ABSTRACT

THOMPSON, KARMETHIA CHANTAL. Solving Nonlinear Constrained Optimization Time Delay Systems with a Direct Transcription Approach. (Under the direction of Dr. Stephen L. Campbell.)

In this thesis we explore the application of $\mathbb{SOCX}$, a direct transcription (DT) optimization software tool that employs Runge-Kutta methods and interpolating polynomials for the solutions to nonlinear constrained optimal control systems containing both state delays and control delays. Direct transcription methods are popular direct methods that in some form transcribe the entire optimal control system into a large sparse nonlinear programming problem (NLP) by some discretization scheme. Optimal control delay solvers are important because they help to provide numerical solutions to large time delay ordinary differential equations (ODEs) or differential algebraic equations (DAEs), optimal control problems (OCPs), and other optimization systems that describe modeled processes that commonly arise in research and industry.

This thesis is partitioned into three main parts. The first portion of this thesis provides an overview of direct transcription and the direct transcription approach implemented in the software. Here we provide the results for typical optimal control delay problems (OCDPs) with both state and control delays. The second portion of this thesis concerns itself with the results generated from control delay systems. Control delays are implemented as algebraic variables and may become free to the optimizer when the mesh is nonuniform. An exogenous input control (EIC) method that uses a weighted cost function, additional state delays, and control variables is presented as an alternative formulation for solving optimization systems with control delays.

The third portion of this thesis seeks to validate implementation of the EIC method by way of studying convergence through analytic techniques. Though numerically robust, the EIC formulation resembles the dynamics of a singular optimal control problem which is well documented as a very hard problem to solve theoretically [4], [34], and [42]. Using a method of steps (MOS) approach and an ε-asymptotic approximation approach we show that the solution of the reformulated control delay problem converges to the original control delay problem as the

added weighting factor tends to zero in the cost function.

$\mathbb{SOCX}$ is a general purpose tool for the modeling and simulation of a large variety of differential algebraic equation systems. DAEs are used to describe an array of industrial and technical processes, and can be used to formulate other time delay differential equation and optimization systems. The remaining part of this thesis explores the software's ability to solve advanced time, mixed-delay, neutral delay, and time-varying systems. An overview of each type of system is given and an example is presented and solved. Furthermore, the results and analysis presented in this thesis show that Runge-Kutta direct transcription methods are appropriate and effective for solving complicated time delay optimization systems as well as unconventional optimal control problems.

Solving Nonlinear Constrained Optimization Time Delay Systems with a Direct Transcription Approach

by
Karmethia Chantal Thompson

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Applied Mathematics

Raleigh, North Carolina

2014

APPROVED BY:

Dr. Hien Tran	Dr. Negash Medhen
Dr. John T. Betts	Dr. Stephen L. Campbell Chair of Advisory Committee

DEDICATION

This thesis is dedicated to the living memories of my aunt, Earlene Lambry. I truly wish that you could have been here to experience this moment with me. I hold my head high knowing that I made you proud.

BIOGRAPHY

Karmethia Thompson was born in Atlanta, GA on April 5, 1986 to parents Catherine Hall and Leon Thompson. She is the middle child with two sisters, Sophia Hall and Kanisha Thompson. She grew up in the East Atlanta community and graduated in 2004 from Southside Comprehensive High School. Throughout her high school years she participated in a variety of activities: basketball, softball, soccer, and band. However, it was her love for math that opened doors to college opportunities. She graduated from the "*GREAT*" Bethune-Cookman University in 2008 with a B.S. in mathematics with a concentration in computer science.

While in college she was a member of the Florida-Georgia Louis Stokes for Minority Participation program (FGLSAMP), an academic support program that provides counseling, mentoring, tutoring and shadowing services to minorities in an effort to encourage them to pursue careers in Science, Technology, Engineering, and Mathematics (STEM). This program gave her exposure to opportunities beyond college. After graduating college Karmethia moved to Raleigh, NC where she attended graduate school at North Carolina State University.

After graduation Karmethia will join the governmental workforce with the Department of Defense.

ACKNOWLEDGEMENTS

This research was in part funded by the National Science Foundation under NSF Grant DMS-1209251.

I am especially grateful for having Dr. Stephen Campbell, committee chairman as my advisor. His knowledge, expertise, passion, and guidance help to facilitate this entire process. His support and encouragement pushed me to work harder in those trying times. Under Dr. Campbell's direction I have learned many things such as optimal control, professorship, writing skills, and even a few stories about the lessons of life. For these skills acquired I am forever thankful.

Additionally, I wish to thank a few of the faculty and staff at North Carolina State University. Special thanks to my committee members Dr. Medhin, Dr. Tran, and Dr. Betts who were more than generous with their expertise, feedback, and precious time. I would like to emphasize that Dr. Betts is a distant committee member and made several trips across the country to support my degree process. I thank him for his commitment and support. To Ms. Denise Seabrooks, Mathematics Department Graduate Service Coordinator, I thank you for the outstanding service that you have given the graduate math department. Your efforts are always filled with love and sincerity. Without your hard work I would have forfeited my preliminary exam. This is just one of the many occasions that you saved me. Thank you so much! NC State you prepared a big stage for me to perform on. Thanks for giving me the support needed to rise to the occasion.

It would not have been possible to write this doctoral thesis without the help and support of my fellow colleagues. I wholeheartedly thank Ms. Nakeya Williams and Mr. Terrance Pendleton for their friendship and support. I could always rely on you anytime of the day and any hour of the night. I am sure that we will remain friends throughout our careers. I wish each of you the best. Special thanks to LaKendra Blash for helping me prepare for my seminar and conference presentations. Your advice and criticism was greatly appreciated.

I give immense thanks to my family and friends back in Atlanta, GA. To my academic mentors Mr. Demarco Mitchell and Mrs. Annette Alexander, I thank you for challenging me to be the best student I could be. It was your faith and encouragement that inspired me to pursue a higher education degree. To my grandmother, Idell Lambry, you are my rock. Thank you for your strength. To my mother and father, Catherine Hall and Leon Thompson, you are the roots that keep me focused and grounded. All that I have achieved and all that I will accomplish is for you.

Last but not least, I would like to thank my Lord and Savior Jesus Christ. It is because of you that I have been blessed with this knowledge and these talents that will continue to service the world throughout the duration of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Background: The Optimal Control Problem

Optimal control theory is a mathematical discipline that focuses on finding optimal ways to control a dynamic system. It is an extension of the calculus of variations, a branch of mathematics concerning problems that seek to find the path, curve, or surface for which a given function has a minimum or maximum. Let $[t_0, t_1]$ be a fixed time interval, then a simple calculations of variations problem minimizes the integral functional

$$J = \int_{t_0}^{t_1} F[x(t), \dot{x}(t), t]\, dt, \tag{1.1a}$$

over the curves $x(t) : [t_0, t_1] \to \mathbb{R}^m$ satisfying the conditions

$$x(t_0) = \hat{x}(t_0), \quad x(t_1) = \hat{x}(t_1), \quad [x(t)^T, \dot{x}(t)^T, t] \in \mathcal{Q} \;\text{ a.e. on } [t_0, t_f], \tag{1.1b}$$

where $\hat{x}_0$, $\hat{x}_1 \in \mathbb{R}^m$ are certain fixed points, $\mathcal{Q} \subset \mathbb{R}^{2m+1}$ is an open set, and $F : \mathcal{Q} \to \mathbb{R}^1$ is a smooth function of its arguments [100].

1.0.1 Problem Description

Optimal control theory studies similar problems but from a more dynamic perspective. A typical problem in optimal control theory involves a system that is composed of a set of differential equations that represent the propagation of the differential or **state** variables as a function of the independent variable, say, time [131]. The components of the state vector may reflect position, velocity, acceleration, or other physical properties and are impacted by the presence of an input variable termed the **control**. In practice, many optimal control problems impose restrictive conditions or **constraints** on both the state and the control. Constraints are broken up into two main categories, **path constraints** or **point constraints**. Path constraints restrict the range of values taken by either the state, control, or both variables over the entire time interval, $[t_0,\ t_1]$, or any time subinterval. Point constraints are usually imposed as **boundary constraints** which require the system's response to achieve a given target at a specific terminal time, t_1 or initial time t_0. Constraints can be better explained through the following example. Consider a dynamical system that wishes to model the flight path of an aircraft from take-off to landing. Requiring the thrust or control to assume a constant value once the aircraft reaches a certain altitude or state is considered a path constraint. Requiring the aircraft to land in a specific place at a specific time is considered a point constraint. Other limitations on the optimal control system can be enforced by its **performance index** (also termed **cost functional**) which is a function that is optimized to meet the desired performance of the system. The performance index is chosen so that emphasis is given to the important system parameters, and is usually minimized over a class of controls.

The general formulation of an optimal control problem for a continuous time-varying system involves a nonlinear dynamical equation

$$\dot{x}(t) = f[x(t), u(t), t], \quad x(t_0) = x_0, \tag{1.2a}$$

with state, $x(t) \in \mathbb{R}^n$ and control $u(t)$ which is constrained to lie in $\mathcal{U} \subseteq \mathbb{R}^m$. The final position, $x(t_1) = x_1$ may be fixed or free. The performance of the dynamical system is to be regulated to minimize the performance index

$$J(u) = \phi[x_1, t_1] + \int_{t_0}^{t_1} \mathcal{L}[x(t), u(t), t]\, dt, \tag{1.2b}$$

where $[t_0, t_1]$ is the time interval of interest, $\phi : \mathbb{R}^n \to \mathbb{R}$ is the **terminal cost**, and the Lagrangian functional $\mathcal{L} : \mathbb{R}^{m+n+1} \to \mathbb{R}$ is the **running cost**. The terminology for function ϕ springs from the fact that it depends on the final state and final time. Similarly, $\mathcal{L}$ depends on the state and control at intermediate times. The structure of the performance index is used to classify the optimal control problem. The optimal control problem is referred to as a *Mayer* problem when the performance index is expressed as the terminal cost and as a *Lagrange* problem when it is expressed as the running cost. More common, the optimal control problem is of the *Bolza* type when the performance index is composed of both integral and final state components as in Eq. (1.2b).

The concept of control plays an important role in determining the nature and mathematical formulation of the optimal control problem. If the control is of the **open-loop** type the goal of the optimal control problem is to find an admissible control $u^*(t,\ x(t_0))$ on the interval $[t_0,\ t_1]$ that drives the system (1.2a) along a trajectory $x^*(t)$ such that (1.2b) is minimized, and such that

$$\psi[x_1, t_1] = 0, \tag{1.2c}$$

for a given function $\psi \in \mathbb{R}^p$. In this case the control law is determined as a function of time, and is only optimal for a particular initial state value. Open loop controllers have a simple implementation, are easier to construct, and have a cheap cost. However, because they are only impacted by the initial value external disturbances cannot be accounted for which may cause output results to be inaccurate and unreliable. For open loop systems changes in output can be corrected by manually changing the input.

Closed-loop or *automatic* controllers feed output back into the system to adjust the input signal in such a manner as to maintain the desired output value. Mathematically, the closed loop control is defined $u^*(t,\, x(t))$. Different from open loop control note that closed-loop controllers are dependent on current values of the state. The feedback properties of closed loop systems automatically account for external disturbances which make them more ideal, accurate, and reliable than open-loop systems. However, their complex construction causes them to be more difficult to design and costly. Detail formulations for both open and closed loop continuous-time varying and discrete-time systems can be found in [60, 92].

1.0.2 Pontryagin's Minimum Principle

Pontryagin's minimum principle (PMP) is a method that uses variational principles to derive necessary conditions that an optimal trajectory must satisfy. Consider an optimal control problem (1.2) where a control, $u(t) \in \mathcal{U}$ is to be chosen to minimize (1.2b) subject to (1.2a) where ϕ and $\mathcal{L}$ are continuously differentiable. Note that the dynamic constraint can be adjoined to the cost by introducing the Lagrange multiplier, $\lambda(t) \in \mathbb{R}^n$ which gives

$$\hat{J}(u) = \phi[x_1, t_1] + \int_{t_0}^{t_1} \mathcal{L}[x(t), u(t), t] + \lambda^T(t)\big(f[x(t), u(t), t] - \dot{x}(t)\big)\, dt. \tag{1.2d}$$

The following *Hamiltonian* function is derived from (1.2d) and can be used to rewrite the necessary conditions,

$$\mathcal{H}[x(t), u(t), \lambda(t), t] = \mathcal{L}[x(t), u(t), t] + \lambda^T f[x(t), u(t), t]. \tag{1.3a}$$

The Hamiltonian is composed of the Lagrangian functional adjoined by the dynamic equations. For this $\lambda(t)$ is alternatively referred to as the **adjoint** variable. The necessary conditions are constructed using the partial derivatives of $\mathcal{H}$, and are defined

State equation:

$$\dot{x}^*(t) = \mathcal{H}_\lambda[x^*(t), u^*(t), \lambda^*(t), t] = f[x^*(t), u^*(t), t], \tag{1.3b}$$

Costate equation:

$$-\dot{\lambda}^*(t) = \mathcal{H}_x[x^*(t), u^*(t), \lambda^*(t), t] = \mathcal{L}_x[x^*(t), u^*(t), t] + f_x^T[x^*(t), u^*(t), t]\,\lambda(t), \tag{1.3c}$$

Stationarity conditon:

$$0 = \mathcal{H}_u[x^*(t), u^*(t), \lambda^*(t), t] = \mathcal{L}_u[x^*(t), u^*(t), t] + f_u^T[x^*(t), u^*(t), t]\,\lambda(t). \tag{1.3d}$$

The stationarity condition in Eq. (1.3d) is for the case where $u(t)$ is unconstrained. If the control is constrained the stationarity condition is such that $u(t)$ minimizes the Hamiltonian. When a Lagrange multiplier, $\lambda(t)$ is associated with a state equation it is termed a **costate** and (1.3c) is considered the costate equation. Boundary conditions vary and depend on the terminal state variable. If the final time t_1 is fixed and the terminal state is free, then

$$\begin{cases} \lambda^*(t_1) = \phi_x[t_1,\ x^*(t_1)] \\ x^*(t_0) = x_0 \end{cases}, \tag{1.3e}$$

are the associated boundary conditions. However, if the terminal state is fixed we have the following boundary conditions

$$\begin{cases} x^*(t_0) = x_0 \\ (\phi_x + \psi_x^{\mathrm{T}}\nu - \lambda)|_{t_1}\ dx(t_1) + (\phi_t + \psi_t^{\mathrm{T}}\nu + \mathcal{H})|_{t_1}\ dt_1 = 0 \end{cases}, \tag{1.3f}$$

The Hamiltonian must be minimized over all admissible $u(t)$ for optimal values of the state and costate. Mathematically, if $u(t)^* \in \mathcal{U}$ is the optimal control that minimizes $\mathcal{H}$, then (1.3d) is replaced by a $u(t)^*$ that satisfies

$$\mathcal{H}[x^*, u^*, \lambda^*, t] \leq \mathcal{H}[x^*, u, \lambda^*, t], \quad \text{for all admissible } u \in \mathcal{U}. \tag{1.4}$$

For an extensive discussion of Pontryagin's minimum principle refer to [44, 114, 115, 131].

Chapter 2

Introduction

2.1 Motivation

In this thesis we investigate the numerical treatment of nonlinear constrained optimal control systems with state delays and control delays by way of a direct transcription optimization software tool. Optimal control delay systems are widely utilized to simulate the mechanics of biological, chemical, industrial, and ecological processes. Time delays are primarily inherited from the physical characteristics and interaction of the components of the modeled system. In biological applications time delays represent gestation times, incubation periods, transport delays, or can simply lump complicated biological processes together, accounting only for the time required for these processes to occur [55]. In engineering systems time delays are often present due to measurement, transmission and transport lags, computational delays, or unmodeled inertias of system components [152]. Time delays can also occur in control systems from either the control operator or the necessary time it takes the controller to sense a certain behavior or send an appropriate signal to parts of the system.

As time delays naturally result from physical processes, they are often artificially introduced to ignite an appropriate or desired system response. Judicious introduction of a delay may stabilize an otherwise unstable system, using, for example, a wait-and-act control strategy, or may improve steady-state tracking error [45]. Whether inherited or introduced, time delays enhance the complexity of the system dynamics further increasing difficulty in providing an optimal solution. Mathematically, a system time delay relates to a state or control variable defined at an earlier time value, and is commonly represented by a function

$$\omega(t) = t - \tau, \qquad \tau > 0. \tag{2.1}$$

where τ is constant. However, τ can take on many other forms which will be discussed further in Section 2.2. Incorporation of the delayed term leads to an infinite number of roots of the related characteristic equation, and makes analysis difficult using classical methods, especially, in determining stability and designing stabilizing controllers [155]. Additionally, the associated necessary conditions for optimal control delay systems are more difficult to construct and are structurally complex. The related minimum principle for the considered problem class leads to a boundary value problem which is retarded in the state dynamics and advanced in the costate dynamics [65].

Because of the difficulties that delays impose on classical control methods few analytic procedures exist for obtaining the solutions to optimal control delay systems. In the absence of delays, theoretical optimal control techniques usually become more difficult to apply when the system is of high order, nonlinear, or continuous-time varying. Even in cases when we can completely solve the variational problem by analysis, translating general theory into a minimizer for a specific problem can be a major undertaking [142]. Computer simulation can be used to determine solutions when the system is of high order or is subjected to complicated inputs that are not easily amenable to analytic solutions such as time-variations and delays [158]. Consequently, focus in the optimal control delay research community has turned to the development of robust numerical techniques to obtain sufficiently accurate estimates to time delay optimal control problems.

Several numerical algorithms have emerged for the approximation of solutions to optimal control delay systems. However, such documented approaches exhibit procedures that focus primarily on a class or specific type of optimal control delay problems (OCDPs), which pose several limitations on accuracy and robustness to the general optimal control network. In [31], Bock's direct multiple shooting algorithm is applied to obtain the numerical solution to an economic problem with constant control delays. In [71], the authors discuss a new forward and backward difference numerical algorithm for the approximation of solutions to an HIV infection immune response delay model. Solutions of time-delayed optimal control problems by the use of modified line-up competition algorithm are presented in [134]. Furthermore, [151] presents a parameterization and gradient computational formula to solve an optimal control switched dynamical system with time delay.

Realistic models increasingly demand the inclusion of delays in order to properly understand, analyze, design, and control real-life systems [6]. From the current literature archives it is evident that many more specialized optimal control delay numerical algorithms exist. Though favorable in some cases, these numerical contributions do not provide the ability to compute the solutions to more general optimal control delay problems. Therefore, it is imperative that a general purpose optimization suite exists for the numerical treatment of various types of optimal control delay systems.

2.2 Optimal Control Delay Systems

An optimal control delay system is an optimal control system such as Eq. (1.2) that incorporates a function Eq. (2.1) in the state and/or control variables. The incorporation of the delayed variable transforms the dynamic equations into a delay differential equation (DDE) system or delay differential algebraic equation (DDAE) system. DDEs are slightly different from ODEs in that its time-dependent solution is not uniquely determined by its initial state at a given moment but, instead, the solution profile on an interval with length equal to the delay or time lag has to be given [50]. This solution profile, $\alpha(t)$, is termed a startup or prehistory function and allows evaluation of the delayed system variables at time instances prior to the initial evaluation interval. Because of the impact that the startup function has on the behavior of the system, choice of prehistory is of particular importance in determining what solutions can be achieved. If an initial discontinuity exists, i.e. $(x(t_0))^+ \neq (\alpha'(t_0))^-$, a cascade of discontinuities could arise. Additionally, lack of consistency in the initial function can cause the solution to terminate prematurely after some bounded interval.

In general, the goal of an unconstrained optimal control delay problem is to minimize or maximize an objective function

$$J = \phi(T) + \int_{t_0}^{T} \mathcal{L}[x(t), u(t), x(\omega(t)), u(\eta(t)), t]\, dt, \tag{2.2a}$$

subject to the DDE

$$\dot{x}(t) = f[x(t), u(t), x(\omega(t)), u(\eta(t)), t], \qquad t_0 \leq t \leq t_f \tag{2.2b}$$

with startup functions

$$x(t) = \alpha(t), \qquad t_0 - r \leq t < t_0 \tag{2.2c}$$

$$u(t) = \beta(t), \qquad t_0 - s \leq t < t_0 \tag{2.2d}$$

and initial and terminal conditions

$$x_0 = q \text{ and } x_T = p. \tag{2.2e}$$

Here the terminal cost is $\phi(T) = \phi[x(T), u(T), T]$ and $\omega(t)$ and $\eta(t)$ are time delay functions. In the case of constant delays $r, s > t_0$ are fixed, and $\omega(t) = t - r$ and $\eta(t) = t - s$ describe the constant delay functions for the state and control respectively. Typically $T \lll \infty$. However, if $T = \infty$ Eq. (2.2) is an infinite horizon optimal control delay problem.

Like optimal control systems, many optimal control delay systems are nonlinear in nature and additionally contain algebraic inequality and equality path constraints of the form

$$\begin{aligned} 0 &= g[x(t), u(t), t], && (2.3) \\ 0 &\leq \tilde{g}[x(t), u(t), t]. && (2.4) \end{aligned}$$

Path constraints help to define the admissible region and may incorporate a combination of both state, control, or delay variables explicitly. Studies involving the incorporation of state delays in optimal control systems are far more common than those that discuss the incorporation of control delays, especially when it comes to analysis regarding convergence and stability. If the control function does not depend on future state values, it is noted that the case of control delays is equally challenging as state delays. However, even in the case of linear constant delay systems, optimal control problems with control delays are arguably the most challenging due to the limited results available for a narrow class of systems. Nonlinear optimal control delay systems are limited by the complexity outlined in the problem as well as the control design. In general, global stabilization of nonlinear input delay systems may not be possible due to the fact that solutions of nonlinear systems may not exist for all times.

The choice of delay highly impacts the behavior of the optimal control delay problem, yet making the solution more difficult to obtain and system dynamics all the more interesting. The delay term may possess various forms to accommodate the physical characteristics of the modeled system. In addition to constant delays some other formats for delays are time-varying, state dependent, neutral, and time advance. Time-varying delays are mathematically represented by a function in which the delay quantity is a function of time

$$\rho(t) = t - \tau(t), \quad \tau(t) > t_0. \tag{2.5}$$

Varying delays commonly occur in large scaled distributed and networked systems, such as the regulation of internet traffic and control over networked communication channels [86]. They are introduced by the randomness in the communication network links that interface the controller with the remote process. State-dependent delays are used to model system behavior that is impacted by the past state or position of the system, and is incorporated into many delay systems in a form similar to

$$\bar{\rho}(t) = t - \tau[x(t), t], \quad \tau[x(t), t] > t_0. \tag{2.6}$$

State-dependent delays appear in various applications such as cardiovascular-respiratory control systems [8], species growth ecological systems [91], and engineering milling processes [78]. Neutral delay functions are unique in that they involve both delayed state variables and derivatives of delayed state variables

$$\tilde{\rho}(t) = t - \tau[x(t), \dot{x}(t), t], \quad \tau[x(t), \dot{x}(t), t] > t_0. \tag{2.7}$$

Neutral-type delay systems can be found in such places as population ecology, distributed networks containing lossless transmission lines, heat exchangers, and robots in contact with rigid environments [73]. Furthermore, time advances are positive time delays. They are functions defined

$$\hat{\rho}(t) = t + \tau, \quad \tau > 0. \tag{2.8}$$

When implemented, these functions cause the behavior of the system to be impacted by a future decision or response. Though time advance functions are less common, they are well expressed in remote-sensing technological applications. These type of technologies use sophisticated computer vision operators that incorporate “look-ahead control policies, i.e. predictive control strategies where information about some of the future disturbances to the controlled system is assumed to be available [37, 74]. Along with (2.8) it is necessary to specify values for the state and/or control variables on the advanced interval, $[T,\ T - \tau]$ to achieve the complete time advance system.

Optimal control applications incorporate many of these delay types in both the state and the control variable. It is not uncommon to see time-varying or state-dependent delays in the control variable (e.g., [7, 82] and [5, 12]) or time advances in the state variable (e.g., [54]). Because of the involvement of state derivatives, neutral delays are limited to the state variable. Usually, derivatives of the control variables do not appear in the optimization system. This thesis work has investigated the solutions of optimal control problems with constant delay, time-varying delay, neutral delay, and time advanced arguments. Currently, the optimization delay package utilized for computation does not handle state-dependent delays. Hence, research concerning state-dependent delays is a topic for future study.

2.2.1 Facts About Delays

Regardless of the delay format all delays contribute to the enhancement of the complexity and overall difficulty for solving the optimization system. Because of the drastic side effects much attention in the optimization community has focused on studying the effects of time delay on the stability, performance, and solutions of control systems.

2.2.1.1 Instabilities

In control systems, it is known that time delays can degrade a system's performance, and even cause system instability [28]. To this end, optimal control problem delay instability has been extensively discussed [48, 153]. This instability is caused by the unsynchronized control forces causing the system to likely miss or overshoot its target in an oscillatory or repetitive fashion. Mathematically, time delays can alter the eigenvalues of the system. We illustrate this with a simple example. Consider the single state delay problem defined

$$\dot{x} = -x(t-\tau), \tag{2.9a}$$

$$x(t) = q, \qquad -\tau \leq t \leq 0, \tag{2.9b}$$

which has the associated characteristic equation

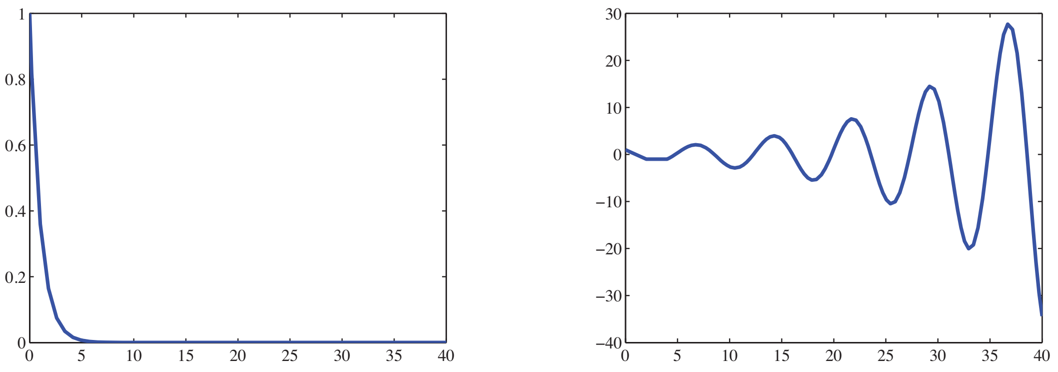
$$\lambda + e^{-\lambda\tau} = 0. \tag{2.10}$$

Letting $\lambda = a + ib$, then substituting into Eq. (2.10) and separating real and imaginary parts gives

$$a + e^{-a\tau}\cos(b\tau) = 0, \tag{2.11}$$

$$b - e^{-a\tau}\sin(b\tau) = 0. \tag{2.12}$$

When $0 \leq b\tau < \dfrac{\pi}{2}$, λ has negative real part. However, when $b\tau \geq \dfrac{\pi}{2}$, $\mathcal{R}e(\lambda)$ changes from negative to positive causing instability.



(a) $\tau = 0$ (b) $\tau = 2$

Figure 2.1: State Delay Problem: Eq. (2.9) solved with $q = 1$

2.2.1.2 Stabilizing Properties

While it is well highlighted in the literature that time delays result in unwanted system responses, we note that the presence of delays may in some cases be a benefit to the dynamic system. In addition to delays providing realistic model detail, delays have the capability to sometimes stabilize an unstable system [29]. Traffic control signals are one of the most literal representations of system stabilizers. Their goal is to create an optimal driving environment by delivering favorable signal timing delays to motorists to decrease likelihood of heavy traffic and collision. Similar properties can be observed in other applications such as communication networks [95, 111, 150, 156], control tracking error [87], or structural engineering [141].

Additionally, time delays are favorable for their ability to reduce the dimension of a system. The high-order system is usually replaced by a lower order system with time delays. An example of this technique is discussed in [154] in which chemical processes such as transcription and translation are lumped together and modeled as a single unit by a time delay in the reduced system [154]. When tracking error is considered as the performance criterion, it is shown that consistent time delays in the feedback path can actually reduce the steady-state tracking error of a control system to polynomial reference inputs (such as ramps) [87].

2.2.1.3 Analytic Techniques

With the increasing adaptation of the usage of optimal control delay systems to model complex industrial and mechanical processes, scientists have turned to studying the derivation of analytic solutions to these type of problems. It is known that for many practical systems analytic solutions are nearly impossible to obtain due to the limitations that time delays impose on the application of analytic techniques. Consequently, works regarding analytic solutions to optimal control delay systems are few in number, and in most cases specific to a class of problems.

The foundation of many of these formulas stem from the classical Pontryagin's minimum principle [114] which defines necessary conditions for obtaining optimality conditions for obtaining an admissible constrained control. The necessary conditions derived from quadratic optimal control of time delay systems generally leads to solving a two-point boundary value problem (TPBVP) with both time delay and time advance terms, which is very difficult to solve analytically with the exception of some simpler cases [137]. Suppose we want to minimize

$$J = \int_0^T x^2 + Ru^2 \, dt, \tag{2.13a}$$

subject to the dynamics

$$\dot{x}(t) = ax(t-\tau) + bu(t), \tag{2.13b}$$

$$x(t) = \phi(t), \qquad -\tau \le t \le 0. \tag{2.13c}$$

When $\tau = 0$ standard techniques apply and we have the associated necessary conditions

$$\dot{x}(t) = ax(t) + bu(t), \tag{2.14a}$$

$$-\dot{\lambda} = 2x(t) + a, \tag{2.14b}$$

$$0 = 2Ru(t) + \lambda(t). \tag{2.14c}$$

Based on [65, 66, 67] when $\tau > 0$ the necessary conditions are

$$\dot{x}(t) = ax(t-\tau) + bu(t), \tag{2.15a}$$

$$-\dot{\lambda}(t) = 2x(t) + a\,\chi_{[0,T-\tau]}(t)\lambda(t+\tau), \tag{2.15b}$$

$$0 = 2Ru(t) + \lambda(t)b, \tag{2.15c}$$

where

$$\chi_I(t) = \begin{cases} 1, & t \in [0,\ T] \\ 0, & \text{otherwise} \end{cases}, \tag{2.15d}$$

which displays a time advance in the costate variable. The idea of this forward and backward structure in state and costate is problematic because the terminal boundary condition is likely to depend on future history of the adjoint variable which could cause discontinuities. Note that the inclusion of more delays in Eq. (2.13) would only amplify the presence of the conflicting constraints.

In the case of DDEs and time delay control systems analytic techniques involving use of the

Lambert W function [77, 80, 128, 155], Padé approximations [136, 157], or Lyapunov matrix functions [3, 88, 101] may be applied to examine the stability of the time delay system, and hence provide an analytic solution. The Lambert W function provides a way to explicitly represent the infinite roots of the transcendental characteristic equation for time delay systems. Padé approximations represent time lags as rational functions resulting in a delay free equation. Lyapunov matrix functionals are used to determine exponential estimates for solutions of exponentially stable time delay systems. Because of the success, many of these analytic techniques have been integrated into the standard libraries of various commercial software packages utilized today.

While it has been proven that implementation of analytic techniques are effective they often require tedious derivations and result in lengthy solution formulations that are difficult to evaluate by hand. Additionally, much of the work concerning analytic techniques for optimal control delay systems and/or delay differential equation systems are based on the linear or linear time-varying cases. Hence to fully analyze OCD or DDE systems of great size and complexity, full-scale computer technology is necessary.

2.3 Time Delay Software and Optimization Tools

Over the past five decades the study of numerical techniques for the solutions of optimal control problems and delay differential equations has been a very active area of study. Numerical techniques were first proposed by Bellman in the late 1950s with his work on dynamic programming for multistage decision processes and optimal control systems [14]. In the 1960s Bellman and company made further contributions to numerical solutions of delay differential equations with the development of the method of steps (MOS) [15, 16, 17]. Although Pontryagin is highly referenced for theoretical contributions to optimal control, it is noted that his ideas sparked the creation of a class of numerical methods for dynamical systems, called indirect methods. The works of Pontryagin and Bellman have been the most influential in the history of optimization and dynamical systems. Their works combined have led to the development of a host of numerical methods that are utilized in a variety of disciplines. Although dynamic programming methods and indirect methods are independently powerful numerical techniques, newer research on optimization and time delay software focuses on hybrid numerical approaches which combine the two techniques [143].

2.3.1 Delay Differential Equation Numerical Techniques

The arena of methods for the numerical solutions for delay differential equations is quite well established and advanced now. Many techniques embedded in solvers today stem from either the method of characteristics (MOC) or the MOS. MOC is used for the simplest constant coefficient DDEs and involves use of the Lambert W function [51]. Since Bellman's introduction, the method of steps has been revisited by many authors [13, 102], and hard coded into several popular mathematical software packages such as MATLAB [124], Mathematica [53], Maple, and S-ADAPT [9]. It is a numerical method that provides an analytic solution to a delay differential equation system.

MOS is universally attractive for its nifty way of transforming a constant DDE system into an ODE system by eliminating the time delay variable. By construction the time interval is divided into N steps or subintervals of the length of the delay, τ. On the initial interval $[t_0 - \tau,\, t_0]$, we have a system of ODEs because the functional value of the delayed term is known. Note that the solution on this interval is unique, and can now provide an initial value for the ODE defined on the following interval, $[t_0,\ \tau]$. By design, it is required that MOS solutions be continuous at the boundaries of the internal intervals. The method of steps process results in a much larger system of ODEs that are simultaneously solved to obtain the solution for the corresponding DDE on $[t_0 - \tau,\ T)$, where T is some specified final time value.

MOS is advantageous in that it eliminates the delayed term and utilizes low computer storage. However, the numerical solution is somewhat complicated if there is a discontinuity introduced by the delay at the initial point. This discontinuity requires careful attention because it can propagate on successive intervals and increase solution error. Additionally, the number of differential equations which have to be handled increases with the number of steps selected. Thus if a system has dimension n, then the MOS system will have dimension nN.

Due to the disadvantages of the method of steps a direct approximation of the retarded argument from the integration points seems to be more favorable [104]. This inspired the adaptation of discrete methods to DDE systems. Several Runge Kutta methods have been employed in software for the treatment of DDEs. Many of these methods involve approximation of the retarded argument by way of an appropriate interpolant, and automatically considers jump discontinuities in the derivatives of the solution. Hermite interpolating polynomials are widely used for their ability to provide equally accurate solutions at nodes between mesh points. Employment of Hermite interpolants for DDEs can be noted in the Fortran DDE solvers, DKLAG6 and DDE_SOLVER [125], Matlab solvers dde23 [126] and ddesd [123], and other works [79, 104]. These solvers are primarily used for their simplicity, robustness, accuracy, and efficiency. The one limitation observed with most discrete numerical solvers for DDEs is that many of them are only applicable to constant lag systems. Hence, the numerical solution for multi-delayed systems is still open.

2.3.2 Optimal Control Numerical techniques

Optimal control numerical methods are characterized as either indirect methods (IMs) or direct methods (DMs). IMs employ an "optimize then discretize" approach. Pontryagin's minimum principle is first applied to the DDE to obtain an equivalent boundary value problem (BVP). The resulting BVP is then solved with a gradient, shooting, or collocation method. Since the optimal solution found must satisfy end-point and/or interior point conditions one must be very cautious when using the method of steps, as local conditions across the interior boundaries could be lost. IMs are nice in that utilization of the theoretical techniques increases the reliability, precision, and optimality of the solution. However, they suffer from difficulties in finding an appropriate initial guess for the multipliers, and are in part limited to unconstrained optimal control systems because associated necessary conditions may not hold for all state and control constraints. Consequently, direct methods are necessary.

Based on Bellman's dynamic programming techniques, direct methods employ a first "discretize then optimize" approach. The OCP is transcribed into a finite dimensional problem, resulting in a nonlinear programming problem (NLP). The resulting NLP is then solved by state-of-the-art numerical techniques. DMs avoid knowledge of the necessary conditions. Hence, they are able to handle multiple constraints on both state and control much easier. However, they only produce a suboptimal approximation. Despite this fact, they are used more than indirect methods due to their simpler implementation on constrained problems and their ability to quickly provide a solution with acceptable accuracy.

Control parameterization is a popular method often used to solve optimal control problems. In this context the running cost J_r becomes a state variable, and the differential equation

$$\dot{J}_r = L[x(t), u(t), t], \qquad J_r(t_0) = 0 \tag{2.16a}$$

is added to Eq. (1.2). To complete the augmented system the cost function is replaced by

$$J(u(t)) = \phi(T) + J_r(T). \tag{2.16b}$$

Before being put into the system the control is parameterized or characterized by a finite set of parameters. As the differential equations are integrated the optimizer is called to minimize the augmented cost (2.16b). Control parameterization can be easy to program. However, incorporating end-point equality constraints can be difficult because there is then a corresponding number of underdetermined terminal values for the adjoint variables and of course the terminal values from the integration do not necessarily satisfy the terminal equality constraints [147].

Over the past decade or so Matlab has been the language of choice for many of the optimal control packages used in academia, industry, and government laboratories. In general, these packages are not fit for solving time delay optimal control systems. However, they help to provide numerical solutions to the general closed and/or open loop optimal control problem. Matlab is connected to solvers such as Gauss Pseudospectral Optimal Control Software II (GPOPS-II) [113], Imperial College London Optimal Control Software (ICLOCS) [52], PROPT [118], DIDO [117], and MISER3 [81]. Optimal control packages written in other languages include C++ solver Pseudospectral Optimal Control Solver (PSOPT) [10] and Fortran solver Sparse Optimal Control Software (SOCS) [27]. PROPT, GPOPS, DIDO, and PSOPT employ pseudospectral theory which parameterizes the state and the control using global Legendre or Chebyshev polynomials, and collocate the differential algebraic equations using nodes obtained from a Gaussian quadrature. In nice enough situations these methods display fast convergence in the states, controls, and costates. However, when solving problems with rapidly changing solutions applying a very large degree polynomial may not even guarantee a respectable solution.

Once the OCP is transcribed into an NLP many of the Matlab based packages require the installation of an NLP solver such as Sparse Nonlinear OPTimizer (SNOPT) [61], Interior Point OPTimizer (IPOPT) [145], or Matlab's embedded NLP solver *fmincon*. Although Matlab is widely used for its user-friendly syntax, it can be disadvantageous when solving large systems because of its interpretative nature which causes it to run very slow. Developed by Boeing, SOCS is a general purpose software tool that discretizes the optimal control problem using Lobatto IIIA formulas. It takes advantage of sparse linear algebra techniques which increase software performance, speed, and the ability to solve very large systems on common desktop computers. The resulting NLP is solved by embedded sequential quadratic programming (SQP) or interior barrier point (IP) methods. Like many of the Matlab solvers, SOCS does not solve time delay optimization systems.

2.4 Dissertation Outline

In this thesis we discuss the solutions of nonlinear constrained optimal control time delay problems with an industrial grade direct transcription optimization delay software, Sparse Optimal Control Extended ($\mathbb{SOCX}$). The demand for a numerical technology to handle optimal control delay problems has been a growing concern for some time now. Because of the success of numerical methods for the solutions of optimal control problems and DDEs, it is propositioned that direct methods may be favorable for the approximation of solutions to single and multi-delay optimization systems. The arena of algorithms for the solutions of optimal control delay systems is sparse. A select few of the methods available can be observed in [31, 33, 41, 134, 148, 138]. Many of the methods that do exist are either limited to a specialized OCDP, to systems solely with delays in the state variables, or to the computation of solutions on uniform grids. In [119], PROPT showcases its ability to solve time delay optimal control systems by way of approximation of the delay arguments with Taylor series. To date there is no general purpose optimization package for the solutions of time delay optimal control systems available for commercial use.

This dissertation contributes to the areas of modeling, simulation, and optimization of constrained nonlinear time delay systems. More specifically, it introduces novel ways to handle both the state delay and the control delay variables by way of interpolation, constraints, and direct transcription. Our greatest contribution is a new method for handling optimal control systems with control delays. This dissertation expands upon previously published works and additionally provides details regarding our newest findings. This thesis is partitioned into three main discussion points: direct transcription and its implementation in $\mathbb{SOCX}$, challenges with control delay optimal control systems and the exogenous input control (EIC) method, and numerical and analytic validation of the EIC method, a remedy for control delay optimal control systems. Chapters are outlined as follows:

Chapter 3 provides details about direct transcription and how it is commonly used to solve optimization problems. Since the resulting product of direct transcription is a nonlinear programming problem, here we briefly discuss NLPs and common iterative methods for nonlinear optimization. A direct transcription approach is implemented in $\mathbb{SOCX}$. In Section 3.2 this approach is discussed, as well as the full numerical procedure implemented in the software. Furthermore, results for a simple delay problem and a commonly solved continuous stirred-tank reactor problem are provided to display software performance. A condensed version of this chapter can be found in [22].

Chapter 4 discusses challenges with control delay optimization systems. Due to the mathematical charcteristics of control variables, control delay variables are computed differently from state delay variables. As our experiments show in [21], this difference in calculation causes interesting things to happen. In this chapter we break down the numerical algorithm implemented in the software as it applies to control delay optimization systems in order to provide plausible reasoning for the difference in solution output from state delay optimization systems. Challenges with control delays motivated the development of the EIC method. The EIC method is referenced in [25], and is discussed in full detail in Section 4.2. To showcase numerical performance of the EIC method examples that illustrate difficulties in Chapter 3 are resolved, and the results are reported in Section 4.2.1.

Chapter 5 seeks to provide numerical and analytic validation of the exogenous input control method by way of examining the simple control delay test problem featured in Section 3.2.2.1. The test problem is of the simplest form necessary for the exogenous input control method to be applied, and serves as a framework for more complicated problems. Material in this chapter does not appear in the current literature. An unwanted consequence of the EIC method is a singularly perturbed optimal control delay system. We begin this chapter with discussion of this topic. Note that it is well documented that singular problems are very hard problems to solve via both analytic methods and numerical methods. In Section 5.1 and Section 5.2 method of steps and ε asymptotic expansions techniques are respectively applied to the EIC formulated simple delay problem in an effort to provide analytic support of convergence of solutions to the solutions of the original problem. Similar to the numerical scheme outlined in Chapter 2 for delay differential equation systems, method of steps can be applied to provide analytic solutions to some optimal control delay systems. The asymptotic expansion approach is based on work from O'Malley and Jameson [108, 109, 110]. It utilizes singular perturbation techniques that involve power series expansions about a small parameter, $\varepsilon > 0$ to formulate the solution to the perturbed optimal control system.

Chapter 6 surveys $\mathbb{SOCX}$ performance on the solutions to popular time delay systems.

Because $\mathbb{SOCX}$ is a general purpose delay optimization software tool, it is able to handle a variety of differently formulated OCDPs, DDAEs, and DDEs. Software results for a delay partial differential equation are given in [23], and for advance time, neutral delay, and mixed delay systems in [24]. In more detail, these results are printed here to showcase some additional features of $\mathbb{SOCX}$, as well as its flexibility and versatility.

Chapter 7 concludes this thesis with a summary of findings and contributions. Here we also point out some of the limitations of the methods proposed. Furthermore, possible continuations or extensions of the topics are discussed.

Chapter 3

Direct Transcription and Software

3.1 Direct Transcription

Direct transcription (DT) is the process in which a direct method is applied to solve a continuous time optimization system. The process transcribes the entire optimal control system into a large discrete nonlinear programming problem by fully discretizing the state and control variables by a selected numerical method.

