SOLVING HIGHER INDEX DAE
OPTIMAL CONTROL PROBLEMS

STEPHEN CAMPBELL
Department of Mathematics
North Carolina State University
Raleigh, NC, 27695-8205 USA

PETER KUNKEL
Mathematisches Institut
Universität Leipzig
Augustusplatz 10, D-04109 Leipzig, Germany

(Communicated by Kok Lay Teo)

ABSTRACT. A number of methods have been proposed for solving optimal control problems where the process being optimized is described by a differential algebraic equation (DAE). However, many of these methods require special circumstances to hold or the user to have special software. In this paper we go over many of these options and discuss what is usually necessary for them to be successful. We use a nonlinear index three control problem to illustrate many of our observations.

1. Introduction. Differential algebraic equations, or DAEs, are being ever more widely used in science and industry to model a number of different kinds of processes. Naturally the optimal control of DAE modeled processes

\[
J(x, u) = M(x(T)) + \int_0^T K(x(t), u(t)) \, dt, \quad (1a)
\]
\[
F(t, x, \dot{x}, u) = 0, \quad x(0) = x_0, \quad (1b)
\]

where \(x\) and \(u\) represent the state and the control of the process given by \(F\) with initial state \(x_0\) and cost functional \(J\) which is to be minimized, has been considered by a number of authors. We mention only [3, 5, 6, 8, 10, 13, 15, 17, 19, 29]. However, the vast majority of this work has focused on index one problems or linear higher index problems. In the context of specific applications there has been work on dealing with higher index models especially those from constrained mechanics. In these cases a specific approach is often advocated. Most of this prior work presents one approach or algorithm.

Our intention in this paper is quite different. We intend to discuss the different issues involved and the options available to a person wanting to solve optimal control
problems with higher index DAE processes. We also want to discuss when some of these options are not available and why.

To illustrate a number of points we will use a specific nonlinear problem. In some cases a particular approach can work but require greater care. We shall comment on that also.

In general, there are several approaches to numerically solving optimal control problems. One is to solve the necessary conditions assuming they can be formulated. Control parameterization parameterizes the controls and then uses simulation to evaluate the cost functional inside an optimizer. Direct transcription (DT) [3] fully discretizes the problem, iteratively solves the resulting nonlinear programming problem and then possibly refines the discretization. We shall discuss all of these in the context of our model problems.

For direct transcription we use here either SOCX [3] or GPOPS II [30]. Some of the boundary value problems (BVP) that arise in a solution of the necessary conditions are done by GPOPS II. Others require a DAE BVP solver as described in [26, 27]. In all of the cases and tests done as part of this paper either the numerical solver succeeded to high accuracy or failed to produce a useful answer in reasonable time.

We assume that the reader is somewhat familiar with DAEs [7, 22] and with the introductory theory of calculus of variations and optimal control. There have been other studies on the advantages of using optimal controlled DAEs in the context of high index prescribed trajectory problems [1]. Here we assume that the decision to solve an optimal control problem has already been made.

The paper is organized as follows. Section 2 presents possible reformulations of the constraining DAE which may be necessary for the various approaches discussed later. It also introduces the accompanying test example with all the needed reformulations. The reformulations mainly consist of types of index reduction depending on the structure of the reduced DAE that has to be achieved. Section 3 gives a list of possible numerical approaches and their requirements. This includes the formulation of necessary conditions as well as direct transcription and control parameterization. All presented approaches are applied to the given test problem in Section 4. In particular, we discuss the needs of the used implementations and their performance. The paper ends with some remarks on more challenging versions of the accompanying example in Section 5 and conclusions in Section 6.

2. Problem reformulations. The original formulation (1) may not have the form required for a specific approach. In order to discuss the various possibilities we use the following illustrative example which is a variation of the classical two dimensional pendulum problem.

In particular, we consider the optimal control problem consisting of the DAE initial value problem,

\[
\begin{align*}
\dot{x} &= -\lambda x - ax + uy, \\
\dot{y} &= -\lambda y - ay - g - ux, \\
0 &= x^2 + y^2 - L^2, \\
x(0) &= 0, \\
y(0) &= L, \\
\dot{x}(0) &= 0,
\end{align*}
\]
Higher Index Optimal Control Problems

\[ \dot{y}(0) = 0, \]  
\[ J(x, u) = \int_0^T cu^2 + d(x - L \sin(t + \alpha))^2 + d(y - L \cos(t + \alpha))^2 dt. \]

The equations (2a)–(2c) describe a pendulum of length \( L \) with friction acting tangential to the circle of motion and with a control \( u \) applied tangential to the motion. The friction terms have \( a \) in them. Note \( a > 0 \). The quantity \( \lambda \) is the force generated by the path constraint (2c). The overall problem (2) is a tracking problem. The goal is to track a target moving around the circle of radius \( L \) as well as possible with some cost to the control \( u \). Having \( \alpha > 0 \) gives the target a head start. Our initial velocity is zero so some control effort is required. In general, we will take the terminal time \( T \) so that the optimal state solution must go more than one quarter of the way around the circle of radius \( L \).

We rewrite (2a)–(2g) as a first order DAE,

\[ \dot{x}_1 = x_2, \]  
\[ \dot{x}_2 = -x_5 x_1 - ax_2 + u x_3, \]  
\[ \dot{x}_3 = x_4, \]  
\[ \dot{x}_4 = -x_5 x_3 - ax_4 - g - u x_1, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ x_1(0) = 0, \]  
\[ x_3(0) = 0, \]  
\[ x_2(0) = x_3(0) = x_4(0) = 0, \]

where \( \lambda \) is now denoted by \( x_5 \). Note that the algebraic state \( x_5 \) and the control \( u \) do not appear in the constraint (3c) so the DAE (3) is at least index 2. In fact it is index 3. The corresponding hidden constraints can easily be obtained as follows.

Differentiating the constraint (3e) gives

\[ 2x_1 x_2 + 2x_3 x_4 = 0. \]

We drop the factor 2 to simplify the formulas. Differentiating (4a) and using (3b) and (3d) gives

\[ 0 = \dot{x}_1 x_2 + x_1 \dot{x}_2 + \dot{x}_3 x_4 + x_3 \dot{x}_4 \]
\[ = x_2^2 + x_4^2 + x_1 (-x_5 x_1 - a x_2 + u x_3) + x_3 (-x_5 x_3 - a x_4 - g - u x_1) \]
\[ = x_2^2 + x_4^2 - x_5 (x_1^2 + x_3^2) - a (x_1 x_2 + x_3 x_4) - x_3 g \]
\[ = x_2^2 + x_4^2 - x_5 L^2 - x_3 g. \]

Thus the information we have available is

\[ \dot{x}_1 = x_2, \]  
\[ \dot{x}_2 = -x_5 x_1 - ax_2 + u x_3, \]  
\[ \dot{x}_3 = x_4, \]  
\[ \dot{x}_4 = -x_5 x_3 - ax_4 - g - u x_1, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1 x_2 + x_3 x_4. \]
\[
0 = x_1^2 + x_4^2 - x_3L^2 - x_3g, \\
x_1(0) = 0, \\
x_3(0) = L, \\
x_2(0) = 0, \\
x_4(0) = 0.
\]

But (5a)–(5g) is an overdetermined DAE. There are seven equations in five state and one control variables. We need a control variable to optimize over so we need to get a system in five equations which are hopefully index one. There are a number of ways, explicitly or implicitly, to deal with this over determination. Several are discussed in the remainder of this paper.

