Forward inner-approximated reachability of non-linear continuous systems

ABSTRACT

We propose an approach for computing inner-approximations (also called under-approximations) of reachable sets of dynamical systems defined by non-linear, uncertain, ordinary differential equations. This is a notoriously difficult problem, much more intricate than outer-approximations for which there exist well known solutions, mostly based on Taylor models. The few methods developped recently for inner-approximation mostly rely on backward flowmaps, and extra ingredients, either coming from optimisation, or involving topological criteria, are required. Our solution, in comparison, builds on rather inexpensive set-based methods, namely a generalized mean-value theorem combined with Taylor models outer-approximations (also called over-approximations) of the flow and its jacobian with respect to the uncertain inputs and parameters. We are able to show with our prototype Matlab implementation that our method is both efficient and precise on classical examples. The combination of such forward inner and outer Taylor-model based approximations can be used as a basis for the verification and falsification of properties of cyber-physical systems.

Categories and Subject Descriptors

F.1.1 [**Theory of Computation**]: Computation by Abstract Devices; G.1.7 [**Mathematics of Computing**]: Numerical Analysis; G.1.0 [**Numerical Analysis**]: General—*Interval arithmetic,Numerical algorithms*
; G.4 [**Mathematical Software**]: Reliability and robustness

Keywords

Inner-approximation, Taylor models, affine arithmetic, modal intervals

1. INTRODUCTION

HSCC'17, April 17–20, 2017, Pittsburgh, Pennsylvania

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-2138-9.

DOI: http://dx.doi.org/XXXX.XXXX

— INTRO A REPRENDRE: D'ABORD IL MANQUE LES REF, MAIS SURTOUT ELLE ETAIT PAS MAL ORIENTEE SYSTEMES HYBRIDES ET PROP TEMPORELLES, ON VEUT SANS DOUTE AFFAIBLIR CA

The verification of software-enabled real-time control systems requires reasoning about non-linear hybrid systems, that exhibit both discrete and continuous behavior. Computing the reachable set of such systems is a central component of model-checking. While the exact reachability problem for hybrid systems is generally undecidable, in the recent years there has been much progress in the computation of outer-approximations of the reachable set, first for the verification of affine hybrid systems [], but also for the more general class of non-linear hybrid systems []. An outer-approximation makes possible the verification of safety properties of such systems. However, the verification of more general temporal properties, such as viability [] properties for instance, or the falsification of safety properties, also require inner-approximating the reachable set, that is computing set of states that are definitely reached, or put differently flowpipes that contain only solutions of the uncertain system.

Methods for inner-approximated reachability are far less developed, and especially in the non-linear case, since most methods in the non-linear case rely on conservative linearizations, which necessarily produce outer approximations.

In this work, we concentrate on the inner-approximation of the reachable sets of the continuous part of hybrid systems. We intend to handle guard conditions in subsequent work. We consider general systems of parametric ODEs, i.e. possibly non-linear, or even non-polynomial, of the form :

$$\dot{x}(t) = f(x, p, t) \tag{1}$$

where the continuous variables x belong to a state-space domain $\mathcal{D} \subseteq \mathbb{R}^n$, the (constant) parameters p belong to the uncertainty domain $\mathcal{P} \subseteq \mathbb{R}^p$, and $f : \mathcal{D} \times \mathcal{P} \times \mathbb{R}^+ \to \mathcal{D}$ is assumed sufficiently smooth on $\mathcal{D} \subseteq \mathbb{R}^n$ (at least $\mathcal{C}^1$, and sometimes more when we will use higher Taylor models, see Section 2.5).

Introducing the new state variable $z = (x, p, t)$ with $\dot{z} = (\dot{x}, 0, 1)$, and defining $\mathcal{Z} = \mathcal{D} \times \mathcal{P} \times \mathbb{R}^+$, the equation (1) can be rewritten with all uncertainties embedded in the initial state vector :

$$\dot{z}(t) = f(z) \tag{2}$$

In the sequel, we will write x_i and f_i for the ith component ($i = 1, \dots, n$) of the state vector x and of the function f.

Contributions:.

This paper extends the work presented in [7, 8], where their authors proposed an approach for direct forward inner-approximated reachability of discrete dynamical systems, and gave a few hints to handle continuous and hybrid systems.

Our method allows for computing inner-approximations of the flow of uncertain initial value problems, defined in Section 2.

There are two main ingredients for our method. The first one is that we only need in fact forward outer-approximations of some dynamics, for deriving inner-approximations, that we can treat using classical Taylor models (that we recap in Section 2.5). But we need to outer-approximate not only the set of reachable states of the dynamics but also of the "variational equations", including the dynamics of the Jacobian of the solutions with respect to the initial values.

This is made possible using the second main ingredient of our method, which is a generalized mean value theorem, that we introduce in Section 2.4. The generalized mean value theorem relies itself on modal intervals, a simple extension of classical interval arithmetics (see Section 2.2 and 2.3).

In many ways, all this is remarkably simple, with respect to other existing methods (using backward propagation of the flow of the dynamics), that we discuss in the related work Section 1. Our method is not much more complex than a classical Taylor model approach for outer-approximations. Still, we have to consider a bigger dynamical system, since we have to consider the Jacobian, with the order of n^2 equations (n being the number of equations in the original dynamical system) instead of n equations. But the Taylor models generated for the Jacobian can easily be derived from the Taylor models of the original equations, as we show in Section 3, greatly reducing the incurred costs.

Finally, we carry out some experiments with a matlab implementation and make tentative comparisons to existing work in Section 4.

Related work:.

Outer-approximations of non-linear systems are well studied, using in particular Taylor models, see [14] for instance. Inner-approximations have been far less studied, since this is a much more complicated problem, except in the case of linear systems, see e.g. [3, 11].

The main existing method for "under-approximating" (or inner-approximating as we put it here) flowpipes is a backward method, described in [1]. The method starts with a general compact and connected set of states X_0 described by a system of polynomial inequalities, and constructs a Taylor model for the backward flowmap Φ of the dynamics. Then, any *connected* set Ω which contains a point x which is mapped by Φ into X_0 is an inner-approximation of the reachable set of states X if Ω does not intersect the boundary of X. The method of [1] relies then on two computational ingredients. First, it builds a Taylor model for the backward flowmap (it is of the same order of complexity as for any forward outer-approximation, or for our inner-approximation method). Then, a candidate inner-approximation Ω that does not intersect the boundary of X is given by a set of polynomial constraints, derived from the Taylor model for the backward flowmap, and the constraints defining the initial set of states X_0. The method of [1] has then to test connectedness, which is intractable in general but can be semi-decided using clever interval methods.

