A simple success check for delay differential-algebraic equations

Ines Ahrens1,∗ and Benjamin Unger1
1 Institut für Mathematik, Technische Universität Berlin, Str. des 17. Juni 136, 10623 Berlin

Solutions of delay differential-algebraic equations (DDAEs) may depend on derivatives and future evaluations of some of its equations. Structural analysis can be used to determine how often each equation needs to be differentiated and shifted. Unfortunately, these numbers are not always correct, such that a post-processing step is required to validate the result. In this contribution, we present the first step towards a success check for DDAEs.

© 2021 The Authors Proceedings in Applied Mathematics & Mechanics published by Wiley-VCH GmbH

1 Introduction

We consider linear initial trajectory problems (ITPs) for DDAEs of the form

\[ E(t)\dot{x} = A(t)x + B(t)\Delta t x + D(t)\Delta t \dot{x} + f(t) \]  

(1)

on some time interval \([0, T]\) with \(\tau > 0\) consisting of \(n\) unknowns and \(n\) equations. Hereby, \((\Delta t f)(t) := f(t + \sigma)\) denotes the shift (forward) operator. Typically, the DDAE (1) is equipped with an initial trajectory

\[ x(t) = \phi(t) \quad \text{for} \quad t \in [-\tau, 0]. \]  

(2)

It is well-known that a solution of a DDAE at time \(t\) may depend on derivatives and evaluations of (1) at \(t + k\tau\) for some \(k \in \mathbb{N}\). For a numerical method, it is thus essential to understand how often equations need to be differentiated and shifted in time. The required number of differentiations and shifts can be determined using structural analysis [1]. Unfortunately, structural analysis might not detect the correct number of differentiations and shifts and consequently, a post-processing step is required to validate the result. For differential-algebraic equations (DAEs), i.e., equations of the form (1) that do not depend on \(x(t - \tau)\) and \(\dot{x}(t - \tau)\), such a success check is presented in [4], which we generalize in the next section to DDAEs. Let us mention that the results presented below also apply to nonlinear DDAEs.

2 A simple success check for DDAEs

Suppose that numbers \(\sigma_i\) and \(\delta_i\) are available that dictate how often we have to shift and differentiate, respectively, the \(i\)th equation in (1). These numbers might come from the structural analysis presented in [1] or may be obtained otherwise. The goal of the success check is to determine if \(\sigma_i\) and \(\delta_i\) are sufficiently large. Let \(s_i\) denote the largest non-negative number such that \(\Delta s_i x_i^{(\ell)}\) appears in the set of shifted and differentiated variables for some \(\ell \in \mathbb{N}_0\) and let \(d_i\) denote the largest number such that \(\Delta d_i x_i^{(d_i)}\) is present. We call \(s_i\) the highest shift for \(x_i\), and \(d_i\) the highest derivative for \(\Delta s_i x_i\). Moreover, we define the highest derivative of \(x_i\) to be the largest number \(d_i\) such that \(\Delta d_i x_i^{(d_i)}\) appears for some \(\ell \in \mathbb{N}_0\). Note that by construction the highest derivatives satisfy \(d_i \geq d_i\) for \(i = 1, \ldots, n\). With these preparations, we are able to present our simple success check in Algorithm 1.

Note that the relation \((*)\) constitutes a delay equation that might include positive and negative time delays, and thus it is not immediately clear if the associated ITP is solvable.

Example 2.1 Applying Algorithm 1 to the DDAE

\[ \dot{x}_1 = f_1, \quad \Delta t x_2 = x_1 + f_2 \]  

(3)

with \(\delta_1 = \delta_2 = 0\) and \(\sigma_1 = \sigma_2 = 1\) results in the system \(\mathcal{F}\) given by \(\Delta \dot{x}_1 = \Delta t f_1, x_2 = \Delta t x_1 + \Delta t f_2\), which is already in form \((*)\). For reformulation \((***)\) we shift the first equation once backward in time. The second equation still depends on \(\Delta t x_1\) such that Algorithm 1 returns failure. If we instead apply Algorithm 1 with \(\delta_1 = 0, \delta_2 = 1\) and \(\sigma_1 = \sigma_2 = 1\), which corresponds to an additional differentiation of the second equation, then \((***)\) is given by \(\dot{x}_1 = f_1, x_2 = \Delta t f_1 + \Delta t f_2\) and Algorithm 1 returns success.

The assumption that \(\mathcal{F}\) does not depend on \(\Delta k x_i^{(\ell)}\) for any \(k > 0\) is a necessary condition to solve \((***)\) with the method of steps. Nevertheless, \(\mathcal{F}\) in \((***)\) may contain variables \(x_i^{(d)}\) with \(d \geq d_i\). In this case it is still not clear if \((***)\) can be used to construct a solution. In the following, we show that the additional assumption \(d_i = d_i\) in line 5 of Algorithm 1 is sufficient to show that the associated DAE for \((***)\), i.e., the DAE that is obtained by substituting a given function for the delayed variables,
Algorithm 1 A simple success check for DDAEs

1: Differentiate the $j$th equation $\delta_j$ times and shift it $\sigma_j$ times. Collect the resulting system of equations in $\mathcal{F}$. Let $s_i$ denote the highest shift of the variable $x_i$, and $d_i$ the highest derivative of $\Delta_{s_i}x_i$. Define $X_0 := [\Delta_{s_1}x_1^{(d_1)}, \ldots, \Delta_{s_n}x_n^{(d_n)}]^T$ and collect the remaining variables in $\mathcal{X}_{<0}$.

