Applied Numerical Mathematics 108 (2016) 185–203

Contents lists available at ScienceDirect

Applied Numerical Mathematics

www.elsevier.com/locate/apnum

Solving optimal control problems with control delays using direct transcription

John T. Betts [a], Stephen L. Campbell [b,*], Karmethia C. Thompson [b]

[a] *Applied Mathematical Analysis LLC, USA*
[b] *Department of Mathematics, North Carolina State University, Raleigh, NC, USA*

ARTICLE INFO

Article history:
Received 6 January 2015
Received in revised form 24 September 2015
Accepted 17 December 2015
Available online 8 June 2016

Keywords:
Delay
Optimal control
Direct transcription
Numerical methods

ABSTRACT

The numerical treatment of optimal control problems with state and control delays is important in a wide variety of scientific and technical applications. Solutions to these types of problems are difficult to obtain via analytic techniques since the system may be nonlinear and subjected to complicated inputs and constraints. There are several numerical methods available to compute the solutions of optimal control problems without delays. One such popular method is direct transcription. Although the numerical solutions of optimal control delay problems are important, less literature and software exists in this area. A general purpose industrial grade direct transcription code that can handle optimal control problems with both state and control constraints and delays is under development. Control delays pose a special challenge. A new technique for treating control delays when using a direct transcription approach is investigated in this paper.

© 2016 IMACS. Published by Elsevier B.V. All rights reserved.

1. Introduction

Optimal control theory is frequently applied to solve problems governing electrical, mechanical, and industrial processes. Often times, the physical nature of these processes introduce time delays into the system. In engineering systems time delays are often present due to measurement, transmission and transport lags, computational delays, or unmodeled inertias of system components [38]. In control systems, it is known that time delays can degrade a system's performance and even cause system instability [11]. The presence of delays often poses several challenges in finding an optimal solution. Necessary conditions become harder to form.

Whether or not delays are present, complex optimal control problems require numerical solutions [36]. There are several numerical approaches that use techniques of nonlinear programming to compute the solutions of nondelay optimal control problems. One such method commonly applied is Direct Transcription (DT) [2]. Direct transcription methods transcribe the entire optimal control system into a large sparse nonlinear programming problem (NLP) by a discretization scheme. To obtain the optimal solution the resulting NLP is solved on a sequence of refined meshes until a desired tolerance is reached. DT methods, and in particular, methods involving orthogonal collocation have become popular in several areas due to their high accuracy in approximating non-analytic solutions with relatively few discretization points [26]. Additionally, DT approaches are popular since several constraints on state and control variables can be accommodated. Encountering complex necessary conditions and discontinuous multipliers prove disadvantageous for some methods. However, these issues

[*] Corresponding author. Tel.: +1 919 51503300.
E-mail addresses: john.betts@comcast.net (J.T. Betts), slc@math.ncsu.edu (S.L. Campbell), kcthomps@ncsu.edu (K.C. Thompson).

http://dx.doi.org/10.1016/j.apnum.2015.12.008
0168-9274/© 2016 IMACS. Published by Elsevier B.V. All rights reserved.

are avoided with DT since DT neither relies on necessary conditions nor multipliers of the control problem. Also a DT formulation can handle problems with advances just as easily as delays since it is similar in some ways to a boundary value solver and in a boundary value context advances and delays can often be treated the same. Provided one adapts sophisticated mesh refinement strategies with state of the art sparse NLP solvers, a DT approach can often provide solutions to many problems that are otherwise difficult or impossible to solve with other methods [5].

There are several optimal control problem solvers currently available for use. Pseudospectral Optimal Control Solver ($\mathcal{PSOPT}$) is an open source optimal control package written in C++ that uses direct collocation methods, which include pseudospectral as well as local discretizations [1]. General Pseudospectral OPtimal Control Software (GPOPS) is an open source MATLAB optimal control software that implements the Gauss and Radau hp-adaptive pseudospectral methods [33]. The latest GPOPS II requires a license. When using either of these optimal control packages, one must also incorporate NLP solvers such as Sparse Nonlinear OPTimizer (SNOPT) [18] or Interior Point Optimizer (IPOPT) [37]. As noted, both of these softwares employ pseudospectral schemes which solve optimal control problems by approximating the time-dependent variables using orthogonal polynomials. A basic pseudospectral method is typically employed as a p-method where a single segment is used, and convergence is achieved by increasing the degree p of the polynomial [16]. 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. hp methods use several intervals with a polynomial on each subinterval.

Since DT has worked well on a wide variety of optimal control problems without delays, we are in the process of extending the approach to optimal control systems with time delays. Our results are being implemented in SOCX (Sparse Optimal Control Extended), a general purpose industrial grade software capable of solving optimal control problems with both state and control delays and state and control constraints and is available from Applied Mathematical Analysis. SOCX is part of the Sparse Optimization Software (SOS). However, the solutions of control delay problems have been problematic. Previous papers have discussed the successful solution of state delay problems [6,8] and the possibilities of making the solution of control delay problems more robust by just modifying the cost [7]. Cost modification was helpful on some problems but did not prove to be of general use. The solutions of control delay problems by DT remain a challenge, and requires further investigation.

In this paper we discuss the use of an exogenous control parameterization technique to treat optimal control problems with control delays. The results should be of use to anyone interested in solving optimal control problems with control delays by DT. Section 2 will discuss the main direct transcription algorithm that we use. In Section 3 we describe a difficulty that is intrinsic to the use of DT with control delays on nonuniform grids. Then a new method we call EIC for trying to overcome this intrinsic difficulty is discussed in Section 4. EIC stands for Exogenous Input Control since we add an exogenous input w to the problem. Computational examples are in Section 5. EIC is then analyzed in Section 6. Finally, conclusions and remarks can be found in Section 7.

The method of steps (MOS) is a way to analyze some delayed ordinary differential equations (ODEs) as non-delayed boundary value problems. To illustrate, suppose that we have $x' = f(x(t), x(t-h), t)$ on $[0, T]$, where $h > 0$ and $N = T/h$ is an integer. Let $x_i(t) = x(t-h)$ for $t \in [ih, (i+1)h]$. Then by stacking the x_i into a bigger vector the original problem is transformed into a higher dimensional boundary value problem without delays on $[0,\ h]$ [24]. Some of the boundary conditions come from the requirement that $x_i(h^-) = x_{i+1}(0^+)$ for $i = 0, \dots N-1$. The idea can be extended to multiple constant delays as long as they are all multiples of some h that divides T [23]. Many texts use MOS to easily show the existence of solutions for some classes of retarded delay equations. We note only [24]. It can also be used as the basis of numerical methods such as in [23] where the uniform grids are chosen compatible with the delays. We shall only use the MOS for looking at some special test problems either to find out what the optimal solution looks like or for purpose of analyzing a specific example. As noted our primary interest is in software that does not require special assumptions on the delays and also can work with variable grids and can return useful control laws. For our purposes, the need to determine an analytic solution and define the necessary conditions is not required. We should note that we are not saying that necessary conditions are not sometimes useful as a way to solve optimal control problems or evaluate the quality of the solution of an optimal control problem. Necessary conditions are an important part of optimal control.

2. General direct transcription algorithm

A DT algorithm begins by rewriting the dynamics of the properly formulated optimal control problem as a differential algebraic equation (DAE) [13,29] or delay differential algebraic equation (DDAE). Multiple varying delays and state and control delays can be accommodated although for simplicity of notation we discuss just one or two delays in this paper. We also omit most state and control inequality and equality constraints from our notation so that we can focus on the key issues even though our problem formulation accepts them. We also do not include non-dynamic parameters in this discussion. Here we consider the optimal control delay problem (OCDP)

$$J = \phi(t_f) + \int_{t_0}^{t_f} L(x(t), u(t), x(\omega(t)), u(\eta(t)), t)\, dt, \tag{1a}$$

$$\dot{x} = f(x(t), x(\omega(t)), u(t), u(\eta(t)), t), \quad t_0 \le t \le t_f \tag{1b}$$

$$0 = g(x(t), x(\omega(t)), u(t), u(\eta(t)), t), \quad t_0 \le t \le t_f \tag{1c}$$

$$x = \alpha(t), \quad -r \le t < t_0 \tag{1d}$$

$$u = \beta(t), \quad -s \le t < t_0 \tag{1e}$$

$$x_0 = q, \tag{1f}$$

with $\phi(t_f) = \phi(x(t_f), u(t_f), t_f)$, and time delay functions $\omega(t)$, $\eta(t)$. In the case of a single constant delay in the state and the control, $\omega(t) = t - r$ and $\eta(t) = t - s$. The problem features a cost (1a), delay differential equation (1b), and prehistory functions (1d) and (1e). If (1c) is present then (1b) and (1c) form a DDAE. All states and control are allowed to be vector valued. Variables x that appear differentiated are called differential variables. Variables u are called algebraic variables. If (1c) is present, then u includes both what the user thinks of as algebraic state variables and control variables. However, to SOS they are just algebraic variables. This enables SOS to solve some problems with DAEs that the user thinks are higher index [17]. SOS can solve a wide variety of delayed state optimal control problems [7,9,10]. Our concern here is strictly with control delays. The software can accommodate time varying delays [15,25,27,28] but in this paper we take the delays constant. Note that the chattering behavior discussed here for nonuniform grids and constant control delays can also occur with uniform grids and non-constant control delays.

Here r and s are taken to be positive. Less common, systems of differential equations sometimes model phenomena requiring knowledge of the future state and/or control variables. If $r, s < 0$ and (1d) and (1e) are replaced by post-history functions

$$x = a(t), \quad t_f \le t < t_f + r \tag{2a}$$

$$u = b(t), \quad t_f \le t < t_f + s \tag{2b}$$

and (1) is considered a time advanced optimal control problem. For simplicity of exposition we will only consider delays but the situation for advances is similar [9].

The algorithm used here reformulates the optimal control problem by introducing pseudovariables y, v for $x(\omega(t))$, $u(\eta(t))$ and then enforcing their relationship to x, u. Thus (1b) and (1c) is internally considered to be a DDAE of the form

$$\dot{x} = f(x(t), u(t), y(t), v(t), t), \tag{3a}$$

$$0 = g(x(t), u(t), y(t), v(t), t), \tag{3b}$$

$$0 = y(t) - x(\omega(t)), \tag{3c}$$

$$0 = v(t) - u(\eta(t)). \tag{3d}$$

Thus the software is always considering DAEs even if (1c) is missing and the dynamics are a delayed ordinary differential equation. This is advantageous in working with more complex problems such as neutral delays [9].

