A NEW SOLVER FOR LINEAR DELAY DIFFERENTIAL-ALGEBRAIC EQUATIONS*

PHI HA† AND VINH THO MA† AND VOLKER MEHRMANN†

Version 0.3 December 22, 2014

Abstract. We discuss the new solver COLDDAE for the numerical solution of initial value problems for linear differential-algebraic equations (DDAEs) with variable coefficients. The implementation is mainly based on the results introduced in [13]. The solver can deal with general DDAEs, which can be either causal or noncausal. The hidden advancedness of the system is detected by the solver, and furthermore, the solver can deal with not only retarded and neutral systems but also a certain class of advanced DDAEs.

Key words. Delay differential-algebraic equation, differential-algebraic equation, delay differential equations, method of steps, derivative array, classification of DDAEs.

AMS subject classifications. 34A09, 34A12, 65L05, 65H10.

1. Introduction. We discuss the new solver COLDDAE for the numerical solution of linear delay differential-algebraic equations (DDAEs) with variable coefficients of the following form

$$E(t)\dot{x}(t) = A(t)x(t) + B(t)x(t - \vec{\tau}(t)) + f(t), \quad t \in \mathbb{I} := [t_0, t_f], \tag{1.1a}$$

where E, $A \in C(\mathbb{I}, \mathbb{R}^{m\times n})$, $B = \begin{bmatrix} B_1 & \dots & B_k \end{bmatrix} \in C(\mathbb{I}, \mathbb{R}^{m\times kn})$, $f \in C(\mathbb{I}, \mathbb{R}^m)$, $\vec{\tau}(t) = \begin{bmatrix} \tau_1(t) & \dots & \tau_k(t) \end{bmatrix}$, where the delay functions τ_i, $i = 1, \dots, k$ are continuous and satisfy $t > \tau_i(t) > 0$ for all $t \in [t_0, t_f]$. For later use in the index reduction procedure, we further assume that the functions τ_i, $i = 1, \dots, k$, are sufficiently smooth. By $x(t - \vec{\tau}(t))$ we denote the vector $\begin{bmatrix} x^T(t - \tau_1(t)) & \dots & x^T(t - \tau_k(t)) \end{bmatrix}^T$.

Set $\underline{\tau} := \min\{\tau_i(t) |\ t \in [t_0, t_f],\ i = 1, \dots, k\}$ and $\bar{\tau} := \max\{\tau_i(t) |\ t \in [t_0, t_f],\ i = 1, \dots, k\}$, let us assume that $\underline{\tau} > 0$, which is often referred in the literature [2, 5] as the *non vanishing delay* case. We further assume that $\bar{\tau} < \infty$. Typically, to form an IVP for the DDAE (1.1a), one needs to add a history function (an initial function)

$$x|_{[t_0-\bar{\tau},t_0]} = \phi \in C([t_0 - \bar{\tau}, t_0], \mathbb{R}^n). \tag{1.1b}$$

We assume that the IVP (1.1) has a unique solution $x \in C(\mathbb{I}, \mathbb{R}^n)$. For systems with constant delays the theoretical analysis for IVPs of the form (1.1) has been discussed in [12–14]. These results, however, are possible to directly generalized for the case of time dependent delays under some minor assumptions on the delay functions. Therefore, the solver that we discuss in this paper is intended for time dependent delays. We will recall only the necessary parts of these works to make the procedure of computing the solution of (1.1) transparent. The most important concepts presented in these works are the causality, the system type, the shift index and the strangeness index of the DDAE (1.1a).

Under the assumption that the continuous, piecewise differentiable solution of the IVP (1.1) exists and is unique, the implementation of the new solver is based

*This work was supported by DFG Collaborative Research Centre 910, *Control of self-organizing nonlinear systems: Theoretical methods and concepts of application*

†Institut für Mathematik, MA 4-5, TU Berlin, Straße des 17. Juni 136, D-10623 Berlin, Germany; {ha,mavinh,mehrmann}@math.tu-berlin.de

on the construction of the regularization procedure introduced in [13], which first determines the shift index, the strangeness index and then transforms the DDAE (1.1a) into a regular, strangeness-free formulation with the same solution set. Using this regularization procedure, we can also compute a consistent initial function and can apply the well-known integration schemes for the resulting regular, strangeness-free DDAEs. Having seen the advantages of collocation Runge-Kutta methods for DDAEs, see [10, 11], in our solver we have implemented collocation Runge-Kutta (RK) schemes.

2. A brief survey of the basic results. The numerical solution of IVPs for DDAEs, until now, has only been considered for square systems, see e.g. [1, 2, 8, 11, 21, 24–27]. For such systems, the solution is usually computed by the classical (Bellman) *method of steps* [4–6], which will be recalled below.
Since $\underline{\tau} > 0$, we have $[t_0, t_f] \subset \bigcup_{j=1,\ldots,\ell+1} [t_0+(j-1)\underline{\tau}, t_0+j\underline{\tau}]$ with $\ell := \lfloor \frac{t_f - t_0}{\underline{\tau}} \rfloor$. For all $t \in [t_0, t_0+\underline{\tau}]$, we have $t - \tau_i(t) \leqslant t_0 + \underline{\tau} - \underline{\tau} = t_0$, and hence $x(t-\tau_i(t)) = \phi(t-\tau_i(t))$. The DDAE (1.1a) restricted on the interval $[t_0, t_0+\underline{\tau}]$ then becomes