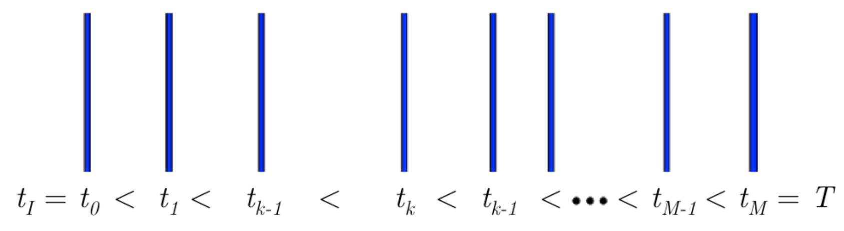

$t_I = t_0 < t_1 < t_{k-1} < t_k < t_{k-1} < \cdots < t_{M-1} < t_M = T$

Figure 3.1: Sample discrete time grid after subdivision on an interval $[t_I,\ T]$

We initially focus on an optimal control problem without delays. Given a differential algebraic equation model (3.1b), (3.1c), a cost (3.1a), and constraints (3.1d)

$$J = min \quad F[y(t), u(t), t], \tag{3.1a}$$

$$\dot{y} = f[y(t), u(t), t], \tag{3.1b}$$

$$0 = g[y(t), u(t), t], \tag{3.1c}$$

$$0 \leq \tilde{g}[y(t), u(t), t], \tag{3.1d}$$

we have an optimal control problem defined on $[t_I,\ T]$. Transcription begins with subdividing the time domain into a finite number of nodes or grid points as in Figure **??**. Note that the grid only consists of points. Vertical line segments were added to better visualize the grid spacing. Although many DT methods subdivide the interval into equally spaced grids points, the constructed mesh need not be uniform. The discrete state and control variables at the grid points are then defined $y_k = y(t_k)$ and $u_k = u(t_k)$ respectively, and are treated as the optimization variables in the NLP. In a similar fashion, the functional values are defined $f_k = f[y_k, u_k, t_k]$, $g_k = g[y_k, u_k, t_k]$, and $\tilde{g}_k = \tilde{g}[y_k, u_k, t_k]$. Then for the Euler's numerical method (3.1) generates a nonlinear programming problem that seeks to find a vector

$$x = [y_1, u_1, y_2, u_2, \ldots, y_M, u_M]^T, \tag{3.2a}$$

that

$$min \quad F(x), \tag{3.2b}$$

subject to the defect constraints

$$0 = y_{k+1} - y_k + h_k f_k, \qquad k = 1, \ldots M-1 \tag{3.2c}$$

and algebraic constraints

$$0 = g_k, \qquad k = 1, \ldots M \tag{3.2d}$$

$$0 \leq \tilde{g}_k, \qquad k = 1, \ldots M \tag{3.2e}$$

where $h_k = t_{k+1} - t_k$.

The resulting NLP is usually sparse, and can be solved by existing well-developed computer algorithms. Numerical methods adopted for direct transcription procedures are commonly based on pseudospectral collocation or Runge-Kutta integration principles instead of the simpler Euler's method used above. It is especially important to note that the direct method chosen influences the dimension of the resulting nonlinear optimization problem. For example, direct shooting or quasilinearization methods generate smaller NLP systems because of the underlying discretization of the already optimal or nearly optimal system. On the other hand, direct collocation methods can generate nonlinear optimization problems within a range of thousands to tens of thousands of variables and constraints. Consequently, employment of a sophisticated mesh refinement procedure that exploits the unique structure and sparsity of the equations is necessary for efficiency for any DT procedure. A full description of direct transcription procedures and nonlinear programming can be found in [18].

DT methods are popular for various reasons. Because of the "discretize then optimize" approach DT is free from requiring estimates for adjoint variables. In many complex problems actually getting the necessary conditions in a useful form can be a very difficult task because there can be a number of constraints going active and inactive with a complex switching structure [19]. DT can accommodate several constraints on both state and control variables. It is also well documented that a direct transcription approach can often provide solutions to many problems that are otherwise difficult or impossible to solve with other methods. DT methods outperform other methods when it comes to inequality constrained problems that exhibit complicated behavior near the constraint boundary and certain equality constrained problems with high index constraints [49]. Direct transcription methods, and in particular, methods involving orthogonal collocation have become quite popular in several field areas due to their high accuracy in approximating non-analytic solutions with relatively few discretization points [76].

In the remaining sections in this chapter we discuss the framework for a direct transcription optimization package in development for treatment of systems with time delays. Direct transcription has been successfully employed in a number of today's numerical solvers. Because of this success we propose that DT is able to combat the instability and difficulty caused by time delays, and provide adequate solutions to many optimization time delay systems. The direct transcription procedure studied extends that of the approach rooted in SOCS, a Boeing general-purpose optimal control software developed to solve very large trajectory, chemical process control, and machine tool path definition optimal control problems [27]. Similar to SOCS, the new optimal control algorithmic procedure is a three step process that first discretizes the continuous time system with implicit Runge-Kutta numerical methods. For (3.1), the transcription formulations are:

- Discretization for 2^{nd} Order Trapezoid (TR)

$$\text{NLP Variables: } x = [y_1, u_1, y_2, u_2, \ldots, y_M, u_M]^T, \tag{3.3a}$$

$$\text{Defect constraints: } 0 = y_{k+1} - y_k - \frac{h_k}{2}(f_k + f_{k+1}), \qquad k = 1, \ldots, M-1 \tag{3.3b}$$

- Discretization for 4^{th} Order Hermite-Simpson (HS)

  Define the midpoint discrete time values $t_{k+1/2} = \dfrac{t_k + t_{k+1}}{2}$ then we have the resulting NLP system,

$$\text{NLP Variables: } x = [y_1, u_1, y_{3/2}, u_{3/2}, y_2, u_2, y_{5/2}, u_{5/2}, \ldots, y_M, u_M]^T, \tag{3.4a}$$

$$\text{Defect constraints: } 0 = y_{k+1} - y_k - \frac{h_k}{6}(f_k + 4f_{k+1/2} + f_{k+1}), \quad k = 1, \ldots, M-1 \tag{3.4b}$$

where

$$f_{k+1/2} = f[y_{k+1/2}, u_{k+1/2}, t_{k+1/2}],$$

with

$$y_{k+1/2} = \frac{1}{2}(y_k + y_{k+1}) + \frac{h_k}{8}(f_k - f_{k+1}) \quad \text{and} \quad u_{k+1/2} = u(t_{k+1/2}).$$

Sparse linear algebra techniques are then applied to solve the NLP. Finally, the accuracy of the approximation is assessed. If the desired tolerance is not met, the discretization is refined and the optimization steps are repeated. What makes the new DT algorithm different is the incorporation of consistency policies and interpolation schemes for treatment of time delay variables. Information regarding the extended DT algorithm can be used to aid or improve code for both nondelay and delay optimal control numerical algorithms that have a similar structure or philosophy to ours.

3.2 Sparse Optimal Control Extended

Sparse Optimal Control Extended ($\mathbb{SOCX}$) is a high performance direct transcription software package composed of a collection of Fortran 90 routines for solving nonlinear optimization, optimal control, parameter estimation, and delay problems with equality and inequality constraints. It is apart of Sparse Optimization Suite (SOS) available from Applied Mathematical Analysis (AMA) [26]. Our research partnership with AMA is primarily outlined to investigate the solutions of continuous time optimal control systems with time delays. A study on the performance of $\mathbb{SOCX}$ on a large set of parameter estimation problems has already been conducted and can be viewed in [30]. For description and quick referencing purposes we state a simplified version of the general optimal control time delay problem that $\mathbb{SOCX}$ seeks to solve.

3.2.0.1 General Optimal Control Delay Problem Formulation

$$min \quad J(u) = \phi[y(t_f), u(t_f), p, t_f] + \int_{t0}^{tf} \mathcal{L}[y(t), u(t), y(t-r), u(t-s), p, t]\, dt, \tag{3.5a}$$

subject to the DDAE

$$\dot{y} = f[y(t), u(t), y(t-r), u(t-s), p, t], \tag{3.5b}$$

$$g_L \leq g[y(t), y(t-r), u(t), u(t-s), p, t] \leq g_U, \tag{3.5c}$$

for $t_0 \leq t \leq t_f$ with state delay $r > 0$ and control delay $s > 0$, with the startup functions defined

$$y(t) \;=\; \alpha(t), \qquad t-r \leq t < 0 \tag{3.5d}$$

$$u(t) \;=\; \beta(t), \qquad t-s \leq t < 0 \tag{3.5e}$$

where

- $p =$ parameters, $y =$ state, $u =$ control, $y(t-r) =$ delayed state, and $u(t-s) =$ delayed control,
- and Eq. (3.5c) contains state and control equality and inequality constraints.

Equality constraints are enforced by setting $g_L = g_U$. Note that Eq. (3.5) allows the OCDP to contain parameters. Similar to constraints, upper and lower bounds can be specified for parameters. The software accepts user defined initial grids. A unique thing about $\mathbb{SOCX}$ is that control variables, algebraic state variables, and delayed variables (both state and control type) are all considered algebraic variables and are treated the same. The approximation schemes will be discussed in Section 3.2.1. Furthermore, $\mathbb{SOCX}$ allows for both time delay variables and time advance variables to appear concurrently in the specified DDAE.

However, such variables are restricted from changing orientations, e.g., a time delayed variable cannot become a time advanced variable. This is primarily due to the design of the software. In the software the state and control variables are initialized before the optimization system is defined. Delay variables can only be constructed from previously defined state and control variables and are initialized in optimization and delay routines. This fact prevents a delay-to-advance switch. As we will see later there can be advances and delays in the same variable. An example of a $\mathbb{SOCX}$ driver for a sample program can be found in Appendix A.1.

3.2.1 $\mathbb{SOCX}$ Algorithm

For a problem with a formulation similar to Eq. (3.5) the software begins by rewriting the dynamics of the OCDP as a constrained differential algebraic equation or delay differential algebraic equation. This step is an automatic process and aids in simplifying the problem by removing delay variables from the right-hand side functions of the state equations. Delay variables are removed from state equations by way of enforcing consistency relationships between $y(t-r)$ and $u(t-s)$ and pseudo variables $w(t)$ and $v(t)$ respectively. Applying this consistency, the dynamics of (3.5) are written as a delay differential algebraic equation of the form

$$\dot{y} = f[y(t), w(t), u(t), v(t), p, t], \tag{3.6a}$$

$$g_L \leq g[y(t), w(t), u(t), v(t), p, t] \leq g_U, \tag{3.6b}$$

$$0 = w(t) - y(t-r), \tag{3.6c}$$

$$0 = v(t) - u(t-s). \tag{3.6d}$$

Next, direct transcription takes place with utilization of the default numerical discretization scheme which starts with the TR numerical discretization or Eq. (3.3) and switches to HS or Eq. (3.4) for higher order and accuracy. Because TR is a second order method it requires use of less grid points and fewer equations for computation. Hence, it is easier to get consistency on coarser grids. The initial TR iterations produce feasible approximations that help to reduce the work of the mesh refinement algorithm when moving to HS. This TR to HS switch helps to reduce the amount of iterations needed to reach the desired tolerance. The default switch can be overridden and either TR or HS can be used independently from start to finish. Subdividing the interval into a finite number of nodes Eq. (3.6) takes on a similar discrete form as outlined in (3.3) or (3.4) (depending on the choice of discretization) with the addition of discrete pseudo variables $w_k = w(t_k)$ and $v_k = v(t_k)$.

3.2.1.1 Treatment of Delay Variables

With the new DDAE implementation note that we have now added two new NLP constraints to consider

$$0 = w_k - y(t_k - r), \tag{3.7a}$$

$$0 = v_k - u(t_k - s). \tag{3.7b}$$

These constraints contain delay quantities and must be carefully treated when being inputted into the dynamics of the system. Recall that time delays require startup functions or prehistory values for time values that are less than the initial time value or when applicable post history functions for time values greater than the final time value. However, what is unspecified is how delayed variables are handled throughout the integration interval. Note that a state delay variable $y(t - r)$ is defined on the interval $[t_0 - r,\ t_f - r]$, and is therefore approximated at the nodes of the grid $\mathcal{G}_r = t - r$. Similarly, a control delay variable $u(t - s)$ is approximated at the nodes of a grid $\mathcal{G}_s = t - s$. Evaluations of this type are needed for each and every delayed state and delayed control variable that appears in the dynamics of the problem. One would instantly suggest multiple grids. Why not? Calculations on multiple grids can lead to issues of increased execution time, storage and memory overflow, and overall a very inefficient code that is very slow. To combat these difficulties a better suggestion would be numerical interpolation. A number of studies such as [2, 133] have shown that interpolating schemes are proper mechanisms for handling delayed terms.

On a single interval the $\mathbb{SOCX}$ Runge-Kutta implicit methods assume knowledge about the values at each of the interval end-points t_k, t_{k+1}, and also the interval midpoints $\bar{t}_k = t_{k+1/2}$ for HS. All interpolation schemes consider this structure and employ a "look-back" consistency policy which forces the routine to look-back at an interval $[t_j,\ t_{j+1}]$ and evaluate the delay variable at a grid point $t_k - \tau$, $\tau > 0$ using the endpoints and or midpoint values defined on that interval. More specifically, it is required that the offset time $t_k - \tau$, $\tau > 0$ satisfy

$$t_j \leq t_k - \tau < t_{j+1}, \tag{3.8a}$$

for some j. Again, $h_j = t_{j+1} - t_j$ denotes the length of the interval and is not necessarily equal on subsequent intervals. Let the midpoint length of the interval be defined $\bar{h}_j = \dfrac{h_j}{2}$. Depending on the choice of discretization, the interpolation carried out may be different. Recall that interpolation is only carried out on the integration interval $[t_0, t_f]$ as delay variables assume the prehistory functional values specified on the delay or startup intervals.

In $\mathbb{SOCX}$, control delay and algebraic variables are expressed in terms of linear piecewise interpolating polynomials for TR and in terms of quadratic piecewise interpolating polynomials for HS. Mathematically, for a control delay function $\gamma_k = t_k - s \in [t_j,\ t_{j+1})$ we define the location of γ_k relative to the beginning of the time interval $[t_j,\ t_{j+1}]$ as $\delta_c = \dfrac{\gamma_k - t_j}{h_j}$. The subscript c denotes that the described quantity refers to the control delay variable. The pseudo control variable then satisfies

$$v_k = u(\gamma_k) = \begin{cases} \beta(t) & \text{if} \quad \gamma_k < 0 \\[2ex] \underbrace{(1-\delta_c)u_j + \delta_c\, u_{j+1}}_{\text{TR}} & \text{if} \quad \gamma_k \geq 0 \text{ and } \gamma_k \in [t_j,\ t_{j+1}) \\[2ex] \underbrace{(1-\delta_c)a_1\, u_j - a_2\, u_{j+1/2} + \delta_c a_3\, u_{j+1}}_{\text{HS}} & \text{if} \quad \gamma_k \geq 0 \text{ and } \gamma_k \in [t_j,\ t_{j+1}) \end{cases}, \tag{3.8b}$$

with coefficients defined

$$a_1 = \frac{t_{j+1/2} - \gamma_k}{\bar{h}_j}, \quad a_2 = \frac{(\gamma_k - t_j)(\gamma_k - t_{j+1})}{\bar{h}_j^2}, \quad \text{and} \quad a_3 = \frac{(\gamma_k - t_{j+1/2})}{\bar{h}_j}. \tag{3.8c}$$

Whether the implicit TR or HS discretization is selected, $\mathbb{SOCX}$ approximates all state delay variables via a cubic Hermite interpolating polynomial. Unlike control variables, state variables are integrated when applying implicit Runge-Kutta methods. Consequently, knowledge of the state derivatives at end-points on an interval is required. We use the state derivatives to generate the Hermite interpolating polynomial. Similar to the control case, linear interpolation or quadratic interpolation could be used to approximate state delays. However, the use of Hermite interpolation for state delays has shown to be more favorable [2, 104, 133].

For the state delay function $\omega_k = t_k - r \in [t_j,\ t_{j+1})$ we define the location of ω_k relative to the beginning of the time interval $[t_j,\ t_{j+1}]$ as $\delta_s = \dfrac{\omega_k - t_j}{h_j}$. The subscript s denotes that the described quantity refers to the state delay variable. The pseudo state variable then satisfies

$$w_k = y(\omega_k) = \begin{cases} \alpha(t) & \text{if} \quad \omega_k < 0 \\ c_1 y_j + c_2 y_{j+1} + c_3 h_j \dot{y}_j + c_4 h_j \dot{y}_{j+1} & \text{if} \quad \omega_k \geq 0 \text{ and } \omega_k \in [t_j,\ t_{j+1}] \end{cases}, \quad (3.8d)$$

where the coefficients c_i are defined

$$c_1 = (1 - 3\delta_s^2 + 2\delta_s^3), \quad c_2 = \delta_s^2(3 - 2\delta_s), \quad c_3 = \delta_s(1-\delta_s)^2, \quad \text{and} \quad c_4 = \delta_s^2(\delta_s - 1). \quad (3.8e)$$

Employing the interpolation schemes for delay variables completes the formulation of the NLP system to be solved. The resulting NLP is solved with an embedded sequential quadratic programming or interior-point solver. $\mathbb{SOCX}$ increases efficiency of the algorithm by automatically constructing and exploiting the sparsity of the large Jacobian and Hessian matrices. The accuracy of the finite dimensional problem is assessed by the mesh refinement algorithm at each iteration. The goal of the mesh-refinement procedure is to select the number and location of the grid points in the new mesh as well as the order of the new discretization. In other words, the algorithm first seeks to reduce the error globally, and then addresses equidistribution of the errors. If necessary the discretization is refined, and the optimization steps are repeated. With special treatment of the delay variables the direct transcription algorithm can be extended to solve various types of DDE, DDAE, and delay optimization systems. Determining the best way to formulate and solve a delayed optimal control problem when using direct transcription is a key focus of this research and is discussed throughout this thesis.

3.2.1.2 $\mathbb{SOCX}$ Output

To use $\mathbb{SOCX}$ the user is required to construct a subroutine for the dynamics to be solved. Along with the subroutine the user is not required to, but is allowed to specify a grid along with an initial guess for both the state and control variables. $\mathbb{SOCX}$ has a very detailed, yet unique output. The printed output from the suite of optimization programs is categorized in four different levels Terse, Standard, Interpretive, and Diagnostic. The default level of output for all subprograms in the optimization suite is set to Standard output. Standard output first prints information regarding the structure of the optimal control problem being solved such as: the initial values for the state, inequality constraints for the control variables, coefficient of cost function, as well as the Jacobian for both the functional and quadrature with respect to both state and control. This optimal control blueprint allows you to determine if the problem setup is correct even before looking at the solutions.

Also included in this type of output are analysis grids which display the grid point number, time value, and respective numerical approximation for the state, control, and delay variables. The design of these grids are unique to the class of problems and or the method used to approximate the solution, e.g. output from an optimal control delay problem is different from a method of steps problem. Currently, there is no general code available that can automatically format $\mathbb{SOCX}$ output for graphical purposes. We have developed Matlab drivers to process data from the analysis grids of standard constrained optimization and MOS formulated systems. These drivers are included in Appendix A.2.

Furthermore, for a successfully executed run $\mathbb{SOCX}$ prints a program summary at the end of the output file. It includes details about the grid number, number of points per grid, number of function calls, differential equation error, and computational time. This program summary is very informative as it gives a general ideal of how the mesh algorithm performed throughout computation. Presence of discretization error in the generated solutions is displayed by an increase in the number of grid points on successive iterations. Generally, if the number of grid points nearly doubles from one grid to another a total grid refinement was issued because of a very large discretization error. $\mathbb{SOCX}$ enforces a grid preservation law which requires the new mesh to always contain the grid points of the old mesh i.e., a new grid $\mathcal{G}_n$ consists of the old grid $\mathcal{G}_o$ plus the new points added to the intervals with maximum error.

3.2.2 Test Problems

The computational studies presented in this section were performed using $\mathbb{SOCX}$Version 2011.02 on a server consisting of dual 3 Ghz quad core Intel Xeon processors (8 total cores) with 8GB RAM. Newer versions of $\mathbb{SOCX}$ with a few updated and newly added features are either currently available or under development. In conjunction with this research we have prepared a test set in [20] to help guide and evaluate the software as it is being developed. Note that this document is an evolving test set as more delay optimal control problems are always desirable. The test set contains a number of biological, chemical, and optimal control delay problems of various levels of complexity. Here we solve three problems from that test set to showcase $\mathbb{SOCX}$ performance and overall efficiency.

The $\mathbb{SOCX}$ solutions for each problem were generated with an initial solve on a uniform grid of 21 points. Note that the initial grid is not required to be uniform as $\mathbb{SOCX}$ allows the user the freedom to choose the initial grid if desired. For comparison purposes each problem is solved using the method of steps. Here, MOS is carried out by way of transforming the original problem into a set of ordinary differential equations with appropriate boundary conditions. The transformed problem is then solved with the default switch from TR to HS direct transcription algorithm. State, control, and delay variables are plotted in separate figures for quick referencing and simplicity. In the grid analysis comparisons $\mathcal{G}_k$ is the k^{th} grid, N represents the number of grid points, F_c is the number of function calls, ϵ refers to the equation error, and $Time$ is the computation time. To prevent repetition of solution description we define the master legend detailing the keys for all solution plots featured in this section in Figure 3.2. Depending on the problem solved and discretization chosen on average $\mathbb{SOCX}$ indicates absolute errors of 10E-06 for state approximations and 10E-03 for control approximations. A full view of the test set can be found in [20].

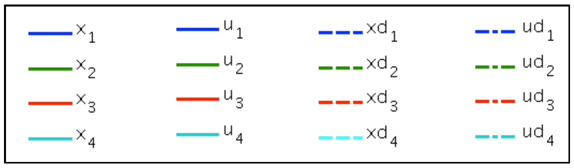


Figure 3.2: Master legend for graphs featured in Chapter 3

3.2.2.1 State Delay Problem

A state delay optimal control model that describes the immune response (IR) of a pathogenic disease process is developed in [132]. The problem is a four dimensional system with 4 state variables, 4 control variables, and two state delay variables. The goal is to minimize the therapeutic treatment cost quantified by

$$F = \frac{1}{2}\left(x_1^2(t_F) + x_4^2(t_F)\right) + \frac{1}{2}\int_0^{t_F} x_1^2(t) + x_4^2(t) + \|u(t)\|\, dt, \tag{3.9a}$$

subject to the delay equations

$$\dot{x}_1(t) = (a_{11} - a_{12}x_3(t))x_1(t) + b_1u_1(t), \tag{3.9b}$$

$$\dot{x}_2(t) = a_{21}(x_4(t))a_{22}x_1(t-r)x_3(t-r) - a_{23}(x_2(t) - x_2^*) + b_2u_2(t), \tag{3.9c}$$

$$\dot{x}_3(t) = a_{31}x_2(t) - (a_{32} + a_{33}x_1(t))x_3(t) + b_3u_3(t), \tag{3.9d}$$

$$\dot{x}_4(t) = a_{41}x_1(t) - a_{42}x_4(t) + b_4u_4(t), \tag{3.9e}$$

for $0 \leq t \leq t_F = 10$ with state delay $r = 1$, and startup functions given by

$$x_1(t) = 0, \quad -r \leq t < 0 \tag{3.9f}$$

$$x_3(t) = 3. \quad -r \leq t < 0. \tag{3.9g}$$

Define the function

$$a_{21}(x_4(t)) = \begin{cases} \cos(\pi x_4) & \text{if} \quad 0 \leq x_4(t) \leq \dfrac{1}{2} \\ 0 & \text{if} \quad \dfrac{1}{2} \leq x_4(t) \end{cases}, \tag{3.9h}$$

and initial conditions

$$x(0) = [3,\ 2\ , 4/3\ , 0]^T. \tag{3.9i}$$

The problem coefficients are defined

$$\begin{array}{llll} a_{11} = 1, & a_{12} = 1, & a_{22} = 3, & a_{23} = 1, \\ a_{31} = 1, & a_{32} = 1.5, & a_{33} = 0.5, & a_{41} = 1, \\ a_{42} = 1, & b_1 = -1, & b_2 = 1, & b_3 = 1, \\ b_4 = -1, & x_2^* = 2. & & \end{array} \tag{3.9j}$$

The analytic solution for IR is plotted in Figure 3.3 and the $\mathbb{SOCX}$ solutions are plotted in Figure 3.4. Since $\mathbb{SOCX}$ outputs the approximations for delay variables the delayed state approximation is plotted as well. In Figure 3.4c observe the application of the startup functions on the interval [0, 1]. On (1, 10] the delay variables assume the values of the corresponding state variables as desired. Comparing MOS and $\mathbb{SOCX}$ solutions we see that the trajectories in each figure display similar trends. The optimal controls decay rapidly on [0, 2] as shown in Figures 3.3b and 3.4b. In Figures 3.3a and 3.4a note the sharp corner in the trajectory of

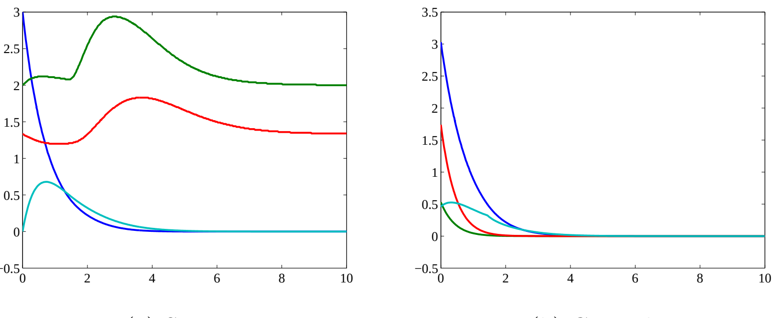

(a) States (b) Controls

Figure 3.3: MOS solution for IR in Eq. (3.9)

$x_2(t)$. The corner occurs from the nondifferentiable function a_{21} when $x_4(t) = 0.5$. Normally, the presence of a corner can cause integration failure. However, the DT algorithm is able to escape this fact because it seeks to locally refine the grid near the point where the smoothness is missing and the iterations continue until the corner is no longer numerically significant. This clustering of additional grid points near the time of reduced smoothness is indicated by the darkest band in Figure 3.5 which is formed by tightly packed grid points.

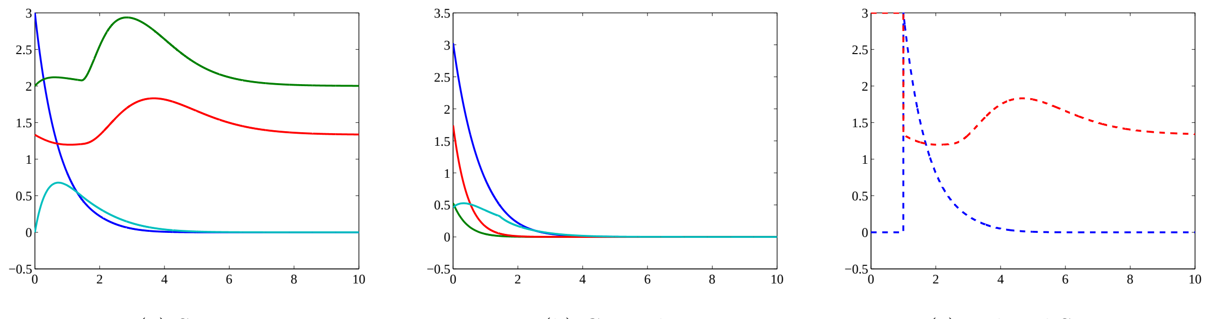

(a) States (b) Controls (c) Delayed States

Figure 3.4: $\mathbb{SOCX}$ solution for IR in Eq. (3.9)

It appears that the direct transcription algorithm is working well to approximate the solutions of optimal control problems with state delays. Estimated max errors are presented in

Table 3.1 to describe the closeness of the $\mathbb{SOCX}$ approximation to truth. Figure 3.2 summarizes the numerical results and mesh refinement performance for $\mathbb{SOCX}$, and presents comparable information when the problem is solved using the method of steps. For this problem we solve the method of steps formulated problem on an initial grid of 21 points also to showcase the drastic difference in execution mechanics. Although the method of steps solution yielded a smaller final grid, the computation time is much longer (about 2 times). MOS can be efficient in some cases, but is very slow because of the large systems needed to be solved.

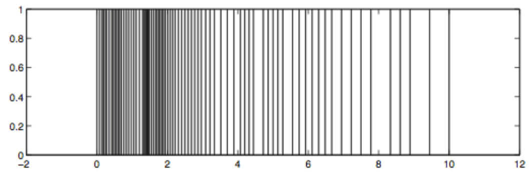


Figure 3.5: Bar graph representation of final time grid for IR in Eq. (3.9)

Table 3.1: Estimated absolute max errors for IR in Eq. (3.9)

State	Error	Control	Error
$x_1(t)$	4.8E-03	$u_1(t)$	5.4E-03
$x_2(t)$	5.1E-03	$u_2(t)$	6.1E-04
$x_3(t)$	5.0E-03	$u_3(t)$	5.3E-03
$x_4(t)$	6.4E-04	$u_4(t)$	5.5E-04

Table 3.2: Solution comparison for IR in Eq. (3.9)

$\mathbb{SOCX}$ Solutions

$\mathcal{G}_k$	N	F_c	F_e	ϵ	Time
1	21	11	111	7.19E-02	3.20E-02
2	41	8	143	4.57E-03	6.14E-02
3	41	6	1271	8.90E-04	1.80E-01
4	81	4	418	9.92E-05	1.90E-01
5	87	3	371	3.61E-06	1.76E-01
6	173	3	371	1.57E-07	4.91E-01
7	185	3	371	9.54E-08	5.39E-01
Total	185	38	3056		1.67E+00

MOS Solutions

$\mathcal{G}_k$	N	F_c	F_e	ϵ	Time
1	21	12	150	7.39E-04	1.82E-01
2	41	5	104	3.86E-05	5.64E-01
3	41	5	1049	1.16E-05	1.88E+00
4	48	3	399	7.70E-07	7.97E-01
5	53	3	399	6.44E-08	9.87E-01
Total	53	28	2101		4.42E+00

3.2.2.2 Control Delay Problem

Studies show that optimal control systems with time delays in the state variable are more frequently considered than time delays in the control variable. Time delays in the control occur just as often as they do in the state. However, they are less often studied for the complexity that they introduce into the optimal control system when they depend on future values of the state variable. A minimum energy problem with a delay in the control variable is featured in [7]. The optimal control problem is to minimize the state $x(t)$ using the minimum energy of control $u(t)$. Consider the scalar linear system

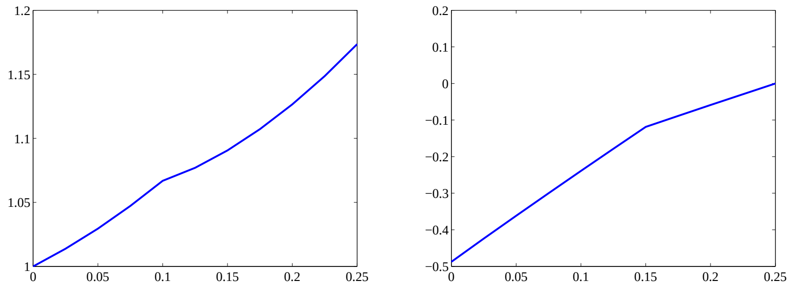
$$\dot{x}(t) = x(t) + u(t - 0.1) + u(t), \tag{3.10a}$$

with the initial condition $x(0) = 1$ and $u(s) = 0$ for $s \in [-0.1,\ 0)$. The optimal control problem is to find the control $u(t),\ t \in [0,\ T]$, that minimizes the criterion

$$J = \frac{1}{2}\int_0^T u^2(t) + x^2(t)\ dt, \tag{3.10b}$$

where $T = 0.25$.

The true solution for Eq. (3.10) is plotted in Figure 3.6. Observe that both the state and control are strictly increasing functions. On $[0,\ 0.1]$ the state is linear as it is impacted by the control delay. The control is being minimized with $u(0.25) \approx 0$. The values of the state and the objective function at the final moment $T = 0.25$ are $x(0.25) = 1.173$ and $J(0.25) = 0.154$. $\mathbb{SOCX}$ terminated with a solution within a tolerance of 10E-07 in 6 iterations with a final grid of 73 points. The final iteration is plotted in Figure 3.7.

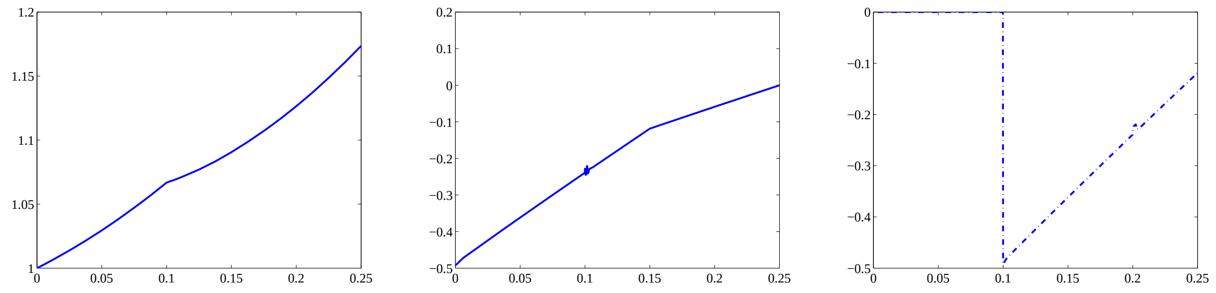


(a) State (b) Control

Figure 3.6: MOS solution for the control delay problem in Eq. (3.10)

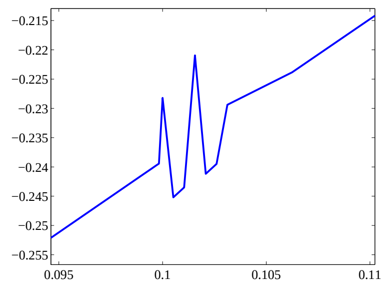
Graphically the final state and control solutions are comparable to the MOS solutions. However, we see some odd behavior in the control near $t = 0.1$. This behavior is carried over into the delay control approximation as shown in Figure 3.7c. In Figure 3.8a we zoom in at this node to get a better view of the solution. Near $t = 0.1$ there are a couple of spikes and dips in the solution. We also plot the first iteration in Figure 3.9 to get a view of the solution on the initial uniform grid of 21 points. Note that the solution results for both state and control are smooth at this iteration. Associated max errors associated with this run are $\|u(t) - u^*(t)\|$=1.42E-02 and $\|x(t) - x^*(t)\|$=4.49E-04.

Comparing the $\mathbb{SOCX}$ and MOS execution data we see some drastic differences in Table 3.3. Note that the MOS run terminated just after 4 iterations with a computation time of $0.01s$ and the $\mathbb{SOCX}$ run terminated at 6 iterations with a computation time of $0.06s$. Although the



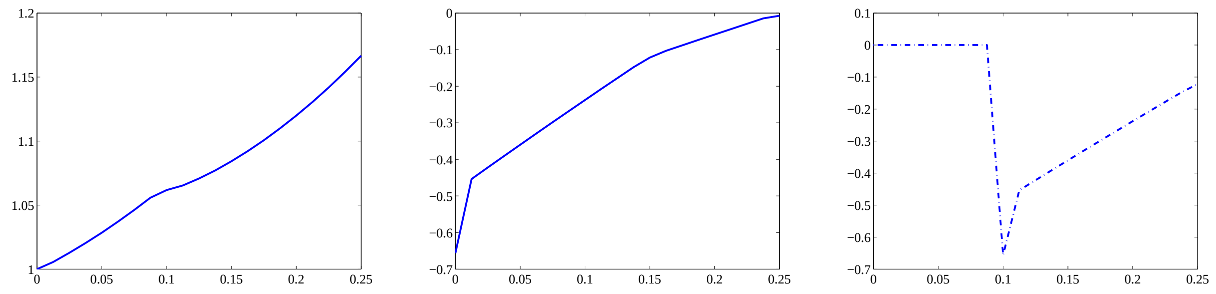
(a) State (b) Control (c) Delayed Control

Figure 3.7: Final iteration $\mathbb{SOCX}$ solution for the control delay problem in Eq. (3.10)



(a) Control

Figure 3.8: Close-up of control for the control delay problem in Eq. (3.10)



(a) State (b) Control (c) Delayed Control

Figure 3.9: Iteration 1 $\mathbb{SOCX}$ solution for the control delay problem in Eq. (3.10)

$\mathbb{SOCX}$ run was 6 times slower, for fewer grid points more function evaluations were necessary for the MOS run. This is primarily due to the multiple ODE equations needed to be solved

Table 3.3: Control delay problem solution comparison

$\mathbb{SOCX}$ Solutions

$\mathcal{G}_k$	N	F_c	F_e	ϵ	Time
1	21	4	25	2.03E-04	5.63E-03
2	28	3	21	9.85E-05	4.84E-03
3	28	3	71	3.41E-04	8.50E-03
4	41	3	71	2.92E-06	1.16E-02
5	48	3	71	1.77E-05	1.31E-02
6	58	3	71	2.04E-08	1.56E-02
Total	58	19	330		5.94E-02

MOS Solutions

$\mathcal{G}_k$	N	F_c	F_e	ϵ	Time
1	2	4	25	1.67E-04	4.22E-03
2	3	3	21	2.17E-05	3.19E-03
3	3	3	71	1.04E-08	3.51E-03
4	3	4	88	1.04E-08	3.65E-03
Total	3	14	205		1.45E-02

at each iteration. Here, the method of steps terminates faster possibly due to the local error reduction strategies of the mesh refinement algorithm in $\mathbb{SOCX}$. In a standard framework, more grid points may be necessary to achieve an error tolerance of order 10E-08. This indeed shows a true advantage of the routines within the $\mathbb{SOCX}$ algorithm.

3.2.2.3 CSTR Mixed Delay Problem

In [66], a mixed delay problem or problem involving state and control delays describes a nonlinear continuous stirred tank reactor (CSTR) system that runs an irreversible chemical reaction. For a very complex described system the integration interval seems to be very short. However, note that chemical reactions take place rather quickly so the short integration interval for this problem is appropriate. The CSTR problem contains three states, two controls, one delayed state, and one delayed control variable. The state delay time value r does not equal the control delay time value s. Hence, the state and control variables are delayed at different times. Additionally, the delayed state and delayed control assume different functional values on their respective startup intervals. The analytic solution is plotted in Figure 3.10. The goal is to minimize

$$F = \int_0^{0.2} \|x\|_2^2(t) + .01\|u\|_2^2(t)\, dt, \tag{3.11a}$$

subject to the delay equations

$$\dot{x}_1 = -x_1(t) - R(t,x), \tag{3.11b}$$

$$\dot{x}_2 = -x_2(t) + 0.9u_2(t-s) + 0.1u_2(t), \tag{3.11c}$$

$$\dot{x}_3 = -2x_3(t) + 0.25R(t,x) - 1.05u_1(t)x_3(t-r), \tag{3.11d}$$

for $0 \le t \le 0.2$ with state delay $r = 0.015$, control delay $s = 0.02$, and startup functions given by

$$x_3(t) = -0.02, \qquad -r \le t \le 0 \tag{3.11e}$$

$$u_2(t) = 1, \qquad -s \le t \le 0 \tag{3.11f}$$

where the function

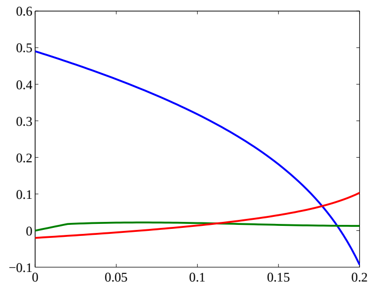
$$R(t,x) = (1 + x_1(t))\,(1 + x_2(t))\exp\left(\frac{25x_3(t)}{1 + x_3(t)}\right), \tag{3.11g}$$

with initial conditions and control bound

$$x(0) = [0.49,\ -0.0002,\ -0.02]^T, \tag{3.11h}$$

$$|u_1(t)| \le 500. \tag{3.11i}$$

Because the control delay example exhibited some unwanted oscillations in the final iteration, we plot the $\mathbb{SOCX}$ solutions to the CSTR problem in steps. In Figure 3.12 the $\mathbb{SOCX}$ approximations are given for the first iteration. Note that the state trajectories appear smooth. The controls both appear continuous but $u_1(t)$ has a corner at 0.02 and $u_2(t)$ has a corner at 0.18. This behavior is not present in the MOS solutions so we know that the generated solutions are not optimal. On successive iterations we notice the familiar odd behavior in the delay con-


(a) States

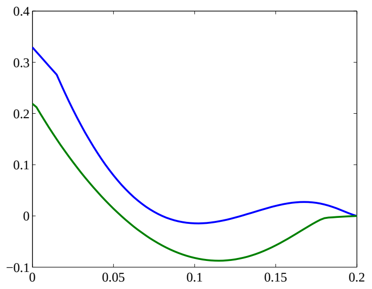
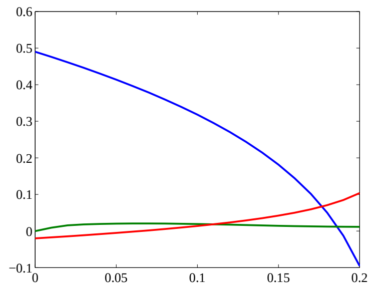
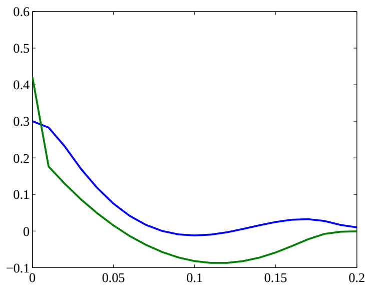
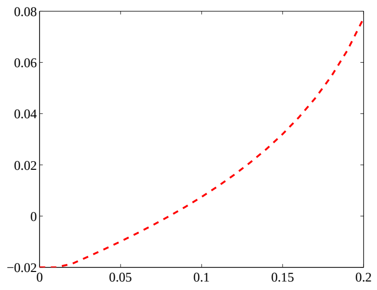

(b) Controls

Figure 3.10: MOS solution for the CSTR problem in Eq. (3.11)

trol variable $u_2(t)$ as shown in Figure 3.12b and Figure 3.12d. In the control graph the chatter is confined to the interval $[0.1,\ 0.15]$.


(a) States


(b) Controls


(c) Delayed State

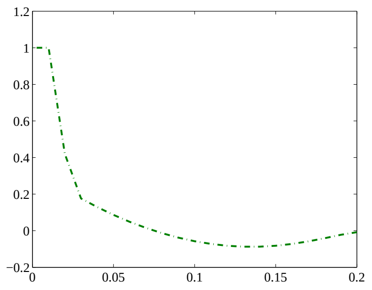
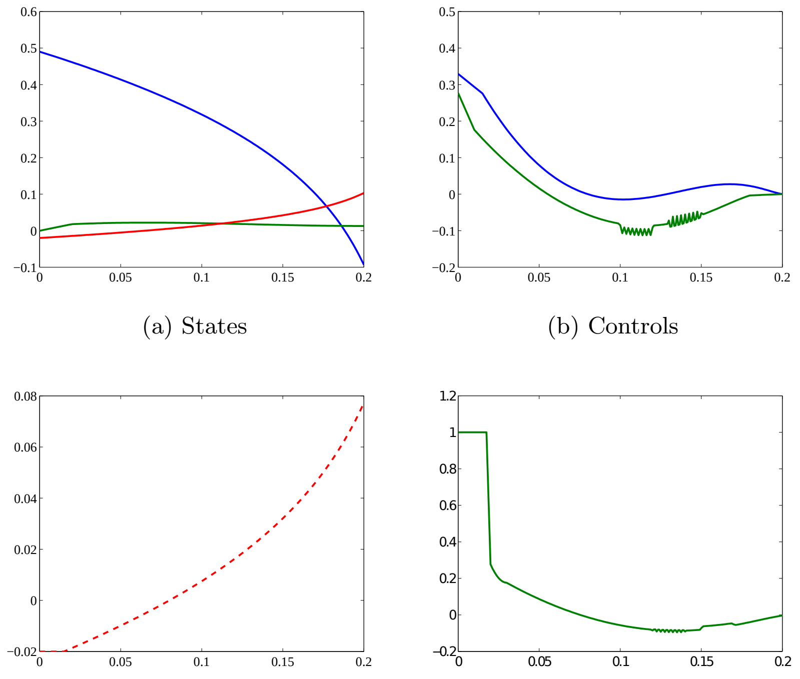

(d) Delayed Control

Figure 3.11: Iteration 1 $\mathbb{SOCX}$ solution for the CSTR problem in Eq. (3.11)

This is very different from the control approximation output in Eq. (3.10) since the chatter-like behavior appeared near the time delay value $t = 0.1$. Recall that the control delay value is 0.02. Here $u_2(t)$ has a jump discontinuity at $t = 0$ since its prehistory was set equal to one. If error were to result in the approximation of the control delay variable one would expect for it to occur at this point. However, this is not the case which is very interesting.