Implementation of the DT approach begins with constructing mesh points t_i by subdividing the time interval into segments $t_0 = t_1, t_2, \ldots, t_N = t_f$. Using either the second order trapezoidal method (TR) or the fourth order Hermite–Simpson (HS) discretization, (1) is transcribed into an NLP. The differential equation in (3a) and (1c) is approximated by a set of algebraic constraints. Here we will just discuss the TR method since the presentation is more straightforward but our analysis and computational studies suggest the situation is the same for the HS method. Additional computation experience with HS and other test problems can be found in [35]. For any function z of t we let z_i be the numerical approximation of $z(t_i)$. If z is a function of other variables than t, then z_i means that all variables are replaced by their numerical approximation at t_i. z can be any of $x, y, u, v, \omega, \eta, f, g$.

The equations (3c) and (3d) are enforced on the grid by adding constraints to the NLP based on interpolation. At a given grid point t_i, the grid interval $\mathcal{I}_i = [t_{j_i},\ t_{j_i+1})$ containing $\omega(t_i)$ is determined. Similarly at a given grid point t_i, the grid interval $\mathcal{K}_i = [t_{k_i},\ t_{k_i+1})$ containing $\eta(t_i)$ is determined. The delayed control v_i is given as a linear interpolant of u_{k_i}, u_{k_i+1}. A standard implementation of TR would also give y_i as a linear interpolant of x_{k_i}, x_{k_i+1}. We shall refer to this approach as STR. However, our implementation provides a different discretization. It treats the delayed states the same for both HS and TR by giving y_i as a linear interpolant of $x_{k_i}, \dot{x}_{k_i}, x_{k_i+1}, \dot{x}_{k_i+1}$. The $\dot{x}_j$ are given by evaluating the right hand side of (3a). This formulation will be referred to as XTR.

Let $f_i = f(x_i, u_i, y_i, v_i, t_i)$ and $g_i = (x_i, u_i, y_i, v_i, t_i)$ for f, g from (3a) and (3b). For (3) the discretization of (3a) is the implicit TR method,

$$x_{i+1} - x_i - \frac{h_i}{2}(f_i + f_{i+1}) = 0, \tag{4}$$

where $h_i = t_{i+1} - t_i$. Equation (1c) becomes just $g_i = 0$. Let $\delta_i = (\eta(t_i) - t_{k_i})/h_{k_i}$ and $\beta_i = (\omega(t_i) - t_{j_i})/h_{j_i}$. Then equation (3d) becomes

$$v_i - \left(\delta_i u_{k_i+1} + (1 - \delta_i) u_{k_i}\right) = 0. \tag{5}$$

For STR (3c) becomes

$$y_i - \left(\beta_i x_{j_i+1} + (1-\beta_i) x_{j_i}\right) = 0. \tag{6}$$

Note that in (5) the linkage between v_i, u_i depends solely on the grid. However, the constraints in (3c) intertwine the variables since the x_i values are interlinked with other values by (4) because of the delayed terms in the f_i. This interlinkage is even stronger when XTR is used since then the f_i appear directly in the interpolation in place of the $\dot{x}_i$. Thus for both HS and XTR we have

$$y_i = \gamma_{i1} x_{j_i} + \gamma_{i2} x_{j_{i+1}} + \gamma_{i3} f_{j_i} + \gamma_{i3} f_{j_{i+1}} \tag{7}$$

with γ_{ik} from the HS interpolant instead of (6).

When SOS is used as an integrator, the control part of the algebraic variables becomes a given function and we have only a state delay. Then as pointed out in [24] if the state delays are bounded away from zero, and the mesh is finer that this bound, then HS, TR, and XTR all can be considered as Runge–Kutta methods applied to an ODE which is the method of steps formulation of the ODE and they will be integrators of orders 4, 2, and 2 respectively. The difficulties we discuss concern their use in the optimal control setting with control delays.

The resulting NLP is large and sparse. Both sparse linear algebra and a sophisticated mesh refinement strategy must be used. After the NLP is solved, the mesh-refinement algorithm assesses the accuracy of the solution and refines the grid if necessary. The goal of the mesh-refinement procedure is to select the number and location of the grid points in the new mesh to minimize the maximum discretization error.

The basic idea of the grid refinement is discussed in [3,4]. We just quickly summarize it here. After an iteration the error on each subinterval is estimated. The order of the approximation on each subinterval is also approximated. This may be less than the theoretical order of the discretization due to activation of state constraints or stiffness of the dynamics for example. Then an optimization problem is done to estimate where adding one more grid point would produce the greatest drop in error. This process is repeated until either tolerances are met, or the maximum number of points is added (default is one less than the current grid) or the maximum number of points have been added to one subinterval (default is 4). Then the NLP problem is solved on the new grid and the process is repeated. One goal is thus the equidistribution of the error across the subintervals before the grid gets too fine. Another goal is to try and avoid repeated iterations on large grids caused by adding just a few points and still not meeting tolerances. We found this was often caused by order reduction in certain parts of the interval of definition say by stiffness. The frequent presence of order reduction along parts of the time interval is why we do not treat the order as given by the discretization. The result for some problems is highly nonuniform grids. This is especially important for problems where there are local regions of great activity and then longer periods with less activity [3,4,12].

To date, we cannot handle state dependent state delays with DT. In [8], a delay partial differential equation example is presented with a time varying state delay. Additionally, time delay and time advance variables are permitted in a single problem. However, such variables are restricted from changing orientations, that is, a time delayed variable cannot become a time advanced variable.

3. The problem on nonuniform grids

In this section we illustrate some of the computational difficulty that occurs when solving optimal control problems with control delays on nonuniform grids using DT.

3.1. Simplest delayed control problem (SDC)

The presence of the chatter in the delayed control approximations with prior formulations led to the consideration of control delays in the Simplest Delay Control problem (SDC) which also exhibited the chatter. We observed earlier that better solutions were produced whenever the delayed control was a part of the objective function. However, this outcome was unique to the SDC problem. For more complex problems, this technique failed when the grid became nonuniform [7]. However, the SDC problem provides a nice problem to illustrate what is happening. The SDC problem can easily be solved using a method of steps (MOS) formulation but our interest is in developing techniques and algorithms for complex problems where MOS is not suitable and uniform grids are not desirable. MOS is only used here to get a “truth” solution for comparison purposes.

The SDC problem is to minimize

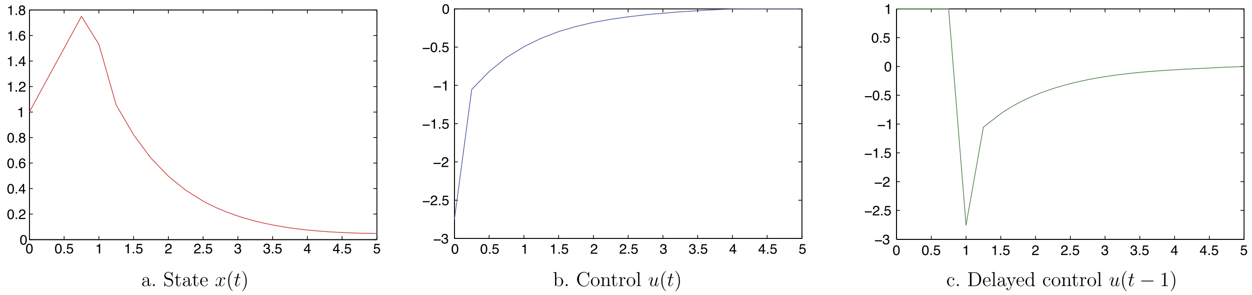
$$J(u) = \int_0^5 x^2(t) + u^2(t)\, dt, \tag{8a}$$

subject to the control delay differential system

$$\dot{x}(t) = u(t-1), \quad 0 \le t \le 5 \tag{8b}$$

$$u(t) = 1, \quad -1 \le t < 0 \tag{8c}$$

$$x(0) = 1. \tag{8d}$$

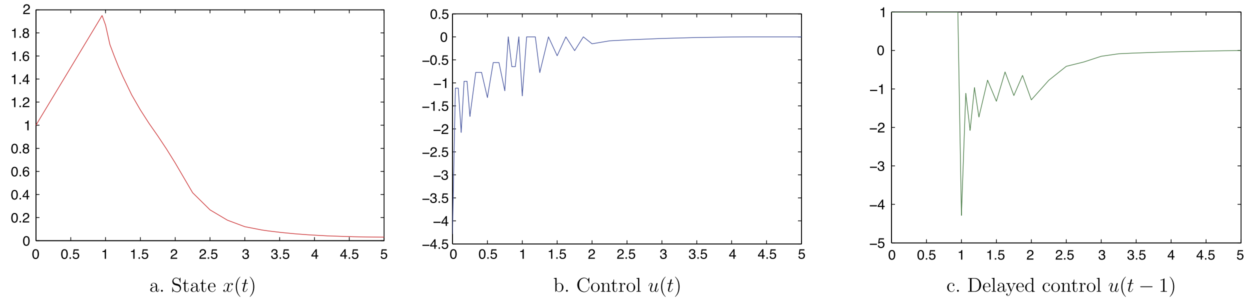


a. State $x(t)$

b. Control $u(t)$

c. Delayed control $u(t-1)$

Fig. 1. Iteration one approximations for (8) on a uniform grid of 21 points.



a. State $x(t)$

b. Control $u(t)$

c. Delayed control $u(t-1)$

Fig. 2. Iteration two approximations for (8).

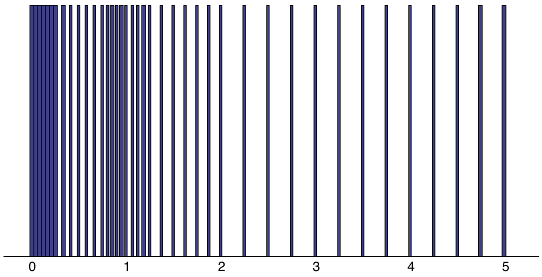


Fig. 3. Grid computed by software for iteration two approximations shown in Fig. 2.

Even though (8) appears to be very simple, it is representative of control problems that have been shown to provide useful models for real physical systems [34].