2: if the Jacobian $\frac{\partial F}{\partial X_0}$ is nonsingular at a consistency point then
3: The implicit function theorem yields the explicit relation
4: \[ X_0 = \varphi(t, \mathcal{X}_{<0}). \] (*)
5: Shift the $i$th equation in ($\ast$) $s_i$ times backwards in time to obtain
6: \[ \tilde{X}_0 := \begin{bmatrix} \tilde{x}_1^{(d_1)} \\ \vdots \\ \tilde{x}_n^{(d_n)} \end{bmatrix} = \begin{bmatrix} \Delta_{-s_1}\varphi_1(t, \mathcal{X}_{<0}) \\ \vdots \\ \Delta_{-s_n}\varphi_n(t, \mathcal{X}_{<0}) \end{bmatrix} =: \psi(t, \tilde{X}). \] (**) 
7: if $\tilde{X}$ does not depend on positive delay and $\tilde{d}_i = d_i$ for $i = 1, \ldots, n$ then return success end if 
8: end if

is equivalent to a regular DAE with index at most 1. We thus assume that Algorithm 1 applied to the DDAE (1) returns success. The assumption $\tilde{d}_i = d_i$ implies that if $x_i^{(d_i)}$ appears in $\mathcal{X}$, then $d \leq \tilde{d}_i$. To obtain a strict inequality, we can proceed as follows. Suppose that the shifts are ordered increasingly, i.e., $0 \leq s_1 \leq s_2 \leq \cdots \leq s_n$. Then it is easy to see that the Jacobian $\frac{\partial \varphi}{\partial X_0}$ is strictly upper triangular and we conclude that $\frac{\partial \varphi}{\partial X_0} = I_n - \frac{\partial \varphi}{\partial X_0}$ is nonsingular.

The implicit function theorem applied to (**) thus yields

\[ X_0 = \tilde{\psi}(t, \tilde{X}_{<0}). \] (4)

where the right-hand side is independent of $\tilde{X}_0$. To obtain a first-order formulation, we apply a trimmed linearization [3] to (4) and reorder the equations to obtain

\[ \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix} \dot{z} = \begin{bmatrix} J(t) & 0 \\ Q(t) & I \end{bmatrix} z + \eta(t, \Delta_{-\tau}z, \ldots, \Delta_{-(s_n-1)\tau}z, \Delta_{-\tau}\dot{z}, \ldots, \Delta_{-(s_n-1)\tau}\dot{z}). \] (5)

Clearly, the corresponding DAE has differentiation index at most 1. Using the shift index concept from [2] we thus conclude that the DDAE (5) has shift index 0. Summarizing our discussion yields our main result.

**Theorem 2.2** If Algorithm 1 returns success, then the numbers $\delta_i$ and $\sigma_j$ are sufficiently large, in the sense that (5) has differentiation index less or equal 1 and shift index 0.

If in addition the coefficient matrices in (1) are independent of time, then [6, Theorem 4] immediately implies that the initial trajectory problem associated with (5) has a unique solution within the space of piecewise smooth distributions [5]. If a continuous solution is required, then the initial trajectory has to satisfy some additional splicing conditions, see for instance [7].

**Acknowledgements** The work of I. Ahrens is supported by the German Research Foundation (DFG), project number 384950143. The work of B. Unger is supported by the DFG Collaborative Research Center 910, project number 163436311. Open access funding enabled and organized by Projekt DEAL.

References

[1] I. Ahrens and B. Unger. The Pantelides algorithm for delay differential-algebraic equations. *Trans. Math. Appl.*, 4:1–36, 2020.
[2] P. Ha and V. Mehrmann. Analysis and numerical solution of linear delay differential-algebraic equations. *BIT Numer. Math.*, 56(2):633–657, 2016.
[3] V. Mehrmann and C. Shi. Transformation of high order linear differential-algebraic systems to first order. *Numer Algor.*, 42(3):281–307, 2006.
[4] J. D. Pryce. A simple structural analysis method for DAEs. *BIT Numer. Math.*, 41(2):364–394, 2001.
[5] S. Trenn. *Distributional differential algebraic equations*. Dissertation, Technische Universität Ilmenau, 2009.
[6] S. Trenn and B. Unger. Delay regularity of differential-algebraic equations. In Proc. 58th IEEE Conf. Decision Control (CDC) 2019, *Nice, France*, pages 989–994, 2019.
[7] B. Unger. Discontinuity propagation in delay differential-algebraic equations. *Electron. J. Linear Algebr.*, 34:582–601, 2018.