$$E(t)\dot{x}(t) = A(t)x(t) + B(t)\phi(t - \vec{\tau}(t)) + f(t), \tag{2.1}$$

where $\phi(t-\vec{\tau}(t)) := \begin{bmatrix} \phi^T(t-\tau_1(t)) & \ldots & \phi^T(t-\tau_k(t)) \end{bmatrix}^T$. This system is the DAE in the variable $x_1 := x|_{[t_0, t_0+\underline{\tau}]}$. The initial vector of the corresponding IVP for the DAE (2.1) is $x(t_0) = \phi(0)$. Suppose that this IVP has a unique solution x_1, we can proceed in the same way to compute the function $x_2 := x|_{[t_0+\underline{\tau}, t_0+2\underline{\tau}]}$, since $t - \tau_i(t) \leqslant t_0 + 2\underline{\tau} - \underline{\tau} = t_0 + \underline{\tau}$, for all $t \in [t_0+\underline{\tau}, t_0+2\underline{\tau}]$. Therefore, the solution x of the IVP (1.1) will be computed step by step on consecutive intervals $[t_0+(j-1)\underline{\tau}, t_0+j\underline{\tau}]$, $1 \leqslant j \leqslant \ell$. On the interval $[t_0+(j-1)\underline{\tau}, t_0+j\underline{\tau}]$, the function $x_j := x|_{[t_0+(j-1)\underline{\tau}, t_0+j\underline{\tau}]}$ is computed from the DAE of the form

$$E(t)\dot{x}_j(t) = A(t)x_j(t) + g_j(t) \quad \text{for all } t \in [t_0+(j-1)\underline{\tau}, t_0+j\underline{\tau}]. \tag{2.2}$$

Clearly, we see that the method of steps successfully handles the IVP (1.1) if and only if for every j, the corresponding IVP for the DAE (2.2) has a unique solution. This requirement means that the solution x at a current point t depends only on the system (1.1a) at current and past time points (i.e., $s \leqslant t$), but not future time points ($s > t$). We call this property *causality*, and a DDAE that satisfies this property *causal*. Restricted to the class of causal systems, different integration strategies based on the method of steps have been successfully implemented for linear DDAEs of the form (1.1a) and also for several classes of nonlinear DDAEs, see e.g. [1, 2, 11, 15, 24]. In contrast to causal DDAEs, this approach is not feasible for noncausal systems, consider for example equation

$$0 \cdot \dot{x}(t) = 0 \cdot x(t) - x(t-\tau) + f(t), \quad \text{for all } t \in (0, \infty). \tag{2.3}$$

The method of steps applied to the DDAE (2.3) results in a sequence of undetermined DAEs of the form

$$0 = g_i(t), \quad \text{for all } t \in [t_0+(j-1)\underline{\tau}, t_0+j\underline{\tau}].$$

Nevertheless, the IVP (1.1) still has a unique solution. The reason for this failure is that the method of steps takes into account only the equation at the current time, which is not enough, due to the noncausality of general DDAEs. Therefore, a regularization procedure for DDAEs, so that the method of steps can be used for the resulting systems, is necessary. Note that for noncausal DDAEs of the form (1.1a), the solvability analysis has only been discussed for the single delay case, i.e., $k = 1$. Even for multiple constant delays, i.e., $\tau_i(t) \equiv \tau_i$, the solvability analysis for noncausal DDAEs is not entirely understood, [12, 13]. The regularization procedure proposed in the package COLDDAE for causal (resp. noncausal) systems will be considered in Subsection 2.1 (resp. 2.2) below.

2.1. Regularization procedure for causal DDAEs with multiple delays. Inherited from the theory of DAEs, we see that even for causal DDAEs, the numerical integration requires a reformulation of the original system in such a way that one can avoid the loss in order of convergence or the drift-off effect, see e.g. [7, 18]. Here we use the regularization procedure associated with the *strangeness index* concept, see [18], which generalizes the well-known *differentiation index* [7] for general under- and over-determined DAEs. Briefly speaking, the strangeness index μ of the DAE

$$E(t)\dot{x}(t) = A(t)x(t) + f(t), \tag{2.4}$$

is the minimum number of differentiation such that from the *derivative arrays* (or *differentiation-inflated* system)

$$\begin{aligned} E(t)\dot{x}(t) - A(t)x(t) &= f(t), \\ \frac{\mathrm{d}}{\mathrm{d}t}\left(E(t)\dot{x}(t) - A(t)x(t)\right) &= f^{(1)}(t), \\ &\cdots \\ \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^{\mu}\left(E(t)\dot{x}(t) - A(t)x(t)\right) &= f^{(\mu)}(t), \end{aligned}$$

one can extract the so-called *strangeness-free formulation*

$$\begin{bmatrix} \hat{E}_1(t) \\ 0 \\ 0 \end{bmatrix} \dot{x}(t) = \begin{bmatrix} \hat{A}_1(t) \\ \hat{A}_2(t) \\ 0 \end{bmatrix} x(t) + \begin{bmatrix} \hat{f}_1(t) \\ \hat{f}_2(t) \\ \hat{f}_3(t) \end{bmatrix}, \tag{2.5}$$

which has the same solution set as the DAE (2.4), where the matrix-valued function $\begin{bmatrix} \hat{E}_1^T & \hat{A}_2^T \end{bmatrix}^T$ has pointwise full row rank. We also call μ the strangeness index of the pair (E, A). For the numerical determination of the strangeness index and the strangeness-free formulation (2.5), we refer the readers to [19, 20].

Now we apply the strangeness-free formulation to the DDAE (1.1a), which is assumed to be causal, to obtain the strangeness-free DDAE

$$\begin{bmatrix} \hat{E}_1(t) \\ 0 \\ 0 \end{bmatrix} \dot{x}(t) = \begin{bmatrix} \hat{A}_1(t) \\ \hat{A}_2(t) \\ 0 \end{bmatrix} x(t) + \begin{bmatrix} \hat{B}_{0,1}(t) \\ \hat{B}_{0,2}(t) \\ 0 \end{bmatrix} x(t-\vec{\tau}(t)) + \sum_{i=1}^{\mu} \begin{bmatrix} 0 \\ \hat{B}_{i,2}(t) \\ 0 \end{bmatrix} x^{(i)}(t-\vec{\tau}(t)) + \begin{bmatrix} \hat{f}_1(t) \\ \hat{f}_2(t) \\ \hat{f}_3(t) \end{bmatrix}, \quad \begin{matrix} d \\ a \\ v \end{matrix} \tag{2.6}$$