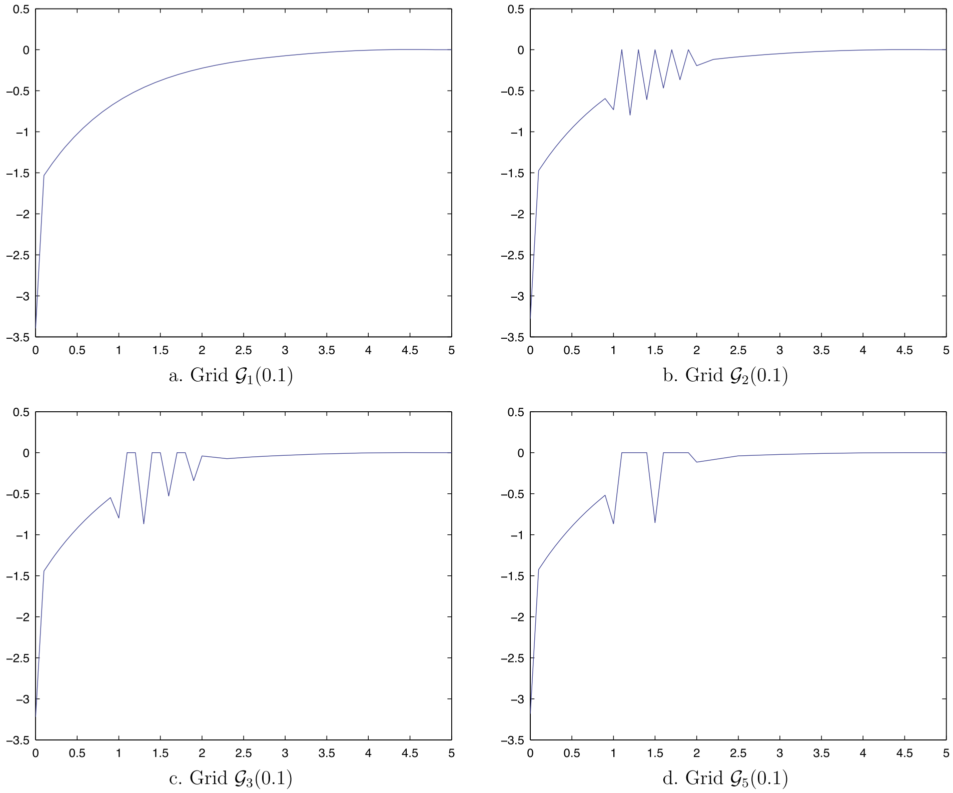
Our numerical solutions are in Fig. 1 and Fig. 2. Fig. 1 shows the numerical solution after one iteration on a coarse uniform grid of 21 points. Fig. 2 shows the solution after the second iteration on a nonuniform grid found by mesh refinement. The new grid had 41 points. Note the smoothness, except at one corner, in the state variables, and the presence of chatter in the control variables on successive iterations. This type of chatter was observed on a number of problems with control delays. The graph of the true solution is in the first three graphs of Fig. 11. Note that with our formulation $u(t)$ and $u(t-1)$ are two different algebraic variables in the DDAE. Accordingly we often graph both of them.

Fig. 3 shows the grid on iteration 2 that the software computed for Fig. 2. Note that it is not uniform. In fact all of the additional grid points have been added in the interval $[0, 2]$ and are most heavily concentrated near $t = 1$ where there is a drop in smoothness and also near $t = 0$.

3.1.1. Special grids

Before examining the use of our approach with varying grids created by the algorithm it is helpful to carefully delineate what is and what is not the cause of the problem and exactly what is helping. The discussion is in terms of the SDC but the behavior is typical.

A careful examination of these test problem results shows that this chatter is due to free control variables. Suppose that a grid is fine and then gets coarser which is often the case. Then there can be several grid points t_j in a single interval $[t_i - \eta(t_i),\ t_{i+1} - \eta(t_{i+1}))$. Some of these v_j will not appear in any constraints and will thus be free. This problem is intrinsic to the use of nonuniform meshes. We shall verify this is the source of the chatter by using specially designed grids and then doing one solve on that grid.



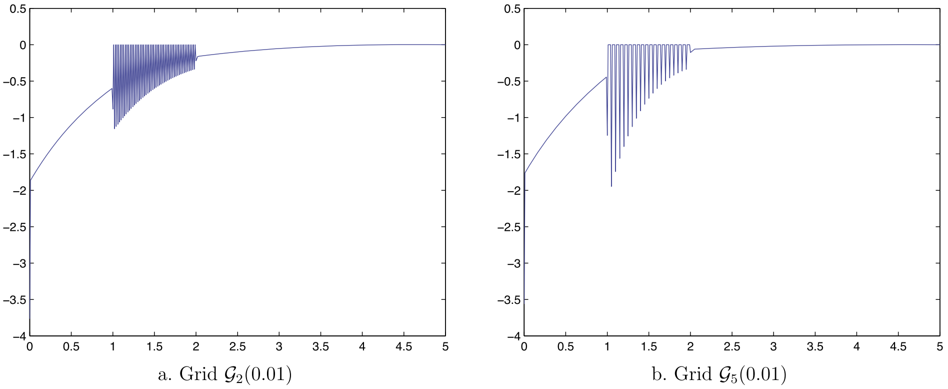
a. Grid $\mathcal{G}_1(0.1)$

b. Grid $\mathcal{G}_2(0.1)$

c. Grid $\mathcal{G}_3(0.1)$

d. Grid $\mathcal{G}_5(0.1)$

Fig. 4. Control for (8) using one iteration on special grids.



a. Grid $\mathcal{G}_2(0.01)$

b. Grid $\mathcal{G}_5(0.01)$

Fig. 5. Control for (8) using one iteration on special grids.

Let $\mathcal{G}_p(h)$ consist of a uniform grid with mesh width h on $[0, 2]$ and a uniform grid with mesh width ph on $[2, 5]$. This grid is designed to create free discretized control variables in $[1, 2]$ with the number of points increasing with p. The value $p = 1$ is a uniform grid. We considered a number of variations but that did not alter the observations. In Fig. 4 we see the effect of the free control variables in the interval $[0, 2]$.

Taking a finer grid does not help. It only makes the control estimate worse as shown in Fig. 5.

Our discussion has focused on the smoothness of the computed control. However, the computed optimal cost is also of interest. Table 1 gives the computed cost from an analytic form of the solution denoted J^* which is possible in this case and the EIC method for two values of ϵ. EIC produced a good estimate.

3.2. Continuous stirred tank reactor problem (CSTR)

While the SDC problem is good for experiments and discussion it is important to make sure the observations and results hold for more realistic applications. The second problem we examine in this paper is based on one from [22]. It is more

Table 1
Computed optimal cost for (8) and the EIC value from (10) for two values of ϵ.

J^*	$\epsilon = 10^{-2}$	$\epsilon = 10^{-4}$
6.330652	6.343906	6.32196

complex than the SDC problem with highly nonlinear dynamics and both a state delay and a control delay. The problem is still of modest size. Also the control appears both delayed and not delayed. The problem is to minimize

$$F = \int_0^{0.2} \|x(t)\|^2 + .01\|u(t)\|^2 \, dt, \tag{9a}$$

with $x = (x_1, x_2, x_3)^T$, $u = (u_1, u_2)^T$, and the Euclidean norm $\|\cdot\|$, subject to the delay differential equations

$$\dot{x}_1(t) = -x_1(t) - R(t, x), \tag{9b}$$

$$\dot{x}_2(t) = -x_2(t) + 0.9u_2(t - s) + 0.1u_2(t), \tag{9c}$$

$$\dot{x}_3(t) = -2x_3(t) + 0.25R(t, x) - 1.05u_1(t)x_3(t - r), \tag{9d}$$

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

$$x_3(t) = -0.02, \quad -r \le t < 0 \tag{9e}$$

$$u_2(t) = 1, \quad -s \le t < 0, \tag{9f}$$

with the function

$$R(t, x) = (1 + x_1(t))\,(1 + x_2(t)) \exp\left(\frac{25x_3(t)}{1 + x_3(t)}\right), \tag{9g}$$

and with initial condition

$$x(0) = (0.49, -0.0002, -0.02)^T. \tag{9h}$$

The bound (9h) plays no role in our solutions. We include it only because it was in the original formulation of [22] where the software needed bounds on variables.

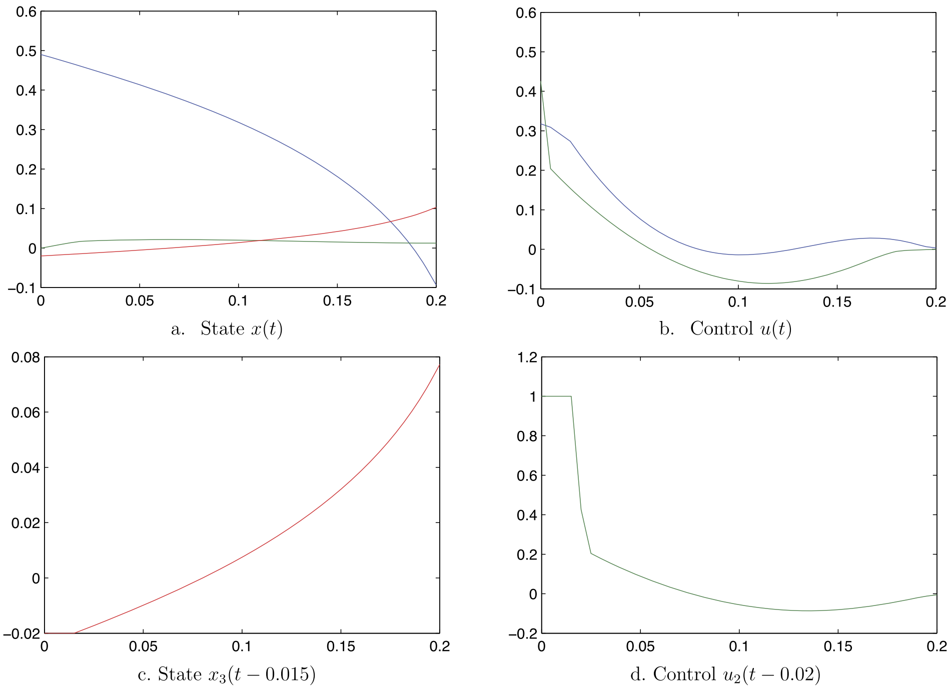
First (9) was solved using SOCX yielding the approximations in Fig. 6 and Fig. 7. The initial grid was uniform with 21 points. Note how much the chatter has propagated in the delayed control variables from the uniform grid iteration one in Fig. 6 to the nonuniform grid in iteration five in Fig. 7. The exact solution is similar to the graph in Figs. 14a and 14b.

4. Exogenous input control method

Previous work shows that the DT numerical solution for state delay optimal control problems with SOCX yields smooth approximations on successive iterations when the solution is smooth. If the solution is piecewise smooth, then the numerical solution will appear piecewise smooth except for t values close to the discontinuity. However, when (1) contains control delay variables, oscillations develop in the controls and their respective time delayed control variables as shown in Figs. 1 and 2. For constant delays, this chatter appeared only when the mesh became nonuniform [7].

Some control delay problems can be successfully solved using uniform grids [14]. Although nonuniform grids pose challenges, they are essential components of highly reliable adaptive methods. Higher accuracy is obtained for fewer grid points on nonuniform grids [30]. In some situations solely relying on uniform grids can increase computation requirements, or make such a solution impossible. 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 numerical solution of optimal control problems with control delays is necessary to ensure a robust and useful software. Only modifying the costs of more complex problems revealed inconsistent results and something more was clearly needed.