2.1. **Index reduction preserving the solution set.** The general idea of index reduction preserving the solution set is to replace the DAE \( F(t, x, \dot{x}, u) = 0 \) by a DAE \( \tilde{F}(t, x, \dot{x}, u) = 0 \) which possesses the same solution set as the original DAE but is index 1 and then to consider the equivalent problem of minimizing

\[
J(x, u) = M(x(T)) + \int_0^T K(x(t), u(t)) \, dt, \\
\tilde{F}(t, x, \dot{x}, u) = 0, \ x(0) = x_0.
\]

See [22] for the theoretical basis of such reduced DAEs.

Having in our example all constraints of the DAE at hand, this kind of index reduction can be interpreted as choosing a combination of the differential equations (5a)–(5d) to complete the constraints to an index 1 DAE. Following [22, 25, 26] this can be achieved by multiplying (5a)–(5d) by a \( 2 \times 4 \) matrix \( Z_1^T \). In order to obtain a DAE which is index one on the entire interval of interest, in general this matrix will have to be either time or state dependent. What we need is that \( Z_1^T F_\dot{x} T_2 \) is nonsingular where \( T_2 \) has columns spanning the null space of the Jacobian of the constraints (5e)–(5g), and \( F_\dot{x} \) is the Jacobian of the DAE (5a)–(5e) with respect to \( \dot{x} \) with invertibility holding at least along the optimal solution.

Here the Jacobian of the algebraic constraints is

\[
A_2 = \begin{bmatrix}
2x_1 & 0 & 2x_3 & 0 & 0 \\
x_2 & x_1 & x_4 & x_3 & 0 \\
0 & 2x_2 & -g & 2x_4 & -L^2
\end{bmatrix}.
\]

Thus, one choice for \( T_2 \) is

\[
T_2 = \begin{bmatrix}
x_3 & 0 \\
x_4 & x_2 \\
-x_1 & 0 \\
x_2 & x_1 \\
\frac{x_1g}{L^2} & \frac{2}{L^2}(x_2x_3 - x_4x_1)
\end{bmatrix}
\]

yielding

\[
F_\dot{x} T_2 = \begin{bmatrix}
x_3 & 0 \\
x_4 & x_3 \\
-x_1 & 0 \\
-x_2 & x_1 \\
0 & 0
\end{bmatrix}.
\]
Hence, we can take
\[ Z_1^T = \begin{bmatrix} x_3 & 0 & -x_1 & 0 & 0 \\ 0 & x_3 & 0 & -x_1 & 0 \end{bmatrix}, \]
since then along all the solutions of the DAE we have
\[ Z_1^T F \dot{x} T_2 = \begin{bmatrix} L^2 & 0 \\ 0 & L^2 \end{bmatrix} \]
which is invertible.

The resulting DAE
\begin{align*}
-x_3 (\dot{x}_1 - x_2) - x_1 (\dot{x}_3 - x_4) &= 0, \quad (7a) \\
x_3 (\dot{x}_2 + ax_2 - ux_3 + x_1 x_5) - x_1 (\dot{x}_4 + ax_4 + ux_1 + x_3 x_5 + g) &= 0, \quad (7b) \\
x_1^2 + x_3^2 - L^2 &= 0, \quad (7c) \\
x_1 x_2 + x_3 x_4 &= 0, \quad (7d) \\
x_2^2 + x_4^2 - x_5 L^2 - gx_3 &= 0, \quad (7e)
\end{align*}
is index 1 by construction but it has state dependent coefficients of the derivatives.

For the particular problem considered here, we may choose a time varying \( Z_1^T \) that will keep \( Z_1^T F \dot{x} T_2 \) invertible along the optimal solution. We shall use
\[ Z_1^T = \begin{bmatrix} \cos t & 0 & -\sin t & 0 & 0 \\ 0 & \cos t & 0 & -\sin t & 0 \end{bmatrix} \]
which also produces a nonsingular \( Z_1^T F \dot{x} T_2 \) for states and controls which are close to the optimal solutions. Hence, a possible index-reduced formulation is given by
\begin{align*}
\cos t (\dot{x}_1 - x_2) - \sin t (\dot{x}_3 - x_4) &= 0, \quad (8a) \\
\cos t (\dot{x}_2 + ax_2 - ux_3 + x_1 x_5) - \sin t (\dot{x}_4 + ax_4 + ux_1 + x_3 x_5 + g) &= 0, \quad (8b) \\
x_1^2 + x_3^2 - L^2 &= 0, \quad (8c) \\
x_1 x_2 + x_3 x_4 &= 0, \quad (8d) \\
x_2^2 + x_4^2 - x_5 L^2 - gx_3 &= 0, \quad (8e)
\end{align*}
which is now linear in the derivatives.

2.2. **Index reduction with extra dynamics.** The DAE (8) is a nonlinear DAE that is of a form that quite a number of approaches cannot deal with. Not only is it not semi-explicit but four differentiated variables appear in the two dynamics equations. Thus it does not fit the usual separation of algebraic variables and differential variables used for example in DT codes.

It has been known for some time that for some problems it is possible to get a lower index problem by keeping the highest index constraints. That is, those constraints that took the most differentiations to get. For example, we could discard (5e) and (5f) and get the DAE
\begin{align*}
\dot{x}_1 &= x_2, \quad (9a) \\
\dot{x}_2 &= -x_5 x_1 - ax_2 + ux_3, \quad (9b) \\
\dot{x}_3 &= x_4, \quad (9c) \\
\dot{x}_4 &= -x_5 x_3 - ax_4 - g - ux_1, \quad (9d) \\
0 &= x_2^2 + x_4^2 - x_5 L^2 - x_5 g, \quad (9e) \\
x_1(0) &= 0, \quad (9f) \\
x_3(0) &= L, \quad (9g)
\end{align*}
which is indeed semi-explicit. Moreover, it is index 1 in \(x_1, x_2, x_3, x_4, x_5\) treating \(u\) as an input, since \((9e)\) can be solved for \(x_5\), the only state variable for which we do not have a differential equation. The price to be paid for this index reduction is that \((9a)–(9e)\) has more solutions than \((5a)–(5g)\) has. This may cause so-called drift-off effects when solving initial value problems for a given \(u\) on longer intervals. In extreme cases this may even lead to a break down of the used integration method.

This problem, however, can be often alleviated by using constraint stabilization [2]. For example, let \(s_1, s_2\) be two negative numbers and \(r + \kappa r + \beta = (r - s_1)(r - s_2)\). Then instead of \((9)\) we could consider

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - ax_2 + ux_3, \\
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= -x_5 x_3 - ax_4 - g - ux_1, \\
0 &= (x_2^2 + x_4^2 - x_5 L^2 - x_3 g) + \kappa (x_1 x_2 + x_3 x_4) + \beta (x_1^2 + x_3^2 - L^2), \\
x_1(0) &= 0, \\
x_3(0) &= L, \\
x_2(0) &= 0, \\
x_4(0) &= 0.
\end{align*}
\]

System \((10)\) is also index 1 in \(x\), the difference is that the additional dynamics are stabilized.

### 2.3. Index reduction to a semi-explicit DAE

Another possibility for preserving the semi-explicit structure is to proceed as in Subsection 2.1 but keeping \(Z_1^T\) constant. The price for this is that the resulting DAE is not globally index 1 but exhibits so-called impasse points. As a consequence we need to switch between different reduced DAEs changing \((6)\) into an optimal control problem for a hybrid system of DAEs. Note that in the context of DT the modes of the hybrid system are also called phases.

In our example we observe that for a motion that goes more than a quarter of the way around the circle it is not possible to pick a single subset of the equations in \((5)\). The two obvious choices are to use coordinate partitioning, which corresponds to special choices of \(Z_1^T\). Selecting \(x_1, x_2\) as differential variables gives

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - ax_2 + ux_3, \\
0 &= x_1^2 + x_3^2 - L^2, \\
0 &= x_1 x_2 + x_3 x_4, \\
0 &= x_2^2 + x_4^2 - x_5 L^2 - x_3 g
\end{align*}
\]

for the first phase and selecting \(x_3, x_4\) as differential variables gives

\[
\begin{align*}
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= -x_5 x_3 - ax_4 - g - ux_1, \\
0 &= x_1^2 + x_3^2 - L^2,
\end{align*}
\]
0 = x_1 x_2 + x_3 x_4, \quad (12d) \\
0 = x_2^2 + x_4^2 - x_5 L^2 - x_3 g \quad (12e)

for the second phase.