(a) States

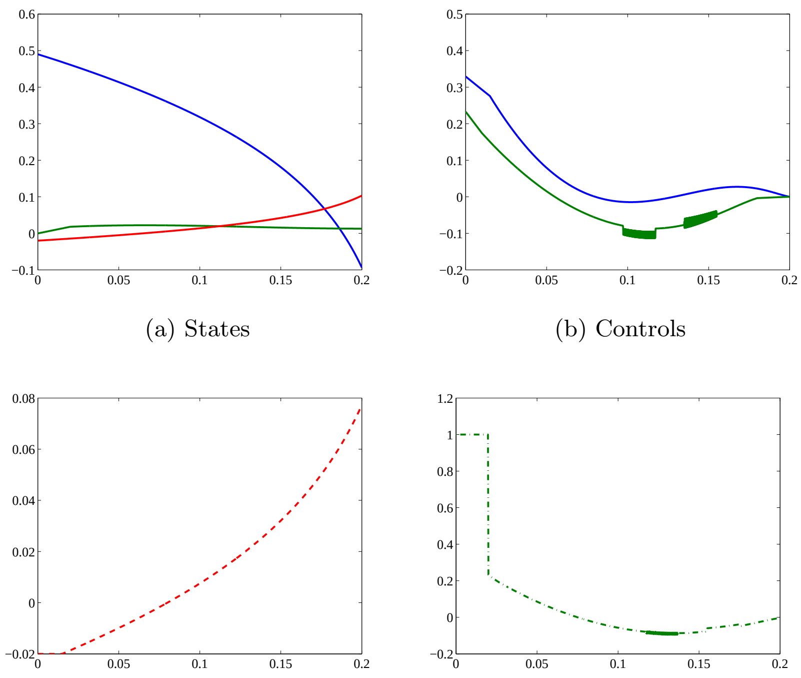
(b) Controls

(c) Delayed State

(d) Delayed Control

Figure 3.12: Iteration 5 $\mathbb{SOCX}$ solution for the CSTR problem in Eq. (3.11)

In Figure 3.13 we see that the present chatter becomes more dense at the final iteration. Note that the state approximations appear unaffected by the unwanted changes in the control solution. Although there is the presence of oscillatory behavior in the $\mathbb{SOCX}$ solutions trajectories produced display similarities to the MOS solutions. We see just how close the two approximations are by analyzing the side-by-side comparison of the MOS and $\mathbb{SOCX}$ solutions



(a) States (b) Controls

(c) Delayed State (d) Delayed Control

Figure 3.13: Final Iteration $\mathbb{SOCX}$ solution for the CSTR problem in Eq. (3.11)

featured in Table 3.5. In Table 3.4 at the final iteration all states indicate an estimated max error of order 10E-05. The delayed control variable has a max error of 1.68E-02.

Table 3.4: Eq. (3.11) Estimated max errors

Variable	Iter 1	Iter 5	Iter 8
$x_1(t)$	4.3E-03	8.0E-05	6.5E-05
$x_2(t)$	2.5E-03	2.3E-04	5.6E-05
$x_3(t)$	1.0E-04	1.9E-05	1.3E-05
$u_1(t)$	2.9E-02	6.1E-05	1.4E-04
$u_2(t)$	2.0E-02	5.8E-02	1.7E-02

Table 3.5: CSTR problem solution comparison

$\mathbb{SOCX}$ Solutions

$\mathcal{G}_k$	N	F_c	F_e	ϵ	Time
1	21	38	583	2.71E-04	6.14E-02
2	21	5	320	2.71E-04	2.33E-02
3	41	4	283	6.78E-05	4.36E-02
4	81	4	283	1.69E-05	1.05E-01
5	161	5	320	4.23E-06	4.03E-01
6	321	5	320	1.05E-06	1.81E+00
7	641	4	283	2.63E-07	7.62E+00
8	1281	4	283	6.59E-08	2.92E+01
Total	1281	69	2675		3.93E+01

MOS Solutions

$\mathcal{G}_k$	N	F_c	F_e	ϵ	Time
1	2	21	5986	1.25E-04	1.10E-01
2	3	6	469	1.73E-05	6.82E-02
3	3	4	5758	3.77E-08	2.84E-01
Total	3	31	12213		4.63E-01

3.2.2.4 Results Summary

In this section we have solved a state delay, control delay, and mixed delay optimal control problem with $\mathbb{SOCX}$. The results observed display that $\mathbb{SOCX}$ in its current state is already a useful tool. In each of the problem cases it is observed that the approximations generated for state and state delay variables are comparable to the analytic solutions. On successive iterations the state approximations converge without any issues. However, we observed a slightly different outcome when dealing with control variables. Although the $\mathbb{SOCX}$ control trajectories display a similar structure to the analytic solutions we see the presence of some inconsistencies in the approximations on successive iterations. Note that this is only observed in delayed control variables as undelayed controls display smooth trajectories (See Figure 3.4b, Figure 3.12b, and Figure 3.13b).

We describe this inconsistent behavior as chattering, which is defined as a succession of quick and inadequate computed points. This pattern was observed in solutions of each of the control delay optimal control problems featured in our test set as well. When this behavior was observed in Eq. (3.10) for the control delay problem it was first assumed that introduction of the delayed control variable into the normal integration interval caused problems for the optimizer (the spike was noticed at $t = 0.1$ in the control with the time delay control value $\tau = 0.1$). However, this theory was disproved after observing the chatter at a node different from the control delay time value in Eq. (3.11) for the CSTR problem. $\mathbb{SOCX}$'s reaction to the presence of control delays is not ideal, and required further investigation.

Since state delay problems do not encounter the oscillatory behavior in the approximations we began investigation with analyzing the differences in calculations of state variables versus control variables. Recall that control delays are interpolated piecewise linearly or quadratic and state delays are interpolated via cubic Hermite polynomials. We hypothesized that interpolation of state delays with a higher order polynomial may be the reason why chatter does not develop in the generated state delay solutions. To investigate this notion we conducted a study that compared and contrasted the results of computing the solutions of a general optimal control state delay problem with the trapezoid method with linear interpolation of state delay variables versus computation with the trapezoid method with Hermite interpolation.

For this investigation we define application of the trapezoid method with linear interpolation as the state standard trapezoid interpolant (STR) and application of the trapezoid method with Hermite interpolation as the state extended trapezoid interpolant (XTR). To compare STR and XTR we formulate a trapezoid integration code in Matlab with the stated interpolations. Although this approach is not the most accurate, it served as a starting place to investigate the effect of interpolants on approximations. Although the resulting NLP is different we predicted that STR and XTR will in principle give the same order of an integrator. The general state delay problem is of the form

$$\dot{x} = ax(t) + bx(t - \tau), \qquad t \in [0,\ t_f] \tag{3.12a}$$

$$x(t) = p, \qquad -1 \le t < 0 \tag{3.12b}$$

$$x(0) = m, \tag{3.12c}$$

where a and b are constant.

To ensure that we were covering all bases involving state delay interpolation formulations our study considered exploration of three cases involving computation of solutions on different grid structures. Grid structure can alter the formation of the resulting NLP, and is thus very important in the optimization process. Let h denote the stepsize of the solution grid then define

- Case 1: uniform grid with $h \mid \tau$,
- Case 2: uniform grid with $h \nmid \tau$, and
- Case 3: nonuniform grid.

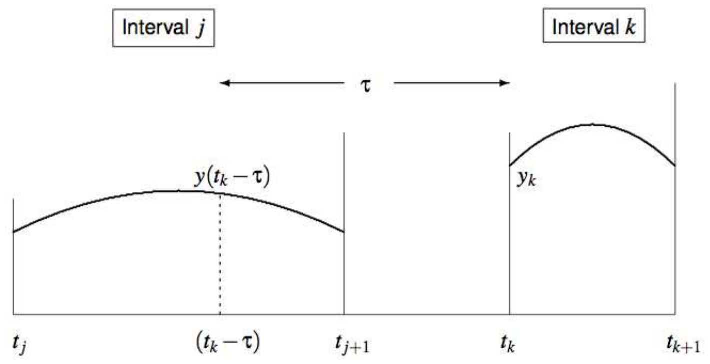
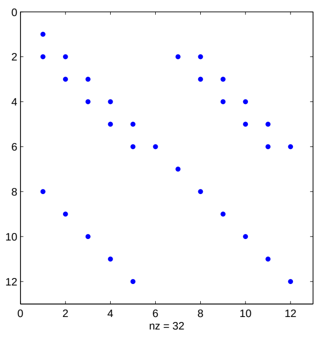


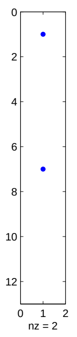
Figure 3.14: Graphical representation of the evaluation of the delayed state variable

In Figure 3.14 the interpolation intervals for a state delay variable are shown. In each uniform grid case $h \mid t_f$ so the nodes are equally spaced. If the stepsize divides the delay value τ the delay variable will always be interpolated at previous nodes i.e., $t_k - \tau \equiv t_k$ for all k. If the stepsize does not divide the delay value, some interpolated time values will be different from previous nodes. In the nonuniform grid case $h \nmid t_f$ i.e., the grid points are not evenly spaced. On nonuniform grids the delay discrete time values do not match any previous nodes i.e., $t_k - \tau \neq t_k$ for all k.

When developing discretization codes indexing can be your worst enemy. Because of this we first write the discretizations for Eq. (3.12) to help to provide a structural diagram of what each type of system to be solved looks like, such as sparsity pattern and matrix bandwidth, etc. Let x_k and xd_k represent the discrete state and state delay variables respectively. Defining $z = [x_0,\ x_1, \ldots, x_5,\ xd_1,\ xd_2, \ldots, xd_5]$ to be the merged discrete state vector allows us to represent the matrix equation related to the system of equations as $Az = B$. Solving for each discrete variable we have the sparsity patterns in Figures 3.15, 3.16, and 3.17 for the matrix A and right-hand side vector B. In each subfigure, rows 1 - 6 pertain to the state x_k equations and the remaining rows 7 - 12 describe the delayed state xd_k equations.



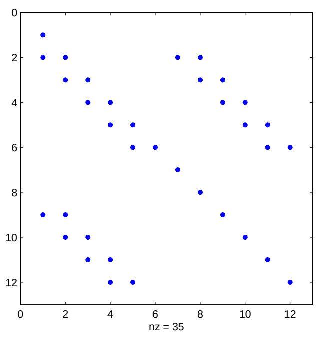
(a) Matrix A



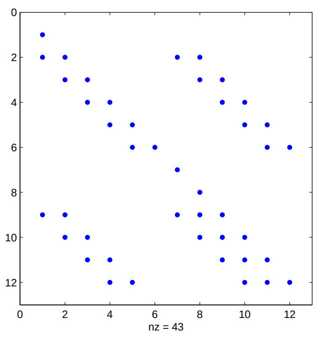
(b) Vector B

Figure 3.15: Case 1 for Eq. (3.12)

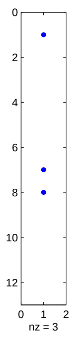
In Figure 3.15 only one matrix appears for Case 1 because STR and XTR yield the same discrete systems on uniform grids. In Figure 3.16 and Figure 3.17 note that the Hermite interpolated equations utilize more points than the linear interpolated equations. Finally, for Case 3 we see the effects of the nonuniform grid on both interpolations. For the selected nonuniform grid none of the state delay interpolated equations depend on state $x_2(t)$. This is displayed in Figure 3.17a and Figure 3.17b in rows 7 - 16 by the gap at column 2.



(a) Linear Matrix A

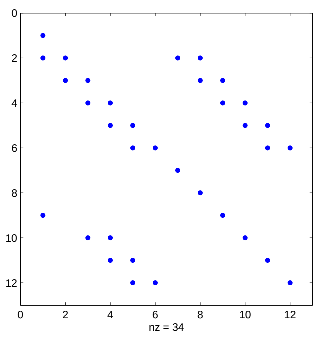


(b) Hermite Matrix A

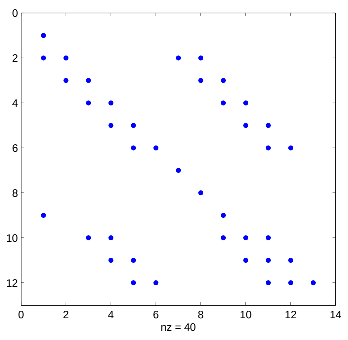


(c) Vector B

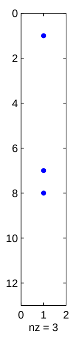
Figure 3.16: Case 2 for Eq. (3.12)



(a) Linear Matrix A



(b) Hermite Matrix A



(c) Vector B

Figure 3.17: Case 3 for Eq. (3.12)

After solving each of the resulting systems we assess interpolation performance by comparing the Matlab solutions to the solutions output by $\mathbb{SOCX}$. This step is necessary as errors can be relatively large although graphically the solutions look the same. Based on Table 3.6 the max errors display that results from using the trapezoid method with either linear interpolation or Hermite interpolation are comparable. Despite the different spy matrices the linearly interpolated solutions and the Hermite interpolated solutions yield similar results for the uniform

Table 3.6: Eq. (3.12) Max Error Order

Variable	STR	XTR
Case 1	1.0E-07	1.0E-07
Case 2	1.0E-07	1.0E-07
Case 3	1.0E-03	1.0E-04

cases. In Case 3 it is no surprise that the Hermite interpolant performs slightly better due to the fact that it considers midpoints between the nonuniform nodes.

The preceding tests were conducted to assess inaccuracies in $\mathbb{SOCX}$'s direct transcription process for the solutions of optimal control delay problems. Although this approach is not the most accurate, it serves as a starting place to investigate the effect of interpolants on approximations. Because state delay solutions are comparable when solved using linear interpolation we can conclude that the choice of interpolant is not a primary contributor to the oscillatory behavior observed in control delays. In this study we also discovered that grid choice plays a very important role in how the associated NLP system is structured, and hence how optimization is carried out. We further explore the occurrence of the oscillatory behavior seen in control delay approximations by investigating the impact of grid choice on the solutions of optimal control systems with control delays.

Chapter 4

Exogenous Input Control Method

4.1 Control Delay Optimization Problems on Nonuniform Grids

Current studies are concerned with optimal control problems that deal with both state time delays and control time delays. To approximate the solutions of optimal control delay problems $\mathbb{SOCX}$ utilizes the trapezoid and/or Hermite-Simpson numerical methods equipped with piecewise linear, piecewise quadratic, or Hermite interpolating polynomials. The trapezoid and Hermite-Simpson methods produce similar results. Despite the choice of mesh, when solving optimal control problems with state delays, $\mathbb{SOCX}$ has shown no issues. However, it has been discovered that grid choice plays a very important role in the output results of optimal control problems with control delays. The user can select the initial grid, but further grid refinement is done automatically by the algorithm. Suppose that a user chooses to solve a problem on an initial grid that is uniform (or uses the default). If execution does not terminate at the first iteration, at some point during grid refinement the new mesh will become nonuniform.

The mesh refinement algorithm tries to equally distribute the error over the approximation before termination. Consider a grid with M points. When assessing the approximation on a single iteration grid points will be added to the interval with maximum error. This step is repeated until a desired tolerance is reached or a grid point threshold is reached i.e., M_1 points have been added to a single interval or $M - 1$ points have been added on a single iteration. In the case that $M - 1$ points have been added we refer to this as a total grid refinement because the new grid has nearly doubled the size of the original grid. Thus, to avoid the occurrence of a nonuniform grid the user must solve the problem on a uniform grid and force termination at the first iteration.

For delayed control variables, an initial solve on a uniform grid yielded smooth approximations. Unfortunately, oscillations began to develop in the solutions of delayed control variables on successive iterations. After experimenting with various meshes we discovered that this chatter appeared when the mesh became nonuniform. This could be due to the fact that nonuniformity can alter the relationship of the control with its corresponding delayed quantity. Consider a control variable $u(t)$ defined on an interval $\mathcal{G}_u = [t_0, t_f]$, and a delayed control variable $u(t - \gamma)$ defined on $\tilde{\mathcal{G}}_u = [t_0 - \gamma, t_f - \gamma]$, where $\gamma > 0$. For certain problems, values for $u(t)$ may differ drastically from $u(t - \gamma)$ if t_f is not divisible by γ. This could also lead to having two sets of parameters for the same function, $u(t)$. When a grid refinement is issued and the new NLP is formulated, some variables may become free to the optimizer since a change in grid can alter consistency relationships and constraints. Note that a grid mismatch can occur between state and state delay variables as well. However, there is something special about state variables that prevents this very thing from happening.

4.1.1 Simple Mixed Delay Problem

To exercise the impact of nonuniform grids on control delay variables in $\mathbb{SOCX}$ we explore a simple mixed delay (SMD) example which is composed of the sum of a single state delay and a single control delay. Consider minimizing the objective function

$$J = \int_0^5 x^2 + u^2 \, dt, \tag{4.1a}$$

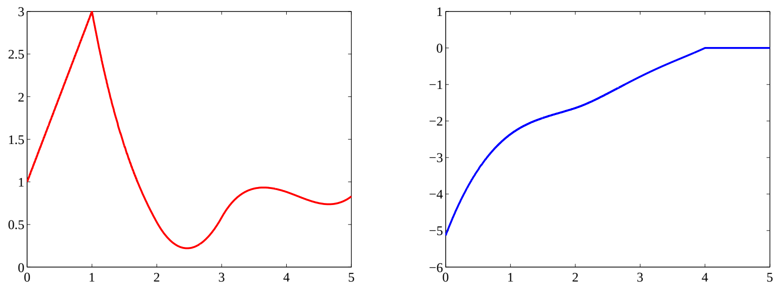
subject to

$$\dot{x} = x(t-2) + u(t-1), \qquad t \in [0,\ 5] \tag{4.1b}$$

$$x = 1, \qquad -2 \le t < 0 \tag{4.1c}$$

$$u = 1, \qquad -1 \le t < 0 \tag{4.1d}$$

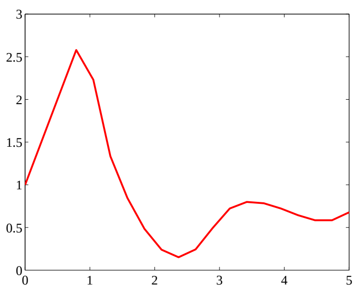
$$x(0) = 1. \tag{4.1e}$$



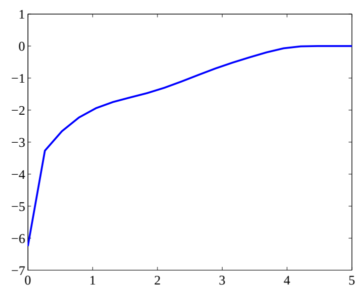
(a) State $x(t)$ (b) Control $u(t)$

Figure 4.1: MOS Solution for SMD in (4.1)

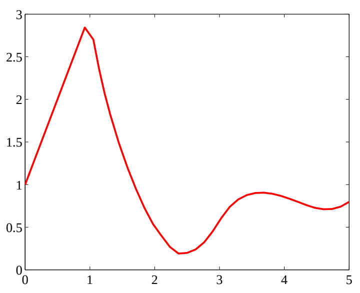
The method of steps solution for the number of steps $N = 5$ is plotted in Figure 4.1. To ensure that oscillatory behavior in the delayed control is consistent we present the $\mathbb{SOCX}$ solution output at several iterations in Figure 4.2. Looking at graphs at each iteration we see that the state variable is converging to the analytic solution by observing the trend of an increase from 1 to 3 on $[0,\ 1]$ followed by a quick decrease on $[1,\ 2.5)$. As for the control variables observe the introduction of chatter at iteration 2, and how it worsens at each successive iteration. We investigate the cause of this behavior by analyzing interpolation of state and control delay variables on a specified set of small grids.



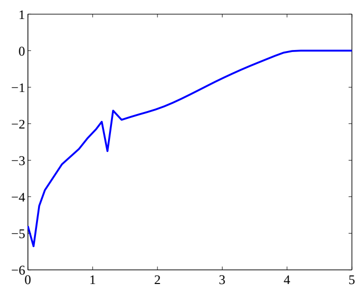
(a) Iteration 1: State $x(t)$



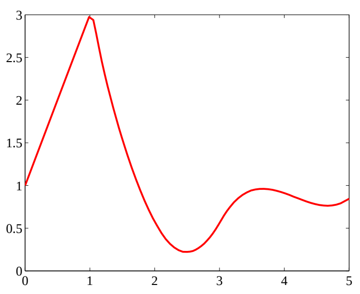
(b) Iteration 1: Control $u(t)$



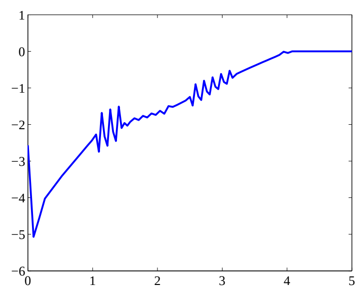
(c) Iteration 2: State $x(t)$



(d) Iteration 2: Control $u(t)$



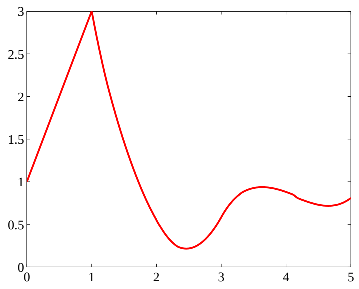
(e) Iteration 3: State $x(t)$



(f) Iteration 3: Control $u(t)$

Figure 4.2: $\mathbb{SOCX}$ solutions for SMD in Eq. (4.1)

Suppose that Eq. (4.1) is solved with the trapezoid discretization on an initial grid of 5 points and $\mathbb{SOCX}$ automatically refines the initial grid as specified in Table 4.1 also using the trapezoid method as the integrator. Since there are two delayed variables present in the SMD problem there are two time delay grids associated with interpolation. These associated


(g) Iteration 6: State $x(t)$

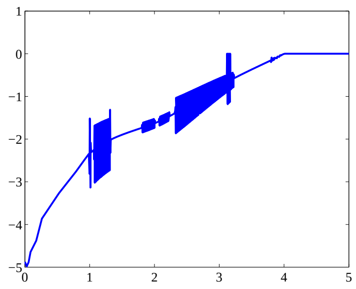

(h) Iteration 6: Control $u(t)$

Figure 4.2: $\mathbb{SOCX}$ solutions for SMD in Eq. (4.1) continued

Table 4.1: Sample grid refinement for (4.1)

Iteration	Grid
1	[0, 1, 2, 3, 4, 5]
2	[0, 1, 1.1, 1.2, 1.3, 2, 2.7, 3, 3.2, 3.5, 4, 4.7, 5]
3	[0, 1, 1.1, 1.2, 1.3, 2, 2.7, 2.8, 2.9, 3, 3.2, 3.5, 4, 4.7, 5]

Table 4.2: Delay time values applied to Grid 2 mesh

Grid	Values
Original	[0, 1, 1.1, 1.2, 1.3, 2, 2.7, 3, 3.2, 3.5, 4, 4.7, 5]
Delayed State	[−2, −1, −0.9, −0.8, −0.7, 0, 0.7, 1.0, 1.2, 1.5, 2, 2.7, 3]
Delayed Control	[−1, 0, 0.1, 0.2, 0.3, 1.0, 1.7, 2.0, 2.2, 2.5, 3.0, 3.7, 4.0]

delay time values are featured in Table 4.2. Recall that before $\mathbb{SOCX}$ solves an optimal control problem with delays it first converts it to a DDAE by employing a consistency relationship between delay variables and pseudo variables. The resulting DDAE for Eq. (4.1) is

$$J = \int_0^5 x^2 + u^2 \, dt, \tag{4.2a}$$

subject to

$$\dot{x} = w(t) + v(t), \qquad t \in [0,\ 5] \tag{4.2b}$$

$$0 = w(t) - x(t-2), \tag{4.2c}$$

$$0 = v(t) - u(t-1), \tag{4.2d}$$

with startup functions and the initial condition defined

$$x = 1, \qquad -2 \leq t < 0 \tag{4.2e}$$

$$u = 1, \qquad -1 \leq t < 0 \tag{4.2f}$$

$$x(0) = 1. \tag{4.2g}$$

Let $\langle x_k \ \ x_{k+1} \ \ f_k \ \ f_{k+1} \rangle$ denote the Hermite interpolation for the state delay variable and $\langle u_k \ \ u_{k+1} \rangle$ denote the linear interpolation for the delayed control variable, then discretizing the $\mathbb{SOCX}$ equations with the trapezoid method using Table 4.2 we write down the full set of equations detailing the NLP to be solved with respect to the second iteration in Eq. (4.3).

$$x_1 = 1, \tag{4.3a}$$

$$x_2 = x_1 + \frac{h_1}{2}(w_1 + v_1 + w_2 + u_1), \tag{4.3b}$$

$$x_3 = x_1 + \frac{h_2}{2}(w_2 + \langle u_1 \ u_2 \rangle + w_3 + \langle u_1 \ u_2 \rangle), \tag{4.3c}$$

$$x_4 = x_1 + \frac{h_3}{2}(w_3 + \langle u_1 \ u_2 \rangle + w_4 + \langle u_1 \ u_2 \rangle), \tag{4.3d}$$

$$x_5 = x_1 + \frac{h_4}{2}(w_4 + \langle u_1 \ u_2 \rangle + w_5 + \langle u_1 \ u_2 \rangle), \tag{4.3e}$$

$$x_6 = x_1 + \frac{h_5}{2}(w_5 + \langle u_1 \ u_2 \rangle + x_1 + u_2), \tag{4.3f}$$

$$x_7 = x_1 + \frac{h_6}{2}(x_1 + u_2 + \langle x_1 \ x_2 \ f_1 \ f_2 \rangle + \langle u_5 \ u_6 \rangle), \tag{4.3g}$$

$$x_8 = x_1 + \frac{h_7}{2}(\langle x_1\ x_2\ f_1\ f_2\rangle + \langle u_5\ u_6\rangle + x_2 + u_6), \tag{4.3h}$$

$$x_9 = x_1 + \frac{h_8}{2}(x_2 + u_6 + \langle x_4\ x_5\ f_4\ f_5\rangle + \langle u_6\ u_7\rangle), \tag{4.3i}$$

$$x_{10} = x_1 + \frac{h_9}{2}(\langle x_4\ x_5\ f_4\ f_5\rangle + \langle u_6\ u_7\rangle + \langle x_5\ x_6\ f_5\ f_6\rangle + \langle u_6\ u_7\rangle), \tag{4.3j}$$

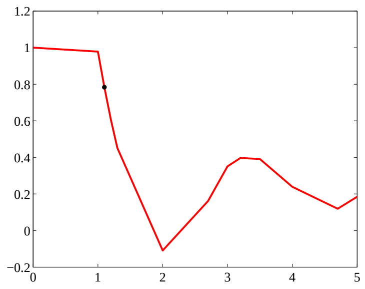
$$x_{11} = x_1 + \frac{h_{10}}{2}(\langle x_5\ x_6\ f_5\ f_6\rangle + \langle u_6\ u_7\rangle + x_6 + u_8), \tag{4.3k}$$

$$x_{12} = x_1 + \frac{h_{11}}{2}(x_6 + u_8 + \langle x_7\ x_8\ f_7\ f_8\rangle + \langle u_{10}\ u_{11}\rangle), \tag{4.3l}$$

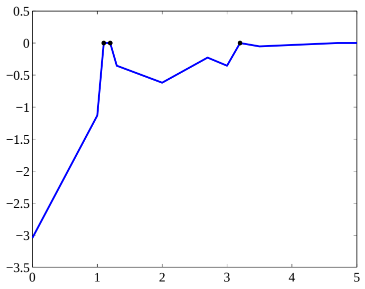
$$x_{13} = x_1 + \frac{h_{12}}{2}(\langle x_7\ x_8\ f_7\ f_8\rangle + \langle u_{10}\ u_{11}\rangle + x_8 + u_{11}). \tag{4.3m}$$

The purpose of listing Eq. (4.3) is to observe which state and control variables take part in interpolation and/or optimization for Eq. (4.2). Recall that the state delay value is $\tau_1 = 2$ and the control delay value is $\tau_2 = 1$. Note that the state variables that can be involved in interpolation are states $x_1, \ldots, x_8$ since state variables $x_9, \ldots, x_{13}$ refer to time values on the interval where the state delay is inactive $[t_f - \tau_1,\ t_f] = [3,\ 5]$. Similarly, valid control variables are $u_1, \ldots, u_{10}$ since the control delay disappears on $[t_f - \tau_2,\ t_f] = [4,\ 5]$. Now while state x_3 does not take part in interpolation of the delay state variables, due to the discretization of the state equation Eq. (4.2b) it appears as a left-hand side variable in Eq. (4.3). Hence, all states appear in the resulting NLP equations. However, the same does not apply for controls. Controls u_3, u_4, u_9, and u_{11} are not utilized during interpolation and do not appear in the system of equations at all. During optimization these unused controls are free variables and will be minimized according to the quadratic objective function.

We now turn attention to the $\mathbb{SOCX}$ results for Grid 2 featured in Figure 4.3. Grid 2 is a coarse grid, so solutions displayed may slightly differ from the analytic solution. The values for the unused variables have been indicated by a black dot. Previously, we mentioned that state $x_3 = x(1.1)$ was neglected. Looking at Figure 4.3a we cannot really see any drastic inconsistencies in the trajectory in comparison to Figure 4.1a. The value computed at this value flows with the trajectory of the analytic solution. However, in Figure 4.3b observe that $u_3 = u(1.1)$, $u_4 = u(1.2)$, and $u_9 = u(3.2)$ are all zero and are out of range in comparison to the MOS control



(a) Grid 2 State

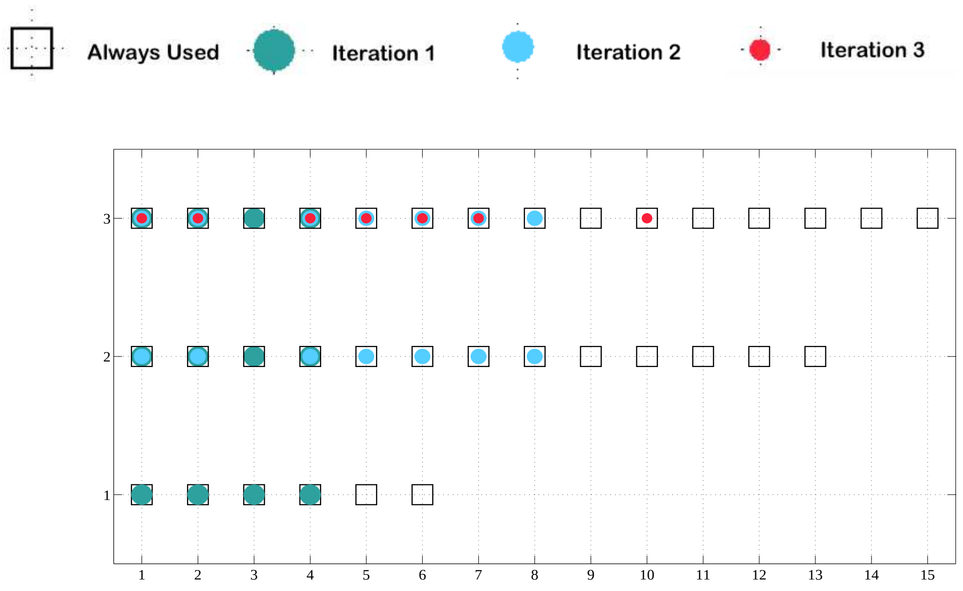


(b) Grid 2 Control

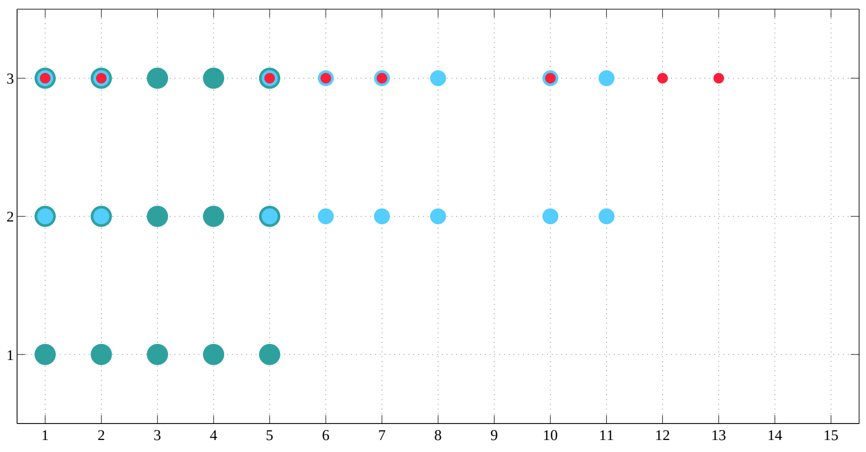
Figure 4.3: $\mathbb{SOCX}$ results for Eq. (4.1) solved on Grid 2

in Figure 4.1b. This is evidence that the optimizer may be minimizing these free controls. If those three points are removed from the graph, we get an increasing trajectory more similar to the true control. Similar conclusions can be drawn from the grid point interpolation assessment of Grid 3 and the plotted results.

An extended breakdown of the states and controls utilized at each iteration of our "constructed" mesh refinement is featured in Figure 4.4. The y-axis denotes the iteration number and the x-axis denotes the grid point number. Note that the mesh refinement algorithm in $\mathbb{SOCX}$ requires that the new mesh contains all points from the previous iteration. This property is captured by the respective colored symbol appearing appropriately on successive iterations. The first iteration displays the six uniform grid points as large jade green dots. The new points added on successive iterations are characterized by medium-sized sky blue circles at the second iteration and small red circles at the third iteration. For iteration 3 no red circles appear at u_3, u_4, and u_9 in Figure 4.4a. These are the free controls that will be minimized to zero at the third iteration. Similarly, no red circles appear at x_3 and x_8 in Figure 4.4a. However, note that squares at these locations do indicate a presence of these variables in the equation dynamics.



(a) State grid points utilized



(a) Control grid points utilized

Figure 4.4: Grid points utilized during interpolation on Grid 2 for Eq. (4.1)

Recall that (4.1) features a cost to be optimized, a state delay, and a control delay. State delays are calculated from Hermite interpolants formed from integrated state variables that are optimized. Control delays are calculated from linear interpolants formed from control variables that are optimized. It is assumed that chatter develops in control delay variables because some discrete control variables are neglected or not used during the interpolation process. Hence if the dynamics solely feature delayed control variables, then these neglected controls will never appear in the discretized equations which causes them to become free to the optimizer. Because minimization is requested, the optimizer is going to make these neglected controls as small as possible with respect to the error tolerance on the relative interval.

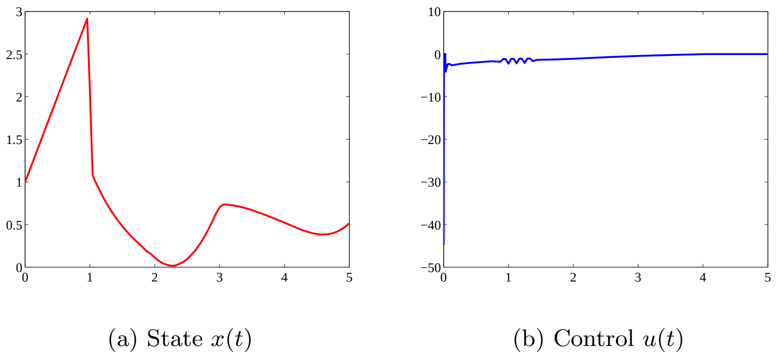


Figure 4.5: $\mathbb{SOCX}$ solution with chattery control on a fine nonuniform grid for (4.1)

So why does chatter not develop in the approximations of state delay variables? First note that the choice of interpolant (linear or Hermite) does not drastically effect the state delay solutions. Hence, it is believed that chatter does not develop in state delays because of integration. Integration of state variables forces each state to be accounted for despite possible neglect during interpolation. Thus, the optimizer does not have much control over choosing what the state should be because of the presence of defect constraints. In the problems that we have solved it appeared that the state approximations were not impacted by the presence of chatter in the control. However if large enough, chattering inefficiencies in controls can alter state solutions if they appear in the dynamics. Observe this fact in Figure 4.5 where the SMD problem was solved on a very fine nonuniform grid. Note the steepness in the state solution near $t = 1$. This is also the nodal point in which interpolation begins for the delayed control.

4.1.2 Nonuniform Grids are Necessary

In this section we have shown that on nonuniform grids 𝕊𝕆ℂ𝕏 approximations for control delays unexpectedly display some oscillatory behavior. Although nonuniform grids pose unfavorable challenges, they are necessary for some physical processes and are essential components of highly reliable adaptive methods because they help to improve overall accuracy. In some situations solely relying on uniform grids can increase computation requirements, or make such a solution impossible. Higher accuracy is obtained for fewer grid points on nonuniform grids [93]. Mesh refinement algorithms employed normally detect areas to be refined based on discretization errors and high gradient estimations.

Introducing nonuniformity in these regions can substantially increase the accuracy and improve the convergence rate of the method used. Consequently, resolving the issues that nonuniform grids introduce in the control delay optimal control systems is necessary to ensure that 𝕊𝕆ℂ𝕏 is a robust and useful tool. In [21] we address the presence of chatter in control delay variables by exploring cost functional design. Note that this is applicable since the form of the objective function is partly the users choice. The results for the problem reported displayed some promise. However, modifying the costs of more complex problems revealed inconsistent results. Hence, developing a better strategy was needed.

4.2 Exogenous Input Control Method

The *exogenous input control* (EIC) method is a technique that applies a regularization to delayed control variables to numerically compute the solutions to optimal control delay problems. The method was developed to handle the issues that nonuniform grids introduce into the solutions of delayed control variables computed in 𝕊𝕆ℂ𝕏. Although the EIC method was created in regards to $\mathbb{SOCX}$, it may be universally applicable to other solvers in similar situations where chattering results from interpolation on nonuniform grids.

The EIC method consists of replacing control delay variables with state delay variables, and inputting a corresponding weighted control to help capture the behavior of the delayed control variable of the original system. This weighted control is termed exogenous because it is not part of the original system, and is an auxiliary variable which drives the delayed control. In applications, the control is often generated by another dynamical system omitted from the model. With the EIC method we are putting the generator back in. The control to state substitutions add new state dynamic equations to the system. The new state replacing the delayed control variable is often referred to as the regularized control variable.

In non-delay optimal control applications similar implementations of constrained variables have been taken into account. In [35, 36] a similar technique is used to model an aircraft wind-shear application featured in [99] with a state variable and two additional constraints. In aircraft applications the angle of attack[1] is normally a control variable. Because of the need to continuously model its position for safety, the angle of attack is expressed as a state variable. After applying the transformation technique, the derivative of the angle of attack depends on auxiliary controls and constraints arise.

Recall that on a sufficiently fine uniform grid $\mathbb{SOCX}$ is able to provide a pretty good estimate of the solution to optimal control problems with constant control delays (i.e., no chatter in the control delay solutions) just after one iteration. This result is consistent with the results from employing the integrality condition proposed by Göllmann et al. in [65, 66]. For a delayed control $u(t-s)$ defined on $[a,\ b]$, so long as $s,\ b-a \in h\mathbb{N}$ the uniform stepsize $h > 0$ can be used to match the delay s. Depending on the size of the grid the solution output may not be optimal since the desired tolerance may not have been achieved. Whether the output solution is optimal or not, the slopes of the control delay solutions can be helpful in approximating the solution on nonuniform grids. The EIC procedure begins with solving the control delay optimal

[1]The difference between where the wing is pointed and the direction of the air flowing over the wing is the **angle of attack**.

control problem on a uniform grid in an effort to estimate the maximum and minimum slopes of the delayed control variables. The full step-by-step procedure for applying the EIC method is outlined in Algorithm 1.

Algorithm 1 EIC Method

```
1. Solve the OCDP on a uniform grid
2. For every control delay variable $u_i(t)$
    2a. Determine the minimum slope $m_i$ and the maximum slope $M_i$
    2b. Create a state variable $x_{n+i}(t)$
    2c. Create an exogenous variable $z_i$
    2d. Add a state equation $\dot{x}_{n+i} = z_i$
    2e. Add a constraint $m_i \leq z_i \leq M_i$ (if necessary)
    2f. Add the value $\varepsilon z_i^2$ to the objective function
    2g. Replace $u_i$ with $x_i$
3. Solve the new OCDP for small $\varepsilon$ and assess solutions
4. If necessary, modify 2a and resolve
```

The EIC method transforms the original OCDP with control delays into a new OCDP with state delays and constraints. We often refer to the new OCDP as the EIC formulated problem or regularized problem. Here the word regularized is used to express that the new problem is formulated to be normal or more predictable.

In the EIC algorithm, n denotes the dimension of the optimal control delay system and ε denotes the weight of the exogenous control. Ideally, $0 \leq \varepsilon \lll 1$ so that the objective function of the original OCDP and the new problem are nearly the same. Note that for each new state equation generated in **2d.** the initial condition, $x_{n+i}(0)$ is unspecified. In some cases an improvement in accuracy of the new state variable solutions can be achieved by specifying initial conditions that are close to the initial value of the corresponding original delayed control variables, $u_i(0)$. However, even a good estimate of $u_i(0)$ can lead to more issues if the grid becomes too fine. The free initial condition proved to be the more suitable choice since it not only produces a sufficient approximation, but also allows the full solution to reflect the work of the optimizer and the perturbation parameter.

The EIC method can produce a good approximation when the exogenous control bounds are not given. Here, it is important to note that **2e.** is optional, and is implemented for numerical purposes to tame the side effects of the optimizer. For small ε, the performance index becomes almost independent of the exogenous control which may result in a poor choice of optimal values by the optimizer. A poor choice of values for the exogenous control can lead to a poor choice of values for the regularized control. Hence when ε is near zero it may be necessary to impose **2e.** to force the behavior of the new state, $x_{n+i}(t)$ to reflect that of the original control, $u_i(t)$. When solving an EIC formulated system with an adequate ε value or with any analytic method the exogenous control bounds are not required.

4.2.1 Control Delay Test Problems Resolved with the EIC Method

We now display the effects of using the EIC method with $\mathbb{SOCX}$ by resolving the control delay and CSTR problems featured in Section 3.2.2 and the simple mixed delay problem featured in Section 4.1.1. Execution was carried out in a similar fashion with each of the problems being solved on an initial uniform grid of 21 points with the default switch from the trapezoid to the Hermite-Simpson discretization occurring at the second iteration. The method of steps solution is still assumed to be the true solution and is used for comparison purposes. Again, the newly created state approximating the control will sometimes be referred to as the regularized control.

