Adaptive Reachability Algorithms for Nonlinear Systems Using Abstraction Error Analysis

Mark Wetzlinger*, Adrian Kulmburg, Alexis Le Penven, Matthias Althoff

Department of Informatics
Technical University of Munich, Boltzmannstr. 3, 85748 Garching b. München, Germany.

Abstract

In many reachability algorithms for nonlinear ordinary differential equations (ODEs), the tightness of the computed reachable sets mainly depends on abstraction errors and the choice of the set representation. One has to mitigate the resulting wrapping effects by suitable tuning of internally-used algorithm parameters since there exists no wrapping-free algorithm to date. In this work, we investigate the fundamentals governing the abstraction error in reachability algorithms—which we also refer to as set-based solvers—and its dependence on the time step size, leading to the introduction of a *gain order*. This order is related to measures for local and global abstraction errors and thus relates the well-known concept of convergence order from classical ODE solvers to set-based solvers. Furthermore, the simplification of the set representation is tackled by limiting the Hausdorff distance between the original and reduced sets; we demonstrate this for zonotopes. Both these theoretical advancements are incorporated in a modular adaptive parameter tuning algorithm suited for multiple classes of nonlinear ODEs whose efficiency is demonstrated on a wide range of benchmarks.

Keywords: convergence order, gain order, parameter tuning, zonotope order reduction, Hausdorff distance, reachability analysis, nonlinear systems,

---

*Corresponding author

Email addresses: `m.wetzlinger@tum.de` (Mark Wetzlinger), `adrian.kulmburg@tum.de` (Adrian Kulmburg), `alexis.le-penven@polytechnique.edu` (Alexis Le Penven), `althoff@tum.de` (Matthias Althoff)

differential-algebraic equations.

1. Introduction

Reachability analysis is a formal verification method for mixed discrete/continuous systems under the influence of uncertainty in the initial states, inputs, and parameters, offering provable guarantees with respect to unsafe states. The computation of exact reachable sets is only possible for a limited number of system classes [1], thus most algorithms compute over-approximative reachable sets in order to retain formal correctness. Despite the design of algorithms for a multitude of system classes and the exploitation of block structures or subspace behavior—all of which enhance the applicability of reachability analysis—the performance of these algorithms still depends heavily on the correct tuning of algorithm parameters. Poor tuning may result in large over-approximations and can ultimately lead to spurious counterexamples, where a given safety property cannot be verified even though a tighter over-approximation would be able to do so. Therefore, the automated tuning of algorithm parameters constitutes a crucial next step in the advancement of reachability algorithms, enabling non-experts and practitioners to apply reachability analysis in their respective fields. In this article, we address this demand by proposing an automated parameter tuning approach for state-space-abstracted reachability analysis of nonlinear ordinary differential equations and nonlinear differential-algebraic equations. An integral part is the tuning of the time step size for which we thoroughly investigate its effect on the error caused by abstracting the system dynamics within the reachability analysis—an error that is well-studied for classical ODE solvers but has not yet been examined in detail for reachability analysis.

Related work. There exist several approaches to compute reachable sets of nonlinear systems: By definition, any invariant set containing the initial set is also a reachable set, thus approaches for invariant generation [2, 3, 4] can be used for reachability analysis although the result may be unnecessarily conservative. One can also obtain reachable sets by solving a reformulation of the original problem to a Hamilton-Jacobi equation [5, 6], whose solution scales exponentially with the system dimension. Another option is to obtain reachable sets from validated simulations with the help of annotations, i.e., additional information about the analyzed system [7], which also scales exponentially as one has to cover the initial set by enough initial states. Current state-of-the-art approaches for reachability analysis either abstract the solution space or the state space: For solution space abstraction, the Picard iteration is lifted to set-based analysis by representing reachable sets using Taylor models [8, 9, 10]. In state space abstraction, the system dynamics are abstracted by a Taylor series and its corresponding Lagrange remainder to obtain a differential inclusion, ranging from hybridization approaches [11, 12, 13, 14] to on-the-fly linearizations [15, 16, 17] or polynomial abstractions [18]. The algorithmic differences between this group and approaches based on solution space abstraction are extensive and impede a joint parameter tuning framework for both groups. Hence, we will restrict ourselves to approaches based on state space abstraction in this work. Nonetheless, some of the presented ideas may be beneficial in the pursuit of similar automated tuning methods for any of the abovementioned methods. Many of the presented approaches are implemented in specialized reachability tools, namely Ariadne [19], C2E2 [20], CORA [21], DynIBEX [22], Flow* [23], Isabelle/HOL [24], and JuliaReach [25].

Reachability algorithms require a suitable set representation balancing an accurate representation of the reachable sets with computational efficiency of the set operations. The main trade-off occurs between the closedness of operations and the growth of the set representation size: As an example, the set representation size of ellipsoids is fixed, but they are not closed under the Minkowski sum, whereas for zonotopes an exact result can be obtained at the cost of increasing the representation size. This creates the need for so-called order reduction methods [26, Sec. 3.4], [27, Sec. 3.2]; comparisons of these methods are presented in [28, 29]. Essentially, two types of approaches exist: The majority of methods computes an over-approximation that is as tight as possible for a desired (lower) user-specified order. The alternative approach in [30, Thm. 3.2] reduces the order as much as possible for a given bound of the induced over-approximation.

Many reachability algorithms depend on algorithm parameters that can be tuned to achieve tighter approximations, at the cost of a longer runtime. Some algorithms return tighter results than others for the same runtime, which offers one way of comparison and allows for a categorization of classical ODE solvers using the concept of consistency order [31, 32]. For set-based reachability analysis, an estimation for the accuracy and convergence of certain algorithms has been performed in [33].

An open research question is how to automatically tune algorithm parameters for reachability analysis. This problem has been extensively studied for classical ODE solvers, which led to a wide variety of different methods and thorough investigations of stability and convergence [34]. Great effort has also been devoted to the automated tuning of the time step size [35, 36], resulting in powerful solvers, which are ubiquitously applied in research and industry alike. As these solvers only compute approximations to the exact solution, a next step was to enclose a single trajectory, for which guaranteed integration methods provide several automated time step size control strategies [37, 38, 39].

In reachability analysis, automated techniques are scarce due to the presence of uncertainty in the initial state, input, and model parameters. This severely complicates matters as the tuning of algorithm parameters does not only comprise the time step size, but also effects related to a number of other algorithm parameters, such as the set representation as well as any emerging interdependency. For linear systems, there are approaches approximating the actual flow within a user-defined error bound [40], or adapting the time step size in each step in order to satisfy a linearly increasing, user-defined error bound [41]. More comprehensive approaches [42, 43] adapt all algorithm parameters using over-approximation measures related to the Hausdorff distance to enable users to tune the desired accuracy. For nonlinear systems, adaptive methods have been explored in [44], where the time step size is tuned within a user-defined range. Another method [45] proposes iterative recomputations of the reachable set from scratch, while refining the parameter values in discrete steps in between runs. The work in [30] presents the first fully automatic reachability algorithm for nonlinear systems, which not only optimizes the time step size, but also other algorithm parameters such as the representation size.

Contributions. Our work is based on the adaptive parameter tuning approach in [30], which is significantly enhanced in the following three ways:

1. We extend the zonotope order reduction method introduced in [30] by two additional bounds for the Hausdorff distance between a zonotope and its box over-approximation. .

2. While the analysis of the abstraction error in [30] was restricted to parameter tuning, we now dive a lot deeper into this topic: In order to lift for the first time convergence orders of classical solvers to reachability algorithms, we rigorously introduce a novel concept called *gain order* which offers a similar yet more accurate description of the influence of the time step size on local and global abstraction errors.

3. Our tuning methods are no longer restricted to only nonlinear continuous-time systems as in [30], but can now also be applied to systems with algebraic constraints, parametric uncertainties, and in discrete time.

This article is structured as follows: A summary of reachability analysis for multiple nonlinear system classes is presented in Sec. 2. In Sec. 3, several bounds on the Hausdorff distance between a zonotope and its reduced counterpart are proven. Next in Sec. 4, we thoroughly analyze the abstraction error of the reachability algorithm leading to the introduction of the *gain order* which translates the concept of convergence order known from numerical ODE theory to set-based solvers. Based on these theoretic novelties, the tuning modules that constitute our adaptive parameter tuning approach are described in Sec. 5. Finally, the evaluation of numerical examples in Sec. 6 demonstrates the practical usability of our tuning methods for a variety of nonlinear system classes, followed by concluding remarks in Sec. 7.

2. Preliminaries

In this section, we recall reachability analysis for nonlinear systems based on abstractions in the state space. This outline is particularly important for the investigation of the abstraction error in Sec. 4 as well as for the adaptive parameter tuning in Sec. 5.

2.1. Notation

We denote vectors by lower-case letters and matrices by upper-case letters. An all-zero vector or matrix of proper dimension is represented by $\mathbf{0}$. For a vector $v \in \mathbb{R}^n$, $|v| \in \mathbb{R}^n$ is the element-wise computed absolute value and v_i returns the i-th entry of v. Analogously, m_{ij} refers to the entry in the i-th row and j-th column of a matrix $M \in \mathbb{R}^{n \times p}$. We denote the concatenation of two matrices M_1 and M_2 by $[M_1 \; M_2]$. The operation $\mathrm{diag}(v)$ returns a matrix with v on its diagonal and otherwise zeros. The identity matrix of proper dimension is denoted by I. Moreover, $\|M\|_\infty$ refers to the matrix norm of M induced by the infinity norm. All sets are represented by calligraphic upper-case letters: We write $\mathcal{B} = [a, b] \subset \mathbb{R}^n$, where $\forall i \in \{1, ..., n\} : a_i \leq b_i$, to denote an n-dimensional axis-aligned box. Its diameter is defined by $d\big(\mathcal{B}\big) := b - a \in \mathbb{R}^n$ and the absolute value by $\mathtt{abs}\big(\mathcal{B}\big) := [-c, c] \subset \mathbb{R}^n$, where $c = \max\{|a|, |b|\}$ is evaluated element-wise [46, eq. (10)]. Furthermore, we abbreviate the Cartesian product of identical lower and upper limits for n consecutive dimensions by $[a, b]^n$. We use upper-case boldface letters to represent interval matrices $\mathbf{I} = [P, Q] \in \mathbb{R}^{m \times n}$, where $\forall i \in \{1, ..., m\}, \forall j \in \{1, ..., n\} : p_{ij} \leq q_{ij}$. For operations on sets, we use $\oplus$ for the Minkowski sum, $\ominus$ for the Minkowski difference with $\mathcal{S}_1 \ominus \mathcal{S}_2 = \{s | s + \mathcal{S}_2 \subseteq \mathcal{S}_1\}$, and introduce the operator $\boxplus$ representing either the Minkowski sum or the exact addition as defined in [47, Prop. 10]; additionally, we define the operators $\mathtt{center}(\mathcal{S})$, $\mathtt{box}\big(\mathcal{S}\big)$, and $\mathtt{vol}\big(\mathcal{S}\big)$, which respectively return the geometric center, the tightest axis-aligned box over-approximation, and the volume of a set $\mathcal{S} \subset \mathbb{R}^n$. The radius of a set is defined as $\mathtt{rad}\big(\mathcal{S}\big) := 0.5 \big\| d\big(\mathtt{box}\big(\mathcal{S}\big)\big) \big\|_2$. Additionally, $\mathcal{S}_i = e_i^\top \mathcal{S}$, with e_i being the i-th basis vector, denotes the projection of $\mathcal{S}$ onto the i-th axis. We also write $\texttt{conv}\big(\mathcal{S}_1, \mathcal{S}_2\big)$ for the convex hull of two sets $\mathcal{S}_1, \mathcal{S}_2 \subset \mathbb{R}^n$. The floor operator $\lfloor k \rfloor$ returns the next smaller integer number k, the sign function is denoted by $\operatorname{sgn}(\cdot)$, and the Frobenius norm by $\|\cdot\|_F$. The set $\mathbb{N}_0$ denotes the natural numbers including 0.

2.2. Reachability analysis of nonlinear systems using abstractions in the state space

The presented techniques for automated parameter tuning are applied to several classes of nonlinear systems, the most general of which are semi-explicit index-1 differential-algebraic (DA) equations [48], which can be formulated as

$$\begin{aligned} \dot{x}(t) &= f(x(t), y(t), u(t)) \\ 0 &= g(x(t), y(t), u(t)), \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is a sufficiently smooth nonlinear function, $x(t) \in \mathbb{R}^n$ is the state vector, $y(t) \in \mathbb{R}^{n_a}$ is the vector of algebraic variables, and $u(t) \in \mathbb{R}^m$ is the input vector. Omitting the algebraic equation and algebraic variables yields a standard nonlinear ordinary differential equation. Note that these also encompass parametric systems: Each constant parameter can be defined as an additional state with the dynamics $\dot{x}_i(t) = 0$ and each time-varying parameter as an additional uncertain input. Moreover, we will consider discrete-time systems $x_{k+1} = f(x_k, u_k)$.

Let us introduce $\xi_{\text{ex}}(t; x(0), y(0), u(\cdot))$ as the solution of (1) at time t for the initial values $x(0)$ and $y(0)$. Then, the exact reachable set $\mathcal{R}_{\text{ex}}\big([0, t_{\text{end}}]\big)$ of (1) over the time horizon $t \in [0, t_{\text{end}}]$ is given by

$$\begin{aligned} \mathcal{R}_{\text{ex}}\big([0, t_{\text{end}}]\big) = \Big\{ \xi_{\text{ex}}(t; x(0), y(0), u(\cdot)) \,\Big|\, & x(0) \in \mathcal{X}^0,\, y(0) \in \mathcal{Y}^0\,, \\ & t \in [0, t_{\text{end}}], \forall \tau \in [0, t] : u(\tau) \in \mathcal{U} \Big\}\,, \end{aligned} \tag{2}$$

with the initial sets $\mathcal{X}^0 \subset \mathbb{R}^n, \mathcal{Y}^0 \subset \mathbb{R}^{n_a}$ and the input set $\mathcal{U} \subset \mathbb{R}^m$. In this work we use abstractions in the state space, where both $f(\cdot)$ and $g(\cdot)$ are abstracted by a Taylor series of order κ at an expansion point z^* [18, eq. (2)] [48, eq. (8)], so that

$$\begin{aligned} \dot{x}_i(t) &\in \sum_{\nu=0}^{\kappa} \frac{\big((z(t)-z^*)^\top \nabla\big)^\nu f_i(\hat{z})}{\nu!}\bigg|_{\hat{z}=z^*} \oplus \mathcal{L}_i^{(x)}(t) \ , \\ 0 &\in \sum_{\nu=0}^{\kappa} \frac{\big((z(t)-z^*)^\top \nabla\big)^\nu g_i(\hat{z})}{\nu!}\bigg|_{\hat{z}=z^*} \oplus \mathcal{L}_i^{(y)}(t) \ , \end{aligned} \tag{3}$$

using the extended vector $z = [x^\top y^\top u^\top]^\top \in \mathbb{R}^{n+n_a+m}$ and the Nabla operator $\nabla = \sum_{i=1}^{n+n_a+m} e_i \frac{\partial}{\partial z_i}$. The Lagrange remainders $\mathcal{L}_i^{(x)}$ and $\mathcal{L}_i^{(y)}$ are defined by [18, eq. (2)] [48, eq. (8)]

$$\begin{aligned} \mathcal{L}_i^{(x)}(t) &:= \left\{ \frac{\big((z(t)-z^*)^\top \nabla\big)^{\kappa+1} f_i(\hat{z})}{(\kappa+1)!} \,\middle|\, \hat{z} = z^* + \alpha(z(t)-z^*), \alpha \in [0,1] \right\} , \\ \mathcal{L}_i^{(y)}(t) &:= \left\{ \frac{\big((z(t)-z^*)^\top \nabla\big)^{\kappa+1} g_i(\hat{z})}{(\kappa+1)!} \,\middle|\, \hat{z} = z^* + \alpha(z(t)-z^*), \alpha \in [0,1] \right\} , \end{aligned} \tag{4}$$

and evaluated using range-bounding techniques such as interval arithmetic [49].

The time horizon $[0, t_{\text{end}}]$ is divided into time intervals $\tau_k = [t_k, t_{k+1}]$ with the individual time step sizes $\Delta t_k = t_{k+1} - t_k > 0$ summing up to t_{end}. The reachable set for the entire time horizon is obtained by unifying the sequence of time-interval reachable sets $\mathcal{R}(\tau_k)$. For notational simplicity, we introduce an equivalent notation for the first terms in (3),

$$\begin{aligned} w_i^{(x)} &= f_i(z^*),\, C_{ij}^{(x)} = \frac{\partial f_i(\hat{z})}{\partial \hat{z}_j}\bigg|_{\hat{z}=z^*},\, D_{ijk}^{(x)} = \frac{\partial^2 f_i(\hat{z})}{\partial \hat{z}_j \partial \hat{z}_k}\bigg|_{\hat{z}=z^*}, \dots \\ w_i^{(y)} &= g_i(z^*),\, C_{ij}^{(y)} = \frac{\partial g_i(\hat{z})}{\partial \hat{z}_j}\bigg|_{\hat{z}=z^*},\, D_{ijk}^{(y)} = \frac{\partial^2 g_i(\hat{z})}{\partial \hat{z}_j \partial \hat{z}_k}\bigg|_{\hat{z}=z^*}, \dots . \end{aligned} \tag{5}$$

Alg. 1 summarizes the reachability algorithm for nonlinear DA systems featured on-the-fly methods, such as the ones in [17, 18], which extends to hybridization approaches [12, 16, 14] with minor adaptations. After its presentation, we will discuss the simplifications that can be made for the other system classes mentioned at the beginning of this overview.

At the start of each step k (Line 4), the operation `taylor` evaluates both Taylor series at the linearization point z^* returned by the operation `linPoint` (Line 3):

$$\begin{aligned} x^* &= \texttt{center}(\mathcal{R}(t_k)) + \frac{1}{2}\,\Delta t_k\, f\big(\texttt{center}(\mathcal{R}(t_k))\big), \\ u^* &= \texttt{center}(\mathcal{U}), \quad \text{and} \quad y^* \leftarrow 0 = g(x^*, y^*, u^*), \end{aligned} \tag{6}$$

Algorithm 1 Reachability analysis of continuous-time nonlinear systems using abstractions in the state space.

