Acta Cybernetica **25** (2022) 677–702.

Verified Integration of Differential Equations with Discrete Delay

Andreas Rauh[a] and Ekaterina Auer[b]

Abstract

Many dynamic system models in population dynamics, physics and control involve temporally delayed state information in such a way that the evolution of future state trajectories depends not only on the current state as the initial condition but also on some previous state. In technical systems, such phenomena result, for example, from mass transport of incompressible fluids through finitely long pipelines, the transport of combustible material such as coal in power plants via conveyor belts, or information processing delays. Under the assumption of continuous dynamics, the corresponding delays can be treated either as constant and fixed, as uncertain but bounded (fixed or time-varying), or even as state-dependent. In this paper, we restrict the discussion to the first two classes and provide suggestions on how interval-based verified approaches to solving ordinary differential equations can be extended to encompass such delay differential equations. Selected close-to-life examples illustrate the theory from the perspective of robustness analysis in engineering applications.

Keywords: interval analysis, delay differential equations, uncertainty, dynamic systems, verified methods

1 Introduction

Delay differential equations arise in many areas of computational science and engineering. Representative examples can be found in the area of modeling biological processes [3], in the area of transport of incompressible substances [17], or in the area of control engineering if signal processing or communication delays are considered [13]. The latter are especially important in the field of networked control systems, where temporally varying communication delays are omnipresent [32]. Long-distance communications included in a closed-loop control framework belong

[a] Carl von Ossietzky Universität Oldenburg, Department of Computing Science, Group: Distributed Control in Interconnected Systems, D-26111 Oldenburg, Germany, E-mail: Andreas.Rauh@uni-oldenburg.de, ORCID: 0000-0002-1548-6547

[b] University of Technology, Business and Design, Department of Electrical Engineering, D-23966 Wismar, Germany, E-mail: Ekaterina.Auer@hs-wismar.de, ORCID: 0000-0003-4059-3982

DOI: 10.14232/actacyb.290904

to the same class of system models, with the most spectacular applications to be found in the fields of tele-operation [1] in medical surgery (i.e., communication and haptic device feedback between various places or even continents) or in space exploration where human operators take part as generators for reference signals or as decision makers.

A common example of the use of delay differential equations is the representation of the population dynamics in which species growth and mortality rates depend not only on the current state values but also on state information that is delayed by a certain finite time span. This helps to take into account the fact that each species first has to reach fertility age before it takes part in the reproduction process [5,6,9]. As mentioned before, similar considerations appear not only in the mathematical modeling of biological reproduction processes but also in epidemiological models or in (technical) control systems. In control, delay equations are employed when control actions depend additionally on certain previous state information due to communication delays, non-negligible time spans for information acquisition and processing, transport phenomena of physical substances or when control actions are decided upon based on (averaged) previous state information.

In this paper, we restrict the discussion to the case in which the system model has a single, finitely long, discrete delay. Then, the dynamic system model can be stated as

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(t, \mathbf{x}(t), \mathbf{x}(t-\tau^*)\right) \ , \quad \mathbf{x}(t) \in \mathbb{R}^n \ , \quad \tau^* \geq 0 \tag{1}$$

with

$$\mathbf{f} \ : \ \mathbb{R} \times \mathbb{R}^{2n} \mapsto \mathbb{R}^n \ . \tag{2}$$

Aside from the initial condition

$$\mathbf{x}(0) = \mathbf{x}_0 \tag{3}$$

at the single point $t = 0$, knowledge about a state initialization function

$$\mathbf{x}^*(t) := \mathbf{x}(t) \tag{4}$$

for the time span $-\tau^* \leq t \leq 0$ is required to determine a unique solution. Throughout this paper, we assume that the initialization function evaluated for $t = 0$ provides the same value as the point-valued initial condition $\mathbf{x}_0$, leading to a solution $\mathbf{x}(t)$ that is continuous at $t = 0$. Moreover, the $*$ symbol consistently denotes exact solutions to the simulation models under consideration and precisely known values for the delay.

A typical floating-point solution procedure for such system models is the so-called *method of steps* [33] in which the problem (1) is transformed into the non-autonomous initial value problem

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(t, \mathbf{x}(t), \mathbf{x}^*(t-\tau^*)\right) \ , \quad \text{with} \quad \mathbf{x}(0) = \mathbf{x}^*(0) \ , \tag{5}$$

that is used to predict the temporal evolution of $\mathbf{x}(t)$ over the time interval $t \in [0 \ ; \ \tau^*]$ for the already known delayed state information $\mathbf{x}^*(t-\tau^*)$. For successive intervals of the length τ^*, the procedure can be continued by utilizing the result from the previous time slice as the new initialization function.

This procedure for a floating point-based approximation of the solution to the delay differential equation is restricted to the case in which the initial state is given by a specific point in the state space and the delay time is known exactly. Uncertainty in the initial system states (or in the initialization function, respectively) could be treated by repeated simulations in a Monte-Carlo-like manner. Here, a well-known disadvantage is a potentially large computational effort that does not allow for determining verified outer bounds for the sets of reachable states if interval bounds for the initial conditions $\mathbf{x}_0$ as well as for the initialization function $\mathbf{x}^*(t)$ over the time interval $t \in [-\tau^* \,;\, 0]$ are given [20]. The same also holds for uncertain, but bounded delays τ^* which are themselves given as intervals. Two practice-relevant cases need to be distinguished here:

1. The delay is uncertain but constant over each time slice of a solution approach corresponding to the above-mentioned method of steps.

2. The delay is bounded from below and above, but may vary arbitrarily within these bounds.

Note that the second case is also strongly linked to scenarios in which only bounds for the initialization function (4) are available but the exact temporal evolution in the past is unknown.

From both a methodological and practical point of view, it is crucial to investigate such phenomena because increasing the delay time (for example, in the feedback path of a closed-loop control system) may turn a system with aperiodic dynamics into a system with oscillatory behavior. In addition, the introduction of delay may also turn asymptotically stable systems into unstable ones. The stability investigation of systems with delay, however, is not trivial and still a subject for ongoing research. For possible references on this topic, see [7, 9, 18, 21].

The main contribution of this paper is the generalization of verified solution techniques for classical, delay-free sets of ordinary differential equations to both cases mentioned above, namely, systems with constant as well as temporally varying but bounded delays. The general solution approach is derived exemplarily for an exponential state enclosure technique published by the authors in [29]. This approach makes use of techniques from the field of interval analysis [11, 19] to compute outer bounds that rigorously enclose all possible state trajectories of uncertain dynamic system models.

In contrast to the existing techniques with result verification for solving delay differential equations [10, 35, 36] (that employ Taylor methods or radii polynomial approaches), we do not focus on obtaining especially tight enclosures, which is necessary for a computational proof of such properties as periodicity of solutions. Instead, we aim at computing guaranteed outer solution enclosures by an approach that represents state trajectories by simple (exponential) functions in a computationally cheap way. It should be pointed out that the reduced complexity (resulting from the simple exponential state enclosures) would allow for an easier reimplementation of the whole algorithm on the GPU [2][1] or for development of online-adjustable control strategies in the frame of model-predictive control. Additionally, our method is directly applicable to scenarios in which parameters and the delay can vary temporally but stay bounded.

This paper is structured as follows. In Sec. 2, there is an overview of the initial value problems for delay differential equations to be solved by the approach suggested in this paper. Before detailed solution procedures are presented in Secs. 4 and 5, the interval-based exponential enclosure technique for classical ordinary differential equations from [29] is summarized in Sec. 3. This is a representative solution approach which is extended in this paper to the case of systems with a finite delay time. In Sec. 6, various numerical examples are presented including systems with exactly known delay times, with an uncertain but constant delay, and with a time-varying bounded delay. Finally, conclusions and an outlook on future work are given in Sec. 7.

2 Problem Formulation

Throughout this paper, the following variants of the delay differential equation model (1)–(4) are considered.

DDE1 The initial conditions and the initialization function in (3), (4), respectively, are both uncertain but bounded. The initial states are assumed to be included in the interval[2]

$$\mathbf{x}_0 \in [\underline{\mathbf{x}}_0 \ ; \ \overline{\mathbf{x}}_0] \quad , \tag{6}$$

where the component-wise defined inequalities $\underline{x}_{0,i} \leq \overline{x}_{0,i}$, $i \in \{1, \ldots, n\}$, hold. Analogously, the initialization function (4) is supposed to be given by the bounds[3]

$$\mathbf{x}^*(t) \in [\mathbf{x}_0] \quad \text{with} \quad \frac{\mathrm{d}\mathbf{x}^*(t)}{\mathrm{d}t} = \mathbf{0} \quad \text{for all} \quad t \in [-\tau^* \ ; \ 0] \quad . \tag{7}$$

In contrast to considering interval bounds for the initial states and for the initialization function, it is assumed that the delay $\tau^* > 0$ is precisely known[4].

[1] Such GPU implementations, accounting for data parallelism, are especially helpful to solve the task of an experiments-based parameter identification of dynamic systems.

[2] If necessary, we use the compact notation $[\mathbf{x}_0]$ throughout this paper to abbreviate the interval $[\underline{\mathbf{x}}_0 \ ; \ \overline{\mathbf{x}}_0]$. Bold face characters are employed to distinguish vectors (lower case) and matrices (upper case) from scalar variables.

[3] The algorithms presented in the following are not restricted to temporally constant initialization functions. They are chosen in this paper mainly to simplify the notation. Non-constant initializations arise naturally at each time instant $t > 0$ at which the integration is restarted if step size control strategies or multi-step simulations are performed.

[4] In **DDE1**, it is assumed that the delay time can be represented exactly by a machine number in the software implementation of the solution approach. If this is not the case, the formulation from **DDE2** can be used instead, where the point value τ^* is enclosed in a tight interval of machine numbers.