Here a new technique called *exogenous input control* (EIC) is used. EIC converts the delayed control to a delayed state in a different problem. In optimal control applications without delays, similar implementations of constrained variables have been taken into account. In [14], a state variable and two additional constraints are imposed on a landing passenger aircraft model to decrease problem solving difficulty. Our use of this idea in the context of optimal control with control delays and DT is new. 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. As a general approach we would augment u by $\dot{u}(t) = q(u(t), w(t), t)$ with a new control variable w. In this first examination of this approach we take $\dot{u} = w$. Whether there are advantages to



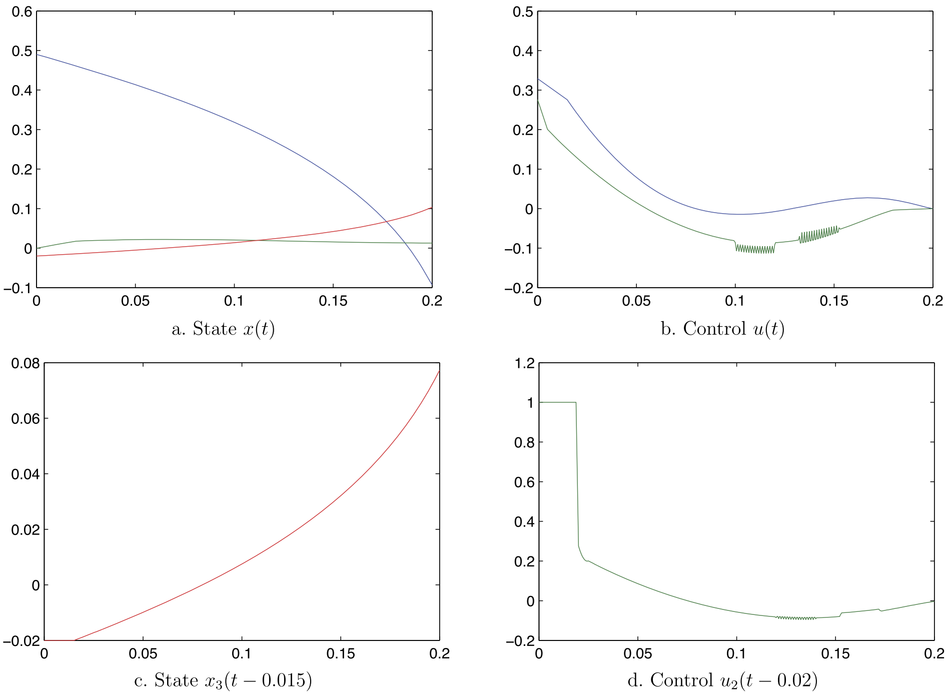
a. State $x(t)$

b. Control $u(t)$

c. State $x_3(t-0.015)$

d. Control $u_2(t-0.02)$

Fig. 6. Iteration one approximations for (9) on a uniform grid of 21 points.



a. State $x(t)$

b. Control $u(t)$

c. State $x_3(t-0.015)$

d. Control $u_2(t-0.02)$

Fig. 7. Iteration five approximations for (9).

using a more general q is a topic for future research. Note that we do not actually care about the value of w since w is only introduced to change how the delayed control is discretized.

It is well known that in order to successfully solve an optimal control problem numerically the controls usually need to either be bounded or to appear in the cost in a nonlinear manner. Previous work on the DT solution of control constrained problems showed that it was often advantageous to add a small multiple of the control to the cost in order to regularize and smooth out the solution if it has jump discontinuities. In this paper we will investigate both bounds and cost weighting of the EIC input w. Note that using EIC means that the solution will be suboptimal. But by weighting w only slightly we make the computed solution only slightly suboptimal as shown later.

Without EIC we see an unacceptably poor solution for the control on a nonuniform grid as shown in Fig. 2. However, the solution without EIC or on a uniform grid can still be helpful in setting up the EIC problem in terms of possible bounds. We shall give some examples. However, space prevents us from fully discussing all of the computational experience we have with EIC. The computational experience reported here will be given in a summary form. More computational detail may be found in [35].

5. Illustration of EIC method

We now examine the effects of using the EIC method by reconsidering the previous examples.

An EIC general reformulation for (8) is,

$$\text{Minimize}\quad \tilde{J} = \int_0^5 x_1^2(t) + x_2^2(t) + \epsilon w^2(t)\, dt, \tag{10a}$$

subject to

$$\dot{x}_1(t) = x_2(t-1), \tag{10b}$$

$$\dot{x}_2(t) = w(t), \tag{10c}$$

$$x_2(t) = 1, \quad -1 \le t < 0 \tag{10d}$$

$$x_1(0) = 1, \tag{10e}$$

with

$$x_2(0) = \rho, \tag{10f}$$

$$a \le x_2 \le b, \tag{10g}$$

$$c \le w \le d, \tag{10h}$$

where ρ, a, b, c, d are optional parameters available to the user.

Looking at the cost functional, (10a) notice that the new differential state $x_2(t)$ has replaced the original algebraic control $u(t)$. Additionally, the new control $w(t)$ has been added. Putting a cost on w puts a cost on u having a large slope and this helps to dampen the chatter, but that does not alter the solution much as shown in Fig. 12. While the solutions in Fig. 2 are not good solutions they can be used to get estimates for the bounds in (10g) and (10h).

For comparison purposes, and to establish a "truth solution," we reformulated both the SDC and CSTR problems as MOS problems without delays and solved them to an accuracy of 10^{-8}.

There are two different mechanisms presented for controlling the oscillation in the control. One is the bounds on w, which are determined from the max and min slopes of the original delayed control u. The other is using ϵ and the control weighting. Figs. 8 and 9 illustrate these two possibilities on (10) for the special test grids.

Figs. 8 and 9 both show an improvement over that of Fig. 4. For this particular computation the bounds on w seemed most effective but that is not always the case.

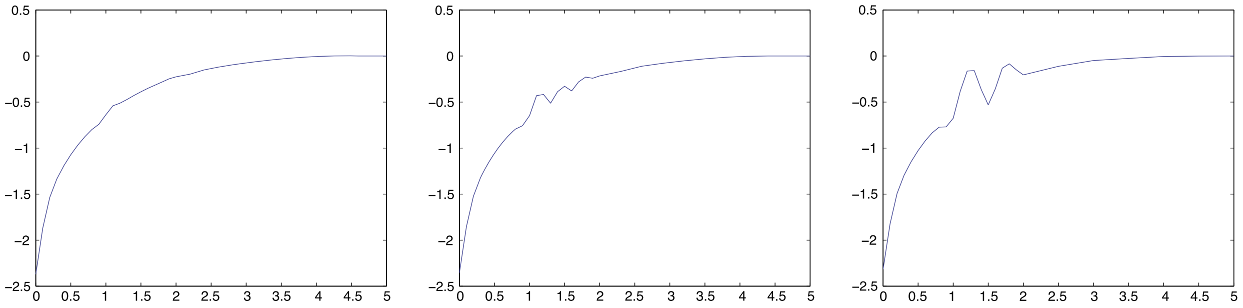
Finally we examine EIC with finer versions of the test grids. Both strategies produced solutions for $p = 2$ that appeared smooth so we will give just $p = 3$ and $p = 5$ in Fig. 10. Looking carefully at Figs. 10c and 10d, we see a small bump at 1 and 2 and an almost invisible chatter on the interval. Using $\epsilon > 0$ produces a smoother solution for u as desired.

From these experiments we can see several things. First, the grid mismatch is the source of the chatter and it gets worse with greater nonuniformity of the grid or finer grids. Secondly, the exogenous control parameterization greatly reduces this chatter. The remaining graphs are all on automatically generated nonuniform grids.

5.1. The EIC variable w

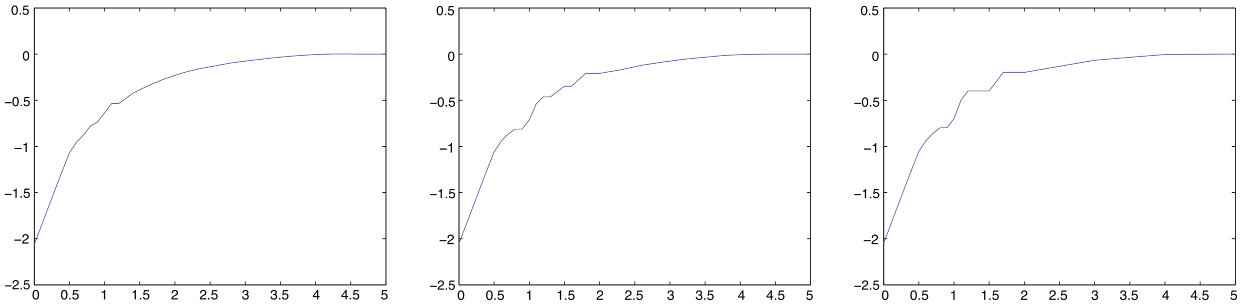
While using either a bound on w or $\epsilon > 0$ can produce similar appearing solutions they can produce quite different w. On one hand one can argue that w is not really of interest. We can just take the computed u, do a high order interpolation, and use this as a good control. On the other hand, w does have an impact on the speed of solution of the numerical problem.

In Figs. 11 and 12 we give the result using the two strategies with full grid refinement. The computed solutions are visually identical to the true solution. However, the EIC control w is quite different. In Fig. 11 we see a lot of bouncing between the control bounds which is to be expected. In Fig. 12 we see chatter around the graph of $\dot{x}_2$. Essentially the software is finding w by going backwards through the trapezoid method from the correct solution for x_2.