where $\mu = \mu(E, A)$ is the strangeness index of the pair (E, A) and the function $\begin{bmatrix} \hat{E}_1^T & \hat{A}_2^T \end{bmatrix}^T$ has pointwise full row rank. Sizes of the block row equations are d, a, v. Due to the causality of the DDAE (1.1a), it follows that $\begin{bmatrix} \hat{E}_1^T & \hat{A}_2^T \end{bmatrix}$ is pointwise nonsingular. For the DAE (2.4), the numerical solution $x(t)$ is obtained by integrating the strangeness-free formulation (2.5), which is more advantageous, see [16–18]. For the DDAE (1.1a), integrating the strangeness-free DDAE (2.6) is not always possible. The reason is that if at least one of the matrix functions $\hat{B}_{i,2}$, $i = 1, \ldots, \mu$, is not identically zero, then the underlying DDE is of advanced type, which is not suitable for the numerical integration, [3]. In fact, until now there is no solver for advanced DDEs. For the numerical solution, solvers based on the method of steps are only suitable for retarded and neutral DDAEs, see [1, 11, 13, 15]. In these cases, the strangeness-free DDAE (2.6) takes the form

$$\begin{bmatrix} \hat{E}_1(t) \\ 0 \\ 0 \end{bmatrix} \dot{x}(t) = \begin{bmatrix} \hat{A}_1(t) \\ \hat{A}_2(t) \\ 0 \end{bmatrix} x(t) + \begin{bmatrix} \hat{B}_{0,1}(t) \\ \hat{B}_{0,2}(t) \\ 0 \end{bmatrix} x(t-\vec{\tau}(t)) + \begin{bmatrix} \hat{f}_1(t) \\ \hat{f}_2(t) \\ \hat{f}_3(t) \end{bmatrix}, \quad \begin{matrix} d \\ a. \\ v \end{matrix} \tag{2.7}$$

The integration strategy we use for causal, retarded/neutral DDAEs of the form (1.1a) is: first, determine the strangeness-free formulation (2.7), and second, apply numerical methods to (2.7) to compute $x(t)$.

For the determination of the strangeness-free DDAE (2.7), we use the derivative arrays for DDAEs as follows

$$\begin{aligned} E(t)\dot{x}(t) - A(t)x(t) &= B(t)x(t-\vec{\tau}(t)) + f(t), \\ \frac{\mathrm{d}}{\mathrm{d}t}\left(E(t)\dot{x}(t) - A(t)x(t)\right) &= \frac{\mathrm{d}}{\mathrm{d}t}\left(B(t)x(t-\vec{\tau}(t)) + f(t)\right), \\ &\cdots \\ \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^{\mu}\left(E(t)\dot{x}(t) - A(t)x(t)\right) &= \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^{\mu}\left(B(t)x(t-\vec{\tau}(t)) + f(t)\right), \end{aligned}$$

which can be rewritten as

$$Mz(t) = Pz(t-\vec{\tau}(t)) + g, \tag{2.8}$$

where

$$M := \begin{bmatrix} -A(t) & E(t) & & & & \\ -\dot{A}(t) & \dot{E}(t) - A(t) & E(t) & & & \\ -\ddot{A}(t) & \ddot{E}(t) - 2\dot{A}(t) & 2\dot{E}(t) - A(t) & E(t) & & \\ & \vdots & & \ddots & \ddots & \\ -A^{(\mu)}(t) & E^{(\mu)}(t) - \mu A^{(\mu-1)}(t) & \dots & \dots & \mu\dot{E}(t) - A(t) & E(t) \end{bmatrix},$$

$$P := \begin{bmatrix} B(t) & & & 0 \\ \dot{B}(t) & B(t)(1-\dot{\vec{\tau}}(t)) & & 0 \\ \ddot{B}(t) & 2\dot{B}(t)(1-\dot{\vec{\tau}}(t)) - B(t)\ddot{\vec{\tau}}(t) & B(t)(1-\dot{\vec{\tau}}(t))^2 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ * & * & * & 0 \end{bmatrix}$$

$$z(t) := \begin{bmatrix} x(t) \\ \dot{x}(t) \\ \vdots \\ x^{(\mu+1)}(t) \end{bmatrix}, \quad g := \begin{bmatrix} f(t) \\ \dot{f}(t) \\ \vdots \\ f^{(\mu+1)}(t) \end{bmatrix}.$$

Notice that the formulas of the other entries of P can be obtained by compute the coefficients of the high derivatives $\left(B(t)x(t-\vec{\tau}(t))\right)^{(i)}$, $i = 2, \dots, m$. Infact, these coefficients is computed by using the Faà di Bruno's formula [9], which describes the

m derivative of the composite function $g(f(x))$ in the variable x as follows

$$\begin{aligned}&\left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^m [g(f(x))]\\ &= \sum \frac{m!}{b_1!\dots b_m!} g^{(k)}(f(x)) \left(\frac{f^{(1)}(x)}{1!}\right)^{b_1} \left(\frac{f^{(2)}(x)}{2!}\right)^{b_2} \cdots \left(\frac{f^{(k)}(x)}{k!}\right)^{b_k},\end{aligned} \tag{2.9}$$

where the sum is over all possible combinations of nonnegative integers $b_1, \dots, b_k$ such that $b_1 + 2b_2 + \cdots + mb_m = m$ and $k = b_1 + b_2 + \cdots + b_m$. Note that, in our solver we assume only that $\mu \leqslant 3$, and there the explicit formula of P is provided.