DDE2 As in **DDE1**, the initial system states $\mathbf{x}_0$ are assumed to be given by (6). However, the delay is now considered to be uncertain but temporally constant according to

$$\tau^* \in [\underline{\tau}^* \ ; \ \overline{\tau}^*] \quad \text{with} \quad \underline{\tau}^* < \overline{\tau}^* \ . \tag{8}$$

Therefore, the initialization function of **DDE1**, cf. (7), needs to be adapted in such a way that interval bounds are available for all $t \in [-\overline{\tau}^* \ ; \ 0]$. A special case of this definition arises for a delay-free lower bound $\underline{\tau}^* = 0$.

DDE3 This scenario is almost identical to **DDE2** except that the delays are not temporally constant. Now, the delay

$$\tau^* \in [\underline{\tau}^* \ ; \ \overline{\tau}^*] \quad \text{with} \quad 0 \leq \underline{\tau}^* < \overline{\tau}^* \tag{9}$$

may vary arbitrarily between its lower and upper bounds. No information on the temporal variation rate is available in this setting. Obviously, this is also true for the state initialization function, where arbitrary variations of $\mathbf{x}^*$ have to be accounted for within the respective interval bounds.

3 Verified Simulation Routine for Asymptotically Stable Delay-Free Ordinary Differential Equations

Delay-free state equations can often be assumed to be asymptotically stable in control engineering applications since, if they are not, an appropriate (state) feedback control law can be designed, in many cases with quasi-linear state-space representations. For finding enclosures of the solutions to such problems, the authors developed a verified exponential state enclosure technique [29] summarized briefly in this section. In the following sections, we extend this example of a solution procedure to the case of delay differential equations since they also play an important role in the area of control. However, any other verified approach for solving initial value problems for ordinary differential equations (e.g., from [14, 15, 20]) can be generalized analogously for the case of delay differential equations if the strategies described in Secs. 4 and 5 are employed.

Definition 1 (Quasi-linear autonomous model)**.** *For a nonlinear system model*

$$\dot{\mathbf{x}}(t) = \mathbf{a}\left(\mathbf{x}(t)\right) \ , \tag{10}$$

a quasi-linear dynamic system representation is given by the state-space representation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\left(\mathbf{x}(t)\right) \cdot \mathbf{x}(t) \ , \tag{11}$$

where the reformulation from (10) to (11) is obtained by exactly factoring out the state vector $\mathbf{x}(t)$ *from the nonlinear right-hand side* $\mathbf{a}\left(\mathbf{x}(t)\right)$ *so that all entries of*

$\mathbf{A}(\mathbf{x}(t))$ *are well-defined and finite for all reachable states*[5].

Definition 2 (Exponential state enclosure)**.** *The time-dependent exponential enclosure function (indicated by the index e)*

$$\check{\mathbf{x}}(t) \in [\mathbf{x}_e](t) := \exp([\mathbf{\Lambda}] \cdot t) \cdot [\mathbf{x}_e](0) \ , \ [\mathbf{x}_e](0) = [\mathbf{x}_0] \tag{12}$$

with the parameter matrix

$$[\mathbf{\Lambda}] := \operatorname{diag}\{[\lambda_i]\} \ , \ i \in \{1, \ldots, n\} \ , \tag{13}$$

defines a verified exponential state enclosure *for the system model (11) with* $\mathbf{x}(0) \in [\mathbf{x}_0]$ *if it is determined according to Theorem 1. It is then guaranteed to enclose all possible exact state trajectories* $\check{\mathbf{x}}(t)$.

Theorem 1 ([25, 29] Iteration for exponential state enclosures)**.** *The exponential state enclosure (12) is guaranteed to contain the set of all reachable states* $\check{\mathbf{x}}(T)$ *at the point of time* $t = T > 0$ *according to*

$$\check{\mathbf{x}}(T) \in [\mathbf{x}_e](T) := \exp([\mathbf{\Lambda}] \cdot T) \cdot [\mathbf{x}_e](0) \ , \tag{14}$$

if the elements on the main diagonal of $[\mathbf{\Lambda}]$ *are computed by the converging iteration*

$$[\lambda_i]^{\langle \kappa+1 \rangle} := \frac{a_i\left(\exp\left([\mathbf{\Lambda}]^{\langle \kappa \rangle} \cdot [t]\right) \cdot [\mathbf{x}_e](0)\right)}{\exp\left([\lambda_i]^{\langle \kappa \rangle} \cdot [t]\right) \cdot [x_{e,i}](0)} \ , \quad \kappa \in \{0, 1, 2, \ldots\} \ , \tag{15}$$

$i \in \{1, \ldots, n\}$, *with the prediction horizon* $[t] = [0 \ ; \ T]$.

Remark 1. As discussed in [25, 29], the iteration (15) is based on the application of the Picard iteration if the type of solution representations is restricted to the exponential expressions according to Def. 2. Therefore, the system models under consideration need to satisfy the same requirements that are needed for applying a Picard iteration (i.e., applicability of Banach's fixed point theorem) for finding verified state enclosures. Especially, we suppose differentiability of $\mathbf{a}(\mathbf{x})$ on the intervals $\exp\left([\mathbf{\Lambda}]^{\langle \kappa \rangle} \cdot [t]\right) \cdot [\mathbf{x}_e](0)$.

Remark 2. A typical initialization of the iteration (15) is $[\mathbf{\Lambda}]^{\langle 0 \rangle} = \operatorname{diag}\left\{[\lambda_i]^{\langle 0 \rangle}\right\}$, $i \in \{1, \ldots, n\}$, where these intervals are centered around the eigenvalues of a linearization of the nonlinear state equations for a representative point from the state enclosure at $t = 0$.

Remark 3. For sufficiently smooth system models (10), this approach can be easily extended to include a differential sensitivity analysis with respect to initial conditions and time-invariant parameters (see [22] for details).

[5] Such quasi-linear reformulations are typically not defined uniquely. For example, there exist infinitely many factorizations of the product $x_1 \cdot x_2 = a_1(x_2) \cdot x_1 + a_2(x_1) \cdot x_2$, where $a_1(x_2) = px_2$ and $a_2(x_1) = (1 - p) \cdot x_1$ with $p \in \mathbb{R}$.

Corollary 1 ([25, 29]). *If quasi-linear state-space representations according to Def. 1 are considered, the component-wise reformulation*

$$\dot{x}_i(t) = a_i\left(\mathbf{x}(t)\right) = \sum_{j=1}^{n} a_{ij}\left(\mathbf{x}(t)\right) \cdot x_j(t) \tag{16}$$

can be used for the state equations. In this case, the interval-related dependency problem [11], the wrapping effect [16], and the computational effort while using formula (15) can be reduced if it is reformulated symbolically into

$$\begin{aligned} \left[\lambda_i\right]^{\langle\kappa+1\rangle} := {} & a_{ii}\left(\left[\mathbf{x}_e\right]^{\langle\kappa\rangle}\left([t]\right)\right) \\ & + \sum_{\substack{j=1\\ j\neq i}}^{n} \left\{ a_{ij}\left(\left[\mathbf{x}_e\right]^{\langle\kappa\rangle}\left([t]\right)\right) \cdot e^{\left(\left(\left[\lambda_j\right]^{\langle\kappa\rangle} - \left[\lambda_i\right]^{\langle\kappa\rangle}\right)\cdot[t]\right)} \cdot \frac{\left[x_{e,j}\right](0)}{\left[x_{e,i}\right](0)} \right\} \ . \end{aligned} \tag{17}$$

This reformulation is especially beneficial if the system matrix $\mathbf{A}\left(\mathbf{x}(t)\right)$ *is diagonally dominant.*

Remark 4. The exponential enclosure technique according to Theorem 1 is applicable to scenarios characterized by solution sets in which the value zero is not contained in the domain of reachable states. This becomes obvious in Eq. (15), where the guaranteed state enclosure appears in the denominator of the iteration formula [25, 28]. For linear systems with oscillatory dynamics and precisely known parameters, this issue can be resolved by a suitable time-invariant change of coordinates as shown in the last example from Sec. 6. For this purpose, Theorem 1 needs to be generalized using complex-valued interval techniques [28, 29]. Alternatively, if the value zero is only *crossed* a finite number of times, enclosures can be obtained by switching to the Picard iteration based ValEncIA-IVP technique or to a low-order Taylor series expansion over the respective time interval.

4 Delay Differential Equations with a Constant, Precisely Known Delay

In general, delay differential equations with bounded uncertainty can be treated by adapting the method of steps to rely on verified solvers for ordinary differential equations instead of floating-point ones. For the case of **DDE1**, this method can be employed directly after defining the time intervals

$$[\mathcal{T}]_m = \left[m\tau^* \ ; \ (m+1)\,\tau^*\right] \ , \quad m \in \mathbb{N}_0 \ , \tag{18}$$

with a length that is equal to the a-priori known constant delay τ^*. Then, an exponential state enclosure can be introduced according to Def. 3 for each $[\mathcal{T}]_m$.