Note that we do not specify the branch of the square root since we will have a starting value on one branch and the nonlinear solver should theoretically take care of that if the initial guess is good enough. However, each of (11) and (12) actually defines two different differential equations due to the nonlinearity in (11c). This can create problems. Accordingly, we can consider the following reformulations into four phases. Phase 1 for \( x_3 > 0 \) is

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - a x_2 + u x_3, \\
x_3 &= \sqrt{L^2 - x_1^2}, \\
x_4 &= -x_1 x_2 / x_3, \\
x_5 &= \frac{1}{L^2} (x_2^2 + x_4^2 - x_3 g).
\end{align*}
\]

For \( x_3 < 0 \) we have Phase 3,

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - a x_2 + u x_3, \\
x_3 &= -\sqrt{L^2 - x_1^2}, \\
x_4 &= -x_1 x_2 / x_3, \\
x_5 &= \frac{1}{L^2} (x_2^2 + x_4^2 - x_3 g).
\end{align*}
\]

The other two phases are Phase 2,

\[
\begin{align*}
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= -x_5 x_3 - a x_4 - g - u x_1, \\
x_1 &= \sqrt{L^2 - x_3^2}, \\
x_2 &= -x_3 x_4 / x_1, \\
x_5 &= \frac{1}{L^2} \left( x_2^2 + x_4^2 - x_3 g \right),
\end{align*}
\]

and Phase 4,

\[
\begin{align*}
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= -x_5 x_3 - a x_4 - g - u x_1, \\
x_1 &= -\sqrt{L^2 - x_3^2}, \\
x_2 &= -x_3 x_4 / x_1, \\
x_5 &= \frac{1}{L^2} \left( x_2^2 + x_4^2 - x_3 g \right).
\end{align*}
\]

Note that these four phases can be formulated as ODEs by eliminating the states which are explicitly given by the constraints. In this way, Phase 1 and Phase 3 becomes an ODE problem in terms of \( x_1, x_2 \) and Phase 2 and Phase 4 becomes an ODE problem in terms of \( x_3, x_4 \).
2.4. **Minimal coordinates.** It is sometimes possible to reparameterize the state manifold to get a nicer system. Sometimes this is done locally and sometimes it can be done globally. A distinguished parameterization is by minimal coordinates, that is by a number of parameters equaling the dimension of the state manifold, since this will turn the DAE into an ODE.

In the case of our working example, it is possible to do a reparameterization to minimal coordinates globally. The nonlinear change of coordinates in $x, y$ space which we apply to the original DAE (2) is

$$x = L \sin \theta, \quad y = L \cos \theta. \tag{17}$$

Then taking the time derivatives, we get

$$\dot{x} = L \cos(\theta) \dot{\theta}, \tag{18a}$$
$$\ddot{x} = -L \sin(\theta) (\dot{\theta})^2 + L \cos(\theta) \ddot{\theta}, \tag{18b}$$
$$\dot{y} = -L \sin(\theta) \dot{\theta}, \tag{18c}$$
$$\ddot{y} = -L \cos(\theta) (\dot{\theta})^2 - L \sin(\theta) \ddot{\theta}. \tag{18d}$$

The resulting dynamics, letting $z_1 = \theta$ and $z_2 = \dot{\theta}$, are

$$\dot{z}_1 = z_2, \tag{19a}$$
$$-L \sin(z_1)(z_2)^2 + L \cos(z_1) \dot{z}_2 = -\lambda L \sin(z_1) - aL \cos(z_1) z_2 + uL, \tag{19b}$$
$$-L \cos(z_1)(z_2)^2 - L \sin(z_1) \dot{z}_2 = -\lambda L \cos(z_1) - L \cos(z_1) z_2 - g - uL \sin(z_1). \tag{19c}$$

We multiply (19b) and (19c) by the invertible matrix

$$\begin{bmatrix}
\cos z_1 & -\sin z_1 \\
\sin z_1 & \cos z_1
\end{bmatrix},$$

which is a row compression of the matrix in front the $\dot{z}_2$ terms and we get

$$\dot{z}_1 = z_2, \tag{20a}$$
$$L \dot{z}_2 = -aL z_2 + g \sin z_1 + uL, \tag{20b}$$
$$0 = L(z_2)^2 - \lambda L - g \cos z_1. \tag{20c}$$

Since $\lambda$ is determined by (20c) and decouples from (20a)–(20b), we are left with the ODE

$$\dot{z}_1 = z_2, \tag{21a}$$
$$\dot{z}_2 = -a z_2 + \frac{g}{L} \sin z_1 + u. \tag{21b}$$

The corresponding initial conditions are

$$z_1(0) = 0, \quad z_2(0) = 0. \tag{22}$$

Note that the cost function is now

$$J(z, u) = \int_0^T cu^2 + d(L \sin(z_1) - L \sin(t+\alpha))^2 + d(L \cos(z_1) - L \cos(t+\alpha))^2 dt. \tag{23}$$

Actually, the ODE system (21) corresponds to the equation of motion one gets when building the Hamiltonian from kinetic and potential energy expressed in the given minimal coordinates and the related momenta (called generalized positions and momenta in this context) and applying the so-called Hamiltonian formalism.
In general, it may not be possible to get an alternative parameterization. However, doing so has proved useful in a number of places including using Euler angles in mechanics and robotics and for some space navigation problems such as the low thrust earth to moon orbit transfer [4].

3. Approaches and their requirements. There is a large variety of approaches to solve (1) numerically. We discuss several prominent methods in the following.

3.1. Necessary conditions using global index reduction. One standard approach, called indirect approach in the context of optimal control problems, is to derive necessary conditions for the optimal solution and to solve the arising problem. In order to get necessary conditions (assuming continuous controls and states) we can use calculus of variation. But this requires the constraining DAE to be index 1 as a control problem (i.e., of virtual index 1), see [23]. For our example, this is the case for (8) since it is index 1 for fixed control.

Introducing Lagrange multipliers yields

\[
\mathcal{L} = \int_0^T \left( cu^2 + d(x_1 - L \sin(t + \alpha))^2 + d(x_3 - L \cos(t + \alpha))^2 \right) dt \quad (24a)
\]

\[
- \int_0^T \lambda_1 (\cos t(\dot{x}_1 - x_2) - \sin t(\dot{x}_3 - x_4)) \, dt \quad (24b)
\]

\[
- \int_0^T \lambda_2 (\cos t(\dot{x}_2 + ax_2 - ux_3 + x_1x_5) - \sin t(\dot{x}_4 + ax_4 + ux_1 + x_3x_5 + g)) \, dt \quad (24c)
\]

\[
- \int_0^T \lambda_3 (x_1^2 + x_2^2 - L^2) \, dt - \int_0^T \lambda_4 (x_1x_2 + x_3x_4) \, dt \quad (24d)
\]

\[
- \int_0^T \lambda_5 (x_2^2 + x_4^2 - x_5^2L^2 - gx_3) \, dt, \quad (24e)
\]

which, using the usual calculus of variations and integration by parts, produces the necessary conditions

\[
\delta \lambda_1 : 0 = \cos t(\dot{x}_1 - x_2) - \sin t(\dot{x}_3 - x_4), \quad x_1(0) = 0, \ x_2(0) = 0, \quad (25a)
\]