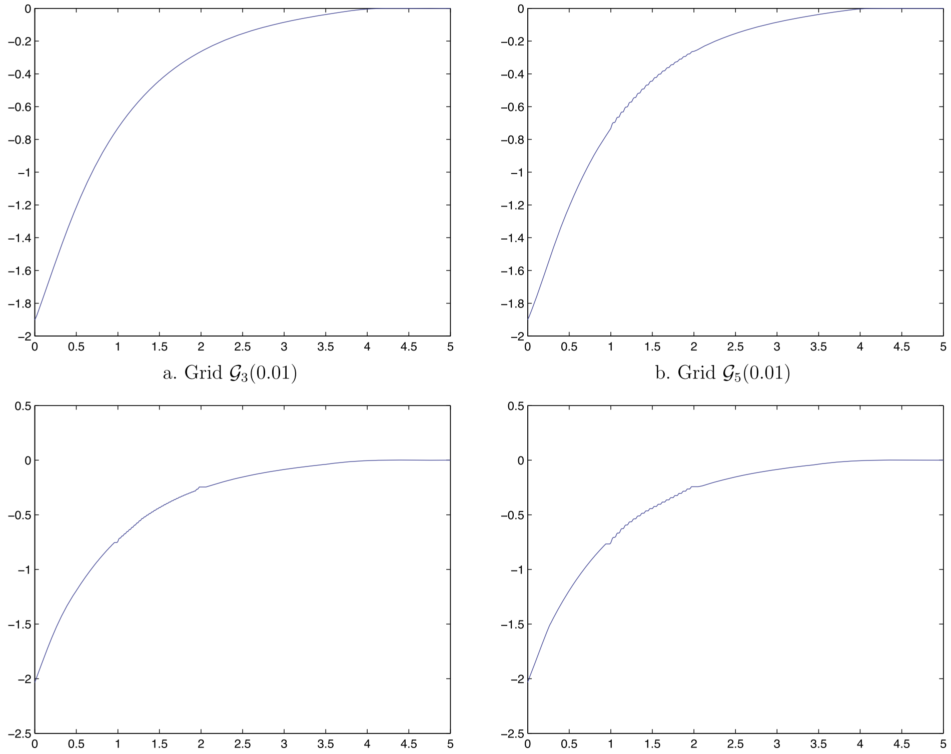
a. Grid $\mathcal{G}_2(0.1)$ b. Grid $\mathcal{G}_3(0.1)$ c. Grid $\mathcal{G}_5(0.1)$

Fig. 8. Computed control u with $\epsilon = 10^{-2}$ and no w bounds for (10) on special grids.



a. Grid $\mathcal{G}_2(0.1)$ b. Grid $\mathcal{G}_3(0.1)$ c. Grid $\mathcal{G}_5(0.1)$

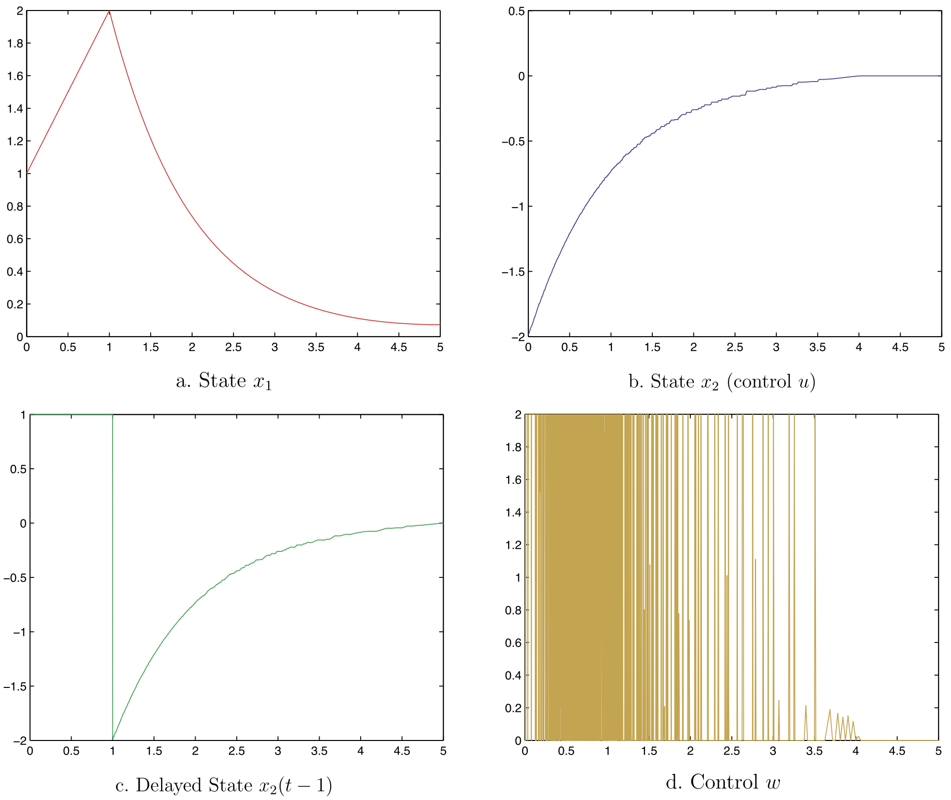
Fig. 9. Computed control u for $\epsilon = 0$ and $0 \le w \le 2$ for (10) on special grids.



a. Grid $\mathcal{G}_3(0.01)$ b. Grid $\mathcal{G}_5(0.01)$

c. Grid $\mathcal{G}_3(0.01)$ d. Grid $\mathcal{G}_5(0.01)$

Fig. 10. Computed control u for (10) with $\epsilon = 10^{-2}$ (top) and w bounded (bottom).



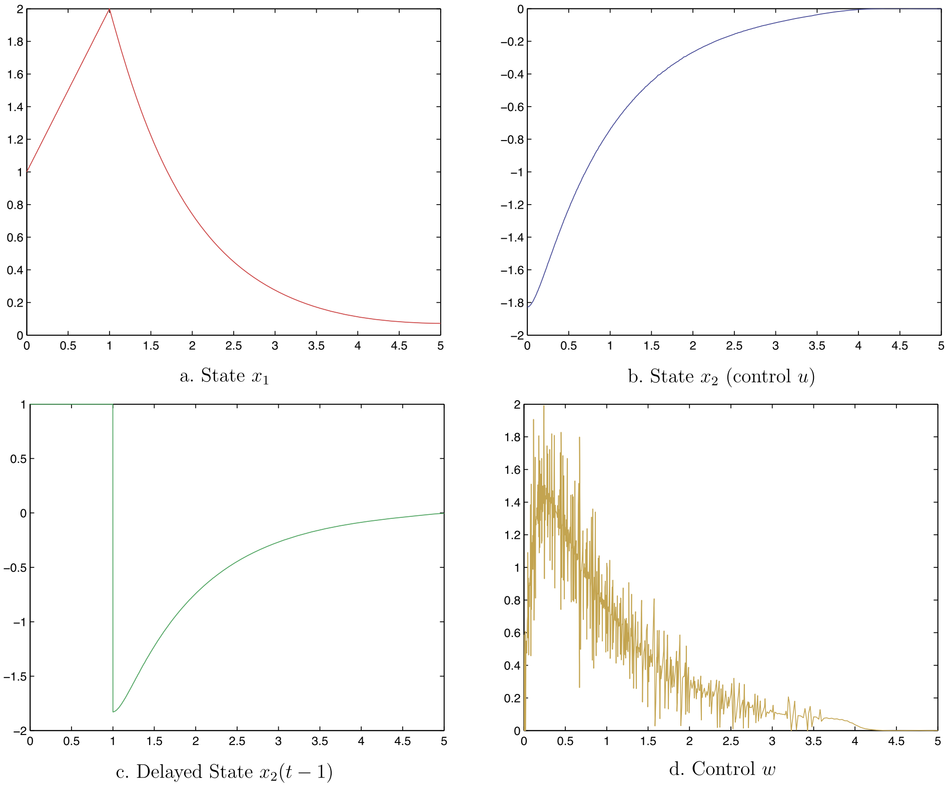
a. State x_1

b. State x_2 (control u)

c. Delayed State $x_2(t-1)$

d. Control w

Fig. 11. Bounded control: $\epsilon = 0$ for (10).



a. State x_1

b. State x_2 (control u)

c. Delayed State $x_2(t-1)$

d. Control w

Fig. 12. $\epsilon = 10^{-2}$ and no bound on control in (10).

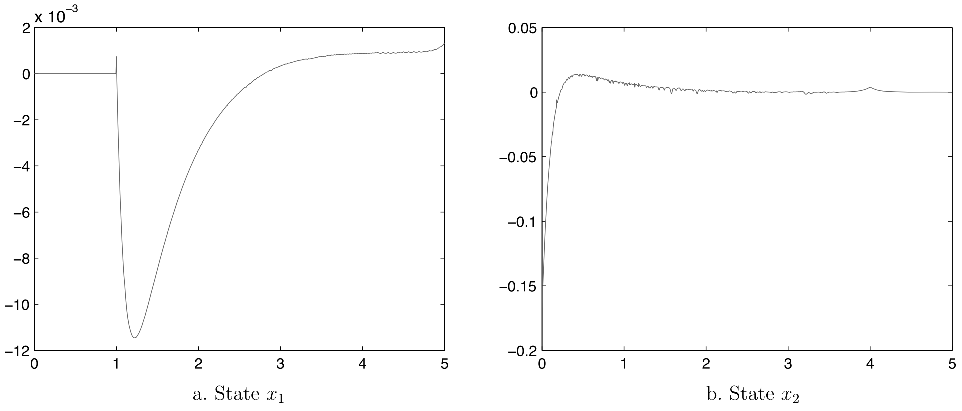


Fig. 13. Error in variable grid solutions with $\epsilon = 10^{-2}$ compared to MOS of original problem.

Visually the graphs of x_1, x_2 (or u) in Fig. 11 are all identical to a method of steps solution of (8) with the graphs appearing to be equal when superimposed. In fact, they differ by $O(10^{-3})$.

The error is generally small except shortly after $t = 1$ as shown in Fig. 13. Note that x_1 error is 10^{-2} on the whole interval. However, the computed control error which is the error in x_2 of the EIC formulation has a very small $O(1)$ error near $t = 1$. We shall comment more on this later.

5.2. EIC on CSTR

We now turn to the more complex CSTR problem. Applying the EIC method to (9) we first compute the bounds for w by determining the slopes of original solution. We estimate the min slope as $m = -5$ and the max slope as $M = 2$ for delayed control, $u_2(t)$. Subsequently, we create two new variables x_4, w where

$$x_4(t) = u_2(t), \tag{11a}$$

$$\dot{x}_4(t) = w(t), \tag{11b}$$

$$-5 \le w \le 2. \tag{11c}$$

Now let $\bar{x}(t) = (x_1,\ x_2,\ x_3)^T$ and $\bar{u} = (u_1,\ x_4)^T$. Then the EIC reformulation for (9) is in problem (12) where the goal is to minimize the cost functional

$$\tilde{F} = \int_0^{0.2} \|\bar{x}(t)\|^2 + 0.01\|\bar{u}(t)\|^2 + \epsilon w^2(t)dt, \tag{12a}$$