A similar backward approach has been proposed in [16]. It is similar in that it also constructs an outer-approximation of the backward flowmap. But their authors construct an outer-approximation of the boundary of the reachable set to find inner-approximations. This is done using interval methods and a careful subdivision of the state-space, which might be very costly given that the boundary of the reachable set of highly non-linear ODEs might be extremely complicated to approximate.

The method we are presenting is much more straightforward, and relies only on classical Taylor models (although for a slightly augmented system of ODEs), and on simple methods for inner-approximating the image of a non-linear vector-valued scalar functions.

Finally, the authors have recently discovered the work [5], whose Section 4 contains ideas that look similar to ours. The main differences seem to be that we are considering more general parameterized dynamical systems, which will later allow us to handle guard conditions for hybrid systems, and that we have a different scheme for bounding the remainder in our inner-approximated Taylor models. But we could not assess the practical differences since the description in [5] is sketchy, and contains no real experiment.

2. PRELIMINARIES

Let us first introduce the ingredients that will be instrumental in the computation of inner-approximations of the range of a function over interval inputs, and in particular generalized intervals and mean-value theorem for inner-approximation . The results and notations quickly introduced in this section are mostly based on the work of Goldsztejn *et al.* on modal intervals [4].

2.1 Interval extensions, outer and inner approximations

Classical intervals [9, 12] are used in many situations to rigorously compute with interval domains instead of reals, usually leading to outer approximations of function ranges over boxes.

The set of classical intervals is denoted by

$$\mathbb{IR} = \{[a, b],\ a \in \mathbb{R}, b \in \mathbb{R}, a \leqslant b\}$$

In what follows, uncertain quantities defined in intervals are noted in bold, outer-approximating interval enclosures are noted in bold face and enclosed within brackets, and inner-approximating intervals are noted in bold face and enclosed within outward facing brackets.

An outer-approximating extension of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a function $[\boldsymbol{f}] : \mathbb{IR}^n \rightarrow \mathbb{IR}$ such that for all $\boldsymbol{x}$ in $\mathbb{IR}^n$, $\text{range}(f, \boldsymbol{x}) = \{f(x), x \in \boldsymbol{x}\} \subseteq [\boldsymbol{f}](\boldsymbol{x})$. The natural interval extension consists in replacing real operations by their interval counterparts in the expression of the function. A generally more accurate extension relies on the mean-value theorem, linearizing the function to compute. Suppose the function f is differentiable over the interval $\boldsymbol{x} = [a, b]$. Then,the mean-value theorem implies that for any choice of $x_0 \in \boldsymbol{x}$, then we have

$$\forall x \in \boldsymbol{x},\ \exists c \in \boldsymbol{x},\ f(x) = f(x_0) + f'(c)(x - x_0).$$

If we can bound the range of the gradient of f over $\boldsymbol{x}$, by $\text{range}(f', \boldsymbol{x}) =\subseteq [\boldsymbol{f'}](\boldsymbol{x})$, we can derive the following interval enclosure, usually called the mean-value extension: for any

$x_0 \in \boldsymbol{x}$

$$\text{range}(f, \boldsymbol{x}) \subseteq f(x_0) + [\boldsymbol{f}'](\boldsymbol{x})(\boldsymbol{x} - x_0)$$

Classical interval computations can be interpreted as quantified propositions. Consider for example $f(x) = x^2 - x$. Its natural interval extension, evaluated on $[2, 3]$, is $[\boldsymbol{f}]([2, 3]) = [2, 3]^2 - [2, 3] = [1, 7]$, which can be interpreted as the proposition

$$(\forall x \in [2, 3])\, (\exists z \in [1, 7])\, (f(x) = z).$$

The mean-value extension gives $f(2.5) + [\boldsymbol{f}']([2, 3]) \times ([2, 3] - 2.5) = [1.25, 6.25]$, and can be interpreted similarly.

Inner-approximations determine a set of values proved to belong to the range of the function over some input box. The fact that some $]\boldsymbol{z}[\in \mathbb{IR}$ satisfies $]\boldsymbol{z}[\subseteq \text{range}(f, \boldsymbol{x})$, i.e., is an inner-approximation of the range of f over $\boldsymbol{x}$, can again be written using quantifiers :

$$(\forall z \in]\boldsymbol{z}[)\, (\exists x \in \boldsymbol{x})\, (f(x) = z).$$

2.2 Generalized intervals

Let us first introduce generalized intervals, i.e., intervals whose bounds are not ordered, and Kaucher arithmetic [10] on these intervals.

The set of generalized intervals is denoted by $\mathbb{IK} = \{[a, b],\ a \in \mathbb{R}, b \in \mathbb{R}\}$. Related to a set of real numbers $\{x \in \mathbb{R},\ a \leqslant x \leqslant b\}$, one can consider two generalized intervals, $[a, b]$, which is called *proper*, and $[b, a]$, which is called *improper*. We define the operations dual $[a, b] = [b, a]$ and pro $[a, b] = [\min(a, b), \max(a, b)]$.

Definition 1 ([4]). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function and $\boldsymbol{x} \in \mathbb{IK}^n$, which we can decompose in $\boldsymbol{x}_{\mathcal{A}} \in \mathbb{IR}^p$ and $\boldsymbol{x}_{\mathcal{E}} \in (\text{dual}\, \mathbb{IR})^q$ with $p + q = n$. A generalized interval $\boldsymbol{z} \in \mathbb{IK}$ is $(f, \boldsymbol{x})$-interpretable if*

$$(\forall x_{\mathcal{A}} \in \boldsymbol{x}_{\mathcal{A}})\, (Q_z z \in \textit{pro}\ \boldsymbol{z})\, (\exists x_{\mathcal{E}} \in \textit{pro}\ \boldsymbol{x}_{\mathcal{E}}), (f(x) = z) \quad (3)$$

where $Q_z = \exists$ if $(\boldsymbol{z})$ is proper, and $Q_z = \forall$ otherwise.

When all intervals in (3) are proper, we retrieve the interpretation of classical interval computation, which gives an outer approximation of $\text{range}(f, \boldsymbol{x})$

$$(\forall x \in \boldsymbol{x})\, (\exists z \in [\boldsymbol{z}])\, (f(x) = z).$$

When all intervals are improper, (3) becomes an inner-approximation of $\text{range}(f, \boldsymbol{x})$

$$(\forall z \in]\text{pro}\ \boldsymbol{z}[)\, (\exists x \in \text{pro}\ \boldsymbol{x})\, (f(x) = z).$$

2.3 Kaucher arithmetic and the generalized interval natural extension

Kaucher arithmetic [10] returns intervals that are interpretable as inner-approximations in some simple cases.

Kaucher addition extends addition on classical intervals by $\boldsymbol{x} + \boldsymbol{y} = [\underline{x} + \underline{y}, \overline{x} + \overline{y}]$ and $\boldsymbol{x} - \boldsymbol{y} = [\underline{x} - \overline{y}, \overline{x} - \underline{y}]$.