In the following, for notational convenience, we will use Matlab notation, [22]. The set of algebraic constraints in the strangeness-free DDAE (2.7) is selected by finding the full row rank matrix Z_2 such that

$$Z_2^T M(:, (n+1) : end) = 0. \tag{2.10}$$

Scaling the system (2.8) with Z_2^T from the left, we obtain the equation

$$Z_2^T M(:, 1 : n)x(t) = Z_2^T P z(t - \vec{\tau}(t)) + Z_2^T g. \tag{2.11}$$

Furthermore, the DDAE (1.1a) is not of advanced type if and only if in (2.11) the derivatives of $x(t - \vec{\tau}(t))$ do not occur. This means that

$$Z_2^T P(:, (kn+1) : end) = 0. \tag{2.12}$$

We consider the following spaces and matrices

$$\begin{array}{lll} T_2 & \text{basis of} & \ker(Z_2^T E), \\ Z_1 & \text{basis of} & \operatorname{range}(E T_2), \\ Y_2 & \text{basis of} & \operatorname{range}(Z_2^T M(:, 1 : n)). \end{array} \tag{2.13}$$

The set of differential equations in the strangeness-free DDAE (2.7), therefore, is

$$Z_1^T E(t)\dot{x}(t) = Z_1^T A(t)x(t) + Z_1^T B(t)x(t - \vec{\tau}(t)) + Z_1^T f(t).$$

In summary, we obtain the strangeness-free DDAE

$$\begin{aligned} Z_1^T E(t)\dot{x}(t) &= Z_1^T A(t)x(t) + Z_1^T B(t)x(t - \vec{\tau}(t)) + Z_1^T f(t), \\ Y_2^T Z_2^T M(:, 1 : n)x(t) &= Y_2^T Z_2^T P(:, 1 : kn)x(t - \vec{\tau}(t)) + Y_2^T Z_2^T g, \end{aligned} \tag{2.14}$$

where $\begin{bmatrix} Z_1^T E(t) \\ Y_2^T Z_2^T M(:, 1 : n) \end{bmatrix}$ is nonsingular.

2.2. Regularization procedure for noncausal DDAEs with single delay. In order to handle noncausal DDAEs with single delay, in [13], the authors proposed the concept of *shift index* as follows. For notational convenience and to be consistent to [13], we will write τ instead of $\vec{\tau}$.

DEFINITION 2.1. *Consider the IVP* (1.1). *For each $t \in \mathbb{I}$, consider the sequence $\{t_{(j)} | j \geq 0\}$, starting with $t_{(0)} = t$, determined via the equation*

$$t_{(j+1)} - \tau(t_{(j+1)}) = t_{(j)}, \quad \textit{for all } j \geq 0. \tag{2.15}$$

Then, the minimum number $\kappa = \kappa(t)$ such that the so-called shift-inflated *system*

$$\begin{aligned} E(t_{(0)})\dot{x}(t_{(0)}) &= A(t_{(0)})x(t_{(0)}) + B(t_{(0)})x(t_{(0)} - \tau(t_{(0)})), \\ E(t_{(1)})\dot{x}(t_{(1)}) &= A(t_{(1)})x(t_{(1)}) + B(t_{(1)})x(t_{(1)} - \tau(t_{(1)})), \\ &\vdots \\ E(t_{(\kappa)})\dot{x}(t_{(\kappa)}) &= A(t_{(\kappa)})x(t_{(\kappa)}) + B(t_{(\kappa)})x(t_{(\kappa)} - \tau(t_{(\kappa)})), \end{aligned} \tag{2.16}$$

uniquely determines $x(t_{(0)})$, is called the shift index *of the DDAE* (1.1a) *with respect to t.*

To guarantee the existence and uniqueness of the sequence $\{t_{(j)} | j \geq 0\}$ in Definition 2.1, we assume that for every $s \in (t_0, t_f)$, the equation

$$t - \tau(t) = s, \tag{2.17}$$

has a unique solution on the time interval (s, t_f).

Remark 2.2. *If the DDAE* (1.1a) *is causal then the shift index κ is* 0 *for all $t \in [t_0, t_f]$.*

Using (2.15), we can rewrite the shift-inflated system (2.16) as

$$\begin{bmatrix} E(t_{(0)}) & & & \\ & E(t_{(1)}) & & \\ & & \ddots & \\ & & & E(t_{(\kappa)}) \end{bmatrix} \begin{bmatrix} \dot{x}(t_{(0)}) \\ \dot{x}(t_{(1)}) \\ \vdots \\ \dot{x}(t_{(\kappa)}) \end{bmatrix}$$

$$= \begin{bmatrix} A(t_{(0)}) & & & \\ B(t_{(1)}) & A(t_{(1)}) & & \\ & \ddots & \ddots & \\ & & B(t_{(\kappa)}) & A(t_{(\kappa)}) \end{bmatrix} \begin{bmatrix} x(t_{(0)}) \\ x(t_{(1)}) \\ \vdots \\ x(t_{(\kappa)}) \end{bmatrix} + \begin{bmatrix} B(t_{(0)})x(t_{(0)} - \tau(t_{(0)})) + f(t_{(0)}) \\ f(t_{(1)}) \\ \vdots \\ f(t_{(\kappa)}) \end{bmatrix}. \tag{2.18}$$

As shown in [13], assuming that the DDAE (1.1a) is not of advanced type, we can extract from the DAE (2.18) a strangeness-free DDAE

$$\begin{bmatrix} \hat{E}_1(t_{(0)}) \\ 0 \end{bmatrix} \dot{x}(t_{(0)}) = \begin{bmatrix} \hat{A}_1(t_{(0)}) \\ \hat{A}_2(t_{(0)}) \end{bmatrix} x(t_{(0)}) + \begin{bmatrix} \hat{B}_1(t_{(0)}) \\ \hat{B}_2(t_{(0)}) \end{bmatrix} x(t_{(0)} - \tau(t_{(0)})) + \begin{bmatrix} \hat{f}_1(t_{(0)}) \\ \hat{f}_2(t_{(0)}) \end{bmatrix}, \begin{matrix} d \\ a \end{matrix} \tag{2.19}$$

where $\begin{bmatrix} \hat{E}_1(t_{(0)}) \\ \hat{A}_2(t_{(0)}) \end{bmatrix}$ is nonsingular, i.e., $d + a = n$.