subject to

$$\dot{x}_1 = -x_1(t) + R(t, \bar{x}), \tag{12b}$$

$$\dot{x}_2 = -x_2(t) + 0.9x_4(t-s) + 0.1x_4(t), \tag{12c}$$

$$\dot{x}_3 = -2x_3(t) + 0.25R(t, \bar{x}) - 1.05u_1(t)x_3(t-r), \tag{12d}$$

$$\dot{x}_4 = w(t), \tag{12e}$$

$$x_3(t) = -0.02, \qquad -0.015 \le t < 0 \tag{12f}$$

$$x_4(t) = 1, \qquad -0.02 \le t < 0 \tag{12g}$$

$$x(0) = (0.49, -0.0002, -0.02, \rho)^T, \tag{12h}$$

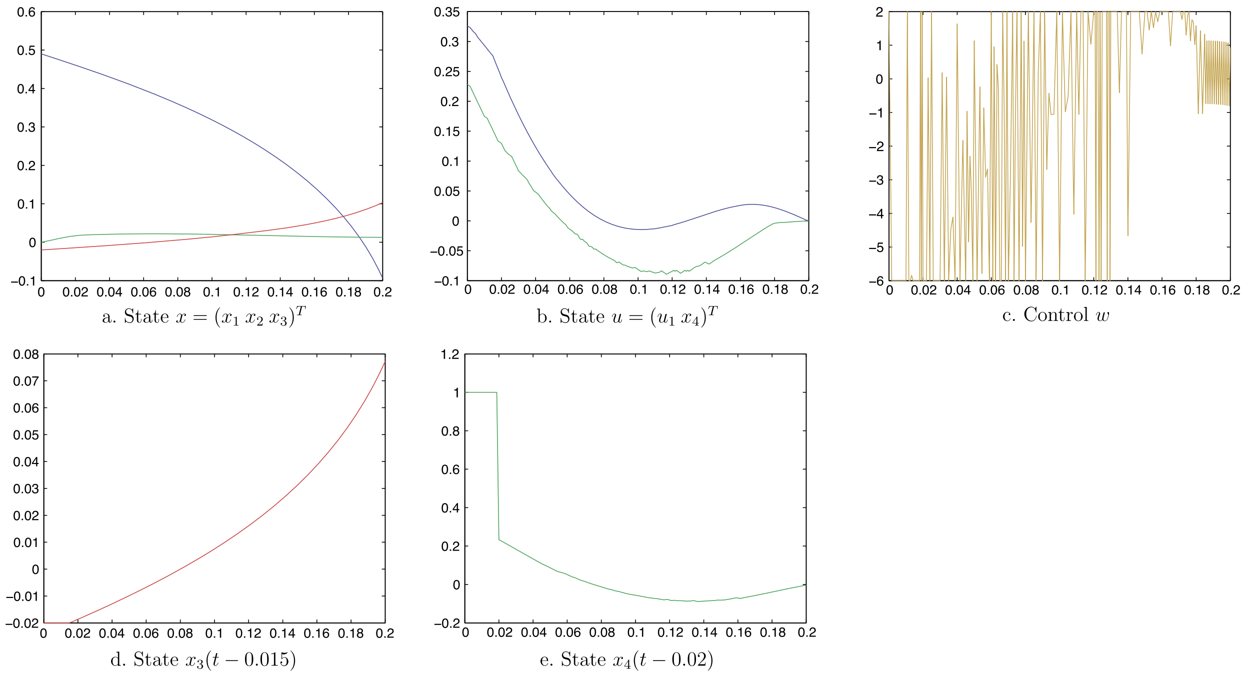
$$0.2 \le \rho \le 0.24, \tag{12i}$$

$$|u_1(t)| \le 500, \tag{12j}$$

$$-5 \le |w(t)| \le 2, \tag{12k}$$

with R given by (9g).

Note two things that are different about reformulation (12) from reformulation (10b). Unlike x_2 in (10b), here x_4 in (12) has a constrained initial condition (12i). We have found in some problems that imposing some constraints on the



a. State $x = (x_1\, x_2\, x_3)^T$ b. State $u = (u_1\, x_4)^T$ c. Control w

d. State $x_3(t - 0.015)$ e. State $x_4(t - 0.02)$

Fig. 14. Bounded control: $\epsilon = 0$ for CSTR problem.

initial value of the control helped in getting feasible results on earlier iterations. This is especially the case if there were jump discontinuities in the exact solution. However, this consideration is still under investigation and has not been carefully investigated. Additionally, x_4 is not bounded. Recall that bounds may not be necessary to achieve good approximations. Solving (12) we get the approximations in Fig. 14 of the original (9).

We also computed the solution of an MOS formulation to obtain a truth model. The graphs of the MOS and EIC solutions were visually identical when plotted together as shown in Fig. 15. We observed errors of 10^{-4} in the states, 10^{-3} in the delayed state and 10^{-4} in the delayed control. We experimented with different weights ϵ for the CSTR problem. With no bounds on w we got the best answer with $\epsilon = 10^{-6}$. Note that the states of the CSTR problem are smaller than the states of the SDC problem. Thus it is reasonable that a smaller value of ϵ is needed to make sure that the ϵ is only regularizing the numerics but also not altering the performance requirements in a major way. In both the SDC and CSTR problem we left the exogenous control initial condition as a free parameter ρ. This was important in that it allowed for a jump discontinuity when required. However, if one took the solution obtained and added an additional initial condition on $u(0)$, then after a little experimentation a slightly more accurate solution was obtained especially on coarser grids.

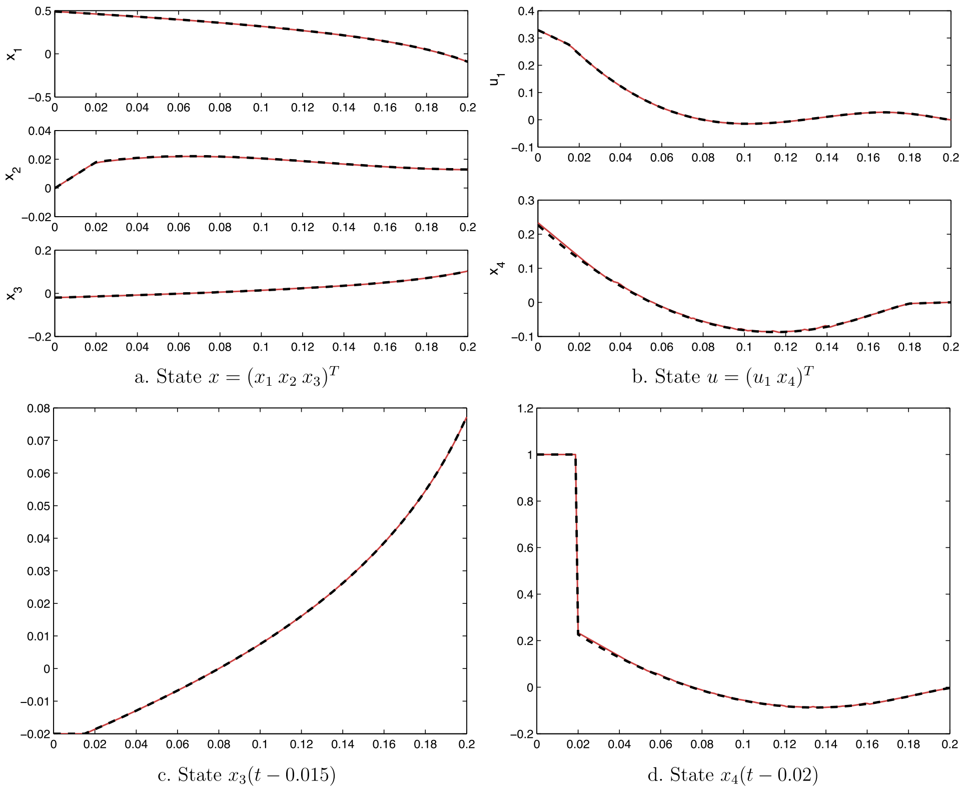
Table 2 gives the computed optimal cost for the CSTR problem using EIC with several values of ϵ. For comparison we give $J^\dagger$ which was computed using a uniform grid of 1000 points. We note that the EIC solved the problem with grids of 150 or 150 points depending on the value of ϵ.

5.3. Exogenous variable w

Some explanation is needed for the noisy variable w. The purpose of adding the exogenous variable w is to change the discretization of the delayed algebraic variable u to a delayed state variable which in turn changes how it is discretized. Looking at the limiting case $\epsilon = 0$ we see that the original problem is solved and the variable w is solved in terms of the solution of the original problem. In general w is a derivative of u. However, this derivative is found by going backwards through the discretization. To illustrate, we take the equation $u' = w$ and discretize it with TR and use a uniform grid of 160 intervals on $[0, 4]$. We use full precision values of $\in(t)$ for $u(t)$. Looking at the left side of Fig. 16 we get the expected $\cos(t)$. However if we look at the right figure we see the error which is around 10^{-4}. Taking a smaller grid makes the error less.

But in actuality using EIC we do not have close to exact values of the discretized optimal control u. We have values that are computed from trying to solve the optimal control NLP and this error is much larger. To illustrate the effect of this we took the full precision values of $\sin(t)$ and perturbed them by random errors of size 3×10^{-4} and used this value for $u(t)$. The result is given in Fig. 17.

Compare Fig. 17 (left) with Fig. 12d, we see a similar noise effect.



a. State $x = (x_1\ x_2\ x_3)^T$

b. State $u = (u_1\ x_4)^T$

c. State $x_3(t - 0.015)$

d. State $x_4(t - 0.02)$

Fig. 15. Solutions compared to MOS of original problem (9).

Table 2
Optimal cost for the CSTR problem on a fine uniform grid and with EIC for three values of epsilon.

$J^\dagger$	$\epsilon = 10^{-2}$	$\epsilon = 10^{-4}$	$\epsilon = 10^{-6}$
2.13496 E−02	2.13481 E−02	2.13450 E−02	2.133339 E−02

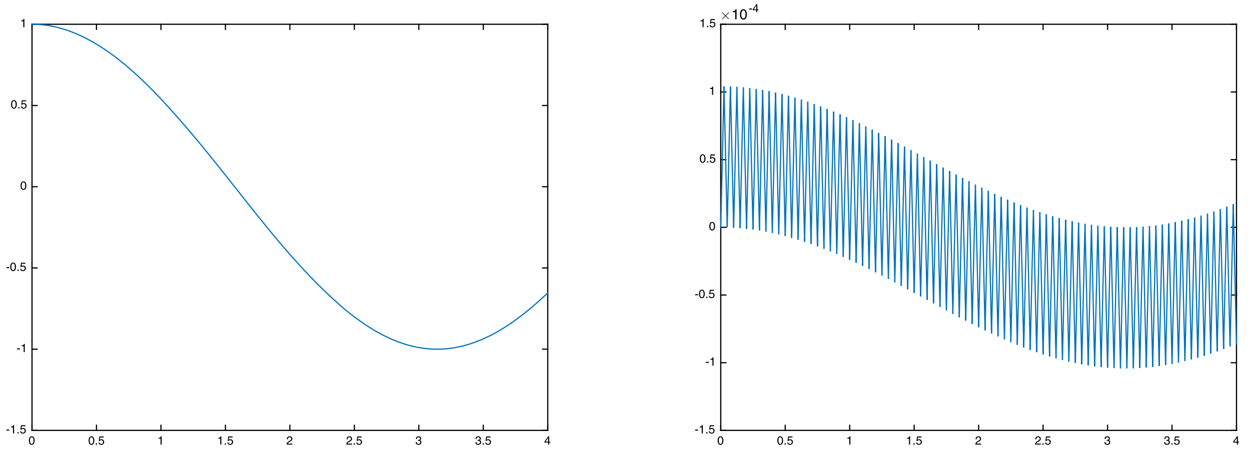


Fig. 16. Solution w (left) and error (right) computed from $u = \sin t$. Note scale on the right is 10^{-4}.