For multiplication, things are a little more complex. Let us decompose $\mathbb{IK}$ in $\mathcal{P} = \{\boldsymbol{x} = [\underline{x}, \overline{x}],\ \underline{x} \geqslant 0 \wedge \overline{x} \geqslant 0\}$, $-\mathcal{P} = \{\boldsymbol{x} = [\underline{x}, \overline{x}],\ \underline{x} \leqslant 0 \wedge \overline{x} \leqslant 0\}$, $\mathcal{Z} = \{\boldsymbol{x} = [\underline{x}, \overline{x}],\ \underline{x} \leqslant 0 \leqslant \overline{x}\}$, and dual $\mathcal{Z} = \{\boldsymbol{x} = [\underline{x}, \overline{x}],\ \underline{x} \geqslant 0 \geqslant \overline{x}\}$. Kaucher multiplication $\boldsymbol{x} \times \boldsymbol{y}$ is described in Table 1, we only give an intuitive explanation of one of its entries below.

Let us interpret the result of the multiplication $\boldsymbol{z} = \boldsymbol{x} \times \boldsymbol{y}$ in one of the cases encountered when $\boldsymbol{y} \in$ dual $\mathcal{Z}$, for instance for $\boldsymbol{x} \in \mathcal{Z}$. Proposition 1 will express the fact that

$\boldsymbol{x} \times \boldsymbol{y}$	$\boldsymbol{y} \in \mathcal{P}$	$\mathcal{Z}$	$-\mathcal{P}$	dual$\mathcal{Z}$
$\boldsymbol{x} \in \mathcal{P}$	$[\underline{x}\underline{y}, \overline{x}\overline{y}]$	$[\overline{x}\underline{y}, \overline{x}\overline{y}]$	$[\overline{x}\underline{y}, \underline{x}\overline{y}]$	$[\underline{x}\underline{y}, \underline{x}\overline{y}]$
$\mathcal{Z}$	$[\underline{x}\overline{y}, \overline{x}\overline{y}]$	$[\min(\underline{x}\overline{y}, \overline{x}\underline{y}),$ $\max(\underline{x}\underline{y}, \overline{x}\overline{y})]$	$[\overline{x}\underline{y}, \underline{x}\underline{y}]$	0
$-\mathcal{P}$	$[\underline{x}\overline{y}, \overline{x}\underline{y}]$	$[\underline{x}\overline{y}, \underline{x}\underline{y}]$	$[\overline{x}\overline{y}, \underline{x}\underline{y}]$	$[\overline{x}\overline{y}, \overline{x}\underline{y}]$
dual$\mathcal{Z}$	$[\underline{x}\underline{y}, \overline{x}\underline{y}]$	0	$[\overline{x}\overline{y}, \underline{x}\overline{y}]$	$[\max(\underline{x}\underline{y}, \overline{x}\overline{y}),$ $\min(\underline{x}\overline{y}, \overline{x}\underline{y})]$

Table 1: Kaucher multiplication

the result can be interpreted as in Definition 1. Interval $\boldsymbol{z}$ can a priori either be proper or improper, let us consider the improper case. We obtain an inner-approximation of the range of the multiplication: according to the quantifiers in Definition 1, computing $\boldsymbol{z} = \boldsymbol{x} \times \boldsymbol{y}$ consists in finding $\boldsymbol{z}$ such that for all $x \in \boldsymbol{x}$, for all $z \in$ pro $\boldsymbol{z}$, there exists $y \in$ pro $\boldsymbol{y}$ such that $z = x \times y$. If $\boldsymbol{x}$ contains zero, which is the case when $\boldsymbol{x} \in \mathcal{Z}$, then $\boldsymbol{z}$ is necessarily 0, the result given in Table 1. Indeed, a property that holds for all $x \in \boldsymbol{x}$, holds in particular for $x = 0$, from which we deduce that for all $z \in$ pro $\boldsymbol{z}$, (there exists $y \in$ pro $\boldsymbol{y}$) $z = 0$.

When restricted to proper intervals, these operations coincide with the classical interval operations.

The important feature of Kaucher arithmetic is that it defines a generalized interval natural extension (see [4]) :

Proposition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function, given by an arithmetic expression where each variable appears syntactically only once. Then for $\boldsymbol{x} \in \mathbb{IK}^n$, $f(\boldsymbol{x})$, computed using Kaucher arithmetic, is $(f, \boldsymbol{x})$-interpretable.*

Kaucher arithmetic can thus be used in some cases to compute an inner-approximation of $\text{range}(f, \boldsymbol{x})$. But the restriction to functions f with single occurrences of variables, that is with no dependency, prevents its direct use. A mean-value extension allows us to by-pass this limitation.

2.4 Generalized interval mean value extension

In the general case of a differentiable function f, the mean-value theorem can be extended to define a generalized interval mean value extension (see [4]) :

Theorem 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable, $\boldsymbol{x} \in \mathbb{IK}^n$ and suppose that for each $i \in \{1, \ldots, n\}$, we can compute $[\boldsymbol{\Delta}_i] \in \mathbb{IR}$ such that*

$$\left\{ \frac{\partial f}{\partial x_i}(x),\ x \in \textit{pro}\ \boldsymbol{x} \right\} \sqsubseteq [\boldsymbol{\Delta}_i]. \quad (4)$$

Then, for any $\tilde{x} \in$ pro $\boldsymbol{x}$, the following interval is $(f, \boldsymbol{x})$-interpretable :

$$\tilde{f}(\boldsymbol{x}) = f(\tilde{x}) + \sum_{i=1}^{n} [\boldsymbol{\Delta}_i](\boldsymbol{x}_i - \tilde{x}_i). \quad (5)$$

Example 1. *Let f be defined by $f(x) = x^2 - x$, for which we want to compute an inner-approximation of the range over $\boldsymbol{x} = [2, 3]$. Due to the two occurrences of x, $f(\boldsymbol{x})$, computed with Kaucher arithmetic, is not $(f, \boldsymbol{x})$-interpretable. The interval $\tilde{f}(\boldsymbol{x}) = f(2.5) + \boldsymbol{f}'([2, 3])(\boldsymbol{x} - 2.5) = 3.75 + [3, 5](\boldsymbol{x} - 2.5)$ given by its mean-value extension, computed with Kaucher arithmetic, is $(f, \boldsymbol{x})$-interpretable. For $\boldsymbol{x} = [3, 2]$, using the multiplication rule for $\mathcal{P} \times$ dual $\mathcal{Z}$, we get $\tilde{f}(\boldsymbol{x}) = 3.75 + [3, 5]([3, 2] - 2.5) = 3.75 + [3, 5][0.5, -0.5] =$*

$3.75 + [1.5, -1.5] = [5.25, 2.25]$*, that can be interpreted as:* $\forall z \in [2.25, 5.25]$*,* $\exists x \in [2, 3]$*,* $z = f(x)$*. Thus,* $[2.25, 5.25]$ *is an inner-approximation of* $\text{range}(f, [2, 3])$*.*

In Section 3, we will be using Theorem 1 with f being the solution of the uncertain dynamical system (2): for this, we need to be able to outer-approximate, at any time t, $f(\tilde{x}), \tilde{x} \in \text{pro } \boldsymbol{x}$, and its jacobian with respect to the (uncertain) initial value of the system, $\left\{ \frac{\partial f}{\partial x_i}(x),\ x \in \text{pro } \boldsymbol{x} \right\}$. Computing an enclosure of the solution of an initial value problem is the objective of Section 2.5.