Analogous to the case of causal DDAEs, we build the derivative arrays (2.8) for system (2.18), and extract from it the strangeness-free DDAE (2.19). To keep the brevity of the paper, we will not repeat the process. It has been observed in [13] that the main difference between causal DDAEs and noncausal DDAEs, in the determination of the strangeness-free DDAEs, are:

i) The size of the derivative arrays (2.8) for noncausal DDAEs is bigger than for causal DDAEs.

ii) For noncausal DDAEs the set of differential equations in the strangeness-free formulation (2.7) cannot be selected from the original DDAE (1.1a), but must be selected from the derivative array (2.8).

3. Algorithms in COLDDAE. First, we observe that the resulting system of the regularization procedure for DDAEs is the following regular, strangeness-free DDAE

$$\begin{bmatrix} \hat{E}_1(t) \\ 0 \end{bmatrix} \dot{x}(t) = \begin{bmatrix} \hat{A}_1(t) \\ \hat{A}_2(t) \end{bmatrix} x(t) + \begin{bmatrix} \hat{B}_1(t) \\ \hat{B}_2(t) \end{bmatrix} x(t - \vec{\tau}(t)) + \begin{bmatrix} \hat{\gamma}_1(t) \\ \hat{\gamma}_2(t) \end{bmatrix}, \tag{3.1}$$

where $\vec{\tau}(t)$ is a scalar function in the noncausal case. We now apply numerical methods to (3.1) to determine $x(t)$. For notational convenience, we consider in the the following only the single delay case, i.e., $\vec{\tau}(t) \equiv \tau(t)$. The solver, however, can handle the multiple delay case, too.
Adopted from the solver RADAR5 [11], we use the Radau scheme for the numerical integration, which is given by nodes

$$0 < \delta_1 < \cdots < \delta_s = 1, \quad s \in \mathbb{N}. \tag{3.2}$$

If we know all the discontinuity points of ϕ, $\dot{\phi}, \ldots, \phi^{(s)}$ then we also know all the discontinuity points of x, $\dot{x}, \ldots, x^{(s)}$ on $[t_0, t_f]$, and hence we can include them into the mesh. However, for simplicity, the solver considers only the case that there is at most one discontinuity point at t_0. We denote a mesh by $\pi \,:\, t_0 < t_1 < \cdots < t_N = t_f$. The collocation points therefore are

$$t_{ij} = t_i + h_i \delta_j, \qquad j = 1, \ldots, s, \tag{3.3}$$

where h_i is the stepsize used at the i-th step. For the numerical approximation of the solution, we seek for the piecewise polynomial X_π of degree s, i.e., $\mathrm{X}_{\pi,i} := \mathrm{X}_\pi|_{[t_i, t_{i+1}]}$ are polynomials of degree s, which are determined by the following set of equations

$$\begin{bmatrix} \hat{E}_1(t_{ij}) \\ 0 \end{bmatrix} \dot{\mathrm{X}}_\pi(t_{ij}) = \begin{bmatrix} \hat{A}_1(t_{ij}) \\ \hat{A}_2(t_{ij}) \end{bmatrix} \mathrm{X}_\pi(t_{ij}) + \begin{bmatrix} \hat{B}_1(t_{ij}) \\ \hat{B}_2(t_{ij}) \end{bmatrix} \mathrm{X}_\pi(t_{ij} - \tau(t_{ij})) + \begin{bmatrix} \hat{\gamma}_1(t_{ij}) \\ \hat{\gamma}_2(t_{ij}) \end{bmatrix}, \tag{3.4}$$

for all $i = 1, \ldots, N$, $j = 1, \ldots, s$.
Due to the presence of $\begin{bmatrix} \hat{B}_1(t_{ij}) \\ \hat{B}_2(t_{ij}) \end{bmatrix} \mathrm{X}_\pi(t_{ij} - \tau(t_{ij}))$ in (3.4), we still have to define the past function $\mathrm{X}_\pi(t_{ij} - \tau(t_{ij}))$ which is an approximation to $x(t_{ij} - \tau(t_{ij}))$. For this we choose

$$\mathrm{X}_\pi(t_{ij} - \tau(t_{ij})) \\ = \begin{cases} \phi(t_{ij} - \tau(t_{ij})) & \text{if } t_{ij} - \tau(t_{ij}) \leqslant 0, \\ \mathrm{X}_{\pi,K}(t_{ij} - \tau(t_{ij})) & \text{for some } 1 \leqslant K \leqslant N \text{ that satisfies } t_K < t_{ij} - \tau(t_{ij}) \leqslant t_{K+1}. \end{cases}$$

The continuous output polynomial $\mathrm{X}_{\pi,K}$ at the K-th step is given by Lagrange interpolation of order s, i.e.,

$$\mathrm{X}_{\pi,K}(t_K + \theta h_K) = \sum_{j=0}^{s} \mathcal{L}_j(\theta) \mathrm{X}_{\pi,K}(t_K + \delta_j h_K), \tag{3.5}$$

where $\mathcal{L}_j(\theta)$ is the Lagrange polynomial of degree s satisfying $\mathcal{L}_j(\delta_K) = \delta_{Kj}$ with δ_{Kj} being the Kronecker delta symbol.

Remark 3.1. *As noticed in [10, 11], one can optionally replace the continuous output polynomial* $\mathrm{X}_{\pi,K}$ *in* (3.5) *by another dense output polynomial given by*

$$\mathrm{X}_{\pi,K}(t_K + \theta h_K) = \sum_{j=1}^{s} \mathcal{L}_j(\theta) \mathrm{X}_{\pi,K}(t_K + \delta_j h_K).$$

The use of only s interpolation nodes δ_j, $j = 1, \ldots, s$ instead of $s+1$ nodes δ_j, $j = 0, \ldots, s$ is beneficial in the presence of a jump in the solution at the point t_K, i.e., $\mathrm{X}_{\pi,K}(t_K) \neq \mathrm{X}_{\pi,K-1}(t_K)$.

The existence and uniqueness, and the convergence results for the numerical approximation X_π are stated in the following theorem.