Definition 3 (Exponential state enclosure for delay differential equations). *The time-dependent exponential enclosure function for the m-th time interval* $t \in [\mathcal{T}]_m$

$$\check{\mathbf{x}}(t) \in \left[\mathbf{x}_e\right]_m(t) := \exp\left(\left[\mathbf{\Lambda}\right]_m \cdot \left(t - m\tau^*\right)\right) \cdot \left[\mathbf{x}_e\right]\left(m\tau^*\right) \tag{19}$$

with $[\mathbf{x}_e](0) = [\mathbf{x}_0]$ *and the parameter matrix*

$$[\mathbf{\Lambda}]_m := \operatorname{diag}\left\{[\lambda_i]\right\}_m \ , \ i \in \{1, \ldots, n\} \ , \tag{20}$$

defines a verified exponential state enclosure *for the delay differential equation of type* ***DDE1*** *if it is determined according to Theorem 2. For compatibility with the initialization function (7),* $[\mathbf{\Lambda}]_{-1}$ *is set to*

$$[\mathbf{\Lambda}]_{-1} := \mathbf{0} \ , \quad \textit{corresponding to} \quad [\mathbf{x}_e]_{-1}(t) \equiv [\mathbf{x}_0] \ . \tag{21}$$

Theorem 2 (Iteration for delay differential equations of type **DDE1**)**.** *The exponential state enclosure (19) is guaranteed to contain the set of all reachable states* $\check{\mathbf{x}}(T)$ *at the point of time* $t = T \in [\mathcal{T}]_m$*, that is,*

$$\check{\mathbf{x}}(T) \in [\mathbf{x}_e]_m(T) := \exp\left([\mathbf{\Lambda}]_m \cdot (T - m\tau^*)\right) \cdot [\mathbf{x}_e](m\tau^*) \ , \tag{22}$$

if the elements on the main diagonal of $[\mathbf{\Lambda}]_m$ *are computed using the converging iteration*

$$[\lambda_i]_m^{\langle\kappa+1\rangle} := \frac{f_i\left([t] \ , \ \exp\left([\mathbf{\Lambda}]_m^{\langle\kappa\rangle} \cdot ([t] - m\tau^*)\right) \cdot [\mathbf{x}_e](m\tau^*) \ , \ [\mathbf{x}_e]_{m-1}([t] - \tau^*)\right)}{\exp\left([\lambda_i]_m^{\langle\kappa\rangle} \cdot ([t] - m\tau^*)\right) \cdot [x_{e,i}](m\tau^*)} \ , \tag{23}$$

$i \in \{1, \ldots, n\}$*, with the prediction horizon* $[t] = [m\tau^* \ ; \ T] \subseteq [\mathcal{T}]_m$*, where all* f_i *are defined according to Eq. (1).*

Proof. Due to the restriction $T \in [\mathcal{T}]_m$, the third argument $[\mathbf{x}_e]_{m-1}([t] - \tau^*)$ of f_i in (23) depends only on solution enclosures from the previous time interval $[\mathcal{T}]_{m-1}$ and is, therefore, completely known. A non-autonomous initial value problem can be solved at this time step following the general idea of the method of steps (5). Hence, the proof is the same as for Theorem 1 (originally published in [25, 29]), if the iteration (23) is substituted for (15). □

Remark 5. Overestimation in the elements $[\lambda_i]_m^{\langle\kappa+1\rangle}$ in (23) appearing due to multiple dependencies on common interval variables can be reduced by exploiting the quasi-linear structure of the problem and reformulating the iteration symbolically (cf. Corollary 1). Another overestimation reduction possibility is to employ a classical subdivision strategy for range computation of interval expressions, also used in global optimization [4]. For that, it is necessary to subdivide the time interval $[t]$ into multiple subintervals, carry out the procedure and, finally, determine the convex interval hull of all resulting enclosures over the subintervals. An additional advantage of subdivision strategies for delay equations is that the time subintervals can also be used while determining the bounds for $[\mathbf{x}_e]_{m-1}([t] - \tau^*) = [\mathbf{x}_e]_{m-1}([(m-1)\tau^* \ ; \ T - \tau^*])$.

The subdivision strategy described in the remark can be used to control the step size. For this purpose, the integration time horizon $[\mathcal{T}]_m$ is split into multiple shorter time slices and solution parameters $[\lambda_i]_{m,\iota}$ are computed successively for each of the temporal subintervals $[m\tau^* \ ; \ m\tau^* + \tau_1]$, $[m\tau^* + \tau_1 \ ; \ m\tau^* + \tau_2]$, ..., $[m\tau^* + \tau_{\iota-1} \ ; \ m\tau^* + \tau_\iota]$, ... with $0 < \tau_1 < \tau_2 < \ldots < \tau_\iota < \ldots < \tau^*$.

5 Delay Differential Equations with Uncertain Delay

In this section, solution procedures for delay differential equations with uncertain delays are presented. Here, we consider two cases of bounded delays: either temporally constant or arbitrarily varying within the respective interval bounds.

5.1 Delay Differential Equations with a Constant, Interval-Bounded Delay

5.1.1 Systems with Strictly Non-Zero Delay

The approach from Sec. 4 can be extended to cover systems with a strictly non-zero time delay in a straightforward way. For that purpose, we introduce the time intervals

$$[\mathcal{T}]_m = [t_m \ ; \ t_{m+1}] \ , \quad m \in \mathbb{N}_0 \ , \quad t_0 = 0 \ , \tag{24}$$

where the infima and suprema of $[\mathcal{T}]_m$ denote the temporal discretization mesh with which the (exponential) solution enclosures for the delay differential equation of type **DDE2** are computed. In this subsection, we further assume that $t_{m+1} - t_m \leq \underline{\tau}^*$ holds. This restriction will be removed in the following subsection, where the case of a possibly vanishing time delay is investigated. The following definition is a generalization of Def. 3 which allows us to represent state enclosures covering multiple points t_m of the temporal discretization mesh.

Definition 4 (Generalized exponential state enclosure for DDEs)**.**
A generalized time-dependent exponential enclosure over a time interval $t \in [\mathcal{T}]_{\mathrm{a}}^{\mathrm{b}} = [t_{\mathrm{a}} \ ; \ t_{\mathrm{b}}]$, $t_{\mathrm{a}} \leq t_{\mathrm{b}}$, $t_{\mathrm{a}} \geq 0$, *is given by*

$$[\mathbf{x}_e]\left([\mathcal{T}]_{\mathrm{a}}^{\mathrm{b}}\right) = \begin{cases} [\mathbf{x}_e]_{m_{\mathrm{a}}}\left([\mathcal{T}]_{\mathrm{a}}^{\mathrm{b}}\right) & \text{if } \ m_{\mathrm{a}} = m_{\mathrm{b}} \ , \\ \bigcup\limits_{j=m_{\mathrm{a}}}^{m_{\mathrm{b}}} [\mathbf{x}_e]_j \left([\mathcal{T}]_{\mathrm{a}}^{\mathrm{b}} \cap [\mathcal{T}]_j\right) & \text{otherwise} \ . \end{cases} \tag{25}$$

Here, the indices m_ι, $\iota \in \{\mathrm{a}, \mathrm{b}\}$, *of the corresponding discretization points are determined according to*

$$m_\iota = \max_{m \in \mathcal{Z}}\{m\} \quad \text{with} \quad \mathcal{Z} := \{m \mid m \in \mathbb{N}_0 \ \text{and} \ 0 \leq t_{m_{\mathrm{a}}} \leq t_\iota < t_{m_\iota + 1}\} \ . \tag{26}$$

The individual interval enclosure functions in (25) are given by

$$\check{\mathbf{x}}(t) \in [\mathbf{x}_e]_m(t) := \exp\left([\mathbf{\Lambda}]_m \cdot (t - t_m)\right) \cdot [\mathbf{x}_e](t_m) \quad \text{for} \quad t \in [\mathcal{T}]_m \tag{27}$$

with the diagonal parameter matrices $[\mathbf{\Lambda}]_m$ *computed as in Theorem 3.*

In addition, the definition (25) can be extended to the case $t_{\mathrm{b}} \leq 0$ *by setting* $[\mathbf{x}_e]([t_{\mathrm{a}} \ ; \ t_{\mathrm{b}}]) \equiv [\mathbf{x}_0]$*; analogously,* $[\mathbf{x}_e]([t_{\mathrm{a}} \ ; \ t_{\mathrm{b}}]) \equiv [\mathbf{x}_0] \cup [\mathbf{x}_e]([0 \ ; \ t_{\mathrm{b}}])$ *holds for* $t_{\mathrm{a}} < 0$ *and* $t_{\mathrm{b}} \geq 0$.

Remark 6. All intersections $[\mathcal{T}]_{\mathrm{a}}^{b} \cap [\mathcal{T}]_m$ of time intervals in Eq. (25) restrict the domains on which the respective functions are evaluated to the domains on which the parameter matrices $[\mathbf{\Lambda}]_m$ and, hence, the corresponding solution tubes are defined.

Theorem 3 (Iteration for delay differential equations of type **DDE2**). *If the discretization mesh is defined such that $t_{m+1} - t_m \leq \underline{\tau}^*$, the exponential state enclosure (25) is guaranteed to contain the set of all reachable states $\check{\mathbf{x}}(T)$ at the point of time $t = T \in [\mathcal{T}]_m$, that is,*

$$\check{\mathbf{x}}(T) \in [\mathbf{x}_e]_m (T) := \exp\left([\mathbf{\Lambda}]_m \cdot (T - m\tau^*)\right) \cdot [\mathbf{x}_e] (m\tau^*) \quad , \tag{28}$$

if $[\mathbf{\Lambda}]_m$ is computed by the converging iteration

$$[\lambda_i]_m^{\langle\kappa+1\rangle} := \frac{f_i\left([t] \ , \ \exp\left([\mathbf{\Lambda}]_m^{\langle\kappa\rangle} \cdot ([t] - t_m)\right) \cdot [\mathbf{x}_e] (t_m) \ , \ [\mathbf{x}_e] ([t] - [\tau^*])\right)}{\exp\left([\lambda_i]_m^{\langle\kappa\rangle} \cdot ([t] - t_m)\right) \cdot [x_{e,i}] (t_m)} \quad , \tag{29}$$

$i \in \{1, \ldots, n\}$, with the prediction horizon $[t] = [t_m \ ; \ T] \subseteq [\mathcal{T}]_m$.

Proof. In the last argument of the numerator term in (29), $[\mathbf{x}_e] ([t] - [\tau^*])$ is independent of the parameter matrix $[\mathbf{\Lambda}]_m^{\langle\kappa\rangle}$ for the current time interval $[\mathcal{T}]_m$. This means that this term can again be interpreted as an external input to a non-autonomous system of ordinary differential equations. Therefore, the proof is identical to the proof of Theorem 2. □

Remark 7. For the time interval $[t]$, a subdivision strategy can be used in full analogy to Sec. 4. In addition, the interval $[\tau^*]$ may be subdivided into multiple time intervals, followed by determining separate solution enclosures $[\mathbf{x}_e]_m (T)$ for each delay subinterval when evaluating Eq. (29). In the final step, the convex interval hull over all individual solutions can be determined to describe the set of reachable states. This is a direct consequence of the assumption of an uncertain but temporally constant delay.

Remark 8. Under the assumptions of this subsection, the Definitions 3 and 4 become identical.