2.5 Enclosing the flow of an uncertain ODE with interval Taylor methods

Consider the uncertain dynamical system (2), where $z = (x, p, t)$ and with initial condition $z(t_0) \in \mathcal{Z}_{\prime}$ at time $t_0 \geq 0$. Let us denote $\mathcal{Z}(t; t_0, \mathcal{Z}_{\prime})$ the set of solutions of (2) at time t for initial conditions in $\mathcal{Z}_{\prime}$ at t_0. We define a time grid $t_0 < t_1 < \ldots < t_N$, and assume $\mathcal{Z}_{\prime} = \boldsymbol{z}_0 = [\underline{z}_0, \overline{z}_0]$ at time $t_0 \geq 0$.

Interval Taylor methods for guaranteed set integration, see [13] for a review, compute flowpipes that are guaranteed to contain the reachable set of solutions $\mathcal{Z}(t; t_0, \mathcal{Z}_{\prime})$ of (2) for all time t in $[t_j, t_{j+1}]$. They first verify the existence and uniqueness of the solution using the Banach fixed point theorem and the Picard-Lindelöf operator, and compute an a priori rough enclosure $[\boldsymbol{r}_{j+1}]$ of $\mathcal{Z}(t)$ for all t in $[t_j, t_{j+1}]$. A tighter enclosure for the set of reachable values for t in $[t_j, t_{j+1}]$ is then computed using a Taylor series expansion of order k of the solution at t_j, where $[\boldsymbol{r}_{j+1}]$ is used to enclose the remaining term :

$$[\boldsymbol{z}](t, t_j, [\boldsymbol{z}_j]) = [\boldsymbol{z}_j] + \sum_{i=1}^{k-1} \frac{(t - t_j)^i}{i!} f^{[i]}([\boldsymbol{z}_j]) + \frac{(t - t_j)^k}{k!} f^{[k]}([\boldsymbol{r}_{j+1}]), \quad (6)$$

where the Taylor coefficients $f^{[i]}$, which are the $i-1$th Lie derivative of f along vector field f, are defined inductively, and can be computed by automatic differentiation as follows (for all $k = 1, \ldots, n$) :

$$f_k^{[1]} = f_k \quad (7)$$

$$f_k^{[i+1]} = \sum_{j=1}^{n} \frac{\partial f_k^{[i]}}{\partial z_j} f_j \quad (8)$$

PROOF. Let $z(t)$ be a solution to Equation (1) starting at time 0 at point z_0. By definition :

$$\begin{aligned} \tfrac{dz}{dt}(t) &= f(z(t)) \\ &= f^{[1]}(z(t)) \end{aligned}$$

and more generally, we can prove by induction on l that $\frac{d^{(l+1)}z}{dt^{(l+1)}}(t) = f^{[l+1]}(z(t))$, since by induction hypothesis :

$$\begin{aligned} \tfrac{d^{(l+1)}z}{dt^{(l+1)}}(t) &= \tfrac{d}{dt}\left(t \to f^{[l]}(z(t))\right) \\ &= \sum_{j=1}^{n} \dot{z}_j(t) \tfrac{\partial f^{[l]}}{\partial z_j}(z(t)) \\ &= \sum_{j=1}^{n} f_j(z(t)) \tfrac{\partial f^{[l]}}{\partial z_j}(z(t)) \\ &= f^{[l+1]}(z(t)) \end{aligned}$$

Equation (6) is then a direct consequence from Taylor-Lagrange expansion formula, for sufficiently smooth functions f. $\square$

Finally, we use enclosure $[\boldsymbol{z}_{j+1}] = [\boldsymbol{z}](t_{j+1}, t_j, [\boldsymbol{z}_j])$ as initial solution set at time t_{j+1} to derive the interval Taylor model on the next time step.

If evaluated plainly in interval arithmetic, scheme (6) yields enclosures of increasing width. A classical way to improve this evaluation is the method introduced by Lohner, that uses QR-factorization. In the experiments presented in Section 4, we chose to control wrapping using affine arithmetic [2] instead of interval arithmetic to evaluate the solution enclosures given by (6).

3. FORWARD INNER REACHABILITY OF CONTINOUS SYSTEMS

As in Section 2.5, we consider the uncertain dynamical system (2), where $z = (x, p, t)$ and with initial condition $z(t_0) \in \mathcal{Z}_{\prime} = \boldsymbol{z}_0 = [\underline{z}_0, \overline{z}_0]$ at time $t_0 \geq 0$, and we denote $\mathcal{Z}(t; t_0, \boldsymbol{z}_0)$ the set of solutions $\{z(t, z_0),\ z_0(t_0) \in \boldsymbol{z}_0\}$ of (2) at time t.

We have seen in Section 2.5, that for a time grid $t_0 < t_1 < \ldots < t_N$, we can compute on each time interval $[t_j, t_{j+1}]$, a flowpipe (6) that is guaranteed to contain the reachable set of solutions of (2) for all time t in $[t_j, t_{j+1}]$.

We now want to compute also an inner-approximating flowpipe of this reachable set, that is for all t in $[t_j, t_{j+1}]$, a range $]\boldsymbol{z}[(t, t_j, [\boldsymbol{z}_j])$ such that all values inside that range are sure to be reached at time t by an execution of system (2). For that, we will apply Theorem 1, at all time t, to the function z from $\mathbb{R}^n$ to R^n, defined by $z_0 \mapsto z(t, z_0)$, solution to the IVP (2).

In Section 3.1, we give the main lines of the computation of inner-approximated flowpipes, and state the algorithm. We then detail and comment each of its steps. In Section 3.2, we show how we use the classical interval Picard-Lindelöf iteration method to get rough enclosures of the solution and its Jacobian on each time step, that we use for computing the remainders of the Taylor models. In Section 3.3, we build the Taylor models, and show that we can compute the Taylor model of the Jacobian as if we were simply derivating the Taylor model for the solution of the initial ODE, which makes its construction very simple and efficient. Finally, in Section 3.4, we comment the actual computation of the inner-approximation flow-pipe, and show how a loss of accuracy in the outer-approximation results in a loss of accuracy in the inner-approximation, and even possibly to an empty inner-approximation.