4.2.1.1 EIC Control Delay Problem (ECDP)

The control delay problem is featured in Eq. (3.3). Referring back to Figure 3.7b and Figure 3.8a we recall the chatter present on the interval $[0.1,\ 0.105]$. The method of steps solution is featured in Figure 3.6. For the single control delay variable we determine that $u(t)$ approximately has a min slope, $m = 0$ and a max slope, $M = 5$. The goal of the regularized system is to minimize

$$J = \frac{1}{2}\int_0^{0.25} x_1^2(t) + x_2^2(t) + \varepsilon z^2(t)\, dt, \tag{4.4a}$$

subject to

$$\dot{x}_1(t) = x_1(t) + x_2(t-0.1) + x_2(t), \tag{4.4b}$$

$$\dot{x}_2(t) = z(t), \tag{4.4c}$$

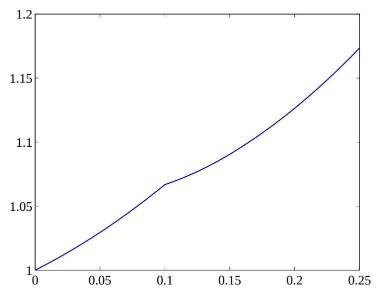
$$x(0) = [1,\ a]^T, \tag{4.4d}$$

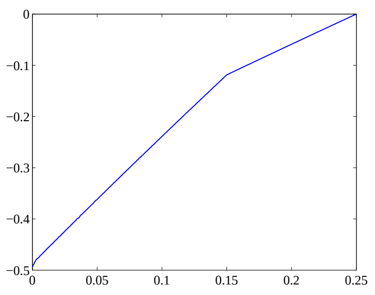
with the initial condition and exogenous control bound defined

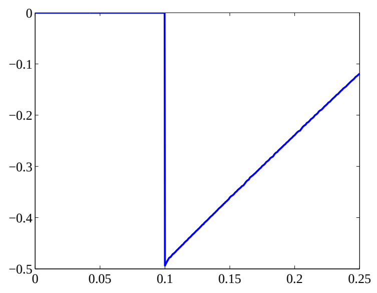
$$x_2(t) = 0, \qquad -0.1 \le t \le 0 \tag{4.4e}$$

$$0 \le z \le 5. \tag{4.4f}$$

In the EIC formulation state $x_2(t)$ has replaced the original control $u(t)$. As displayed in Eq. (4.4d) the initial condition for $x_2(t)$ is unspecified or free to the optimizer. In the above formulation the weight for the exogenous control $z(t)$ is not explicitly stated. Instead, we initially set ε=1.0E-01 and solve the associated system until the absolute max error for the regularized control is less than or equal to that of the absolute max error determined for the delayed control variable in the original problem. For instance, let u_{exact} be the true solution for the delayed control. This solution can be computed analytically, via MOS, or using $\mathbb{SOCX}$ on a sufficiently fine uniform grid, or any other method of your choosing. The idea hear is that the true solution is "good enough". As for the EIC method after a solve with a specified ε, if $\|u_{exact}(t) - x_2(t)\|_\infty \le \|u_{exact}(t) - u(t)\|_\infty$ the problem is considered to be successfully solved and the value for the weight is accepted. If not, the solve is considered unsuccessful and the value for the weight is reduced by a factor of 10 and the process is repeated.


(a) $x_1(t)$


(b) $x_2(t)$


(c) $x_2(t-0.1)$

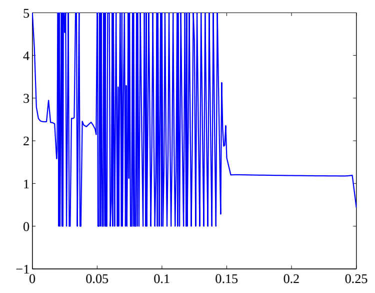

(d) $z(t)$

Figure 4.6: Final iteration for ECDP in Eq. (4.4), $\varepsilon = 1.0\text{E-}06$

The results for solving Eq. (4.4) with $\varepsilon = 1.0\text{E-}06$ are plotted in Figure 4.6. In comparing the solutions to Figure 3.6 we see that the regularized solutions output are comparable to the MOS solutions. More interesting, in Figure 4.6b observe that the approximation for $x_2(t)$ does not contain the notable oscillations observed in the original control $u(t)$. The closeup in Figure 4.7 shows a reduction of chatter and a smoother approximation for the regularized control on the region [0.1, 0.105].

In Figure 4.6d the exogenous control approximation is plotted, and it is not pretty! This variable is not a component of the original control delay problem, and is therefore not compared with any MOS solution. Observe that the control is banging back and forth between the min and max slopes computed until the time reaches 0.15. The delayed state variable, $x_2(t-0.1)$ is only active for $t \in [0,\ 0.15]$. Consequently on [0.15, 0.25], the exogenous control is equal to zero because here the state delay variable vanishes. At this time we are not concerned with the

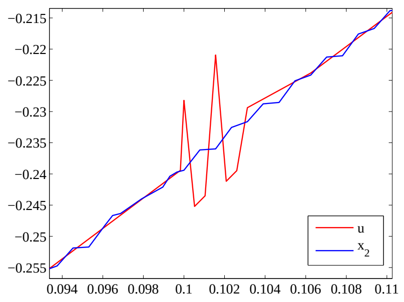


Figure 4.7: Regularized control in Eq. (4.4) for $\varepsilon = 1.0\text{E-}06$ vs. original control in Eq. (3.3)

bang-bang appearance of the exogenous control solution since it does not represent any realistic model characteristics for this problem. This is the expected behavior for a bounded control with a small weighted coefficient in the cost.

Table 4.3: Max errors for ECDP in Eq. (4.4) computed at ε

Variable	1.0E-01	1.0E-02	1.0E-04	1.0E-06	1.0E-08	0
$x_1(t)$	2.62E-02	1.7E-02	4.35E-04	4.7E-04	4.63E-04	4.63E-04
$x_2(t)$	2.66E-01	1.95E-01	2.41E-02	5.7E-03	6.3E-03	6.3E-03

Furthermore, in Table 4.3 the max errors for the solution output at the final iteration are computed for various weights ε. Primarily, we are only concerned with the errors related to the newly added state variable $x_2(t)$. However, we check the errors for both variables to determine if accuracy was reduced. For the original control delay problem the control approximation yielded a max error of 1.21E-02 and the state yielded a max error of 4.85E-04. In Table 4.3 note that the max errors computed for both the state and the regularized control for $\varepsilon \leq 1.0\text{E-}06$ are less than the error computed for the original variables. Hence, application of the EIC method has physically and numerically improved the solutions for the control delay test problem. Additional information about the optimal control analysis summary for the regularized problem for $\varepsilon = 1.0\text{E-}06$ is featured in Table 4.4.

Table 4.4: Optimal control summary for ECDP in Eq. (4.4)

EIC $\mathbb{SOCX}$ Solutions for ε = 1.0E-06

$\mathcal{G}_k$	N	F_{calls}	F_{evals}	ϵ	Time
1	21	4	25	1.9634E-04	7.0010E-03
2	21	3	98	7.8901E-05	9.3400E-03
3	41	3	98	6.9848E-06	1.9478E-02
4	76	3	98	3.4361E-06	6.2513E-02
5	127	3	98	2.4153E-06	1.6863E-01
6	157	3	98	1.3663E-06	2.6532E-01
7	175	3	98	6.1740E-07	3.3402E-01
8	211	3	98	3.4020E-07	5.4470E-01
9	226	3	98	9.6803E-08	6.1909E-01
Total	226	28	809		2.0301E+00

4.2.1.2 EIC Simple Mixed Delay (ESMD) Problem

The simple mixed delay problem is featured in Eq. (4.1). In the control delay solution output in Figure 4.2, spikes in the approximation were first observed at the second iteration. On successive iterations chatter was observed on multiple intervals. This example displayed how miscalculated points can have an impact on the approximation, and create more problems on later parts of the solution interval. The method of steps solution is featured in Figure 4.1. To apply the EIC method we first determine that the min and max slopes for the original control delay variable are $m = 0$ and $M = 4.5$. The goal of the regularized problem is to then minimize

$$J = \frac{1}{2}\int_0^5 x_1^2(t) + x_2^2(t) + \varepsilon z^2(t)\, dt, \tag{4.5a}$$

subject to

$$\dot{x}_1(t) = x_1(t-2) + x_2(t-1), \tag{4.5b}$$

$$\dot{x}_2(t) = z(t), \tag{4.5c}$$

with the following startup functions and initial condition

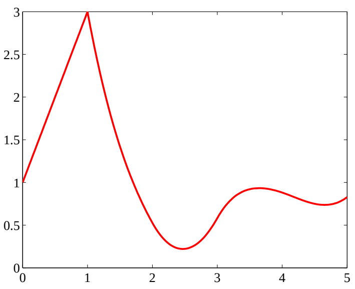
$$x_1(t) = 1, \qquad -2 \leq t < 0 \tag{4.5d}$$

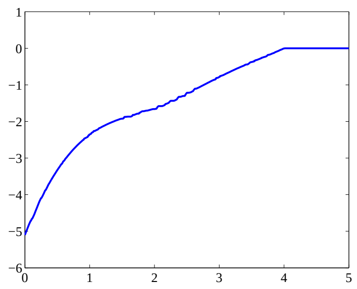
$$x_2(t) = 1, \qquad -1 \leq t < 0 \tag{4.5e}$$

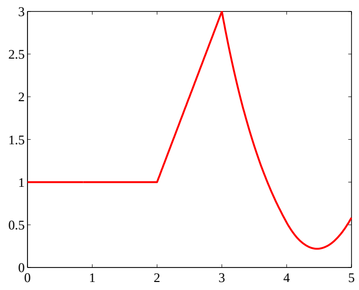
$$x(0) = [1,\ a]^T, \tag{4.5f}$$

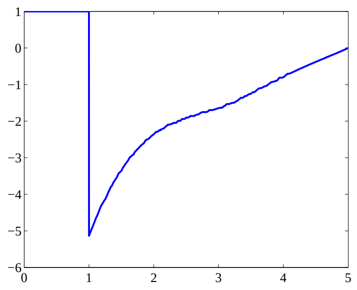
and exogenous control bound defined

$$0 \leq z \leq 4.5. \tag{4.5g}$$


(a) $x_1(t)$


(b) $x_2(t)$


(c) $x_1(t-2)$


(d) $x_2(t-1)$

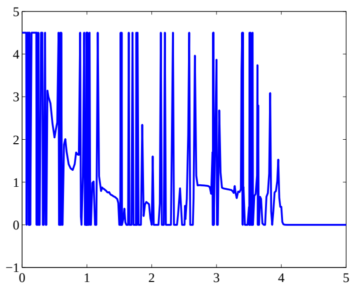
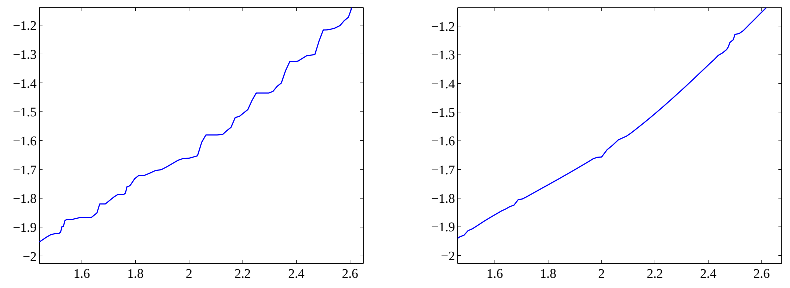

(e) $z(t)$

Figure 4.8: Final iteration for ESMD problem in Eq. (4.5), ε =1.0E-04

We solve Eq. (4.5) and plot the final iteration for ε = 1.0E-04 in Figure 4.8. The problem terminated in 11 iterations on a final grid of 332 points in 12.3s. Observe that the approximation for new state $x_2(t)$ in Figure 4.8b is significantly smoother than the approximation for the original delayed control plotted in Figure 4.2h. Beforehand, the max absolute error for the control solution was 8.41E-01. After application of EIC with ε = 1.0E-04 the max error for the regularized control is 3.4E-02.

Although there is an improvement in error, we observe small "craters" in the solution for the regularized control. A closeup of this behavior is displayed in Figure 4.9 for ε = 1.0E-04 and $\varepsilon = 0$. Here we point out that as ε goes to zero these craters begin to flatten out. In Figures 4.8b and 4.8e observe the zero values computed on $[4,\ 5]$ due to the regularized control being inactive on the interval. Additionally, notice that the solution for state $x_1(t)$ in Figure 4.8a appears unchanged in comparison to the original state. However, the max absolute error has been reduced from 4.32E-02 to 1.5E-03. Maximum errors computed at other values of ε can be viewed in Table 4.5.



(a) ε = 1.0E-04 (b) $\varepsilon = 0$

Figure 4.9: Closeup of the regularized control for ESMD problem in Eq. (4.5)

Table 4.5: Max errors for ESMD problem in Eq. (4.5) computed at ε

Variable	1.0E-01	1.0E-02	1.0E-04	1.0E-08	0
$x_1(t)$	1.4E-01	2.6E-02	1.8E-03	1.7E-03	1.6E-03
$x_2(t)$	9.5E-01	3.8E-01	3.5E-02	3.09E-02	3.08E-02

4.2.1.3 EIC CSTR Mixed Delay Problem (ECSTR)

The CSTR problem was solved in Section 3.2.2.3, and yielded the approximations featured in Figures 3.11, 3.12, and 3.13. For the delayed control variable $u_2(t)$ we saw chatter on the interval [0.1, 0.15]. The MOS solution is plotted in Figure 3.10. Applying the EIC method to Eq. (3.2.2.3) we first estimate a min slope of $m = -6$ and a max slope of $M = 2$ for $u_2(t)$. Let $\bar{x}(t) = [x_1,\ x_2,\ x_3]$ and $\bar{u} = [u_1,\ x_4]$ then the goal for the regularized problem is to minimize the objective function

$$\tilde{F} = \int_0^{0.2} \|\bar{x}\|_2^2(t) + 0.01\|\bar{u}\|_2^2(t) + \varepsilon z^2(t)\, dt, \tag{4.6a}$$

subject to

$$\dot{x}_1 = -x_1(t) + R(t,x), \tag{4.6b}$$

$$\dot{x}_2 = -x_2(t) + 0.9x_4(t-s) + 0.1x_4(t), \tag{4.6c}$$

$$\dot{x}_3 = -2x_3(t) + 0.25R(t,x) - 1.05u_1x_3(t-r), \tag{4.6d}$$

$$\dot{x}_4 = z(t), \tag{4.6e}$$

with prehistory functions

$$x_3(t) = -0.02, \qquad -0.015 \le t < 0 \tag{4.6f}$$

$$x_4(t) = 1, \qquad -0.02 \le t < 0 \tag{4.6g}$$

and initial condition

$$x(0) = [0.49, -0.0002, -0.02, a]^T, \tag{4.6h}$$

and constraints

$$|u_1(t)| \leq 500, \tag{4.6i}$$

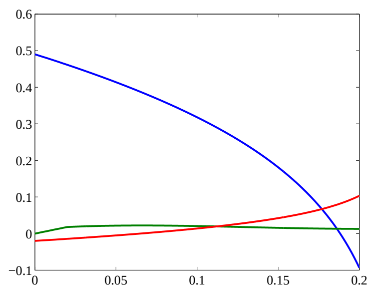
$$-6 \leq z(t) \leq 2, \tag{4.6j}$$

with the function

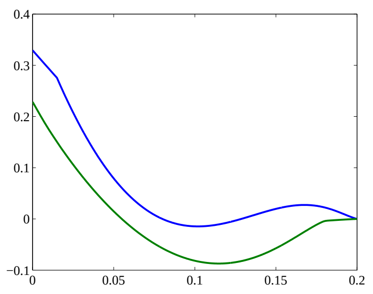
$$R(t, x) = (1 + x_1(t))(1 + x_2(t)) \exp\left(\frac{25x_3(t)}{1 + x_3(t)}\right). \tag{4.6k}$$

The solution for the CSTR regularized problem with $\varepsilon = 1.0\text{E-}08$ is plotted in Figure 4.10. Execution terminated in 67s in 12 iterations with a final grid of 1281 points. Graphically the state solutions, $\bar{x}(t)$ and the control solution $u_1(t)$ remain unchanged. In Figure 4.10e $u_1(t)$ is plotted as a blue line and the regularized control $x_4(t)$ is plotted in green. Observe that the chattering behavior has been removed from $x_4(t)$ on the interval $[0,\ 0.15]$. The solution now converges in a manner similar to the MOS solution. Take a look at the exogenous control plotted in Figure 4.10e. It is discontinuous at $t = s$ and $t = T - s = 0.18$. Recall that the exogenous controls in Figures 4.6d and 4.8e in the previous examples banged back and forth between the minimum and maximum slopes defined for most of the solution interval. However, the CSTR exogenous control only displays oscillatory behavior on $[0.1,\ 0.15]$. Recall that $\mathbb{SOCX}$ displayed issues on this interval for the original delayed control in Figures 3.12b and 3.13b as well. At approximately $t = 0.13$ the derivative of the delayed control $u_2(t)$ is zero, and thus $u(0.13)$ is a local minimum. Consequently, near this point $\mathbb{SOCX}$ introduces points in this region in an effort to minimize the error to achieve this minimum.

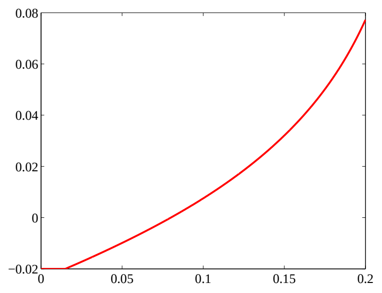
The absolute max error for the regularized control is 9.4E-03 which is an improvement in accuracy from the original delayed control which yielded an error 1.7E-02. Errors for other variables at varying ε can be viewed below in Table 4.6. A run for $\varepsilon = 0$ could not be obtained due to system timeout errors.



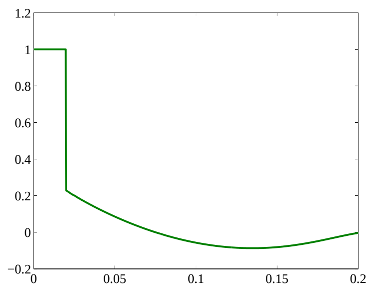
(a) $\bar{x}(t) = [x_1, \ , x_2, \ , x_3]$



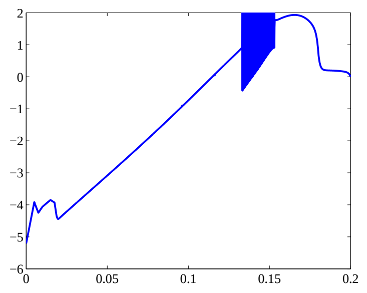
(b) $\bar{u}(t) = [u_1, \ x_4]$



(c) $x_3(t - 0.15)$



(d) $x_2(t - 0.02)$



(e) $z(t)$

Figure 4.10: Final iteration for ECSTR problem in Eq. (4.6), ε = 1.0E-08

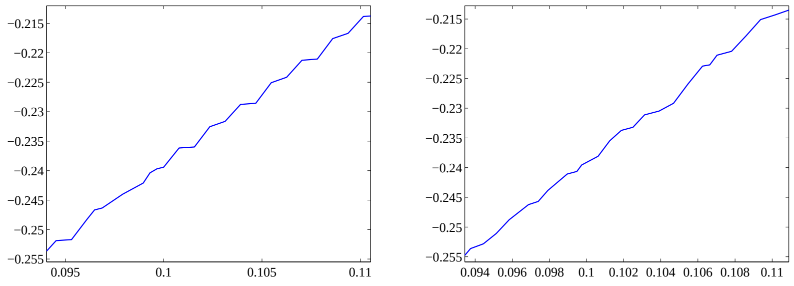
Table 4.6: Max errors for CSTR in Eq. (4.6) computed at ε

Variable	1.0E-01	1.0E-02	1.0E-04	1.0E-08
$x_1(t)$	2.72E-02	2.66E-02	9.3E-03	5.0E-04
$x_2(t)$	3.52E-02	3.44E-02	1.07E-02	1.0E-04
$x_3(t)$	6.8E-03	6.7E-03	2.5E-03	2.0E-04
$u_1(t)$	3.3E-02	3.23E-02	1.09E-02	5.0E-04
$u_2(t)$	2.87E-01	2.81.0E-01	1.05E-01	9.4E-03

4.2.2 Results Summary

Use of the exogenous input control method for the solutions of optimal control delay problems appears to be effective in solving the issues with chattering in delayed control variables on nonuniform grids, as well as showing improvement in the accuracy of the output approximations. In this section the EIC method was applied to the control delay, simple mixed delay, and mixed delay CSTR problems. Each of the regularized problems was solved on an initial uniform grid. Recall that nonuniform grids are introduced on subsequent iterations when $\mathbb{SOCX}$ refines the grid to reduce equation error. In each of the examples the solution output for the new state or the regularized control variable graphically appears to be closer to truth.

For the EIC control delay and simple mixed delay problems we observed small craters in the solutions for the regularized control variables. Taking ε closer to zero could possibly improve this behavior as shown in Figures 4.9 and 4.11. Minor variations in these solutions are not that big of a surprise since they are influenced by an exogenous control that is of bang-bang type.



(a) ε = 1.0E-06 (b) $\varepsilon = 0$

Figure 4.11: Closeup of regularized control in Eq. (4.4)

The solution for the regularized control for the CSTR problem does not feature this behavior. However, the bangs in the CSTR exogenous control are confined to a single interval. In comparison to the original control delay optimal control solutions there was an improvement in accuracy by factors in a range of [1.0E-03, 1.0E-01] after application of the EIC method.

Although the regularized problems seem to be less difficult for $\mathbb{SOCX}$ to solve, in theory they are more complicated problems to solve for three primary reasons:

1. the dimension of the problem is increased,

2. the initial condition for the new states are free,

3. and the inclusion of the exogenous control.

For every delayed control variable a state, exogenous control, and state equation is added. This increases the number of NLP variables which in turn increases the execution time. However, $\mathbb{SOCX}$ exploits the sparsity of the equations which significantly reduces the work that would otherwise be necessary to yield a feasible result. Secondly, we allow the optimizer to freely choose the initial condition for each new state variable. This was important in that it allowed for a jump discontinuity when required. In this case the initial condition is influenced by the parameter, ε.

At the start, $\mathbb{SOCX}$ requires the user to submit an initial guess. Depending on the closeness of the initial guess the algorithm will either add points in the region to recalculate the value or bypass it as a valid value. Either way, because the initial value is not specified error can propagate in this region on successive iterations. Consequently, the largest error between the new state and the true delayed control mostly occurs near the initial value point. To minimize this error we experimented with specifying and varying the initial condition for the delayed control variable. It was observed that tiny variations in the initial condition could produce oscillations in the solution. However, when the initial condition was taken to be close to the true value, $u(0)$ the error of the approximation was significantly reduced over the entire interval, and a slightly more accurate solution was obtained provided the grid was not too fine.

Whether delayed or undelayed complex problems of this type can be very difficult to solve numerically. As noted in [98], optimization problems with free initial conditions can generally be solved very fast with sequential quadratic programming methods, but the global solution may be difficult to achieve when the system under consideration is highly nonlinear or multi-modal. The presence of free initial conditions is often accompanied by some type of bounded control as displayed in [85] and [98]. The bounds are necessary to assist in achieving the best control policies. Free initial condition optimal control problems are less studied and continue to be an open area of focus.

4.2.2.1 EIC and the Exogenous Control Variable

In each of the above examples the exogenous input control method was applied to address the chatter observed in delayed control approximations due to issues with interpolation on nonuniform grids (See Section 4.1). The results showed that application of the EIC method helped to remove the chatter and improved the overall approximation of the delayed control and other system variables. Although there was an improvement in accuracy of the solutions, we now have another issue! The exogenous control is an "undelayed" control, yet it is chattery as well. It was hypothesized that oscillatory behavior would only appear in controls that were delayed. However, this is not the case!

The exogenous control variable is an unrelated variable introduced into the regularized system to force the regularized control to simulate the behavior of the original delayed control. When inputting outside variables into a system one must take caution as unwanted side effects may result. $\mathbb{SOCX}$ rewrites the system dynamics of optimal control problems with delays as constrained delay differential algebraic equations and uses interpolants to compute the values for the delayed terms. Since the effect of inputting the exogenous control into the system appears to be positive, we wonder if the chatter seen in the exogenous control approximation is derived from the software itself. Is it something about the design of the algorithm in $\mathbb{SOCX}$ that causes this to happen? We investigate the presence of the chattery behavior seen in exogenous controls by developing a Matlab code that works in a similar fashion as $\mathbb{SOCX}$. The code is designed to discretize the regularized simple mixed delay example in (4.5) with the trapezoid method and interpolate delayed state variables accordingly with Hermite polynomials. The Matlab optimization solver, *fmincon* is then called with the NLP algorithm '*sqp*' and default tolerances to optimize the NLP system.

The NLP that $\mathbb{SOCX}$ Solves for ESMD

The $\mathbb{SOCX}$ DDAE formulated for Eq. (4.5) is

$$Minimize \quad J = \int_0^5 x_1^2 + x_2^2 + \epsilon z^2 \, dt, \tag{4.7a}$$

subject to the dynamics

$$\begin{aligned}
\dot{x}_1(t) &= y_1 + y_2, && [0,\ 5] && (4.7b)\\
\dot{x}_2(t) &= z, && && (4.7c)\\
y_1(t) &= 1, && -2 \le t < 0 && (4.7d)\\
y_2(t) &= 1, && -1 \le t < 0 && (4.7e)\\
0 &= y_1 - x_1(t-2), && && (4.7f)\\
0 &= y_2 - x_2(t-1), && && (4.7g)
\end{aligned}$$

with the following initial, startup, and boundary conditions

$$x_1(0) = 1, \tag{4.7h}$$

$$x_2(0) = a, \tag{4.7i}$$

$$0 \leq z \leq 5. \tag{4.7j}$$

Recall that $y_1(t)$ and $y_2(t)$ are pseudo variables implemented to remove the delayed variables from the state equations. Additionally, note the consistency relationship in (4.7f) and (4.7g). We discretize (4.7) using the trapezoid method, and interpolate the delayed state variables with Hermite polynomials. The goal of the resulting NLP is to minimize the objective function

$$J = \sum_{k=0}^{N} x_{1k}^2 + x_{2k}^2 + \epsilon z_k^2, \tag{4.8a}$$

subject to the defect constraints

$$0 = x_{1k+1} - x1_k - \frac{h_k}{2}\left(y_{1k} + y_{2k} \ + \ y_{1k+1} + y_{2k+1}\right), \qquad k = 0, \ldots, N-1 \tag{4.8b}$$

$$0 = x_{2k+1} - x_{2k} - \frac{h_k}{2}\left(z_k \ + \ z_{k+1}\right), \qquad k = 0, \ldots, N-1 \tag{4.8c}$$

and the algebraic constraints defined

$$0 = \begin{cases} y_{1k} - 1, & t_{k-2} < 0 \\ y_{1k} - c_1 x_{1j} + c_2 x_{1j+1} + c_3 f_{1j} + c_4 f_{1j+1}, & t_{k-2} \geq 0 \quad \text{and} \quad t_j \leq t_{k-2} \leq t_{j+1} \end{cases}, \tag{4.8d}$$

$$0 = \begin{cases} y_{2k} - 1, & t_{k-1} < 0 \\ y_{2k} - c_1 x_{2j} + c_2 x_{2j+1} + c_3 f_{2j} + c_4 f_{2j+1}, & t_{k-1} \geq 0 \quad \text{and} \quad t_j \leq t_{k-1} \leq t_{j+1} \end{cases}, \tag{4.8e}$$

with the following boundary values

$$1 \leq x_{10} \leq 1, \tag{4.8f}$$

$$0 \leq z_k \leq 5, \qquad k = 0, \ldots, N. \tag{4.8g}$$

Here Eq. (4.8d) and Eq. (4.8e) are defined for $k = 0, \ldots, N$ where the coefficients for the Hermite interpolating polynomials are defined as in Eq. (3.8d). In Matlab we solve (4.8) on [0, 5] with a uniform mesh of 21 points, and then on the 41 point and 73 point nonuniform grids produced by $\mathbb{SOCX}$ at the second and third iterations.

NLP Results Generated with Matlab and $\mathbb{SOCX}$

In Figures 4.2 and 4.8 are plotted the results for the original and regularized formulated simple mixed delay examples. The results from the NLP runs are plotted in Figures 4.12-4.14. Each figure displays Matlab and $\mathbb{SOCX}$ solutions for state $x_1(t)$, regularized control $x_2(t)$, and exogenous control $z(t)$. The results show that $\mathbb{SOCX}$ does a better job of finding the optimal values for the state variables. In Figure 4.12 although the Matlab approximation for $x_2(t)$ appears similar, $\mathbb{SOCX}$ displays a closer approximation since $x_2(0) \approx u(0) \approx -5.13$.

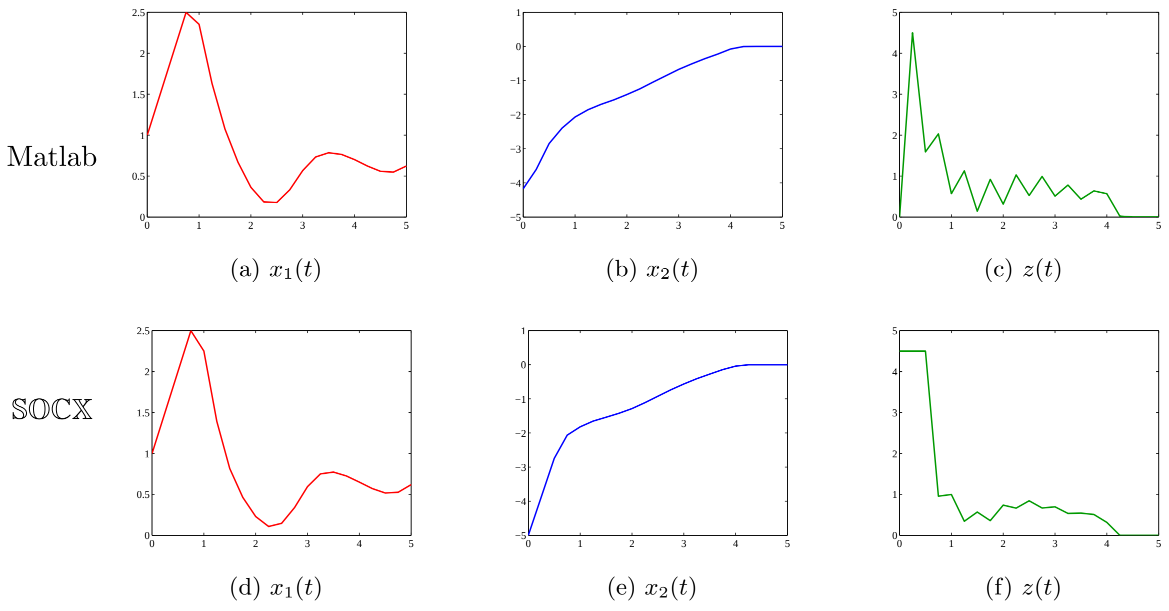
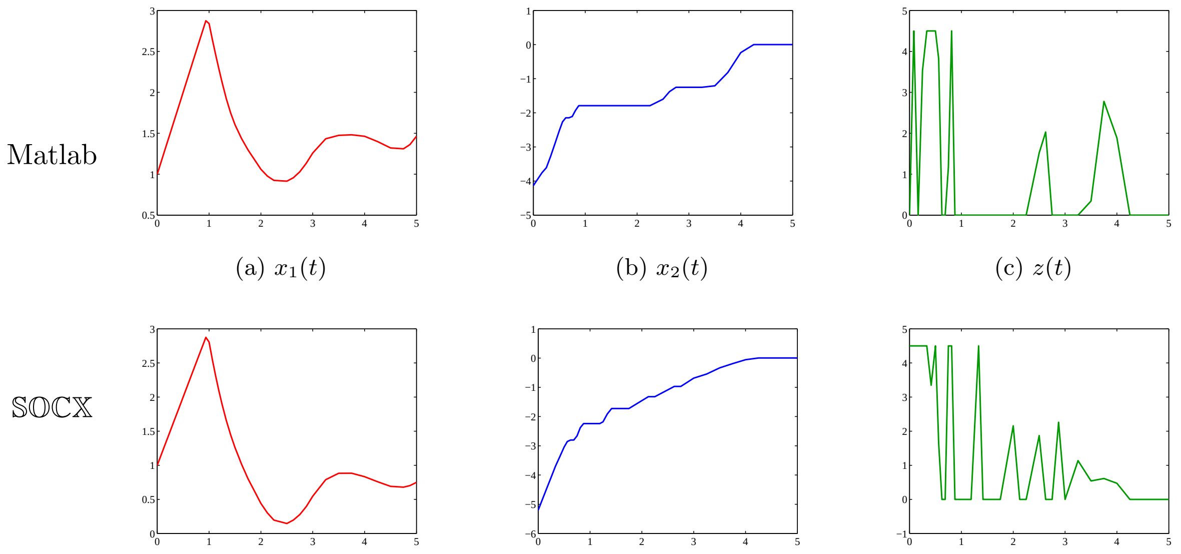


Figure 4.12: Iteration 1: Solutions for (4.8), $\varepsilon = 10^{-4}$

In Figure 4.13 we observe drastic differences in the values computed for the states. Note that the tail end of $x_1(t)$ computed with Matlab has increased from approximately 0.7 to 1.5. The only physical change for $x_1(t)$ for $\mathbb{SOCX}$ occurs at $t = 1$. Additionally, note how the regularized control computed by Matlab is constant on the interval [1, 2] and [3, 4]. These intervals are constant due to the exogenous control in Figure 4.13c being zero. This connection is expected, and can be seen in the $\mathbb{SOCX}$ solutions for the regularized and exogenous controls as well. In Figure 4.14 there is an improvement in the Matlab computed regularized control. However, note that $x_2(0) \approx -4$ and that $x_1(t)$ is increasing on [3, 5].

What is more on topic here is the solution output for the exogenous control. The true exogenous control for the ESMD regularized problem with ε=1.0E-04 is plotted in Figure 4.15b.



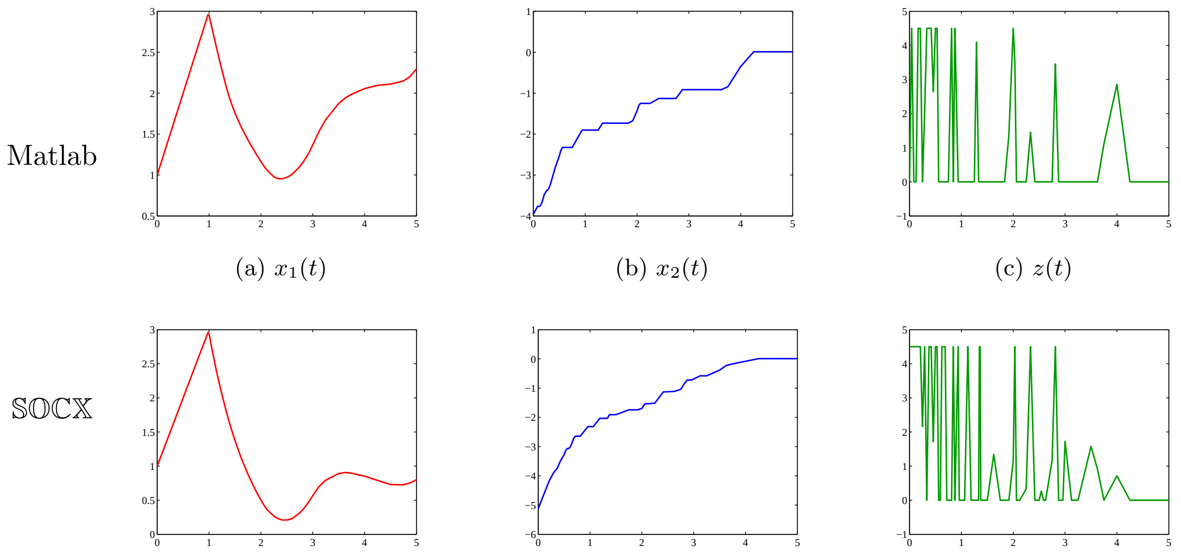
(a) $x_1(t)$ (b) $x_2(t)$ (c) $z(t)$

(d) $x_1(t)$ (e) $x_2(t)$ (f) $z(t)$

Figure 4.13: Iteration 2: Solutions for (4.8), $\varepsilon = 10^{-4}$

Matlab produces similar exogenous controls in comparison to $\mathbb{SOCX}$. In iteration 1 the solutions are computed on a 21 point uniform grid. If the sharp internal nodes could be removed from the solutions in Figures 4.13c and 4.13f, the exogenous control solutions for both Matlab and $\mathbb{SOCX}$ would appear close to the true exogenous control solution. This potentially shows that Matlab performs better with uniform grids. In iterations 2 and 3 the exogenous control solutions are slightly different. However, both Matlab and $\mathbb{SOCX}$ produce some type of chattery or bang-bang exogenous control.

The Matlab results show that the issue with the chattery exogenous controls is not just an issue



(a) $x_1(t)$ (b) $x_2(t)$ (c) $z(t)$

(d) $x_1(t)$ (e) $x_2(t)$ (f) $z(t)$

Figure 4.14: Iteration 3: Solutions for (4.8), $\varepsilon = 10^{-4}$

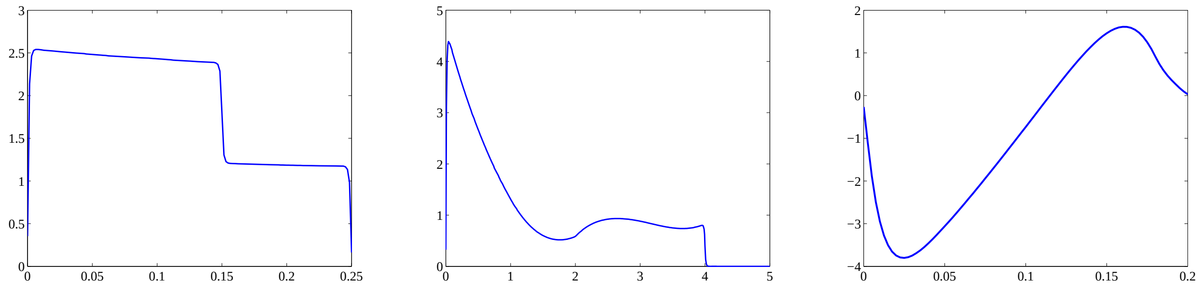
intrinsic to 𝕊𝕆ℂ𝕏. Instead, it reveals a deeper issue more so related to system construction and system dynamics. What is it about the characteristics of EIC formulated systems that causes the solution of the exogenous control to be chattery or bang-bang?

EIC Creates a Singular Optimal Control Delay Problem

The most alarming side effect of the regularization method is the fact that the exogenous controls are nearly linear in the optimal control delay system when $\varepsilon \approx 0$. When the slope bounds are present the regularized problem is considered a constrained optimal control problem. Recall that Pontryagin's minimum principle applied to a constrained control problem seeks to find a control that minimizes the Hamiltonian as in Eq. (1.4). When $\varepsilon > 0$, in theory Pontryagin's minimum principle can be applied to determine an optimal control of bang-bang type. However, only simple problems may be solved analytically. When $\varepsilon = 0$, the exogenous control is linear in the dynamics and does not appear in the cost function. In this case PMP fails to determine a unique value for the control, and hence the control is **singular** [44].

Research indicates that singular control problems may produce chattery controls or bang-bang controls. In [97], a treatment of the control of a class of nonlinear mechanical systems is considered which requires the objective to zero the states of the system with a bounded control, while considering some state dependent cost. The author notes that such controls often produce chatter due to the nonlinearities and bounds on the control. In [83], the authors state that optimal control problems linear in both dynamics and performance index may actually consist of a combination of intervals of variable control effort (called "singular control") combined with intervals of bang-bang control (as seen in 4.10e).

Because the exogenous control is not associated with the original OCDP and is only used to drive the values of the regularized control to the values of the true delayed control variable, we were not concerned with its solution. However, after considering its involvement in the regularized system we now view the exogenous control as a nearly singular or singular control relative to the value of ε. Hence, it is important to know if $\mathbb{SOCX}$ is approximating them appropriately. To investigate this we solved the regularized systems with MOS. The exogenous control solutions for the test problems are plotted in Figures 4.6d, 4.8e, and 4.10e. However in Figure 4.15, method of steps solutions for the EIC formulated problems reveal very different exogenous controls.



(a) ECDP, ε = 1.0E-06 (b) ESMD, ε = 1.0E-04 (c) CSTR , ε = 1.0E-08

Figure 4.15: MOS exogenous controls for test problems

The MOS solutions do not appear to be of the bang-bang type.

While we are pleased with the results of the EIC method, this fact possibly exposes a class of problems that may be difficult for $\mathbb{SOCX}$ and other direct transcription algorithms to handle. Optimal control problems with singular controls are challenging to solve both analytically and numerically. In [83], conditions which characterize singular control for a specific class of singular control problems are derived and techniques for detecting and calculating these singular controls are given. The authors indicate that there seems to be some uncertainty as to general methods of handling such problems. In [144], the authors apply two numerical methods to a singular control problem, and demonstrate how the numerical solution methods can easily get trapped in what may appear to be local optima when applied to this problem. As we further analyze the convergence of the regularized solutions to the true solutions it is necessary to keep in mind that application of singular techniques may be necessary to adequately assess the performance of the EIC method. Singular optimal control problems are discussed further in the next chapter.

Chapter 5

Analysis and Convergence of EIC

In this chapter we study the convergence of the exogenous input control solutions. The EIC method is a method that regularizes optimal control problems with control delays by reformulating the problem as a singular optimal control problem with state delays, also termed regularized controls. In addition to the three problems solved in Section 4.2.1, the EIC method was used to approximate the solutions to other optimal control delay problems featured in our test set. The average max approximation error was of a magnitude of 10E-04. Consequently, numerical results show that the EIC method is successful in approximating the solutions to a variety of optimal control delay problems. However, it is of particular interest to determine just how well the EIC method is doing. In other words it is important to know if the exogenous input control solutions are converging to the analytic solutions of the original problem. If so, in what sense are they converging? Convergence of the EIC solutions to the solutions of the original problem will provide analytic evidence that the regularized problem is equivalent to the original optimal control problem.

To carry out the exogenous input control method, state delay variables replace control delay variables and state equations are added to the original system. The exogenous control is constrained and entered into the cost function as a weighted quadratic term. Aside from the state delay variable replacing the control delay variable, this addition to the cost function slightly changes the problem. If ε is large, optimization may possibly yield very different solutions in comparison to the optimal solutions of the original problem. However, the EIC method takes $\varepsilon > 0$ to be small so that its objective function reflects the objective function of the original problem. Very few studies concerning the convergence properties of the solutions to optimal control delay systems using theoretical techniques are available [62, 63, 64]. Consequently, we transform our system into an undelayed problem and apply familiar techniques that have been used to study the solutions of more general optimal control systems.

Our first effort conducts system analysis using a method of steps approach. Using this concept we show that the solution of the EIC problem converges to the optimal solution of the original optimal control delay problem by assessing the dynamics of the solution as $\varepsilon \to 0$. Secondly, we use an ε asymptotic approximation approach to analyze the convergence of the EIC method from a singular optimal control point of view. Asymptotic expansions are used in analysis to describe the limiting behavior and properties of a solution when a very small parameter is involved. The asymptotic expansions help to reduce the primary system into a set of simpler equations in which analytic forms of the solution may sometimes be obtained. With this approach we derive the asymptotic expansions for the regularized problem and assess the solution as $\varepsilon \to 0$.

Analysis of the EIC method is initially restricted to a specific time invariant optimal control delay problem whose goal is to minimize

$$J = \frac{1}{2}\int_0^T q_1 x^2(t) + q_2 u^2(t)\, dt, \tag{5.1a}$$

subject to

$$\dot{x}(t) = u(t-s), \tag{5.1b}$$

$$u(t) = 1, \qquad -s \le t < 0 \tag{5.1c}$$

$$x(0) = q_3. \tag{5.1d}$$

The test problem was created purposely to exclude state delays in an effort to isolate issues related to control delays. The EIC formulation for (5.1) is featured in (5.2). In [21], [22], and [23] are numerical results related to (5.1) with parameters $q_1 = q_2 = 2$, $q_3 = 1$, $s = 1$, and $T = 5$. For consistency purposes we conduct analysis with the same parameters, but note that results provided are directly related to any problem with a constant delay, s. Additionally, we note that the test problem is of the simplest form necessary for the exogenous input control method to be applied, and serves as a framework for more complicated problems. More general problems are discussed later. After applying the EIC method the goal of the regularized problem is to minimize the objective function

$$\tilde{J} = \frac{1}{2}\int_0^T x^T(t)\begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix} x(t) + \varepsilon z_\varepsilon^2(t)\, dt, \tag{5.2a}$$

subject to

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t-s) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} z_\varepsilon(t), \tag{5.2b}$$

with the startup function and initial condition defined

$$x_2(t) = 1, \qquad -1 \le t < 0 \tag{5.2c}$$

$$x_1(0) = q_3, \tag{5.2d}$$

with the initial condition for $x_2(0)$ unspecified. The optional bound for the exogenous control is defined

$$0 \le z_\varepsilon(t) \le 2. \tag{5.2e}$$

In (5.2) the subscript appears on the exogenous control to emphasize the impact that the perturbation parameter has on its approximated value.