Input: nonlinear function $f(z)$, time horizon t_{end}, initial set $\mathcal{R}(t_0) = \mathcal{X}^0$, input set $\mathcal{U}$; *only DA*: algebraic equation $g(z)$, initial algebraic set $\mathcal{R}^y(t_0) = \mathcal{Y}^0$
Output: reachable set $\mathcal{R}([0, t_{\mathrm{end}}])$

```
1:  $k = 0, t_k = 0$
2:  **while** $t_k < t_{\mathrm{end}}$ **do**
3:      $z^*(t_k) \leftarrow \texttt{linPoint}(\mathcal{R}(t_k), f, \mathcal{R}^y(t_k), g)$
4:      $w^{(x)}, w^{(y)}, C^{(x)}, C^{(y)}, \ldots \leftarrow \texttt{taylor}\big(f(z), z^*(t_k)\big)$
5:      $w, A, B \leftarrow \texttt{linSys}(w^{(x)}, w^{(y)}, C^{(x)}, C^{(y)})$
6:      $\mathcal{R}_{\mathrm{lin}}(t_{k+1}), \mathcal{R}_{\mathrm{lin}}(\tau_{k+1}) \leftarrow \texttt{linReach}(\mathcal{R}(t_k), w, A, B)$
7:      $\overline{\Psi} = \mathbf{0}$
8:      **do**
9:          $\overline{\Psi} \leftarrow \texttt{enlarge}(\overline{\Psi})$
10:         $\Psi, \mathcal{R}^y(t_{k+1}) \leftarrow \texttt{abstrErr}(\mathcal{R}_{\mathrm{lin}}(\tau_{k+1}), \overline{\Psi}, \kappa)$
11:     **while** $\Psi \not\subseteq \overline{\Psi}$
12:     $\mathcal{R}_{\mathrm{abs}} \leftarrow \texttt{abstrSol}(\Psi)$
13:     $\mathcal{R}(t_{k+1}) = \mathcal{R}_{\mathrm{lin}}(t_{k+1}) \boxplus \mathcal{R}_{\mathrm{abs}}$
14:     $\mathcal{R}(\tau_{k+1}) = \mathcal{R}_{\mathrm{lin}}(\tau_{k+1}) \boxplus \mathcal{R}_{\mathrm{abs}}$
15:     $\mathcal{R}(t_{k+1}) \leftarrow \texttt{red}\big(\mathcal{R}(t_{k+1})\big), \mathcal{R}(\tau_{k+1}) \leftarrow \texttt{red}\big(\mathcal{R}(\tau_{k+1})\big)$
16:     $t_{k+1} \leftarrow t_k + \Delta t_k,\ k \leftarrow k + 1$
17: **end while**
18: $\mathcal{R}([0, t_{\mathrm{end}}]) = \bigcup_{j=0}^{k-1} \mathcal{R}(\tau_j)$
```

where $y^* \in \mathcal{R}^y(t_k)$ is obtained by solving the algebraic equation using a Newton-Raphson algorithm [48, Sec. IV-A]. Next, we abstract the nonlinear system by a differential inclusion

$$\dot{x}(t) \in \underbrace{Ax(t) + Bu(t) + w}_{f_{\mathrm{lin}}(t)} \boxplus \Psi, \tag{7}$$

using the linearized vector field f_{lin} and an uncertainty set Ψ enclosing all higher-order terms including the Lagrange remainder. The constant offset w, the state matrix $A \in \mathbb{R}^{n\times n}$, and the input matrix $B \in \mathbb{R}^{n\times m}$ are returned by the operation `linSys` [48, Sec. IV]:

$$w = w^{(x)} - \tilde{Y}\tilde{Z}^{-1}w^{(y)},\ A = \tilde{A} - \tilde{Y}\tilde{Z}^{-1}\tilde{V},\ B = \tilde{B} - \tilde{Y}\tilde{Z}^{-1}\tilde{W},$$
$$\text{using } C^{(x)} = [\tilde{A}\ \tilde{Y}\ \tilde{B}],\ C^{(y)} = [\tilde{V}\ \tilde{Z}\ \tilde{W}],$$

where $\tilde{A} \in \mathbb{R}^{n\times n}, \tilde{Y} \in \mathbb{R}^{n\times n_a}, \tilde{B} \in \mathbb{R}^{n\times m}, \tilde{V} \in \mathbb{R}^{n_a\times n}, \tilde{Z} \in \mathbb{R}^{n_a\times n_a}, \tilde{W} \in \mathbb{R}^{n_a\times m}$. The reformulation in (7) allows us to apply the superposition principle for linear systems and separate the computation of the next reachable set $\mathcal{R}(t_{k+1})$ into two parts: First, the reachable set $\mathcal{R}_{\text{lin}}$ of the linearized dynamics (Lines 4-6); second, the set of abstraction errors $\mathcal{R}_{\text{abs}}$ based on the abstraction error Ψ [48, eq. (15)] (Lines 7-12). Let us briefly highlight an important difference between two groups of algorithms: Linearization algorithms [17] may also use polynomial abstractions ($\kappa \geq 2$) in (3), but the evaluation of all higher-order terms until the Lagrange remainder loses the state correlation with respect to the linear dynamics which results in an essentially linear approximation of $f(x)$ despite the polynomial abstraction. In contrast, polynomialization algorithms [18] retain the state correlation in the evaluation of all time-constant, higher-order ($\kappa \geq 2$) terms in (3) allowing for a truly polynomial abstraction. For simplicity, we will restrict the remainder of this overview to linearization algorithms; the extension to polynomialization algorithms can be found in [18].

The operation `linReach` (Line 6) returns the reachable set $\mathcal{R}_{\text{lin}}$ using a reachability algorithm for the linearized dynamics $Ax(t) + Bu(t) + w$, e.g., [50, Sec. 3.2]. The computation of the abstraction error Ψ requires to resolve the mutual dependency between Ψ and $\mathcal{R}_{\text{lin}}$. To this end, we initially estimate Ψ to be $\overline{\Psi}$ (Line 7) and use the operation `enlarge` to enlarge this set by a constant factor greater than 1 (Line 9) until the containment $\Psi \subseteq \overline{\Psi}$ (Line 8) is ensured. Within this loop, the set Ψ is iteratively computed using the operation `abstrErr` (Line 10), which depends on the correlation of the state x with the abstraction error [17, Prop. 1] [18, Sec. 4.1]. The algebraic reachable set $\mathcal{R}^y(t_{k+1})$ is obtained as a byproduct [48, Prop. 2]. Next, the operation `abstrSol` (Line 12) computes the set of abstraction errors based on Ψ by [17, Sec. VI.]

$$\mathcal{R}_{\text{abs}} = \bigoplus_{p=0}^{\eta_{\text{abs}}} \frac{\Delta t^{p+1}}{(p+1)!} A^p \Psi \oplus \mathbf{E}(\Delta t, \eta_{\text{abs}})\, \Delta t\, \Psi, \tag{8}$$

with the remainder of the exponential matrix [51, Prop. 2]

$$\begin{aligned} \mathbf{E}(\Delta t, \eta) &= [-E(\Delta t, \eta), E(\Delta t, \eta)], \\ E(\Delta t, \eta) &= e^{|A|\Delta t} - \sum_{i=0}^{\eta} \frac{\big(|A|\Delta t\big)^i}{i!}, \end{aligned} \tag{9}$$

where $\mathbf{E} = \mathcal{O}(\Delta t^{\eta+1})$ for $\Delta t \to 0$.

By exploiting the superposition principle in (7), the next reachable set $\mathcal{R}(t_{k+1})$ is obtained by the addition of $\mathcal{R}_{\text{lin}}$ and $\mathcal{R}_{\text{abs}}$ (Lines 13-14). For reasons of computational efficiency, we require to reduce the set representation size using the operation $\texttt{red}(\cdot)$ at the end of each step (Line 15). Then, we obtain the next time-point solution as

$$\mathcal{R}(t_{k+1}) = \texttt{red}\big(\underbrace{e^{A\Delta t_k}\mathcal{R}(t_k) \oplus \mathcal{P}(\tau_k)}_{=:\,\mathcal{R}_{\text{lin}}(t_{k+1})} \boxplus \mathcal{R}_{\text{abs}}\big). \tag{10}$$

For later use, we expand the set $\mathcal{R}_{\text{lin}}$ obtained from the operation `linReach` into its homogeneous and particular solutions, $e^{A\Delta t_k}\mathcal{R}(t_k)$ and $\mathcal{P}(\tau_k)$, respectively. For the time-interval solution, we compute the reachable set of the linearized dynamics $\mathcal{R}_{\text{lin}}(\tau_k)$ using the convex hull of sets at the beginning and end of the time interval and enlarge the result by an error set $\mathbf{F}_x\mathcal{R}(t_k)$ with [50, Prop. 3.1]

$$\mathbf{F}_x = \bigoplus_{p=2}^{\eta_{\text{lin}}} \underbrace{\big[(p^{\frac{-p}{p-1}} - p^{\frac{-1}{p-1}})\Delta t^p, 0\big]\frac{A^p}{p!}}_{:=\,\mathbf{T}^{(p)}} \oplus \mathbf{E}(\Delta t, \eta_{\text{lin}}), \tag{11}$$

which ultimately yields [17, Sec. IV.-A]

$$\mathcal{R}(\tau_{k+1}) = \underbrace{\texttt{conv}\big(\mathcal{R}(t_k), e^{A\Delta t_k}\mathcal{R}(t_k) \oplus \mathcal{P}(\tau_k)\big) \oplus \mathbf{F}_x\mathcal{R}(t_k)}_{=:\,\mathcal{R}_{\text{lin}}(\tau_k)} \boxplus \mathcal{R}_{\text{abs}}(\tau_k), \tag{12}$$

assuming $0 \in \mathcal{U}$, with an extension to arbitrary inputs in [50, Sec. 3.2.2].

Finally, let us briefly highlight the algorithmic simplifications for the other aforementioned system classes: The changes required to accommodate standard ordinary differential equations follow directly by omitting each occurrence of the algebraic equation. For discrete-time systems, we replace (7) by the following difference inclusion

$$x_{k+1} \in \underbrace{Ax_k + Bu_k + w}_{f_{\text{lin}}(x_k,u_k)} \boxplus \Psi. \tag{13}$$

The operation `linReach` (Line 6) therefore becomes the straightforward evaluation of $f_{\text{lin}}(x_k, u_k)$. Next, the process of estimation and containment check (Lines 7-11) is shortened to a single evaluation of the operation `abstrErr` on the start set $\mathcal{R}(t_k)$. The next time-point solution (Line 13) is computed by the addition of $\mathcal{R}_{\text{lin}}$ and Ψ following directly from (13). Naturally, there is neither a computation of $\mathcal{R}_{\text{abs}}$ (Line 12) nor of a time-interval solution (Line 14).

3. Hausdorff Reduction

Reachable sets are often represented by zonotopes because they are closed under linear maps and Minkowski sums, and these operations can be computed efficiently. Over subsequent steps of Alg. 1, the representation size of zonotopes grows, necessitating *order reduction*. In this section, we will provide several bounds so that the representation size of a zonotope can be reduced while satisfying any desired over-approximation error. In comparison with our previous work [30], we derive two more error bounds and also a novel generator selection criterion for the order reduction. Additionally, we provide insights into the performance of each bound for different classes of zonotopes.

Let us first define zonotopes and the order reduction operation:

Definition 1. (Zonotopes) [26, Def. 1] Given a center $c \in \mathbb{R}^n$ and $\gamma \in \mathbb{N}$ generator vectors $G = [g^{(1)} \dots g^{(\gamma)}]$, a zonotope is defined as

$$\mathcal{Z} := \Big\{ x \in \mathbb{R}^n \;\Big|\; x = c + \sum_{i=1}^{\gamma} \alpha_i \, g^{(i)} \, , -1 \leq \alpha_i \leq 1 \Big\}.$$

We also define its order by $\rho := \frac{\gamma}{n}$ and the shorthand $\langle c, G \rangle_Z$. □

For later use, let us also introduce the operations $\texttt{center}(\mathcal{Z}) = c$ and $\texttt{box}\big(\mathcal{Z}\big) = \langle c, \mathrm{diag}(\sum_{i=1}^{\gamma} |g^{(i)}|)\rangle_Z$ returning a box over-approximation of $\mathcal{Z}$. For order reduction, we select the method introduced in [26, Sec. 3.4], which is comprised of the following steps:

1. Split the given zonotope $\mathcal{Z}_{\text{full}}$ into two parts $\mathcal{Z}_{\text{full}} = \mathcal{Z}' \oplus \mathcal{Z}$.
2. Enclose $\mathcal{Z}$ by a tight box $\mathcal{Z}_{\text{box}} = \texttt{box}\big(\mathcal{Z}\big) \supseteq \mathcal{Z}$.
3. Compose the reduced zonotope as $\mathcal{Z}_{\text{red}} = \mathcal{Z}' \oplus \mathcal{Z}_{\text{box}} \supseteq \mathcal{Z}_{\text{full}}$.

To quantitatively measure the error induced by the order reduction, we use the Hausdorff distance:

Definition 2. (Hausdorff distance) For two compact sets $\mathcal{V}, \mathcal{W} \subset \mathbb{R}^n$, the Hausdorff distance is defined as

$$d_H(\mathcal{V}, \mathcal{W}) := \max\big\{d_H^{(1)}(\mathcal{V}, \mathcal{W}), d_H^{(2)}(\mathcal{V}, \mathcal{W})\big\},$$

where

$$\begin{aligned} d_H^{(1)}(\mathcal{V}, \mathcal{W}) &:= \max_{v\in\mathcal{V}} \min_{w\in\mathcal{W}} \|v - w\|_2 \\ d_H^{(2)}(\mathcal{V}, \mathcal{W}) &:= \max_{w\in\mathcal{W}} \min_{v\in\mathcal{V}} \|w - v\|_2. \end{aligned} \qquad \square$$

For subsequent deriviations, let us introduce a short lemma:

Lemma 1. *For the compact sets $\mathcal{S}, \mathcal{V}, \mathcal{W} \subset \mathbb{R}^n$, the following inequality holds:*

$$d_H(\mathcal{S} \oplus \mathcal{V}, \mathcal{S} \oplus \mathcal{W}) \leq d_H(\mathcal{V}, \mathcal{W}).$$

PROOF. We only prove the inequality for $d_H^{(1)}$

$$\begin{aligned} d_H^{(1)}(\mathcal{S} \oplus \mathcal{V}, \mathcal{S} \oplus \mathcal{W}) &= \max_{\substack{v\in\mathcal{V} \\ s^{(1)}\in\mathcal{S}}} \min_{\substack{w\in\mathcal{W} \\ s^{(2)'}\in\mathcal{S}}} \|v + s^{(1)} - w - s^{(2)'}\|_2 \\ &\leq \max_{\substack{v\in\mathcal{V} \\ s^{(1)}\in\mathcal{S}}} \min_{w\in\mathcal{W}} \|v + s^{(1)} - w - s^{(1)}\|_2 \\ &= \max_{v\in\mathcal{V}} \min_{w\in\mathcal{W}} \|v - w\|_2 \\ &= d_H(\mathcal{V}, \mathcal{W}). \end{aligned}$$

The reasoning is analogous for $d_H^{(2)}$. $\square$

In the context of zonotope order reduction, Lemma 1 implies

$$d_H(\mathcal{Z}_{\text{full}}, \mathcal{Z}_{\text{red}}) = d_H(\mathcal{Z}' \oplus \mathcal{Z}, \mathcal{Z}' \oplus \mathcal{Z}_{\text{box}}) \leq d_H(\mathcal{Z}, \mathcal{Z}_{\text{box}}),$$

which lets us restrict our attention to the computation of the Hausdorff distance between the partial zonotope $\mathcal{Z}$ and its box over-approximation $\mathcal{Z}_{\text{box}}$ without loss of generality. Computing the exact Hausdorff distance between two zonotopes is NP-hard in general, since the Hausdorff distance between a point (represented by a zonotope without generators) and an arbitrary zonotope can be reformulated as a *longest vector sum* problem [52][1]; in fact, the work in [52] even shows that this problem is APX-hard. Hence, in practice, we are limited to finding bounds on the Hausdorff distance:

Theorem 1. *Let $\mathcal{Z} = \langle c, G\rangle_Z \subset \mathbb{R}^n$ be a zonotope and $\mathcal{Z}_{box} := \texttt{box}(\mathcal{Z}) = \langle c, G_{box}\rangle_Z \supseteq \mathcal{Z}$ its box over-approximation. Due to the containment $\mathcal{Z} \subseteq \mathcal{Z}_{box}$, the Hausdorff distance d_H is given by*

$$d_H(\mathcal{Z}, \mathcal{Z}_{box}) = \max_{x_{box} \in \mathcal{Z}_{box}} \min_{x \in \mathcal{Z}} \|x_{box} - x\|_2 . \tag{14}$$

This distance is over-approximated by the following three bounds:

$$\omega_{\max}(\mathcal{Z}) := 2 \left\| \sum_{p=1}^{\gamma} \widehat{g}^{(p)} \right\|_2, \tag{15}$$

$$\omega_{rad}(\mathcal{Z}) := \sum_{p=1}^{\gamma} \left\| g^{(p)} \right\|_2, \tag{16}$$

$$\omega_{cut}(\mathcal{Z}) := \sqrt{2} \sum_{p=1}^{\gamma} \|g^{(p)}\|_2 \sqrt{1 - \frac{\sum_{i=1}^{n} \left(g_i^{(p)}\right)^4}{\left\|g^{(p)}\right\|_2^4}}, \tag{17}$$

with

$$\widehat{g}_i^{(p)} = \begin{cases} |g_i^{(p)}|, & \text{if } i \neq i^*, \\ 0, & \text{otherwise,} \end{cases} \tag{18}$$

where i^* is the (first) index for which $g_{i^*}^{(p)} = \left\|g^{(p)}\right\|_\infty$.

[1]This equivalence is easier to see using the arguments from [53, p. 268].

PROOF. The proof can be found in Appendix A. □

The accuracy of the bounds ω_{max}, ω_{rad}, and ω_{cut} depends on the reduced zonotopes: While ω_{rad} performs better on average for random zonotopes, in practice ω_{max} performs better for reachability analysis. This originates from a bias of zonotopes in the reachability algorithm towards axis-aligned generators $g^{(p)}$ resulting in orthogonal $\widehat{g}^{(p)}$. This bias can be explained by the fact that on several occasions in the reachability algorithm, interval boxes are added to the current solution. Reducing the order of zonotopes by using box over-approximations further adds to this bias. The bound ω_{cut} performs slightly worse than ω_{max} in those biased cases. This can be seen by analyzing the effect of adding another generator g^* to the zonotope. Two effects can influence the performance of either bound: On the one hand, if g^* is diagonal (i.e., if the components of g^* have the same length, as opposed to g^* having only one non-zero component), ω_{max} will grow larger than ω_{cut}, meaning that ω_{max} will perform worse. On the other hand, if $\widehat{g}^*$ is orthogonal to the other vectors $\widehat{g}^{(p)}$ then ω_{max} performs better than ω_{cut}.

In other words, ω_{max} performs better if generators are added that are axis-aligned and orthogonal up to one component, which is the case for the generators of an interval box. Nevertheless, the overall performance of ω_{cut} is similar to ω_{max} in low dimensions and significantly better in higher dimensions.

Since the computation (17) of ω_{cut} contains the formula (16) for ω_{rad}, we combine both bounds to

$$\omega_{\mathrm{rad,cut}}(\mathscr{Z}) := \sum_{p=1}^{\gamma} \left\| g^{(p)} \right\|_2 \min \left\{ 1, \sqrt{2} \sqrt{1 - \frac{\sum_{i=1}^{n} \left(g_i^{(p)} \right)^4}{\left\| g^{(p)} \right\|_2^4}} \right\}. \tag{19}$$

In Fig. 1, we see that this new bound generally yields better results than ω_{max} as it combines the advantages of ω_{cut} and ω_{rad}. Note that the joint bound $\omega_{\mathrm{rad,cut}}$ performs better than either of the bounds ω_{rad} and ω_{cut} as the minimum in (19) is taken generator-wise. A potential combination of all three bounds is dismissed due to the resulting computational overhead.

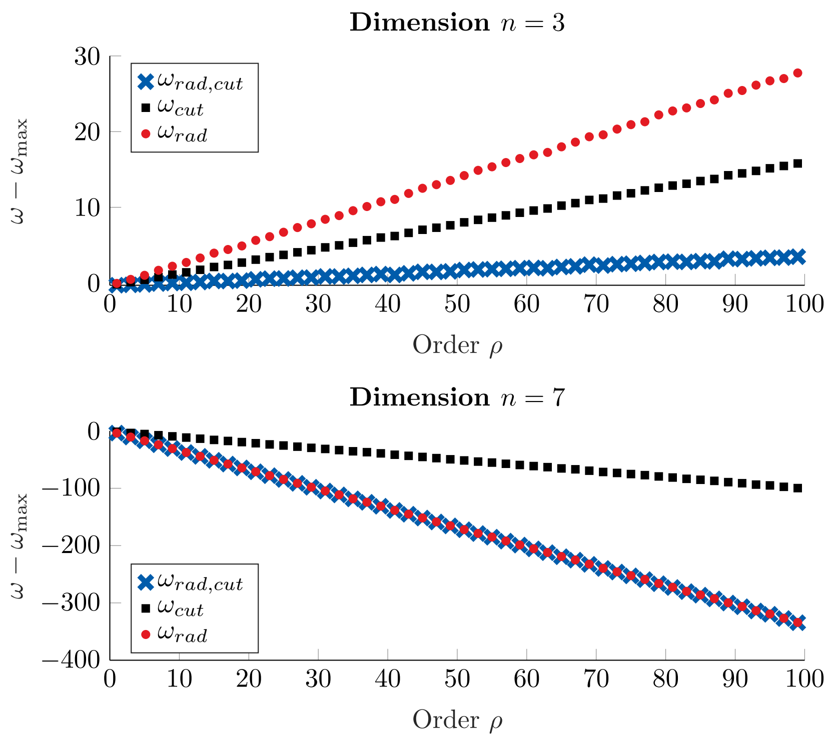

Figure 1: Comparison of ω_{rad}, ω_{cut}, and $\omega_{\text{rad,cut}}$ to $\omega_{\max}$ (i.e., $\omega_{\max}$ coincides with the x-axis), by computing each bound for 1000 zonotopes centered at the origin with generator matrices that have random entries between -1 and $+1$, and taking the mean. Negative values of $\omega - \omega_{\max}$ mean that the bound performs better than $\omega_{\max}$. One can see that for low dimensions, the combination of ω_{rad} and ω_{cut} can lead to better results, which are comparable to those of $\omega_{\max}$. For high dimensions, ω_{rad} is typically much more precise than ω_{cut} or $\omega_{\max}$.