THEOREM 3.2. *Consider the IVP* (1.1) *and assume that it is uniquely solvable and of either retarded or neutral type. For $N \in \mathbb{N}$ and $s \geq 1$, define the mesh π and the collocation points t_{ij}, $j = 1, \ldots, s$ as in* (3.3). *Then the following assertions hold.*

i) For sufficiently small mesh widths $h_0, \ldots, h_{N-1}$ there exists one and only one continuous piecewise polynomial X_π that solves the DAE sequence (3.4) *and it is consistent at all the mesh point t_i.*

ii) The convergence order of the collocation method with schemes δ_j as in (3.2) *is s, i.e.,*

$$\|\mathrm{X}_e(t) - \mathrm{X}_\pi(t)\|_\infty = \sup_{t \in \mathbb{I}} \|\mathrm{X}_e(t) - \mathrm{X}_\pi(t)\| = O(h^s),$$

where X_e is the exact solution $x \in C^{s+1}(\mathbb{I}, \mathbb{C}^n)$ to the IVP (1.1).

Proof. For the proof see Theorem 4, [15] or Theorem 4.2, [11]. □

4. Using COLDDAE. The package COLDDAE contains two solvers for handling specific situations: `colddae_causal` can deal with linear, causal DDAEs with multiple time varying delays, while `colddae_noncausal` can handle both causal and noncausal DDAEs, but it is only applicable for systems with single delay. In the following we will describe the parameters inside these solvers.

4.1. `colddae_causal` - a solver for linear, causal DDAEs with multiple delays. We assume that the strangeness index of (E, A) is not too big, otherwise large errors may occur, since we compute all derivatives by finite differences.

4.1.1. Input parameters.

- `E` The matrix function $E : [t_0, t_f] \to \mathbb{R}^{m,n}$.
- `A` The matrix function $A : [t_0, t_f] \to \mathbb{R}^{m,n}$.
- `B` The matrix function $[B_1, \ldots, B_k] : [t_0, t_f] \to \mathbb{R}^{m,kn}$.
- `tau` The vector function of delays $t \mapsto [\tau_1(t), \ldots, \tau_k(t)]$.
- `phi` The history function ϕ, i.e., $x(t) = \phi(t)$ for $t < t_0$.
- `tspan` The solution interval $[t_0, t_f]$, `tspan(1)`$= t_0$, `tspan(2)`$= t_f$.
- `options` A `struct` containing the optional parameters.

4.1.2. Optional input parameters. Optional parameters can be passed by the input parameter `options` by the command

`options.`*field_name* = *field_value*.

The following fields are applicable in this solver:

- `Iter` The number of time steps, default: `100`.
- `Step` The (constant) step size of the Runge-Kutta method, must be smaller than $\min_{i=1,\ldots,k} \tau_i(t)$ for all $t \in [t_0, t_f]$, default: $\frac{t_f - t_0}{100}$.
- `AbsTol` Absolute tolerance, default: `1e-5`.
- `RelTol` Relative tolerance, default: `1e-5`.
- `StrIdx` Lower bound for the strangeness index, default: `0`.
- `MaxStrIdx` Upper bound for the strangeness index, default: `3`.
- `InitVal` Initial value, not necessarily consistent, default: `phi(tspan(1))`.

- `IsConst` A boolean, `true` if E and A are constant (then the strangeness-free form is computed only once, i.e., the solver needs less computation time), default: `false`.

4.1.3. Output parameters.
- `t` The discretization of `tspan` by `Iter`+1 equidistant points.
- `x` The numerical solution at `t`.
- `info` A struct with information.

4.2. `colddae_noncausal` - a solver for linear, noncausal DDAEs with single delay. Again, we assume that the strangeness index of (E, A) is not too big. The shift index must be less or equal to three (because of hard coding, could be arbitrary in principle). In this solver, we have implemented step size control and so-called long steps, i.e., the step size may become bigger than the delay.

4.2.1. Input parameters.
- `E` The matrix function $E : [t_0, t_f] \to \mathbb{R}^{m,n}$.
- `A` The matrix function $A : [t_0, t_f] \to \mathbb{R}^{m,n}$.
- `B` The matrix function $B : [t_0, t_f] \to \mathbb{R}^{m,n}$.
- `tau` The scalar delay function $t \mapsto \tau(t)$.
- `phi` The history function ϕ, i.e., $x(t) = \phi(t)$ for $t < t_0$.
- `tspan` The solution interval $[t_0, t_f]$, `tspan(1)`$= t_0$, `tspan(2)`$= t_f$.
- `options` A `struct` containing the optional parameters.

4.2.2. Optional input parameters. Optional parameters can be passed by the input parameter `options` by the command

`options.`*field_name* = *field_value*.

The following fields are applicable in this solver:
- `MaxIter` Upper bound for the total number of time steps (excluding rejected time steps), default: `10000`.
- `MaxReject` Upper bound for the number of rejections per time step, default: `100`.
- `MaxCorrect` Upper bound for the number of correction steps when using long steps (step size bigger than the lag), default: `10`.
- `InitStep` The initial step size of the Runge-Kutta method, default: $\frac{t_f - t_0}{100}$.
- `MinStep` A lower bound for the step size, default: `0`.
- `MaxStep` An upper bound for the step size, default: `inf`.
- `AbsTol` Absolute tolerance, default: `1e-5`.
- `RelTol` Relative tolerance, default: `1e-5`.
- `StrIdx` Lower bound for the strangeness index, default: `0`.
- `MaxStrIdx` Upper bound for the strangeness index, default: `3`.
- `Shift` Lower bound for the strangeness index, default: `0`.
- `MaxShift` Upper bound for the strangeness index, default: `3`.
- `InitVal` Initial value, not necessarily consistent, default: $\phi(t_0)$.

4.2.3. Output parameters.
- `t` A discretization of `tspan` with variable step size.
- `x` The numerical solution at `t`.
- `info` A struct with information.

5. Numerical experiments. We use COLDDAE with its default values to solve four different DDAEs. For the first three we use `colddae_causal` and for the last one `colddae_noncausal`. Example 5.1, 5.2 are taken from the DDE test set [23].

