schemes and partial differential equations*. SIAM, Philadelphia, PA, second edition.

[Sun and Liu, 2010] Sun, L.-Y. and Liu, J. (2010). Constraint preconditioning for nonsymmetric indefinite linear systems. *Numer. Linear Algebra Appl.*, 17(4):677–689.

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

[Takacs and Zulehner, 2013] Takacs, S. and Zulehner, W. (2013). Convergence analysis of all-at-once multigrid methods for elliptic control problems under partial elliptic regularity. *SIAM J. Numer. Anal.*, 51(3):1853–1874.

[Thomas, 1995] Thomas, J. W. (1995). *Numerical partial differential equations: finite difference methods*. Springer-Verlag, New York.

[Thomas, 1999] Thomas, J. W. (1999). *Numerical partial differential equations: conservation*

laws and elliptic equations. Springer-Verlag, New York.

[Toselli and Widlund, 2005] Toselli, A. and Widlund, O. (2005). *Domain decomposition methods—algorithms and theory*. Springer-Verlag, Berlin.

[Trefethen and Bau, 1997] Trefethen, L. N. and Bau, III, D. (1997). *Numerical linear algebra*. SIAM, Philadelphia, PA.

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

[Trottenberg et al., 2001] Trottenberg, U., Oosterlee, C. W., and Schüller, A. (2001). *Multigrid*. Academic Press Inc., San Diego, CA.

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

[Zuazua, 2005] Zuazua, E. (2005). Propagation, observation, and control of waves approximated by finite difference methods. *SIAM Rev.*, 47(2):197–243.

VITA

Graduate School
Southern Illinois University

Name: Jun Liu Email: gdctor@gmail.com, junliu2010@siu.edu

Education:
M.S., Computational Mathematics, South China Normal University, China, 2010.
B.S., Information and Computing Science, Guangdong University of Technology, China, 2004.

Employment:
Software Engineer, China National Software and Service Co., Ltd., China, 2004-2007.

Publications:

1. **Jun Liu**, and Mingqing Xiao, A leapfrog semi-smooth Newton-multigrid method for semi-linear parabolic optimal control problems, to appear in *Computational Optimization and Applications* (DOI:10.1007/s10589-015-9759-z), 2015.

2. **Jun Liu** and Mingqing Xiao, A new semi-smooth Newton multigrid method for control-constrained semi-linear elliptic PDE problems, to appear in *Journal of Global Optimization* (DOI:10.1007/s10898-014-0206-y), 2014.

3. **Jun Liu** and Haiwei Sun, A fast high-order sinc-based algorithm for pricing options under jump-diffusion processes, *International Journal of Computer Mathematics*, 91(10), pp. 2163–2184, 2014.

4. Xuejun Gao, Tingwen Huang, Yu Huang, **Jun Liu**, and Mingqing Xiao, Observer design for axial flow compressor, *ASME Journal of Dynamic Systems, Measurement, and Control*, 136, 051017:1–12, 2014.

5. Spike T. Lee, **Jun Liu**, and Haiwei Sun, Combined compact difference scheme for linear second-order partial differential equations with mixed derivative, *Journal of Computational and Applied Mathematics*, 264, pp. 23–37, 2014.

6. **Jun Liu**, and Mingqing Xiao, Rank-one characterization of joint spectral radius of finite matrix family, *Linear Algebra and its Applications*, 438(8), pp. 3258–3277, 2013.

7. Xiongping Dai, Yu Huang, **Jun Liu**, Mingqing Xiao, The finite-step realizability of the joint spectral radius of a pair of d-by-d matrices one of which being rank-one, *Linear Algebra and its Applications*, 437(7), pp. 1548–1561, 2012.

8. Xiaoshan Chen, Wen Li, Xiaojun Chen, and **Jun Liu**, Structured backward errors for generalized saddle point systems, *Linear Algebra and its Applications*, 436(9), pp. 3109–3119, 2012.

9. **Jun Liu** and Haiwei Sun, Sinc-Galerkin method for the option pricing under jump-diffusion model, *East-West Journal of Mathematics*, pp. 317–327, 2009.

10. Liying Sun and **Jun Liu**, Constraint preconditioning for nonsymmetric indefinite linear systems, *Numerical Linear Algebra with Applications*, 17(4), pp. 677–689, 2009.

11. Buyang Li, **Jun Liu**, and Mingqing Xiao, Leapfrog multigrid methods for parabolic optimal control problems, to appear in *Proc. of the 27th Chinese Control and Decision Conference*, 2015. (**In the Finalists for Zhang Si-Ying Outstanding Youth Paper Award**).

12. **Jun Liu**, Tingwen Huang, and Mingqing Xiao, A semismooth Newton multigrid method for constrained elliptic optimal control problems, *Advances in Global Optimization, Springer Proceedings in Mathematics & Statistics Vol. 95*, pp. 397–405, 2015.

13. **Jun Liu** and Mingqing Xiao, A new semi-smooth Newton multigrid method for parabolic PDE optimal control problems, *Proc. of the 53rd IEEE Conference on Decision and Control*, pp. 5568–5573, 2014. (**Received the IEEE Student Travel Award**).

14. **Jun Liu**, Yu Huang, Haiwei Sun, and Mingqing Xiao, High-order numerical methods for wave equations with van der Pol type boundary conditions, *Proc. of the SIAM Conference on Control and Its Applications (CT13)*, pp. 144–151, 2013.

15. **Jun Liu** and Mingqing Xiao, Computation of joint spectral radius for network model associated with rank-one matrix set, *Proc. of the 19th International Conference on Neural Information Processing, Lecture Notes in Computer Science, Springer, Vol. 7665*, pp. 356–363, 2012.

16. Xuejun Gao, Tingwen Huang, **Jun Liu**, and Mingqing Xiao, Local observer for axial flow aeroengine compressors, *Proc. of the 10th World Congress on Intelligent Control and Automation*, pp. 2233–2238, 2012.

Preprints:

1. Buyang Li, **Jun Liu**, and Mingqing Xiao, A fast and stable preconditioned iterative method for optimal control of wave equations, under review, 2015.

2. **Jun Liu**, Brittany D. Froese, Adam M. Oberman, and Mingqing Xiao, A multigrid solver for the three dimensional Monge-Ampere equations, under review, 2014.

3. Buyang Li, **Jun Liu**, and Mingqing Xiao, A second-order leapfrog multigrid method for the numerical solution of parabolic optimal control problems, under review, 2014.

4. **Jun Liu**, Yu Huang, Haiwei Sun, and Mingqing Xiao, Numerical methods for weak solution of wave equation with van der Pol type nonlinear boundary conditions, under review, 2014.