Let us now provide a heuristic for generator selection, where we aim to reduce as many generators as possible while respecting a given threshold for the Hausdorff distance between the original and reduced set: Given a zonotope $\mathcal{Z} = \langle c, G \rangle_Z$, we sort the generators in $G \in \mathbb{R}^{n \times \gamma}$ using a cost function $\varrho(g)$. For $\omega_{\max}$, this cost function is

$$\varrho_{\max}(g) := \|g\|_1 - \|g\|_\infty, \tag{20}$$

originally proposed in [26]. For $\omega_{\text{rad,cut}}$, we exploit that the contribution of each generator to $\omega_{\mathrm{rad,cut}}$ can be separated, yielding the cost function

$$\varrho_{\mathrm{rad,cut}}(g) := \|g\|_2 \min\left\{1, \sqrt{2}\sqrt{1 - \frac{\sum_{i=1}^{n}(g_i)^4}{\|g\|_2^4}}\right\}. \tag{21}$$

Note that the cost function $\varrho_{\mathrm{rad}} = \|\cdot\|_2$ is exactly the cost function described in [27] and [54] to sort generators. However, since this is already computed when evaluating (21), the analysis of the computational complexity and the accuracy of ϱ_{rad} is similar to that of $\varrho_{\mathrm{rad,cut}}$. We will utilize the presented novel bounds for zonotope order reduction in our adaptive parameter tuning approach in Sec. 5.3.

4. Gain Order of Set-Based ODE Solvers

We first recall some basics about the numerical approximation of ODE solutions using classical solvers and discuss why an immediate transferability of these concepts to reachability algorithms is inadequate. Afterwards, we will outline our solution, which will be presented in the remainder of this section.

In classical ODE theory [34, 31], the quality of a numerical approximation of an ODE

$$\dot{x} = f(x,t) \tag{22}$$

is typically measured by the concept of *convergence order*:

Definition 3. (Convergence order of classical ODE solvers) Let $\xi_{\mathrm{ex}}(\Delta t; t_0, x_0)$ be the exact solution of (22) at time $t_0 + \Delta t$, with initial condition $x(t_0) = x_0$, and let $\widehat{\xi}(\Delta t; t_0, x_0)$ be an approximation of the exact solution at time $t_0 + \Delta t$. The *convergence order* (in short: *order*) of the solver is the number $q \in \mathbb{N}_0$, for which the following inequality holds:

$$\varepsilon_{\Delta t,\mathrm{loc}} := |\xi_{\mathrm{ex}}(\Delta t; t_0, x_0) - \widehat{\xi}(\Delta t; t_0, x_0)| \leq c\Delta t^{q+1}, \quad \forall t_0, x_0, \tag{23}$$

where c is a constant that neither depends on t_0, x_0, nor Δt, and $\varepsilon_{\Delta t,\mathrm{loc}}$ is called the *local truncation error*. □

The error (23) is a *local* measure as it only measures the error committed in a single time step. To estimate the accumulated error over the entire time interval $[0, t_{\text{end}}]$, let us denote by $\widehat{\xi}_N$ the approximation after N steps using the time step size $\Delta t = t_{\text{end}}/N$. Then, the *global* error ε_N between the exact point $\xi_{\text{ex}}(t_{\text{end}}; 0, x_0)$ and the approximation $\widehat{\xi}_N$ is bounded by [32, p. 318, Theorem 12.2]

$$\varepsilon_N = |\xi_{\text{ex}}(t_{\text{end}}; t_0, x_0) - \widehat{\xi}_N| \leq \widehat{c}\Delta t^q, \tag{24}$$

where $\widehat{c}$ is a constant that depends on t_{end}, but not on Δt. Thus, the order q provides an estimate on how small the time step size has to be in order to achieve good approximations—low-order methods, such as the explicit Euler method [31, Chapter 2] with order $q = 1$, will typically need much smaller time step sizes than high-order methods like the third-order variant of Heun's method [31, Chapter 9.5] with order $q = 3$.

For set-based methods used in reachability analysis, this error estimation cannot be generalized directly for several reasons:

- **Difference between sets:** Expressions such as $|\xi_{\text{ex}}(\Delta t; t_0, x_0) - \widehat{\xi}(\Delta t; t_0, x_0)|$ in (23) are difficult to evaluate if $\xi_{\text{ex}}(\Delta t; t_0, x_0)$ and $\widehat{\xi}(\Delta t; t_0, x_0)$ are replaced by their set-based analogues, i.e., the reachable set and some approximation that outputs a set.

- **Wrapping effects:** The magnitude of some approximation errors, primarily due to wrapping effects, such as the iterative order reduction operations discussed in Section 3, is independent of the time step size and thus cannot be reduced by choosing a smaller time step size. Even worse, reducing the time step size leads to a higher number of iterations and might result in an overall larger error, which does not happen for classical ODEs.

To address these problems, we propose a novel concept for the convergence order of set-based solvers. Instead of focusing on a local truncation error $\varepsilon_{\Delta t,\text{loc}}$, we will define a *gain function* $\varphi(t; \delta)$ that measures the overall error for varying time step sizes, and for which holds that

$$\varepsilon_N \leq c\Delta t^{-1}\varphi(t_{\text{end}}; \frac{1}{N}), \tag{25}$$

where ε_N is an appropriate measure for the global error of a set-based solver and c is a constant that neither depends on Δt nor N.

For conciseness, our subsequent analysis will only be applied to systems of ordinary differential equations (22), where we will assume a smooth right-hand side. Under certain simple assumptions, we will show that $\varphi(t_{\text{end}}; \frac{1}{N}) \leq 1/N^{q+1}$ for some $q \in \mathbb{N}_0$ that may depend on the right hand side f of (22), leading to the result

$$\varepsilon_N \leq \widehat{c}\Delta t^q, \tag{26}$$

where $\widehat{c}$ is a constant that depends on t_{end}, but not on Δt nor N. As a consequence, the gain function $\varphi(t;\delta)$ allows us to mimic the result from (24) that one can get via classical ODE theory, without having to analyze the precise behavior of the local truncation error for each point in the initial set.

The next subsection Sec. 4.1 defines local and global abstraction errors of set-based solvers, followed by an investigation of the local error over varying time step sizes Δt using a formal definition of the aforementioned gain function in Sec. 4.2. Finally, we extend the obtained results from local to global errors in Sec. 4.3. Most of the proofs are to be found in Appendix B to enhance the fluidity of our presentation.

4.1. Errors of set-based ODE solvers

In order to define the notion of order for a given set-based solver, we need an estimation of the local error that such a solver produces (similarly to Def. 3). We focus our attention on the set of abstraction errors $\mathcal{R}_{\text{abs}}$ (8), which by definition collects the approximation errors induced by $\mathcal{R}_{\text{lin}}$ in (10). This is comparable to the work in [33]. More generally, we propose the following measure for the local and global error:

Definition 4. (Errors of set-based solvers) Let $\mathcal{R}(\Delta t; t_0, \mathcal{X}^0)$ be a set-based approximation for (22), with initial value $x(t_0) \in \mathcal{X}^0$ and time step size Δt.

Using the abstraction error as defined in (8), we have that

$$\mathcal{R}(\Delta t; t_0, \mathcal{X}^0) = \widehat{\mathcal{R}}(\Delta t; t_0, \mathcal{X}^0) \oplus \mathcal{R}_{\text{abs}}(\Delta t; t_0, \mathcal{X}^0), \tag{27}$$

where $\widehat{\mathcal{R}}(\Delta t; t_0, \mathcal{X}^0)$ is the approximation of the solution to the ODE (22). Let $\mathcal{R}(t_k)$ be the output after $k \in \{0, 1, ..., N\}$ iterations of $\mathcal{R}$ and let $\widehat{\mathcal{R}}(t_k)$ be the output after iteratively using $\widehat{\mathcal{R}}$ instead of $\mathcal{R}$. Then, the *local abstraction error* at time t_k is defined as

$$\begin{aligned} \varepsilon_{k,\text{loc}} &:= \left\| d\big(\texttt{box}\big(\mathcal{R}(\Delta t; t_k, \mathcal{R}(t_k)) \ominus \widehat{\mathcal{R}}(\Delta t; t_k, \mathcal{R}(t_k))\big)\big) \right\|_\infty \\ &= \left\| d\big(\texttt{box}\big(\mathcal{R}_{\text{abs}}(\Delta t; t_k, \mathcal{R}(t_k))\big)\big) \right\|_\infty, \end{aligned} \tag{28}$$

whereas

$$\varepsilon_N := \left\| d\big(\texttt{box}\big(\mathcal{R}(t_{\text{end}}) \ominus \widehat{\mathcal{R}}(t_{\text{end}})\big)\big) \right\|_\infty. \tag{29}$$

is the *global accumulated abstraction error* over N steps. □

Similarly to classical ODE theory, the global error can be linked to the local error in the following manner:

Lemma 2. *(Local and global abstraction error) For solvers of the form described in* (10)*, the global abstraction error after N steps can be bounded by the maximal local abstraction error as*

$$\varepsilon_N \leq \frac{c}{\Delta t} \max_{1 \leq \ell \leq N} \varepsilon_{\ell,loc} \tag{30}$$

for some constant c that may depend on t_{end}, but not on Δt nor N.

Proof. The proof can be found in Appendix B. □

4.2. The gain function

Now that we know how to quantify the error of a set-based solver, we investigate the behavior of the error for varying time step sizes Δt. For simplicity, we will drop the the arguments k, t_k, t_0, and $\mathcal{X}^0$ for the rest of this section to focus on the effect of Δt on the local abstraction error. Moreover, for the sake of simplicity, instead of working with the local abstraction error as defined in (28), we will use the more accurate formulation

$$\varepsilon_{\text{loc}} = \Big\| d\big(\texttt{box}\big(\mathcal{R}^{\infty}_{\text{abs}}(\Delta t)\big)\big) \Big\|_{\infty}, \tag{31}$$

where $\mathcal{R}^{\infty}_{\text{abs}}(\Delta t)$ is the *non-truncated* set of abstraction errors replacing (8), i.e.,

$$\mathcal{R}^{\infty}_{\text{abs}}(\Delta t) = \bigoplus_{p=0}^{\infty} \frac{\Delta t^{p+1}}{(p+1)!} A^p \Psi(\Delta t). \tag{32}$$

While the local error (31) explicitly depends on the time step size Δt, Ψ also depends on the initial set $\mathcal{X}^0$. The Taylor expansion of (31) entails that there exist some non-negative integers q_s and q_t such that

$$\varepsilon_{\text{loc}} = \Delta t \Big[\mathcal{O}(\Delta t^{q_t}) + \mathcal{O}(d(\mathcal{X}^0)^{q_s}) \Big] \tag{33}$$

for small enough Δt and $d(\mathcal{X}^0)$, which is similar to classical ODE theory [31, Chapter 9]. The error ε_{loc} can thus be decreased either by reducing the time step size, or by splitting the initial set as in [33]. As Lemma 2 shows, the global error ε_N is bounded by a term proportional to ε_{loc}. Its dependency on the set size and the time step size is illustrated in Fig. 2: For classical solvers, we have $d(\mathcal{X}^0) = 0$ yielding the black curve which converges to $\varepsilon_N = 0$ for $\Delta t \to 0$. In constrast, for set-based solvers we obtain a behavior like the red curve as we evaluate $\varepsilon_N(d(\mathcal{X}^0), \Delta t)$ on a projection (indicated by the gray plane) at $d(\mathcal{X}^0) = d^* > 0$. Crucially, this results in a value $\varepsilon_N > 0$ for $\Delta t \to 0$.

The estimate (33) can be made independent of the right hand side f of the ODE, up to a multiplicative constant depending on the smoothness of f (see for example [31, Chapter 9.4]). Thus, it would allow one to separately define a *time* order q_t and a *space* order q_s. However, these concepts of order are difficult to measure locally as one would need to measure the combined error as a function of several variables. In contrast, classical ODE theory only considers the local error with respect to time.

Therefore, a different approach is more enticing: Instead of analyzing the precise behavior of ε_{loc} as a function of time and space (as in [33]), we consider its overall expansion over time by defining the following *gain function*:

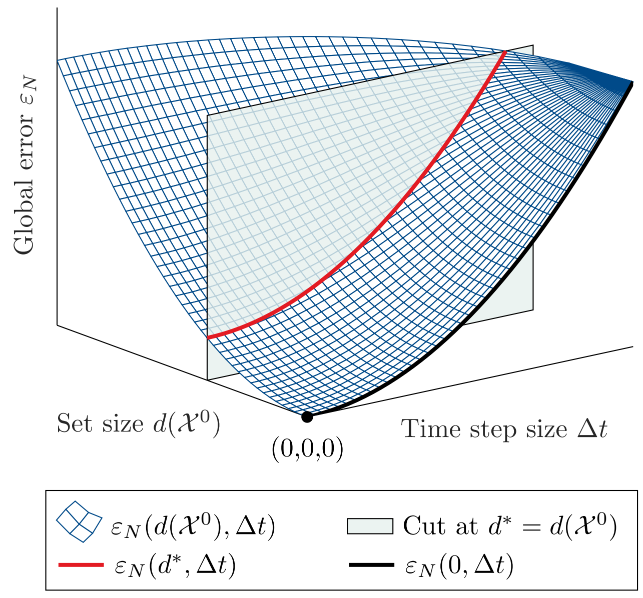

Figure 2: Schematic evaluation of the local abstraction error $\varepsilon_{\mathrm{loc}}$ as a function of space (set size $d(\mathcal{X}^0)$) and time (time step size Δt); two projections for $d(\mathcal{X}^0) = 0$ (classical solvers) and $d(\mathcal{X}^0) = d^* > 0$ (set-based solvers), resulting in the black and red curve, respectively, show a different limit value in the limit $\Delta t \to 0$.

Definition 5. (Gain function) The *gain* $\varphi(h; \delta)$ over the time span h with relative increments $\delta \in [0, 1]$ is the function

$$\varphi(h; \delta) = \max_{i \in \{1,\dots,n\}} \varphi_i(h; \delta) \quad \text{with} \quad \varphi_i(h; \delta) = \frac{d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(\delta h)\big)\big)}{d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(h)\big)\big)}\,, \tag{34}$$

where $\mathcal{R}^{\infty}_{\mathrm{abs}}(h)$ is computed by (32). $\square$

If $\Psi(h)$ in (32) is represented by a zonotope, the gain φ can be simplified using the following Proposition:

Proposition 1. *(Diameter of set of abstraction errors) Suppose $\Psi(h)$ is given as the zonotope $\langle c(h), G(h)\rangle_Z$. Then we obtain*

$$d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{abs}(h)\big)\big) = 2 \sum_{p=0}^{\infty} \frac{h^{p+1}}{(p+1)!} \left\| A^p G(h) \right\|_{1,i}\,, \tag{35}$$

where for a matrix M, $\|M\|_{1,i}$ denotes the 1-norm of the i-th row of M.

PROOF. The proof can be found in Appendix B. □

Following the result of Prop. 1, we can analyze the behavior of φ by examining the behavior of the coefficients $\|A^p G(h)\|_{1,i}$, for which we now present a more concrete characterization:

Lemma 3. *(Expansion of coefficients) Assume that the right-hand side f of (22) is smooth. Then, $G(h)$ is smooth, and for all $p \in \mathbb{N}_0$ and $i = \{1, ..., n\}$, the coefficient $\|A^p G(h)\|_{1,i}$ may either be written as*

$$\|A^p G(h)\|_{1,i} = h^{q_i^{(p)}} \|a_i^{(p)} + b_i^{(p)}(h)\|_1, \tag{36}$$

for some $q_i^{(p)} \in \mathbb{N}_0$, $a_i^{(p)} \in \mathbb{R}\backslash\{0\}$, $b_i^{(p)}(h) = \mathcal{O}(h)$, or as

$$\|A^p G(h)\|_{1,i} \equiv 0, \tag{37}$$

in which case we use the convention that $q_i^{(p)} = \infty$, as well as $a_i^{(p)} = 0$ and $b_i^{(p)}(h) = 0$. Furthermore, for all $p \in \mathbb{N}_0$ and $i = \{1, ..., n\}$, the function

$$Q_i^{(p)}(h) = |a_i^{(p)} + b_i^{(p)}(h)| \tag{38}$$

is non-negative and piecewise smooth.

PROOF. The proof can be found in Appendix B. □

We can now use this knowledge about the numerator and denominator defining $\varphi(h; \delta)$ in order to investigate its behavior for $h \to 0$, through which a certain notion of order will arise naturally:

Theorem 2. *(Limit gain) Suppose the right hand side f of (22) is smooth, and for each $p \in \mathbb{N}_0$ and $i = \{1, ..., n\}$ let $a_i^{(p)}$ and $q_i^{(p)}$ be defined as in Lemma 3. Then for the gain in the i-th dimension $\varphi_i(h; \delta)$ there holds*

$$\lim_{h \to 0} \varphi_i(h; \delta) = \delta^{q_i+1}, \tag{39}$$

where

$$q_i = \max\left\{ j \in \mathbb{N}_0 \,\middle|\, \sum_{p+q_i^{(p)}=j} |a_i^{(p)}| \neq 0 \right\}. \tag{40}$$

Note that the condition on j in (40) is met for a given j if and only if either there does not exist a tuple $(p, q_i^{(p)})$ such that $p + q_i^{(p)} = j$, or if for any such tuple there holds $a_i^{(p)} = 0$, i.e., $[A^p(h)G(h)]_i \equiv 0$.

Proof. Using (35), we deduce by (36) that

$$\begin{aligned} d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\text{abs}}(h)\big)\big) &= 2\sum_{p=0}^{\infty} \frac{h^{p+1}}{(p+1)!} h^{q_i^{(p)}} \|a_i^{(p)} + b_i^{(p)}(h)\|_1 \\ &= 2\sum_{j=0}^{\infty} h^{j+1} \sum_{p+q_i^{(p)}=j} \frac{\|a_i^{(p)} + b_i^{(p)}(h)\|_1}{(p+1)!} \end{aligned}$$

Inserting this into (34) yields

$$\varphi_i(h;\delta) = \frac{\sum_{j=0}^{\infty} h^{j+1}\delta^{j+1} \sum_{p+q_i^{(p)}=j} \frac{\|a_i^{(p)}+b_i^{(p)}(h\delta)\|_1}{(p+1)!}}{\sum_{j=0}^{\infty} h^{j+1} \sum_{p+q_i^{(p)}=j} \frac{\|a_i^{(p)}+b_i^{(p)}(h)\|_1}{(p+1)!}} \xrightarrow[h\to 0]{} \delta^{q_i+1}, \tag{41}$$

where while passing to the limit we used the fact that $b_i^{(p)} = \mathcal{O}(h)$ together with the assumption that

$$\sum_{p+q_i^{(p)}=q_i} \frac{|a_i^{(p)}|}{(p+1)!} \neq 0,$$

which is equivalent to (40). □

The quantities q_i have the unique property that they can describe the overall behavior of the error for different time step sizes. Consequently, they constitute the basis of our concept of order:

Definition 6. (Gain order of set-based solvers) Let $\mathcal{R}$ be an approximation of (22), and let q_i be defined as above. Then $q := \min_i q_i$ is called the *gain order* of the method, and q_i is called the gain order of the i-th dimension. Note that $\mathcal{R}$ yields zero abstraction error if and only if $q = \infty$. This is because from (35) it follows that the abstraction error is zero if and only if $\|A^pG(h)\|_{1,i} = 0$ for all p and i, which is equivalent to $q = \infty$. □

We aim to show that, under certain simple assumptions, φ is monotonically decreasing in h. This will be crucial in practice because it shows that even if the global abstraction error does not decrease by $\mathcal{O}(\Delta t^p)$ for some $p \geq 1$ (where Δt is the time step size), it does indeed decrease notably for smaller time step sizes.