We will illustrate the computations on the following example:

EXAMPLE 2. *We consider the Brusselator equation:*

$$f(x) = \begin{pmatrix} 1 - 2x_1 + \frac{3}{2}x_1^2 x_2 \\ x_1 - \frac{3}{2}x_1^2 x_2 \end{pmatrix}$$

with $x = (x_1, x_2)$*, over the time interval* $[0, h]$ $\left(h = \frac{1}{20}\right)$*, and with initial conditions* $x_0 = ([2, 2.15], [0.1, 0.15])$*.*

3.1 Principle of the algorithm

We thus need an (outer) enclosure of $z(t, \tilde{z}_0)$ for some $\tilde{z}_0 \in \boldsymbol{z}_0$, and of its Jacobian with respect to z_0, evaluated over range $\boldsymbol{z}_0$, defined by the coefficients $J_{ij}(t,\boldsymbol{z}_0) = \frac{\partial z_i}{\partial z_{0,j}}(t,\boldsymbol{z}_0)$, for i and j between 1 and n, and where z_i is the i-th component of the vector flow function z, and $z_{0,j}$ the j-th component of the vector of initial conditions z_0.

We compute these outer-approximations by applying the Taylor method of Section 2.5 to $z(t,\tilde{z_0})$ and $J(t,z_0)$ where $z_0 \in \boldsymbol{z}_0$ and with initial condition $J(t_0) = Id$ the identity matrix : $z(t,\tilde{z_0})$ satisfies system (2) with $z(t_0) = \tilde{z_0} \in \boldsymbol{z}_0$, so that we can directly use the Taylor expansion (6) on each time interval $[t_j,t_{j+1}]$ to compute $\mathcal{Z}(t;t_0,\tilde{z_0})$. The coefficients of the Jacobian matrix of the flow satisfy the following differential equations :

$$\dot{J}_{ij}(t,z_0) = \sum_{k=1}^{n} \frac{\partial f_i}{\partial z_k}(z).J_{kj}(t,z_0) \tag{9}$$

that can be rewritten

$$\dot{J}(t,z_0) = \mathrm{Jac}_z f(z(t,z_0)).J(t,z_0). \tag{10}$$

with $J(t_0) = Id$ (these are the "variational equations" used in particular in [17] for improving outer-approximations of continuous dynamical systems). We will write equivalently, by a slight abuse of notation, $\mathrm{Jac}_z f$ as the function of the z_i and J_{ij} (linear in J_{ij}) which is the right hand side of Equation 10. Hence, its pq entry is :

$$(\mathrm{Jac}_z f)_{pq} = \sum_{k=1}^{n} \frac{\partial f_p}{\partial z_k} J_{kq} \tag{11}$$

A Taylor expansion can thus be used to outer-approximate the solution of (10) noted $\mathcal{J}(t;t_0,\boldsymbol{z}_0)$ on each time interval $[t_j,t_{j+1}]$, using the outer-approximation for $z(t,z_0)$ given by Taylor expansion (6). We just have to note that Equations (9) together with Equations (1) define a system of $n(n+1)$ ordinary differential equations in $n(n+1)$ variable (the z_i and J_{ij}). We call the corresponding vector field F and write similarly, by an abuse of notation, for H a function in the z_i and J_{ij}, $H^{[i]}$ the $i-1$th Lie derivative of H along the (augmented) vector field H. More explicitly, it is defined inductively as follows :

$$\begin{aligned} H^{[1]} &= H && (12)\\ H^{[i+1]} &= (H^{[i]})^{[2]} && (13) \end{aligned}$$

where the first Lie derivative is :

$$H^{[2]} = \sum_{i=1}^{n} \frac{\partial H}{\partial z_i} f_i + \sum_{k,l=1}^{n} \frac{\partial H}{\partial J_{kl}} \left(\sum_{s=1}^{n} \frac{\partial f_k}{\partial z_s} J_{sl} \right) \tag{14}$$

Proof. Consider a solution $t \to z_i(t)$ and $t \to J_{ij}(t)$ of Equations (9) and (1). The Lie derivative of H is the time derivative of $\tilde{H}(t) = H(z_1(t),\ldots,z_n(t),J_{11}(t),\ldots,J_{nn}(t))$:

$$\dot{\tilde{H}}(t) = \sum_{i=1}^{n} \frac{\partial H}{\partial z_i}(t)\dot{z}_i(t) + \sum_{k,l=1}^{n} \frac{\partial H}{\partial J_{kl}}(t)\dot{J}_{kl}(t)$$

We get the formula (14) by replacing $\dot{z}_i$ by its expression in Equation (1) and $\dot{J}_{kl}$ by its expression in Equation (9). □

Example 3. *We consider the continous system of Example 2 :*

$$f(x) = \begin{pmatrix} 1-2x_1+\frac{3}{2}x_1^2x_2 \\ x_1 - \frac{3}{2}x_1^2x_2 \end{pmatrix}$$

The Jacobian that appears in Equation (10) is :

$$Jac_z f(z(t,z_0)) = \begin{pmatrix} -2+3x_1x_2 & \frac{3}{2}x_1^2 \\ 1-3x_1x_2 & -\frac{3}{2}x_1^2 \end{pmatrix}$$

Altogether, the algorithm to compute an inner-approximated flow-pipe of the uncertain initial-value problem given a time grid $t_0 < t_1 < \ldots < t_N$, an initial range $\boldsymbol{z}_0$, and some $\tilde{z_0} \in \boldsymbol{z}_0$, is as follows :

Initialize: $j=0$, $t_j = t_0$, $[\boldsymbol{z}_j] = \boldsymbol{z}_0$, $[\tilde{z}_j] = \tilde{z}_0$, $[\boldsymbol{J}_j] = Id$
Then on each time interval $[t_j,t_{j+1}]$ do:
Step 1. compute a priori enclosures $[\boldsymbol{r}_{j+1}]$ of $\mathcal{Z}(t;t_j,\boldsymbol{z}_j)$ for all t in $[t_j,t_{j+1}]$, $[\tilde{\boldsymbol{r}}_{j+1}]$ of $\mathcal{Z}(t;t_j,\tilde{z}_j)$ for all t in $[t_j,t_{j+1}]$, and $[\boldsymbol{R}_{j+1}]$ of $\mathcal{J}(t;t_j,\boldsymbol{z}_j)$
Step 2. build the Taylor Models valid on $[t_j,t_{j+1}]$:

$$[\boldsymbol{z}](t,t_j,[\boldsymbol{z}_j]) = [\boldsymbol{z}_j] + \sum_{i=1}^{k-1} \frac{(t-t_j)^i}{i!} f^{[i]}([\boldsymbol{z}_j]) + \frac{(t-t_j)^k}{k!} f^{[k]}([\boldsymbol{r}_{j+1}]). \tag{15}$$

$$[\tilde{\boldsymbol{z}}](t,t_j,[\tilde{\boldsymbol{z}}_j]) = [\tilde{\boldsymbol{z}}_j] + \sum_{i=1}^{k-1} \frac{(t-t_j)^i}{i!} f^{[i]}([\tilde{\boldsymbol{z}}_j]) + \frac{(t-t_j)^k}{k!} f^{[k]}([\tilde{\boldsymbol{r}}_{j+1}]). \tag{16}$$

$$[\boldsymbol{J}](t,t_j,[\boldsymbol{z}_j]) = [\boldsymbol{J}_j] + \sum_{i=1}^{k-1} \frac{(t-t_j)^i}{i!} \mathrm{Jac}_x(f^{[i]})([\boldsymbol{r}_{j+1}])[\boldsymbol{J}_j] + \frac{(t-t_j)^k}{k!} \mathrm{Jac}_x(f^{[k]})([\boldsymbol{r}_{j+1}])[\boldsymbol{R}_{j+1}] \tag{17}$$

Step 3. deduce an inner-approximation valid for t in $[t_j,t_{j+1}]$: if $]\boldsymbol{z}[(t,t_j)$ defined by Equation (18) is an improper interval

$$]\boldsymbol{z}[(t,t_j) = [\tilde{\boldsymbol{z}}](t,t_j,[\tilde{\boldsymbol{z}}_j]) + [\boldsymbol{J}](t,t_j,[\boldsymbol{z}_j]) * ([\overline{z_0},\underline{z_0}] - \tilde{z_0}) \tag{18}$$

then interval pro $]\boldsymbol{z}[(t,t_j)$ is an inner-approximation of the set of solutions $\{z(t,z_0),\ z_0(t_0) \in \boldsymbol{z}_0\}$ of (2) at time t, otherwise the inner-approximation is empty.

Step 4. initialize the next iteration by computing $[\boldsymbol{z}_{j+1}] = [\boldsymbol{z}](t_{j+1},t_j,[\boldsymbol{z}_j])$, $[\tilde{\boldsymbol{z}}_{j+1}] = [\tilde{\boldsymbol{z}}](t_{j+1},t_j,\ [\tilde{\boldsymbol{z}}_j])$, $[\boldsymbol{J}_{j+1}] = [\boldsymbol{J}](t,t_j,[\boldsymbol{z}_j])$

3.2 Step 1: computing the rough enclosures

In order the kth term in Equations (15) and (16) we need to compute $[\boldsymbol{r}_j]$ (respectively $[\boldsymbol{R}_j]$), i.e. intervals which over-approximate the components of the solutions z and J to the variational equations over the time interval $[t_j,t_{j+1}]$. This is done by using the classical interval Picard-Lindelöf method. This goes as follows : note first that Equation (1) can be rewritten in the form of the integral equation

$$z(t) = z_0 + \int_{t_j}^{t_{j+1}} f(z(s))\mathrm{d}s \tag{19}$$

Hence calling F the functional which to function z associates the right-hand side of Equation (19), under the condition that f is Lipschitz, it possesses a unique fixpoint, that is the solution to Equations (1) and (19). The interval version $F^\sharp$ of the Picard-Lindelöf operator F enjoys the same property and is derived using the obvious rough interval approximation of the integral : $F^\sharp([z]) = z_0 + [t_j, t_{j+1}]f([z])$ (where $[z]$ will denote ultimately the "rough" enclosure of the solutions to Equation (1) and f denotes also the interval extension of function f). Simple Jacobi like iteration suffices to reach the fixpoint of $F^\sharp$: $[z]_0 = z_0$, $[z]_{i+1} = F^\sharp([z]_i)$ for all $i \in \boldsymbol{N}$. Finite time convergence can be ensured using outwards rounding in finite precision, numerical acceleration techniques, widening techniques etc.

EXAMPLE 4. *We carry on with the computation of outer-approximations for solutions and Jacobians for the Brusselator on the first time step.*

We will write $[x_i](t)$ instead of $[x_i](t, 0, [x_0])$ as we are only considering the first time step. We first need to determine the rough enclosures, the remainders for $k = 2$ (first order Taylor model for Equations (15) and (17)) will be determined in Example 6.

Then we compute "rough enclosures" $[\boldsymbol{r}_1]_i$ and $[\boldsymbol{R}_1]_{i,j}$ of the $x_i(t)$ and $J_{ij}(t)$ over $t \in [0, h]$, $x_1 \in [(x_1)_0]$, $x_2 \in [(x_2)_0]$. We use the interval Picard-Lindelöf method of Section 3.2 and find : $[\boldsymbol{r}_1] = \begin{pmatrix} [1.8609, 2.1500] \\ [0.1000, 0.2316] \end{pmatrix}$, $[\boldsymbol{R}_1] = \begin{pmatrix} [0.9279, 1.0000] & [0.0000, 0.3467] \\ [-0.0247, 0.0221] & [0.6533, 1.0000] \end{pmatrix}$

3.3 Step 2: building the Taylor models

Efficient computations of the Lie derivatives of the Jacobian.

The formulation of Equation (17), relies on the possibility to commute the ith Lie derivative with the calculation of the Jacobian. Without this, we would have written:

$$[\boldsymbol{J}](t, t_j, [\boldsymbol{z}_j]) = [\boldsymbol{J}_j] + \sum_{i=1}^{k-1} \frac{(t-t_j)^i}{i!} (\mathrm{Jac}_x(f))^{[i]}([\boldsymbol{r}_{j+1}], [\boldsymbol{J}_j]) + \frac{(t-t_j)^k}{k!} (\mathrm{Jac}_x(f))^{[k]}([\boldsymbol{r}_{j+1}], [\boldsymbol{R}_{j+1}]) \quad (20)$$

($\mathrm{Jac}_x(f)$ being seen as a function of the z_i and J_{ij}, which is linear in the J_{ij} as in Equation (11)) Equations (17) and (20) are equivalent since the two derivatives (the Jacobian calculation and the Lie derivative) commute.