\[
\delta \lambda_2 : 0 = \cos t(\dot{x}_2 + ax_2 - ux_3 + x_1x_5) - \sin t(\dot{x}_4 + ax_4 + ux_1 + x_3x_5 + g), \quad (25b)
\]

\[
\delta \lambda_3 : 0 = x_1^2 + x_2^2 - L^2, \quad (25c)
\]

\[
\delta \lambda_4 : 0 = x_1x_2 + x_3x_4, \quad (25d)
\]

\[
\delta \lambda_5 : 0 = x_2^2 + x_4^2 - x_5^2L^2 - gx_3, \quad (25e)
\]

\[
\delta x_1 : 0 = \frac{d}{dt}(\lambda_1 \cos t) - \lambda_2 \cos tx_5 + \lambda_2 \sin t \, u - 2\lambda_3 x_1 - \lambda_3 x_2 + 2d(x_1 - L \sin(t + \alpha)), \quad (25f)
\]

\[
\delta x_2 : 0 = \frac{d}{dt}(\lambda_2 \cos t) + \lambda_1 \cos t - \lambda_2 \cos tu - \lambda_4 x_1 - 2\lambda_5 x_2, \quad (25g)
\]

\[
\delta x_3 : 0 = - \frac{d}{dt}(\lambda_1 \sin t) + \lambda_2 \cos tu + \lambda_2 \sin tx_5 - 2\lambda_3 x_3 - \lambda_4 x_4 + \lambda_5 g + 2d(x_3 - L \cos(t + \alpha)), \quad (25h)
\]

\[
\delta x_4 : 0 = - \frac{d}{dt}(\lambda_2 \sin t) - \lambda_1 \sin t + \lambda_2 \sin tu - \lambda_4 x_3 - 2\lambda_5 x_4, \quad (25i)
\]

\[
\delta x_5 : 0 = - \lambda_2 \cos tx_1 + \lambda_2 \sin tx_3 + \lambda_5 L^2, \quad \lambda_1(T) = 0, \ \lambda_2(T) = 0. \quad (25j)
\]

\[
\delta u : 0 = \lambda_2 \cos tx_3 + \lambda_2 \sin tx_1 + 2cu. \quad (25k)
\]
Taking the time derivatives in (25f) through (25i) we get our necessary conditions as

\[ 0 = \cos(\dot{x}_1 - x_2) - \sin(t(\dot{x}_3 - x_4)), \quad x_1(0) = 0, \ x_2(0) = 0, \]  
\[ 0 = \cos(\dot{x}_2 + ax_2 - ux_3 + x_1x_5) - \sin(t(\dot{x}_4 + ax_4 + ux_1 + x_3x_5 + g)), \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1x_2 + x_3x_4, \]  
\[ 0 = x_2^2 + x_4^2 - x_5L^2 - gx_3, \]

Thus the necessary conditions obtained constitute a nonlinear DAE boundary value problem where again the DAE is linear in the derivatives but with a time-dependent matrix.

3.2. Necessary conditions using phases. In order to get rid of the time dependence of the previous subsection, we can derive necessary conditions for the problem formulation with phases.

In the present case we consider the two phases (11) and (12). To simplify the relationship and derivation of the necessary conditions we rewrite them as

\[ 0 = \dot{x}_1 - x_2, \]  
\[ 0 = \dot{x}_3 + ax_2 - ux_3 + x_1x_5, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1x_2 + x_3x_4, \]  
\[ 0 = x_2^2 + x_4^2 - x_5L^2 - gx_3, \]

and

\[ 0 = \dot{x}_3 - x_4, \]  
\[ 0 = \dot{x}_4 + ax_4 + ux_1 + x_3x_5 + g, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1x_2 + x_3x_4, \]  
\[ 0 = x_2^2 + x_4^2 - x_5L^2 - gx_3. \]

Calculus of variations then leads to necessary conditions for each phase consisting of

\[ 0 = \dot{x}_1 - x_2, \]  
\[ 0 = \dot{x}_2 + ax_2 - ux_3 + x_1x_5, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1x_2 + x_3x_4, \]  
\[ 0 = \dot{x}_3 - x_4, \]  
\[ 0 = \dot{x}_4 + ax_4 + ux_1 + x_3x_5 + g, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1x_2 + x_3x_4, \]  
\[ 0 = x_2^2 + x_4^2 - x_5L^2 - gx_3. \]
\[ 0 = x_2^2 + x_4 - x_3 L^2 - gx_3, \]  
\[ 0 = \lambda_1 - \lambda_2 x_5 - 2 \lambda_3 x_1 - \lambda_4 x_2 + 2d(x_1 - L \sin(t + \alpha)), \]  
\[ 0 = \lambda_2 + \lambda_1 - \lambda_2 a - \lambda_4 x_1 - 2 \lambda_5 x_2, \]  
\[ 0 = \lambda_2 x_3 - 2 \lambda_5 x_4, \]  
\[ 0 = -\lambda_2 x_1 + \lambda_5 L^2, \]  
\[ 0 = \lambda_2 x_3 + 2cu, \] 
\[ \text{and} \]
\[ 0 = \dot{x}_3 - x_4, \]  
\[ 0 = \dot{x}_4 + ax_4 + ux_1 + x_3 x_5 + g, \]  
\[ 0 = x_1^2 + x_3^2 - L^2, \]  
\[ 0 = x_1 x_2 + x_3 x_4, \]  
\[ 0 = x_2^2 + x_4^2 - x_5 L^2 - gx_3, \]  
\[ 0 = -\lambda_2 u - 2 \lambda_3 x_1 - \lambda_4 x_2 + 2d(x_1 - L \sin(t + \alpha)), \]  
\[ 0 = -\lambda_4 x_1 - 2 \lambda_5 x_2, \]  
\[ 0 = \lambda_1 - \lambda_2 x_5 - 2 \lambda_3 x_3 - \lambda_4 x_4 + \lambda_5 g + 2d(x_3 - L \cos(t + \alpha)), \]  
\[ 0 = \lambda_2 + \lambda_1 - \lambda_2 a - \lambda_4 x_3 - 2 \lambda_5 x_4, \]  
\[ 0 = -\lambda_2 x_3 + \lambda_5 L^2, \]  
\[ 0 = -\lambda_2 x_1 + 2cu. \] 

Note that we may get these relations from the global case if we formally replace \((\cos t, \sin t)\) by \((1, 0)\) and \((0, -1)\), respectively.

Using the first phase alone, the boundary conditions are given by
\[ x_1(0) = 0, \quad x_2(0) = 0, \quad \lambda_1(T) = 0, \quad \lambda_2(T) = 0. \]

This leads to a well-posed boundary-value problem for smaller \(T\). For larger \(T\) the DAE exhibits an impasse point such that we need to switch to the second phase, say at the point \(\hat{t}\) prior to the impasse point. Of course, we require continuity with respect to the state variables, although since there is a switch in what is differential and what is algebraic in the two phases, it might not be correct to explicitly require continuity in all state variables since that could give an overdetermined BVP. Also since the Lagrange multipliers \(\lambda_1\) and \(\lambda_2\) are related to different equations, requiring continuity in the multipliers may not be correct.