Proposition 2. *(Derivative of gain) Let $q_i^{(p)}$ and $Q_i^{(p)}$ be defined as in Lemma 3, and let q_i be defined as in (40). Assume that all $Q_i^{(p)}$ are differentiable at $t = 0$. Then, for $\delta \in [0, 1]$, there holds*

$$\lim_{h \to 0} \frac{\mathrm{d}\varphi_i(h; \delta)}{\mathrm{d}h} \leq 0. \tag{42}$$

Furthermore, for fixed h, if $\forall i \in \{1, ..., n\}$ and $p \in \mathbb{N}_0$ the value of $Q_i^{(p)}(t)$ is constant over $t \in [0, h]$, then

$$\varphi_i(t; \delta) \leq \delta^{q_i+1}, \quad \forall t \in [0, h],\ i \in \{1, ..., n\}. \tag{43}$$

Proof. The proof can be found in Appendix B. □

As we have seen, the gain function provides an alternative way of estimating the dependency of the local abstraction error with respect to the time step size, and this estimation can describe non-integer orders by means of the function φ.

4.3. Global abstraction error via gain order

After estimating the local error using the gain function, we now want to extend these results to an estimation of the global abstraction error. To do so, we select h to be a fixed finite time horizon that we will use as a unit of measurement, and through which we will comparatively observe the effect of reducing the time step size $\Delta t < h$. For some fixed $\delta \in [0, 1]$, by (42) we can choose h to be small enough such that $\varphi(t; \delta)$ is decreasing on $t \in [0, 2h]$. By (39), the limit of $\varphi(t; \delta)$ for $t \to 0$ is δ^{q+1}, therefore on $[0, 2h]$ there must hold that $\varphi(t; \delta) \leq \delta^{q+1}$. For $t = h$, this yields the relation

$$\varphi(h; \delta) \leq \delta^{q+1}. \tag{44}$$

The latter inequality holds even for relatively large h in practice, as we shall discuss later on.

Since the gain φ provides an estimate for the variation of the local abstraction error, φ depends on the step k just like $\varepsilon_{k,\mathrm{loc}}$ (28) does. Let φ_k denote the gain function corresponding to step k.

From (30) it follows that the global error ε_N after N steps can be estimated by

$$\begin{aligned}
\varepsilon_N &\overset{(30)}{\leq} c\Delta t^{-1} \max_{1\leq k\leq N} \max_i d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(\Delta t; t_k, \mathcal{R}(t_k))\big)\big) \\
&\overset{\text{Def. 5}}{\leq} c\Delta t^{-1} \max_{1\leq k\leq N} \varphi_k(h; \frac{1}{N}) \max_i d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(h; t_k, \mathcal{R}(t_k))\big)\big) \\
&\leq c\Delta t^{-1} \max_{1\leq k\leq N} \varphi_k(h; \frac{1}{N}) \max_{1\leq k\leq N} \max_i d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(h; t_k, \mathcal{R}(t_k))\big)\big) \\
&\overset{(44)}{\leq} c\Delta t^{-1} \frac{1}{N^{q+1}} \max_{1\leq k\leq N} \max_i d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(h; t_k, \mathcal{R}(t_k))\big)\big).
\end{aligned}$$

From [18, (11)] it obviously follows that $\mathcal{R}^{\infty}_{\mathrm{abs}}(h + t_k; 0, \mathcal{X}^0) \subseteq \mathcal{R}^{\infty}_{\mathrm{abs}}(2h; 0, \mathcal{X}^0)$, since $t_k \leq h$. Therefore, we can write

$$\varepsilon_N \leq c\Delta t^{-1} \frac{1}{N^{q+1}} \mathcal{R}^{\infty}_{\mathrm{abs}}(2h; 0, \mathcal{X}^0)$$

which eventually yields a bound

$$\varepsilon_N \leq c'\Delta t^{-1} \frac{1}{N^{q+1}}.$$

Since $N = h/\Delta t$, we conclude that

$$\varepsilon_N \leq \widehat{c}(h)\Delta t^q,$$

where $\widehat{c}(h)$ is a constant that depends on h, but not on Δt nor N. Consequently, our definition of the order of a solver yields similar results to classical ODE theory, except that $\varphi(h; \frac{1}{N})$ can give even more precise information, e.g., non-integer orders, about the local improvement one would get by decreasing the time step size.

In practice, we will primarily use $\varphi(h; \delta)$ to determine an approximation of $\varepsilon_{k,\mathrm{loc}}$ for small variations of Δt. As we saw earlier, (42) implies that $\varphi(h; \delta) \leq \delta^{q+1}$ if h is small enough, a restriction that can be loosened for larger h using Prop. 2 under the assumption that $A^k G(h)$ in Prop. 1 does not vary much with respect to h, as this would imply that the $Q_i^{(p)}$ are constant. This assumption holds as long as $\Psi(h)$ does not vary much, which is true in practice as long as the computation of $\Psi(h)$ converges.

In the next section, we will make use of the gain function φ (see Def. 5), its limit value (see Theorem 2), and its derivative (see Prop. 2 and the discussion above) in order to construct an optimization function for the crucial tuning of the time step size.

5. Adaptive Parameter Tuning Methods

In this section, we propose individual tuning methods to adaptively tune each algorithm parameter used in Alg. 1. Fig. 3 provides an overview of the reachable set computation within one time step, where arrows indicate the effect of algorithm parameters on sets. All described tuning methods are modular, that is, the algorithm parameters are adapted independently of one another. The final composition of all tuning modules in an adaptive tuning algorithm yields a general framework for state-space abstracted reachability algorithms such as Alg. 1. Each tuning module can simply be exchanged for a different one, e.g., if different reachable set computations or set representations are chosen.

We will generally omit the index k for the current step as all algorithm parameters are adapted in each step. The remainder of this section describes the individual tuning methods for the propagation parameters η (Sec. 5.1), the abstraction order κ (Sec. 5.2), the set representation size ρ (Sec. 5.3)—all of which are tuned by threshold conditions determining sufficient accuracy— and the time step size Δt (Sec. 5.4) obtained by solving an optimization problem that minimizes the unavoidable over-approximation over a finite time horizon. Finally, we introduce the automated tuning algorithm as an enhancement to Alg. 1 in Sec. 5.5.

5.1. Propagation parameters

As schematically indicated in Fig. 3, the computation of the sets $\mathcal{R}_{\text{abs}}$ in (8) and $\mathcal{R}_{\text{lin}}$ in (10)-(12) requires us to tune the order η of the finite Taylor series

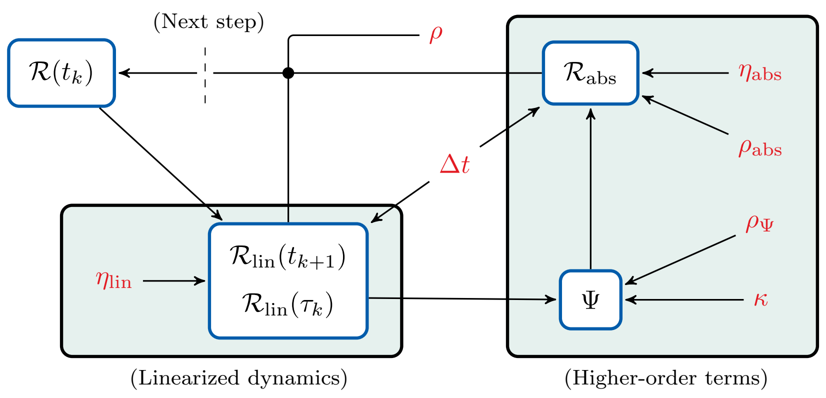

Figure 3: Main workflow for one time step in Alg. 1 and the influence of algorithm parameters on different sets (an arrow $A \to B$ means that A is used to compute B): The time step size Δt affects both the linearized dynamics and the higher-order terms, while the abstraction order κ only influences the latter. The propagation parameters η affect the precision of the exponential matrix and the set representation parameters ρ represent the reduction operation, which is applied to various sets within one step.

of the exponential matrix. The main idea is to exploit that the contribution of higher-order terms eventually vanishes. As a consequence, we truncate the respective Minkowski sums once the contribution of the additional term has become small enough to determine the orders η_{lin} and η_{abs}.

The over-approximation error in $\mathcal{R}_{\text{lin}}(\tau_k)$ is dominated by the error term $\mathbf{F}_x\mathcal{R}(t_k)$ [43, eq. (21)]. Thus, we tune η_{lin} using a fixed threshold $0 < \zeta_{T,\text{lin}} \ll 1$ related to the influence of additional terms $\mathbf{T}^{(p)}$in (11):

$$\eta_{\text{lin}} = \min_{\substack{p\in\mathbb{N}\\ p\geq 2}} p \qquad \text{such that} \qquad 1 - \frac{\|\mathbf{T}^{(p-1)}\|_F}{\|\mathbf{T}^{(p)}\|_F} \leq \zeta_{T,\text{lin}}. \tag{45}$$

Using the same idea of comparing successive orders, we determine the order η_{abs} by truncating the sum in (8) according to the criterion

$$\eta_{\text{abs}} = \min_{p\in\mathbb{N}_0} p \qquad \text{such that} \qquad \max_{i\in\{1,\dots,n\}} \frac{d_i\big(\mathtt{box}\big(\mathcal{R}_{\text{abs}}^{(p+1)}\big)\big)}{d_i\big(\mathtt{box}\big(\mathcal{R}_{\text{abs}}^{(p)}\big)\big)} \leq \zeta_{T,\text{abs}}, \tag{46}$$

with $\mathcal{R}_{\text{abs}}^{(p)}$ denoting the sum in (8) truncated at order p and $0 < \zeta_{T,\text{abs}} \ll 1$.

As both criteria (45) and (46) can be evaluated during the iterative computation of the respective sets, the tuning itself yields negligible computational overhead. Note that there are no propagation parameters in the reachable set computation of discrete-time systems.

5.2. Abstraction order

According to Fig. 3, the abstraction order κ in (3) directly influences the abstraction error Ψ and subsequently the set of abstraction errors $\mathcal{R}_{\mathrm{abs}}$. In general, larger values of κ are computationally more demanding due to the evaluation of higher-order maps, but also decrease the size of the set of abstraction errors $\mathcal{R}_{\mathrm{abs}}$.

For a linearization approach, we restrict the admissible values of the abstraction order to $\kappa = \{1, 2\}$, as the evaluation of cubic or higher-order maps is highly over-approximative using zonotopes. In constrast, non-convex set representations are closed under higher-order maps but significantly increase the set representation size which cannot be handled by current reduction methods. Hence, the abstraction order for the polynomialization approach is fixed to $\kappa = 2$.

We propose a two-step selection criterion to limit the computational overhead:

1. The set of abstraction errors $\mathcal{R}_{\mathrm{abs}}$ of the current step k is compared to the one of the last comparison, denoted by k'. To establish a level playing field, we estimate the size of $\mathcal{R}_{\mathrm{abs}}(\Delta t_k)$ for the time step size $\Delta t_{k'}$ using the gain φ^* based on (34). The condition

$$\left| \frac{\varphi^* \, \mathtt{rad}\big(\mathcal{R}_{\mathrm{abs}}(\Delta t_k)\big)}{\mathtt{rad}\big(\mathcal{R}_{\mathrm{abs}}(\Delta t_{k'})\big)} \right| > 1 - \zeta_K, \quad \zeta_K \in (0, 1), \tag{47}$$

   decides whether we progress to the second step below. Informally, we compare the sizes of the set of abstraction errors and only if the size difference is large enough, the value for κ is re-tuned.

2. If the condition (47) is fulfilled, we also compute $\mathcal{R}_{\mathrm{abs}}(\Delta t_k)$ for the other

value of κ and decide the next value according to the following condition:

$$\kappa \leftarrow \begin{cases} 1, & \text{if } \forall i \in \{1, ..., n\} \,\text{with}\, d_i(\mathcal{R}_{\text{abs}}) > 0 : \frac{d_i(\texttt{box}(\mathcal{R}_{\text{abs}}(\kappa=2)))}{d_i(\texttt{box}(\mathcal{R}_{\text{abs}}(\kappa=1)))} \geq \zeta_K \\ 2, & \text{otherwise.} \end{cases} \tag{48}$$

The closer ζ_K is to 1, the more conservative the selection becomes, i.e., the more often $\kappa = 2$ will be chosen resulting in both a tighter result as well as longer computation times. For the first step, we use the initial set $\mathcal{R}(t_0) = \mathcal{X}^0$ to compute Ψ and immediately evaluate (48) to compute the first abstraction order κ. For discrete-time systems, we exploit two properties to greatly simplify the computation: First, the abstraction error Ψ is used directly in the reachable set computation replacing $\mathcal{R}_{\text{abs}}$ as seen in (13). Second, we do not need to compensate for different time step sizes in subsequent steps so that we always have $\varphi^* = 1$ in (47).

5.3. Set representation

A reduction of the representation size can only avoid large over-approximations if the reduction error is restricted by an upper bound. We utilize the two bounds ω_{max} (15) and $\omega_{\text{rad,cut}}$ (19) for the Hausdorff distance between the original zonotope and its reduced counterpart from Sec. 3. For either bound, we sort the generators of $\mathcal{Z}$ in ascending order with respect to its respective cost function ϱ_{max} (20) or $\varrho_{\text{rad,cut}}$ (21). Then, we select the first $\gamma^* \leq \gamma$ generators, until we reach the upper bound

$$\sum_{p=1}^{\gamma^*} \varrho\big(g^{(p)}\big) \leq \zeta_Z \left\| d\big(\texttt{box}(\mathcal{Z})\big) \right\|_2, \tag{49}$$

where we use a fixed fraction $0 < \zeta_Z \ll 1$ of the diagonal of the box over-approximation of the original zonotope $\mathcal{Z}$. The exact Hausdorff distance between the original and the reduced set is smaller than the left-hand side in (49) by Theorem 1.

Our polynomialization approach is best used with a non-convex set representation, where we choose polynomial zonotopes allowing us to exploit their similarities to zonotopes which we extensively covered in Sec. 3. The reduction method for polynomial zonotopes described in [47, Prop. 10] is based on zonotope order reduction, so that we can reuse the bound (49) to decrease the representation size.

5.4. Time step size

Alg. 1 contains two main sources for over-approximation, both of which are related to the time step size Δt: First, we have the set of abstraction errors $\mathcal{R}_{\text{abs}}$ whose behavior over Δt has been thoroughly investigated in Sec. 4. By *decreasing* the time step size, we reduce the size of $\mathcal{R}_{\text{abs}}$ and thus alleviate the wrapping effect originating from the iterative addition of $\mathcal{R}_{\text{abs}}$. Second, the reduction operation induces another wrapping effect whose effect is diminished by *increasing* time step sizes. Sec. 5.3 describes the error induced by a single reduction operation, which we now have to consider over multiple steps.

In order to obtain a tight reachable set, we require to tune Δt so that the trade-off between both wrapping effects is optimized as shown in Fig. 4: The start set is propagated over a finite time horizon h using different candidate time step sizes $\Delta t^{(\iota)} = \frac{h}{\iota}$, $\iota \geq 1$. The optimal time step size Δt_* (gray sets) balances both wrapping effects so that neither the set of abstraction errors $\mathcal{R}_{\text{abs}}$ is too large (red sets) nor is the reduction operation applied too often (blue sets)—both of which yield a larger set at time $t + h$.

In order to efficiently solve the optimization problem in each step, we make some design choices without which the comparison of different time step sizes $\Delta t^{(\iota)}$ would become infeasible in practice:

(a) We assume the system matrix A, i.e., the Jacobian matrix of $f(x)$, to be constant over $[t, t + h]$ and use $A = A(t)$.

(b) We will neglect the particular solution $\mathcal{P}(\tau_k)$ in the propagation formula for the reachable set (10).

(c) For each $\Delta t^{(\iota)}$, we assume the set of abstraction errors $\mathcal{R}_{\text{abs}}(\Delta t^{(\iota)})$ to be constant over $[t, t + h]$ and use $\mathcal{R}_{\text{abs}} = \mathcal{R}_{\text{abs}}(\Delta t^{(\iota)})$ obtained at time t.

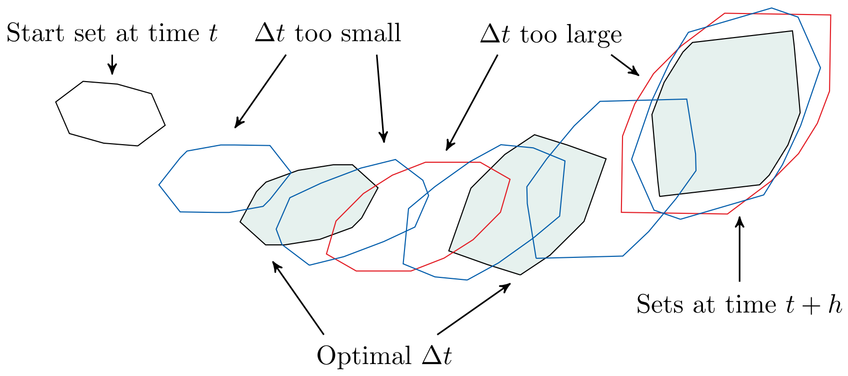

Figure 4: The optimal value Δt_* (gray sets) for the set propagation over a time horizon $[t, t+h]$ is obtained by balancing the wrapping effects: If Δt is too large (red sets), the set of abstraction errors $\mathcal{R}_{\text{abs}}$ is too large; if Δt is too small (blue sets), the reduction operation excessively increases the set size.

(d) We linearly interpolate the gain φ (34) between $\varphi(\Delta t = h) = \varphi^{(1)}$ and the limit gain computed in Theorem 2 $\lim_{\Delta t \to 0} \varphi(\Delta t) = \delta^{q+1}$ to obtain

$$\varphi(\Delta t) \approx \zeta_\delta + \frac{\varphi^{(1)} - \zeta_\delta}{h} \Delta t, \tag{50}$$

where we replace δ by the fixed value $\zeta_\delta \in (0, 1)$ and choose the lowest order $q = 0$, potentially underestimating the actual gain in order to prevent the time step size from decreasing too much. We will use this interpolation for all $\Delta t^{(\iota)}$.

Note that we will explicitly consider $\iota \in \mathbb{R}$ to facilitate any candidate time step size $\Delta t^{(\iota)} \in (0, h]$, which requires to take a last incomplete step of length $b\Delta t^{(\iota)} = (\iota - \lfloor \iota \rfloor)\Delta t^{(\iota)}$ into account in order to compare the resulting sets at the same point in time as shown in Fig. 4. In addition to the given design choices, we simplify the set-based evaluation to scalar values in three ways:

- The sizes of the start set $\mathcal{R}(t)$ and the set of abstraction errors $\mathcal{R}_{\text{abs}}(\Delta t^{(\iota)})$ are approximated by their respective radii $r_0 = \texttt{rad}\big(\mathcal{R}(t)\big)$ and $r_{\text{abs}}^{(\iota)} = \texttt{rad}\big(\mathcal{R}_{\text{abs}}(\Delta t^{(\iota)})\big)$.
- The effect of the exponential matrix is captured by its determinant, which

over the entire finite horizon can be estimated by $\det(e^{Ah}) = e^{\mathrm{tr}(Ah)}$, leading to a scaling factor of

$$\zeta_A^{\frac{1}{\iota}} = \Big(e^{\mathrm{tr}(Ah)}\Big)^{\frac{1}{\iota}} \tag{51}$$

for each partial step of length $\Delta t^{(\iota)}$. For the scaling of the last incomplete step, we have $\zeta_A^{\frac{b}{\iota}}$.