EXAMPLE 5.1. *We consider the following DDE with constant delay.*

$$\begin{aligned}
\dot{x}_1(t) &= x_3(t),\\
\dot{x}_2(t) &= x_4(t),\\
\dot{x}_3(t) &= -2mx_2(t) + (1+m^2)(-1)^m x_1(t-\pi),\\
\dot{x}_4(t) &= -2mx_1(t) + (1+m^2)(-1)^m x_2(t-\pi),
\end{aligned} \tag{5.1a}$$

for $t > 0$ *and* $x(t) = \phi(t)$ *for* $t \leqslant 0$ *with*

$$\begin{aligned}
\phi_1(t) &= \sin(t)\cos(mt),\\
\phi_2(t) &= \cos(t)\sin(mt),\\
\phi_3(t) &= \cos(t)\cos(mt) - m\sin(t)\sin(mt),\\
\phi_4(t) &= m\cos(t)\cos(mt) - \sin(t)\sin(mt).
\end{aligned} \tag{5.1b}$$

Analytical solution: $x(t) = \phi(t)$ *for* $t > 0$.
The relative error of the numerical solution of the IVP (5.1) *is presented in Figure 5.1.*

EXAMPLE 5.2. *We consider the following DDAE*

$$\begin{aligned}
\begin{bmatrix}1 & -1\\ 0 & 0\end{bmatrix}\dot{x}(t) &= \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}x(t) + \begin{bmatrix}0 & 0\\ -1 & 0\end{bmatrix}x(t-1), \quad \text{for all } t > 0,\\
\phi(t) &= \begin{bmatrix}1\\ 0\end{bmatrix}, \qquad\qquad\qquad\qquad \text{for all } t \leqslant 0.
\end{aligned} \tag{5.2}$$

Note that the DDAE (5.2) *is reformulated from the neutral DDE* $\dot{x}(t) = x(t) + \dot{x}(t-1)$ *by introducing a new variable to present* $x(t-1)$.

Analytical solution: *The IVP* (5.2) *possesses the unique solution* $x(t) = \begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix}$ *given by*

$$x_1(t) = \begin{cases}
e^t & \text{for } 0 < t \leqslant 1,\\
(t-1)e^{t-1} + e^t & \text{for } 1 < t \leqslant 2,\\
\frac{1}{2}(t^2 - 2t)e^{t-2} + (t-1)e^{t-1} + e^t & \text{for } 2 < t \leqslant 3,\\
\frac{1}{6}(t^3 - 3t^2 - 3t + 9)e^{t-3} + \frac{1}{2}(t^2 - 2t)e^{t-2} + (t-1)e^{t-1} + e^t & \text{for } 3 < t \leqslant 4,
\end{cases}$$

and $x_2(t) = x_1(t-1)$. *The relative error of the numerical solution of the IVP* (5.1) *is also presented in Figure 5.1.*

EXAMPLE 5.3. *We consider the following DDAE with constant coefficients and multiple time-varying delays. This DDAE is causal and it has strangeness index two or differentiation index three.*

$$\begin{aligned}
\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{bmatrix} &= \begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix} + \begin{bmatrix}x_2(t-1) + x_3(\frac{t}{2}-1)\\ 0\\ 0\end{bmatrix} + f(t), \quad \text{for all } t > 0,\\
\phi(t) &= \begin{bmatrix}e^t\\ 1\\ \sin(t)\end{bmatrix}, \qquad\qquad\qquad\qquad \text{for all } t \leqslant 0.
\end{aligned} \tag{5.3}$$

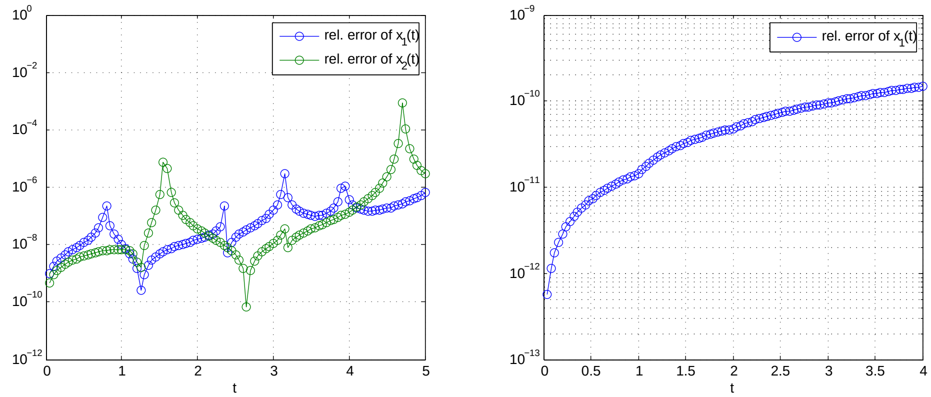

FIG. 5.1. *Relative error of the solution of the IVP* (5.1) *(left) and* (5.2) *(right).*

The function $f(t)$ is chosen such that the analytical solution is $x(t) = \phi(t)$. The relative error of the numerical solution of the IVP (5.3) *is presented in Figure 5.2.*

EXAMPLE 5.4. *We consider the following linear DDAE with constant coefficients and time-varying delay. This DDAE is noncausal and it has strangeness index one and shift index one.*

$$
\begin{aligned}
\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} &= \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1\left(t - 1 + \frac{\sin(t)}{2}\right) \\ x_2\left(t - 1 + \frac{\sin(t)}{2}\right) \end{bmatrix} + f(t), \quad \textit{for all } t > 0, \\
\phi(t) &= \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}, \qquad \textit{for all } t \leqslant 0.
\end{aligned}
\tag{5.4}
$$

The function $f(t)$ is chosen such that the IVP (5.4) *has the unique solution $x(t) = \phi(t)$. The relative error of the numerical solution of the IVP* (5.4) *is also presented in Figure 5.2.*

Acknowledgment. The first and second authors would like to thank Benjamin Unger for useful suggestions and comments.

References.

[1] U. M. Ascher and L. R. Petzold. The numerical solution of delay-differential algebraic equations of retarded and neutral type. *SIAM J. Numer. Anal.*, 32:1635–1657, 1995.