PROOF. We prove this equivalence by induction on the number of Lie derivations. For $i = 1$, we have, by definition $\mathrm{Jac}_x(f^{[1]}) = \mathrm{Jac}_x(f) = (\mathrm{Jac}_x(f))^{[1]}$. Suppose now we have, as an induction step $\mathrm{Jac}_x(f^{[i]}) = (\mathrm{Jac}_x(f))^{[i]}$. We now write :

$$\begin{aligned} \mathrm{Jac}_x(f^{[i+1]})_{pq} &= \sum_{k=1}^{n} \frac{\partial f_p^{[i+1]}}{\partial z_k} J_{kq} \\ &= \sum_{k=1}^{n} \frac{\partial}{\partial z_k} \left(\sum_{l=1}^{n} \frac{\partial f_p^{[i]}}{\partial z_l} f_l \right) J_{kq} \end{aligned}$$

hence,

$$\mathrm{Jac}_x(f^{[i+1]})_{pq} = \sum_{k,l=1}^{n} \left(\frac{\partial^2 f_p^{[i]}}{\partial z_k \partial z_l} f_l + \frac{\partial f_p^{[i]}}{\partial z_l} \frac{\partial f_l}{\partial z_k} \right) J_{kq} \quad (21)$$

On the other hand we have :

$$(\mathrm{Jac}_x(f)_{pq})^{[i+1]} = (\mathrm{Jac}_x(f^{[i]})_{pq})^{[2]}$$

by Definition (14) and by the induction step. Using now Equation (14) and using the fact (Equation (11)) that :

$$(\mathrm{Jac}_x(f^{[i]}))_{pq} = \sum_{k=1}^{n} \frac{\partial f_p^{[i]}}{\partial z_k} J_{kq}$$

we have :

$$(\mathrm{Jac}_x(f)_{pq})^{[i+1]} = \sum_{k,l=1}^{n} \frac{\partial^2 f_p^{[i]}}{\partial z_k \partial z_l} f_l J_{kq} + \sum_{r=1}^{n} \frac{\partial f_p^{[i]}}{\partial z_r} \left(\sum_{t=1}^{n} \frac{\partial f_r}{\partial z_t} J_{tq} \right) \quad (22)$$

which is thus seen to be equal to $\mathrm{Jac}_x(f^{[i+1]})_{pq}$ by Equation (21). $\square$

Equation (17) is simpler to compute since we already computed the Lie derivative of f, in Equation (16), the Jacobian calculation being by itself rather inexpensive.

Computing the center of the inner-approximation.

Note that Equation (16) is required for estimating the inner-approximations as in Equation (18) : we need to propagate a "center" in a flowpipe of solutions of ODE (1), at each time step t_j in our time grid, and this central solution is certainly not in general derivable from the outer-approximation of the flowpipe, e.g. as its midpoint.

We then use the same Taylor expansion, but with different initial conditions, to compute in (15) an outer-approximation of the solution of system(2) with $z(t_0) = \tilde{z_0}$, used as the center in inner-approximation (18), and in (16) an outer-approximation of the solution of the same system but with uncertain $z(t_0) \in \boldsymbol{z}_0$, used to compute the Taylor coefficients in Equation (17) which outer-approximates the Jacobian of the flow with respect to the initial condition z_0.

EXAMPLE 5.

— JE VAIS COMPLETER

$([2.0750, 2.0751], [0.1249, 0.1251])$

Calculations of coefficients of the Taylor models.

We need to outer-approximate the values of some functions (in particular, all Lie derivatives, which are coefficients in the Taylor models) in Equations (15-17). We have a wide choice from all the existing set-based methods, from interval computations to Bernstein polynomials [**?**] if f is polynomial. We will use in our running example simple affine arithmetic [2] and will compare it with interval computations, that we both used in our prototype implementation, in Section 4. Affine arithmetic was also used in [7] for inner-approximations of discrete dynamical systems. The interest of this approach, using affine arithmetic, is that we can use the results from [6] to get good estimates of the joint inner range of the state variables z_j, altogether, when needed. In practice though, this is rarely needed, as, either for estimating guards of the form $p(z)?0$ (where ? may be equality, less or equal etc.) when analyzing hybrid systems, or for proving some specification depending on arithmetic expressions of the state variables p again, we can just evaluate $]p(z)[$ by using simple affine arithmetic and Equations (18)

Example 6. *We compute a first-order Taylor model for the Brusselator, using Equation (6) with $k = 2$, and using affine arithmetic to compute $f^{[i]}([\boldsymbol{z}_j])$. and $Jac_x(f^{[i]})([\boldsymbol{r}_{j+1}])[\boldsymbol{J}_j]$*

— Tout a coup j'ai un doute, ce n'est pas $\text{Jac}_x(f^{[i]}([\boldsymbol{z}_j])[\boldsymbol{J}_j]$ plutot?

We start with $x_0 = ([2, 2.5], [0.1, 0.15])$, hence, in affine arithmetic, $x_0 = \left(\frac{83}{40} + \frac{3}{40}\epsilon_1, \frac{1}{8} + \frac{1}{40}\epsilon_2\right)$. Evaluating $f^{[1]} = f$ using simple rules from affine arithmetic, we find :

$$f^{[1]}(x_0) = \begin{pmatrix} \left(-\frac{119919}{51200} - \frac{1173}{12800}\epsilon_1 + \frac{41361}{256000}\epsilon_2 \right. \\ \left. + \frac{27}{51200}\eta_1 + \frac{3015}{256000}\eta_2\right) \\ \left(\frac{64879}{51200} + \frac{213}{12800}\epsilon_1 - \frac{41361}{256000}\epsilon_2 \right. \\ \left. - \frac{27}{51200}\eta_1 - \frac{3015}{256000}\eta_2\right) \end{pmatrix} \quad (23)$$

$$Jac_x(f^{[1]})([x_0])_{11} = -\frac{391}{320} + \frac{9}{320}\epsilon_1 + \frac{249}{1600}\epsilon_2 + \frac{9}{1600}\eta_3 \quad (24)$$

$$Jac_x(f^{[1]})([x_0])_{12} = \frac{41361}{6400} + \frac{747}{1600}\epsilon_1 + \frac{27}{6400}\eta_1 \quad (25)$$

$$Jac_x(f^{[1]})([x_0])_{21} = \frac{71}{320} - \frac{9}{320}\epsilon_1 - \frac{249}{1600}\epsilon_2 - \frac{9}{1600}\eta_3 \quad (26)$$

$$Jac_x(f^{[1]})([x_0])_{22} = -\frac{41361}{6400} - \frac{747}{1600}\epsilon_1 - \frac{27}{6400}\eta_1 \quad (27)$$

To determine the remainders, we first compute $f_i^{[2]}$ and then $Jac_x(f_i^{[2]})$. For instance we have :

$$f_1^{[2]} = -2 + 4x_1 + 3x_1x_2 + \frac{3}{2}x_1^3 - 9x_1^2x_2 + \frac{9}{2}x_1^3x_2^2 - \frac{9}{4}x_1^4x_2 \quad (28)$$

$$\begin{aligned} Jac_x(f^{[2]})_{11}(x_1, x_2)(J_{11}, J_{21}) &= (3x_2f_1 + 3x_1f_2)J_{11} \\ &+ (-2 + 3x_1x_2)Jac_x(f^{[1]})_{11}(x_1, x_2)(J_{11}, J_{21}) \\ &+ 3x_1f_1J_{21} + \frac{3}{2}x_1^2Jac_x(f^{[1]})_{21}(x_1, x_2)(J_{11}, J_{21}) \end{aligned} \quad (29)$$

where $Jac_x(f^{[1]})_{11}$ and $Jac_x(f^{[1]})_{21}$ are given in Example 3.