5.1 Method of Steps Approach

In Section 4.2.1 assessment of solutions and max errors provided numerical evidence that the solutions of the regularized problems were close to the method of steps solutions for the original optimal control delay problem. We further support this with an analytic argument that uses the method of steps, Pontryagin's Minimum Principle, and basic concepts of convergence. Let ORG denote the original optimal control delay problem in (5.1), OMOS denote the optimized method of steps formulation for the original optimal control delay problem, and let EMOS denote the optimized method of steps formulation for the EIC problem (5.2). Then we derive the following proposition.

Proposition 5.1.1. *If the EMOS solution coverges to the OMOS solution as $\varepsilon \to 0$, then the EIC solution converges to the ORG solution as $\varepsilon \to 0$ in some manner.*

The idea stems from the fact that the method of steps combined with PMP provides an analytic solution to the optimal control delay problem. We know that the OMOS solution is the optimal solution for the ORG problem. Similarly, the EMOS solution is the optimal solution for the EIC problem. Consequently, if we can show that the OMOS solution is the limiting solution for the EMOS problem as $\varepsilon \to 0$ then we can conclude that the EIC solution is indeed converging to the solution for the ORG problem as $\varepsilon \to 0$ since equivalent forms will converge in the same manner. To construct the OMOS and EMOS systems we first apply the method of steps to remove the delay and then optimize using PMP. The MOS formulations are carried out with $N = 5$ steps. The calculations are expressed in terms of vectors for a clearer view of the application of the method of steps. For both OMOS and EMOS the optimal solutions are plotted and the convergence properties are discussed.

5.1.1 OMOS: PMP applied to the MOS formulation for (5.1)

To formulate the problem using a method of steps technique we break the domain $[0,\ 5]$ into N steps of length $s = 1$. The MOS integration interval is $[0,\ \delta]$, where $\delta = N/T = 1$. We denote the values in step k by x_k and u_k with

$$x_k(\alpha) = x(t), \qquad (k-1)\delta \le t \le k\delta \tag{5.3a}$$

$$u_k(\alpha) = u(t), \qquad (k-1)\delta \le t \le k\delta \tag{5.3b}$$

where $0 \le \alpha \le \delta$ and $t = \alpha + (k-1)\delta$ for $t \in [(k-1)\delta,\ k\delta]$. The goal is then to minimize the objective function

$$F = \frac{1}{2}\int_0^{\delta} \sum_{k=1}^{N} x_k^2(\alpha) + u_k^2(\alpha)\, d\alpha, \tag{5.3c}$$

subject to the system of ODEs

$$\dot{x}_k = u_{k-1}, \qquad k = 1, \ldots, N \tag{5.3d}$$

defined in the region $0 \leq \alpha \leq \delta$. The initial condition and startup value are

$$x_1(0) = 1 \quad \text{and} \quad u_0(\alpha) = 1. \tag{5.3e}$$

The continuity requirements on the internal intervals lead to the boundary conditions

$$x_k(0) = x_{k-1}(\delta), \qquad k = 2, \ldots, N \tag{5.3f}$$

$$u_k(0) = u_{k-1}(\delta), \qquad k = 2, \ldots, N. \tag{5.3g}$$

The optimal control system on $[0,\ T]$ is reformulated as a system of 5 ODEs with respect to α on the interval $[0, \delta]$. Next we apply PMP outlined in Section 1.0.2 to formulate the optimality conditions. The Hamiltonian related to Eq. (5.3) is computed as

$$\mathcal{H} = \frac{1}{2} \sum_{k=1}^{N} x_k^2 + u_k^2 + \lambda_1 + \sum_{k=1}^{N-1} \lambda_{k+1} u_k, \tag{5.4a}$$

and the necessary conditions for an optimal trajectory are defined

$$\dot{x}_1 = 1, \tag{5.4b}$$

$$\dot{x}_{k+1} = u_k, \qquad k = 1, \ldots, (N-1) \tag{5.4c}$$

$$\dot{\lambda}_k = -x_k, \qquad k = 1, \ldots, N \tag{5.4d}$$

$$0 = u_k + \lambda_{k+1}, \qquad k = 1, \ldots, (N-1) \tag{5.4e}$$

$$0 = u_N. \tag{5.4f}$$

Equations (5.4e) and (5.4f) determine the optimal control. Substituting values for u_k into Eq. (5.4c) we have the following updated state equation

$$\dot{x}_{k+1} = -\lambda_{k+1}, \qquad k = 1, \ldots, (N-1). \tag{5.4c*}$$

The terminal state for the OMOS problem is free. Hence, the terminal value for the adjoint variable, $\lambda_N(\delta) = 0$. In addition to the boundary conditions derived from PMP we must employ the boundary conditions generated by the method of steps in (5.3f) and (5.3g). The updated boundary conditions are defined

$$x_1(0) = 1, \tag{5.4g}$$

$$x_{k+1}(0) = x_k(\delta), \qquad k = 1, \ldots, (N-1) \tag{5.4h}$$

$$\lambda_{k+1}(0) = \lambda_k(\delta), \qquad k = 1, \ldots, (N-1) \tag{5.4i}$$

$$\lambda_N(\delta) = 0. \tag{5.4j}$$

5.1.2 EMOS: PMP applied to the MOS formulation for (5.2)

The method of steps is carried out for the EIC problem in a similar manner. At the k^{th} step the state variables are denoted x_{1k} and x_{2k}, and the control variable is denoted z_k. For the two dimensional system we break up the domain into N steps of length $s = 1$. The MOS integration interval is $[0,\ \delta]$, where $\delta = N/T = 1$. We define the values at each step as

$$x_{1k}(\alpha) = x_1(t), \qquad (k-1)\delta \leq t \leq k\delta \tag{5.5a}$$

$$x_{2k}(\alpha) = x_2(t), \qquad (k-1)\delta \leq t \leq k\delta \tag{5.5b}$$

$$z_k(\alpha) = z_\varepsilon(t), \qquad (k-1)\delta \leq t \leq k\delta \tag{5.5c}$$

for $k = 0, \ldots, N$ and $0 \leq \alpha \leq \delta$ with $t = \alpha + (k-1)\delta$ for $t \in [(k-1)\delta,\ k\delta]$. The goal of the problem is then to minimize

$$\hat{F} = \frac{1}{2}\int_0^\delta \sum_{k=1}^{N} x_{1k}^2(\alpha) + x_{2k}^2(\alpha) + \varepsilon z_k^2(\alpha)\, d\alpha, \tag{5.5d}$$

subject to the $2N$ ordinary differential equations defined

$$\dot{x}_{1k} = x_{2k-1}, \qquad k = 1, \ldots, N \tag{5.5e}$$

$$\dot{x}_{2k} = z_k, \qquad k = 1, \ldots, N \tag{5.5f}$$

over $0 \leq \alpha \leq \delta$. The initial condition and startup value are given by

$$x_{11}(0) = 1 \quad \text{and} \quad x_{20}(\alpha) = 1. \tag{5.5g}$$

Continuity leads to the boundary conditions

$$x_{1k}(0) = x_{1k-1}(\delta), \qquad k = 2, \ldots, N \tag{5.5h}$$

$$x_{2k}(0) = x_{2k-1}(\delta), \qquad k = 2, \ldots, N \tag{5.5i}$$

$$z_k(0) = z_{k-1}(\delta), \qquad k = 2, \ldots, N. \tag{5.5j}$$

To obtain the optimality conditions for Eq. (5.5) we formulate the following Hamiltonian

$$\hat{\mathcal{H}} = \frac{1}{2}\sum_{k=1}^{N} x_{1k}^2 + x_{2k}^2 + \varepsilon z_k^2 + \lambda_{2k} z_k + \lambda_{11} + \sum_{k=1}^{N-1} \lambda_{1k+1} x_{2k}, \tag{5.6a}$$

where the k^{th} step for the Lagrange multipliers, $\lambda_1(\alpha)$ and $\lambda_{2\varepsilon}(\alpha)$ are defined λ_{1k} and λ_{2k} respectively. Here, the parameter subscript appears on $\lambda_{2\varepsilon}$ because the costate varies directly with z_ε. The necessary conditions that follow are defined

$$\dot{x}_{11} = 1, \tag{5.6b}$$

$$\dot{x}_{1k+1} = x_{2k}, \qquad k = 1, \ldots, (N-1) \tag{5.6c}$$

$$\dot{x}_{2k} = z_k, \qquad k = 1, \ldots, N \tag{5.6d}$$

$$\dot{\lambda}_{1k} = -x_{1k}, \qquad k = 1, \ldots, N \tag{5.6e}$$

$$\dot{\lambda}_{2k} = -x_{2k} - \lambda_{1k+1}, \qquad k = 1, \ldots, (N-1) \tag{5.6f}$$

$$\dot{\lambda}_{2N} = -x_{2N}, \tag{5.6g}$$

$$0 = \varepsilon z_k + \lambda_{2k}, \qquad k = 1, \ldots, N. \tag{5.6h}$$

Note that the control bound in Eq. (5.2e) is not required since we are deriving the solution analytically. However, in any case of an active control constraint the resulting optimality condition may be a switching function, singular arc, or combination of both depending on the value of ε. For a more detailed explanation see [92]. Similarly, the associated Hamiltonian system in $x_{1k}(\alpha)$, $x_{2k}(\alpha)$, $\lambda_{1k}(\alpha)$, and $\lambda_{2k}(\alpha)$ can be formed by solving Eq. (5.6h) for z_k and substituting the result in Eq. (5.6d). Now utilizing boundary condition information and ensuring continuity across the internal intervals the EMOS boundary conditions are defined

$$x_{11}(0) = 1, \tag{5.6i}$$

$$x_{1k+1}(0) = x_{1k}(\delta), \qquad k = 1, \ldots, (N-1) \tag{5.6j}$$

$$x_{2k+1}(0) = x_{2k}(\delta), \qquad k = 1, \ldots, (N-1) \tag{5.6k}$$

$$\lambda_{1k+1}(0) = \lambda_{1k}(\delta), \qquad k = 1, \ldots, (N-1) \tag{5.6l}$$

$$\lambda_{1N}(\delta) = 0, \tag{5.6m}$$

$$\lambda_{21}(0) = 0, \tag{5.6n}$$

$$\lambda_{2k+1}(0) = \lambda_{2k}(\delta), \qquad k = 1, \ldots, (N-1) \tag{5.6o}$$

$$\lambda_{2N}(\delta) = 0. \tag{5.6p}$$

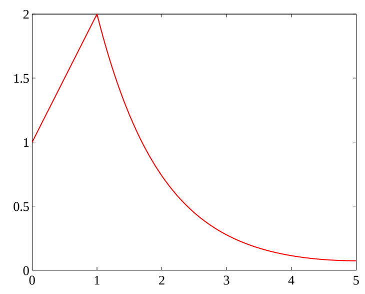
It is important to note a few things about the optimized regularized system. The exogenous input control method does not require an initial condition for the new state variables. Hence, the new state variables are free at each end-point. Consequently, in the optimized system the associated multiplier is zero at each end-point as noted in (5.6n) and (5.6p). Note that with the additional boundary condition for the multiplier, the associated Hamiltonian system for Eq. (5.6) constitutes a square BVP with $x_{11}(0)$ specified and $\lambda_{1N}(\delta) = 0$. With this variation, the solution of the regularized control is dependent upon other variables.

5.1.3 Results Summary for OMOS and EMOS

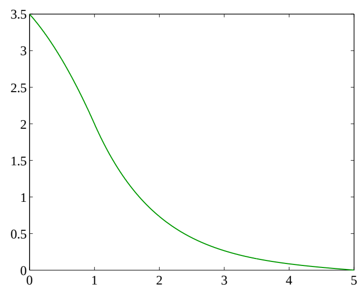
A Matlab program was generated to solve the OMOS and EMOS systems. The 5 part solution was pieced together to form the optimal trajectories on $[0,\ 5]$. Hence, $x(t) = [x_1\ x_2\ x_3\ x_4\ x_5]^T$ in OMOS and $x_1(t) = [x_{11}\ x_{12}\ x_{13}\ x_{14}\ x_{15}]^T$ in EMOS. Definition of the other variables follow analogously. The optimal solutions for the OMOS and EMOS problems are featured in Figures 5.1 and 5.2. We compare the two figures to assess the convergence of the EMOS solutions. Within our discussion we reference a few theorems commonly applied when mathematically studying the convergence of solutions.

Theorem 5.1.1 (Rolle's Theorem). *Let f be continuous on a closed interval $[a,\ b]$ and differentiable on the open interval $(a,\ b)$. If $f(a) = f(b)$, then there is at least one point c in $(a,\ b)$ where $f'(c) = 0$.*

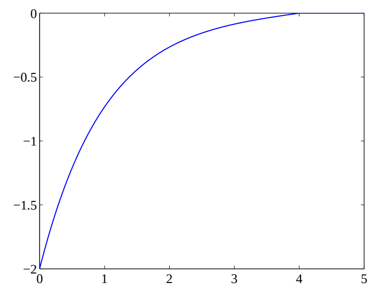
Theorem 5.1.2 (Weierstrass M-test for Uniform Convergence). *Suppose that $f_n \to f$ pointwise, and let $M_n = sup_{x\in E}\ d(f_n, f)$ where $(E,\ d)$ is a metric space with distance $d : E \times E \to \mathbb{R}$. Then $f_n \to f$ uniformly iff $M_n \to 0$ as $n \to \infty$.*



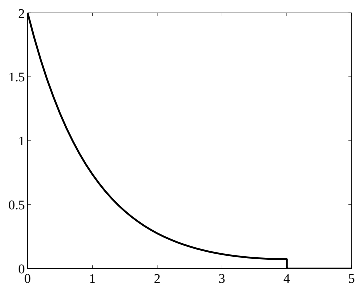
(a) State $x(t)$



(b) Costate $\lambda(t)$



(c) Control $u(t)$

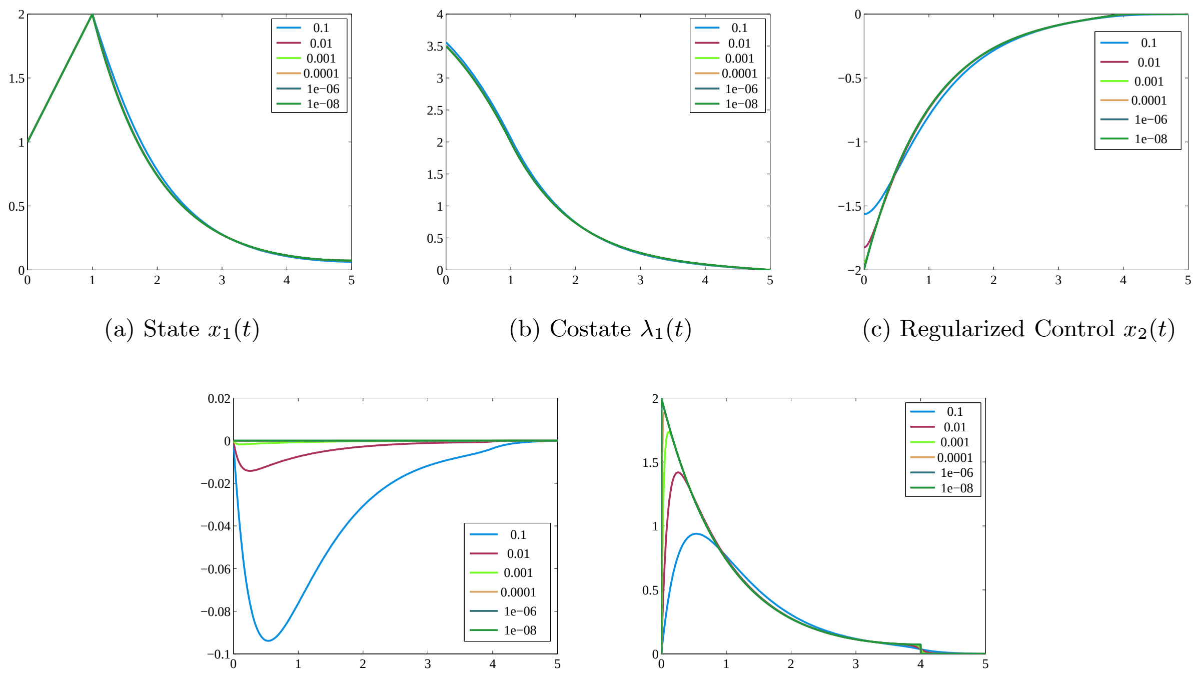


(d) Control Derivative $\dot{u}(t)$

Figure 5.1: OMOS Solution

To simplify notation we first define the solution vector to a system with variables $a(t)$, $b(t)$, and $c(t)$ as $[[a(t), b(t), c(T)]] = [a(t)^T, b(t)^T, c(t)^T]^T$. Now note that with the substitution $x_2(t) = u(t)$ that there is a distinct similarity between the EMOS solution vector, $[[x_1(t),\ \lambda_1(t),\ x_2(t)]]$ and the OMOS solution vector, $[[x(t),\ \lambda(t),\ u(t)]]$. The OMOS dynamic equations (5.4b)-(5.4d) for state $x(t)$ and costate $\lambda(t)$ are the same as the EMOS dynamic equations (5.6b), (5.6c), and (5.6e) for state $x_1(t)$ and costate $\lambda_1(t)$. However since $x_2(t)$ varies with ε in the EMOS problem, each variable is influenced by the parameter ε. On $[0,\ 1]$ note that $x_1(t) \equiv x(t)$. On $[1,\ 4]$, Figure 5.2a shows that all solutions lie within some epsilon band of the true state solution. Hence, $x_1(t) \to x(t)$ uniformly on $[1,\ 4]$. Similarly, in Figures 5.1b-5.1c and Figures 5.2b-5.2c, we also determine that $\lambda_1 \to \lambda$ and $x_2 \to u$ uniformly on $[0,\ T]$. Thus, the EMOS solution vector converges uniformly to the true solution on $[0,\ T - s]$.

Although the costate $\lambda_{2\varepsilon}(t)$ and exogenous control $z_\varepsilon(t)$ are not a part of the solution vector with respect to the OMOS problem, they have an important influence on the computation of the optimal



(a) State $x_1(t)$ (b) Costate $\lambda_1(t)$ (c) Regularized Control $x_2(t)$

(d) Costate $\lambda_{2\varepsilon}(t)$ (e) Exogenous Control $z_\varepsilon(t)$

Figure 5.2: EMOS Solution

regularized control. With $x_2(t) = u(t)$ note that if $\lambda_{2\varepsilon}(t) = 0$ and $\varepsilon = 0$, then Eq. (5.6) is identically equal to Eq. (5.4). In this case we also have $z_\varepsilon(t) \equiv \dot{u}(t)$ which is also evident in Eq. (5.2). Thus, in order to have full convergence of the EMOS solutions to the OMOS solutions we must in some sense have $\lambda_{2\varepsilon}(t)$ converging to zero and $z_\varepsilon(t)$ converging to $\dot{u}(t)$ as $\varepsilon \to 0$.

Figure 5.2d displays the optimal solution for costate $\lambda_{2\varepsilon}(t)$ computed for various ε. Because of its dependence on the perturbation parameter, $\lambda_{2\varepsilon}(t)$ can be viewed as an ε sequence of costate solutions defined $\Lambda_n(t) = \lambda_{2\varepsilon_n}(t)$. Note that the end-point values for Λ_n is 0 for every n as required by the boundary conditions. Application of Rolle's theorem tells us that $|\Lambda_n| \leq M_n$, where $M_n = |\Lambda_n'(c_\varepsilon)|$, for some c_ε. Note that $M_n \to 0$ as $n \to \infty$, and hence by Theorem 5.1.2 $\lambda_n \to 0$ uniformly on $[0,\ T]$. In Figure 5.2e observe that the exogenous control z_ε is bounded between 0 and 2, and is also equal to zero at the end-points. Similarly, we define $z_n(t) = z_{\varepsilon_n}(t)$ to be the sequence describing the exogenous control for various ε. Because of the impulse near $t = 0$, z_n cannot converge uniformly. However, note that as $n \to \infty$ the sequence z_n is getting closer to $\dot{u}(t)$. Moreover, the area between the two curves is getting smaller. Thus, $\int_0^T |\dot{u} - z_n|\,dt \to 0$ as $n \to \infty$ giving that z_n converges to $\dot{u}(t)$ in L^1 and z_n converges pointwise to $\dot{u}(t)$ on $(0,\,T]$ but not on the full interval $[0,\,T]$.

The computed results stated in the above discussion strongly suggests that the analytic solution to the exogenous input control is converging to the analytic solution of the original optimal control delay problem. Consequently, it is also suggested that the EIC solution is converging to the solution of the original problem. Similar convergence results could also be obtained by deriving the necessary and optimality conditions for the original and EIC problems in normal form. Recall that the presence of the delay causes a product between a characteristic function, $\mathcal{X}(t)$ and an advanced adjoint variable, $\lambda_{2\varepsilon}(t+s)$ to appear in the costate equation. The effect of the this product results in a jump discontinuity in the associated costate (See Figure 5.3). This is the case for $\lambda_{2\varepsilon}$ at $t = 4$ in the EIC problem. As shown in Figure 5.2e note that this jump discontinuity is also transferred to the exogenous control at $t = 4$ since $z_\varepsilon = -\lambda_{2\varepsilon}/\varepsilon$. For a complete statement of the necessary conditions for (5.1) and (5.2) see Appendix B.

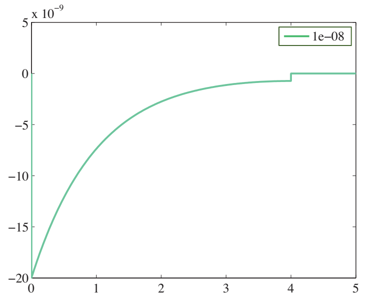


Figure 5.3: Jump Discontinuity in $\lambda_{2\varepsilon}(t)$ at $t = 4$

5.1.4 A Special Consequence of the Exogenous Input Control Method

Because of the convergence properties of the EIC solutions we inherit two interesting results that involve the relationship between the objective functions of the original optimal control delay problem and the regularized optimal control delay problem. First note that convergence of the EIC solution to the ORG solution leads to the following result

$$\left|J - \tilde{J}\right| = \frac{1}{2}\int_0^T \left|(x^2 + u^2) - (x_1^2 + x_2^2 + \varepsilon z_\varepsilon^2)\right| dt \to 0 \quad \text{as} \quad \varepsilon \to 0. \tag{5.7}$$

This tells us that there is an optimal vector $\bar{y}^* = [[x_1^*,\ x_2^*,\ z_\varepsilon^*]]$ that minimizes J, and also an optimal vector $y^* = [[x^*,\ u^*]]$ that minimizes $\tilde{J}$. In other words, y^* is a solution to EIC and $\bar{y}^*$ is a solution to ORG. We use this fact to show the following.

Proposition 5.1.2. *For the regularized problem in Eq. (5.2) z_ε is bounded by $\|\dot{u^*}\|$ in Eq. (5.1) in L^2.*

Proof. Consider that optimality has been obtained for both the original optimal control delay problem and the regularized problem. Then $y^* = [[x^*,\ u^*,\ \lambda^*]]$ is the optimal solution to OMOS and $\bar{y}^* = [[x_1^*,\ x_2^*,\ \lambda_1^*,\ \lambda_2^*,\ z_\varepsilon^*]]$ is the optimal solution for EMOS. Note that y^* is a solution to the EIC system by setting $z_\varepsilon^* = \dot{u^*}$. Similarly, $\bar{y}^*$ is a solution to ORG by setting $u^* = x_2^*$. Now since $[[x_1^*,\ x_2^*]]$ is a solution to ORG, we have

$$J(x^*,\ u^*) \leq J(x_1^*,\ x_2^*). \tag{5.8a}$$

Since $[[x^*,\ u^*,\ \dot{u^*}]]$ is a solution to EMOS we also have

$$\tilde{J}(x_1^*,\ x_2^*,\ z_\varepsilon^*) \leq \tilde{J}(x^*,\ u^*,\ \dot{u^*}). \tag{5.8b}$$

Now Eq. (5.8a) and Eq. (5.8b) give the following relations

$$\frac{1}{2}\int_0^T x^{*2} + u^{*2}\, dt \leq \frac{1}{2}\int_0^T x_1^{*2} + x_2^{*2}\, dt, \tag{5.8c}$$

and

$$\frac{1}{2}\int_0^T x_1^{*2} + x_2^{*2} + \varepsilon z_\varepsilon^{*2}\, dt \leq \frac{1}{2}\int_0^T x^{*2} + u^{*2} + \varepsilon \dot{u^*}^2\, dt. \tag{5.8d}$$

Now by combining inequalities we have

$$\int_0^T x^{*2} + u^{*2}\, dt \le \int_0^T (x^{*2} + u^{*2}) + \varepsilon z_\varepsilon^{*2}\, dt \le \int_0^T x_1^{*2} + x_2^{*2} + \varepsilon z_\varepsilon^{*2}\, dt \le \int_0^T x^{*2} + u^{*2} + \varepsilon \dot{u^*}^2\, dt. \quad (5.8e)$$

Thus, from the second and last terms in the above inequality we conclude that

$$\|z_\varepsilon^*\|_2^2 \ \le\ \|\dot{u^*}\|_2^2 \ \text{ for all } \ \varepsilon. \quad (5.8f)$$

In summary, if $\|\dot{u^*}\|$ is bounded, then z_ε^* is bounded in L^2 for all ε. □

Corollary 5.1.3. *For the optimized regularized problem in Eq. (5.6) $\lambda_{2\varepsilon}$ converges to the 0 function in L^2.*

Proof. Assume that $\dot{u^*}$ is bounded in L^2 by some $M > 0$ and consider $[x^*,\ u^*,\ \dot{u^*}]$ a solution to the EMOS problem, then according to (5.6h) and (5.8f) we have

$$\int_0^T \|\lambda_{2\varepsilon}^*\|_2^2\, dt \quad = \quad |-\varepsilon^2| \int_0^T \|z_\varepsilon^*\|_2^2\, dt \quad \le \quad |-\varepsilon^2| \int_0^T \|\dot{u^*}\|_2^2\, dt \quad \le \quad \varepsilon^2 M. \quad (5.9a)$$

Thus, $\lambda_{2\varepsilon} \to 0$ as $\varepsilon \to 0$ in L^2. □

Note that the proof for Proposition 5.1.2 holds for both delayed and undelayed optimal control problems with dynamics $\dot{x}(t) = Ax(t) + Bu(t-s)$ and general LQR cost function as in (5.11b). Proposition 5.1.2 will be useful in our discussion on the convergence of the asymptotic solutions for the EIC problem in the remaining parts of this chapter.

5.2 ε Asymptotic Approximation Approach

In this section we obtain an asymptotic solution to the simple regularized control delay test problem in Eq. (5.2). As noted in Section 4.2.2.1 application of the EIC method results in a nearly singular optimal control problem for small ε. When $\varepsilon = 0$, for linear optimal control problems application of the EIC method results in a singular optimal control problem. In each of these cases singular perturbation techniques may be required to determine a proper solution if Pontryagin's minimum principle cannot fully describe the control. In the presence of both nearly singular and singular controls chatter is usually present in the solution due to the discontinuous control laws. In mechanical systems chatter can cause wearing of mechanical components and may induce resonant modes; it should be avoided whenever possible [97]. Application of singular control techniques can eliminate this behavior.

Singular control problems have been studied extensively in the optimal control community [4, 34, 38, 39, 56, 83, 89, 90, 97, 107, 108]. Many of these papers employ a Hamiltonian approach in which the Hamiltonian matrices are scaled and permuted to preserve a singular perturbed form. A main result of this method is the use Kalman filtering techniques to decompose the associated algebraic Ricatti equations into reduced-order pure-slow and pure-fast, algebraic Riccati equations [59]. Alternatively, classical methods such as Taylor series expansion, asymptotic expansion, Picard, Newton, and averaging and continuation algorithms can be applied. In some cases the use of series or expansion methods is not suggested because generating higher order expansions for those methods have been analytically pretty cumbersome and numerically pretty inefficient, especially for high-dimensional control systems [59]. Although expansion methods are computationally expensive when a higher order of accuracy is required, the methods are well developed and require simple implementation. They are traditionally used for their advantage in being theoretical tools with the ability to remove ill-conditioning of the original problems and produce well conditioned, approximate reduced-order subproblems [58]. When permitted, an asymptotic expansion with $\mathcal{O}(\varepsilon)$ accuracy is sufficient in providing a function that is asymptotically equivalent to the solution of a given problem.

The method of matched asymptotic expansions (MMAE) is a basic, yet powerful way to obtain solutions to singularly perturbed systems. Singular problems mainly arise when a small parameter ε appears as a multiplicative factor in the governing equation e.g., $\varepsilon\ddot{y} = f(y,t)$. When $\varepsilon = 0$, the order of the system is reduced and an exact solution is lacking. The standard approach to handle this situation is to express the solution as an asymptotic power series of the form

$$y(x,\varepsilon) = y_0(x) + \varepsilon y_1(x) + \varepsilon^2 y_2(x) + \mathcal{O}(\varepsilon^3). \tag{5.10}$$

The expansion is then inserted into the governing equation, and a series of subsystems are formed by matching the coefficients of powers of ε. The solutions to each subsystem are combined to form a single approximate solution that describes the behavior of the solution over the entire interval as $\varepsilon \to 0$. The approach described is a very basic approach and is often not enough to obtain a full asymptotic solution to the singular optimal control problem.

The most widely applied asymptotic analysis for singular control problems involves obtaining asymptotic solutions for the problem on three regions:

- *outer* layer- describes the solution outside of the boundary,
- *initial* layer- describes the solution at the initial boundary, and
- *terminal* layer- describes the solution at the terminal boundary.

For consistency purposes the initial and terminal boundary solutions are usually expressed in stretched time variables in order to satisfy boundary conditions. The full asymptotic solution is then expressed as the sum of the outer, initial, and terminal solution minus the common pieces. Asymptotic expansions of this form for singularly perturbed systems have been studied intensively by Robert E. O'Malley Jr., and can be reviewed in [107]. Since publication of his singular perturbations book, O'Malley has contributed both solo and team publications to the application of asymptotic analysis to singular control problems [90, 106, 108, 109, 110]. In the next section we highlight results from O'Malley and Jameson to help to derive the asymptotic solution to the EIC test problem.

5.2.1 An Asymptotic Solution to the EIC Test Problem

While asymptotic expansions are commonly used to obtain solutions to singular optimal control systems, very seldom have they been used in the study of singular optimal control systems with time delays [62, 68, 116]. For this reason manipulation of the problem may be required before applying any of the singular perturbation techniques currently available. Before moving into the discussion about the asymptotic solution to the regularized simple control delay test problem it is convenient to first state O'Malley and Jameson's results. They prove that there exists an asymptotic solution to the *cheap control* problem. A cheap control problem is characterized by a small penalty factor applied to the control in the quadratic performance criterion, usually one or more orders of magnitude smaller than that of the state penalty term [58]. The existence of the asymptotic solution to the cheap control problem is based upon the characteristics of the system matrices. We outline O'Malley and Jameson's results in the following theorem. We note that the authors use ε to denote the perturbation parameter, but here we use θ instead to avoid confusion with the notation for the exogenous input control method.

Theorem 5.2.1 (O'Malley and Jameson's Asymptotic Expansion Theorem for Singular Optimal Control Systems). *Consider the n-dimensional state regulator problem*

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad x(0) \text{ prescribed} \tag{5.11a}$$

with the scalar quadratic performance index

$$J(\theta) = \frac{1}{2}\int_0^T x^T(t)Qx(t) + \theta^2 u^T(t)Ru(t)\, dt, \tag{5.11b}$$

to be minimized by selection of the n_u-dimensional control vector u for symmetric matrices $Q \geq 0$ and $R > 0$. If $n_u < n$ and $B^TQB > 0$, then for each sufficiently small $\theta > 0$ and each $N > 0$, the optimal control u, corresponding trajectory x, and the optimal cost J^* will have the uniform asymptotic approximations as follows

$$u(t,\theta) = \frac{v_0(\tau)}{\theta} + \sum_{j=0}^{N} \left(U_j(t) + v_{j+1}(\tau) + w_j(\sigma)\right)\, \theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.12a}$$

$$x(t,\theta) = X_0(t) + m_0(\tau) + \sum_{j=1}^{N} \left(X_j(t) + m_j(\tau) + n_{j-1}(\sigma)\right) \theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.12b}$$

and

$$J^*(\theta) = \frac{1}{2}\sum_{j=0}^{N} J_j^*\theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.12c}$$

with the stretched time coordinates defined

$$\tau = \frac{t}{\theta} \qquad \text{and} \qquad \sigma = \frac{T-t}{\theta}. \tag{5.12d}$$

Here the functions of τ decay to zero as $\tau \to \infty$ and the functions of σ decay to zero as $\sigma \to \infty$. The computation of the asymptotic expansion requires infinite differentiability of the coefficients A, B, Q and R [106].

For any optimal control problem with control delays application of the exogenous input control method leads to a singular optimal control problem with delays in the state variables. Note that with the exception of the state delays, (5.2) has a form similar to Eq. (5.11). Along with the removal of the delay variable the regularized simple delay test problem achieves form (5.11) by setting $\theta = \sqrt{\varepsilon}$. Recall that delay problems can often be transformed into a system of ODEs via method of steps. While that technique would work here, it is not favorable because it would increase the dimension of the system which would further increase the difficulty of finding an asymptotic solution. Thus, it is necessary to apply a technique that would have a more clean result.

Proposition 5.2.1. *The EIC test problem in (5.2) defined on* $[0,\ T]$ *can be transformed into an undelayed optimal control system defined on* $[0,\ \ell]$*, where* $\ell = T - s$ *and* $s > 0$ *is the constant delay value.*

Proof. Recall Eq. (5.2). Note that the value for $x_1(t)$ is completely determined on $[0,\ s]$ because of the startup condition for $x_2(t)$ on the delay interval. Thus, $x_1(t)$ has a fixed contribution to the cost. Secondly, because the time delay is only active on $[0,\ \ell]$ note that both $x_2(t)$ and $z(t)$ are zero on $[\ell,\ T]$, and therefore do not contribute to the cost on this interval. This can also be recognized in Figures 5.2c and 5.2e. Solving (5.2b) for $x_1(t)$ on $[0,\ s]$ we obtain $x_1(t) = t + s$. Now letting $\hat{x}(t) = x(t+s)$ we obtain an equivalent undelayed system for the regularized test problem on the shifted interval $[0,\ \ell]$. The goal of the shifted problem is to now minimize

$$\hat{J} = \frac{1}{2}\int_0^{\ell} \hat{x}^T(t)Q\hat{x}(t) + \varepsilon z_{\varepsilon}^T(t)Rz_{\varepsilon}(t)\,dt, \tag{5.13a}$$

subject to the dynamics

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bz_{\varepsilon}(t), \tag{5.13b}$$

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \text{and} \quad R = 1,$$

with the initial condition

$$\hat{x}_1(0) = x_1(s), \tag{5.13c}$$

where $\hat{x}_2(0) = x_2(s)$ is unspecified. Note that $x_2(t)$ on $[0,\ \ell]$ is equal to $x_2(t-s)$ on $[s,\ T]$. Hence, the shift in $x_2(t)$ is a "neutral" shift which causes $z_\varepsilon(t)$ to remain unchanged as well. Hence, (5.13) is a valid representation of (5.2) on $[0,\ \ell]$. $\square$

We refer to Eq. (5.13) as the undelayed regularized test problem (URTP). It is important to note that all properties of the EIC problem in (5.2) are preserved under the shift transformation. Since URTP was constructed for an analytic study the bound for the exogenous control in (5.2e) is neglected in (5.13). Here, notice that $Q \geq 0$ and $R > 0$ are symmetric matrices, and $B^TQB > 0$. Thus, with the delay removed Eq. (5.13) begins to assume a form that is valid for the application of Theorem 5.2.1. However, there is one thing. The initial value for $\hat{x}(0)$ is only partially prescribed since $\hat{x}_2(0)$ is unspecified. This is a direct consequence of the exogenous input control method. Note that after (5.13) has been solved an initial condition for $\hat{x}_2(0)$ can be obtained. Utilizing this condition, $\hat{x}(0)$ is then fully prescribed and (5.13) has a form identical to that of (5.11). Throughout the remainder of this section we take $\theta = \sqrt{\varepsilon}$, and make the appropriate substitutions in Eq. (5.13). Now by Theorem 5.2.1, URTP has an asymptotic solution of form (5.12) with the exogenous control $z(t, \theta) = u(t, \theta)$.

While it is preferred for the initial condition on the regularized controls to be left free, we note that for "large enough" $\varepsilon > 0$ the EIC method can also be efficient when the initial condition for the regularized control is given. Under these circumstances when applying the EIC method to many of the problems in our test set, we noticed that whenever an initial value for the regularized control was taken to be close to the true initial value of the original delayed control, i.e., $x_{n+i}(0) \approx u(0)$, a slightly more accurate solution could be obtained. However, as ε became sufficiently small we observed irregularities and oscillations reappearing in the solution which is the issue that we were trying to eliminate in the first place. Furthermore, since much guessing and checking would be needed to determine what taking $x_{n+i}(0)$ "close" to $u(0)$ means exactly, leaving the initial condition for the regularized control free was concluded to be the best selection from a general viewpoint.

Now, in Theorem 5.2.1 O'Malley and Jameson state that $x(0)$ is prescribed which is a very broad statement. Does this suggest that $x(0)$ is fully prescribed? Partially prescribed? Here, we interpret this statement as $x(0)$ is "sufficiently" prescribed, meaning that there are enough initial conditions to obtain an asymptotic solution via the techniques of O'Malley and Jameson. Hence, Theorem 5.2.1 is still applicable for obtaining the asymptotic solution to (5.13) as stated.

5.2.2 A Reduced Form of the Asymptotic Solution to URTP

It was stated earlier that the asymptotic solution to (5.13) has form (5.12). However, due to the convergence properties estabilished in Section 5.1.4 we point out that some of the terms may be omitted in our derived expansions. Additionally, because our boundary conditions are different, computation of the expansion terms is slightly altered. O'Malley and Jameson point out that the asymptotic solution for the optimal control in (5.12a) features the following term defined on the initial layer

$$\frac{v_0(\tau)}{\theta} = \frac{f_0(\tau)}{\theta} = \frac{e^{-C\tau}}{\theta}\, m_0(0), \tag{5.14}$$

where $m_0(0) = x(0) - X(0)$ is the initial state vector on $(0,\ \tau)$ and $C = \sqrt{R^{-1/2}B^TQBR^{-1/2}} > 0$ or a similar positive definite matrix such that $m_0(\tau)$ is a decaying solution. The authors note that because of this term the optimal control is impulsive at the initial boundary and will be unbounded at $t = 0$ as $\theta \to 0$. We note that all limits here refer to the right-hand limit since $\theta > 0$. Note that for all θ and for all $T > 0$ (5.14) converges to 1 in L^1. Hence, the function in (5.14) behaves like a delta function (shown in Figure 5.4) with the initial impulse reflecting a combination of the impulses $\delta, \delta', \ldots,\ \delta^{(N-1)}$ as $\theta \to 0$ [110].

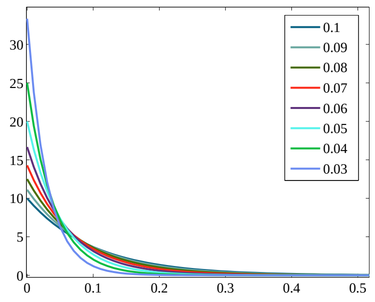


Figure 5.4: Delta function depiction of (5.14)

We investigate the behavior of (5.14) as $\theta \to 0$ with application of the L^2 norm. First recall that on the initial layer we take $\tau = t/\theta$. Then the L^2 norm of (5.14) is

$$\begin{aligned}
\left\| \frac{e^{-Ct/\theta}}{\theta} m_0(0) \right\|_2^2 &= \frac{1}{\theta^2} \int_0^\ell \left\| e^{-C\,t/\theta} m_0(0) \right\|_2^2 \, dt, \\
&= \frac{1}{\theta^2} \int_0^\ell m_0(0)^T e^{-C^T\,t/\theta} e^{-C\,t/\theta} m_0(0)\, dt, \\
&= \frac{m_0(0)^T}{\theta^2} \left(\int_0^\ell e^{-2C\,t/\theta}\, dt \right) m_0(0), \qquad (5.15a)
\end{aligned}$$

since C is symmetric. Now C is positive definite ($R > 0$ and $B^T Q B > 0$) so there exists a similarity transformation such that $C = YDY^{-1}$, where $D = [d_{ij}]$ is a diagonal matrix containing the eigenvalues of C. Additionally, since C is postive definite, the values of D are all positive. Substituting the similarity transformation into (5.15a) and evaluating the integral we find that

$$\begin{aligned}
\left\| \frac{e^{-Ct/\theta}}{\theta} m_0(0) \right\|_2^2 &= -\frac{1}{\theta} m_0(0)^T Y \left(\frac{e^{-2D\,\ell/\theta} + I}{2} \right) Y^{-1} m_0(0), \\
&= -\frac{1}{2} y^T \left(\frac{e^{-2D\,\ell/\theta}}{\theta} \right) y - \frac{1}{2} y^T y \left(\frac{1}{\theta} \right), \qquad (5.15b)
\end{aligned}$$

where $y = Y^{-1} m_0(0) \in \mathbb{R}^{n \times 1}$.