- The enlargement caused by the zonotope order reduction is measured by multiplying the radius with $(1 + 2\zeta_Z)$ following (49). The factor for the last incompete step is $(1 + 2\zeta_Z)^b$.

As only time-point solutions are reused in subsequent steps, we repeatedly apply (10) to compute the reachable set after time h, omitting the particular solution as stated in design choice (b):

$$\tilde{\mathcal{R}}(t+h) = \mathtt{red}\big(e^{Ab\Delta t^{(\iota)}}\mathtt{red}\big(e^{A\Delta t^{(\iota)}}...\mathtt{red}\big(e^{A\Delta t^{(\iota)}}\mathcal{R}(t)\oplus\mathcal{R}_{\mathrm{abs}}\big)...\oplus\mathcal{R}_{\mathrm{abs}}\big)\oplus b\mathcal{R}_{\mathrm{abs}}\big), \tag{52}$$

where $e^{A\Delta t^{(\iota)}}$ and $\mathcal{R}_{\mathrm{abs}}$ are scaled to $e^{Ab\Delta t^{(\iota)}}$ and $b\mathcal{R}_{\mathrm{abs}}$ in the last incomplete step. Based on the aforementioned simplifications, we now rewrite the set-based formula (52) to a scalar estimate for the set size of the reachable set using the recursive formula

$$r_R(t + j\Delta t^{(\iota)}) = (1 + 2\zeta_Z)\big(\zeta_A^{\frac{1}{\iota}} r_R(t + (j-1)\Delta t^{(\iota)}) + r_{\mathrm{abs}}^{(\iota)}\big), \tag{53}$$

which starts with the set size estimate at time t given by $r_R(t) := r_0$. To obtain an estimate after time h, we apply the recursion $\lfloor\iota\rfloor$ times and then include the last incomplete step:

$$\begin{aligned} r_R(t+h) = (1 + 2\zeta_Z)^b \cdot \\ \big(\zeta_A^{\frac{b}{\iota}} \underbrace{(1 + 2\zeta_Z)\big[\zeta_A^{\frac{1}{\iota}}...(1 + 2\zeta_Z)(\zeta_A^{\frac{1}{\iota}} r_0 + r_{\mathrm{abs}}^{(\iota)})... + r_{\mathrm{abs}}^{(\iota)}\big]}_{\overset{(53)}{=} r_R(t+\lfloor\iota\rfloor\Delta t^{(\iota)})} + b\, r_{\mathrm{abs}}^{(\iota)}\big). \end{aligned}$$

Summarizing the first $\lfloor\iota\rfloor$ steps yields

$$r_R(t+h) = (1+2\zeta_Z)^b\Big(\zeta_A^{\frac{b}{\iota}}\big[r_0(1+2\zeta_Z)^{\lfloor\iota\rfloor}\zeta_A^{\frac{\lfloor\iota\rfloor}{\iota}} + r_{\mathrm{abs}}^{(\iota)}\sum_{j=1}^{\lfloor\iota\rfloor}(1+2\zeta_Z)^j\zeta_A^{\frac{j-1}{\iota}}\big] + b\, r_{\mathrm{abs}}^{(\iota)}\Big),$$

after which we include the last incomplete step and rearrange to

$$r_R(t+h) = r_0(1+2\zeta_Z)^{\iota}\zeta_A + r_{\text{abs}}^{(\iota)}\,\zeta_{A,Z}(\iota), \tag{54}$$

$$\text{where} \quad \zeta_{A,Z}(\iota) = \sum_{j=1}^{\lfloor \iota \rfloor}(1+2\zeta_Z)^{b+j}\,\zeta_A^{\frac{b+j-1}{\iota}} + b(1+2\zeta_Z)^{b}$$

contains all factors of the term $r_{\text{abs}}^{(\iota)}$.

Since the evaluation of (54) would require us to compute $r_{\text{abs}}^{(\iota)}$ for each ι (which is obviously undesirable in practice due to the large computational overhead), we approximate $r_{\text{abs}}^{(\iota)}$ utilizing design choice (d). Let us first define $\iota' \in \mathbb{N}$ as the number of times h has been scaled by a fixed ζ_δ. Hence, $\iota = \zeta_\delta^{-\iota'} \in \mathbb{R}$ is the number of times $\Delta t^{(\iota)}$ divides into h and using

$$\varphi^{(j)} = \varphi(\zeta_\delta^{j-1}h) = \zeta_\delta + (\varphi^{(1)} - \zeta_\delta)\zeta_\delta^{j-1}, \tag{55}$$

we obtain an estimate for $r_{\text{abs}}^{(\iota)}$ based only on $r_{\text{abs}}^{(1)}$ and $\varphi^{(1)}$:

$$\iota\, r_{\text{abs}}^{(\iota)} = \varphi^{(1)} \cdot \ldots \cdot \varphi^{(\iota')}\, r_{\text{abs}}^{(1)} \quad \Rightarrow \quad r_{\text{abs}}^{(\iota)} = \frac{r_{\text{abs}}^{(1)}}{\iota}\prod_{j=1}^{\iota'}\varphi^{(j)}. \tag{56}$$

One can also compute $\varphi^{(1)}$ given $r_{\text{abs}}^{(1)}$ and $r_{\text{abs}}^{(\iota)}$ by solving the following implicit equation for $\varphi^{(1)}$ based on combining (55)-(56), which will be used later on in the tuning algorithm for Δt:

$$\varphi^{(1)} \cdot \big(\zeta_\delta + (\varphi^{(1)} - \zeta_\delta)\zeta_\delta\big) \cdot \ldots \cdot \big(\zeta_\delta + (\varphi^{(1)} - \zeta_\delta)\zeta_\delta^{\iota'-1}\big) = \iota\,\frac{r_{\text{abs}}^{(\iota)}}{r_{\text{abs}}^{(1)}}. \tag{57}$$

Inserting (56) in (54) yields the cost function

$$r_R(t+h) = r_0(1+2\zeta_Z)^{\iota}\zeta_A + \frac{r_{\text{abs}}^{(1)}}{\iota}\,\zeta_{A,Z}(\iota)\prod_{j=1}^{\iota'}\varphi^{(j)}, \tag{58}$$

which we minimize to obtain the optimal time step size

$$\Delta t_* = h\,\zeta_\delta^{\iota'_*} \quad \text{where} \quad \iota'_* = \operatorname*{arg\,min}_{\iota' \in \mathbb{N}}\, r_R(t+h). \tag{59}$$

For the evaluation, we simply increase ι' until the objective value $r_R(t+h)$ increases again as such a simple scalar formula does not require more sophisticated algorithms.

As a final step, we have to determine the finite horizon h for the evaluation of the cost function (58). The key element in the derivation of the optimization problem is the approximative evaluation (56) of $r_{\text{abs}}^{(\iota)}$ based on the gain φ (34). The proposed linear interpolation (50) reflects the actual progression of φ over Δt more accurately the closer the used gain $\varphi^{(1)}$ is to the limit gain $\lim_{\Delta t \to 0} \varphi(\Delta t) = \zeta_\delta^{q+1}$, see Cor. 2, which depends on the order q. Using a threshold value $\zeta_h(q)$, we determine h by

$$h = \min \tau \quad \text{such that} \quad \varphi^{(1)}(\tau) \geq \zeta_h(q). \tag{60}$$

Additionally, we restrict $\zeta_h(q)$ to be smaller than ζ_δ^{q+1} since this value is reached from below for small values of Δt as discussed in Sec. 4.3.

5.5. Automated parameter tuning algorithm

Let us now present Alg. 2, which enhances the reachable set computation shown in Alg. 1 by the adaptive parameter tuning methods introduced in this section. In order to reduce the computational effort, we utilize available information from previous steps for the adaptation of the time step size Δt and the abstraction order κ.

First, we update the finite horizon h (Line 1) by the value in (60). Using the finite horizon as the time step size Δt_k, we follow the procedure for the computation of the sets $\mathcal{R}_{\text{lin}}$ and $\mathcal{R}_{\text{abs}}$ known from Alg. 1, where the operations `linReachAdaptive` and `abstrSolAdaptive` contain the automated tuning of the propagation parameters η_{lin} and η_{abs} as described in Sec. 5.1. For conciseness, we comprise lines 7-11 from Alg. 1 by the operation `abstrErrLoop`. At the end of this computation, the operation `optDeltat` computes the optimal time step size Δt_*, using the just computed value $r_{\text{abs}}^{(1)} = \texttt{rad}\big(\mathcal{R}_{\text{abs}}(h)\big)$ and the gain φ_1 from the last step (Line 10). In the second iteration, we compute the sets $\mathcal{R}_{\text{lin}}$ and $\mathcal{R}_{\text{abs}}$ using $\Delta t_k = \Delta t_*$ and tune the abstraction order κ (Line 13) by the operation `tuneAbstrOrder`, comprising the method from Sec. 5.2. Additionally, we approximate the gain φ_1 (Line 16) by implicitly solving (57), denoted by the operator `estimateGain`, taking the estimates $r_{\text{abs}}^{(1)}$ and $r_{\text{abs}}^{(\iota_*)}$ for the finite

Algorithm 2 Adaptively-tuned reachable set computation for one step $k > 1$

Input: nonlinear function $f(z)$, algebraic equation $g(z)$, start set $\mathcal{R}(t_k)$, algebraic start set $\mathcal{R}^y(t_k)$, input set $\mathcal{U}$, abstraction order κ_k, gain of last step $\varphi^{(1)}_{k-1}$, finite horizon of last step h_{k-1}, $r^{(\iota_*)}_{\text{abs},k'}$ of step k' (last evaluation of (48)), set of global parameters ζ

Output: $\mathcal{R}(t_{k+1}), \mathcal{R}(\tau_{k+1}), \mathcal{R}^y(t_{k+1}), \kappa_{k+1}, h_k, \varphi^{(1)}_k$

```
1:  $h_k \leftarrow h_{k-1} \frac{\zeta_\delta - \zeta_h}{\zeta_\delta - \varphi^{(1)}_{k-1}}$, $\Delta t_k \leftarrow h_k$
2:  for $a = 1 : 2$ do
3:      $z^*(t_k) \leftarrow \texttt{linPoint}(\mathcal{R}(t_k), f, \mathcal{R}^y(t_k), g)$
4:      $w^{(x)}, w^{(y)}, C^{(x)}, C^{(y)}, D^{(x)}, D^{(y)} \leftarrow \texttt{taylor}\big(f(z), z^*(t_k), \kappa_k\big)$
5:      $w, A, B \leftarrow \texttt{linSys}(w^{(x)}, w^{(y)}, C^{(x)}, C^{(y)})$
6:      $\mathcal{R}_{\text{lin}}(t_{k+1}), \mathcal{R}_{\text{lin}}(\tau_{k+1}) \leftarrow \texttt{linReachAdaptive}(\mathcal{R}(t_k), w, A, B)$
7:      $\Psi, \mathcal{R}^y(t_{k+1}) \leftarrow \texttt{abstrErrLoop}(\mathcal{R}_{\text{lin}}(\tau_{k+1}), \overline{\Psi}, \kappa_k)$
8:      if $a = 1$ then
9:          $\mathcal{R}_{\text{abs}}(h), r^{(1)}_{\text{abs},k} \leftarrow \texttt{abstrSolAdaptive}(\Psi)$
10:         $\Delta t_* \leftarrow \texttt{optDeltat}(h_k, \varphi^{(1)}_k, r^{(1)}_{\text{abs},k})$, $\Delta t_k \leftarrow \Delta t_*$
11:     else
12:         $\mathcal{R}_{\text{abs}}(\Delta t_*), r^{(\iota_*)}_{\text{abs},k} \leftarrow \texttt{abstrSolAdaptive}(\Psi)$
13:         $\kappa_{k+1} \leftarrow \texttt{tuneAbstrOrder}(r^{(\iota_*)}_{\text{abs},k}, r^{(\iota_*)}_{\text{abs},k'})$
14:     end if
15: end for
16: $\varphi^{(1)}_k \leftarrow \texttt{estimateGain}(r^{(1)}_{\text{abs},k}, r^{(\iota_*)}_{\text{abs},k})$
17: $\mathcal{R}(t_{k+1}) = \mathcal{R}_{\text{lin}}(t_{k+1}) \boxplus \mathcal{R}_{\text{abs}}(\Delta t_*)$, $\mathcal{R}(\tau_{k+1}) = \mathcal{R}_{\text{lin}}(\tau_{k+1}) \boxplus \mathcal{R}_{\text{abs}}(\Delta t_*)$
18: $\mathcal{R}(t_{k+1}) \leftarrow \texttt{redAdaptive}\big(\mathcal{R}(t_{k+1})\big), \mathcal{R}(\tau_{k+1}) \leftarrow \texttt{redAdaptive}\big(\mathcal{R}(\tau_{k+1})\big)$
```

horizon and the optimal time step size, respectively, as input arguments. At the end of the step, the reachable sets $\mathcal{R}(t_{k+1})$ and $\mathcal{R}(\tau_{k+1})$ are computed and subsequently reduced by the operation `redAdaptive`, according to the method described in Sec. 5.3.

For the time step size, we first decrease an arbitrarily initialized Δt until the condition in (60) is met, yielding h with the associated error $\mathcal{R}_{\text{abs}}(h)$ and its scalar correspondence $r_{\text{abs}}^{(1)}$ as well as $\varphi^{(1)}$ in the process. Then, the operation `optDeltat` returns the optimal time step size Δt_*, after which the remainder of the step is executed as shown in Alg. 2.

Table 1: Setting of the global parameters ζ.

Approach	$\zeta_{T,\text{lin}}$	$\zeta_{T,\text{abs}}$	ζ_Z	ζ_K	$\zeta_h(q=0)$	$\zeta_h(q=1)$	ζ_δ
Linearization	0.0005	0.005	0.0005	0.90	0.85	0.76	0.90
Polynomialization	0.0005	0.005	0.0001	—	0.80	0.80	0.90

Finally, let us briefly discuss the set of global parameters ζ introduced in the respective tuning methods of the algorithm parameters in this section: Table 1 shows the values to which all global parameters ζ have been fixed. The first three $\zeta_{T,\text{lin}}, \zeta_{T,\text{abs}}$, and ζ_Z constitute threshold values representing sufficient accuracy of the tuned set operation. The value of ζ_K is a similarity measure for the comparison of two different abstraction orders. The final two values ζ_h (depending on the order q) and ζ_δ allow us to determine the finite horizon and candidate time step sizes for the optimization function tuning the time step size. Further development of the proposed tuning methods may change the value of a specific ζ, however, the current setting is justified by the tight reachable sets obtained for a wide variety of different nonlinear systems as shown in the next section.

6. Numerical Examples

In this section, we evaluate the adaptive parameter tuning approach presented in the previous section on all system classes introduced in Sec. 2. We first analyze two selected benchmark systems from the ARCH competition [55, 56] allowing us to compare our approach to expert-tuned state-of-the-art reachability tools. Then, a wide variety of different benchmark systems taken from various sources [17, 44, 57, 58] is used to provide a general overview of the performance. The adaptive parameter tuning approach was implemented in MATLAB R2022a and evaluated on an Intel® Core™ i7-9850 CPU @2.59GHz with 32GB memory. The following evaluation is based on [30], but considers more configurations of the ARCH benchmarks and extends the additional benchmark systems by including differential-algebraic and discrete-time systems.

6.1. ARCH benchmarks

In the ARCH competition, reachability tools compete with one another in solving benchmark systems, where the computation time and an accuracy measure are used for evaluation. Due to the lack of automated parameter tuning, each tool has to be tuned manually for each system. We select two benchmarks, namely the production-destruction benchmark (PRDE20) and the Laub-Loomis benchmark (LALO20), to assess the quality of our results in comparison with state-of-the-art tools. Let us first introduce the PRDE20 benchmark.

Example 1. *(PRDE20) This benchmark models a bio-geochemical reaction, describing an algal bloom transforming nutrients (x_1) into detritus (x_3) using phytoplankton (x_2) [59, Sec. 3]. The dynamics presented in [55, Sec. 3.1.1] also contain parametric uncertainty. Based on the initial state $x(0) = (9.98, 0.01, 0.01)^\top$ and the parameter $a = 0.3$, there are three configurations of this benchmark:*

1. *(Case I) Only uncertainty in the first initial state: $x_1(0) \in [9.50, 10.00]$.*
2. *(Case P) Only the parameter is uncertain: $a \in [0.296, 0.304]$.*
3. *(Case I&P) Uncertainty in the first initial state and the parameter: $x_1(0) \in [9.80, 10.00]$, $a \in [0.298, 0.302]$.*

The time horizon is $t_{end} = 100s$. □

Due to the small size of the initial set $\mathcal{X}^0$, a linearization approach already yields tight reachable sets in all three cases. Fig. 5 shows the reachable sets for case I, which serves as an illustrative example for the tuning of the algorithm parameters. The projections show a sharp turn (in the time interval

$t \in [10.6, 11.6]$) imposing strongly nonlinear behavior which is both preceded and succeeded by rather calm dynamics.

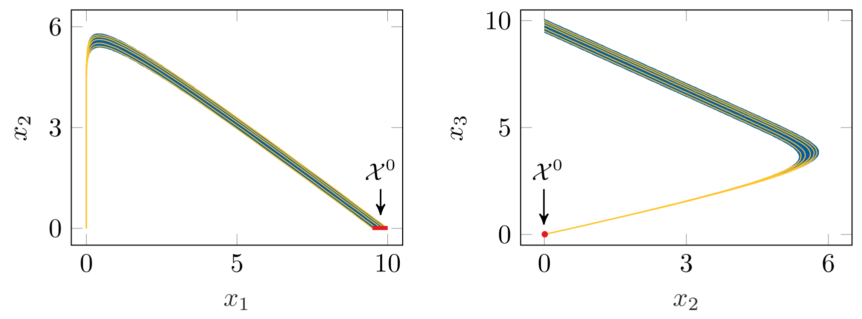

Figure 5: Projections of the reachable set $\mathcal{R}([0, t_{\mathrm{end}}])$ of Example 1, case I. Initial set in red, reachable sets in blue, single simulation runs in yellow.

Fig. 6 shows how the adaptive parameter tuning reacts to the change in the degree of nonlinearity: The left graph plots the time step size Δt over time, which reaches its minimum value $\Delta t \approx 0.012$ during the sharp turn. There, the optimization function reduces Δt, thus decreases the abstraction error in order to optimize the estimated over-approximation error at that time. Afterwards, the value gradually increases towards its maximum value $\Delta t \approx 0.4$, which exploits that the dynamics are better approximated in rather linear regions, yielding small abstraction errors even for relatively large time step sizes. The right graph plots the zonotope order ρ over time whose behavior can be explained in a similar way: At the sharp turn, more generators have to be kept in order to avoid inducing large over-approximations, yielding a maximum zonotope order of $\rho = 20$. As a consequence of the calmer dynamics after the sharp turn, the complexity of the shape decreases as we observe in Fig. 5: The sets after the turn are much more straightened compared to the "bent" sets at the time of the turn. This reduces the number of generators required for an accurate representation of the set and thus lowers the zonotope order.

The propagation parameters η_{lin} and η_{abs} do not change much over time as we have $\eta_{\mathrm{lin}} \in \{4, 5, 6\}$ and $\eta_{\mathrm{abs}} \in \{2, 3\}$, where the respective maxima are reached at the sharp turn. The tuning of the abstraction order κ results in

$\kappa = 2$ at the beginning until $t \approx 17.3$ and $\kappa = 1$ for the remainder of the time horizon, thereby confirming the rather linear dynamics after the sharp turn.