We derive then what the correct boundary conditions are at \(\hat{t}\). Observe that satisfying the DAEs of the necessary boundary-value problem yields the remaining terms
\[ \delta \mathcal{L} = -\gamma_1 (\delta \mathcal{L}_1(\hat{t}) - \delta \mathcal{L}_1(\hat{t})) - \gamma_2 (\delta \mathcal{L}_2(\hat{t}) - \delta \mathcal{L}_2(\hat{t})) - \lambda_1 \delta \mathcal{L}_1|_{0}^{\hat{t}} - \lambda_2 \delta \mathcal{L}_2|_{0}^{\hat{t}}, \]
where \(\gamma_1, \gamma_2\) denote the Lagrange multipliers for the continuity conditions and the bars distinguish the variables of the two phases. An underbar is up to \(\hat{t}\) and an overbar is after \(\hat{t}\).
Using $\delta x_1(0) = 0$, $\delta x_2(0) = 0$, due to the given initial values we have

$$
\delta L = (\gamma_1 - \delta_1(\hat{t}) \delta x_1(\hat{t}) + (\gamma_2 - \delta_2(\hat{t})) \delta x_2(\hat{t}) - \gamma_1 \delta x_1(\hat{t}) - \gamma_2 \delta x_2(\hat{t})
- \lambda_1(T) \delta x_3(T) + \bar{\lambda}_1(\hat{t}) \delta x_3(\hat{t}) - \lambda_2(T) \delta x_4(T) + \bar{\lambda}_2(\hat{t}) \delta x_4(\hat{t}).
$$

(31)

Variation of $\delta x_1(\hat{t})$, $\delta x_2(\hat{t})$ and $\delta x_3(T)$, $\delta x_4(T)$ in (31) yields

$$
\gamma_1 = \delta_1(\hat{t}), \quad \gamma_2 = \delta_2(\hat{t}), \quad \bar{\lambda}_1(T) = 0, \quad \bar{\lambda}_2(T) = 0,
$$

(32)

and thus (31) becomes just

$$
\delta L = -\delta_1(\hat{t}) \delta x_1(\hat{t}) - \delta_2(\hat{t}) \delta x_2(\hat{t}) + \bar{\lambda}_1(\hat{t}) \delta x_3(\hat{t}) + \bar{\lambda}_2(\hat{t}) \delta x_4(\hat{t}).
$$

(33)

However, we cannot vary each of $\delta x_1(\hat{t})$, $\delta x_2(\hat{t})$, $\delta x_3(\hat{t})$, $\delta x_4(\hat{t})$ independently since there are relationships from the algebraic constraints. Instead we must take into account the dependencies due to the algebraic constraints. Taking the differentials of (5e) and (5f) we get

$$
x_1 \delta x_1 + x_3 \delta x_3 = 0,
$$

(34)

$$
x_1 \delta x_2 + x_2 \delta x_1 + x_3 \delta x_1 + x_4 \delta x_3 = 0.
$$

(35)

Note that the third constraint (5g) does not matter since it determines $\delta x_3$. Multiplying (33) by $\bar{\lambda}_1(\hat{t})$ we get

$$
\delta L \bar{\lambda}_1(\hat{t}) = -\delta_1(\hat{t}) \lambda_1(\hat{t}) \delta x_1(\hat{t}) - \delta_2(\hat{t}) \lambda_1(\hat{t}) \delta x_2(\hat{t}) + \bar{\lambda}_1(\hat{t}) \lambda_1(\hat{t}) \delta x_3(\hat{t})
+ \bar{\lambda}_2(\hat{t}) \lambda_1(\hat{t}) \delta x_4(\hat{t})
= \lambda_1(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t}) + \lambda_2(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t})
+ \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t}) + \bar{\lambda}_1(\hat{t}) \lambda_1(\hat{t}) \delta x_3(\hat{t}) + \bar{\lambda}_2(\hat{t}) \lambda_1(\hat{t}) \delta x_4(\hat{t}).
$$

Multiplying again by $\bar{\lambda}_1(\hat{t})$ gives

$$
\delta L \bar{\lambda}_1(\hat{t})^2 = \lambda_1(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t}) - \lambda_2(\hat{t}) \lambda_2(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t})
+ \lambda_3(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t})
+ \bar{\lambda}_1(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t})
+ \bar{\lambda}_2(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}).
$$

At this stage we can vary $\delta x_3(\hat{t})$, $\delta x_4(\hat{t})$ independently to obtain the conditions

$$
\lambda_1(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t}) - \lambda_2(\hat{t}) \lambda_2(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t})
+ \lambda_3(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t})
+ \bar{\lambda}_1(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t})
+ \bar{\lambda}_2(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}) = 0,
$$

(37)

$$
\lambda_2(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}) + \bar{\lambda}_2(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}) = 0.
$$

(38)

Since $\hat{t}$ is such that $\bar{\lambda}_1(\hat{t}) \neq 0$ we can divide (38) by $\bar{\lambda}_1(\hat{t})$ to give

$$
\lambda_2(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}) + \bar{\lambda}_2(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}) = 0.
$$

(39a)

But then using (39a) in (37) and then dividing by $\bar{\lambda}_1(\hat{t})$ gives

$$
\lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t}) + \bar{\lambda}_2(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_4(\hat{t}) + \bar{\lambda}_1(\hat{t}) \lambda_1(\hat{t}) \lambda_3(\hat{t}) \lambda_3(\hat{t}) \delta x_3(\hat{t}) = 0.
$$

(39b)

Thus the correct boundary conditions are (32) and (39).

Although a similar splitting into two phases is not necessary for the case of the previous subsection, we want to note that the resulting necessary conditions would consist of two equal DAEs, the initial and end conditions

$$
x_1(0) = 0, \quad x_2(0) = 0, \quad \bar{\lambda}_1(T) = 0, \quad \bar{\lambda}_2(T) = 0
$$
and the continuity conditions

\[
\begin{align*}
\cos \hat{t} x_1(\hat{t}) - \sin \hat{t} x_3(\hat{t}) &= \cos \hat{t} x_1(\hat{t}) - \sin \hat{t} x_3(\hat{t}), \\
\cos \hat{t} x_2(\hat{t}) - \sin \hat{t} x_4(\hat{t}) &= \cos \hat{t} x_2(\hat{t}) - \sin \hat{t} x_4(\hat{t}), \\
\lambda_1(\hat{t}) &= \lambda_1(\hat{t}), & \lambda_2(\hat{t}) &= \lambda_2(\hat{t}),
\end{align*}
\]

which can be reduced to

\[
\begin{align*}
x_1(\hat{t}) &= x_1(\hat{t}), & x_2(\hat{t}) &= x_2(\hat{t}), & \lambda_1(\hat{t}) &= \lambda_1(\hat{t}), & \lambda_2(\hat{t}) &= \lambda_2(\hat{t}),
\end{align*}
\]

or

\[
\begin{align*}
x_3(\hat{t}) &= x_3(\hat{t}), & x_4(\hat{t}) &= x_4(\hat{t}), & \lambda_1(\hat{t}) &= \lambda_1(\hat{t}), & \lambda_2(\hat{t}) &= \lambda_2(\hat{t}),
\end{align*}
\]

depending on the location of \( \hat{t} \).

### 3.3. Necessary conditions using minimal coordinates.

Of course, it is also possible to use necessary conditions for the formulation (21) in minimal coordinates. In the same way as above, we obtain

\[
\begin{align*}
0 &= \dot{z}_1 - z_2, & (40a) \\
0 &= \dot{z}_2 + az_2 - \frac{g}{L} \sin z_1 - u, & (40b) \\
0 &= \dot{\lambda}_1 + \frac{g}{L} \cos z_1 \lambda_2 + 2dl^2(\sin z_1 - \sin(t + \alpha)) \cos z_1 \nonumber \\
&\quad - 2dl^2(\cos z_1 - \cos(t + \alpha)) \sin z_1, & (40c) \\
0 &= \dot{\lambda}_2 + \lambda_1 - a\lambda_2, & (40d) \\
0 &= \lambda_2 + 2cu, & (40e)
\end{align*}
\]

together with the boundary conditions

\[
z_1(0) = 0, \quad z_2(0) = 0, \quad \lambda_1(T) = 0, \quad \lambda_2(T) = 0. \quad (41)
\]

Note that even though the constraining DAE is an ODE here, the necessary conditions (40) form a DAE containing algebraic constraints.