Now letting $\theta \to 0$ in (5.15b), observe that the limit of the first term has a $0/0$ indeterminant form. Note that since D is a positive diagonal matrix, the exponential matrix $e^{-2Dr/\theta}$ is just a matrix with diagonal entries $\{e^{-2d_{ii}r/\theta}\}_{i=1}^n$ that converge to 0 as $\theta \to 0$ for all i. Thus, $e^{-2Dr/\theta} \to 0_{n\times n}$. It is trivial to see that the limit of θ is 0. To obtain a valid form we apply L'Hospital's Rule and find that the limit of the first term is 0. As for the second term in (5.15b) it is also trivial to see that $\theta^{-1} \to \infty$ as $\theta \to 0$. Thus (5.15b) tends to ∞ as $\theta \to 0$. This result is consistent with O'Malley and Jameson's remark that (5.14) is an impulsive term. Thus, in general for an optimal control problem (5.11), (5.14) is a part of the $\mathcal{O}(1)$ term for the uniform asymptotic optimal control $u(t, \theta)$. However, note that URTP in (5.13) is a specialized form of (5.11). Since URTP preserves properties of the EIC problem in (5.2), by Proposition 5.1.2 we know that the optimal exogenous control in (5.13) is bounded in L^2. Thus, all parts of its governing asymptotic expansion are also bounded in L^2. Hence, we conclude that the impulsive term in (5.14) cannot appear in our expansion. Moreover, since each of the terms in the limiting solution on the initial layer are composed of this term, i.e., $[m_0(\tau),\ f_0(\tau)]$ are a product of θ^{-1}, the $\mathcal{O}(1)$ asymptotic solution on the initial layer disappears. This means that for problem URTP the outer solution satisfies the initial boundary conditions. Hence, the asymptotic solution to (5.13) has the reduced form

$$z(t,\theta) = \sum_{j=0}^{N} \left(U_j(t) + v_{j+1}(\tau) + w_j(\sigma)\right)\ \theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.16a}$$

$$x(t,\theta) = X_0(t) + \sum_{j=1}^{N} \left(X_j(t) + m_j(\tau) + n_{j-1}(\sigma)\right)\theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.16b}$$

and

$$J^*(\theta) = \frac{1}{2}\sum_{j=0}^{N} J_j^*\theta^j + \mathcal{O}(\theta^{N+1}). \tag{5.16c}$$

5.2.2.1 Computation of the Asymptotic Solution

To compute the asymptotic solution we first optimize the undelayed regularized problem and form the associating Hamiltonian system defined in the state $\hat{x}(t) = [\hat{x}_1,\ \hat{x}_2]^T$ and the costate $p(t) = [p_1,\ p_2]^T$ variables. In the remaining discussion we expand the matrix vector calculations in (5.13) to obtain a clearer view of application of O'Malley and Jameson's Asympotic Expansion Theorem. Let $z_\theta(t)$ be chosen to minimize (5.13a), then

$$z_\theta^*(t) = -\frac{1}{\theta^2}p_2, \tag{5.17a}$$

and substituting z_θ^* into the associated necessary conditions one obtains the two-point boundary value problem defined

$$\dot{\hat{x}}_1(t) = \hat{x}_2, \qquad \hat{x}_1(0) \text{ specified} \tag{5.17b}$$

$$\theta^2\dot{\hat{x}}_2(t) = -p_2, \tag{5.17c}$$

$$\dot{p}_1(t) = -\hat{x}_1, \qquad p_1(\ell) = 0 \tag{5.17d}$$

$$\dot{p}_2(t) = -\hat{x}_2 - p_1, \qquad p_2(0) = p_2(\ell) = 0. \tag{5.17e}$$

Again note that since $\hat{x}_2(0)$ is free in URTP, $p_2(t)$ is zero at the end-points as displayed in (5.17e). We seek an asymptotic solution to Eq. (5.17) of the form

$$\hat{x}_1(t,\theta) = X_1(t,\theta) + \theta m_1(\tau,\theta) + \theta^2 n_1(\sigma,\theta), \tag{5.18a}$$

$$\hat{x}_2(t,\theta) = X_2(t,\theta) + m_2(\tau,\theta) + \theta n_2(\sigma,\theta), \tag{5.18b}$$

$$p_1(t,\theta) = P_1(t,\theta) + \theta f_1(\tau,\theta) + \theta^2 g_1(\sigma,\theta), \tag{5.18c}$$

$$p_2(t,\theta) = \theta^2 P_2(t,\theta) + \theta f_2(\tau,\theta) + \theta^2 g_2(\sigma,\theta), \tag{5.18d}$$

in which each asymptotic variable is represented by a power series expansion of the form $Y(t,\theta) = \sum_{j=0}^{N} Y_{kj}(t)\theta^j,\ k = 1,\ldots,n$. The θ coefficients are present to force appropriate cancellation. They could have been omitted and zero terms would result after much effort, since the solution and its asymptotic representation in the form (5.18) are unique [109]. Computation of the asymptotic solution to the two-point BVP in (5.17) requires derivation of the governing systems on each of the three regions. Since the outer layer is defined in terms of t, the governing system of equations has a form similar to (5.17). However to obtain systems on the initial and terminal layers, a change of variables is necessary. Note that since $\theta d\tau = dt$ and $-\theta d\sigma = dt$, systems on the initial and terminal layers have the following respective forms

$$\frac{dm_1}{d\tau} = \theta m_2, \tag{5.19a}$$
$$\theta\frac{dm_2}{d\tau} = -f_2, \tag{5.19b}$$
$$\frac{df_1}{d\tau} = -\theta m_1, \tag{5.19c}$$
$$\frac{df_2}{d\tau} = -\theta(m_2 + f_1), \tag{5.19d}$$

and

$$\frac{dn_1}{d\sigma} = -n_{2j}, \tag{5.20a}$$
$$\theta\frac{dn_2}{d\sigma} = g_{2j}, \tag{5.20b}$$
$$\frac{dg_1}{d\sigma} = \theta n_1, \tag{5.20c}$$
$$\frac{dg_2}{d\sigma} = \theta(n_2 + g_1). \tag{5.20d}$$

The asymptotic solutions on the initial and terminal layers are composed of decaying exponential terms in τ and σ respectively. Hence, it is assumed that the initial layer expansions at $t = \ell$ and the terminal layer expansions at $t = 0$ are asymptotically negligible since the exponential terms at these respective values decay to 0 as θ goes to 0. Thus, the boundary conditions in (5.17) are asymptotically equivalent to

$$\hat{x}_1(0) = X_{10}(0); \qquad X_{1j}(0) = -m_{1j-1}(0), \text{ for } j \geq 1 \tag{5.21a}$$

$$\hat{x}_2(0) = X_{20}(0) + m_{20}(0); \qquad X_{2j} = -m_{2j}(0), \text{ for } j \geq 1 \tag{5.21b}$$

$$p_1(\ell) = P_{10}(\ell); \qquad P_{1j}(\ell) = -g_{1j-2}(0), \text{ for } j \geq 1 \tag{5.21c}$$

$$P_{2j}(\ell) = -g_{2j}(0), \text{ for all } j. \tag{5.21d}$$

The i^{th} order asymptotic system is then found by substituting

- $[\, X_1(t,\theta),\ X_2(t,\theta),\ P_1(t,\theta),\ \theta^2 P_2(t,\theta)\,]$,

- $[\, \theta m_1(t,\theta),\ m_2(t,\theta),\ \theta f_1(t,\theta),\ \theta f_2(t,\theta)\,]$, and

- $[\, \theta^2 n_1(t,\theta),\ \theta n_2(t,\theta),\ \theta^2 f_1(t,\theta),\ \theta^2 g_2(t,\theta)\,]$,

into the appropriate governing system, and equating like powers of θ^i, $i \geq 0$. The corresponding i^{th} order asymptotic solution is obtained by determining exponentially decaying solutions on the initial and terminal regions and combining them with the solution computed on the outer region. We compute the $\mathcal{O}(\theta)$ order asymptotic solution to (5.17). Computation details are outlined below, and are followed by a few remarks discussing the asymptotic solution.

- $\mathcal{O}(1)$ **Solution**

1. The *Outer* system is defined

$$\dot{X}_{10} = X_{20}, \qquad X_{10}(0) = \hat{x}_1(0) \tag{5.22a}$$

$$\dot{P}_{10} = -X_{10}, \qquad P_{10}(\ell) = 0 \tag{5.22b}$$

$$\dot{X}_{20} = -P_{20}, \tag{5.22c}$$

$$0 = -X_{20} - P_{10}, \tag{5.22d}$$

which can be reduced to the 2-dimensional system

$$\dot{X}_{10} = -P_{10}, \tag{5.23a}$$

$$\dot{P}_{10} = -X_{10}, \tag{5.23b}$$

with appropriate substitutions. The solution to the reduced system is

$$X_{10}(t) = c_1 e^t + c_2 e^{-t}, \tag{5.24a}$$

$$P_{10}(t) = -c_1 e^t - c_2 e^{-t}. \tag{5.24b}$$

Now applying the boundary conditions in (5.22) we determine the unknown coefficients

$$c_1 = X_{10}(0) - c_2 \quad \text{and} \quad c_2 = \frac{X_{10}(0)e^{2\ell}}{(1+e^{2\ell})}. \tag{5.24c}$$

Substituting (5.24) into (5.22) and integrating appropriately, the zeroth order asymptotic solution for the remaining terms X_{20} and P_{20} can be uniquely determined.

2. Note that the zeroth order solution for each asymptotic variable on the *Initial* region is equal to zero as a consequence of Proposition 5.1.2.

3. The *Terminal* system is defined

$$\frac{dn_{10}}{d\sigma} = -n_{20}, \tag{5.25a}$$
$$\frac{dn_{20}}{d\sigma} = g_{20}, \tag{5.25b}$$
$$\frac{dg_{10}}{d\sigma} = 0, \tag{5.25c}$$
$$\frac{dg_{20}}{d\sigma} = n_{20}, \quad g_{20}(0) = -P_{20}(\ell). \tag{5.25d}$$

Differentiating (5.25b) and (5.25d) we can derive the second order system

$$\frac{d^2 g_{20}}{d\sigma^2} = g_{20}. \tag{5.26a}$$

Note that the general solution to (5.26a) is $g_{20}(\sigma) = a_1 e^{\sigma} + a_2 e^{-\sigma}$. Recall that all solutions computed on the terminal region are required to decay exponentially. Hence, we find that an admissible solution is $g_{20}(\sigma) = a_2 e^{-\sigma}$. Utilizing the boundary conditions we find that

$$g_{20}(\sigma) = e^{-\sigma} g_{20}(0). \tag{5.26b}$$

Sequentially integrating (5.25b) and (5.25a) the solutions for $n_{20}(\sigma)$ and $n_{10}(\sigma)$ are easily obtained. From Eq. (5.25c) we see that $g_{10}(\sigma)$ is a constant solution. Taking

$$g_{10}(\sigma) = 0, \tag{5.26c}$$

the zeroth order solution on the terminal region is now fully determined.

- $\mathcal{O}(\theta)$ **Solution**

1. The *Outer* system for $j = 1$ is of the same form as (5.22) with constants c_3 and c_4 and boundary conditions $X_{11}(0) = -m_{10}(0) = 0$ and $P_{11}\ell) = -g_{1-1}(0) = 0$. Thus, choosing $c_4 = 1$ gives $c_3 = \dfrac{e^{-\ell} - 1}{1 - e^{\ell}}$ and the definition for each term of the solution follows from (5.24).

2. The $\mathcal{O}(\theta)$ system on the *Initial* layer is defined

$$\frac{dm_{11}}{d\tau} = m_{21}, \tag{5.27a}$$
$$\frac{dm_{21}}{d\tau} = -f_{21}, \quad m_{21}(0) = -X_{21}(0) \tag{5.27b}$$
$$\frac{df_{11}}{d\tau} = -m_{10}, \tag{5.27c}$$
$$\frac{df_{21}}{d\tau} = -m_{21} - f_{10}. \tag{5.27d}$$

Combining (5.27b), (5.27d) we obtain the second order system

$$\frac{d^2m_{21}}{d\tau^2} = m_{21}, \tag{5.28}$$

since $f_{10}(\tau) = 0$. The general solution to (5.28) is $m_{21}(\tau) = b_1e^{\tau} + b_2e^{-\tau}$. Similar to (5.26a), here we are to determine an exponentially decaying solution. Taking $m_{21}(\tau) = b_2e^{-\tau}$ and utilizing the boundary condition in (5.27b) we determine the following full solution on the initial layer

$$m_{11}(\tau) = -e^{-\tau}m_{21}(0), \tag{5.29a}$$
$$m_{21}(\tau) = e^{-\tau}m_{21}(0), \tag{5.29b}$$
$$f_{11}(\tau) = b_3, \tag{5.29c}$$
$$f_{21}(\tau) = e^{-\tau}m_{21}(0). \tag{5.29d}$$

where we take $b_3 = 0$ for simplicity.

3. The $\mathcal{O}(\theta)$ system on the *Terminal* region is defined

$$\frac{dn_{11}}{d\sigma} = -n_{21}, \tag{5.30}$$
$$\frac{dn_{21}}{d\sigma} = g_{21}, \tag{5.31}$$
$$\frac{dg_{11}}{d\sigma} = n_{10}, \tag{5.32}$$
$$\frac{dg_{21}}{d\sigma} = n_{21} + g_{10}, \quad g_{21}(0) = -P_{21}(\ell). \tag{5.33}$$

Similar to (5.26a), we can determine a second order system in terms of $g_{21}(\sigma)$ since $g_{10}(\sigma) = 0$. Utilizing the boundary condition in (5.33), the exponentially decaying solution is computed as

$$n_{11}(\sigma) = e^{-\sigma} g_{21}(0), \tag{5.34a}$$

$$n_{21}(\sigma) = -e^{-\sigma} g_{21}(0), \tag{5.34b}$$

$$g_{11}(\sigma) = -e^{-\sigma} g_{20}(0), \tag{5.34c}$$

$$g_{21}(\sigma) = e^{-\sigma} g_{21}(0). \tag{5.34d}$$

In order to complete the $\mathcal{O}(\theta)$ asymptotic solution for the control, note that $v_2(\tau)$ is needed. Computation of this term requires knowledge of the $\mathcal{O}(\theta^2)$ terms $X_{22}(t)$ in the outer layer and $m_{22}(\tau)$ in the initial layer. Computation of higher order approximations are obtained in an analogous manner as outlined where the solutions on the initial and terminal regions are exponentially decaying solutions. We find that

$$X_{22}(t) = \int_0^{\ell} \left(c_5 - t\frac{c_1}{2}\right) e^{t} + \left(c_6 + t\frac{c_2}{2}\right) e^{-t} \, dt, \tag{5.35}$$

where c_5 and c_6 are determined from the boundary conditions $X_{12}(0) = -m_{11}(0)$ and $P_{12}(\ell) = -g_{10}(0)$ and

$$m_{22}(\tau) = \frac{m_{21}(0)\tau}{2} e^{-\tau} - m_{22}(0)e^{-2\tau}, \quad m_{22}(0) = -X_{22}(0) \tag{5.36}$$

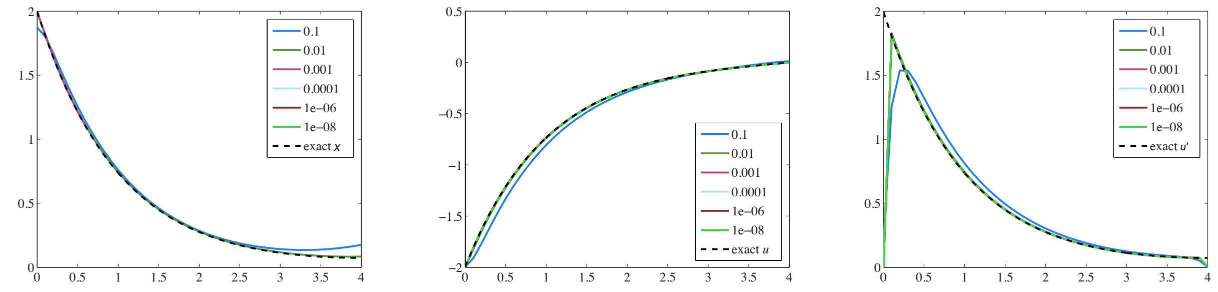
$$v_2(\tau) = \int_0^{\ell} m_{22}(\tau) \, d\tau. \tag{5.37}$$

After computation of $v_2(\tau)$, applying (5.16) we determine the full $\mathcal{O}(\theta)$ asymptotic solutions for $\hat{x}(t)$ and $z_\theta(t)$ to (5.17) on the interval $[0,\ \ell]$.

5.2.2.2 Remarks About the Computed Asymptotic Solution

The asymptotic solution is plotted in Figure 5.5 for various θ. Additionally, the optimal OMOS true solutions in Figure 5.1 are replotted here as the black dashed line for comparison. Observe in

Figures 5.5a and 5.5b that $\hat{x}_1$ and $\hat{x}_2$ respectively converge uniformly to the true solution $x(t)$ and $u(t)$ as $\theta \to 0$ on $[0,\ \ell]$. The exogenous control, $z(t)$ converges to $\dot{u}(t)$ uniformly on compact subsets within $(0,\ \ell]$ as $\theta \to 0$. Note that we do not have uniform convergence over the entire interval because of the peak at the initial value $t = 0$. It is important to note that with the exclusion of the zeroth order initial layer terms that the asymptotic optimal control is bounded at the initial boundary as $\theta \to 0$ as required by Proposition 5.1.2. The convergence properties of the asymptotic solution to the shifted EIC problem (5.13) is consistent with the convergence results featured in Section 5.1.



(a) State $\hat{x}_1(t)$ (b) Regularized control $\hat{x}_2(t)$ (c) Control $z_\theta(t)$

Figure 5.5: $\mathcal{O}(\theta)$ order asymptotic solution for URTP in Eq. (5.13)

In this discussion we have mentioned very little about the computation of the cost in (5.12c). For any problem (5.11) the optimal cost is given by

$$J^*(\theta) = \sum_{j=1}^{N} x_j(t,\theta)^T Q_{jj} x_j(t,\theta) + \frac{1}{\theta^2} p_2^T(t,\theta) R^{-1} p_2(t,\theta) + 2\sum_{i=0}^{N}\sum_{j=0}^{N} x_i^T(t,\theta) Q_{ij} x_j(t,\theta). \quad (5.38)$$

Substituting the asymptotic solution in (5.18) into the above equation, the equivalent asymptotic expansion in (5.12c) can be obtained. The optimal values for the $\mathcal{O}(1)$ and $\mathcal{O}(\theta)$ asymptotic performance index associated with (5.18) evaluated for various θ are shown in Table 5.1. For larger θ notice how the higher order term is affected by the boundary layer corrector values. However, as θ goes to zero the boundary layer terms decay to zero causing the higher order cost to approach the limiting optimal cost. The computed method of steps cost is presented for comparison.

Table 5.1: $J^*(\varepsilon)$: Performance index for (5.17)

ε	$\mathcal{O}(1)$	$\mathcal{O}(\varepsilon)$	MOS
1.0E-01	2.0087	2.2962	2.0081
1.0E-02	1.9988	2.0301	1.9999
1.0E-03	1.9987	2.0018	1.9998
1.0E-04	1.9987	1.9990	1.9998
1.0E-06	1.9987	1.9987	1.9998
1.0E-08	1.9987	1.9987	1.9998

5.2.3 Analysis Summary

In this section we have provided analytic answers that support our numerics relative to the convergence of the exogenous input control solutions to the solutions of the original control delay problem. As a study, we applied two analytic methods to determine the solutions to the simple control delay regularized test problem in Eq. (5.2). Application of the MOS approach displayed convergence of the EIC solution to the original control delay solutions. Additionally, convergence results established some useful principles regarding the exogenous control and the adjoint variable associated with the new state variable.

With the asymptotic expansions approach we were able to derive an asymptotic solution for the shifted problem in (5.13) which is also an asymptotic solution to (5.2) on $[0,\ \ell]$. The problem was shifted by s in the state variable in order to remove the delay and apply the O'Malley and Jameson Asymptotic Expansion Theorem. Results in Figure 5.5 show that the asymptotic solutions are converging to the original solution as $\theta \to 0$, and more specifically as $\varepsilon \to 0$ since $\varepsilon = \theta^2$. We note that the asymptotic solution obtained is of order $\mathcal{O}(\theta)$. It is expected that computation of higher order terms will lead to a more accurate asymptotic expansion. For a more in depth discussion of asymptotic solutions to the cheap singular control problem using O'Malley and Jameson's approach see [108, 109, 110].

The study outlined in this section serves as a blue print for linear time invariant quadratic regulator (LQR) problems with similar dynamics to (5.2). Note that the propositions and derivations stated in this section hold for any optimal control problem with form similar to (5.2). All results are independent of the system coefficients, time interval, and delay values. While results were established for a 2-dimensional optimal control problem containing a single control delay variable, it is determined that results can be extended to problems of higher dimension. So long as the control delay function is constant, the regularized problem can be represented by an equivalent undelayed problem that is shifted in $x(t)$ by the delay value. The shifted problem preserves the perturbed form, and also inherits the characteristics and properties of the original system. Thus, the shifted problem can be used to obtain an asymptotic solution to the regularized problem (by Theorem 5.2.1) that converges to the solution to original problem. Based on the results it turns out that application of the EIC method is not only useful for obtaining a close approximation to a cheap LQR optimal control problem, but also leads to some very useful consequences that reduce the difficulty in applying analysis to study the behavior of the system. We summarize our conclusions in the following generalized theorems.

Lemma 5.2.2 (Exogenous Input Control Lemma). *Consider an optimal control delay problem with the goal to minimize the performance index*

$$J = \frac{1}{2}\int_0^T x^T Q x + u^T R u \, dt, \tag{5.39a}$$

subject to the delay system

$$\dot{x}(t) = Ax(t) + Bu(t-s), \tag{5.39b}$$

where $s > 0$ is a constant delay value, $x \in \mathbb{R}^n$, $u \in \mathbb{R}^q$, with $q < n$ and the initial conditions

$$x(0) = \alpha, \tag{5.39c}$$

and startup functions for the delayed controls defined

$$u(t) = \beta(t), \qquad -s \le t < 0. \tag{5.39d}$$

Letting y=u(t), and

$$\bar{x}(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}, \quad \bar{A} = \begin{bmatrix} A_{n\times n} & B_{n\times q} \\ 0_{q\times n} & 0_{q\times q} \end{bmatrix}, \quad \bar{B} = \begin{bmatrix} 0_{n\times 1} \\ I_{q\times 1} \end{bmatrix}, \quad \bar{Q} = \begin{bmatrix} Q_{n\times n} & 0_{n\times q} \\ 0_{q\times n} & R_{q\times q} \end{bmatrix}, \quad \textit{and} \quad \bar{R} = 1,$$

by application of the EIC method in Algorithm 1 there exists an equivalent higher dimension state delay optimal control problem with the objective defined by

$$min \quad \bar{J} = \frac{1}{2}\int_0^T \bar{x}^T \bar{Q}\bar{x} + \varepsilon z_\varepsilon^T \bar{R} z_\varepsilon \, dt, \tag{5.40a}$$

subject to the delay equations

$$\dot{\bar{x}}(t) = \bar{A}\bar{x}(t) + \bar{B}z_\varepsilon(t), \tag{5.40b}$$

where $z \in \mathbb{R}^q$. Suppose that $\dot{u}^*$ is piecewise smooth where u^* is obtained from (5.39). Then

$$\|z_\varepsilon\|_2^2 \leq \|\dot{u^*}\|_2^2, \tag{5.40c}$$

by Proposition 5.1.2. The initial condition is

$$\bar{x}(0) = [x(0),\ y(0)]^T, \tag{5.40d}$$

where $\bar{x}_2(0) = y(0)$ is unspecified and the startup functions for the regularized delayed controls are defined

$$y(t) = \beta(t), \qquad\qquad -s \leq t < 0. \tag{5.40e}$$

Theorem 5.2.2 (Exogenous Input Control Asymptotic Expansion Theorem). *Define*

$$\hat{x}(t) = \bar{x}(t+s),$$

in (5.40) and make the assumption of Lemma 5.2.2, then for every exogenous input control problem with form (5.39) by Proposition 5.2.1 there exists an equivalent undelayed optimal control problem over the shifted interval $[0,\ T-s],\ s>0$. *The goal is to minimize*

$$\hat{J} = \frac{1}{2}\int_0^{T-s} \hat{x}^T\bar{Q}\hat{x} + \varepsilon z_\varepsilon^T \bar{R} z_\varepsilon \, dt, \tag{5.41a}$$

subject to the differential equations

$$\dot{\hat{x}} = \bar{A}\hat{x} + \bar{B}z_\varepsilon, \tag{5.41b}$$

where

$$\|z_\varepsilon\|_2^2 \leq \|\dot{u^*}\|_2^2, \tag{5.41c}$$

and the initial condition is

$$\hat{x}(0) = \bar{x}(s), \tag{5.41d}$$

with $\hat{x}_2(0) = y(0)$ *is unspecified. If* $\bar{Q} \geq 0$ *and* $\bar{R} > 0$ *are symmetric matrices with* $\bar{B}^T\bar{Q}\bar{B} > 0$, *then by setting* $\theta = \sqrt{\varepsilon}$ *Theorem 5.2.1 gives existence of an asymptotic solution to (5.41) of the form*

$$z(t,\theta) = \sum_{j=0}^{N}\left(U_j(t) + v_{j+1}\left(\frac{t}{\theta}\right) + w_j\left(\frac{T-t}{\theta}\right)\right)\theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.42a}$$

$$x(t,\theta) = X_0(t) + \sum_{j=1}^{N}\left(X_j(t) + m_j\left(\frac{t}{\theta}\right) + n_{j-1}\left(\frac{T-t}{\theta}\right)\right)\theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.42b}$$

and

$$J^*(\theta) = \frac{1}{2}\sum_{j=0}^{N} J_j^*\theta^j + \mathcal{O}(\theta^{N+1}), \tag{5.42c}$$

that converges to the true solution of (5.39) on $[0, T-s]$ *as* $\theta \to 0$.

Chapter 6

$\mathbb{SOCX}$, DAEs, and More General Delay Problems

The primary focus of this thesis was contributed to the study of the solutions to optimal control systems with time delays in the state and control variables with $\mathbb{SOCX}$, a direct transcription optimization software tool. Not only can $\mathbb{SOCX}$ solve delay optimization problems, but it serves as a general purpose tool for the modeling and simulation of various types of differential algebraic equation systems. Many physical systems are most naturally modeled by DAEs, that is mixed systems of differential and algebraic equations [32]. These systems can be found in a wide variety of scientific and engineering applications, including circuit analysis, computer-aided design and real-time simulation of mechanical (multibody) systems, power systems, chemical process simulation, and optimal control [40]. Thus, having software that can work with DAEs is extremely useful.

The degree of difficulty for solving a DAE system is often classified by its **index**, the number of times differentiation is necessary to obtain a system of ODEs. Differentiation is usually carried out with respect to the independent variable. Many of the existing numerical algorithms are constructed to compute the solutions to index-1 DAE systems. In the presence of higher index DAEs the software will often either terminate or apply index reduction techniques until an index-2, or more preferably an index-1 system is reached. All of the classical discretizations of DAEs only converge for index three or less and also require that the DAE have special structure if the index is greater than one [49]. In contrast, depending on the optimal control, cost function, and form of the dynamics and the constraints $\mathbb{SOCX}$ has shown that it can sometimes produce an approximation for higher index DAEs.

$\mathbb{SOCX}$'s ability to handle difficult DAE systems gives it the capability to solve various types of time delay systems. Recall that $\mathbb{SOCX}$ automatically transforms any delay problem into another DDAE. If there is no cost function to optimize, then $\mathbb{SOCX}$ becomes a numerical integrator. However, it performs as a boundary value solver for a problem with initial conditions. While $\mathbb{SOCX}$ does not permit a given term to switch from being delayed to being advanced (or from advanced to delayed), it does have the ability to treat systems with variables that are simultaneously advanced and delayed. In this section we extend our discussion on solving time delay systems with $\mathbb{SOCX}$ with a discussion on the solutions to time advanced, mixed-type, neutral, and time-varying delay systems. The difficulties that each of these type of systems carry is well documented in the literature. Each problem is solved using the default TR to HS discretization. This section aids in displaying the flexibility and versatility of direct transcription algorithms to provide numerical solutions to different types of delay problems commonly found in industrial and engineering applications. The ability to work with DAEs is key in this chapter.

6.1 Advanced Time Systems

Advance differential equations (ADEs) are less common than DDEs; however, they do occur. Advanced arguments appear in mathematical models in economics to reflect the dependency on anticipated capital stock [47, 84]. Like equations with time delays, ADEs are typically unstable with infinitely many poles. While linear DDEs have infinitely many poles on the left half of the complex plane, linear ADEs have infinitely many poles on the right half of the complex plane, and are therefore always strongly unstable [78]. Because of the impulsive nature of ADEs many authors have dedicated research to the existence of oscillatory and nonoscillatory solutions for these types of problems [1, 135, 149]. The existing methods that rely on numerical integrators have trouble dealing with advances in the states and often in the control. In principle, since direct transcription discretizes the entire problem and then uses sparse solvers, it should not matter whether there is a delay or an advance in the dynamics.

ADEs are less discussed in the literature primarily because they inherit the properties of delay differential equations, i.e., ADEs can be formulated as DDEs and DDEs can be formulated as ADEs. To demonstrate this fact we solve an OCDP in [96] as an optimal control advance problem. The goal is to minimize the quadratic performance index

$$I = \frac{1}{2}\int_0^5 10x_1^2 + x_2^2 + u^2 \; dt, \tag{6.1a}$$

subject to the dynamics

$$\dot{x}_1(t) = x_2(t), \tag{6.1b}$$

$$\dot{x}_2(t) = -10x_1(t) - 5x_2(t) - 2x_1(t-\tau) - x_2(t-\tau) + u(t), \tag{6.1c}$$

with the initial conditions defined

$$x_1(t) = x_2(t) = 1 \text{ if } t \in [-\tau,\ 0]. \tag{6.1d}$$

To convert (6.1) into an advance time system let $z(t) = x(5-t)$ and $v(t) = u(5-t)$, then we have

$$z'(t) = x'(5-t)(-1) = -x'(5-t),$$

and then substitution yields the following DDE system

$$\begin{aligned}\dot{x}_1(5-t) &= -x_2(5-t),\\ \dot{x}_2(5-t) &= 10x_1(5-t) + 5x_2(5-t) + 2x_1(4-t) + x_2(4-t) - u(5-t).\end{aligned}$$

However, note that

$$\begin{aligned}z(t+1) &= x(5-(t+1)) = x(4-t),\\ z(5) &= x(5-5) = x(0),\\ z(6) &= x(5-6) = x(-1).\end{aligned}$$

Utilizing the above result, we form the equivalent optimal control advanced problem with the goal to minimize

$$\bar{I} = \frac{1}{2}\int_0^5 10z_1^2 + z_2^2 + v^2 \, dt, \tag{6.2a}$$

subject to

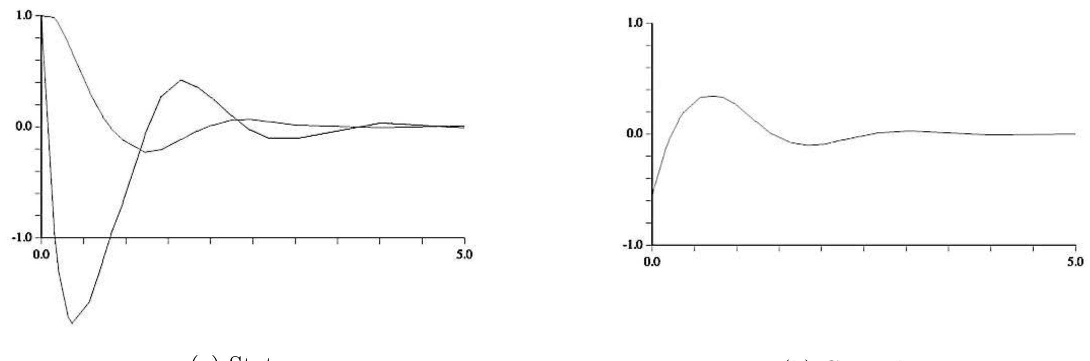
$$\dot{z}_1(t) = -z_2(t), \tag{6.2b}$$

$$\dot{z}_2(t) = 10z_1(t) + 5z_2(t) + 2z_1(t+1) + z_2(t+1) - v(t), \tag{6.2c}$$

with post-history values defined

$$z_1(t) = z_2(t) = 1 \text{ if } t \in [5, 6]. \tag{6.2d}$$

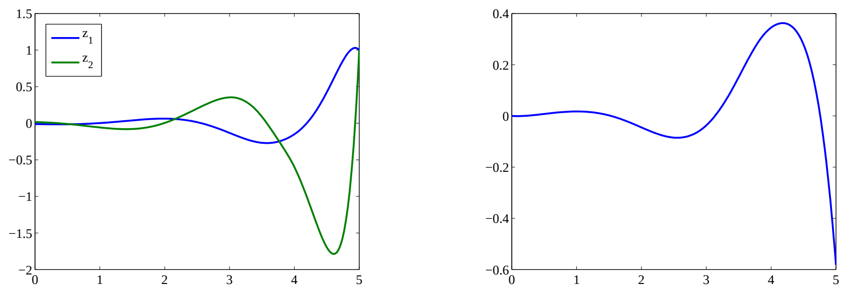
Figure 6.1 features the optimal solutions to (6.1) that were computed using an evolutionary algorithm, and are documented in [130]. The solutions are reported here for comparison. In Figure 6.2 the $\mathbb{SOCX}$ solutions are reported.



(a) States (b) Control

Figure 6.1: Evolutionary method optimal solutions for Eq. (6.1)

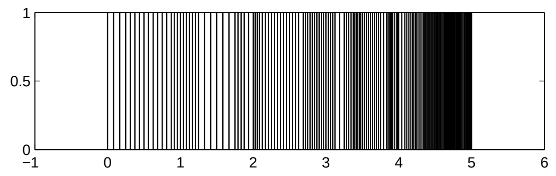
$\mathbb{SOCX}$ successfully solved (6.2) in five iterations in approximately 0.2 seconds. The initial grid



(a) State $z(t)$ (b) Control $v(t)$

Figure 6.2: $\mathbb{SOCX}$ solution for the ADE problem in Eq. (6.2)

contained 21 points, and the final grid contained 161 points. In comparing Figure 6.1 to 6.2 the advanced problem solutions appear to be the delay problem solutions from right to left as expected. Note that the solution changes more rapidly on [3, 5] in Figure 6.2 or on [1, 3] for (6.1). Notice in Figure 6.3 the darker blocks on this interval. This is an indication of additional points introduced by the mesh refinement algorithm. From this test problem (and a few others in our test set) it appears that $\mathbb{SOCX}$ can efficiently solve state time advance optimal control problems. However, since control advances are also backward control delays the chattering issue with control delays is also an issue for control advances. It is hypothesized that application of the exogenous input control method to time advance problems will work in a similar manner. This hypothesis has not yet been tested, and is left as a future study.



(a) State $z(t)$

Figure 6.3: $\mathbb{SOCX}$ final grid for Eq. (6.2)

6.2 Mixed-type Delay Systems

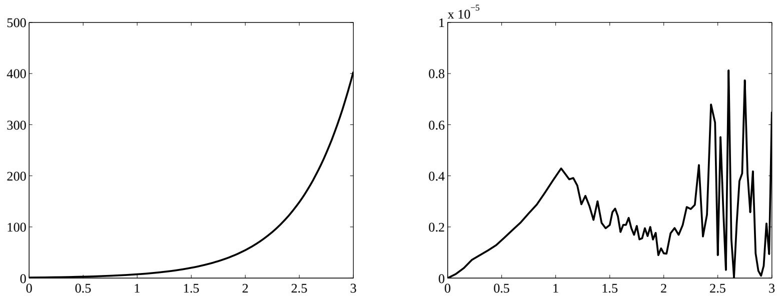
Mixed-type or forward-backward systems contain both time delay and time advance arguments. In the presence of state delays, these type of systems occur naturally after optimality principles are applied due to the involvement of the advanced costate variable in the costate equations. In a practical sense, mixed-type delay (MTD) systems are attractive for their ability to simultaneously model deviating components. A balance with applications is provided through a number of papers dealing with problems involving singularities, impulsive systems, traveling waves, climate modeling, and economic control [75]. As an illustration we solve one example from [94] which is

$$\dot{x}(t) = (m - 0.5e^{-m} - 0.5e^{m})x(t) + 0.5x(t-1) + 0.5x(t+1), \tag{6.3a}$$

with the following boundary conditions

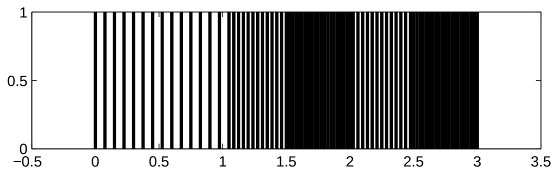
$$\begin{cases} \phi_1(t) = e^{mt}, & t \in [-1,\ 0] \\ f(t) = e^{mt}, & t \in (k-1, k] \end{cases}, \tag{6.3b}$$

where $m \in \mathbb{R},\ m \neq 0$. The exact solution is $x(t) = e^{mt}$. The problem was solved for $(m, k) = (2, 3)$.



(a) State $x(t)$ (b) Error

Figure 6.4: $\mathbb{SOCX}$ solution for MTD problem in Eq. (6.3)



(a) State $x(t)$

Figure 6.5: Final grid for MTD problem in Eq. (6.3)

In Figures 6.4 and 6.5 are the solution and absolute solution error and the final grid for (6.3). Starting with a uniform grid of 41 points execution terminated in four iterations with a final grid of 81 points. Because the solution increases rapidly on the latter parts of the interval, as to be expected the generated grid is much denser to the right as shown in Figure 6.5. In Figure 6.4b the approximated solution is compared to the analytic solution. Note that this error corresponds to an absolute error smaller than 10E-05.

6.3 Neutral Delay Systems

Neutral delay equations (NDEs) are differential equations in which the highest state derivative and corresponding state both depend on past time values. It is well known that time delays can introduce discontinuities into the system. This increases the difficulty in being able to solve these type of systems since derivatives may be discontinuous as well. Many numerical solvers for continuous-time systems assume that the solution is continuous. However, for neutral delay differential equations the right-hand side can be multi-valued, when one or several delayed arguments cross a **breaking point** [69]. A breaking point is the first time value in which a time delay variable becomes zero. In the event that a breaking point is encountered, many codes will typically stop the integration at such a breaking point with the message that too small step sizes are needed [70]. NDEs occur in a wide variety of disciplines. Thus, the ability to efficiently compute the solutions to NDE problems are important.

We solve a nonlinear neutral delay model from [146] that is often used to model species growth, the spread of epidemics, and the dynamics of capital stocks. The delayed state derivative on the left-hand side can be viewed as the change in the system according to its past growth rate, or the relapse of the infectious disease. The problem is defined

$$\frac{d}{dt}\left(x(t) - \frac{1}{2}x(t-r)\right) = \cos(4t)x(t) + 2\sin(4t)x(t-r) - 4\left(x(t) - \frac{1}{2}x(t-r)\right)^2, \tag{6.4a}$$

with the following startup function and initial condition

$$x(t) = 1, \qquad -r \le t \le 0 \tag{6.4b}$$

$$x(0) = 1. \tag{6.4c}$$

where the final time is $T = 4$ and the state delay value is $r = 0.3$. To solve the problem we must first regularize the problem by removing the derivative of the delayed state variable. We rewrite the problem as a DDAE by utilizing a change of variables by setting $z(t) = x(t) - \frac{1}{2}x(t-r)$. In its new form (6.4) becomes

$$\dot{z}(t) = \cos(4t)x(t) + 2\sin(4t)x(t-r) - 4z^2, \tag{6.5a}$$

subject to the algebraic equality constraint

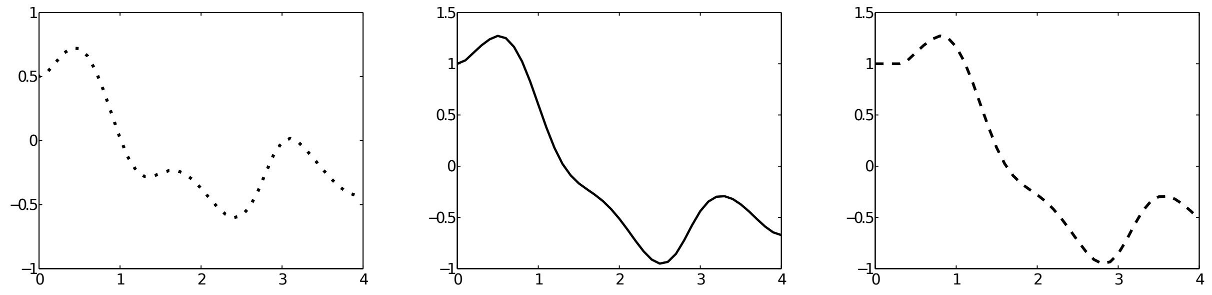
$$0 = z(t) - x(t) + \frac{1}{2}x(t-r), \tag{6.5b}$$

with the startup function and initial condition defined

$$x(t) = 1, \qquad -r \le t \le 0 \tag{6.5c}$$

$$z(0) = \frac{1}{2}. \tag{6.5d}$$

Because this problem was one of the first neutral delay problems solved with $\mathbb{SOCX}$ we also solved the problem using a method of steps Matlab code. What we found to be very interesting is in using this approach the code would terminate prematurely. On the interval [1.5, 1.6] the solution appears to be unstable, and Matlab reports a singular Jacobian error. Mathematically, the system is approaching a breaking point which is causing the solution to be inconsistent. Note that $x(t-0.3) \approx 0$ near $t = 1.5$.



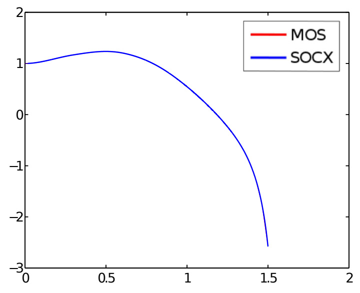
(a) $z(t)$ (b) $x(t)$ (c) $x(t-0.3)$

Figure 6.6: $\mathbb{SOCX}$ solutions for NDE problem in Eq. (6.5)

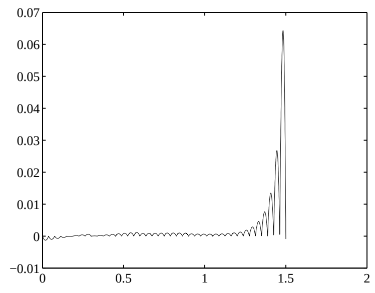
Since a solution was not able to be computed on the entire interval for MOS in Figure 6.7 we compare the $\mathbb{SOCX}$ solution to the Matlab MOS solution on $[0,\ 1.5]$. While they appear very close, note that the error between the two solutions drastically increases as the solution nears 1.5. The large error is due to the instability of the Matlab MOS solutions. Recall that MOS assumes that the solution is continuous across the internal boundaries of the time domain. The presence of the breaking point causes the continuity assumption to fall apart. However, $\mathbb{SOCX}$ produced a solution on the full interval with no problem. This displays an advantage of direct transcription algorithms. Because of the relaxed continuity requirements of direct transcription methods and the grid refinement procedure $\mathbb{SOCX}$ is able to avoid the side effects of singularities.

6.4 Time-Varying Delay Systems

In the past decade much attention has been given to time-varying delays (TVDs) for their ability to characterize the realistic motion and speed of moving targets and varying network processes. Time-varying delays have become very useful in pursuit-evasion applications such as air combat. Inclusion of time-varying delays have improved modeling of missile/target interception and prediction of the miss distance in missile defense scenarios [127]. Other applications are related to communication and teleoperations [43, 139], congestion control [103], and neural networks [72]. Time-varying delays have



(a) Solutions



(b) Error

Figure 6.7: Matlab MOS and $\mathbb{SOCX}$ solutions and error for Eq. (6.5) plotted on [0, 1.5]

received less attention, possibly due to the perceived difficulty of the problem—changes in delay make the system's state space vary with time, which complicates the use of standard analysis tools [112].

Consider the following nonlinear control time-varying delay problem from [11],

$$\dot{x}_1(t) = x_2(t) - x_2(t)^2 u(\phi(t)), \tag{6.6a}$$

$$\dot{x}_2(t) = u(\phi(t)), \tag{6.6b}$$

$$u(t) = -x_1(t) - 2x_2(t) - \frac{1}{3}x_2^3, \tag{6.6c}$$

for $t \in [0,\ 8]$ where

$$u(\phi(t)) = 0, \qquad \phi(t) = t - \frac{1+t}{1+2t} \leq 0. \tag{6.6d}$$

The $\mathbb{SOCX}$ solutions to (6.6) is plotted in Figure 6.8. The problem was solved on an initial grid of 21 points, and terminated with a final grid of 400 points with a total computation time of approximately 35 seconds. Note that (6.6) is not an optimization problem. Here, u is a known nonlinear state feedback with time delay. Thus, control delays are not affected by the algorithm during the interpolation process, and the EIC method is not needed. Note the non-chattery consistent solution for the control in Figure 6.8c. In Figure 6.9 the final grid is displayed. Note that the grid is denser on early parts of the interval where larger variations in the solution take place. The look-back strategy in $\mathbb{SOCX}$ gives it the capability

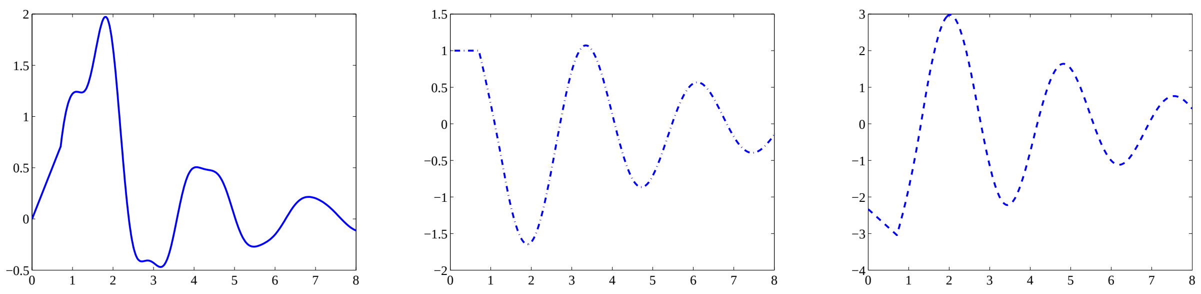

(a) $x_1(t)$ (b) $x_2(t)$ (c) $u(t)$

Figure 6.8: $\mathbb{SOCX}$ solutions for TVD problem in Eq. (6.6)

to handle time-varying delays. As the delay value changes the algorithm looks back to the appropriate location to compute the value of the delayed term.