5.1.2 Systems Involving Zero Delay

If $0 \in [\tau^*]$ or if a discretization mesh with $t_{m+1} - t_m > \underline{\tau}^*$ is employed, Theorem 3 needs to be adjusted according to the following Corollary 2.

Corollary 2 (Iteration for **DDE2** with potentially zero delay). *In the case of a potentially vanishing delay, the exponential state enclosure (28) contains the set of all reachable states $\check{\mathbf{x}}(T)$ at the point of time $t = T \in [\mathcal{T}]_m$ if $[\mathbf{\Lambda}]_m$ is set to the outcome of the iteration (29), where the last numerator term $[\mathbf{x}_e] ([t] - [\tau^*])$ is evaluated according to Def. 4 with $t_{\mathrm{a}} = \inf([t] - [\tau^*])$ and $t_{\mathrm{b}} = \sup([t] - [\tau^*])$ as a function of all parameters $[\lambda_i]_m^{\langle\kappa\rangle}$.*

Remark 9. Splitting both intervals $[t]$ and $[\tau^*]$ into subintervals as described in the previous subsection remains admissible due to the time invariance of the delay τ^*.

5.2 Delay Differential Equations with Uncertain, Temporally Varying Delays

Both Theorem 3 and Corollary 2 can be applied to the case of time-varying, uncertain, bounded delays. Note, however, that subdivision strategies suggested as a countermeasure against overestimation in Sec. 5.1 need to be handled with more care.

If the intervals $[t]$ and $[\tau^*]$ are subdivided, it is necessary to compute multiple results $[\lambda_i]_m^{\langle\kappa+1\rangle}$ followed by the convex hull operation unifying them for each subsequent evaluation of the iterations according to Theorem 3 and Corollary 2. This is the only admissible subdivision strategy for reducing the dependency problem in this case. Note that this subdivision approach is equally valid for reduction of overestimation that is caused by wide intervals $[\lambda_i]_m^{\langle\kappa\rangle}$ resulting from the previous iteration step.

6 Numerical Examples

In this section, representative application scenarios are presented for the proposed interval technique. They are linear scalar (cf. Sec. 6.1) and multi-dimensional system models (cf. Sec. 6.5) with exactly known delay, a nonlinear process model with exactly known and uncertain delay that is inspired by mathematical models from the field of population dynamics (cf. Sec. 6.2 and 6.3) as well as the simulation of Wright's equation with an uncertain parameter (cf. Sec. 6.4). The specified values for delay times are assumed to be represented by the closest floating point number, where for the cases of an exactly known time delay this value is an integer multiple of the underlying sampling time. Note that not exactly representable delay times can easily be accounted for by the setting in **DDE2**.

6.1 A System Model with an Exact Analytic Solution

As the first application scenario, consider the dynamic system model [30, Chap. 12]

$$\dot{x}(t) = a \cdot x(t - \tau^*) \tag{30}$$

of type **DDE1** with the exactly known, non-zero delay $\tau^* > 0$. If the initialization function for $t \leq 0$ is equal to the constant $x(t) \equiv x_0$ and if the identical initial condition $x(0) = x_0$ is considered, the exact solution $x_m(t)$ can be computed for each time interval $t \in [\mathcal{T}]_m$, $m \in \mathbb{N}_0$, cf. (18), by applying the method of steps in a recursive manner. For that purpose, the system model (30) is reformulated into

$$\int\limits_{x_{m-1}(m\tau^*)}^{x_m(t)} \mathrm{d}\chi = a \cdot \int\limits_{m\tau^*}^{t} x_{m-1}(\eta - \tau^*)\,\mathrm{d}\eta \tag{31}$$

by separating the variables t and x. This leads to the solution representation

$$x_m(t) = x_{m-1}(m\tau^*) + a \cdot \int\limits_{m\tau^*}^{t} x_{m-1}\left(\eta - \tau^*\right) \mathrm{d}\eta \ , \tag{32}$$

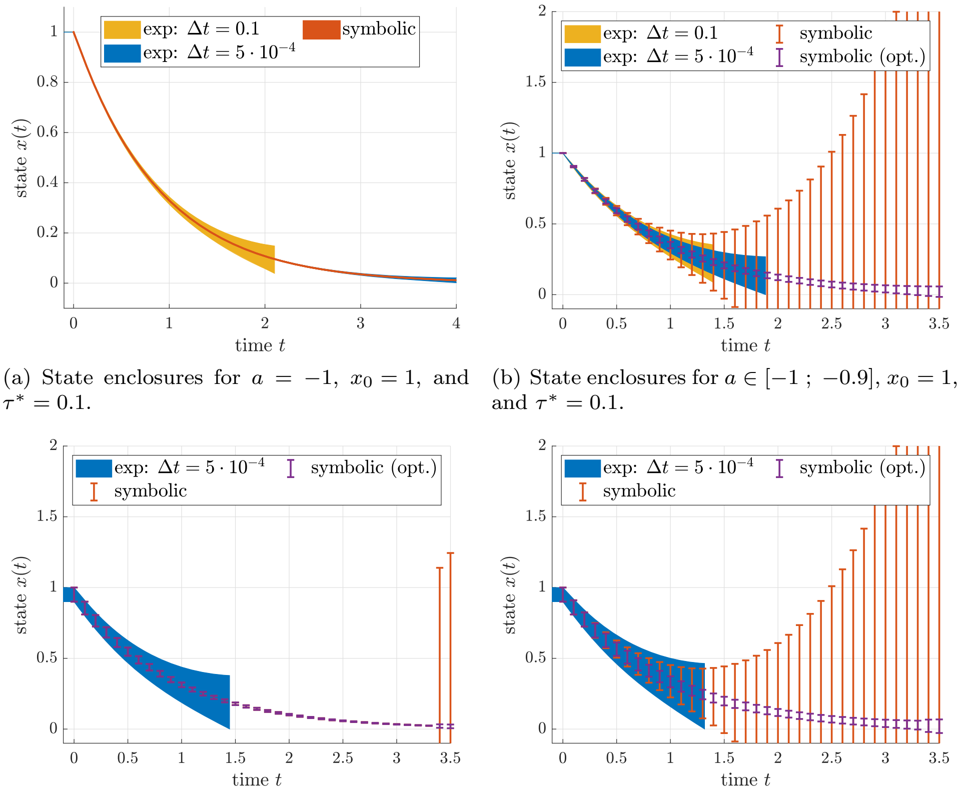
where both formulas (31) and (32) are initialized with $x_{-1}(t) \equiv x_0$. Evaluating the expression (32) at integer multiples of the delay time, i.e., for $t = m\tau^*$, $m \in \mathbb{N}_0$, yields the closed-form solution representation

$$x(t) = \begin{cases} x_0 & \text{for} \ \ t = 0 \\ x_0 \cdot (a\tau^* + 1) & \text{for} \ \ t = \tau^* \\ \frac{x_0}{2}\left((a\tau^*)^2 + 4a\tau^* + 2\right) & \text{for} \ \ t = 2\tau^* \\ \frac{x_0}{6}\left((a\tau^*)^3 + 12\,(a\tau^*)^2 + 18a\tau^* + 6\right) & \text{for} \ \ t = 3\tau^* \\ \frac{x_0}{24}\left((a\tau^*)^4 + 32\,(a\tau^*)^3 + 108\,(a\tau^*)^2 + 96a\tau^* + 24\right) & \text{for} \ \ t = 4\tau^* \\ \ldots \ . \end{cases} \tag{33}$$

The result (33) can be extended by the well-known rules of interval analysis to outer state enclosures, if uncertainty in the initial state $x_0 \in [x_0]$ and in the time-invariant parameter $a \in [a]$ needs to be accounted for. This interval representation (evaluated in the following in a naive way in terms of the natural interval extension [11] in IntLab [31]) serves as one of the references to which the novel simulation procedure according to Sec. 4 can be compared.

Fig. 1 provides a comparison of the novel iteration approach with an interval-based evaluation of the analytic solution representation according to Eq. (33). The first simulation result in Fig. 1(a) visualizes the influence of the integration step size on the tightness of the solution obtained by the application of Theorem 2. Exemplarily, the constant integration step sizes $\Delta t = \tau^* = 0.1$ (corresponding to the direct generalization of the method of steps) and $\Delta t = \frac{\tau^*}{200} = 5 \cdot 10^{-4}$ are compared. It can be seen that the simulation for the larger step size breaks down after $t = 2.1$ because overestimation leads to solutions that include the value zero in the denominator of the iteration of Theorem 2. To some extent, this can be avoided by reducing the step size. Hence, the other versions of the example are investigated using this reduced step size. Alternatively, it is possible to apply a different enclosure definition (such as the basic state enclosure of ValEncIA-IVP) during those time spans in which the solution crosses zero.

The subplot in Fig. 1(b) shows that a naive interval extension of the analytic solution representation leads to significantly wider interval bounds than the proposed iteration procedure if a is the only uncertain parameter in the model. This is mainly caused by multiple dependencies on the interval parameter $[a]$ in the analytic solution representation. This dependency effect can be reduced by advanced interval evaluation techniques. In Fig. 1, enhanced enclosures are visualized which are computed by subdividing the parameter domain into an equidistant grid with 100 subintervals for each of the parameters. The dependency is less critical if a



(a) State enclosures for $a = -1$, $x_0 = 1$, and $\tau^* = 0.1$.

(b) State enclosures for $a \in [-1\ ;\ -0.9]$, $x_0 = 1$, and $\tau^* = 0.1$.

(c) State enclosures for $a = -1$, $x_0 \in [0.9\ ;\ 1.0]$, and $\tau^* = 0.1$.

(d) State enclosures for $a \in [-1\ ;\ -0.9]$, $x_0 \in [0.9\ ;\ 1.0]$, and $\tau^* = 0.1$.