[2] C. T. H. Baker, C. A. H. Paul, and H. Tian. Differential algebraic equations with after-effect. *J. Comput. Appl. Math.*, 140(1-2):63–80, Mar. 2002.

[3] A. Bellen and M. Zennaro. *Numerical Methods for Delay Differential Equations.* Oxford University Press, Oxford, UK, 2003.

[4] R. Bellman. On the computational solution of differential-difference equations. *J. Math. Anal. Appl.*, 2(1):108 – 110, 1961.

[5] R. Bellman and K. L. Cooke. *Differential-difference equations.* Mathematics in Science and Engineering. Elsevier Science, 1963.

[6] R. Bellman and K. L. Cooke. On the computational solution of a class of functional differential equations. *J. Math. Anal. Appl.*, 12(3):495 – 500, 1965.

[7] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value Problems in Differential Algebraic Equations.* SIAM Publications, Philadelphia, PA, 2nd edition, 1996.

[8] S. L. Campbell and V. H. Linh. Stability criteria for differential-algebraic equations with multiple delays and their numerical solutions. *Appl. Math Comput.*, 208(2):397 – 415, 2009.

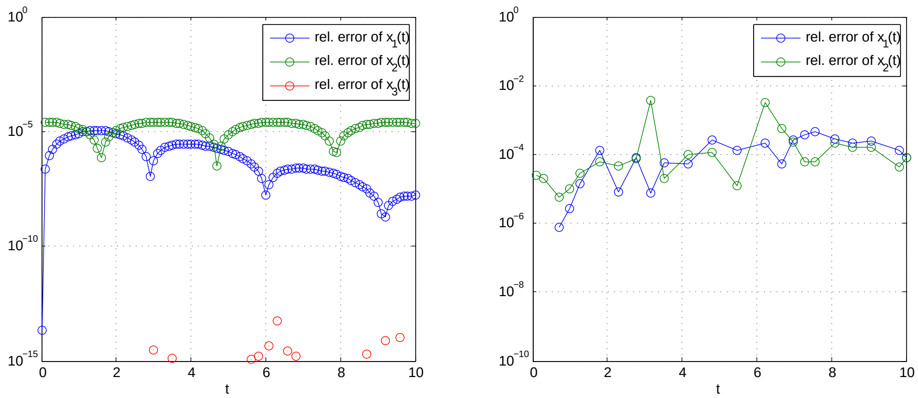

FIG. 5.2. *Relative error of the solution of the IVP* (5.3) *(left) and* (5.4) *(right).*

[9] A. D. D. Craik. Prehistory of Faà di Bruno's formula. *The American Mathematical Monthly*, 112(2):pp. 119–130, 2005.

[10] N. Guglielmi and E. Hairer. Implementing Radau IIA methods for stiff delay differential equations. *Computing*, 67(1):1–12, 2001.

[11] N. Guglielmi and E. Hairer. Computing breaking points in implicit delay differential equations. *Adv. Comput. Math.*, 29:229–247, 2008.

[12] P. Ha. *Analysis and numerical solutions of delay differential-algebraic equations*. Dissertation, Institut für Mathematik, TU Berlin, Berlin, Germany, 2015.

[13] P. Ha and V. Mehrmann. Analysis and numerical solution of linear delay differential-algebraic equations. In prepatation, Institut für Mathematik, TU Berlin, D-10623 Berlin, 2014.

[14] P. Ha, V. Mehrmann, and A. Steinbrecher. Analysis of linear variable coefficient delay differential-algebraic equations. *J. Dynam. Differential Equations*, pages 1–26, 2014.

[15] R. Hauber. Numerical treatment of retarded differential algebraic equations by collocation methods. *Adv. Comput. Math.*, 7:573–592, 1997.

[16] P. Kunkel and V. Mehrmann. Local and global invariants of linear differential-algebraic equations and their relation. *Electr. Trans. Num. Anal.*, 4:138–157, 1996.

[17] P. Kunkel and V. Mehrmann. A new class of discretization methods for the solution of linear differential algebraic equations with variable coefficients. *SIAM J. Numer. Anal.*, 33:1941–1961, 1996.

[18] P. Kunkel and V. Mehrmann. *Differential-Algebraic Equations – Analysis and Numerical Solution*. EMS Publishing House, Zürich, Switzerland, 2006.

[19] P. Kunkel, V. Mehrmann, W. Rath, and J. Weickert. A new software package for linear differential–algebraic equations. *SIAM J. Sci. Comput.*, 18:115–138, 1997.

[20] P. Kunkel, V. Mehrmann, and S. Seidel. A MATLAB package for the numerical solution of general nonlinear differential-algebraic equations. Technical Report 16/2005, Institut für Mathematik, TU Berlin, Berlin, Germany, 2005. `url: http://www.math.tu-berlin.de/preprints/`.

[21] Y. Liu. Runge - Kutta collocation methods for systems of functional differential and functional equations. *Adv. Comput. Math.*, 11(4):315–329, 1999.

[22] The MathWorks, Inc., Natick, MA. *MATLAB Version 8.3.0.532 (R2014a)*, 2014.

[23] C. A. Paul. A test set of functional differential equations. Technical report, Department of Mathematics, University of Manchester, 1994.

[24] L. F. Shampine and P. Gahinet. Delay-differential-algebraic equations in control theory. *Appl. Numer. Math.*, 56(3-4):574–588, Mar. 2006.

[25] H. Tian, Q. Yu, and J. Kuang. Asymptotic stability of linear neutral delay differential-algebraic equations and Runge–Kutta methods. *SIAM J. Numer. Anal.*, 52(1):68–82, 2014.

[26] W. Zhu and L. R. Petzold. Asymptotic stability of linear delay differential-algebraic equations and numerical methods. *Appl. Numer. Math.*, 24:247 – 264, 1997.

[27] W. Zhu and L. R. Petzold. Asymptotic stability of Hessenberg delay differential-algebraic equations of retarded or neutral type. *Appl. Numer. Math.*, 27(3):309 – 325, 1998.