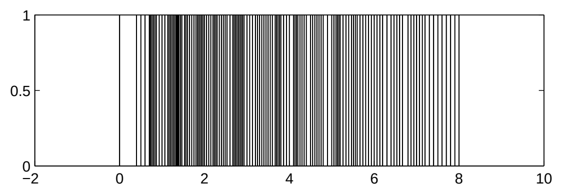


Figure 6.9: Final grid for TVD problem in Eq. (6.6)

Chapter 7

Conclusion

Direct transcription is a popular direct method frequently used to solve complex problems in industry. Currently, there are very few optimal control delay packages that apply direct transcription through the use of Runge-Kutta methods. In this thesis we have shown that Runge-Kutta methods can be used to successfully solve optimal control delay problems, delay differential algebraic equations, as well as other optimization systems through experimentation with $\mathbb{SOCX}$, an implicit RK direct transcription optimization package. $\mathbb{SOCX}$ is part of Sparse Optimization Suite available from Applied Mathematical Analysis [26]. To solve delay equations $\mathbb{SOCX}$ uses the implicit 2^{nd} order trapezoid method and/or the implicit 4^{th} order Hermite-Simpson method as integrators along with an interpolation scheme selected based on the integrator to approximate the solutions of delayed variables.

The DT algorithm in $\mathbb{SOCX}$ could be applied in a straightforward manner to approximate the solutions to optimal control problems with state delays. However, there were a few intrinsic difficulties on nonuniform grids when using this approach on optimal control delay systems with control delays. The solutions of delayed control variables displayed chattering behavior on nonuniform grids. In this thesis we have shown that issues with nonuniform grids were due to free control variables being introduced into the NLP by the discretization and interpolation scheme. What we have neglected to state in this thesis are the many challenges that we faced with trying to resolve this issue. In [21] we address the presence of chatter in control delay variables by exploring cost functional design. For example, we replaced the control $u(t)$ with the delayed control $u(t-1)$. We tried parameterizing the startup function for the delayed control. We tried replacing the control with the delayed control and the control with an advanced control. We tried several other things, but modifications revealed inconsistent results whenever the problem was formulated differently.

Fortunately, by the time my oral preliminary exam came around we were able to develop a regularization technique, called the exogenous input control method. The method was created to address the difficulties that $\mathbb{SOCX}$ currently has with the solutions to control delay optimal control problems on nonuniform grids. In the method the delayed state and exogenous control variables are introduced as pairs. All delayed controls are replaced by delayed states, and all newly added state equations have right-hand side functions equal to the exogenous control. Various right-hand side functions, $f(z)$ were tested until deciding that the linear function was the best choice. Additionally, the exogenous controls are weighted in the objective function by a small parameter ε, and are constrained by the minimum and maximum derivatives of the original delayed control variable. Note that the constraints on the exogenous controls are optional, and are only imposed numerically to help the software. Here, exogenous means that the control originated outside of the original system, and is in other words disjoint from the original system. Computational examples suggested that the EIC method could yield a "more smooth" and better approximation than that produced by the $\mathbb{SOCX}$ algorithm alone. In Section 4.2 we demonstrated this by resolving examples featured in Section 3.2.2.

Numerically, the advantageous piece of the EIC method is the parameter acting on the exogenous control in the cost function. Regulation of this parameter enabled us to easily change the cost function to yield the desired solution. Analytically, the disadvantageous piece of the EIC method is the parameter acting on the exogenous control in the cost function. In most cases "best" theoretical solution was achieved when $\varepsilon = 0$, which is no surprise because then the original and regularized problems have the same objective function as noted in Eq. (5.7). Because the EIC method performed well numerically, we wanted to investigate its performance appropriately through the use of analysis. However, when $\varepsilon = 0$ challenges arise because the system becomes singular and classical techniques cannot be applied. Fortunately, we were able to apply some of the existing techniques proven to analytically study the behavior of the EIC solutions.

In Chapter 5, we established the conditions for which the mathematical solution of the EIC formulation converges to the desired solution in an appropriate sense. Analysis was carried out on a simple control delay test problem, whose results serve as a basis for future analysis of more complex problems. To analytically study the behavior of the regularized solutions as $\varepsilon \to 0$ we first applied a combination of method of steps and Pontryagin's minimum principle. MOS was used to construct an undelayed version of the original system, and then PMP was applied to obtain optimality principles. Results in Section 5.1.3 suggested that the EIC solutions were converging to the true solution of the simple control delay test problem as $\varepsilon \to 0$. This was very meaningful to our work because we finally had some analytic evidence to support our numerical conclusions. Additionally, we discovered that application of EIC yielded some special consequences regarding the exogenous control and the associated Lagrange multiplier. It was shown that for any LQR problem with a single control delay variable that application of the exogenous input control method leads to a state delay problem with an exogenous control that is bounded in L^2 as noted in Proposition 5.1.2 and stated more formally in Theorem 5.2.2. This result was another key discovery which helped us to further analyze the EIC solutions using a singular perturbations technique.

Positive results with the MOS approach fueled the need to deal with the resulting singular control problem directly. We selected a well-known approach developed by Robert E. O'Malley and Jameson in [108, 109, 110]. O'Malley and Jameson derive an asymptotic solution to the perturbed state regulator problem defined in Eq. (5.11). This problem is free of delay variables. We note that there are a few methods that approximate the solutions of optimal control delay problems directly. In [63], the author obtains a blockwise estimate of the fundamental matrix to a linear differential system with small delay. Additionally, a minimizing sequence for a stochastic linear-quadratic optimal control problem with state delays is constructed by using a regularization approach and singular perturbation technique as outlined in [64]. Many of these techniques are fairly new. Thus, in order to apply more standard singular perturbation techniques the delay must be removed.

After some manipulation we were able to achieve an undelayed form of our test problem in (5.13), and were able to apply the O'Malley and Jameson Asymptotic Expansion Theorem outlined in Theorem 5.2.1 to obtain an asymptotic solution to our problem. Here we take $\theta = \sqrt{\varepsilon}$. The convergence of the asymptotic solution to the test problem is consistent with our numerical observations as well as the results from the MOS approach. With analysis, we are able to conclude that for LQR problems with a single control delay the exogenous input control method produces a regularized problem whose solution both numerically and analytically converges to the solution of the original optimal control delay problem. Hence, the EIC problem is a sufficient alternative form of the original problem. It is important to note that although the EIC method was developed to regulate control delay systems in $\mathbb{SOCX}$, it is general enough to be applied and tested in other software packages that are also experiencing computation difficulty with interpolation of delayed variables on nonuniform grids.

Furthermore, we discussed $\mathbb{SOCX}$'s ability to handle complex differential algebraic equation systems and other time delay systems in Section 6. Because of the tremendous effort in numerical analysis of DAE systems, DAEs have become powerful instruments for the mathematical modeling of real-life systems in a variety of scientific applications. DAEs can provide an alternative structure to time delay systems which results in a simpler structure that's more suitable for most software packages. The delay term may possess various forms to accommodate the physical characteristics of the modeled system. In addition to constant delays some other formats for delays are time-varying, state dependent, neutral, and time advance. The choice of delay highly impacts the behavior of the optimal control delay problem, and increases solving difficulty both analytically and numerically. We solved a few examples to display some alternative ways that a direct transcription algorithm with a DAE approach can be used to solve different delay problems.

7.1 Future Work

In extending optimization algorithms to accommodate delayed terms, it has been shown that Runge-Kutta direct transcription algorithms with appropriate interpolation schemes are suitable for obtaining the solutions to optimal control delay problems. The algorithm embedded in $\mathbb{SOCX}$ is effective for solving state delay problems, and with implementation of the exogenous input control method solutions to control delay problems can be obtained as well. For the EIC method we have provided interesting numerical results, and for a special class of LQR problems we have provided some analytic results. Although useful results have been obtained, a few open questions remain for future study.

Analysis: EIC Method, Nonlinear Problems, and Nonquadratic Costs

While it is suggested that the exogenous input control method is analytically a sufficient method for approximating the solutions to LQR optimal control problems with delayed controls, the results of this study motivate the investigation of the convergence of the EIC solutions of more complicated problems. Numerically, it is suggested that the EIC solutions of nonlinear optimal control delay problems converge to the true solution as indicated by the results for the regularized CSTR problem featured in Section 4.2.1.3 (and other problems in our test set). However, note that each of the problems presented in this thesis have a quadratic performance index. This was done primarily for consistency. While this is pretty standard in optimization applications, more intricate cost functions are sometimes needed to achieve the desired control.

Recall the general formulation for optimal control delay problems in (2.2). Applying the EIC method to (2.2) with $y(t) = u(t)$ results in the following formulation

$$\tilde{J} = \phi[x(T), y(T), T] + \int_{t_0}^{T} \mathcal{L}[x(t), y(t), x(\omega(t)), y(\eta(t)), t] + \varepsilon \|z_\varepsilon\|_2^2 \, dt, \tag{7.1a}$$

subject to the DDE

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \tilde{f}[x(t), y(t), x(\omega(t)), y(\eta(t)), z_\varepsilon(t)], \qquad t_0 \le t \le t_f \tag{7.1b}$$

with startup functions and boundary conditions defined

$$x(t) = \alpha(t), \qquad t_0 - r \le t < t_0 \tag{7.1c}$$

$$y(t) = \beta(t), \qquad t_0 - s \le t < t_0 \tag{7.1d}$$

$$x_0 = q \text{ and } x_T = p, \tag{7.1e}$$

with the exogenous control satisfying the optional bounds

$$\vec{m} \le z_\varepsilon \le \vec{M}. \tag{7.1f}$$

Here, recall that $r,\ s > 0$ and $\omega(t) = t - r$ and $\eta(t) = t - s$ are time delay functions, and $\vec{m}$ and $\vec{M}$ are the min and max slopes determined from the delayed control variables of the original optimal control problem. Here the min and max slopes are displayed in vector form to emphasize the point that each exogenous control $z_{\varepsilon i}$ is associated with a slope pair $(m_i,\ M_i)$. Considering general formulations for OCDPs note that Proposition 5.1.2 still applies. Suppose that optimality has been obtained for both (2.2) and (7.1), and let $v = [x^*(t), u^*(t)]$ and $\tilde{v} = [x^*(t), y^*(t)]$. Then utilizing the cost function relationship argument outlined in Section 5.1.4 we obtain

$$\phi[v, T] + \int_{t_0}^{T} \mathcal{L}[v, t] \, dt \quad \le \quad \phi[\tilde{v}, T] + \int_{t_0}^{T} \mathcal{L}[\tilde{v}, t] \, dt, \tag{7.2}$$

and

$$\phi[\tilde{v},T]+\int_{t_0}^{T}\mathcal{L}[\tilde{v},t]+\varepsilon\|z_\varepsilon\|_2^2\,dt \quad\leq\quad \phi[v,T]+\int_{t_0}^{T}\mathcal{L}[v,t]+\varepsilon\|\dot{u^*}\|_2^2\,dt. \tag{7.3}$$

Using the above relations we have

$$\begin{aligned}\phi[v,T]+\int_{t_0}^{T}\mathcal{L}[v,t]\,dt &\leq \phi[\tilde{v},T]+\int_{t_0}^{T}\mathcal{L}[\tilde{v},t]+\varepsilon\|z_\varepsilon\|_2^2\,dt, &(7.4)\\ &\leq \phi[v,T]+\int_{t_0}^{T}\mathcal{L}[v,t]+\varepsilon\|\dot{u^*}\|_2^2\,dt, &(7.5)\\ &\leq \phi[\tilde{v},T]+\int_{t_0}^{T}\mathcal{L}[\tilde{v},t]+\varepsilon\|\dot{u^*}\|_2^2\,dt. &(7.6)\end{aligned}$$

Now considering the right-hand side of (7.4) and the inequality in (7.6) we obtain

$$\|{z_\varepsilon}^*\|_2^2 \leq \|\dot{u^*}\|_2. \tag{7.7}$$

Thus, if the derivative of the original optimal control is bounded, application of the exogenous input control method will produce an optimal exogenous control that is bounded in L^2 for any optimal control delay system with form (2.2) with any nonquadratic or quadratic cost function.

Now it is appropriate to discuss application of singular perturbation techniques to EIC formulated nonlinear optimal control delay problems with more general cost functions to study and analyze convergence of the solutions. Because Theorem 5.2.1 refers to an LQR problem it is not applicable in this case. Similar to LQR delay problems before applying most singular perturbation techniques to nonlinear delay problems removal of the delay terms may be necessary. Recall that this is a simple task for problems involving a single control delay variable (See Section 5.2.3). However, note that because general problems may involve control variables and/or inequivalent state delay and control delay functions (i.e., $x(t-r)$, $u(t-s)$, $r \neq s$), applying a time shift may prove difficult or impossible.

If the delay can be removed, the literature suggests several singular perturbation techniques for the solutions to nonlinear systems. O'Malley has constructed expansions for a certain class of nonlinear problems in [105]. In [121], Sannuti presents a more direct way of constructing the expansions for a class of nonlinear systems, and generalizes his approach and further relaxes the previously defined stability hypothesis in [120]. Freedman and Granoff formally construct an asymptotic solution for a nonlinear

Mayer type problem with a C^{∞} Hamiltonian function in [57]. Furthermore, many of the existing singular perturbation techniques for the solutions to nonlinear optimal control problems involve principles that pose conditions on the eigenvalues and/or the Hamiltonian function of the optimized system. In Chapter 10 of [129], Smith provides a more in depth discussion about some of these principles as well as surveys a few of the singular pertubation methods for nonlinear systems.

On the other hand if the delay cannot be removed, there is theoretical evidence that supports the application of asymptotic approximations to nonlinear optimal control systems with small scale delays. Interesting and significant extensions are to problems with small parameters multiplying derivatives and small delays [90]. Reddy and Sannuti study a fixed final time free end-point optimal problem with a small time delay in [122]. Recall that application of Pontryagin's minimum principle to an optimal control delay problem leads to a boundary-value problem consisting of both advanced and retarded type arguments. For the resulting nonlinear forward and backward BVP, the authors are able to construct an asymptotic power series solution in terms of the time delay. While there are many singular perturbation techniques available for both delayed and undelayed nonlinear systems it is often suggested to try to simplify the initial model by discarding nonlinearities or decomposing the initial problem into subproblems of reduced dimension and apply a singular perturbations method for linear systems [46].

User Guidance for Application of the EIC Method

The numerical and analytical results presented displayed that the exogenous input control method takes care of the computational difficulties with delayed control variables when solving optimal control delay systems on nonuniform grids. Not only does the EIC method improve the accuracy of the solution for the delayed control, but it many cases it also improves the accuracy for other system variables. While the usefulness of EIC is now established some technical points remain. In particular, the EIC method has three main components, the perturbation parameter acting on the exogenous control in the cost function, the bound for the exogenous control determined by the maximum and minimum slope of the original delayed control variable, and the free initial conditions for the newly added state/regularized control variables. In some cases a good approximation could be determined without activating the control bounds. In another case a very small value for the perturbation parameter and control bounds were needed to obtain a feasible solution. In other cases, specifying an initial value for the regularized control variable greatly improved the accuracy in the approximations. Because there are so many variations in the implementation for the EIC method, having a theory based guidance for users on how to set up the EIC formulation is desirable.

It is noted that results obtained for a specific EIC implementation is invariant with respect to the choice of discretization. Both TR and HS perform well as numerical integrators for each of the above EIC implementations. Hence, we hypothesize that the choice of implementation may be related to problem complexity (nonlinearity), problem mechanics (is a physical system described), and stability. The reasoning behind this hypothesis primarily stems from the results of the ECSTR problem defined in (4.6) and plotted in Figure 4.10. ECSTR is nonlinear, and it describes an actual chemical system. In this example $\varepsilon = 1.0$E-8 which means that the control is weighted very lightly in the cost function. In Figure 4.10e note that only the upperbound for the exogenous control was used on [0.1, 0.15]. Hence, neither the perturbation parameter nor the control bounds played a huge part in the computation of the solution. In [21], an initial condition was specified for the regularized control and a solid approximation was obtained. Additionally, note that the exogenous control plotted in Figure 4.10e is very close to the exogenous control output by MOS in Figure 4.15c. Maybe an EIC implementation with only an initial condition is necessary to obtain an adequate approximation for nonlinear problems with physical mechanics. This hypothesis will be examined further.

Stability of Lobatto IIIA Methods with Lagrange Interpolation

We have begun to numerically study the stability properties of Lobatto IIIA methods with Lagrange interpolation on DAE systems with time delays. Lobatto IIIA methods are implicit RK methods that satisfy the following relation:

$$c_i = \sum_{j=1}^{s} a_{ij}, \quad i = 1, \dots, s \tag{7.8a}$$

and the conditions

$$B(P) = \sum_{i=1}^{s} b_i c_i^{k-1} = \frac{1}{k}, \quad k = 1, \dots, p \tag{7.8b}$$

$$C(P) = \sum_{i=1}^{s} a_{ij} c_j^{k-1} = \frac{c_i^k}{k}, \quad k = 1, \dots, q \ \forall\, i \tag{7.8c}$$

$$D(r) = \sum_{i=1}^{s} b_i c_i^{k-1} a_{ij} = \frac{b_j}{k}(1 - c_j)^k, \quad k = 1, \ldots, r \;\; \forall\, j \tag{7.8d}$$

with $B(2s-2)$, $C(s)$, $D(s-2)$, $c_1 = 0$, $c_s = 1$, and $b_i = a_{si}$. This topic is motivated by [140]. While this topic is not directly tied to the goals of this dissertation, it serves as a very important supporting detail to the implicit trapezoid and Hermite-Simpson methods, and the direct transcription algorithm embedded in $\mathbb{SOCX}$.

The primary focus of this research study was dedicated to the investigation and exploration of a Runge-Kutta direct transcription software's ability to solve optimal control delay systems. To compute a solution the algorithm in $\mathbb{SOCX}$ first transforms the problem into a delay differential algebraic equation. Next it applies either the trapezoid method with Hermite interpolation of delayed state variables and piecewise linear interpolation of delayed control variables, or the Hermite-Simpson method with Hermite interpolation of delayed state variables or piecewise quadratic interpolation of delayed control variables to the resulting DDAE (See Section 3.2.1). Note that TR and HS are Lobatto IIIA methods, and that piecewise linear and piecewise quadratic interpolating polynomials are Lagrangian interpolators. Recent results in the literature have shown that Lobatto IIIA methods may have some undesirable stability properties when used to numerically integrate DAEs which have delays. Results in this thesis counter this statement, demonstrating that with these interpolation schemes some Lobatto IIIA methods do not exhibit this undesirable behavior. It is hypothesized that undesirable behavior reported in the literature is tied to the use of Lagrange interpolation, and the author's use of an ODE integrator as opposed to a boundary value formulation.

$\mathbb{SOCX}$'s success with computation of solutions to delay systems possibly highlights some advantages of direct transcription algorithms over ODE integrators for the numerical solutions to delay systems. Using the default discretization algorithm in $\mathbb{SOCX}$, we have shown how the trapezoid and Hermite-Simpson discretizations implemented with alternative interpolations numerically preserve the asymptotic stability of DDAE systems in [140]. To rule out the possibility that Hermite interpolation of delayed state variables enhances the computational results, we forced $\mathbb{SOCX}$ to use piecewise linear interpolation for delayed states in TR and piecewise quadratic interpolation for delayed states in HS so that Lagrange interpolation is carried out for both delayed controls and delayed state variables. Similar results were obtained with this implementation as well. It is concluded that different results were obtained in our experiments due to the fact that direct transcription solvers act like boundary value solvers. Hence, one cannot just apply initial value theory to understand the numerical behavior of direct transcription codes. Formal results of this study are being organized for future journal submission.

In conclusion, to assist in future software testing, development, and determination of what new capabilities are needed, good application based problems can always be helpful. Readers with any optimization, DDE, or delay problem that they would like to discuss should contact J. Betts, S. L. Campbell, or K. Thompson.

Chapter 8

Contributions of Thesis

The following are the main contributions of this dissertation.

8.1 Papers

- "*Optimal control software for constrained nonlinear systems with delays*"
  - Co-authors: *J. T. Betts and S. L. Campbell*
  - Proc. IEEE Mulit-Conference on Systems and Control (2011 MSC), Denver, 2011, 444-449.

  This paper reports on progress in developing a general purpose industrial grade software package to numerically solve complex optimal control problems with delays and state and control constraints using direct transcription. This paper was the initial stage of this dissertation research. Here we familiarize ourselves with the software, illustrate its capabilities, and examine its accuracy and efficiency by solving three examples. From the results we draw three primary conclusions: 1. problem formulation can have a large impact on how quickly the problem is solved, 2. if the delays are only in the state, then the software as it stands can efficiently solve many problems, and 3. delays in the controls present additional challenges and require the implementation of phases and guidelines on their use. Both Lagrange and Mayer formulations are considered. DT codes like $\mathbb{SOCX}$ that use implicit RK methods for discretization have an underlying continuity assumption. In places where continuity does not hold there is often some chatter in the control. This chatter can usually be removed by using problem phases. MOS solutions are computed for comparison purposes, and also illustrate another point about the effect of problem formulation.

- "*Optimal control of delay partial differential equations*"
  - Co-authors: *J. T. Betts and S. L. Campbell*
  - Editors: *L. Biegler, S. Campbell, and V. Mehrmann*
  - In Control and Optimization with Differential-Algebraic Constraints, SIAM, 2012, 213-231.

In this paper we discuss the extension of the direct transcription method to solve partial differential equations (PDEs) with delays. The direct transcription algorithm treats the problem in a global fashion, first discretizing the entire problem, and then computing the solution of the discrete approximation using a large scale optimization algorithm. The method has demonstrated success for problems with ordinary differential-algebraic systems. Here we show how the direct transcription method can efficiently accommodate delay terms through solving a PDE delay problem. The PDE delay problem is converted into an ODE delay system using the method of lines. Results are presented for problems with constant delay, time-varying delay, and delay that has both time and spatial dependence. All results reported pertain to state delays only.

- "*Direct transcription solution of optimal control problems with control delays*"
  - Co-authors: *J. T. Betts and S. L. Campbell*
  - Numerical Analysis and Applied Mathematics ICNAAM AIP Conference Proceedings, vol. 1389, no. 1, pp. 38-41, 2011.

This paper reports on the progress, challenges, and treatment of control delays using a direct transcription approach. Since delays can link behavior on different parts of the solution, an adaptive mesh refinement algorithm may be necessary to reduce error in the regions of the solution with minimal continuity. The adaptive property of the mesh refinement algorithm causes solution grids to become nonuniform at some point. Previously, we noted that the accommodation of control delays proved to be a more difficult task than state delays. After experimenting with varying meshes, we discovered that nonuniformity was the cause of unwanted oscillations appearing in the solutions of control delays. Here, we report on the impact of nonuniform grids on control delay optimal control problems, as well as the development of the exogenous input control method, a new regularization technique applied to obtain the solutions to control delay problems. A select few optimal control delay problems are solved using the new regularization method and the results are discussed.

- "*Simulation and Optimization of Systems with Delays*"
  - Co-authors: *J. T. Betts and S. L. Campbell*
  - Society for Modeling and Simulation Series 2013 Proceedings, San Diego, 2013, 1084-1085.

This paper focuses on the software's ability to simulate and model systems that involve both differential algebraic equations and delays. Some of the results of this paper are reported in Chapter 6. Simulation and control of systems with delays is an important task in working with many industrial, military, and mechanical applications. Many physical systems are most naturally modeled by DAEs. Because the software uses DAEs or DDAEs to reformulate the dynamics of optimal control problems, multiple variations of delays may be accommodated. In this paper we solve five different types of delay problems with $\mathbb{SOCX}$. The key focus of this study was to determine the best way to reformulate and solve a delayed optimal control problem when using a direct transcription approach that incorporates DAE formulations. Results show that with the proper formulations direct transcription packages can be extended to solve optimal control problems beyond the normal constant time delay problems.

- "*Direct transcription solution of optimal control problems with differential algebraic equations with delays*"
  - Co-authors: *J. T. Betts and S. L. Campbell*
  - Proc. 14th IASTED International Symposium on Intelligent Systems and Control (ISC 2013), Marina del Rey, 2013, 166-173.

This paper is a more in depth discussion of direct transcription, DAEs, and delays. It is shown that the DAE formulation allows for the handling of a variety of delayed problems including those with advances and those which are neutral or of mixed type. Mixed-type or forward-backward systems have both delays and advances. These types of systems arise as the necessary conditions for optimal control problems with state delays. Computational examples are given, and the solutions are discussed.

8.2 Presentations

- *Simulation and Optimization of Systems with Delays* April 2013

  SCS Spring Simulation Multi-Conference,

  San Diego, California

- *Solving Optimal Control Problems with Control Delays* July 2012

  SIAM Annual Meeting,

  Minneapolis, MN

- *Direct Transcription Software for Optimal Control Delay Problems* May 2012

  Electric Machines Technology Symposium,

  Philadelphia, PA

- *Optimal Control Software for Constrained Nonlinear Systems with Delays* Sept. 2011

  IEEE Multi-Conference on Systems and Control,

  Denver, CO

REFERENCES

[1] R. P. Agarwal and S. R. Grace, *Oscillations of forced functional differential equations generated by advanced arguments*, Aequationes Mathematicae, 63 (2002), pp. 26–45.

[2] A. Al-Mutib, *One-step implicit methods for solving delay differential equations*, International Journal of Computer Mathematics, 16 (1984), pp. 157–168.

[3] C. F. Alastruey and M. de la Sen, *Stability of time-delay systems via Lyapunov functions*, Mathematical Problems in Engineering, 8 (2002), pp. 197–205.

[4] M. D. Ardema, *Solution algorithms for non-linear singularly perturbed optimal control problems*, Optimal Control Applications and Methods, 4 (1983), pp. 283–302.

[5] R. B. Asher and H. R. Sebesta, *Optimal control of systems with state-dependent time delay*, International Journal of Control, 14 (1971), pp. 353–366.

[6] F. M. Atay, *Complex time-delay systems: Theory and applications*, Springer Complexity, Springer, 2010.

[7] M. Basin and J. Rodriguez-Gonzalez, *Optimal control for linear systems with multiple time delays in control input*, IEEE Transactions on Automatic Control, 51 (2006), pp. 91–97.

[8] J. J. Batzel, F. Kappel, and S. Timischl-Teschl, *A cardiovascular-respiratory control system model including state delay with application to congestive heart failure in humans*, Journal of Mathematical Biology, 50 (2005), pp. 293–335.

[9] R. J. Bauer, G. Mo, and W. Krzyzanski, *Solving delay differential equations in S-ADAPT by method of steps*, Computer Methods and Programs in Biomedicine, 111 (2013), pp. 715–734.

[10] V. M. Becerra, *$\mathcal{PSOPT}$ optimal control solver user manual release 2 build*, Reading RG6 6AY, University of Reading School of Systems Engineering United Kingdom, 2010.

[11] N. Bekiaris-Liberis and M. Krstic, *Compensation of time-varying input and state delays for nonlinear systems*, Journal of Dynamic Systems, Measurement, and Control, 134 (2012), p. 11009.

[12] ——, *Compensation of state-dependent input delay for nonlinear systems*, Automatic Control, IEEE Transactions on, 58 (2013), pp. 275–289.

[13] A. Bellen and M. Zennaro, *Numerical methods for delay differential equations*, Oxford University Press, 2013.

[14] R. E. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, USA, 1 ed., 1957.

[15] ——, *On the computational solution of differential-difference equations*, Journal of Mathematical Analysis and Applications, 2 (1961), pp. 108–110.

[16] R. E. Bellman, J. D. Buell, and R. E. Kalaba, *Numerical integration of a differential-difference equation with a decreasing time-lag*, Commun. ACM, 8 (1965), pp. 227–228.

[17] R. E. Bellman and K. L. Cooke, *On the computational solution of a class of functional differential equations*, Journal of Mathematical Analysis and Applications, 12 (1965), pp. 495–500.

[18] J. T. Betts, *Practical Methods for Optimal Control using Nonlinear Programming*, Society for Industrial and Applied Mathematics, Philadelphia, PA., 2001.

[19] J. T. Betts, S. L. Campbell, and A. Engelsone, *Direct transcription solution of inequality constrained optimal control problems*, in Proceedings, 2004 American Control Conference, Boston, MA, June 2004, pp. 1622–1626.

[20] J. T. Betts, S. L. Campbell, and K. C. Thompson, *Delay problem test set*. `http://www4.ncsu.edu/~kcthomps/images/DelayTestSetr.pdf`, 2010.

[21] ———, *Direct transcription solution of optimal control problems with control delays*, in Numerical Analysis and Applied Mathematics ICNAAM 2011: International Conference on Numerical Analysis and Applied Mathematics, vol. 1389, AIP Publishing, 2011, pp. 38–41.

[22] ———, *Optimal control software for constrained nonlinear systems with delays*, in Computer-Aided Control System Design (CACSD), 2011 IEEE International Symposium on, IEEE, 2011, pp. 444–449.

[23] ———, *Optimal control of a delay partial differential equation*, in Control and Optimization with Differential-Algebraic Constraints, L. Biegler, S. Campbell, and V. Mehrmann, eds., Philadelphia, 2012, SIAM, pp. 444–449.

[24] ———, *Direct transcription solution of optimal control problems with differential algebraic equations with delays*, Marina del Rey, 2013, 4th IASTED International Symposium on Intelligent Systems and Control, pp. 166–173.

[25] ———, *Simulation and optimization of systems with delays*, in Proceedings of the 2013 Spring Simulation Multi-conference poster session, Posters '13, San Diego, CA, USA, 2013, Society for Computer Simulation International, pp. 3:1–3:2.

[26] J. T. Betts and D. R. Ferguson, *Applied Mathematical Analysis @ONLINE*. `http://www.appliedmathematicalanalysis.com/`, Dec. 2010. [Online; accessed Dec.-2013].

[27] J. T. Betts and W. P. Huffman, *Sparse Optimal Control Software* $\mathbb{SOCS}$, Mathematics and Engineering Analysis Technical Document MEA-LR-085, Boeing Information and Support Services, The Boeing Company, PO Box 3707, Seattle, WA 98124-2207, July 1997.

[28] H. Bevrani, *Robust power system frequency control*, Power Electronics and Power Systems, Springer, New York, USA, January 2009.

[29] K. B. Blyuss, Y. N. Kyrychko, P. Hövel, and E. Schöll, *Control of unstable steady states in neutral time-delayed systems*, The European Physical Journal B, 65 (2008), pp. 571–576.

[30] J. Boisvert, M. Donaldson, and R. Spiteri, *Chapter 12: Solving parameter estimation problems with* $\mathbb{SOCX}$, ch. 12, pp. 253–272.

[31] U. Brandt-Pollmann, R. Winkler, S. Sager, U. Moslener, and J. P. Schlöder, *Numerical solution of optimal control problems with constant control delays*, Computational Economics, 31 (2008), pp. 181–206.

[32] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical solution of initial-value problems in differential-algebraic equations*, Vol. 14 of Classics in Applied Mathematics, SIAM, Philadelphia, PA., 1996.

[33] L. Brus, *MATLAB software for feedforward optimal control of systems with flow varying time delays - revision 2*, Tech. Rep. 2008-006, Department of Information Technology, Uppsala University, Mar. 2008. The software package can be downloaded from `http://www.it.uu.se/research/publications/reports/2008-006/NOCSoftwareV2.zip`.

[34] A. Budhiraja and K. Ross, *Existence of optimal controls for singular control problems with state constraints*, Ann. Appl. Prob, 16 (2006), pp. 2235–2255.

[35] R. Bulirsch, F. Montrone, and H. J. Pesch, *Abort landing in the presence of windshear as a minimax optimal control problem, Part 1: Necessary conditions*, Journal of Optimization Theory and Applications, 70 (1991), pp. 1–23.

[36] ———, *Abort landing in the presence of windshear as a minimax optimal control problem, Part 2: Multiple shooting and homotopy*, Journal of Optimization Theory and Applications, 70 (1991), pp. 223–254.

[37] V. Bulitko, I. Levner, and R. Greiner, *Real-time look-ahead control policies*, American Association for Artificial Intelligence, (2002).

[38] A. J. Calise, *Singular perturbation techniques for on-line optimal flight-path control*, AIAA Journal of Guidance and Control, 3 (1981), pp. 398–405.

[39] A. J. Calise and D. D. Moerder, *Singular perturbation techniques for real time aircraft trajectory optimization and control*, Tech. Rep. CR-3597, NASA, Aug. 1982.

[40] E. Çelik and M. Bayram, *The numerical solution of physical problems modeled as a systems of differential-algebraic equations (DAEs)*, Journal of the Franklin Institute, 342 (2005), pp. 1–6.

[41] C. Chen, D. Sun, and C. Chang, *Numerical solution of time-delayed optimal control problems by iterative dynamic programming*, Optimal Control Applications and Methods, 21 (2000), pp. 91–105.

[42] Y. Chen and J. Huang, *A new computational approach to solving a class of optimal control problems*, International Journal of Control, 58 (1993), pp. 1361–1383.

[43] N. Chopra, M. W. Spong, S. Hirche, and M. Buss, *Bilateral teleoperation over the internet: The time varying delay problem 1*, Urbana, 101 (2003), p. 61801.

[44] S. J. Citron, *Elements of optimal control*, Holt, Rinehart and Winston, New York, 1969.

[45] K. C. Craig, *Handling time delays.*, EDN: Electrical Design News, 57 (2012), p. 24.

[46] M. G. Dmitriev and G. A. Kurina, *Singular perturbations in control problems*, Automation and Remote Control, 67 (2006), pp. 1–43.

[47] D. M. Dubois, *Extension of the Kaldor-Kalecki model of business cycle with a computational anticipated capital stock*, Journal of Organization Transformation & Social Change, 1 (2004).

[48] L. Dugard and E. I. Verriest, *Stability and control of time-delay systems*, Springer Berlin Heidelberg, London, New York, 1998.

[49] A. Engelsone, *Direct transcription methods in optimal control: Theory and practice*, PhD thesis, North Carolina State University, May 2006.

[50] T. Erneux, *Applied delay differential equations*, Springer New York, New York, NY, 2009.

[51] C. E. Falbo, *Some elementary methods for solving functional differential equations @ONLINE*. `http://www.mathfile.net/hicstat_FDE.pdf`. Sonoma State University.

[52] P. Falugi, E. Kerrigan, and E. V. Wyk, *Imperial College London Optimal Control Software User Guide (ICLOCS)*, Department of Electrical and Electronic Engineering, Imperial College London London England, UK, 2010.

[53] W. R. (Firm), *Wolfram Mathematica tutorial collection: Advanced numerical differential equations solving in Mathematica*, Wolfram Mathematica Tutorial Collection, Wolfram Research, Incorporated, 2008.

[54] N. J. Ford and P. M. Lumb, *Mixed-type functional differential equations: A numerical approach*, Journal of Computational and Applied Mathematics, 229 (2009), pp. 471–479. Special Issue: Analysis and Numerical Approximation of Singular Problems.

[55] J. E. Forde, *Delay differential equation models in mathematical biology*, PhD thesis, The University of Michigan, May 2006.

[56] G. Fraser-Andrews, *Finding candidate singular optimal controls: A state of the art survey*, Journal of Optimization Theory and Applications, 60 (1989), pp. 173–190.

[57] M. I. Freedman and B. Granoff, *Formal asymptotic solution of a singularly perturbed nonlinear optimal control problem*, Journal of Optimization Theory and Applications, 19 (1976), pp. 301–325.

[58] Z. Gajic, *Optimal control of singularly perturbed linear systems and applications*, CRC Press, 2001.

[59] Z. Gajic, X. Shen, and M. Lim, *High accuracy techniques for singularly perturbed control systems– An overview,* invited paper, no. 65, Proceedings of Indian Academy of Sciences–PINSA, pp. 117–127.

[60] M. Gerdts, *Optimal control of ODEs and DAEs*, Walter de Gruyter, 2012.

[61] P. E. Gill, W. Murray, and M. A. Saunders, *User's guide for SNOPT 7: A Fortran package for large-scale nonlinear programming*, Systems Optimization Laboratory, Stanford University, California, 2007.

[62] V. Y. Glizer, *Asymptotic solution of a cheap control problem with state delay*, Dynamics and Control, 9 (1999), pp. 339–357.

[63] ——, *Blockwise estimate of the fundamental matrix of linear singularly perturbed differential systems with small delay and its application to uniform asymptotic solution*, Journal of mathematical analysis and applications, 278 (2003), pp. 409–433.

[64] ——, *Stochastic singular optimal control problem with state delays: Regularization, singular perturbation, and minimizing sequence*, SIAM Journal on Control and Optimization, 50 (2012), pp. 2862–2888.

[65] L. Göllmann, D. Kern, and H. Maurer, *Optimal control of constrained time lag systems: Necessary conditions and numerical treatment*, PAMM, 7 (2007), pp. 1151701–1151702.

[66] ——, *Optimal control problems with delays in state and control variables subject to mixed control-state constraints*, Optimal Control Applications and Methods, (2008).

[67] L. Göllmann and H. Maurer, *Theory and applications of optimal control problems with multiple time-delays*, To appear in Journal of Industrial and Management Optimization, 10 (April 2014), pp. 413–441.

[68] N. Guglielmi and E. Hairer, *Asymptotic expansions for regularized state-dependent neutral delay equations.*, SIAM J. Math. Analysis, 44, pp. 2428–2458.

[69] ——, *Regularization of neutral delay differential equations with several delays*, Journal of Dynamics and Differential Equations, (2013), pp. 1–20.

[70] ——, *Numerical approaches for state-dependent neutral delay equations with discontinuities*, Mathematics and Computers in Simulation, 95 (2014), pp. 2–12. Discontinuous Differential Systems : Theory and Numerical Methods.

[71] K. Hattaf and N. Yousfi, *Optimal control of a delayed HIV infection model with immune response using an efficient numerical method*, ISRN Biomathematics, 2012 (2012), pp. 2795–2800.

[72] Y. He, G. Liu, and D. Rees, *New delay-dependent stability criteria for neural networks with time-varying delay*, Neural Networks, IEEE Transactions on, 18 (2007), pp. 310–314.

[73] Y. He, M. Wu, J. She, and G. Liu, *Delay-dependent robust stability criteria for uncertain neutral systems with mixed delays*, Systems and Control Letters, 51 (2004), pp. 57–65.

[74] E. Hellström, M. Ivarsson, J. Åslund, and L. Nielsen, *Look-ahead control for heavy trucks to minimize trip time and fuel consumption*, in Fifth IFAC Symposium on Advances in Automotive Control, Monterey, CA, USA, 2007.

[75] J. Henderson, *Boundary value problems for functional differential equations*, World Scientific, 1995.

[76] G. T. Huntington, *Advancement and analysis of a Gauss pseudospectral transcription for optimal control problems*, PhD thesis, 2007.

[77] C. Hwang and Y. Cheng, *A note on the use of the Lambert W function in the stability analysis of time-delay systems*, Automatica, 41 (2005), pp. 1979–1985.

[78] T. Insperger, R. Wohlfart, J. Turi, and G. Stepan, *Equations with advanced arguments in stick balancing models*, in Time Delay Systems: Methods, Applications and New Trends, R. Sipahi, T. Vyhldal, S.-I. Niculescu, and P. Pepe, eds., vol. 423 of Lecture Notes in Control and Information Sciences, Springer Berlin Heidelberg, 2012, pp. 161–172.

[79] F. Ismail, R. A. Al-Khasawneh, A. S. Lwin, and M. Suleiman, *Numerical treatment of delay differential equations by Runge-Kutta method using Hermite interpolation*, Matematika, 18 (2002), pp. 79–90.

[80] E. Jarlebring and T. Damm, *The Lambert W function and the spectrum of some multidimensional time-delay systems*, Automatica, 43 (2007), pp. 2124–2128.

[81] L. S. Jennings, M. E. Fisher, K. L. Teo, and C. J. Goh, *MISER3 - Optimal control toolbox.* http://school.maths.uwa.edu.au/ les/miser/OCTmanual.pdf.

[82] X. Jiao, J. Yang, and Q. Li, *Adaptive control for a class of nonlinear systems with time-varying delays in state and input*, Journal of Control Theory and Applications, 9 (2011), pp. 183–188.

[83] C. Johnson and J. Gibson, *Singular solutions in problems of optimal control*, Automatic Control, IEEE Transactions on, 8 (1963), pp. 4–15.

[84] A. Kaddar and H. T. Alaoui, *Fluctuations in a mixed is-lm business cycle model*, Electronic Journal of Differential Equations, 2008 (2008), pp. 1–9.

[85] L. Kahina and A. Mohamed, *Direct method for resolution of optimal control problem with free initial condition*, International Journal of Differential Equations, 2012 (2012).

[86] C. Kao and A. Rantzer, *Stability analysis of systems with uncertain time-varying delays*, Automatica, 43 (2007), pp. 959–970.

[87] A. A. Khan, D. M. Tilbury, and J. R. Moyne, *Favorable effect of time delays on tracking performance of type-I control systems*, Control Theory Applications, IET, 2 (2008), pp. 210–218.

[88] V. Kharitonov and E. Plischke, *Lyapunov matrices for time-delay systems*, Systems and Control Letters, 55 (2006), pp. 697–706.

[89] P. V. Kokotovic, *Singular perturbations in optimal control*, Rocky Mountain J. Math., 6 (1976), pp. 767–774.

[90] P. V. Kokotovic, R. E. O'Malley, and P. Sannuti, *Singular perturbations and order reduction in control theory – An overview*, Automatica, 12 (1976), pp. 123–132.

[91] Y. Kuang, *Delay differential equations: with applications in population dynamics*, Academic Press Boston, 1993.

[92] F. Lewis, D. Vrabie, and V. Syrmos, *Optimal control*, John Wiley & Sons, 2012.

[93] Y. Li and L. Baldacchino, *Implementation of some higher-order convection schemes on non-uniform grids*, International Journal for Numerical Methods in Fluids, 21 (1995), pp. 1201–1220.

[94] P. M. Lima, M. F. Teodoro, N. J. Ford, and P. M. Lumb, *Finite element solution of a linear mixed-type functional differential equation*, Numerical Algorithms, 55 (2010), pp. 301–320.

[95] X. Lou, Q. Ye, and B. Cui, *Impulsive stabilization of fuzzy neural networks with time-varying delays*, Arabian Journal of Mathematics, 2 (2013), pp. 65–79.

[96] R. Luus, *Iterative dynamic programming*, CRC Press, 2010.

[97] D. B. McDonald, *Quickest descent control of nonlinear systems with singular control chatter elimination*, in Proceedings of the 2008 IAJC-IJME Conference. Nashville: California State University, 2008.

[98] W. Mekarapiruk and R. Luus, *On solving optimal control problems with free initial condition using iterative dynamic programming*, The Canadian Journal of Chemical Engineering, 79 (2001), pp. 777–784.

[99] A. Miele, T. Wang, and W. W. Melvin, *Optimal abort landing trajectories in the presence of windshear*, Journal of Optimization Theory and Applications, 55 (1987), pp. 165–202.