Figure 1: State enclosures for the linear system model (16) with constant delay.

is set to a point value in Fig. 1(c), where the interval evaluation of the analytic solution representation deteriorates rapidly if overestimation leads to the fact that the value zero is included in the solution set. Note that the exponential enclosure technique in this case breaks down before $t = 1.5$. This can be avoided by subdividing the initial state interval and by subsequently performing multiple simulations for the respective subintervals[6]. For the case in Fig. 1(d), where both the initial condition and the system parameter are set to interval quantities, the numerical and analytic solutions are quite similar up to the point where the exponential enclosure technique is no longer valid due to the inclusion of the value zero in the computed state bounds.

[6] As already mentioned in the introduction of this paper, such subdivisions are typically employed anyway for an experiments-based parameter identification of dynamic systems. The proposed solution algorithm can easily be adjusted to a GPU implementation [2] which performs a parallelized simulation for all subintervals by exploiting the concept of data parallelism.

Note that exponential state enclosures computed for a single parameter interval $[a]$ and a single box of initial conditions comprising the bounds on the initialization function $[x_0]$ are generally wider than an optimized interval evaluation of the closed-form solution. This is caused by the fact that Theorem 2, evaluated for a single interval box, provides state enclosures that are valid for arbitrary temporal parameter variations within the respective box, while the analytic solution representation assumes a time invariant parameter with vanishing time derivatives of the initialization function for $t < 0$.

6.2 A Nonlinear System Model with Constant Bounded Delay

As the second application, consider the nonlinear system model

$$\dot{x}\left(t\right) = a \cdot x(t) + b \cdot x^3\left(t - \tau^*\right) \quad \text{with} \quad x(t) \equiv x_0 \quad \text{for} \quad t \leq 0 \ , \tag{34}$$

where $a \in [a] = [-0.2\ ;\ -0.1]$, $b \in [b] = [0.01\ ;\ 0.02]$, $x_0 \in [x_0] = [0.9\ ;\ 1.0]$, and $\tau^* \in [\tau^*] = [0.1\ ;\ 1.0]$ are temporally constant interval parameters.

This delay differential equation model reflects a simplified problem from the field of population dynamics [3], where the state variable x represents a species concentration, the parameter a a decay rate (i.e., due to mortality), and b the rate of reproduction depending in a cubic manner on the delayed state information. The parameter τ^* describes the uncertain age of maturation after which the members of the population participate in the reproduction process.

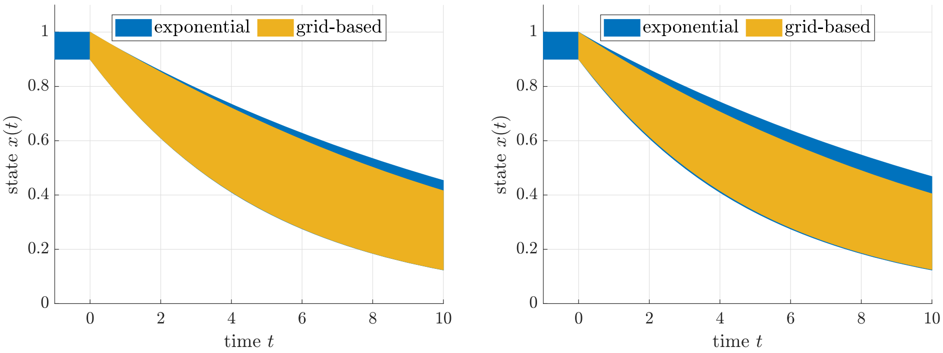
Since the delay parameter is uncertain, the interval-based solution to this model is computed using the methods of Sec. 5.1. A grid-based floating point solution, obtained with the help of the MATLAB routine `dde23` and a maximum step size $\Delta t = 0.01$, is included for comparison in Fig. 2(a). It can be seen that the exponential interval technique encloses the grid-based evaluation in tight lower and upper bounds, where the lower bound especially has almost no overestimation. Note that the grid-based simulation consists of 10^4 individual system simulations because all four uncertain parameters were independently subdivided into 10 points each.

In contrast, the exponential state enclosure was determined without subdividing the interval bounds $[a]$ and $[b]$. For a sake of comparison with the assumption of arbitrary varying initialization functions and delays in the interior of the respective interval bounds (which is the subject of the following subsection), $[x_0]$ and $[\tau^*]$ were both divided into 10 equally wide subintervals, leading to a total of 100 interval simulations.

6.3 A Nonlinear System Model with Time-Varying Bounded Delay

As a third example, the model in Eq. (34) is reconsidered. Now, both the time delay and the state initialization function are assumed to be arbitrarily variable within their respective interval bounds. Hence, only a single interval evaluation (instead of the 100 simulations from the previous subsection) was performed to obtain the exponential state enclosure. It can be noticed that the temporal variability of both of these quantities has only a minor influence on the solution enclosures because the interval bounds obtained by the exponential enclosure technique in Fig. 2(b) are only slightly wider than those in Fig. 2(a). However, as expected, the previous time-invariant case represents a subset of the solution to the time-varying scenario.

For the sake of comparison, a grid-based simulation is included in Fig. 2(b). It is based on the evaluation of the dynamic system model (34) with the help of the MATLAB routine `ddesd` with a maximum step size of $\underline{\tau}^* = 0.1$. In total, $2,000$ equally distributed random sequences for the state initialization function and for the time delay were generated to mimic the influence of the uncertain quantities.



(a) Uncertain but constant delay.

(b) Time-varying bounded delay.

Figure 2: State enclosures for the linear system model (34) with uncertain delay.

For both Secs. 6.2 and 6.3, the use of the interval-based simulation approach has the advantage of a much smaller number of required system evaluations than in the grid-based counterpart, while further providing state enclosures that contain all reachable state values with certainty.

In Tab. 1, a comparison of the computing times[7] for the grid-based floating point implementation using the routine `ddesd` and the novel exponential enclosure approach implemented with the help of the interval library INTLAB [31] (version 10.2) is given[8]. For identical discretization step sizes, and a grid-based simulation consisting of 2,000 individual runs, the exponential enclosure technique is faster by a factor of at least 175. Although the interval simulation has not been optimized for speed, it is faster by a factor of 8.77 even if the grid-based simulation is carried out with the largest investigated step size and the interval-based simulation with the smallest. In addition, it can be seen from Tab. 1 that a reduction of the

[7] All simulations were performed in MATLAB R2019b on a notebook computer under Windows 10, 64bit, 8 GB RAM, Intel Core i7-4500U CPU (@1.80GHz).

[8] Prototypical implementations are available for download on `https://github.com/ValEncIA-IVP/` and `www.valencia-ivp.com`.

discretization step size in the proposed approach leads to tighter interval bounds. A meaningful reduction of the computed interval diameters can be observed until $\Delta t = 0.01$ in the example considered in Sec. 6.3.

Table 1: Comparison of the grid-based simulation and the exponential enclosure technique for the example of Sec. 6.3.

step size	`ddesd`	proposed approach	speedup	$\mathrm{diam}\{[x_e]\,(10)\}$
$\Delta t = 0.1$	0.1272 s	0.9811 s	259.3	0.3462
$\Delta t = 0.01$	1.2297 s	12.474 s	197.1	0.3445
$\Delta t = 0.005$	2.5573 s	29.017 s	176.2	0.3444

6.4 Simulation of Wright's Equation with an Uncertain Parameter

A further nonlinear application scenario in this section illustrates a possible approach to circumvent cases in which the proposed simulation technique is not directly applicable due to a division by zero in the Theorems 2 and 3. We consider Wright's equation as published in [37]. Originally, it was formulated as

$$\dot{y}(t) = -p \cdot y(t-1) \cdot (1 + y(t)) \tag{35}$$

with the parameter $p > 0$. After the time-invariant change of coordinates

$$x(t) = 1 + y(t) \ , \tag{36}$$

the equivalent formulation

$$\dot{x}(t) = -p \cdot (x(t-1) - 1) \cdot x(t) \tag{37}$$

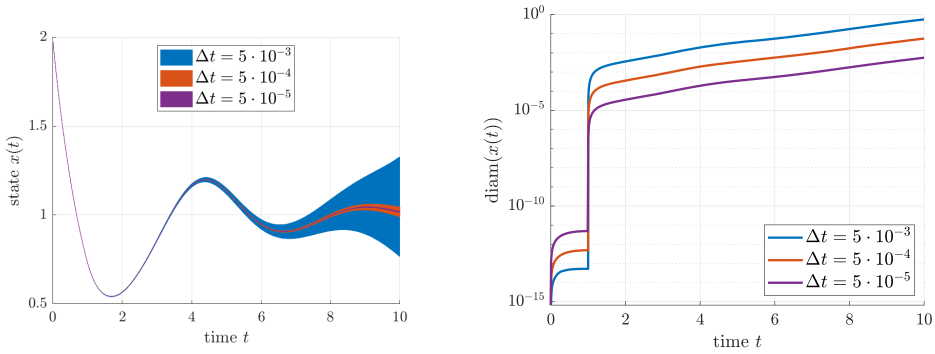
is obtained, for which we assume a constant initialization function $x(t) = x_0$ for $t \leq 0$ with the consistent initial condition $x_0 = 2$ at the point $t = 0$ in the remainder of this subsection.

The change of coordinates (36) helps to avoid that solutions cross the value $y = 0$ if initialized with non-negative functions $y(t) > 0$ for $t \leq 0$ and positive parameters $p > 0$. The advantage of the exclusion of the solution $y = 0$ from the solution set is that a singularity in the iterations of Theorems 2 and 3 as well as Corollary 2 can be avoided.