### 3.4. Direct transcription.

The idea of direct transcription, also called the direct approach, is to completely discretize the given optimal control problem by choosing finite dimensional vectors \( x_h \) and \( u_h \) and then to solve the arising finite dimensional optimization problem

\[
\begin{align*}
\min & \tilde{J}(x_h, u_h), & (42a) \\
\tilde{F}(x_h, u_h) &= 0, & (42b)
\end{align*}
\]

by standard techniques from finite optimization like steepest descent or sequential quadratic programming. Note that there is a huge variety as to how we can discretize the given optimal control problem which consists of discretizing the integral and the constraining DAE. In particular, this can be done independently or, rewriting the integral as a solution of an ODE, by the same method.

The problem with direct transcription is that it may not work if the constraining DAE does not possess special properties such as index 1 or being semi-explicit.
3.5. **Control parametrization.** The idea of control parameterization is that one parameterizes the set of allowable controls by a finite set of parameters. Then one simulates the dynamics for a given value of the parameters and evaluates the cost. This evaluation of the cost for a given control is the function fed into an optimizer which optimizes over the control parameters. So two pieces of software are needed: an optimizer and a simulation package. In this way control parametrization can be seen as to semi-discretize the given optimal control problem by requiring the controls to lie in a finite dimensional subspace of the space of continuous functions. Choosing a basis in this finite dimensional subspace, e. g., a Lagrange type basis, a control can be specified by a finite number of vectors $u_1, \ldots, u_N$. We may write this as

$$u = U(\cdot; u_1, \ldots, u_N).$$

(43)

For a given set of parameters and a given (consistent) initial value, we can integrate the DAE. If the initial guess of the parameters is good enough, this defines (locally) a mapping according to

$$x = S(\cdot; U(\cdot; u_1, \ldots, u_N)).$$

(44)

Thus, control parametrization can be written as

$$J(u_1, \ldots, u_N) = J(S(\cdot; U(\cdot; u_1, \ldots, u_N)), U(\cdot; u_1, \ldots, u_N)) \to \min.$$  \hspace{1cm} (45)

Of course, we need an integration method which is able to solve the arising DAE initial value problems. This may again require performing one of the reformulations presented above. The resulting discrete optimization problem can again be treated by standard methods.

4. **Numerical experiments.** In order to show the needs and problems of the various possible numerical approaches, we performed a wide range of numerical tests using the accompanying example with parameters $T = 2.2$, $a = 0.5$, $\alpha = 0.1$, $g = 4$, $c = 1$, $d = 100$, $L = 2$ if not stated otherwise. The choice of the parameters $c, d$ emphasizes tracking over the size of the control.

4.1. **Direct transcription.** Our starting point was the use of direct transcription because we had the codes SOCX [3] or GPOPS II [30] at hand.

Although in [9] we solved a simpler index three problem that was similar to this one, we cannot expect that this is usually the case. Actually, both GPOPS II and SOCX failed to solve the problem (3) as stated.

As the typical requirement for DT is that the system is semi-explicit index 1 we turned to the formulation (9). GPOPS II was then able to solve the optimal control problem consisting of (9) and (2h) with mesh tolerances of $10^{-8}$. Equation (9e) will hold to the accuracy of function evaluation if it is incorporated into the dynamics evaluation. The residuals in (5e) and (5f) are plotted in Figure 1. Note that the constraint error is distributed along the time interval.

We also tried to solve the problem using GPOPS II and Phases 1 and 2 in the form of (13) and (15) in order to avoid working with additional artificial dynamics. The problem failed to work as written.

For the discussion we focus on Phase 1 since that suffices. Similar comments hold for Phase 2. It is important to note that with DT not all equations are handled the same. Equations that are part of a function evaluation are evaluated as written to the accuracy of the expressions. On the other hand, equations that are written as
path constraints become part of the NLP problem that is solved. Mathematically equivalent formulations can have very different computational properties.

Option 1. We include the equations for $x_3, x_4, x_5$ into the dynamics and there are no path constraints. That is on each phase we have an ODE. For example, on Phase 1 we would have

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - ax_2 + u x_3, \\
x_3 &= \sqrt{L^2 - x_1^2}, \\
x_4 &= -x_1 x_2 / x_3, \\
x_5 &= \frac{1}{L^2} (x_2^2 + x_4^2 - x_3 g),
\end{align*}
\]

where (46c)–(46e) are used to evaluate $\dot{x}_1, \dot{x}_2$. Here, given $x_1, x_2$, the evaluation of $x_3, x_4, x_5$ and then $\dot{x}_1, \dot{x}_2$ is carried out to function evaluation accuracy. With this formulation of the phase we easily solved the problem with GPOPS II. The results are shown in Figures 2–4.

Option 2. In Option 2 we treat the phase as a DAE and put the constraints in as path constraints so that we have

\[
\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - ax_2 + u x_3, \\
0 &= -x_3 + \sqrt{L^2 - x_1^2}, \\
0 &= -x_4 - x_1 x_2 / x_3, \\
0 &= -x_5 + \frac{1}{L^2} (x_2^2 + x_4^2 - x_3 g).
\end{align*}
\]

This version failed to be solvable with GPOPS II, at least it did not converge for the initial guesses used for Option 1. With Option 2 the path constraints at each grid point become part of the hundreds of equations that must be solved by the NLP.
Figure 2. Trajectory and control with Option 1 formulation (46).

Figure 3. State solutions for (46) using GPOPS II with target (dashed line).

Figure 4. Tracking error of state solutions for (46).

solver. This means both that they are solved to lower accuracy and that either the solver could not deal with the nonlinear equations or we needed much better initial guesses for Option 2 than for Option 1.
Option 3. To test whether the problem with Option 1 was in the nonlinear solver we examined the following formulation which had path constraints like Option 2 but moved the nonlinearities from the path constraints into the function evaluations like Option 1 by using

\begin{align}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -x_5 x_1 - ax_2 + u x_3, \\
0 &= -\tilde{x}_3 + \sqrt{L^2 - x_1^2}, \\
0 &= -\tilde{x}_4 - x_1 x_2 / \tilde{x}_3, \\
0 &= -\tilde{x}_5 + \frac{1}{L^2} (\tilde{x}_3^2 + \tilde{x}_4^2 - \tilde{x}_3 g),
\end{align}

with the path constraints

\begin{align}
0 &= \tilde{x}_3 - x_3, \\
0 &= \tilde{x}_4 - x_4, \\
0 &= \tilde{x}_5 - x_5.
\end{align}

The way this formulation works is that \( \tilde{x}_3, \tilde{x}_4, \tilde{x}_5 \) are evaluated as functions of \( x_1, x_2 \) at each grid point as function evaluations, but then the path constraints (48f)–(48h) determine \( x_3, x_4, x_5 \) in evaluating the dynamic equations (48a) and (48b). GPOPS II solved this formulation easily with the same initial guesses that failed with Option 2.

Finally, we should mention that GPOPS II also easily solved the problem in the formulation (40) as expected.

In summary, both GPOPS II and SOCX can be applied to some DAEs but for many problems they have a need to designate variables as either continuous or algebraic. While this works fine it creates some challenges, for example if there are different phases and the choice of what is a state and a control changes across the phase.

In order to get more insight into the possible behavior of DT we implemented a version of DT based on collocation with grid adaptation using Gauß nodes for the differential equations including the evaluation of the costs and Lobatto nodes (called Legendre-Gauß-Lobatto by Hager [12]) for the algebraic equations similar to the BVP approach in [26]. The discrete optimization problem was treated by a damped Gauß-Newton method and the linear systems were solved by a public domain sparse LU package. Trying to solve the optimal control problem in the original version failed due to rank deficiency of the arising linear systems. Turning to rank revealing QR decomposition and drastically relaxing the required tolerance finally yielded a correct solution. Nevertheless the needed computing time was in the range of several hours and in general one would not trust the obtained solution because of the observed rank deficiency. All other versions of the problem formulation addressed above were successfully solved within several minutes computing time. However, it should be noted that this was done by exploiting several packages and CPU time was far from optimized.