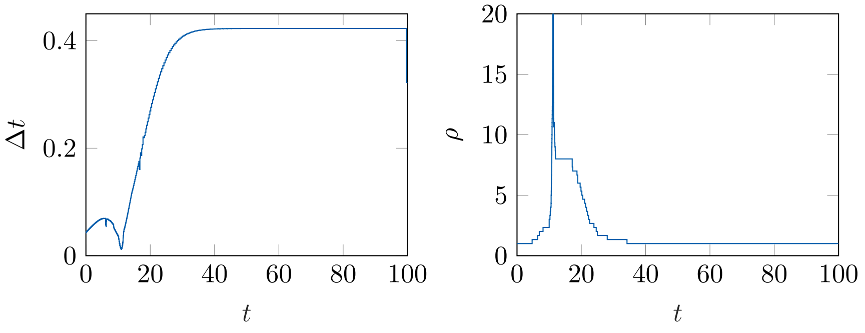

Figure 6: Time step size Δt and zonotope order ρ of Example 1, case I, over time.

Table 2 allows us to compare the results obtained by our adaptive parameter tuning approach with state-of-the-art reachability tools for all three cases specified in Example 1. The obtained accuracy ranks among the best, even topping the chart in cases P and I&P. The computation time is average in all cases, partly caused by the speed discrepancy in programming languages as C++ and Julia are known to operate faster than MATLAB. The comparison with CORA in particular shows competitiveness since our computation is faster due to the large ratio of the largest to the smallest time step size saving many time steps. Next, we consider the Laub-Loomis benchmark.

Example 2. *(LALO20) The dynamics of this benchmark [56, Sec. 3.3.1] represent changes in enzymatic activities introduced in [60, (1-7)]. The initial set is given by* $\mathcal{X}^0 = [x(0) - W, x(0) + W]$*, where* $x(0) = (1.2, 1.05, 1.5, 2.4, 1, 0.1, 0.45)^\top$ *is enlarged by either of the uncertainties* $W \in \{0.01, 0.05, 0.1\}$*, representing configurations of increasing difficulty. The time horizon is* $t_{end} = 20s$*.* □

While a linearization approach still suffices for small ($W = 0.01$) and moderate ($W = 0.05$) sizes of the initial set $\mathcal{X}^0$, the largest size ($W = 0.1$) can only be solved using a polynomialization approach. For conciseness, we only plot the time step size Δt and the zonotope order ρ over time for all three cases ($W \in \{0.01, 0.05, 0.1\}$ in blue, black, and yellow, respectively) in Fig. 7: All

Table 2: ARCH benchmark PRDE20: Comparison of our approach with state-of-the-art reachability tools in terms of computation time and the tightness measurement $\mu = \texttt{vol}\big(\texttt{box}\big(\mathcal{R}(t_{\text{end}})\big)\big)$ as in [55].

Benchmark	**PRDE20**					
Tool (Language)	**Case I**		**Case P**		**Case I&P**	
	Time	μ	Time	μ	Time	μ
Alg. 2 (Matlab)	10.8s	7.8e−21	12.4s	**3.5e−24**	18.6s	**8.6e−24**
Ariadne (C++)	8.6s	1.7e−13	79s	2.3e−17	39s	1.0e−13
CORA (Matlab)	16s	**1.2e−21**	28s	2.2e−23	28s	2.0e−23
DynIbex (C++)	12s	3.9e−17	13s	4.8e−17	26s	1.2e−17
Flow* (C++)	4.1s	8.0e−21	9s	1.4e−22	5.2s	4.8e−21
Isabelle/HOL (SML)	11s	3.3e−20	12s	7.3e−21	26s	2.6e−20
JuliaReach (Julia)	**1.5s**	3.3e−20	**3.9s**	6.5e−21	**3.0s**	1.0e−20

curves for Δt increase similarly over time in multiple waves. We also note an offset between the different configurations, showing that a smaller set size allows larger time step sizes and vice versa. The curve of the zonotope order ρ for the case $W = 0.1$ (yellow) differs from the one for $W \in \{0.05, 0.1\}$ because the polynomialization approach uses non-convex sets. The vertical drops of ρ are caused by the restructuring operation [47, Prop. 17], where all independent generators are first reduced and then converted to dependent generators for reasons of computational accuracy in subsequent steps. Using a linearization approach, the curves for ρ reach their maximum at the end of the time horizon at values of 10 and 20 for $W = 0.01$ and $W = 0.05$, respectively, which keeps the set operations efficient without compromising the tightness of the reachable sets.

The evaluation of the LALO20 benchmark in both computation time and accuracy is shown in Table 3, offering a similar picture as for the PRDE20 benchmark: Again, our computation times are only average across all tools, mainly due to the costly evaluation of the abstraction error Ψ as well as the computationally demanding reduction of the set representation size for the sys-

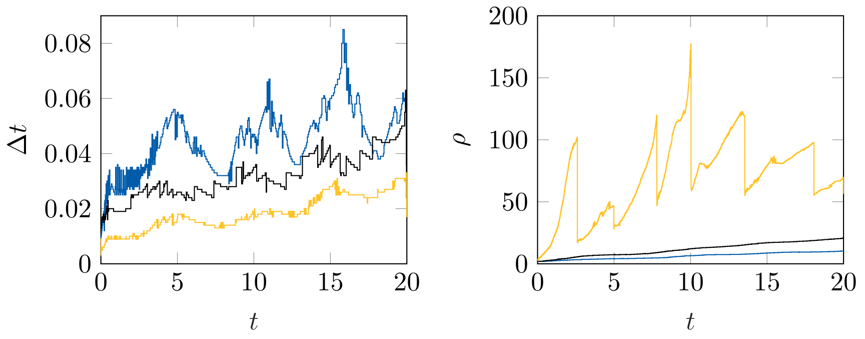

Figure 7: Time step size Δt and zonotope order ρ of Example 2 over time: Different cases $W = \{0.01, 0.05, 0.1\}$ in blue, black, and yellow, respectively.

tem dimension $n = 7$. In contrast, the accuracy is better than most others, being co-leader for the smallest size $W = 0.01$ and second for the largest size $W = 0.1$. This demonstrates the competitiveness of our adaptive parameter tuning for both linearization and polynomialization approaches.

Table 3: ARCH benchmark LALO20: Comparison of our approach with state-of-the-art reachability tools in terms of computation time and the tightness measurement $\mu = l_4$, where $l = d\big(\texttt{box}\big(\mathcal{R}(t_{\text{end}})\big)\big)$ as in [56].

Benchmark	**LALO20**					
Tool (Language)	$W = 0.01$		$W = 0.05$		$W = 0.1$	
	Time	μ	Time	μ	Time	μ
Alg. 2 (Matlab)	3.9s	**0.004**	7.8s	0.049	91s	0.068
Ariadne (C++)	5.7s	0.01	11s	0.031	31s	0.071
CORA (Matlab)	1.9s	0.005	8.4s	0.035	38s	0.116
DynIbex (C++)	10s	0.01	27s	0.40	1851s	2.07
JuliaReach (Julia)	**1.1s**	**0.004**	**1.5s**	**0.017**	**1.4s**	**0.033**
Kaa (Python)	238s	22	253s	23	257s	49

6.2. Further benchmarks

After the detailed discussion of the ARCH benchmarks, we now analyze the performance on a broader range of benchmarks, also including differential-algebraic (DA) and discrete-time systems. Table 4 provides some information about the benchmarks, such as the system dimension n and algebraic dimension n_a as well as the time horizon t_{end} and the initial set $\mathcal{X}^0$. The benchmarks range from standard models like the van-der-Pol oscillator over chaotic systems, such as the Roessler attractor and Lorenz attractor, to higher-dimensional biologically and mechanically inspired models. Both differential-algebraic models are power systems, namely a 3-bus system and a single machine infinite bus (SMIB) system, where the algebraic equations originate from the network constraints. The SMIB system has different dynamics for standard operation and fault scenario caused by a loss in the network connection occurring at $t \in [0.01, 0.02]$. Finally, we discretized a six-dimensional water tank benchmark whose dynamics are based on Toricelli's Law using a time step size of $\Delta t = 0.05\mathrm{s}$.

The tightness of the reachable sets is quantified using two different metrics: First, we provide the longest edge of the box over-approximation of the final set $\mathcal{R}(t_{\mathrm{end}})$, namely,

$$d_{\max} = \max_{i \in \{1,\dots,n\}} d_i\big(\mathtt{box}\big(\mathcal{R}(t_{\mathrm{end}})\big)\big). \tag{61}$$

Second, we use the ratio of under-approximation to over-approximation proposed in [68, Sec. VI.]

$$\gamma_{\min} = \min_{i \in \{1,\dots,n\}} \frac{d_i\big(\mathtt{box}\big(\mathcal{R}_{\mathrm{sim}}(t_{\mathrm{end}})\big)\big)}{d_i\big(\mathtt{box}\big(\mathcal{R}(t_{\mathrm{end}})\big)\big)}, \tag{62}$$

where $\mathcal{R}_{\mathrm{sim}}(t_{\mathrm{end}})$ denotes the set of states at t_{end} of 1000 simulation runs. The tightness increases for $\gamma_{\min} \rightarrow 1$ as the under-/over-approximation approach one another. Space constraints prevent a detailed discussion of every result, which is why we will discuss general tendencies as well as unexpected results.

Table 5 shows the results for all systems from Table 4 using a linearization and a polynomialization approach with adaptively tuned algorithm parameters. The linearization approach is by construction limited to systems with only mild

Table 4: List of considered benchmarks: n: system dimension, n_a: algebraic dimension, t_{end}: time horizon, $\mathcal{X}^0$: initial set.

Benchmark	n	n_a	t_{end}	$\mathcal{X}^0$
Jet Engine [61, (19)]	2	–	8	$[0.9, 1.1]^n$
van der Pol [17, Sec. VII]	2	–	6.74	$\big([1.30, 1.50]\ [2.35, 2.45]\big)^\top$
Brusselator [44, Ex. 3.4.1]	2	–	5	$\big([0.9, 1.0]\ [0.0, 0.1]\big)^\top$
Roessler [62, (2)]	3	–	6	$\big([-0.2, 0.2]\ [-8.6, -8.2]\ [-0.2, 0.2]\big)^\top$
Lorenz [63, (25-27)]	3	–	2	$\big([14.9, 15.1]\ [14.9, 15.1]\ [34.9, 35.1]\big)^\top$
Spring-Pendulum [44, Ex. 3.3.12]	4	–	1	$\big([1.1, 1.3]\ [0.4, 0.6]\ [0.0, 0.1]\ [0.0, 0.1]\big)^\top$
Lotka-Volterra [64, (1)]	5	–	5	$[0.90, 1.00]^n$
Biological Model [65]	7	–	2	$[0.99, 1.01]^n$
Genetic Model [66, (1)]	9	–	0.1	see [57, Sec. V.]
3-Bus [67, Sec. 4]	2	6	5	$\big([379.90, 380.10]\ [0.69, 0.71]\big)^\top$
SMIB [58, Sec. 2.5.1.2]	2	4	0.23	$\big([0.65075, 0.66675]\ [0.008, 0.008]\big)^\top$
Tank-DT [17, Sec. VII]	6	–	80	$\big([1.8, 2.2]\ [3.8, 4.2]\ [3.8, 4.2]\ [1.8, 2.2]\ [9.8, 10.2]\ [3.8, 4.2]\big)^\top$

nonlinearities, leading to low values of $\gamma_{\min}$ for the Roessler attractor and the van-der-Pol oscillator (similar results for the latter have already been discussed in [47, Sec. 4]). The tightness is still satisfactory in most cases, especially where $\gamma_{\min} > 0.7$. Moreover, the computation times and tightness measures are similar over increasing system dimension, showing the scalability of our proposed tuning methods. In contrast, the polynomialization approach yields both higher computation times and improved accuracy according to the tightness measures $d_{\max}$ and $\gamma_{\min}$. Both approaches exploit the range of different time step sizes of 1-2 orders of magnitude on average. The total number of steps in the analysis is drastically reduced, thus significantly speeding up the computation compared to fixed time step sizes.

For the linearization approach, the maximum zonotope order $\rho_{\max}$ is often rather low (between 10 and 20); for other cases, it should be noted that the

Table 5: Evaluation of nonlinear benchmark systems using an adaptively tuned linearization and polynomialization approaches: $[\Delta t_{\min}, \Delta t_{\max}]$: range of time step sizes, $\rho_{\max}$: max. zonotope order, $d_{\max}$ and $\gamma_{\min}$: measurements by (61) and (62).

Benchmark	**Linearization Approach**					**Polynomialization Approach**				
	Time	$[\Delta t_{\min}, \Delta t_{\max}]$	$\rho_{\max}$	$d_{\max}$	$\gamma_{\min}$	Time	$[\Delta t_{\min}, \Delta t_{\max}]$	$\rho_{\max}$	$d_{\max}$	$\gamma_{\min}$
Jet Engine	2.5s	$[0.007, 0.117]$	13.5	0.0562	0.5594	11.8s	$[0.002, 0.049]$	26	0.0395	0.7955
van der Pol	5.5s	$[0.002, 0.051]$	16.5	1.79	0.2004	26s	$[0.0005, 0.0127]$	47.5	0.5234	0.6621
Brusselator	2.0s	$[0.011, 0.062]$	80.5	0.075	0.7901	9.5s	$[0.004, 0.021]$	219.5	0.065	0.9469
Roessler	2.7s	$[0.006, 0.047]$	10.33	4.34	0.1725	13.7s	$[0.0022, 0.0328]$	20.33	2.47	0.6623
Lorenz	4.1s	$[0.0004, 0.0098]$	10	0.268	0.8337	10.8s	$[0.0004, 0.0067]$	46	0.237	0.9562
Spring-Pendulum	4.5s	$[0.006, 0.022]$	12.75	0.522	0.6484	10.7s	$[0.002, 0.009]$	42.75	0.424	0.7884
Lotka-Volterra	1.0s	$[0.010, 0.107]$	12.2	0.083	0.8722	4.1s	$[0.003, 0.078]$	195.2	0.074	0.9794
Biological Model	1.8s	$[0.004, 0.019]$	44.71	0.117	0.7115	10.7s	$[0.001, 0.008]$	182.14	0.094	0.9163
Genetic Model	0.7s	$[0.0005, 0.0023]$	6	5.55	0.7939	2.7s	$[0.0002, 0.0010]$	37.11	5.30	0.9509
3-Bus	4.3s	$[0.015, 0.054]$	13	2.28	0.6791	–	–	–	–	–
SMIB	36s	$[0.00005, 0.00200]$	6.5	0.0004	0.3059	–	–	–	–	–
Tank-DT	13s	fixed to 0.05	28.67	0.6246	0.7517	47s	fixed to 0.05	35.5	0.6044	0.7689

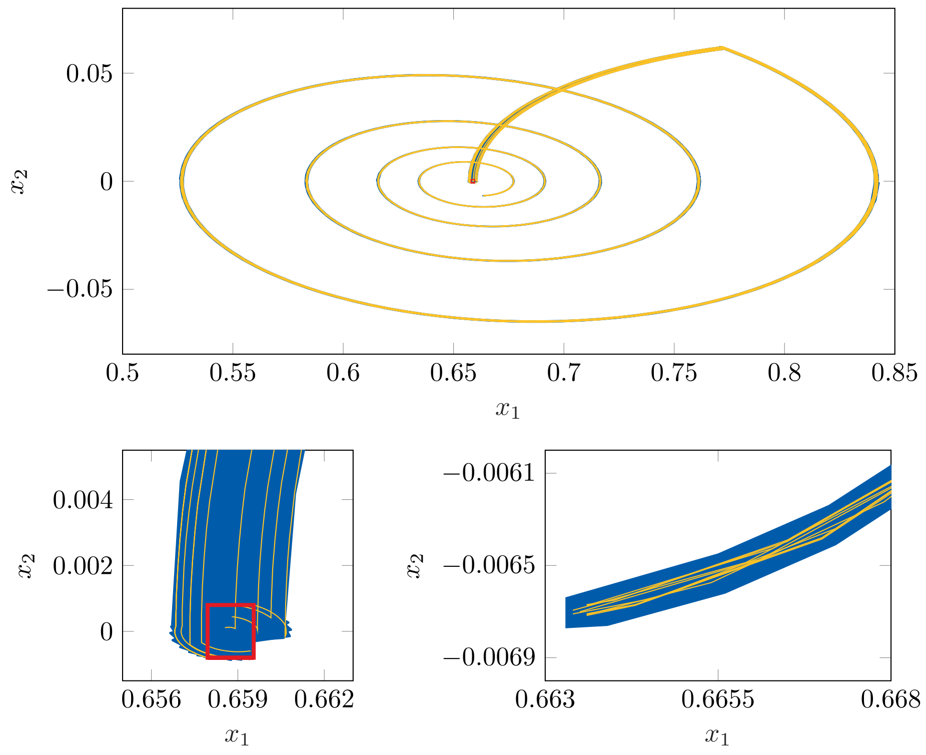

Figure 8: Reachable sets of the SMIB system, see Table 4, over the full time horizon (top), at the start (bottom left), and at the end (bottom right). The divergence from the initial set (red) is caused by the fault scenario, after the return to the normal operation mode the system behavior stabilizes. The reachable sets (blue) are largely covered by single simulation runs (yellow).

highest order may only last for a few steps as shown in Fig. 6 and therefore does not pose major problems to the efficiency of the computations. For the polynomialization approach, higher zonotope orders are reached because the reduction of polynomial zonotopes is too over-approximative to allow for substantial reductions. This also entails an increase in computation time since the set operations then become more costly for larger set representation sizes, as well as an increase in accuracy, where we note that five systems reach a value of $\gamma_{\min} > 0.9$. This leads to the conclusion that the reduction of the set representation within the polynomialization approach is its limiting factor.

Our adaptive parameter tuning would greatly benefit, especially for polynomial zonotopes, from improvements in order reduction techniques similar to our considerations presented in Sec. 3 for (linear) zonotopes, as updated methods can simply replace existing ones due to the modularity of our tuning framework.

Finally, we discuss the results for the DA systems, for which there does not yet exist a polynomialization approach. The evaluation of both DA systems does not show major differences to standard nonlinear systems, except for the high computation time and low tightness measure $\gamma_{\min}$ for the SMIB system. Fig. 8 shows the reachable sets, putting the low value for $\gamma_{\min}$ in perspective, as the enclosure of the simulated trajectories is still tight. In the discrete-time example, the time step size is fixed by definition, yielding a total number of 2000 steps. The suitability of the remaining tuning methods is shown by the fairly low value for $\rho_{\max}$ and high value for $\gamma_{\min}$.

In summary, our evaluation shows that the presented methods for adaptive parameter tuning allow us to obtain tight reachable sets in a broad variety of different systems. Due to the fully automatic tuning, the computation of the reachable sets is executed in a single run as opposed to the many trial-and-error runs in manual tuning.

7. Conclusion

We presented the first fully-automatic reachability algorithm for nonlinear systems. To this end, the fundamentals of the two main wrapping effects in reachability analysis of nonlinear systems have been thoroughly investigated: First, we presented an exhaustive derivation of various bounds for the Hausdorff distance between a zonotope and its box over-approximation, with associated generator selection strategies for the order reduction of zonotopes. Second, a rigorous examination of the set of abstraction errors accounting for higher-order nonlinearities led to the introduction of a *gain order* which describes the effect of the time step size on the local and global abstraction error in the analysis. These theoretical insights were then utilized in our adaptive parameter tuning algorithm, most notably for the derivation of an optimization function to tune the time step size. The evaluation on multiple nonlinear system classes showed competitiveness with state-of-the-art reachability tools, as well as an efficient computation of tight reachable sets in a variety of further benchmark problems. Our approach requires no longer expert knowledge about the reachability algorithm, which greatly simplifies the usage of reachability analysis in a broad variety of possible applications.