For the Wright equation, this change of coordinates leads to a simplification of the iteration formula (23) according to

$$[\lambda_i]_m^{\langle\kappa+1\rangle} := -p \cdot [x_e]_{m-1}\left([t] - \tau^*\right) \quad \text{with} \quad \tau^* = 1 \ . \tag{38}$$

For the known parameter $p = 1$, Fig. 3 shows a comparison of the solution enclosures with the corresponding widths of the computed interval bounds for different discretization step sizes Δt. It can be seen clearly that the reduction in the



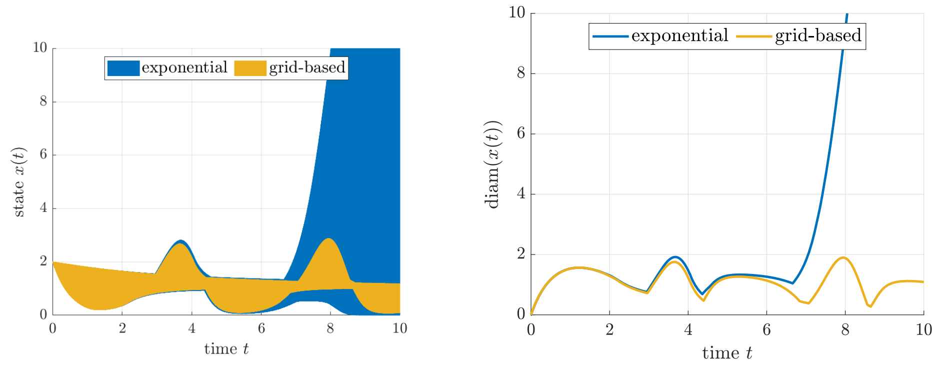
(a) State enclosure for different discretization step sizes.

(b) Interval diameters for different discretization step sizes.

Figure 3: Simulation of the transformed Wright equation (37) for $p = 1$ with exactly known initialization function and initial condition.

interval widths is proportional to the reduction of the discretization step size, going along with a proportional increase of the computing time. It should be pointed out additionally that the reformulated iteration in (38) has the advantage for this specific benchmark scenario that its right-hand side is independent on the parameter to be computed and, hence, can be resolved explicitly by exploiting already pre-computed solution enclosures.

Using the step size $\Delta t = 5 \cdot 10^{-4}$, the simulation was repeated in Fig. 4 for the uncertain parameter interval $p \in [0.1\ ;\ 2]$ in combination with equidistantly subdividing it into 100 subdomains. The resulting exponential enclosures tightly



(a) State enclosures: Exponential solution technique vs. grid-based approximation.

(b) Interval diameters: Exponential solution technique vs. grid-based approximation.

Figure 4: Simulation of the transformed Wright equation (37) for $p \in [0.1\ ;\ 2]$ with exactly known initialization function and initial condition.

enclose a grid-based non-verified simulation performed with the routine `ddesd` up to $t \approx 6$. Afterwards, interval-related overestimation leads to a rapid inflation of the computed bounds. In future work, this can be countered by extending the complex-valued enclosure approach from [28, 29] to the case of scalar differential equations with delay. The oscillatory behavior of Wright's equation can then be better represented by not choosing purely real solution parameters $[\lambda_i]_m^{\langle\kappa+1\rangle}$. For multi-dimensional systems, this approach is already implemented, see the simulation results in the following subsection.

6.5 Spring-Mass-Damper System

As a final application scenario, the oscillation attenuation of a spring-mass-damper system with the position variable x_1, the velocity x_2, and the actuating force x_3 is considered. The state equations

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \begin{bmatrix} 0 & 1 & 0 \\ p_1 \cdot a_{21} & p_2 \cdot a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix} \cdot \mathbf{x}(t) + \begin{bmatrix} 0 \\ 0 \\ b_3 \end{bmatrix} \cdot u(t) \\ &= \mathbf{A}\left(p_1, p_2\right) \cdot \mathbf{x}(t) + \mathbf{b} \cdot u(t)\end{aligned} \tag{39}$$

with a delay-free realization of the control input $u(t)$ were presented in [26, 27] as a benchmark scenario for the design of a robust output feedback controller in which the input $u(t)$ was chosen to be proportional to the velocity $x_2(t)$. From an engineering viewpoint, this model describes the simplest linear representation of an actively controlled wheel suspension system with a first-order lag behavior of the actuator. In the following, the control law is defined as

$$u(t) = 0.8 \cdot x_2(t - \tau^*) \ , \tag{40}$$

where the constant gain factor 0.8 guarantees asymptotic stability of the nominal system with the parameters $a_{21} = -200$, $a_{22} = -15$, $a_{23} = -400$, $a_{33} = -200$, $b_3 = 10$, $p_1 = 1$, and $p_2 = 1$.

For the simulations in this subsection, we consider the cases of a delay-free system ($\tau^* = 0$), a relatively small delay ($\tau^* = 10^{-4}$) and a large delay ($\tau^* = 0.1$). In all scenarios, the integration step size is constant with $\Delta t = 10^{-5}$. Moreover, the independent parameters p_i, $i \in \{1, 2\}$, with their midpoints $p_{i,\mathrm{m}} = 1$ are assumed to have the following identical bounds in the four cases shown in Figs. 5–7:

P1 $p_i = p_{i,\mathrm{m}}$;

P2 $p_i \in p_{i,\mathrm{m}} + [-0.005 \ ; \ 0.005]$;

P3 $p_i \in p_{i,\mathrm{m}} + [-0.1 \ ; \ 0.1]$;

P4 $p_i \in p_{i,\mathrm{m}} + [-0.5 \ ; \ 0.5]$.

In all cases, the system's initial conditions (and temporally constant initialization functions for $\tau^* > 0$) are defined as

$$\mathbf{x}_0 = \begin{bmatrix} 1 & 0 & 0.5 \end{bmatrix}^T \ . \tag{41}$$

To reduce the influence of the wrapping effect, a time invariant change of coordinates is performed according to

$$\mathbf{z}(t) = \mathbf{V}^{-1} \cdot \mathbf{x}(t) \ , \tag{42}$$

where $\mathbf{V}$ is the columnwise defined matrix of eigenvectors of

$$\mathbf{A}\left(p_{1,\mathrm{m}}, p_{2,\mathrm{m}}\right) + \mathbf{b} \cdot \begin{bmatrix} 0 & 0.8 & 0 \end{bmatrix} \tag{43}$$

in the delay-free case and the eigenvector matrix of $\mathbf{A}\left(p_{1,\mathrm{m}}, p_{2,\mathrm{m}}\right)$ in the case $\tau^* > 0$. For a non-zero delay, the change of coordinates leads to a complex-valued set of state equations as introduced in [28, 29].

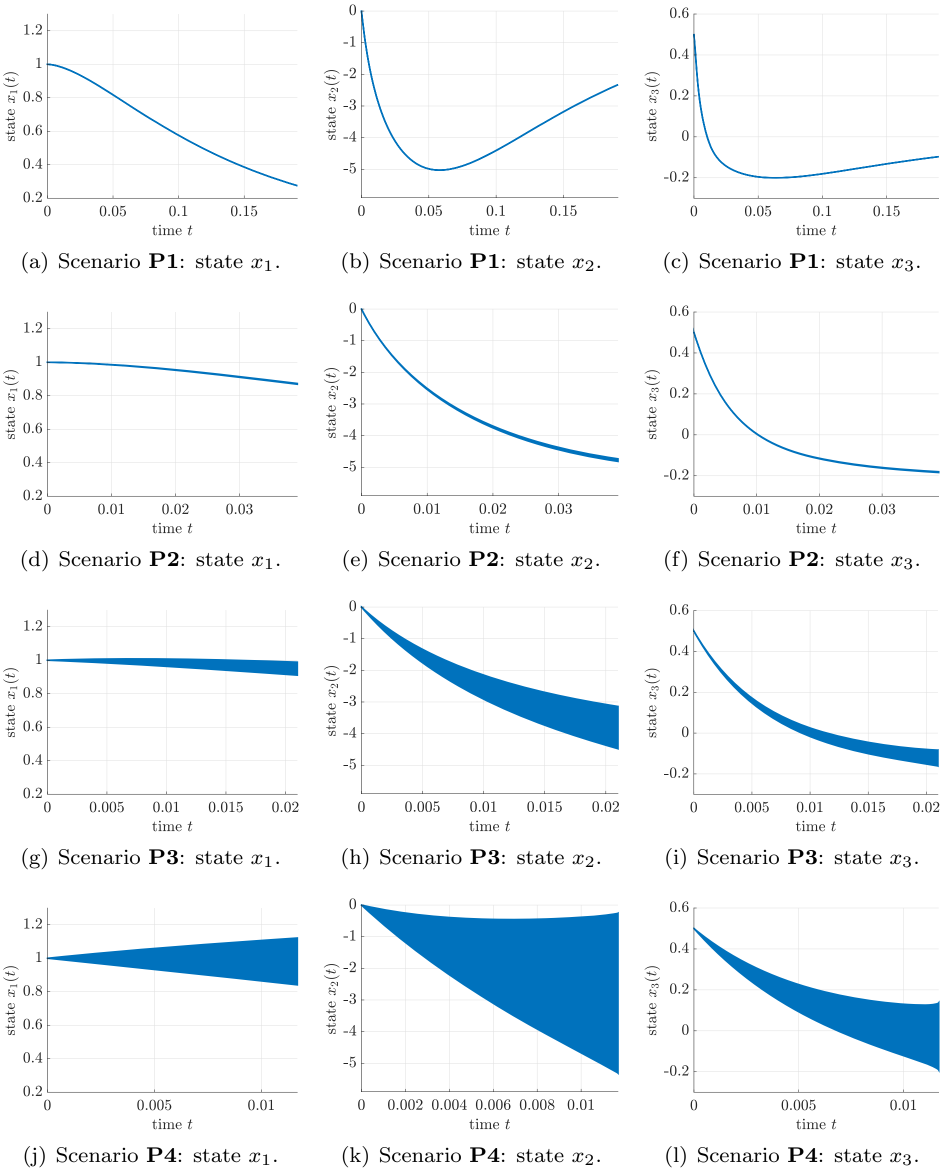
In all simulations, it can be seen from the figures that the computed state enclosures are tight for the cases **P1** and **P2**. In contrast, the computed bounds start to inflate in **P4** for at least one of the state variables. To analyze this phenomenon with the help of suitable Lyapunov-Krasovskii functionals [8] is our future work. If the overall dynamics can be proven to be stable despite uncertain parameters and non-zero delay, Lyapunov-Krasovskii functionals can assist in excluding parts of the state enclosures that certainly do not belong to the reachable domains. This strategy can be interpreted as a generalization of the interval-based Lyapunov function technique presented in [12]. Moreover, such kind of analysis might help to distinguish the reasons for a blow-up of the computed enclosures. Possible causes are

(a) the inflation due to excessively large discretization step sizes Δt,

(b) the inflation due to the wrapping effect that can be countered by splitting parameter intervals or performing a change of coordinates, or

(c) the inflation of the bounds due to a destabilization of the system dynamics due to a large delay in the feedback control law (40).