6. Analysis of EIC method

Determining what can be reasonably expected for the EIC method, and what its limitations are, requires some analysis. As is the case for many numerical procedures they often work on more complicated problems than the analysis can be carried out for.

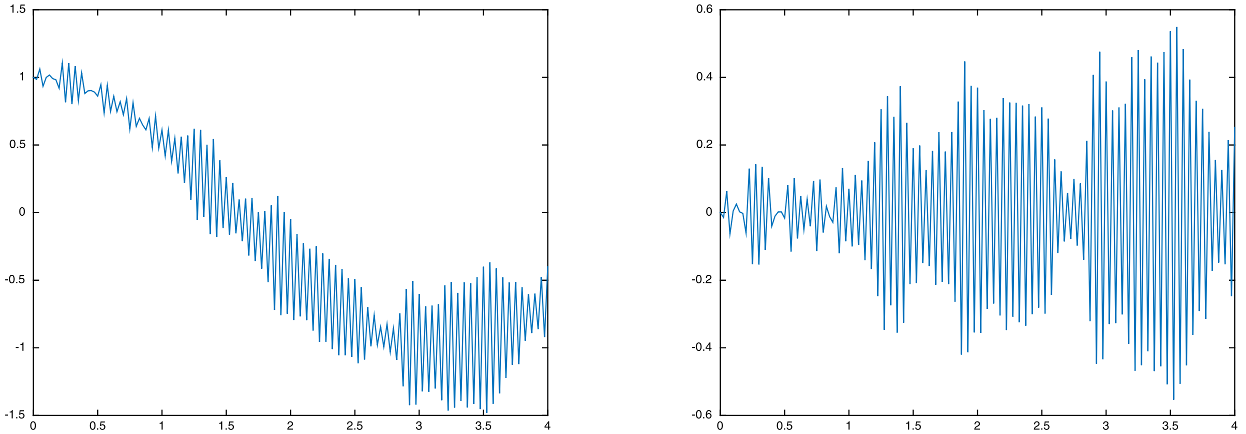


Fig. 17. Solution w (left) and error (right) from truth computed from perturbed values of $u = \sin t$.

We begin with the problem (13) with a control delay r. One could add extra terms to the cost but that would not change anything. Note that Theorem 1 does not require the delay to be constant.

$$\min \ J(u) = \int_0^T L(x(t), u(t), u(t-r), t)\, dt, \tag{13a}$$

$$\dot{x}(t) = f(x(t), u(t), u(t-r), t), \tag{13b}$$

$$x(0) = x_0, \tag{13c}$$

$$u(t) = \phi(t), \quad -r \le t < 0. \tag{13d}$$

The EIC regularization of (13) is (14)

$$\min \ J_\epsilon(u) = \int_0^T L(x(t), u(t), u(t-r), t) + \epsilon \|w(t)\|^2\, dt, \tag{14a}$$

$$\dot{x}(t) = f(x(t), u(t), u(t-r), t), \tag{14b}$$

$$\dot{u}(t) = w(t), \tag{14c}$$

$$x(0) = x_0, \tag{14d}$$

$$u(t) = \phi(t), \quad -r \le t < 0. \tag{14e}$$

Note that because the delay equation is part of an optimization we do not need to specify $u(0)$ in either (13) or (14). This is important later.

Theorem 1. *Suppose that (13) has an optimal solution x^*, u^* and that u^* is piecewise smooth and $\dot{u}^*$ is in L^2. Let $\|\cdot\|_2$ be the L^2 norm. Suppose that (14) has an optimal solution $x^*_\epsilon, u^*_\epsilon, w^*_\epsilon$. Then $\|\dot{u}^*\|^2 \ge \|\dot{u}^*_\epsilon\|^2$ for all $\epsilon > 0$ and $\lim_{\epsilon \to 0^+} J_\epsilon(u^*_\epsilon) = \lim_{\epsilon \to 0^+} J(u^*_\epsilon) = J(u^*)$.*

Proof. Note that letting $w = \dot{u}^*$ gives u^*, x^*, w is a feasible solution of (14). Thus

$$J_\epsilon(u^*) \ge J_\epsilon(u^*_\epsilon). \tag{15}$$

Conversely, we have that $u^*_\epsilon, x^*_\epsilon$ is a feasible solution of (13). Thus

$$J(u^*_\epsilon) \ge J(u^*). \tag{16}$$

But for any piecewise differentiable $u \in L^2$ we have

$$J_\epsilon(u) = J(u) + \epsilon \|\dot{u}\|_2^2. \tag{17}$$

Using (17) in (15) and then using (16) gives

$$J(u^*) + \epsilon \|\dot{u}^*\|_2^2 \ge J(u^*_\epsilon) + \epsilon \|\dot{u}^*_\epsilon\|_2^2 \tag{18}$$

$$\ge J(u^*) + \epsilon \|\dot{u}^*_\epsilon\|_2^2 \tag{19}$$

and $\|\dot{u}^*\|_2^2 \ge \|\dot{u}^*_\epsilon\|_2^2$ as desired. The limit result now follows. □

Note that in (14) the control variable is now w. This problem is what is known as a cheap control problem and there are a number of results on convergence and the existence of asymptotic expansions [31,32]. However, it is not quite the standard cheap control problem in two respects. For one, there are the delays. But secondly there is only an initial value on x and not on u and in (14) the state is made up of both x and u. This will also be important later.

There are two separate issues. One has to do with the analytic question of how the solutions act as functions of ϵ. The second question is how the numerics impact on direct transcription. We shall see that these two considerations are related but not equivalent. We focus first on the analytic question.

There has been some prior work on cheap control of delayed systems [19–21]. However, those formulations are difficult to use for the insight we wish to gain as they do not make it obvious how to exploit the special structure to our problem. Note that in our problem, the $\epsilon = 0$ case is a well defined optimal control problem in its own right. This is important.

Theorem 1 is a general statement that applies to nonlinear problems. But it does not address the question of in what sense x_ϵ^* and u_ϵ^* converge as $\epsilon \to 0^+$. To see what is the best that can be expected we look at the special case of (13) where it is an LQR (Linear Quadratic Regulator) problem.

6.1. Analytically equivalent problem

Suppose that the original control delay problem is a control delayed LQR problem

$$\min \int_0^T x(t)^T Q_1 x(t) + u^T(t) Q_2 u(t)\, dt, \tag{20a}$$

$$\dot{x}(t) = Ax(t) + Bu(t-r), \tag{20b}$$

$$u(t) = \phi(t),\ t \in [-r,\ 0], \tag{20c}$$

$$x(0) = x_0, \tag{20d}$$

with $Q_2 > 0$ and $Q_1 \geq 0$. We take A, B constant but that is not important. The EIC formulation of (20) is

$$\min \int_0^T x_1^T(t) Q_1 x_1(t) + x_2^T(t) Q_2 x_2(t) + \epsilon^2 w(t)^T w(t)\, dt, \tag{21a}$$

$$\dot{x}_1(t) = Ax_1(t) + Bx_2(t-r), \tag{21b}$$

$$\dot{x}_2(t) = w(t), \tag{21c}$$

$$x_2(t) = \phi(t),\ t \in [-r, 0], \tag{21d}$$

$$x_1(0) = x_0. \tag{21e}$$

Now x_1 is determined on $[0, r]$ by (21b), (21d), and (21e). Thus x_1 on $[0, r]$ makes a fixed contribution to the cost. But $x_2 = 0$ on $[T - r, T]$, $w = 0$ on $[T - r, T]$ since these terms appear in the cost (21a) but are freely variable. Thus solving (21) is mathematically equivalent to solving

$$\min \int_r^T x_1(t)^T Q_1 x_1(t) + \hat{x}_2(t)^T Q_2 \hat{x}_2(t) + \epsilon^2 \hat{w}^T(t) \hat{w}(t)\, dt, \tag{22a}$$

$$\dot{x}_1(t) = Ax_1(t) + B\hat{x}_2(t), \tag{22b}$$

$$\dot{\hat{x}}_2(t) = \hat{w}(t), \tag{22c}$$

$$x_1(r) = \hat{x}_0, \tag{22d}$$

where we have dropped x_1 on $[0, r]$, restricted x_1 to $[r, T]$, and replaced w, x_2 by $\hat{w}(t) = w(t - r)$, $\hat{x}_2(t) = x_2(t - r)$. But this reformulated problem (22) is what is called a classical cheap control problem in the singular perturbations literature. From [31] we have

Theorem 2. *Suppose that* $\widetilde{B}^T \widetilde{Q} \widetilde{B} > 0$, $\dot{x}(t) = \widetilde{A}x(t) + \widetilde{B}u(t)$, $x(0) = x_0$, *and*

$$J(\epsilon) = \frac{1}{2} \int_0^T x^T(t) \widetilde{Q} x(t) + \epsilon^2 u^T(t) \widetilde{R} u(t)\, dt. \tag{23}$$

Then for each sufficiently small $\epsilon > 0$ *and each integer* $M \geq 1$, *the optimal control, corresponding trajectory, and optimal cost are uniquely determined and satisfy the asymptotic expansion;*

$$u(t,\epsilon)=\frac{v_0(\tau)}{\epsilon}+\sum_{j=0}^{M}\big(U_j(t)+v_{j+1}(\tau)+w_j(\sigma)\big)\epsilon^j+O(\epsilon^{M+l}), \tag{24a}$$

$$x(t,\epsilon)=X_0(t)+m_0(\tau)+\sum_{j=1}^{M}\big(X_j(t)+m_j(\tau)+n_{j-1}(\sigma)\big)\epsilon^j+O(\epsilon^{M+1}) \tag{24b}$$

$$J(\epsilon)=\frac{1}{2}\sum_{j=0}^{M}\Big(x^T(0)K_j(0)x(0)\Big)\epsilon^j+O(\epsilon^{M+1}) \tag{24c}$$

uniformly on $0\le t\le T$. Terms which are functions of $\tau=t/\epsilon$ (or $\sigma=(T-t)/\epsilon$) decay to zero as $\tau\to\infty$ (or $\sigma\to\infty$).

Here $U_j, v_j, w_j, X_j, K_j(0)$ are the coefficients of the expansion. If (24a), (24b), (24c) are substituted into (21), (23), then equality holds uniformly up to $O(\epsilon^{M+1})$.