4.2. Necessary conditions. For the solution of the DAE BVPs comprising the necessary conditions we used a solver as described in [26] based on collocation with Gauß nodes for the differential equations and with Lobatto nodes for the algebraic equations. Since the implementation does not require a special structure of the
DAE, it could be applied to the formulation (26). Nevertheless, we also treated the formulations (29)–(30) using phases and (40) in minimal coordinates. All problem formulations were solved successfully within a few seconds of computing time.

This example also illustrates the usefulness of having a DAE BVP solver that can accept problems where the derivatives are not given explicitly.

4.3. Control parameterization. In order to solve the given problem by control parameterization we need an integrator to solve the constraining DAE for a given control. Since the publicly available software for this task typically requires the DAE to be semi-explicit (or at least being properly stated in the sense of März and coworkers [28]) a good choice of the formulation to use seems to be (9). But since we have to solve initial value problems here, we are confronted with drift-off effects due to additional dynamics as the constraints (5e) and (5f) are no longer imposed. To illustrate that this can be a major problem with control parameterization, we took (9), applied a control \( u \), and simulated with an integrator designed for index one DAEs. We used the Matlab BDF based integrator ode15s.

Choosing the simple control of \( u = 2 \) on the interval \([0, 2]\) we get the trajectory in Figure 5 and the residuals in the constraints shown in Figure 6. The calculation was done with a relative error tolerance of \( 10^{-8} \) which is the same as the mesh tolerance used with GPOPS II.

![Figure 5. Trajectory for (9) using Matlab ode15s with requested RelTol of \( 10^{-8} \).](image)

We see that in Figure 5 that the trajectory appears circular as expected. However, look at Figure 6 which shows the integration error in the other constraints. Two things are immediately apparent. First unlike in Figure 1 the error does not appear random. It is much smoother. Also the error in Figure 6 is a couple of orders magnitude greater than the error in Figure 1. The situation becomes even more dramatic on a longer interval as shown in Figures 7 and 9.

We see that the trajectory is not the expected circular motion on a circle in Figure 8. Looking at Figure 9 we see that as expected when drift is present the results are reasonable for a while but then the drift builds up and the results are no longer physically reasonable. In fact the error is now \( O(1) \) on a moderately short interval.

Consequently, the stabilized version (10) should be used. But there is another type of stabilization going on in that our model has friction present. To illustrate the different impacts of using a stabilized and unstabilized dynamics and the model
having internal damping we shall take the control \( u \) given in Figure 10 and integrate system (9) and the stabilized system (10) with and without internal friction. The requested tolerances were \( 10^{-6} \) relative tolerance and \( 10^{-7} \) absolute tolerance. The time interval was \( 0 \leq t \leq 40. \)

With \( a = 0.5 \) and no stabilization the integration failed at \( t = 13.8 \) due to step size failure. The other three cases are below in Figures 11–13. The stabilized dynamics used a polynomial of \( r^2 + 3r + 2 \).

As noted for \( a = 0.5 \) we needed stabilized dynamics to compute the answer. For \( a = 0 \) we see that the stabilized dynamics gave better residuals for the index one and index two constraints but they gave the same error in the index three constraint.

One can think of direct transcription as control parameterization carried to the extreme where the number of parameters is a multiple of the time grid. That means that the algebraic variables \( x_5 \) and \( u \) are discretized the same. However, in the normal application of control parameterization the number of control parameters

---

**Figure 6.** Residual for constraints (5e) (length of the pendulum) and (5f) (velocity constraint) using the index one formulation (9). Requested error is \( 10^{-8} \) in both constraints using ode15s.

---

**Figure 7.** Residual for constraints (5e) (length of the pendulum) and (5f) (velocity constraint) using the index one formulation (9). Requested error is \( 10^{-8} \) in both constraints using ode15s.
is much lower and the time grid fine enough so that the computed cost accurately reflects the cost of using the approximate parameterized control. That means
that $x_5$ and $u$ are approximated differently and $u$ is approximated by a lower order approximation.
In this particular problem that is not a big issue. But for problems where the
direct transcription works because there is an index one virtual control, this could
cause problems because the variable $u$ must now be considered part of the state and
the discretization is now lower order in some of the variables.

We now give a simple illustration of using control parametrization on our original
problem on the interval $[0, 2]$. As a truth model we solve (10) for $x_5$ and then solve
the resulting ODE control system with GPOPS II. The optimal control is given in
Figure 14.

To illustrate the use of control parametrization we will parametrize the control
by cubic splines on a uniform grid and use the Matlab ode1s integrator. There are
several issues that must now be dealt with. First, like all nonlinear optimization problems a good initial guess for the control parameters may be required. Secondly if the order of the integrator is higher than the order of the splines, then the integration may fail because the steps get too small at places of reduced smoothness. For the case of (10) we will use cubic splines as provided by Matlab and restrict the integrator to being third order.

We found that our simple code failed if the initial guess was too far away. As an initial guess we took the function in (15) and 21 parameters.

![Figure 15](image)

**Figure 15.** Initial guess using control parameterization.

The initial guess in Figure 15 resulted in the control in Figure 16 which should be compared to Figure 14. It should be noted that this same code produced less

![Figure 16](image)

**Figure 16.** Control computed from initial guess in Figure 15. Cost 14.7309.

optimal estimates with less desirable initial guesses. Without restricting the order of the integrator to three there were repeated failures of the function evaluation. Also we have to balance the error of the numerical integrator with the requested tolerances in the optimization. The integration error needs to be tighter than the desired cost evaluation error in the optimizer.

In order to judge the quality of the obtained result we also implemented the same kind of control parameterization but using our own IVP solver based on the same
discretization as used for solving the BVPs in Subsection 4.2 combined with steepest descent. To avoid problems with nonsmooth $u$ we did the integration piecewise with respect to the grid defining $u$. We were able to compute a solution with cost 14.548. The solution was about the same when using cubic splines or linear interpolation and when using (8) or (10). The computing time was about 35 minutes due to a very poor convergence behavior of steepest descent which may be related to the involved numerical differentiation of the integration process that includes a step size control.

Control parameterization is a popular technique in industry because, in principle, it is easy to implement using standard software packages. On many problems it can get a useful answer. On the other hand it has a number of limitations.

There are two cases where the use of control parameterization can be helpful. One is where there is a restricted class of controls which are easily parametrized. Another is where the value of the cost of the control is coming out of some sort of models or collection of models and you do not have easy access to the equations.

If the dynamics are given by an ODE and the optimal control is smooth, then control parameterization is often successful. However, a number of issues can present themselves. Problems where highly nonuniform grids are needed, there are discontinuities or corners in the optimal control, or state constraints all present challenges. If the dynamics are described by a DAE, then a DAE integrator must be used and a control parameterization with appropriate smoothness may be required. A nice survey of several options for integrating DAEs is found in the recent study [31].

5. More challenging versions of the problem. To illustrate some of the additional challenges that may be faced in other optimal control problems we consider three variants of our accompanying test problem.

Since in applications not all values for the control can be realized we often have bounds on the control in the form

$$|u(t)| \leq \rho$$

for a positive constant $\rho$.

Another generalization would be that we allow for a more complex motion of the target described by a more general cost

$$J(u) = \int_0^T c u^2 + d(x - L \sin(\alpha(t)))^2 + d(y - L \cos(\alpha(t)))^2 dt$$

where $\alpha$ is a given smooth real valued function. Note that this allows for the target to speed up, slow down, or even reverse direction.