Note that a point-valued simulation of the system considered in this section shows that the controlled model with the matrix $\sup\left(\mathbf{A}\left([p_1], [p_2]\right)\right)$ in **P4** is unstable for $\tau^* \approx 0.103$ (and also further increased delays) which is only slightly larger than the delay considered in Fig. 7. This observation explains the rapid blow-up of the state enclosures in Figs. 7(j) and 7(k).



(a) Scenario **P1**: state x_1.

(b) Scenario **P1**: state x_2.

(c) Scenario **P1**: state x_3.

(d) Scenario **P2**: state x_1.

(e) Scenario **P2**: state x_2.

(f) Scenario **P2**: state x_3.

(g) Scenario **P3**: state x_1.

(h) Scenario **P3**: state x_2.

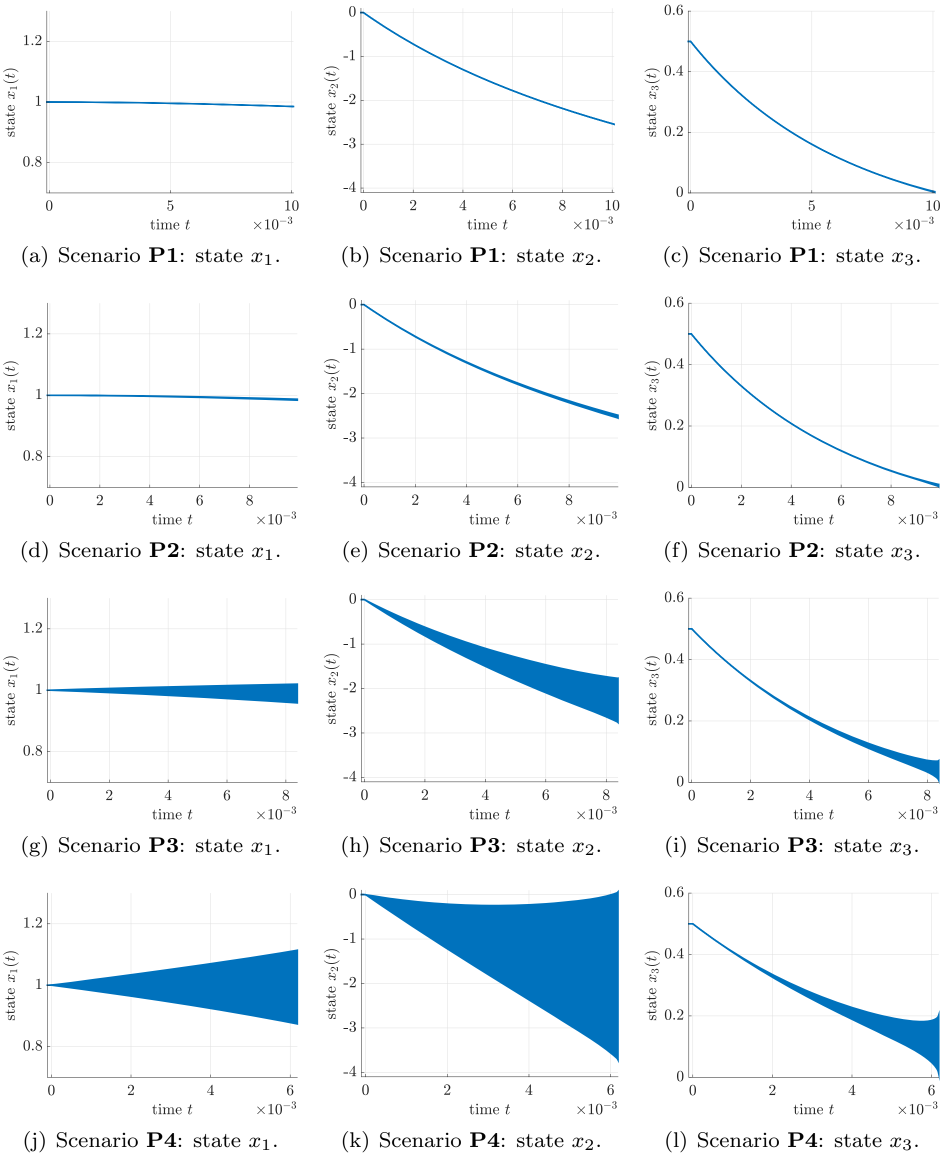
(i) Scenario **P3**: state x_3.

(j) Scenario **P4**: state x_1.

(k) Scenario **P4**: state x_2.

(l) Scenario **P4**: state x_3.

Figure 5: Simulation of the system model (39) for $\tau^* = 0$.



(a) Scenario **P1**: state x_1.

(b) Scenario **P1**: state x_2.

(c) Scenario **P1**: state x_3.

(d) Scenario **P2**: state x_1.

(e) Scenario **P2**: state x_2.

(f) Scenario **P2**: state x_3.

(g) Scenario **P3**: state x_1.

(h) Scenario **P3**: state x_2.

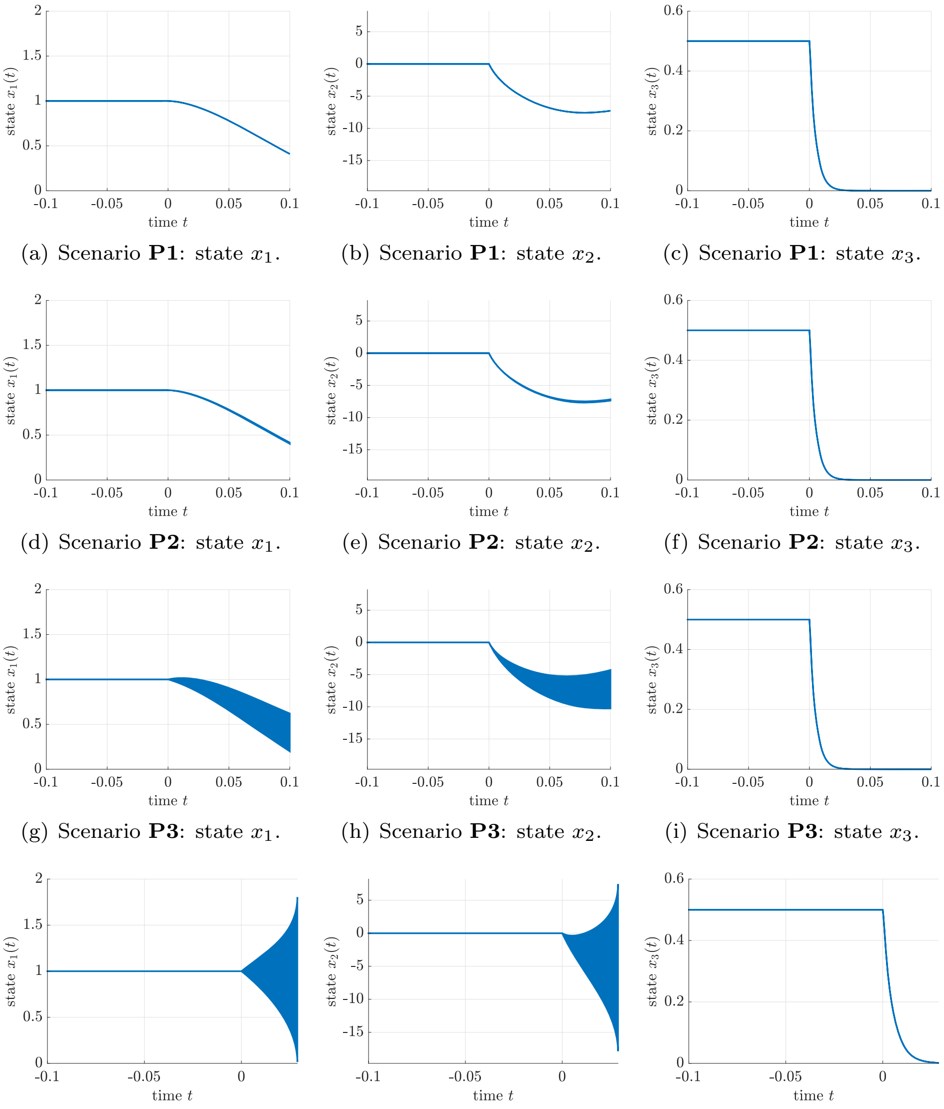
(i) Scenario **P3**: state x_3.

(j) Scenario **P4**: state x_1.

(k) Scenario **P4**: state x_2.

(l) Scenario **P4**: state x_3.

Figure 6: Simulation of the system model (39) for $\tau^* = 10^{-4}$.



(a) Scenario **P1**: state x_1.

(b) Scenario **P1**: state x_2.

(c) Scenario **P1**: state x_3.

(d) Scenario **P2**: state x_1.

(e) Scenario **P2**: state x_2.

(f) Scenario **P2**: state x_3.

(g) Scenario **P3**: state x_1.

(h) Scenario **P3**: state x_2.

(i) Scenario **P3**: state x_3.

(j) Scenario **P4**: state x_1.

(k) Scenario **P4**: state x_2.

(l) Scenario **P4**: state x_3.

Figure 7: Simulation of the system model (39) for $\tau^* = 0.1$.

In conclusion, we would like to point out the following fact. In many control engineering applications, it is possible to intersect the computed bounds with information from measurements at discrete time instants. In practice, this additional information allows us to work even with wide parameter bounds to forecast the domains of reachable states in a computationally cheap manner by providing simple enclosures between two distinct measurement points. This well-known predictor–corrector concept can be implemented using the approach suggested in this paper even within real-time capable state estimation and (model-predictive) control frameworks. The predictor–corrector idea with continuous dynamics and discrete-time measurements is based on the so-called hybrid Kalman filter for systems with stochastic uncertainty [34].