From [32] we have that

$$v(t/\epsilon)=\frac{1}{\epsilon}Ce^{-Ct/\epsilon}\alpha, \tag{25}$$

where $C=(\widetilde{B}^T\widetilde{Q}\widetilde{B})^{1/2}>0$ and α is a constant vector. Thus v_0 is an approximation of a distribution since

$$\int_0^T\frac{1}{\epsilon}Ce^{-Ct/\epsilon}\alpha\,dt=\int_0^{T/\epsilon}Ce^{-\tau}d\tau\,\alpha,$$

which as $\epsilon\to 0^+$ has the finite limit

$$\int_0^\infty Ce^{-\tau}d\tau\,\alpha.$$

But (25) is not bounded in L^2 if $\alpha\neq 0$ since

$$\int_0^T\frac{1}{\epsilon^2}\alpha^TC^2e^{-2Ct/\epsilon}\alpha\,dt$$

goes to ∞ as $\epsilon\to 0^+$. But Theorem 1 tells us that in the application of EIC that if the optimal control u is piecewise smooth, then the EIC control is L^2 bounded.

The only terms in the expansion of the control that might not go to zero uniformly on $[0,T]$ as ϵ goes to zero are v_1, w_0. We shall illustrate what happens with v_1. Let $\delta_1>0$ and $\delta_2>0$ be two small numbers. Since the limit at infinity of v_1 is zero, there is an $N>0$ such that $\|v_1(\tau)\|\le\delta_1$ for $\tau\ge N$. Let $\epsilon\le\delta_2/N$. Then for $t\in[\delta_2,T]$ we have $\|v_1(t)\|\le\delta_1$ for $\epsilon\le\delta_2$. That is, as ϵ goes to zero we have v_1 converges to zero uniformly on compact subsets of $(0,T]$. We will see that this behavior is reflected in the numerical solution.

Theorem 3. *Under the assumptions of Theorem 2, we have that $v_0(t)=0$ and the distributional part of the expansion is missing. Thus we get that the EIC control solution converges to the $\epsilon=0$ case uniformly on compact subsets of $(0,T)$, pointwise on $[0,T]$, and in L^1.*

The question is then why does the O'Malley result fail to produce a distributional solution at the left end with EIC? The answer is that O'Malley assumes that the full initial state is prescribed. In EIC only part of the initial state is prescribed and the new state that was a control has a free initial value. Thus the solution of the EIC problem is a solution of the O'Malley problem but for a very special initial state value which makes $\alpha=0$. Fortunately, the user does not need to know this special initial state value.

One might assume that this could be exploited if one could estimate what that initial state is. However, what happens is that there is always some error and numerically you get a small number times a distribution. Thus if you have a good estimate for $u(0)$ at first you do better but if the grid gets fine enough the error will still go unbounded at the left. On the other hand, if you leave $u(0)$ free, the optimizer produces the desired solution.

If we look at Fig. 13b and the $x_2(t)=u(t)$ error, we see what the error is typically like with EIC. It is generally small with a small error on the left that looks like $e^{-t/\epsilon}$. Since this is small in L^1 we can easily get a piecewise smooth control that is almost optimal.

7. Conclusion

DT is a popular method frequently used to solve complex problems in industry. Previously it has been shown that while a straightforward application to state delay control problems using a formulation like (3) was successful, that there were intrinsic difficulties with using this approach on control delay problems with nonuniform grids. In this paper we have shown that the intrinsic problem on nonuniform grids was due to free (or lightly weighted) control variables. A reformulation called Exogenous Input Control (EIC) parameterization was introduced and shown to take care of this intrinsic problem in an efficient manner. While the usefulness of EIC is now established some technical points remain. In particular there are options in how to implement the interpolation constraints used for the exogenous control w. The choice does not impact TR or HS as integrators of delayed problems but having theory based guidance for users on how to set up the EIC formulation would be desirable. This will be examined.

Acknowledgements

Research supported in part by NSF Grant DMS-1209251. The authors thank the reviewers for a number of helpful suggestions.

References

[1] V.M. Becerra, $\mathcal{PSOPT}$ Optimal Control Solver User Manual Release 2, University of Reading School of Systems Engineering, Reading, United Kingdom, 2010.

[2] J.T. Betts, Issues in the direct transcription of optimal control problems to sparse nonlinear programs, in: R. Bulirsch, D. Kraft (Eds.), Computational Optimal Control, in: Int. Ser. Numer. Math., vol. 115, Birkhäuser Verlag, Basel, 1994, pp. 3–18.

[3] J.T. Betts, Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, 2nd ed., SIAM, Philadelphia, 2010.

[4] J.T. Betts, N. Biehn, S.L. Campbell, W. Huffman, Compensating for order variation in mesh refinement for direct transcription methods II: computational experience, J. Comput. Appl. Math. 143 (2002) 237–261.

[5] J.T. Betts, S.L. Campbell, A. Engelsone, Direct transcription solution of inequality constrained optimal control problems, in: Proc. 2004 American Control Conference, Boston, MA, 2004, pp. 1622–1626.

[6] J.T. Betts, S.L. Campbell, K.C. Thompson, Optimal control software for constrained nonlinear systems with delays, in: Proc. IEEE Multi-Conference on Systems and Control, MSC 2011, Denver, 2011, pp. 1373–1378.

[7] J.T. Betts, S.L. Campbell, K.C. Thompson, Direct transcription solution of optimal control problems with control delays, in: AIP Conf. Proc., vol. 1389, 2011, pp. 38–41.

[8] J.T. Betts, S.L. Campbell, K.C. Thompson, Optimal control of a delay partial differential equation, in: L. Biegler, S. Campbell, V. Mehrmann (Eds.), Control and Optimization with Differential-Algebraic Constraints, SIAM, Philadelphia, 2012, pp. 213–231.

[9] J.T. Betts, S.L. Campbell, K.C. Thompson, Direct transcription solution of optimal control problems with differential algebraic equations with delays, in: Proc. 4th IASTED International Symposium on Intelligent Systems and Control, ISC 2013, Marina del Rey, California, 2013, pp. 166–173.

[10] J.T. Betts, S.L. Campbell, K.C. Thompson, Simulation and optimization of systems with delays, in: Society for Modeling and Simulation Series 2013 Proc., San Diego, 2013, pp. 1084–1085.

[11] H. Bevrani, Robust Power System Frequency Control, Power Electronics and Power Systems, Springer, New York, USA, 2009.

[12] N. Biehn, J.T. Betts, S.L. Campbell, W. Huffman, Compensating for order variation in mesh refinement for direct transcription methods, J. Comput. Appl. Math. 125 (2000) 147–158.

[13] K.E. Brenan, S.L. Campbell, L.R. Petzold, Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations, SIAM, Philadelphia, 1996.

[14] R. Bulirsch, F. Montrone, H.J. Pesch, Abort landing in the presence of a windshear as a minimax optimal control problem, part 1: necessary conditions, preprint, 1991.

[15] M. Cloosterman, N. van de Wouw, W. Heemels, H. Nijmeijer, Stability of networked control systems with uncertain time-varying delays, IEEE Trans. Autom. Control 54 (2009) 1575–1580.

[16] C.L. Darby, W.W. Hager, A.V. Rao, An hp-adaptive pseudospectral method for solving optimal control problems, Optim. Control Appl. Methods 32 (2011) 476–502.

[17] A. Engelsone, S.L. Campbell, J.T. Betts, Direct transcription solution of higher-index optimal control problems and the virtual index, Appl. Numer. Math. 57 (2007) 281–296.

[18] P.E. Gill, W. Murray, M.A. Saunders, User's Guide for SNOPT 7: A Fortran Package for Large-Scale Nonlinear Programming, Systems Optimization Laboratory, Stanford University, California, 2007.

[19] V.Y. Glizer, Asymptotic solution of a cheap control problem with state delay, Dyn. Control 9 (1999) 339–357.

[20] V.Y. Glizer, Blockwise estimate of the fundamental matrix of linear singularly perturbed differential systems with small delay and its application to uniform asymptotic solution, J. Math. Anal. Appl. 278 (2003) 409–433.

[21] V.Y. Glizer, Stochastic singular optimal control problem with state delays, regularization, singular-perturbation, and minimizing sequence, SIAM J. Control Optim. 50 (2012) 2862–2888.

[22] L. Göllmann, D. Kern, H. Maurer, Optimal control problems with delays in state and control variables subject to mixed control-state constraints, Optim. Control Appl. Methods 30 (2009) 341–365.

[23] L. Göllmann, D. Kern, H. Maurer, Theory and applications of optimal control problems with multiple time delays, J. Ind. Manag. Optim. 10 (2014) 413–441.

[24] E. Hairer, S.P. Norsett, G. Wanner, Solving Ordinary Differential Equations I, Nonstiff Problems, second revised edition, Springer, 2000.

[25] Y. He, G. Liu, D. Rees, M. Wu, Stability analysis for neural networks with time-varying interval delay, IEEE Trans. Neural Netw. 18 (2007) 1850–1854.

[26] G.T. Huntington, Advancement and analysis of a Gauss pseudospectral transcription for optimal control problems, Ph.D. dissertation, Massachusetts Institute of Technology, Dept. of Aeronautics and Astronautics, 2007.

[27] C. Ionescu, R. Hodrea, R. De Keyser, Variable time-delay estimation for anesthesia control during intensive care, IEEE Trans. Biomed. Eng. 58 (2011) 363–369.

[28] C.-Y. Kao, B. Lincoln, Simple stability criteria for systems with time-varying delays, Automatica 40 (2004) 1429–1434.

[29] P. Kunkel, V. Mehrmann, Differential-Algebraic Equations: Analysis and Numerical Solution, European Mathematical Society, Zürich, 2006.

[30] Y. Li, L. Baldacchino, Implementation of some higher-order convection schemes on non-uniform grids, Int. J. Numer. Methods Fluids 21 (1995) 1201–1220.

[31] R.E. O'Malley Jr., A more direct solution of the nearly singular linear regulator problem, SIAM J. Control Optim. 14 (1976) 1063–1077.
[32] R.E. O'Malley Jr., A. Jameson, Singular perturbations and singular arcs—part I, IEEE Trans. Autom. Control AC-20 (1975) 218–226.
[33] A.V. Rao, D.A. Benson, C. Darby, M.A. Patterson, C. Francolin, I. Sanders, G.T. Huntington, Algorithm 902: GPOPS, a MATLAB software for solving multiple-phase optimal control problems using the gauss pseudospectral method, ACM Trans. Math. Softw. 37 (2010) 22:1–22:39.
[34] O. Santos, L. Rodríguez-Guerro, O. López, Experimental results of a control time delay system using optimal control, Optim. Control Appl. Methods 33 (2011) 100–113.
[35] K.C. Thompson, Solving nonlinear constrained optimization time delay systems with a direct transcription approach, Ph.D. thesis, North Carolina State University, Department of Mathematics, Raleigh, NC, 2014.
[36] R. Vinter, Optimal Control, Modern Birkhäuser Classics, 10th edition, Springer Verlag, New York, 2010.
[37] A. Wächter, L.T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Math. Program. 106 (2006) 25–57.
[38] F. Wu, K.M. Grigoriadis, LPV systems with parameter-varying time delays: analysis and control, Automatica 37 (2001) 221–229.