Appendix A. Proof of Theorem 1

Let us express each point $x_{\text{box}} \in \mathcal{Z}_{\text{box}}$ as

$$x_{\text{box}} = M^{(1)}|g^{(1)}| + ... + M^{(\gamma)}|g^{(\gamma)}| \, ,$$

where each $M^{(p)}$ is a diagonal matrix with diagonal entries $\mu_i^{(p)} \in [-1,1]$ where $i \in \{1,...,n\}$. The function $\|x_{\text{box}} - x\|_2$ is convex w.r.t. $\mu_i^{(p)}$ allowing us to restrict our attention to the cases where $\mu_i^{(p)} \in \{-1,1\}$, since by the Bauer maximum principle (see [69]), the maximum of a convex function over the polytope $[-1,1]^{\gamma n}$ is always reached at one of its vertices, i.e., some element of $\{-1,1\}^{\gamma n}$. Let us write the difference between any $x_{\text{box}} \in \mathcal{Z}_{\text{box}}$ and $x \in \mathcal{Z}$ as

$$x_{\text{box}} - x = \big(M^{(1)}|g^{(1)}| - \alpha_1 g^{(1)}\big) + ... + \big(M^{(\gamma)}|g^{(\gamma)}| - \alpha_\gamma g^{(\gamma)}\big) \tag{A.1}$$

where $\forall p \in \{1,...,\gamma\} : \alpha_p \in [-1,1]$. Note that the minimum of $\|x_{\text{box}} - x\|_2$ w.r.t. α_p does not have a closed-form formula in general [70, Sec. 9]. However, one can obtain a bound on $x_{\text{box}} - x$ by choosing a specific α_p for each $g^{(p)}$. The bounds ω_{max} (15), ω_{rad} (16), and ω_{cut} (17) are obtained by different choices for α_p and subsequently derived in detail:

Bound ω_{max}: Let us start with the following choice for α_p:

$$\alpha_p = \mu_{i^*}^{(p)} \operatorname{sgn}\big(g_{i^*}^{(p)}\big), \tag{A.2}$$

with an individual i^* for each p as in Theorem 1. Consequently, we can eliminate the largest possible entry in

$$v^{(p)} = M^{(p)}|g^{(p)}| - \alpha_p g^{(p)}, \tag{A.3}$$

for which we obtain the bound

$$v_i^{(p)} \in \begin{cases} \big[-2|g_i^{(p)}|, 2|g_i^{(p)}|\big], & \text{if } i \neq i^* \\ 0, & \text{otherwise,} \end{cases}$$

which we can rewrite to $v_i^{(p)} \in [-2\widehat{g}_i^{(p)}, 2\widehat{g}_i^{(p)}]$ using (18). Applying (A.2) to each generator, we obtain the bounds

$$x_{\text{box}} - x = v^{(1)} + ... + v^{(\gamma)} \in [-2\widehat{z}, 2\widehat{z}]\,,$$

where $\widehat{z} = \widehat{g}^{(1)} + ... + \widehat{g}^{(\gamma)}$ and ultimately,

$$\|x_{\text{box}} - x\|_2 \leq \|2\widehat{z}\|_2 = 2\|\widehat{z}\|_2\,.$$

The above bound holds for any $x_{\text{box}} \in \mathcal{Z}_{\text{box}}$, which fulfills the assumption of [30, Lemma 3.1] and thus proves that $d_H(\mathcal{Z}, \mathcal{Z}_{\text{box}}) \leq \omega_{\max}$.

Bound ω_{rad}: Another way to choose α_p is to find an optimal minimum of $\|v^{(p)}\|_2$ defined in (A.3) w.r.t. α_p and then use the inequality

$$\|x_{\text{box}} - x\|_2 \leq \sum_{p=0}^{\gamma} \big\|v^{(p)}\big\|_2.$$

Since the exact minimum of $\|v^{(p)}\|_2$ equal to the minimum of $\|v^{(p)}\|_2^2$, we insert (A.3) into the squared expression, differentiate w.r.t α_p, and solve for α_p, yielding the minimizer

$$\alpha_p^* = \frac{\sum_{i=1}^n \mu_i^{(p)} \operatorname{sgn}(g_i^{(p)}) \left(g_i^{(p)}\right)^2}{\big\|g^{(p)}\big\|_2^2}.$$

By inserting the expression for α_p^* back into (A.3), one obtains

$$\min_{x\in\mathcal{Z}} \big\|v^{(p)}\big\|_2 = \big\|g^{(p)}\big\|_2 \sqrt{1 - \left(\frac{\sum_{i=1}^n \mu_i^{(p)} \operatorname{sgn}(g_i^{(p)}) \left(g_i^{(p)}\right)^2}{\big\|g^{(p)}\big\|_2^2}\right)^2}.$$

Since we maximize this expression w.r.t. $\mu_i^{(p)} \in \{-1, 1\}$ and $v^{(l)}$ does not depend on any $\mu_i^{(p)}$ if $p \neq l$, we can replace expressions such as $\mu_i^{(p)} \operatorname{sgn}(g_i^{(p)})$ by

$\widehat{\mu}_i^{(p)} \in \{-1, 1\}$, which yields

$$\max_{x_{\text{box}} \in \mathcal{Z}_{\text{box}}} \min_{x \in \mathcal{Z}} \|x_{\text{box}} - x\|_2 \leq \max_{\widehat{\mu}_i^{(l)} \in \{-1,1\}^\gamma} \sum_{p=0}^{\gamma} \left\|g^{(p)}\right\|_2 \sqrt{1 - \left(\frac{\sum_{i=1}^n \widehat{\mu}_i^{(p)} \left(g_i^{(p)}\right)^2}{\left\|g^{(p)}\right\|_2^2} \right)^2}.$$

Each summand depends on exactly one $\mu_i^{(p)}$, thus the sum and the maximum commute, yielding

$$\max_{x_{\text{box}} \in \mathcal{Z}_{\text{box}}} \min_{x \in \mathcal{Z}} \|x_{\text{box}} - x\|_2 \leq \sum_{p=0}^{\gamma} \left\|g^{(p)}\right\|_2 \sqrt{1 - \min_{\mu \in \{-1,1\}^n} \left(\frac{\sum_{i=1}^n \mu_i \left(g_i^{(p)}\right)^2}{\left\|g^{(p)}\right\|_2^2} \right)^2}. \tag{A.4}$$

To get a new bound on the Hausdorff distance, we can therefore restrict ourselves to finding a lower-approximation of

$$C_p := \min_{\mu \in \{-1,1\}^n} \left[\sum_{i=1}^n \mu_i \left(g_i^{(p)}\right)^2 \right]^2 = \min_{\mu \in \{-1,1\}^n} \sum_{i=1}^n \sum_{j=1}^n \mu_i \mu_j \left(g_i^{(p)}\right)^2 \left(g_j^{(p)}\right)^2. \tag{A.5}$$

Using the trivial estimate $C_p \geq 0$, we can simplify (A.4) to

$$\max_{x_{\text{box}} \in \mathcal{Z}_{\text{box}}} \min_{x \in \mathcal{Z}} \|x_{\text{box}} - x\|_2 \leq \sum_{p=1}^{\gamma} \left\|g^{(p)}\right\|_2,$$

which proves that ω_{rad} is a valid over-approximation of the Hausdorff distance.

Bound ω_{cut}: For our final bound, we reformulate (A.5) to

$$C_p = \min_{\mu \in \{-1,1\}^n} \sum_{i=1}^n \left(g_i^{(p)}\right)^4 + \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n \mu_i \mu_j \left(g_i^{(p)}\right)^2 \left(g_j^{(p)}\right)^2.$$

Since $\mu_i \in \{-1, 1\}$, and thus $\forall i, j = \{1, ..., n\} : \mu_i \mu_j \geq -1$, we deduce that

$$C_p \geq 2 \sum_{i=1}^n \left(g_i^{(p)}\right)^4 - \left\|g^{(p)}\right\|_2^4,$$

which yields

$$\max_{x_{\text{box}} \in \mathcal{Z}_{\text{box}}} \min_{x \in \mathcal{Z}} \|x_{\text{box}} - x\|_2 \leq \sum_{p=1}^{\gamma} \left\|g^{(p)}\right\|_2 \sqrt{2 - 2 \frac{\sum_{i=1}^n \left(g_i^{(p)}\right)^4}{\left\|g^{(p)}\right\|_2^4}},$$

proving that ω_{cut} is also an over-approximation of the Hausdorff distance. □

Appendix B. Proofs for Sec. 4

Proof of Lemma 2. (inspired by [32, p. 318, Theorem 12.2], with a few adaptations due to the set-based nature of the computations)

For simplicity, we will ignore all order reduction operations. In that case, adding an arbitrary zonotope $\mathcal{Z}$ centered at 0 to the initial set $\mathcal{X}^0$ does not influence the expansion point z^* (see (6)), thus the set of particular solutions $\mathcal{P}([t_0, t_0 + \Delta t])$ is unaffected (see also [50, Eq. (3.5)]). Since $\mathcal{R}_{\text{abs}}(\Delta t; t_k, \mathcal{R}(t_k))$ is either a zonotope centered at the origin, or can w.l.o.g. be over-approximated by one, by Def. 4 we can write

$$\forall k \in \{0, 1, ..., N\}: \quad \mathcal{R}(t_{k+1}) = e^{A\Delta t}\mathcal{R}(t_k) \oplus \mathcal{P}(\tau_k) \boxplus \mathcal{R}_{\text{abs}}(\Delta t; t_k, \mathcal{R}(t_k)), \tag{B.1}$$

$$\widehat{\mathcal{R}}(t_{k+1}) = e^{A\Delta t}\widehat{\mathcal{R}}(t_k) \oplus \mathcal{P}(\tau_k). \tag{B.2}$$

Crucially, both (B.1) and (B.2) share the same term $\mathcal{P}(\tau_k)$. Therefore, the global abstraction error ε_{k+1} after $k+1$ steps defined in Def. 4 may be written as

$$\begin{aligned}
\varepsilon_{k+1} = \; & \Big\| d\big(\texttt{box}\big(\mathcal{R}(t_{k+1}) \ominus \widehat{\mathcal{R}}(t_{k+1})\big)\big) \Big\|_\infty \\
\overset{\text{(B.1) and (B.2)}}{=} & \Big\| d\big(\texttt{box}\big(e^{A\Delta t}\mathcal{R}(t_k) \oplus \mathcal{P}(\tau_k) \boxplus \mathcal{R}_{\text{abs}}(\Delta t; t_k, \mathcal{R}(t_k)) \ominus e^{A\Delta t}\widehat{\mathcal{R}}(t_k) \ominus \mathcal{P}(\tau_k)\big)\big) \Big\|_\infty \\
\overset{\mathcal{P}(\tau_k)\ominus\mathcal{P}(\tau_k)=\{0\}}{=} & \Big\| d\big(\texttt{box}\big(e^{A\Delta t}(\mathcal{R}(t_k) \ominus \widehat{\mathcal{R}}(t_k)) \boxplus \mathcal{R}_{\text{abs}}(\Delta t; t_k, \mathcal{R}(t_k), \Delta t)\big)\big) \Big\|_\infty \\
\overset{\text{Triangle ineq.}}{\leq} & \Big\| d\big(\texttt{box}\big(e^{A\Delta t}(\mathcal{R}(t_k) \ominus \widehat{\mathcal{R}}(t_k))\big)\big) \Big\|_\infty + \Big\| d\big(\texttt{box}\big(\mathcal{R}_{\text{abs}}(\Delta t; t_k, \mathcal{R}(t_k))\big)\big) \Big\|_\infty .
\end{aligned}$$

With a few simple calculations one can show that In order to make this bound independent of Δt, we can use the fact that $\big\| \sum_{i=0}^{\infty} \frac{A^{i+1}\Delta t^i}{(i+1)!} \big\|_\infty$ is continuous w.r.t. Δt, and is thus Lipschitz continuous over the bounded domain $\Delta t \in [0, t_{\text{end}}]$ with a Lipschitz constant $\widehat{L} \geq 0$ that may depend on t_{end} but not Δt. Combining this again with $\Delta t \leq t_{\text{end}}$, we obtain a bound

$$\Big\| \sum_{i=0}^{\infty} \frac{A^{i+1}\Delta t^i}{(i+1)!} \Big\|_\infty \leq \widehat{L}\Delta t \leq \widehat{L} t_{\text{end}} =: L,$$

where L is now a constant that is independent of Δt. This implies the following iterative bound on ε_{k+1}:

$$\varepsilon_{k+1} \leq (1 + L\Delta t)\varepsilon_k + \varepsilon_{k,\mathrm{loc}}.$$

From this point onwards, we can use the same argument as in [32, p. 318, Theorem 12.2] to show that

$$\varepsilon_k \leq \frac{\max_{1\leq \ell \leq N} \varepsilon_{\ell,\mathrm{loc}}}{\Delta t} \frac{1}{L}(e^{Lt_{\mathrm{end}}} - 1).$$

The coefficient $\frac{1}{L}(e^{Lt_{\mathrm{end}}} - 1)$ may depend on t_{end}, but not N nor Δt, which yields (30). $\square$

Proof of Prop. 1. Let h be arbitrary, but fixed. Let $\mathcal{S} \subset \mathbb{R}^n$ be some bounded, centrally symmetric set with center c. Then we have

$$d_i\big(\mathtt{box}(\mathcal{S})\big) = \max_{s\in\mathcal{S}} s_i - \min_{s\in\mathcal{S}} s_i = 2 \max_{s\in\mathcal{S}} |s_i - c_i|. \tag{B.3}$$

After inserting the definition $\Psi(h) = \langle c(h), G(h)\rangle_Z$ in (32) and extracting the center $\widehat{c} := \sum_{p=0}^{\infty} \frac{h^{p+1}}{(p+1)!} A^p(h)c(h)$, we can apply (B.3) to (32) to obtain

$$d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(h)\big)\big) = 2 \max_{\substack{\beta^{(p)}\in[-1,1]^m\\ p\in\mathbb{N}_0}} \left| \sum_{p=0}^{\infty} \frac{h^{p+1}}{(p+1)!} A^p G(h)\beta^{(p)} \right|_i .$$

Since the maximization term is convex w.r.t. the $\beta^{(p)}$, we can change the domain of $\beta^{(p)}$ from $[-1,1]^m$ to $\{-1,1\}^m$ by the Bauer maximum principle (see [69]). Each summand depends on exactly one $\beta^{(p)}$, thus we obtain

$$\begin{aligned} d_i\big(\mathtt{box}\big(\mathcal{R}^{\infty}_{\mathrm{abs}}(h)\big)\big) &= 2 \sum_{p=0}^{\infty} \max_{\beta\in\{-1,1\}^m} \left| \frac{h^{p+1}}{(p+1)!} A^p G(h)\beta \right|_i \\ &= 2 \sum_{p=0}^{\infty} \frac{h^{p+1}}{(p+1)!} \|A^p G(h)\|_{1,i}, \end{aligned}$$

using the fact that the maximum of $|M\beta|_i$ for a matrix M and $\beta \in \{-1,1\}^m$ can easily be seen to be $\|M\|_{1,i}$. $\square$

Proof of Lemma 3. The fact that $G(h)$ is smooth follows for example from [18, p. 4] since f is smooth. From this, it follows from the Taylor expansion of $[A^pG(h)]_i$ that for any arbitrarily large N, there exists an expansion of the form

$$[A^pG(h)]_i = \sum_{j=0}^{N} c_j h^j + \epsilon(h), \tag{B.4}$$

where $c_j \in \mathbb{R}$ for all $j \in \{1, ..., N\}$ and $\epsilon(h) = \mathcal{O}(h^{N+1})$. Let j be the smallest index such that $c_j \neq 0$, and define $q_i^{(p)}$ to be this index. If for all $N \in \mathbb{N}_0$ such an index does not exist, it trivially follows that $[A^pG(h)]_i \equiv 0$, and in that case we can set $q_i^{(p)} = \infty$. If $q_i^{(p)} < \infty$, we may rewrite (B.4) for all $N > q_i^{(p)}$ as

$$[A^pG(h)]_i = c_{q_i^{(p)}} h^{q_i^{(p)}} + \sum_{j=q_i^{(p)}+1}^{N} c_j h^j + \epsilon(h).$$

Clearly, since $\epsilon(h) = \mathcal{O}(h^{N+1})$ this function may be written as $\epsilon(h) = h^{q_i^{(p)}}\widehat{\epsilon}(h)$ where $\widehat{\epsilon}(h) = \mathcal{O}(h^{N+1-q_i^{(p)}})$, and more specifically $\widehat{\epsilon}(h) = \mathcal{O}(h)$. Therefore, by defining $a_i^{(p)} = c_{q_i^{(p)}}$ and $b_i^{(p)}(h) = \sum_{j=q_i^{(p)}+1}^{N} c_j h^{j-q_i^{(p)}} + \widehat{\epsilon}(h)$, we obtain the form (36).

Finally, the function $Q_i^{(p)}(h)$ is easily seen to be non-negative since $\|\cdot\|_1$ is non-negative, and piecewise smooth since $\|\cdot\|_1$ is smooth almost everywhere, and $b_i^{(p)}(h)$ is smooth since G(h) is smooth. □

Proof of Prop. 2. We begin by proving (42). Differentiating φ_i (34) using Lemma 3 yields

$$\begin{aligned}\frac{\mathrm{d}\varphi_i(h;\delta)}{\mathrm{d}h} &= \frac{\mathrm{d}}{\mathrm{d}h}\frac{d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\mathrm{abs}}(\delta h)\big)\big)}{d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\mathrm{abs}}(h)\big)\big)} \\ &= \frac{\delta\,\frac{\mathrm{d}}{\mathrm{d}h'}\left[d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\mathrm{abs}}(h')\big)\big)\right]\big|_{h'=\delta h} - \varphi_i(h;\delta)\,\frac{\mathrm{d}}{\mathrm{d}h'}\left[d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\mathrm{abs}}(h')\big)\big)\right]\big|_{h'=h}}{d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\mathrm{abs}}(h)\big)\big)}.\end{aligned}$$

By using the expansion of $d_i\big(\mathtt{box}\big(\mathcal{R}^\infty_{\mathrm{abs}}(h)\big)\big)$ as in the proof of Theorem 2 together with the fact that, by definition of q_i, the coefficients

$$\sum_{p+q_i^{(p)}=j} \frac{\|a_i^{(p)} + b_i^{(p)}(h)\|_1}{(p+1)!}$$

are zero for all $j < q_i$, we conclude that

$$\frac{\mathrm{d}\varphi_i(h;\delta)}{\mathrm{d}h} = \frac{\sum_{j=q_i}^{\infty} h^j \sum_{p+q_i^{(p)}=j} \frac{1}{(p+1)!} \left[\delta^{j+1} - \varphi_i(h;\delta)\right] \left[(j+1)Q_i^{(p)}(\delta h) + h\dot{Q}_i^{(p)}(\delta h)\right]}{\sum_{j=q_i}^{\infty} h^{j+1} \sum_{p+q_i^{(p)}=j} \frac{1}{(p+1)!} Q_i^{(p)}(h)}. \tag{B.5}$$

We then expand the numerator for $j = q_i$, $j = q_i + 1$, and $j > q_i + 1$, and the denominator for $j = q_i$ and $j > q_i$:

$$\begin{aligned}\frac{\mathrm{d}\varphi_i(h;\delta)}{\mathrm{d}h} = &\frac{h^{q_i}\left(\delta^{q_i+1} - \varphi_i(h;\delta)\right)\sum_{p+q_i^{(p)}=q_i} \frac{(q_i+1)Q_i^{(p)}(\delta h)+h\dot{Q}_i^{(p)}(\delta h)}{(p+1)!}}{h^{q_i+1}\sum_{p+q_i^{(p)}=q_i}\frac{Q_i^{(p)}(h)}{(p+1)!} + \mathcal{O}(h^{q_i+2})} \\ &+ \frac{h^{q_i+1}\left(\delta^{q_i+2} - \varphi_i(h;\delta)\right)\sum_{p+q_i^{(p)}=q_i+1} \frac{(q_i+2)Q_i^{(p)}(\delta h)+h\dot{Q}_i^{(p)}(\delta h)}{(p+1)!}}{h^{q_i+1}\sum_{p+q_i^{(p)}=q_i}\frac{Q_i^{(p)}(h)}{(p+1)!} + \mathcal{O}(h^{q_i+2})} \\ &+ \frac{\mathcal{O}(h^{q_i+3})}{h^{q_i+1}\sum_{p+q_i^{(p)}=q_i}\frac{Q_i^{(p)}(h)}{(p+1)!} + \mathcal{O}(h^{q_i+2})}.\end{aligned}$$

Taking advantage of the fact that $\forall i, p : Q_i^{(p)}(0) > 0$, passing to the limit $h \to 0$ yields

$$\begin{aligned}\lim_{h\to 0}\frac{\mathrm{d}\varphi_i(h;\delta)}{\mathrm{d}h} = &(q_i+1)\lim_{h\to 0}\frac{\delta^{q_i+1} - \varphi_i(h;\delta)}{h} \\ &+ \delta^{q_i+1}(\delta-1)(q_i+2)\frac{\sum_{p+q_i^{(p)}=q_i+1}\frac{Q_i^{(p)}(0)}{(p+1)!}}{\sum_{p+q_i^{(p)}=q_i}\frac{Q_i^{(p)}(0)}{(p+1)!}},\end{aligned}$$

where we used the fact that $\varphi_i(h;\delta) \to \delta^{q_i+1}$ for $h \to 0$, as shown in Theorem 2. Using the same tools as in the proof of Theorem 2, one can easily show that

$$\lim_{h\to 0}\frac{\delta^{q_i+1} - \varphi_i(h;\delta)}{h} = \delta^{q_i+1}(1-\delta)\frac{\sum_{p+q_i^{(p)}=q_i+1}\frac{Q_i^{(p)}(0)}{(p+1)!}}{\sum_{p+q_i^{(p)}=q_i}\frac{Q_i^{(p)}(0)}{(p+1)!}},$$

which implies that

$$\lim_{h\to 0}\frac{\mathrm{d}\varphi_i(h;\delta)}{\mathrm{d}h} = -(1-\delta)\delta^{q_i+1}\frac{\sum_{p+q_i^{(p)}=q_i+1}\frac{Q_i^{(p)}(0)}{(p+1)!}}{\sum_{p+q_i^{(p)}=q_i}\frac{Q_i^{(p)}(0)}{(p+1)!}} \le 0.$$

This shows (42).