[100] A. A. Milyutin and N. P. Osmolovskii, *Calculus of variations and optimal control*, vol. 180 of Translations of Mathematical Monographs, American Mathematical Society, Providence, RI, 1998. Translated from the Russian manuscript by Dimitrii Chibisov.

[101] S. Mondié, G. Ochoa, and B. Ochoa, *Instability conditions for linear time delay systems: a lyapunov matrix function approach*, International Journal of Control, 84 (2011), pp. 1601–1611.

[102] A. D. Myshkis, *Differential equations, ordinary with distributed arguments*, vol. 3, Kluwer Academic Publishers, 1989, pp. 144–147.

[103] S. Niculescu and K. Gu, *Robust stability of some oscillatory systems including time-varying delay with applications in congestion control*, ISA transactions, 42 (2003), pp. 595–603.

[104] H. J. Oberle and H. J. Pesch, *Numerical treatment of delay differential equations by Hermite interpolation*, Numerische Mathematik, 37 (1981), pp. 235–255.

[105] R. E. O'Malley, *On the asymptotic solution of certain nonlinear singularly perturbed optimal control problems.*, In: Preprints of a Symposium on Singular Perturbations, JACC, 1972, pp. 44–56.

[106] ——, *A more direct solution of the nearly singular linear regulator problem*, SIAM Journal on Control and Optimization, 14 (1976), pp. 1063–1077.

[107] ——, *Introduction to singular perturbations*, vol. 14, Elsevier, 2012.

[108] R. E. O'Malley and A. Jameson, *Cheap control of the time-invariant regulator*, Applied Mathematics and Optimization, 1 (1975), pp. 337–354.

[109] ——, *Singular perturbations and singular arcs–Part I*, Automatic Control, IEEE Transactions on, 20 (1975), pp. 218–226.

[110] ——, *Singular perturbations and singular arcs–Part II*, Automatic Control, IEEE Transactions on, 22 (1977), pp. 328–337.

[111] G. Orosz, J. Moehlis, and R. M. Murray, *Controlling biological networks by time-delayed signals*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 368 (2010), pp. 439–454.

[112] A. Papachristodoulou, M. M. Peet, and S. Niculescu, *Stability analysis of linear systems with time-varying delays: Delay uncertainty and quenching*, in Decision and Control, 2007 46th IEEE Conference on, IEEE, 2007, pp. 2117–2122.

[113] M. A. Patterson and A. Rao, *GPOPS-II: A Matlab software for solving multiple-phase optimal control problems using hp–Adaptive Gaussian quadrature collocation methods and sparse nonlinear programming*, ACM Trans. Math. Soft., 39 (2013).

[114] L. S. Pontryagin, *The Mathematical Theory of Optimal Processes*, Wiley-Interscience, New York, NY, 1962.

[115] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko, *The mathematical theory of optimal processes*, A Pergamon Press Book. The Macmillan Co., New York, 1964. Translated by D. E. Brown.

[116] P. B. Reddy and P. Sannuti, *Optimal control of singularly perturbed time delay systems with an application to a coupled core nuclear reactor*, in Decision and control including the 13th Symposium on Adaptive Processes, 1974 IEEE Conference on, vol. 13, Nov 1974, pp. 793–803.

[117] I. M. Ross and F. Fahroo, *User's manual for DIDO: A Matlab application for solving optimal control problems*, Tech. Rep. AAS-01-03, Naval Postgraduate School, Monterey, CA., 2001.

[118] P. E. Rutquist and M. M. Edvall, *PROPT: Matlab optimal control software*, Washington, 2008, Tomlab Optimization, Inc.

[119] ——, *PROPT Example Suite: Matlab optimal control software*, Tomlab Optimization Inc, 260 (2009).

[120] P. Sannuti, *Asymptotic series solution of singularly perturbed optimal control problems*, Automatica, 10 (1974), pp. 183–194.

[121] ——, *Asymptotic expansions of singularly perturbed quasi-linear optimal systems*, SIAM Journal on Control, 13 (1975), pp. 572–592.

[122] P. Sannuti and P. B. Reddy, *Asymptotic series solution of optimal systems with small time delay*, Automatic Control, IEEE Transactions on, 18 (1973), pp. 250–259.

[123] L. F. Shampine, *Solving ODEs and DDEs with residual control*, Applied Numerical Mathematics, 52 (2005), pp. 113–127.

[124] L. F. Shampine and S. Thompson, *Solving {DDEs} in Matlab*, Applied Numerical Mathematics, 37 (2001), pp. 441–458.

[125] ——, *A friendly Fortran DDE solver*, Applied Numerical Mathematics, 56 (2006), pp. 503–516.

[126] L. F. Shampine, S. Thompson, and J. Kierzenka, *Solving delay differential equations with dde23*, URL http://www. runet. edu/˜ thompson/webddes/tutorial. pdf, (2000).

[127] T. Shima and J. Shinar, *Time-varying linear pursuit-evasion game models with bounded controls*, Journal of Guidance, Control, and Dynamics, 25 (2002), pp. 425–432.

[128] H. Shinozaki and T. Mori, *Robust stability analysis of linear time-delay systems by Lambert function: Some extreme point results*, Automatica, 42 (2006), pp. 1791–1799.

[129] D. R. Smith, *Singular-perturbation theory: an introduction with applications*, Cambridge University Press, 1985.

[130] S. Smith, *Optimal control of delay differential equations using evolutionary algorithms*, in Proc. Asia. pacific. complex. Sys. Conf, 2004, pp. 400–409.

[131] J. L. Speyer and D. H. Jacobson, *Primer on optimal control theory*, Society for Industrial and Applied Mathematics, Philadephia, PA, 2010.

[132] R. F. Stengel, R. Ghigliazza, N. Kulkarni, and O. Laplace, *Optimal control of innate immune response*, Optimal Control Applications and Methods, 23 (2002), pp. 91–104.

[133] M. B. Suleiman and F. Ismail, *Solving delay differential equations using componentwise partitioning by Runge–Kutta method*, Applied Mathematics and Computation, 122 (2001), pp. 301–323.

[134] D. Sun and T. Huang, *The solutions of time-delayed optimal control problems by the use of modified line-up competition algorithm*, Journal of the Taiwan Institute of Chemical Engineers, 41 (2010), pp. 54–64.

[135] J. Tadeusz, *Advanced differential equations with nonlinear boundary conditions*, Journal of Mathematical Analysis and Applications, 304 (2005), pp. 490–503.

[136] S. Takahashi, K. Yamanaka, and M. Yamada, *Detection of dominant poles of systems with time delay by using Padé approximation*, International Journal of Control, 45 (1987), pp. 251–254.

[137] G. Y. Tang, C. Li, and L. Sun, *Optimal tracking control for large-scale interconnected systems with time-delays*, Computers and Mathematics with Applications, 53 (2007), pp. 80–88.

[138] W. X. Tao, *A numerical approach of optimal control for generalized delay systems by general Legendre wavelets*, International Journal of Computer Mathematics, 86 (2009), pp. 743–752.

[139] D. Taoutaou, S.-I. Niculescu, and K. Gu, *Robust stability of teleoperation schemes subject to constant and time-varying communication delays*, in Advances in Time-Delay Systems, Springer, 2004, pp. 327–338.

[140] H. Tian, Q. Yu, and J. Kuang, *Asymptotic stability of linear neutral delay differential-algebraic equations and Runge–Kutta methods*, SIAM Journal on Numerical Analysis, 52 (2014), pp. 68–82.

[141] F. E. Udwadia, H. von Bremen, R. Kumar, and M. Hosseini, *Time delayed control of structural systems*, Earthquake Engineering and Structural Dynamics, 32 (2003), pp. 495–535.

[142] R. Vinter, *Optimal control*, Modern Birkhauser Classics, Springer, 2010.

[143] O. von Stryk and R. Bulirsch, *Direct and Indirect Methods for Trajectory Optimization*, Annals of Operations Research, 37 (1992), pp. 357–373.

[144] G. Vossen, V. Rehbock, and A. Siburian, *Numerical solution methods for singular control with multiple state dependent forms*, Optimization Methods and Software, 22 (2007), pp. 551–559.

[145] A. Wächter and L. T. Biegler, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, Mathematical Programming, 106 (2006), pp. 25–57. 10.1007/s10107-004-0559-y.

[146] L. Wang, Z. Wang, and X. Zou, *Periodic solutions of neutral functional differential equations*, Journal of the London Mathematical Society, 65 (2002), pp. 439–452.

[147] L. Watson and J. A. Ford, *Optimization and nonlinear equations*, Journal of computational and applied mathematics, Elsevier, 2001.

[148] Q. Wei, H. Zhang, D. Liu, and Y. Zhao, *An optimal control scheme for a class of discrete-time nonlinear systems with time delays using adaptive dynamic programming*, Acta Automatica Sinica, 36 (2010), pp. 121–129.

[149] J. Werbowski, *Oscillations of first-order differential equations generated by advanced arguments*, no. 30, Funkcial. Ekvan, 1987, pp. 69–79.

[150] E. Witrant, C. Canudas-De-Wit, D. Georges, and M. Alamir, *Remote stabilization via time-varying communication network delays: application to TCP networks*, in Control Applications, 2004. Proceedings of the 2004 IEEE International Conference on, vol. 1, 2004, pp. 474–479 Vol.1.

[151] C. Wu, K. L. Teo, R. Li, and Y. Zhao, *Optimal control of switched systems with time delay*, Applied Mathematics Letters, 19 (2006), pp. 1062–1067.

[152] F. Wu and K. M. Grigoriadis, *LPV systems with parameter-varying time delays: analysis and control*, Automatica, 37 (2001), pp. 221–229.

[153] M. Wu, Y. He, and J. She, *Stability analysis and robust control of time-delay systems*, Springer Berlin Heidelberg, London ; New York, 2010.

[154] L. M. y Tern-Romero, M. Silber, and V. Hatzimanikatis, *The origins of time-delay in template biopolymerization processes*, PLoS Comput Biol, 6 (2010), p. e1000726.

[155] S. Yi, *Time-delay systems: analysis and control using the Lambert W function*, World Scientific, 2010.

[156] L. Yin and X. Li, *Impulsive stabilization for a class of neural networks with both time-varying and distributed delays*, Advances in Difference Equations, 2009 (2009), p. 859832.

[157] D. L. Zackon, *Variable time delay by Padé approximation*, Electronic Computers, IRE Transactions on, EC-10 (1961), pp. 783–783.

[158] W. Zhong, *Duality system in applied mechanics and optimal control*, Advances in Mechanics and Mathematics, Kluwer Academic Publishers, 2004.

APPENDICES

Appendix A

Supporting $\mathbb{SOCX}$ Files

A.1 Driver for Optimal Control Delay Problem

smdelay.f

The following program is the driver for the simple mixed delay problem in Eq. (4.1).

```
       PROGRAM SMDELAY
C      ****************************************************************
      IMPLICIT DOUBLE PRECISION  (A-H,O-Z)
C
      PARAMETER  (MXIW=100000000,MXRW=100000000
     $           ,MXC=100000,MXPHS=15,MXDP=50)
C
      DIMENSION  IWORK(MXIW),WORK(MXRW)
      DIMENSION  CSTAT(MXC),IPCPH(MXPHS+1),DPARM(MXDP),IPDPH(MXPHS+1)
C
      EXTERNAL   DUMYPF,  DUMYPR, DSLCED, DUMYIG
      external   DSLCIN,  DSLCDE,  DSLCIG
      external   MSLCIN,  MSLCDE
C
C  ----SPARSE OPTIMAL CONTROL INPUT PROCEDURE
```

```
        CALL INSOCX('default')
        call insocx('qdrtol=1.d-7')
        call insocx('odetol=1.d-7')
        CALL insocx('aeqtol=1.d-7')
        call insocx('objctl=1.d-7')
        call insocx('ipgrd=20')
        call insocx('ipode=20')
        call insocx('mitode=15')
        call insocx('itswch=2')
        call insocx('mtswch=3')
        call insocx('nsswch=1')
C
C  ----SPARSE NONLINEAR PROGRAMMING INPUT PROCEDURE
        CALL INSNLP('sparse default')
        call insnlp('ioflag=20')
        call insnlp('tolpvt=.1d0')
        call insnlp('tolfil=10.')
c
C  ----OPTIMAL CONTROL DELAY FORMULATION (DSLCIN, DSLCDE, DSLCED)
          call socxdl(dslcin,dumyig,dslcde,dumypf,dumypr,dslced,
     &                iwork,mxiw,work,mxrw,mxphs,
     &                cstat,mxc,ipcph,dparm,mxdp,ipdph,needed,ier)
C
C  ----METHOD OF STEPS FORMULATION (MSLCIN, MSLCDE)
c
C        **** UNCOMMENT FOR METHOD OF STEPS ****
c
C        call socxmn(mslcin,dumyig,mslcde,dumypf,dumypr,
C     &                iwork,mxiw,work,mxrw,mxphs,
C     &                cstat,mxc,ipcph,dparm,mxdp,ipdph,needed,ier)
C
```

```
      STOP
      END
c ----------------------------------------------------------------------
C ----------------OPTIMAL CONTROL DELAY FORMULATION CODE----------------
c ----------------------------------------------------------------------
c
      SUBROUTINE DSLCIN(IPHASE,NPHS,METHOD,NSTG,NCF,NPF,NPV,NAV,NGRID,
     &                  INIT,MAXMIN,MXPARM,P0,PLB,PUB,PLBL,
     &                  MXSTAT,Y0,Y1,YLB,YUB,STSKL,STLBL,MXPCON,CLB,CUB,
     &                  CLBL,MXTERM,COEF,ITERM,TITLE,IER)
C
C     *****************************************************************
c
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C  Arguments:
C
      INTEGER    IPHASE,NPHS,METHOD,NSTG,NCF(6),NPF(2),NPV,NAV,NGRID,
     &           INIT(2),MAXMIN,MXPARM,MXSTAT,MXPCON,MXTERM,
     &           ITERM(4,MXTERM),IER
      DIMENSION  P0(MXPARM),PLB(MXPARM),PUB(MXPARM),Y0(0:MXSTAT),
     &           Y1(0:MXSTAT),YLB(-1:1,0:MXSTAT),YUB(-1:1,0:MXSTAT),
     &           STSKL(0:MXSTAT+MXPARM,2),CLB(MXPCON),CUB(MXPCON),
     &           COEF(MXTERM)
      CHARACTER  TITLE(3)*60,PLBL(MXPARM)*80,STLBL(0:MXSTAT)*80,
     &           CLBL(0:MXPCON)*80
C
      PARAMETER (ZERO=0.D0,ONE=1.D0,TWO=2.D0)
c
C     =================================================================
      TITLE(1) = 'Simple Mixed Delay Problem'
      TITLE(2) = 'Optimal Delay Formulation'
C    -------------------------------------------------------------------
c
C   ----NUMBER OF PHASES
        NPHS = 1
c
C   ----SPECIFY THE NUMBER OF CONTINUOUS FUNCTIONS ON THE PHASE
C     ::NUMBER OF DIFFERENTIAL EQUATIONS
        NCF(1) = 1
C     ::NUMBER OF QUADRATURE FUNCTIONS
        NCF(3) = 1
C     ::NUMBER OF DELAY FUNCTIONS
        NCF(6) = 2
c
C   ----NUMBER OF GRID POINTS IN INITIAL GRID
        NGRID = 21
c
C   ----DEFAULT GUESS ROUTINE
        init(1) = 1
c
C   ----INITIALIZE RUNNING COUNTER FOR THE NUMBER OF TERMS,
C       NUMBER OF CONSTRAINTS, NUMBER OF DELAY EQUATIONS,
C       AND NUMBER OF ALGEBRAIC VARIABLES
C
      NTERM = 0
      NKON = 0
      NDE = 0
      NAV = 0
C     ==================================================================
C     =========TIME=====================================================
C     ==================================================================
c
      STLBL(0) = 'TIME    Time'
c
C    ----FIX INITIAL AND FINAL TIME
C
      Y0(0) = 0.D0
      Y1(0) = 5.D0
c
      YLB(-1,0) = Y0(0)
      YUB(-1,0) = Y0(0)
      YLB(1,0) = Y1(0)
      YUB(1,0) = Y1(0)
c
C     ==================================================================
C     ========Differential Variables=====================================
C     ==================================================================
c
      nde = nde + 1
C
      STLBL(nde) = 'X(T)        X-STATE'
c
C    ----FIX INITIAL STATE
        y0(nde) = 1.D0
        y1(nde) = 0.D0
        ylb(-1,nde) = y0(nde)
        yub(-1,nde) = y0(nde)
c
c     ==================================================================
C     ========Algebraic Variables========================================
C     ==================================================================
c
```

```
      nav = nav + 1
      STLBL(nde+nav) = 'U(T)        U-CTRL'
c
C    ----SPECIFY INITIAL GUESS FOR CONTROL
        y0(nde+nav) = 0.d0
        y1(nde+nav) = 0.d0
c
c     ==================================================================
C     ========Boundary Conditions=======================================
C     ==================================================================
c
c       FREE FINAL STATE
c
c     ==================================================================
C     =========Quadrature Function======================================
C     ==================================================================
c
      MAXMIN = -1
c
      CLBL(0) = 'QDRINT  Integral Performance Index'
C
      NTERM = NTERM + 1
      ITERM(1,NTERM) = 0
      ITERM(2,NTERM) = iphase
      ITERM(3,NTERM) = 0
      ITERM(4,NTERM) = -ncf(2)-ncf(3)
      COEF(NTERM) = 1.d0
C
      RETURN
      END
      SUBROUTINE DSLCDE(IPHASE,T,Y,NY,P,NP,FRHS,NFRHS,IFERR)
C
C         COMPUTES THE RIGHT HAND SIDES OF THE DIFFERENTIAL EQUATIONS.
C     *****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C  ARGUMENTS:
C
      DIMENSION  Y(NY),P(NP),FRHS(NFRHS)
c
      parameter (zero=0.d0,one=1.d0,two=2.d0)
C     *****************************************************************
C    ----SET FUNCTION ERROR FLAG.
      IFERR = 0
c
C    ----DEFINE VARIABLES IN ORDER (STATES,CONTROLS,DELAY VARIABLES)
      xt = y(1)
      ut = y(2)
      utm1 = y(3)
      xtm2 = y(4)
c
C    ----DEFINE RHS FUNCTIONS
      frhs(1) = xtm2 + utm1
c
C    ----DEFINE QUADRATURE FUNCTIONS
      frhs(2) = xt**2 + ut**2
c
      RETURN
      END
      SUBROUTINE DSLCED(IPHASE,TIME,TI,TF,IDELAY,NY,IPHDLY,ISTDLY,
     &                  TDELAY,YDELAY,IFERR)
C     ****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
c
c         input:
c
c             iphase    current phase number
c             time      current time
c             ti        initial time of phase
c             tf        final time of phase
c             idelay    number of delay function about which data requested
c                       (subscript "k" in omega_{k}):  (k = 0,1,...,ndelay)
c
c         output:
c
c             iphdly    phase number at delay time;  iphdly(nde+nav)
c             istdly    state number of delay;       istdly(nde+nav)
c             tdelay    delay time; (omega_{k})
c             ydelay    dynamic variable value at delay time;  yvec[omega_{k}]
c                       ydelay(nde+nav)
c
c
      dimension ydelay(ny),iphdly(ny),istdly(ny)
C
      IFERR = 0
c
      if(idelay.eq.1) then
C
c         evaluate delay time
c
```

```
        omega1 = time - 2.d0
        tdelay = omega1
c
        if(omega1.lt.ti) then
         iphdly(1) = 0
            istdly(1) = 0
c
c    ----DEFINE STARTUP FUNCTIONS FOR STATE DELAY VARIABLES
c
          xt = 1.d0
          ydelay(1) = xt
c
        endif
c
      elseif(idelay.eq.2) then

        iphdly(2) = 0
        istdly(2) = 0
C
c         evaluate delay time
c
        omega2 = time - 1.d0
        tdelay = omega2
c
        if(omega2.lt.ti) then

         iphdly(2) = 0
            istdly(2) = 0
c
c    ----DEFINE STARTUP FUNCTIONS FOR CONTROL DELAY VARIABLES
c
```

```
          ut = 1.d0
          ydelay(2) = ut
        endif
      endif
c
      RETURN
      END

c ----------------------------------------------------------------------
C -------------------------METHOD OF STEPS CODE-------------------------
c ----------------------------------------------------------------------
      SUBROUTINE MSLCIN(IPHASE,NPHS,METHOD,NSTG,NCF,NPF,NPV,NAV,NGRID,
     &                 INIT,MAXMIN,MXPARM,P0,PLB,PUB,plbl,
     &                 MXSTAT,Y0,Y1,YLB,YUB,STSKL,stlbl,MXPCON,CLB,CUB,
     &                 clbl,MXTERM,COEF,ITERM,TITLE,IER)
C     ******************************************************************
C     ******************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      PARAMETER (ZERO=0.D0,ONE=1.D0,ONEEP1=1.D+01,ONEEP3=1.D+03)
C
C  Arguments:
      INTEGER    IPHASE,NPHS,METHOD,NSTG(2),NCF(6),NPF(2),NPV,NAV,NGRID,
     &           INIT(2),MAXMIN,MXPARM,MXSTAT,MXPCON,MXTERM,
     &           ITERM(4,MXTERM),IER
      DIMENSION  P0(MXPARM),PLB(MXPARM),PUB(MXPARM),Y0(0:MXSTAT),
     &           Y1(0:MXSTAT),YLB(-1:1,0:MXSTAT),YUB(-1:1,0:MXSTAT),
     &           STSKL(0:MXSTAT+MXPARM,2),CLB(MXPCON),CUB(MXPCON),
     &           COEF(MXTERM)
      character  title(3)*60,plbl(mxparm)*80,stlbl(0:mxstat)*80,
     &           clbl(0:mxpcon)*80
```

```
c
      parameter (Nstep=5)
      parameter (tF=5.d0,x0=1.d0,tau=1.d0)
c
      CHARACTER*100 RITOUT
c
      INCLUDE '../commons/odeprb.cmn'
C     ******************************************************************
      TITLE(1) = 'Simple Mixed Delay Problem'
      TITLE(2) = 'Method of Steps Formulation'
C    -------------------------------------------------------------------
c
C   ----NUMBER OF PHASES
        NPHS = 1
c
C   ----NUMBER OF GRID POINTS IN INITIAL GRID
        NGRID = 2
c
C   ----DEFAULT GUESS ROUTINE
        init(1) = 1
c
C   ----INITIALIZE RUNNING COUNTER FOR THE NUMBER OF TERMS,
C       NUMBER OF CONSTRAINTS, NUMBER OF DELAY EQUATIONS,
C       AND NUMBER OF ALGEBRAIC VARIABLES
C
      NTERM = 0
      NKON = 0
      NDE = 0
      NAV = 0
C   ----SPECIFY THE PARAMETER FOR NUMBER OF VARIABLES
C     ::NUMBER OF STATES
```

```
        nst = 1
C     ::NUMBER OF CONTROLS
        nct = 1
c
C     ===================================================================
C     =========TIME======================================================
C     ===================================================================
c
C    ----FIX INITIAL AND FINAL TIME
C
      Y0(0) = zero
      Y1(0) = tau
c
      YLB(-1,0) = Y0(0)
      YUB(-1,0) = Y0(0)
      YLB(1,0) = Y1(0)
      YUB(1,0) = Y1(0)
C     ===================================================================
C     ========Differential Variables=====================================
C     ===================================================================
c
      stateloop: do k = 1,Nstep
c
        nde = nde + 1
c
        ritout = repeat(' ',100)
        ritout(1:42) = 'X         State on Step.................'
        write(ritout(2:4),'(i3.3)') k
        write(ritout(41:43),'(i3.3)') k
        STLBL(nde) = ritout
c
```

```
C   ----SPECIFY INITIAL CONDITION FOR STATE X
        y0(nde) = 1.d0
        y1(nde) = 0.d0
c
      enddo stateloop
c
C   ----SPECIFY THE NUMBER OF DIFFERENTIAL EQUATIONS ON THE PHASE
        NCF(1) = NDE
c
c     ==================================================================
C     ========Algebraic Variables=======================================
C     ==================================================================
c
      cntrlloop: do k = 1,Nstep
c
        nav = nav + 1
c
        ritout = repeat(' ',100)
        ritout(1:42) = 'U         Control on Step...............'
        write(ritout(2:4),'(i3.3)') k
        write(ritout(41:43),'(i3.3)') k
        STLBL(nde+nav) = ritout
c
C   ----SPECIFY INITIAL GUESS FOR CONTROL U
        y0(nde+nav) = 0.1d0
        y1(nde+nav) = 0.1d0

      enddo cntrlloop
c
c     ==================================================================
C     ========Boundary Conditions=======================================
C     ==================================================================
c
      ndlst = nst*Nstep
      ndlct = nct*Nstep
C
C   ----FIX INITIAL INITIAL VALUE FOR STATE
      ylb(-1,1:nst) = y0(1:nst)
      yub(-1,1:nst) = y0(1:nst)
c
C   ----LINK STATES ON INTERIOR BOUNDARIES
c
      jj = 0
      stbdloop: do k = 0,Nstep-2
c
        do l = 1,nst
          CALL LINKST(IPHASE,+1,jj+l,IPHASE,-1,jj+l+nst,NKON,
     $      NTERM,ITERM,COEF,CLB,CUB)
        enddo
c
        jj = jj + nst
c
      enddo stbdloop
c
      jj = ndlst
      ctbdloop: do k = 0,Nstep-2
c
C   ----LINK CONTROLS ON INTERIOR BOUNDARIES
        do l = 1,nct

          CALL LINKST(IPHASE,+1,jj+l,IPHASE,-1,jj+l+nct,NKON,
     $      NTERM,ITERM,COEF,CLB,CUB)
```

```
          enddo
c
          jj = jj + nct
c
        enddo ctbdloop
c
c       ==================================================================
C       ========Quadrature================================================
C       ==================================================================
        clBL(0) = 'QUADOBJ Quadratic Cost Function'
        MAXMIN = -1
c
        do kk = 1,Nstep
c
          ncf(3) = ncf(3) + 1
          NTERM = NTERM + 1
          ITERM(1,NTERM) = 0
          ITERM(2,NTERM) = iphase
          ITERM(3,NTERM) = 0
          ITERM(4,NTERM) = -ncf(2) - ncf(3)
          COEF(NTERM) = one
c
        enddo
C
        RETURN
        END
C
        subroutine MSLCDE(iphase,t,y,ny,p,np,frhs,nfrhs,iferr)
c
c           computes the right hand sides of the (delay) differential equations.
c
```

```
c     ******************************************************************
      implicit double precision (a-h,o-z)
c
      parameter  (zero=0.d0,half=0.5d0,one=1.d0,two=2.d0)
c
      dimension  y(ny),p(np),frhs(nfrhs)
c
      integer   sigmax, sigmau
      parameter (Nstep=5)
      parameter (tF=5.d0,x0=1.d0,tau=1.d0)
      dimension xstep(1:Nstep),ustep(1:Nstep)
c
c     ******************************************************************
C   ----SET FUNCTION ERROR FLAG.
      IFERR = 0
c
C   ----DEFINE DELAY VALUES FOR STATE AND CONTROL
      sigmax = nint(2.d0*tau)
      sigmau = nint(1.d0*tau)
c
C   ----DEFINE DIFFERENTIAL EQUATION VARIABLES
c
C   ::INITIALIZE STATE AND CONTROL VECTORS
      xstep(1:Nstep) = y(1:Nstep)
      ustep(1:Nstep) = y(Nstep+1:2*Nstep)
c
C   ----DEFINE STATE AND CONTROL AT EACH STEP
      krhs = 0
      steploop: do kstep = 1,Nstep
c
        xk = xstep(kstep)
        uk = ustep(kstep)
C
C   ----DEFINE DELAY VARIABLES AND FUNCTIONS AT EACH STEP
        if(kstep.le.sigmax) then
         xkm2 = 1.d0
        else
         xkm2 = xstep(kstep - sigmax)
        endif
C
        if(kstep.le.sigmau) then
         ukm1 = 1.d0
        else
         ukm1 = ustep(kstep - sigmau)
        endif
c
C   ----DEFINE RHS FUNCTIONS
        krhs = krhs + 1
        frhs(krhs) = xkm2 + ukm1
c
      cycle steploop
      enddo steploop
c
C   ----DEFINE QUADRATURE INTEGRANDS
      quadloop: do kstep = 1,Nstep
c
        xk = xstep(kstep)
        uk = ustep(kstep)
c
        krhs = krhs + 1
        frhs(krhs) = xk**2 + uk**2
c
```

```
      enddo quadloop
c
      RETURN
      END
```

A.2 Matlab Files for Graphing $\mathbb{SOCX}$ Output

The following programs are designed to graph the output featured in the *Optimal Control Analysis Grids*.

A.2.1 Code for Standard Optimization Problems

Analysis grids for standard optimization problems list the grid point number, time value, state variables, and then algebraic variables (control, delayed state, and delayed control variables). A sample analysis grid for the simple mixed delay problem solved with the optimal control delay formulation is featured below.

Table A.1: Standard Optimal Control Analysis Grid for Eq. (4.1)

```
----------------------------------------------------------------------------
Gridpt    TIME          X(T)          U(T)          Y1(o1)        U1(o2)
----------------------------------------------------------------------------
 1    0.0000000E+00  1.0000000E+00 -3.3085118E+00  1.0000000E+00 1.0000000E+00
 2    4.1666667E-02  1.0833333E+00 -3.1949686E+00  1.0000000E+00 1.0000000E+00
 3    8.3333333E-02  1.1666667E+00 -1.5291373E-11  1.0000000E+00 1.0000000E+00
```

loadSOCX.m

```
function loadDatar(A,nx,nu,ndu,ndy)


format long
%Loading the matrix A
```

```
%   A  = matrix of output data = [x1-xn, u1-um, ud1-udq, yd1-ydr]
%   x  = states
%   u  = control
%   ud = delayed control
%   yd = delayed states

%Defining corresponding indices
%   nx  = number of states
%   nu  = number of controls
%   ndu = number of delayed controls
%   ndy = number of delayed states

%Number of columns and rows for graphing matrix
m=1; n=1;

var=['Y     ','U     ','U2(o1)', 'Y3(o2)'];
grid=A(:,1);          %Extract grid points
t=A(:,2);             %Extract time vector

A=A(:,3:end);         %Redefine A as [x, u, ud, yd]
J=find(grid==1);

for i=1:4
    j=1;    %Used to index subplots
    f=figure;
    set(f,'name',var(6*i-5:6*i),'numbertitle','off')
```

```
for k=1:length(J)
    %plots states x
    if i==1 && nx > 0
        if k<length(J)
            T=t(J(k):J(k+1)-1);
            B=A(J(k):J(k+1)-1,1:nx);
            subplot(n,m,j)
            plot(T,B,'LineWidth',2);set(gca,'Fontname',...
                'Timesnewroman','Fontsize',16)
            title(j,...
            'FontWeight','bold')
            j=j+1;
        end
        if k==length(J)
           B=A(J(k):end,:);
           T=t(J(k):end);
           subplot(n,m,j)
           plot(T,B(:,1:nx),'LineWidth',2);set(gca,'Fontname',...
               'Timesnewroman','Fontsize',16)
           title(j,...
           'FontWeight','bold')
        end
    end
    %plots controls u
     if i==2 && nu > 0
        if k<length(J)
           T=t(J(k):J(k+1)-1);
           B=A(J(k):J(k+1)-1,nx+1:nx+nu,'LineWidth',2);set(gca,...
               'Fontname','Timesnewroman','Fontsize',16)
           subplot(n,m,j)
           plot(T,B,'LineWidth',2);set(gca,'Fontname',...
               'Timesnewroman','Fontsize',16)
           title(j,...
           'FontWeight','bold')
           j=j+1;
        end
        if k==length(J)
           B=A(J(k):end,:);
           T=t(J(k):end);
           subplot(n,m,j)
           plot(T,B(:,nx+1:nx+nu),'LineWidth',2);set(gca,...
               'Fontname','Timesnewroman','Fontsize',16)
           title(j,...
           'FontWeight','bold')
        end
     end
    %plot delayed controls ud
    if i==3 && ndu > 0
       if k<length(J)
          T=t(J(k):J(k+1)-1);
          B=A(J(k):J(k+1)-1,nx+nu+1:nx+nu+ndu);
          subplot(n,m,j)
          plot(T,B,'LineWidth',2);set(gca,'Fontname',...
              'Timesnewroman','Fontsize',16)
          title(j,...
           'FontWeight','bold')
          j=j+1;
       end
        if k==length(J)
           B=A(J(k):end,:);
           T=t(J(k):end,:);
           subplot(n,m,j)
           plot(T,B(:,nx+nu+1:nx+nu+ndu),'LineWidth',2);set(gca,...
               'Fontname','Timesnewroman','Fontsize',16)
           title(j,...
           'FontWeight','bold')
        end
    end
    %plot delayed states yd
    if i==4 && ndy > 0
       if k<length(J)
          T=t(J(k):J(k+1)-1);
          B=A(J(k):J(k+1)-1,nx+nu+ndu+1:end);
          subplot(n,m,j)
          plot(T,B,'LineWidth',2);set(gca,'Fontname',...
              'Timesnewroman','Fontsize',16)
          title(j,...
          'FontWeight','bold')
          j=j+1;
       end
        if k==length(J)
```

```
                B=A(J(k):end,:);
                T=t(J(k):end,:);
                subplot(n,m,j)
                plot(T,B(:,nx+nu+ndu+1:end),'LineWidth',2);set(gca,...
                    'Fontname','Timesnewroman','Fontsize',16)
                title(j,...
                'FontWeight','bold')
            end
        end
    end
end
```

A.2.2 Code for Method of Steps Problems

Analysis grids for method of steps problems list the grid point number, time value, state variables at each step and then algebraic control variables at each step. Delayed variables are not a part of the solution output for MOS. A sample analysis grid for the simple mixed delay problem solved with a 5-step method of steps formulation is featured in Table A.2.

Table A.2: MOS Optimal Control Analysis Grid for Eq. (4.1)

```
-----------------------------------------------------------------------------------------------
Gridpt       TIME           X001          X002          X003          X004          X005
                            U001          U002          U003          U004          U005
-----------------------------------------------------------------------------------------------
   1      0.0000E+00     1.0000E+00    3.0000E+00    7.1462E-01    4.1449E-01    9.4185E-01
                        -3.6973E+00   -2.8734E+00   -1.7268E+00   -9.3309E-01   -2.3108E-01
   2      1.0000E+00     3.0000E+00    7.1462E-01    4.1449E-01    9.4185E-01    9.2432E-01
                        -2.8734E+00   -1.7268E+00   -9.3309E-01   -2.3108E-01    0.0000E+00
```

loadMOS.m

```
function [T,x,u]=loadmosr(A,t,tf,nrx,nru,nx,nu)


%Loading the matrix A

%A  = matrix of output data = [x1...xn; u1...um] with all grids stacked

%   x  = states

%   u  = control


%Defining corresponding indices

%t   = time grid from MOS output

%tf  = final time for problem

%nrx = number of rows corresponding to states

%nru = number of rows corresponding to controls

%nx  = number of states

%nu  = number of controls


%Define number of grid points

ngp = length(t);


%Initialize state and control vector

X=[];

U=[];


%Separate states and controls in matrix A

for k=1:ngp

    [m,n] = size(A(1:nrx,:));

    X=[X; reshape(A(1:nrx,:)',1,m*n)];
    [m,n] = size(A(nrx+1:nrx+nru,:));
    U=[U ;reshape(A(nrx+1:nrx+nru,:)',1,m*n)];
    A(1:nrx+nru,:)=[];
end

%Reorder states at each step as columns x1, x2, ...
[m,n]=size(X);
x=[];
for i = 1:n/nx
    x=[x;X(:,1:nx)];
    X(:,1:nx)=[];
end

%Reorder controls at each step as columns u1, u2, ...
[m,n]=size(U);
u=[];
for i = 1:n/nu
    u=[u;U(:,1:nu)];
    U(:,1:nu)=[];
end

%Initialize time
T=[];
for i=1:(tf/t(end))
    %Shift time interval
    T=horzcat(T,t'+(i-1)*t(end));
```

```
end

%Plot states
[m,n]=size(x);
for i=1:n
    color = [rand rand rand];
    figure
    title(['State' num2str(i)])
    plot(T,x(:,i),'Color',color,'LineWidth',2);set(gca,'Fontname',...
        'Timesnewroman','Fontsize',16)
end

%Plot controls
[m,n]=size(u);
for i=1:n
    color = [rand rand rand];
    figure
    title(['Control' num2str(i)])
    plot(T,u(:,i),'Color',color,'LineWidth',2);set(gca,'Fontname',...
        'Timesnewroman','Fontsize',16,'Linewidth',1)
end
```

Appendix B

Analytic Solution for the Control Delay Problem

In Section 5.1 we apply the method of steps to remove the delay in order to determine an analytic solution to the simple control delay test problem in Eq. (5.1) and the EIC simple control delay test problem in Eq. (5.2). An alternate approach to obtain the solution would be to explicitly compute the necessary conditions as outlined in [65, 66]. The computation of the solutions for the original and regularized control delay test systems are outlined below.

B.1 Analytic Solution to the Original Simple Control Delay Test Problem

The Hamiltonian associated with Eq. (5.1) is defined

$$\mathcal{H}[x(t), u(t), v(t), \lambda(t), t] = \frac{1}{2}(x^2 + u^2) + \lambda v, \tag{B.1}$$

where $v(t) = u(t-1)$. The necessary conditions are

$$\dot{x}(t) = u(t-1), \tag{B.2}$$

$$\dot{\lambda}(t) = -x, \tag{B.3}$$

$$0 = u(t) + \chi_{[0,4]}\lambda(t+1)H_v(t+1) \implies u = -\chi_{[0,4]}\lambda(t+1), \tag{B.4}$$

$$x(0) = 1, \quad \lambda(5) = 0, \quad \text{and} \quad u(t-1) = 1, \ [-1, 0). \tag{B.5}$$

We compute the solution in pieces and enforce continuity of the solution at $t = 1$.

- State solution on $[0,\ 1]$

$$\dot{x}(t) = 1 \implies x(t) = t + 1. \tag{B.6}$$

- State and costate solution on $[1,\ 5]$

$$\dot{x}(t) = u(t-1) = -\lambda(t-1+1) = -\lambda(t), \tag{B.7}$$

$$\dot{\lambda}(t) = -x. \tag{B.8}$$

Combining the above equations we have the second order ODE in λ defined

$$\ddot{\lambda}(t) = \lambda(t), \tag{B.9}$$

$$\implies \lambda(t) = c_1 e^t + c_2 e^{-t} \quad \text{and} \quad x(t) = -c_1 e^t + c_2 e^{-t}. \tag{B.10}$$

Now using the boundary conditions we have the following coefficients

$$c_1 = \frac{-2}{e(1+e^8)} \quad \text{and} \quad c_2 = \frac{2e^9}{(1+e^8)}. \tag{B.11}$$

- Costate on [0, 1]

$$\dot{\lambda}(t) = -x = -(t+1). \tag{B.12}$$

To ensure that $\lambda(t)$ is continuous across the boundary we have

$$\lambda(1^-) = \lambda(1^+) = \frac{-2}{(1+e^8)} + \frac{2e^8}{(1+e^8)}. \tag{B.13}$$

Now the costate is determined by

$$\lambda(t) = \lambda(1^+) - \int_1^t \tau + 1 \, d\tau = \lambda(1^+) + \frac{3}{2} - \frac{t^2+2t}{2}. \tag{B.14}$$

- Control on [0, 5]

  Substituting values for $\lambda(t)$ in (B.4) defines the values for the optimal control

$$u(t) = \begin{cases} -\lambda(t+1), & t \in [0,\ 4] \\ 0, & t \in [4,\ 5] \end{cases}. \tag{B.15}$$

B.2 Analytic Solution to the Regularized Simple Control Delay Test Problem

The Hamiltonian associated with Eq. (5.2) is defined

$$\mathcal{H}[x_1(t), x_2(t), v(t), \lambda_1(t), \lambda_2(t), z(t), t] = \frac{1}{2}(x_1^2 + x_2^2 + \varepsilon z^2) + \lambda_1 v + \lambda_2 z, \tag{B.16}$$

where $v(t) = x_2(t-1)$. The necessary conditions are

$$\dot{x}_1(t) = x_2(t-1), \tag{B.17}$$

$$\dot{x}_2(t) = z, \tag{B.18}$$

$$\dot{\lambda}_1(t) = -x_1, \tag{B.19}$$

$$\dot{\lambda}_2(t) = -x_2 - \chi_{[0,\ 4]}\mathcal{H}_v = -x_2 - \chi_{[0,\ 4]}\, \lambda_1(t+1), \tag{B.20}$$

$$0 = \varepsilon z(t) + \lambda_2 \implies z(t) = -\frac{\lambda_2}{\varepsilon}, \tag{B.21}$$

with the following boundary conditions

$$x_1(0) = 1, \quad \lambda_1(5) = 0, \quad \lambda_2(0) = \lambda_2(5) = 0, \quad \text{and} \quad x_2(t-1) = 1,\ [-1, 0). \tag{B.22}$$

- State x_1 solution on $[0,\ 1]$

$$\dot{x}_1(t) = 1 \implies x_1(t) = t + 1 \tag{B.23}$$

- State $x_2(t)$ and costate $\lambda_2(t)$ solution on $[4,\ 5]$

  Since the χ function is zero on this interval we have the boundary conditions $\lambda_2(5) = 0$ and $v(5) = x_2(4) = 0$. The governing equations for $x_2(t)$ and $\lambda_2(t)$ on this interval are defined

$$\dot{x}_2(t) = -\frac{\lambda_2}{\varepsilon}, \tag{B.24a}$$

$$\dot{\lambda}_2(t) = -x_2. \tag{B.24b}$$

Combining equations we have the following second order ODE in defined in $\lambda_2(t)$

$$\ddot{\lambda}_2(t) = \frac{\lambda_2}{\varepsilon}, \tag{B.24c}$$

$$\implies \lambda_2(t) = c_3 e^{t/\varepsilon} + c^4 e^{-t/\varepsilon} \quad \text{and} \quad x_2(t) = -\frac{c_3}{\varepsilon} e^{t/\varepsilon} + \frac{c_4}{\varepsilon} e^{-t/\varepsilon}. \tag{B.24d}$$

Now applying the boundary conditions we find that $c_3 = c_4 = 0$. Thus, on [4, 5] $x_2(t) = \lambda_2(t) = 0$ as expected.

- State x_1 and costate λ_2 can be determined on $[0,\ 4]$

Let

$$\hat{x}_1(t) = x_1(t) \text{ then } \hat{x}_1(0) = x_1(2) = 2.$$

We can combine equations to form the 2-dimensional second order system

$$\ddot{\hat{x}}_1(t) = \dot{x}_2 = (t + 1 - 1) = \dot{x}_2(t) = -\frac{\lambda_2}{\varepsilon}, \tag{B.25a}$$

$$\ddot{\lambda}_2(t) = -\dot{x}_2 - \dot{\lambda}_1(t + 1) = \frac{\lambda_2}{\varepsilon} + x_1(t + 1) = \frac{\lambda_2}{\varepsilon} + \hat{x}_1. \tag{B.25b}$$

Let $q_1 = \hat{x}_1$, $q_2 = \dot{\hat{x}}_1$, $q_3 = \lambda_2$, and $q_2 = \dot{\lambda}_2$ then we have the equivalent 4-dimensional first order system

$$\dot{q}_1(t) = q_2, \tag{B.25c}$$

$$\dot{q}_2(t) = -\frac{1}{\varepsilon} q_3, \tag{B.25d}$$

$$\dot{q}_3(t) = q_4, \tag{B.25e}$$

$$\dot{q}_4(t) = q_1 + \frac{1}{\varepsilon} q_3, \tag{B.25f}$$

with the boundary conditions defined $q_1(0) = 2$, $q_2(4) = q_3(0) = q_4(0) = 0$. Solving the above system fully determines the optimal solution on $[0,\ 5]$. Note that as ε nears 0 the system becomes ill-conditioned.