Of course these more general problems can still be reformulated as before and then become a more conventional problem.

If there are control bounds like in (49), then direct transcription is not immediately effected since this is just additional NLP constraints. However, we expect that the control may have reduced smoothness in some places. For the solution of the necessary conditions there is a more major change. The condition that the derivative of the Hamiltonian with respect to $u$ is zero is now that $u$ minimizes the Hamiltonian. Thus instead of (25k) we have that $u$ minimizes an expression where it appears with a coefficient of $\lambda_2 (\cos(t)x_3 - \sin(t)x_1)$ if we are using formulation (24).

Observe also that in such more general cases when we may not have a good idea of where the solution is, it may not be possible to choose phases a priori.
6. Conclusions. We have discussed the numerical solution of DAE optimal control problems using an index three trajectory control problem as an illustration. We have considered reparametrization of the solution manifold and the use of phases which are both ODEs and DAEs. Problems have been treated by direct transcription, by solving the necessary conditions, and by control parameterization. Several observations have been made. One is that having software that can solve fully implicit DAE boundary value problems can be very useful when working with the necessary conditions for reduced index nonlinear problems because the index reduction requires time varying or nonlinear state dependent matrices.

We have also shown that the use of phases can be very helpful but success can be highly dependent on good initial guesses and in how the path constraints are utilized. In our example, to illustrate, we could solve the algebraic equations from the derivative array analytically. However, for more complex examples that would not be an option.

In some ways for this particular problem solving the necessary conditions was the most robust approach.

Many optimal control problems do have control constraints. This can greatly complicate the use of necessary conditions particularly if bang bang controls are present. With direct transcription it is easy to include a variety of inequality control constraints.

The discussion in this paper illustrates that while there are a wide variety of ways to formulate a DAE optimal control problem, in general with higher index dynamics something additional has to be done.

REFERENCES

[1] R. Altmann and J. Heiland, Simulation of multibody systems with servo constraints through optimal control, Oberwolfach Preprint OWP 2015-18, 2015.
[2] J. Baumgarte, Stabilization of constraints and integrals of motion in dynamical systems, Comp. Meth. Appl. Mech. Eng., 1 (1972), 1–16.
[3] J. T. Betts, Practical Methods for Optimal Control and Estimation using Nonlinear Programming, 2nd ed., Philadelphia, SIAM, 2010.
[4] J. T. Betts and S. O. Erb, Optimal low thrust trajectories to the moon, SIAM J. Appl. Dyn. Syst., 2 (2003), 144–170.
[5] J. T. Betts, S. L. Campbell and A. Engelsone, Direct transcription solution of optimal control problems with higher order state constraints: theory vs practice, Optim. Eng., 8 (2007), 1–19.
[6] H. G. Bock, M. M. Diehl, D. B. Leineweber and J. P. Schöder, A direct multiple shooting method for real-time optimization of nonlinear DAE processes, Nonlinear model predictive control (Ascona, 1998), Progr. Systems Control Theory, 26, Birkhäuser, Basel, (2000), 245–267.
[7] K. E. Brenan, S. L. Campbell and L. R. Petzold, Numerical Solution of Initial Value Problems in Differential-Algebraic Equations, SIAM, Philadelphia, PA, 1996.
[8] C. Büskens and H. Maurer, SQP-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real time control, J. Comp. Appl. Math., 120 (2000), 85–108.
[9] S. L. Campbell and J. T. Betts, Comments on direct transcription solution of constrained optimal control problems with two discretization approaches, Numerical Algorithms, 73 (2016), 807–838.
[10] S. L. Campbell and R. März, Direct transcription solution of high index optimal control problems and regular Euler-Lagrange equations, J. Comp. Appl. Math., 202 (2007), 186–202.
[11] S. L. Campbell, P. Kunkel and V. Mehrmann, Regularization of linear and nonlinear descriptor systems, in Control and Optimization with Differential-Algebraic Constraints (eds. L. T. Biegler, S. L. Campbell and V. Mehrmann) SIAM, Philadelphia, (2012), 17–34.
[12] C. L. Darby, W. W. Hager and A. V. Rao, An hp-adaptive pseudospectral method for solving optimal control problems, Optimal Control Applications and Methods, 32 (2011), 476–502.
[13] A. L. Dontchev and W. W. Hager, A new approach to Lipschitz continuity in state constrained optimal control, *Syst. Control Lett.*, 35 (1998), 137–143.

[14] A. L. Dontchev and W. W. Hager, Lipschitzian stability for state constrained nonlinear optimal control, *SIAM J. Control Optim.*, 36 (1998), 698–718.

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

[16] A. Engelsone, S. L. Campbell, and J. T. Betts, Order of convergence in the direct transcription solution of optimal control problems, *Proc. IEEE Conf. Decision Control - European Control Conference*, Seville, Spain, 2005.

[17] W. F. Feehery, J. R. Banga and P. I. Barton, A novel approach to dynamic optimization of ODE and DAE systems as high-index problems, *AIChE annual meeting*, Miami, 1995.

[18] W. F. Feehery and P. I. Barton, Dynamic simulation and optimization with inequality path constraints, *Comp. Chem. Eng.*, 20 (1996), S707–S712.

[19] F. Ghanbari and F. Goreish, Convergence analysis of the pseudospectral method for linear DAEs of index-2, *Int. J. Comp. Methods*, 10 (2013), 1350019-1–1350019-20.

[20] W. W. Hager. Runge-Kutta methods in optimal control and the transformed adjoint system, *Numer. Math.*, 87 (2000), 247–280.

[21] D. H. Jacobson, M. M. Lele and J. L. Speyer, New necessary conditions of optimality for control problems with state variable inequality constraints, *J. Math. Anal. Appl.*, 35 (1971), 255–284.

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

[23] P. Kunkel and V. Mehrmann, Optimal control for unstructured nonlinear differential-algebraic equations of arbitrary index, *Math. Control Signal*, 20 (2008), 227–269.

[24] P. Kunkel and V. Mehrmann, Formal adjoints of linear DAE operators and their role in optimal control, *Electron. J. Linear Algebra*, 22 (2011), 672–693.

[25] P. Kunkel, V. Mehrmann and I. Seufer, GENDA: A software package for the numerical solution of general nonlinear differential-algebraic equations, Institut für Mathematik, TU Berlin Technical Report 730, Berlin, Germany, 2002.

[26] P. Kunkel, V. Mehrmann and R. Stöver, Symmetric collocation for unstructured nonlinear differential-algebraic equations of arbitrary index, *Numer. Math.*, 98 (2004), 277–304.

[27] P. Kunkel and R. Stöver, Symmetric collocation methods for linear differential-algebraic boundary value problems, *Numer. Math.*, 91 (2002), 475–501.

[28] R. Lamour, R. März and E. Weinmüller, Boundary-value problems for differential algebraic equations: a survey, *Surveys in Differential Algebraic Equations III*, Springer, 2015.

[29] R. Pytlak, Runge-Kutta based procedure for the optimal control of differential-algebraic equations, *J. Optim. Theory Appl.*, 97 (1998), 675–705.

[30] A. V. Rao, D. A. Benson, C. Darby, M. A. Patterson, C. Francolin, I. Sanders and G. T. Huntington, Algorithm 902: GPOPS, a MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method, *ACM Trans. Math. Software*, 37 (2010), 22:1–22:39.

[31] A. Steinbrecher, M001 - The simple pendulum, Preprint 2015/26, Technische Universität Berlin, Institut of Mathematik, 2015.

[32] A. Wächter and L. T. Biegler, On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming, *Math. Program.*, 106 (2006), 25–57.

Received October 2016; 1st revision October 2016; final revision November 2016.

E-mail address: slc@ncsu.edu
E-mail address: Peter.Kunkel@math.uni-leipzig.de