Now, we prove the second statement of Prop. 2, i.e., the inequality (43). This requires an intermediate step: if the $Q_i^{(p)}$ are constant (i.e., $\dot{Q}_i^{(p)} = 0$), the following implication holds

$$\varphi_i(h;\delta) \geq \delta^{q_i+1} \quad \Rightarrow \quad \frac{\mathrm{d}\varphi_i(h;\delta)}{\mathrm{d}h} \leq 0. \tag{B.6}$$

Indeed, if $\delta^{q_i+1} \leq \varphi_i(h;\delta)$, it follows that $\delta^{q_i+1+\ell} \leq \varphi_i(h;\delta)$ for any $\ell \geq 0$, since $\delta \leq 1$. Additionally, $Q_i^{(p)} \geq 0$ by definition. Using these facts together with (B.5) and $\dot{Q}_i^{(p)} = 0$ yields the implication (B.6). We can now show the inequality (43), by using a proof by contradiction:

Assume, for the sake of contradiction, that there exists a time $t \in [0,h]$ such that $\varphi(t;\delta) > \delta^{q+1}$, and let $\mathcal{T}$ be the set of all those elements. Let $t_- := \inf_{t\in\mathcal{T}} t$ be the highest lower bound of $\mathcal{T}$. Since φ is continuous in t, the set $\mathcal{T}$ is open, and we can find some $t_+ \in \mathcal{T}$ such that $(t_-, t_+) \subseteq \mathcal{T}$. Since φ is strictly decreasing for any element of $\mathcal{T}$, it is also strictly decreasing over the interval (t_-, t_+). If $\varphi(t_-,\delta) \leq \delta^{q+1}$, since φ is decreasing we conclude that $\delta^{q+1} \leq \varphi(t_+;\delta)$, which contradicts our assumption on $t_+ \in \mathcal{T}$. Therefore, there must hold $\varphi(t_-,\delta) > \delta^{q+1}$. We can thus find an intermediary value $\delta^{q+1} < \varphi_* < \varphi(t_-,\delta)$.

On the other hand, as we have seen in Theorem 2, $\varphi(t;\delta) \to \delta^{q+1}$ and $\frac{d}{dt}\varphi(t;\delta) \leq 0$ for $t \to 0$, so that there always exists a small enough $t' \geq 0$ such that $\varphi(t';\delta) \leq \delta^{q+1}$ and $t' \leq t_-$. By the intermediary value theorem, there exists t_* such that $t' \leq t_* \leq t_-$ and $\varphi_* = \varphi(t_*,\delta)$. By assumption, $\varphi_* < \varphi(t_-,\delta)$, hence $t_* \neq t_-$, and since $\varphi(t_*,\delta) > \delta^{q+1}$ we also have $t_* \in \mathcal{T}$. However, this contradicts the definition of t_-, as it should be a lower bound of $\mathcal{T}$. We thus get a contradiction, proving that $\varphi(t;\delta) \leq \delta^{q+1}$ must hold for all $t \in [0,h]$. $\square$

Acknowledgements

The authors gratefully acknowledge partial financial supports from the research training group ConVeY funded by the German Research Foundation under grant GRK 2428 and the project justITSELF funded by the European Research Council (ERC) under grant agreement No 817629.

References

[1] T. Gan, et al., Reachability analysis for solvable dynamical systems, IEEE Transactions on Automatic Control 63 (7) (2018) 2003–2018. doi:10.1109/TAC.2017.2763785.

[2] J. Liu, et al., Computing semi-algebraic invariants for polynomial dynamical systems, in: Proc. of the 9th ACM International Conference on Embedded Software, 2011, pp. 97–106. doi:10.1145/2038642.2038659.

[3] K. Ghorbal, A. Platzer, Characterizing algebraic invariants by differential radical invariants, in: Proc. of the 20th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2014, pp. 279–294. doi:10.1007/978-3-642-54862-8_19.

[4] M. Boreale, Complete algorithms for algebraic strongest postconditions and weakest preconditions in polynomial ODEs, Science of Computer Programming 193. doi:10.1016/j.scico.2020.102441.

[5] I. Mitchell, et al., A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games, IEEE Transactions on Automatic Control 50 (7) (2005) 947–957. doi:10.1109/TAC.2005.851439.

[6] S. Bansal, M. Chen, S. Herbert, C. Tomlin, Hamilton-Jacobi reachability: A brief overview and recent advances, in: Proc. of the 56th Annual Conference on Decision and Control, IEEE, 2017, pp. 2242–2253. doi:10.1109/CDC.2017.8263977.

[7] P. Duggirala, S. Mitra, M. Viswanathan, Verification of annotated models from executions, in: Proc. of the International Conference on Embedded Software, IEEE, 2013. doi:10.1109/EMSOFT.2013.6658604.

[8] J. Hoefkens, et al., Scientific computing, validated numerics, interval methods, Springer, 2001, Ch. Verified high-order integration of DAEs and higher-order ODEs, pp. 281–292. doi:10.1007/978-1-4757-6484-0_23.

[9] K. Makino, M. Berz, Rigorous integration of flows and ODEs using Taylor models, in: Proc. of Symbolic-Numeric Computation, ACM, 2009, pp. 79–84. doi:10.1145/1577190.1577206.

[10] X. Chen, et al., Taylor model flowpipe construction for non-linear hybrid systems, in: Proc. of the 33rd Real-Time Systems Symposium, IEEE, 2012, pp. 183–192. doi:10.1109/RTSS.2012.70.

[11] E. Asarin, T. Dang, A. Girard, Reachability analysis of nonlinear systems using conservative approximation, in: 6th International Workshop on Hybrid Systems: Computation and Control, Springer, 2003, pp. 20–35. doi:10.1007/3-540-36580-X_5.

[12] E. Asarin, et al., Hybridization methods for the analysis of nonlinear systems, Acta Informatica 43 (2007) 451–476. doi:10.1007/s00236-006-0035-7.

[13] Z. Han, B. Krogh, Reachability analysis of nonlinear systems using trajectory piecewise linearized models, in: Proc. of the American Control Conference, IEEE, 2006, pp. 1505–1510. doi:10.1109/ACC.2006.1656431.

[14] D. Li, S. Bak, S. Bogomolov, Reachability analysis of nonlinear systems using hybridization and dynamics scaling, in: International Conference on Formal Modeling and Analysis of Timed Systems, LNCS 12288, Springer, 2020, pp. 265–282. doi:10.1007/978-3-030-57628-8_16.

[15] T. Dang, C. Le Guernic, O. Maler, Computing reachable states for nonlinear biological models, in: International Conference on Computational Methods in Systems Biology, Springer, 2009, pp. 126–141. doi:10.1007/978-3-642-03845-7_9.

[16] T. Dang, et al., Accurate hybridization of nonlinear systems, in: Proc. of the 13th ACM International Conference on Hybrid Systems: Computation and Control, 2010, pp. 11–19. doi:10.1145/1755952.1755956.

[17] M. Althoff, O. Stursberg, M. Buss, Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization, in: Proc. of the 47th IEEE Conference on Decision and Control, 2008, pp. 4042–4048. doi:10.1109/CDC.2008.4738704.

[18] M. Althoff, Reachability analysis of nonlinear systems using conservative polynomialization and non-convex sets, in: Proc. of the 16th ACM International Conference on Hybrid Systems: Computation and Control, 2013, pp. 173–182. doi:10.1145/2461328.2461358.

[19] L. Benvenuti, et al., Assume-guarantee verification of nonlinear hybrid systems with ARIADNE, International Journal of Robust and Nonlinear Control 24 (4) (2014) 699–724. doi:10.1002/rnc.2914.

[20] P. Duggirala, et al., C2E2: A verification tool for stateflow models, in: Proc. of the 21st International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Springer, 2015, pp. 68–82. doi:10.1007/978-3-662-46681-0_5.

[21] M. Althoff, An introduction to CORA 2015, in: Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems, 2015, pp. 120–151. doi:10.29007/zbkv.

[22] J. Alexandre dit Sandretto, A. Chapoutot, DynIBEX: a differential constraint library for studying dynamical systems, 2016.

[23] X. Chen, et al., Flow*: An analyzer for non-linear hybrid systems, in: Proc. of the 25th International Conference Computer-Aided Verification, LNCS 8044, Springer, 2013, pp. 258–263. doi:10.1007/978-3-642-39799-8_18.

[24] F. Immler, Tool presentation: Isabelle/HOL for reachability analysis of continuous systems, in: Proc. of the 2nd Workshop on Applied Verification for Continuous and Hybrid Systems., 2015, pp. 180–187. doi:10.29007/b3wr.

[25] S. Bogomolov, et al., JuliaReach: a toolbox for set-based reachability, in: Proc. of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, 2019, pp. 39–44. doi:10.1145/3302504.3311804.

[26] A. Girard, Reachability of uncertain linear systems using zonotopes, in: 8th International Workshop on Hybrid Systems: Computation and Control, Springer, 2005, pp. 291–305. doi:10.1007/978-3-540-31954-2_19.

[27] C. Combastel, A state bounding observer based on zonotopes, in: European Control Conference (ECC), IEEE, 2003, pp. 2589–2594. doi:10.23919/ECC.2003.7085991.

[28] A.-K. Kopetzki, B. Schürmann, M. Althoff, Methods for order reduction of zonotopes, in: Proc. of the 56th IEEE Conference on Decision and Control, 2017, pp. 5626–5633. doi:10.1109/CDC.2017.8264508.

[29] X. Yang, J. K. Scott, A comparison of zonotope order reduction techniques, Automatica 95 (2016) 378–384. doi:10.1016/j.automatica.2018.06.006.

[30] M. Wetzlinger, A. Kulmburg, M. Althoff, Adaptive parameter tuning for reachability analysis of nonlinear systems, in: Proc. of the 24th ACM International Conference on Hybrid Systems: Computation and Control, 2021. doi:10.1145/3447928.3456643.

[31] D. Griffiths, D. Higham, Numerical methods for ordinary differential equations: initial value problems, Springer, 2010.

[32] E. Süli, D. F. Mayers, An introduction to numerical analysis, Cambridge University Press, 2003. doi:10.1017/CBO9780511801181.

[33] M. Rungger, M. Zamani, Accurate reachability analysis of uncertain nonlinear systems, in: Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control, 2018, p. 61–70. doi:10.1145/3178126.3178127.

[34] J. C. Butcher, Numerical methods for ordinary differential equations, John Wiley & Sons, 2016. doi:10.1002/9781119121534.

[35] L. Lapidus, J. H. Seinfeld, Numerical solution of ordinary differential equations, Academic press, 1971.

[36] U. M. Ascher, et al., Numerical solution of boundary value problems for ordinary differential equations, SIAM, 1994. doi:10.1137/1.9781611971231.

[37] M. Kerbl, Stepsize strategies for inclusion algorithms for ODE's, Computer Arithmetic, Scientific Computation, and Mathematical Modelling, IMACS Annals on Computing and Applied Mathematics 12 (1991) 437–452.

[38] W. Rufeger, E. Adams, A step size control for Lohner's enclosure algorithm for ordinary differential equations with initial conditions, in: Mathematics in Science and Engineering, Vol. 189, Elsevier, 1993, pp. 283–299. doi:10.1016/S0076-5392(08)62849-0.

[39] N. Nedialkov, Computing rigorous bounds on the solution of an initial value problem for an ordinary differential equation., Dissertation, University of Toronto (2000).

[40] P. Prabhakar, M. Viswanathan, A dynamic algorithm for approximate flow computations, in: Proc. of the 14th ACM International Conference on Hybrid Systems: Computation and Control, 2011, pp. 133–142. doi:10.1145/1967701.1967722.

[41] G. Frehse, et al., SpaceEx: Scalable verification of hybrid systems, in: Proc. of the 23rd International Conference on Computer Aided Verification, LNCS 6806, Springer, 2011, pp. 379–395. doi:10.1007/978-3-642-22110-1_30.

[42] G. Frehse, R. Kateja, C. Le Guernic, Flowpipe approximation and clustering in space-time, in: Proc. of the 16th ACM International Confer-

ence on Hybrid Systems: Computation and Control, 2013, pp. 203–212. doi:10.1145/2461328.2461361.

[43] M. Wetzlinger, N. Kochdumper, M. Althoff, Adaptive parameter tuning for reachability analysis of linear systems, in: Proc. of the 59th IEEE Conference on Decision and Control, 2020, pp. 5145–5152. doi:10.1109/CDC42340.2020.9304431.

[44] X. Chen, Reachability analysis of non-linear hybrid systems using Taylor models, Dissertation, RWTH Aachen University (2015).

[45] S. Bak, et al., High-level hybrid systems analysis with hypy, in: Proc. of the Workshop on Applied Verification of Continuous and Hybrid Systems, 2016, pp. 80–90. doi:10.29007/4f3d.

[46] G. Alefeld, G. Mayer, Interval analysis: Theory and applications, Computational and Applied Mathematics 121 (1-2) (2000) 421–464. doi:10.1016/S0377-0427(00)00342-3.

[47] N. Kochdumper, M. Althoff, Sparse polynomial zonotopes: A novel set representation for reachability analysis, IEEE Transactions on Automatic Control 66 (2) (2021) 4043–4058. doi:10.1109/TAC.2020.3024348.

[48] M. Althoff, B. H. Krogh, Reachability analysis of nonlinear differential-algebraic systems, IEEE Transactions on Automatic Control 59 (2) (2014) 371–383. doi:10.1109/TAC.2013.2285751.

[49] M. Berz, G. Hoffstätter, Computation and application of Taylor polynomials with interval remainder bounds, Reliable Computing 4 (1998) 83–97. doi:10.1023/A:1009958918582.

[50] M. Althoff, Reachability analysis and its application to the safety assessment of autonomous cars, Dissertation, Technische Universität München (2010).

[51] M. Althoff, C. Le Guernic, B. H. Krogh, Reachable set computation for uncertain time-varying linear systems, in: Proc. of the 14th ACM International Conference on Hybrid Systems: Computation and Control, 2011, pp. 93–102. `doi:10.1145/1967701.1967717`.

[52] V. V. Shenmaier, Complexity and approximation of finding the longest vector sum, Computational Mathematics and Mathematical Physics 58 (6) (2018) 850–857. `doi:10.1134/S0965542518060131`.

[53] A. Baburin, A. Pyatkin, Polynomial algorithms for solving the vector sum problem, Journal of Applied and Industrial Mathematics 1 (3) (2007) 268–272. `doi:10.1134/S1990478907030027`.

[54] T. Alamo, J. Bravo, E. Camacho, Guaranteed state estimation by zonotopes, Automatica 41 (2005) 1035–1043. `doi:10.1016/j.automatica.2004.12.008`.

[55] L. Geretti, et al., ARCH-COMP20 category report: Continuous and hybrid systems with nonlinear dynamics, in: ARCH20. 7th International Workshop on Applied Verification of Continuous and Hybrid Systems, EasyChair, 2020, pp. 49–75. `doi:10.29007/zkf6`.

[56] L. Geretti, J. Alexandre Dit Sandretto, M. Althoff, L. Benet, A. Chapoutot, P. Collins, P. Duggirala, M. Forets, E. Kim, U. Linares, D. Sanders, C. Schilling, M. Wetzlinger, ARCH-COMP21 category report: Continuous and hybrid systems with nonlinear dynamics, in: G. Frehse, M. Althoff (Eds.), Proc. of the 8th International Workshop on Applied Verification of Continuous and Hybrid Systems, 2021, pp. 32–54. `doi:10.29007/2jw8`.

[57] X. Chen, S. Sankaranarayanan, Decomposed reachability analysis for nonlinear systems, in: Proc. of the 37th Real-Time Systems Symposium, IEEE, 2016, pp. 13–24. `doi:10.1109/RTSS.2016.011`.

[58] A. El-Guindy, Control and stability of power systems using reachability analysis, Dissertation, Technische Universität München (2017).

[59] S. Kopecz, A. Meister, On order conditions for modified Patankar–Runge–Kutta schemes, Applied Numerical Mathematics 123 (2018) 159–179. doi:10.1016/j.apnum.2017.09.004.

[60] M. Laub, W. Loomis, A molecular network that produces spontaneous oscillations in excitable cells of dictyostelium, Molecular biology of the cell 9 (12) (1998) 3521–3532. doi:10.1091/mbc.9.12.3521.

[61] E. Aylward, P. Parrilo, J.-J. Slotine, Stability and robustness analysis of nonlinear systems via contraction metrics and SOS programming, Automatica 44 (8) (2008) 2163–2170. doi:10.1016/j.automatica.2007.12.012.

[62] O. Rössler, An equation for continuous chaos, Physics Letters A 57 (5) (1976) 397–398. doi:10.1016/0375-9601(76)90101-8.

[63] E. Lorenz, Deterministic nonperiodic flow, Journal of the atmospheric sciences 20 (2) (1963) 130–141. doi:10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

[64] J. Vano, et al., Chaos in low-dimensional Lotka–Volterra models of competition, Nonlinearity 19 (10) (2006) 2391. doi:10.1088/0951-7715/19/10/006.

[65] E. Klipp, et al., Systems biology in practice: concepts, implementation and application, John Wiley & Sons, 2005. doi:10.1002/3527603603.

[66] J. Vilar, et al., Mechanisms of noise-resistance in genetic oscillators, Proc. of the National Academy of Sciences 99 (9) (2002) 5988–5992. doi:10.1073/pnas.092133899.

[67] Y. Chen, A. Domínguez-García, Assessing the impact of wind variability on power system small-signal reachability, in: Proc. of the 44th Hawaii International Conference on System Sciences, IEEE, 2011, pp. 1–8. doi:10.1109/HICSS.2011.77.

[68] X. Chen, S. Sankaranarayanan, E. Ábrahám, Under-approximate flowpipes for non-linear continuous systems, in: Formal Methods in Computer-Aided Design (FMCAD), IEEE, 2014, pp. 59–66. doi:10.1109/FMCAD.2014.6987596.

[69] H. Bauer, Minimalstellen von funktionen und extremalpunkte, Archiv Der Mathematik - ARCH MATH 9 (1958) 389–393. doi:10.1007/BF01898615.

[70] G. Golub, M. Saunders, Linear least squares and quadratic programming, Integer and Nonlinear Programming (1969) 41.

[71] F. Barahona, et al., An application of combinatorial optimization to statistical physics and circuit layout design, Operations Research 36 (1988) 493–513. doi:10.1287/opre.36.3.493.