7 Conclusions and Future Work

In this paper, a novel interval-based solution method for certain classes of delay equations was presented. It extends an exponential enclosure technique that was originally developed for delay-free systems of ordinary differential equations.

As future work, we plan to extend the exponential enclosure technique to fractional-order differential equations by replacing the exponential terms with so-called Mittag-Leffler functions [23, 24]. Fractional-order models have a large practical relevance in the context of control and state estimation of electrochemical energy converters and storage elements such as fuel cells and batteries. Moreover, extensions of the proposed technique for solving delay differential equations to systems including the value zero in the set of reachable states will be investigated. Our goal will be to find non-trivial alternatives based on the complex-valued iteration technique published in [29] with the focus on osciallatory dynamics.

References

[1] Arcara, P. and Melchiorri, C. Control schemes for teleoperation with time delay: A comparative study. *Robotics and Autonomous Systems*, 38(1):49–64, 2002. DOI: `10.1016/S0921-8890(01)00164-6`.

[2] Auer, E., Rauh, A., and Kersten, J. Experiments-based parameter identification on the GPU for cooperative systems. *Journal of Computational and Applied Mathematics*, 371:112657, 2019. DOI: `10.1016/j.cam.2019.112657`.

[3] Cooke, K.L. Stability of delay differential equations with applications in biology and medicine. In Capasso, V., Grosso, E., and Paveri-Fontana, S.L., editors, *Mathematics in Biology and Medicine*, pages 439–446, Berlin, Heidelberg, 1985. Springer. DOI: `10.1007/978-3-642-93287-8_59`.

[4] Csendes, T. and Ratz, D. Subdivision direction selection in interval methods for global optimization. *SIAM Journal on Numerical Analysis*, 34(3):922–938, 1997. DOI: `10.1137/S0036142995281528`.

[5] Driver, R.D. *Ordinary and Delay Differential Equations*. Springer–Verlag, New York, 1977. DOI: `10.1007/978-1-4684-9467-9`.

[6] Erneux, T. *Applied Delay Differential Equations*. Springer–Verlag, New York, 2009. DOI: `10.1007/978-0-387-74372-1`.

[7] Fridman, E. *Introduction to Time-Delay Systems: Analysis and Control*. Birkhäuser, Basel, 2014. DOI: `10.1007/978-3-319-09393-2`.

[8] Fridman, E. Tutorial on Lyapunov-based methods for time-delay systems. *European Journal of Control*, 20(6):271–283, 2014. DOI: `10.1016/j.ejcon.2014.10.001`.

[9] Gopalsamy, K. *Stability and Oscillations in Delay Differential Equations of Population Dynamics*. Springer–Verlag, Dordrecht, 1992.

[10] Groothedde, C. and Mireles James, J. Parameterization method for unstable manifolds of delay differential equations. *Journal of Computational Dynamics*, 4(1&2):21–70, 2017. DOI: `10.3934/jcd.2017002`.

[11] Jaulin, L., Kieffer, M., Didrit, O., and Walter, É. *Applied Interval Analysis*. Springer–Verlag, London, 2001. DOI: `10.1007/978-1-4471-0249-6`.

[12] Kersten, J., Rauh, A., and Aschemann, H. Interval methods for robust gain scheduling controllers: An LMI-based approach. *Granular Computing*, 5:203–216, 2020. DOI: `10.1007/s41066-018-00147-1`.

[13] Kyrychko, Y. and Hogan, S. On the use of delay equations in engineering applications. *Journal of Vibration and Control*, 16(7–8):943–960, 2010. DOI: `10.1177/1077546309341100`.

[14] Lin, Y. and Stadtherr, M.A. Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57(10):1145–1162, 2007. DOI: `10.1016/j.apnum.2006.10.006`.

[15] Lohner, R. Enclosing the solutions of ordinary initial and boundary value problems. In Kaucher, E.W., Kulisch, U.W., and Ullrich, C., editors, *Computer Arithmetic: Scientific Computation and Programming Languages*, pages 255–286, Stuttgart, 1987. Wiley-Teubner Series in Computer Science.

[16] Lohner, R. On the ubiquity of the wrapping effect in the computation of the error bounds. In Kulisch, U., Lohner, R., and Facius, A., editors, *Perspectives on Enclosure Methods*, pages 201–217, Wien, New York, 2001. Springer–Verlag. DOI: `10.1007/978-3-7091-6282-8_12`.

[17] Martins, A., Laranjeira, P., Dias, M., and Lopes, J.C. Mass transport modelling in porous media using delay differential equations. *Defect and Diffusion Forum*, 258:586–591, 2006. DOI: `10.4028/www.scientific.net/DDF.258-260.586`.

[18] Michiels, W. and Niculescu, S.I. *Stability and Stabilization of Time-delay Systems: An Eigenvalue-based Approach.* Advances in Design and Control. SIAM, Philadelphia, PA, 2007. DOI: `10.1137/1.9780898718645`.

[19] Moore, R.E., Kearfott, R.B., and Cloud, M.J. *Introduction to Interval Analysis.* SIAM, Philadelphia, 2009. DOI: `10.1137/1.9780898717716`.

[20] Nedialkov, N.S. Interval tools for ODEs and DAEs. In *CD-Proceedings of the 12th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics SCAN 2006*, Duisburg, Germany, 2007. IEEE Computer Society. DOI: `10.1109/SCAN.2006.28`.

[21] Olgac, N., Jenkin, R., and Zalluhoglu, U. A practical approach to the complete stability of TDS with multiple identical imaginary roots. In *Proceedings of the 18th European Control Conference ECC 2020*, St. Petersburg, Russia, 2020. DOI: `10.23919/ECC51009.2020.9143924`.

[22] Rauh, A. *Sensitivity Methods for Analysis and Design of Dynamic Systems with Applications in Control Engineering.* Shaker–Verlag, 2017. DOI: `10.2370/9783844054989`.

[23] Rauh, A. and Jaulin, L. Novel techniques for a verified simulation of fractional-order differential equations. *Fractal and Fractional*, 5(1):17, 2021. DOI: `10.3390/fractalfract5010017`.

[24] Rauh, A. and Kersten, J. Toward the development of iteration procedures for the interval-based simulation of fractional-order systems. *Acta Cybernetica*, 24(3):343–372, 2020. DOI: `10.14232/actacyb.285660`.

[25] Rauh, A., Kersten, J., and Aschemann, H. Techniques for verified reachability analysis of quasi-linear continuous-time systems. In *Proceedings of the 24th International Conference on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 2019. DOI: `10.1109/MMAR.2019.8864648`.

[26] Rauh, A. and Romig, S. Linear matrix inequalities for an iterative solution of robust output feedback control of systems with bounded and stochastic uncertainty. *Sensors*, 21(9):3285, 2021. DOI: `10.3390/s21093285`.

[27] Rauh, A., Romig, S., and Aschemann, H. When is naive low-pass filtering of noisy measurements counter-productive for the dynamics of controlled systems? In *Proceedings of the 23rd IEEE International Conference on Methods and Models in Automation and Robotics MMAR 2018*, Miedzyzdroje, Poland, 2018. DOI: `10.1109/MMAR.2018.8486099`.

[28] Rauh, A., Westphal, R., and Aschemann, H. Verified Simulation of Control Systems with Interval Parameters Using an Exponential State Enclosure Technique. In *CD-Proceedings of the IEEE International Conference on Methods and Models in Automation and Robotics MMAR*, Miedzyzdroje, Poland, 2013. DOI: `10.1109/MMAR.2013.6669913`.

[29] Rauh, A., Westphal, R., Aschemann, H., and Auer, E. Exponential enclosure techniques for initial value problems with multiple conjugate complex eigenvalues. In Nehmeier, M., Wolff von Gudenberg, J., and Tucker, W., editors, *Scientific Computing, Computer Arithmetic, and Validated Numerics*, pages 247–256, Cham, 2016. Springer International Publishing. DOI: `10.1007/978-3-319-31769-4_20`.

[30] Roussel, Marc R. *Nonlinear Dynamics: A Hands-On Introductory Survey.* Morgan & Claypool Publishers, 2019. DOI: `10.1088/2053-2571/ab0281`.

[31] Rump, S.M. IntLab — INTerval LABoratory. In Csendes, T., editor, *Developments in Reliable Computing*, pages 77–104. Kluver Academic Publishers, 1999. DOI: `10.1007/978-94-017-1247-7_7`.

[32] Seuret, A., Hetel, L., Daafouz, J., and Johansson, K.H. *Delays and Networked Control Systems.* Advances in Delays and Dynamics. Springer International Publishing, Switzerland, 2016. DOI: `10.1007/978-3-319-32372-5`.

[33] Shampine, L.F. and Thompson, S. Numerical solution of delay differential equations. In Gilsinn, D.E., Kalmár-Nagy, T., and Balachandran, B., editors, *Delay Differential Equations: Recent Advances and New Directions*, pages 245–271. Springer–Verlag, Boston, MA, 2009. DOI: `10.1007/978-0-387-85595-0_9`.

[34] Stengel, R. *Optimal Control and Estimation.* Dover Publications, Inc., 1994.

[35] Szczelina, R. and Zgliczynski, P. Algorithm for rigorous integration of delay differential equations and the computer-assisted proof of periodic orbits in the Mackey–Glass equation. *Foundations of Computational Mathematics*, 18:1299–1332, 2018. DOI: `10.1007/s10208-017-9369-5`.

[36] van den Berg, J.B., Groothedde, C., and Lessard, J.-P. A general method for computer-assisted proofs of periodic solutions in delay differential problems. *Journal of Dynamics and Differential Equations*, pages 1–44, 2020. DOI: `10.1007/s10884-020-09908-6`.

[37] Wright, E.M. The Non-Linear Difference-Differential Equation. *The Quarterly Journal of Mathematics*, os-17(1):245–252, 1946. DOI: `10.1093/qmath/os-17.1.245`.

Received 29th October 2020