Now again, we are applying affine arithmetic to compute $f^{[2]}([\boldsymbol{r}_1])$ and $Jac_x(f^{[2]})([\boldsymbol{r}_1])[\boldsymbol{R}_1]$ given the rough enclosures $\boldsymbol{r}_1$ and $\boldsymbol{R}_1$ computed in Example 4 and we find :

$$f^{[2]}([\boldsymbol{r}_1]) = \left(\textit{A completer} \right)$$

$$Jac_x(f^{[2]})([\boldsymbol{r}_1])[\boldsymbol{R}_1] = \left(\textit{A completer} \right)$$

3.4 Step 3: computing the inner-approximation

It must be noted that the algorithm described in Section 3.1 fully relies on outer-approximations in each step, to deduce an inner-approximation at Step 3. This means that we can soundly compute and implement our approach using interval-based methods with outward rounding as classically.

Also, the wider the outer-approximation in Taylor models (15-17), the tighter thus the less accurate the inner-approximation (18): it can even lead to an empty inner-approximation if the result of Equation (18) in Kaucher arithmetic is not an improper interval.

The phenomenom we mentionned above can occur in two ways. First, note that $[\overline{z_0}, \underline{z_0}] - \tilde{z}_0$ is an improper interval that belongs to dual $\mathcal{Z}$ as defined in Section 2. The outer-approximation of the Jacobian matrix, $[\boldsymbol{J}](t, t_j, [\boldsymbol{z}_j])$ is a proper interval. The Kaucher multiplication as defined in Table 1 will yield a non-zero improper interval only if$[\boldsymbol{J}](t, t_j, [\boldsymbol{z}_j])$ does not contain 0. And, in this case, the result of this multiplication will depend on the lower bound of the absolute value of the Jacobian (while the same mean-value theorem used for outer-approximation would imply a multiplication of proper intervals that would, in the same case, depend on the upper bound of the absolute value of the Jacobian). The larger this lower bound, the wider the inner-approximation.

Suppose now that the Kaucher multiplication indeed yields an improper interval. It will then be added to (proper) outer-approximation $[\tilde{\boldsymbol{z}}](t, t_j, [\tilde{\boldsymbol{z}}_j])$ of the solution at time t of the solution of the system starting from point $\tilde{\boldsymbol{z}}_0$. Ideally, this should be tight, but if this interval is wider than the improper interval resulting from the Kaucher multiplication, then the sum of the two intervals - computed using the extension of interval addition - will be proper, and the inner-approximation empty. This will be examplified in the experiments.

It must be noted that the quality of the inner-approximation is strongly tied to the quality of the outer-approximation. In particular, as we rely on Taylor models, if necessary we can locally improve the quality by using higher-order models. And as we know that the exact enclosure of the uncertain system lies between the inner and outer-approximated flows, we can bound the approximation error at each instant, and use this information to dynamically refine the appproximation when this error becomes larger than some threshold: indeed, nothing prevents us from using different orders of the Taylor models at different time steps.

Example 7.

— Eric, completer (calcul final d'une sous-approx

4. EXPERIMENTS AND BENCHMARKS

We have implemented a first prototype version of our approach in Matlab, relying on the Intlab [15] toolbox for computations with interval and affine arithmetic. It allows us to first demonstrate in Section 4.1 the good behaviour of our inner-approximated flowpipes on the quite difficult Brusselator model. Then, in Section 4.2, we provide some elements of comparisons to the experimental results given in the related work.

4.1 Brusselator

We consider in this section the following instance of the Brusselator system, slightly different from the version of Example 2, but which is the one studied in the related work:

$$\begin{cases} \dot{x}_1 = 1 + x_1^2x_2 - 2.5x_1 \\ \dot{x}_2 = 1.5x_1 - x_1^2x_2 \end{cases}$$

with $x_1(0) \in [0.9, 1]$ and $x_2(0) \in [0, 0.1]$.

We first use Taylor models of order 3 in time, which we first evaluate for inner and outer-approximations with plain interval arithmetic. When computing flowpipes for time ranging between 0 and 1, we see, in Figure 1, that the width of the outer-approximation (external dashed lines) quickly increases, while the width of the inner-approximation (internal dotted lines) decreases even more quickly, until it disappears at time 0.6: after this time, the inner-approximation is empty. Indeed, it is well known that effective methods should be used to control this wrapping effect, for instance mean-value forms or matrix preconditioning. In our implementation, we control wrapping by using affine arithmetic to evaluate these models. While accurate, it will be more costly than using a method specially optimized for these Taylor model. However, this is convenient for prototyping, as we rely on the implementation of affine arithmetic present in the Intlab toolbox.

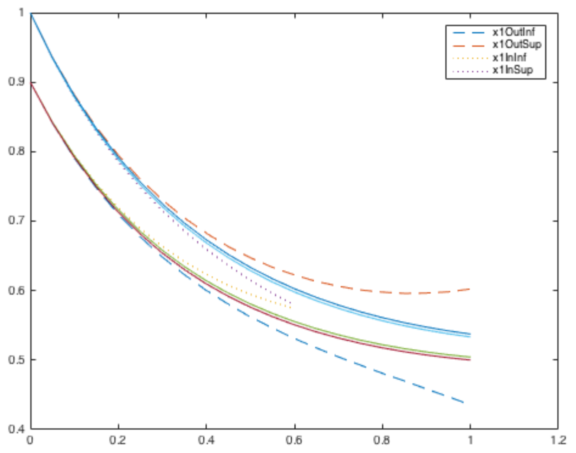

Figure 1: Brusselator, evaluation of the Taylor models (order 3) with interval vs affine arithmetic

In the following experiments, we now use Taylor models of order 4, and represent respectively in Figures 2 and 3, the inner and outer approximated flowpipes for variables $x1$ and $x2$, with an evaluation in affine arithmetic, up to a maximum time $t = 10$. We see that for both variables, while the width of the outer-approximation stays relatively stable with time, the width of the inner-approximation can decrease until the inner-approximation becomes empty, but the width can still later be non-zero again. This is the case for instance on variable x_2 around time $t = 4$. At first look, this could appear as a bug in our implementation, but this is not the case. This phenomenon is an illustration of the fact detailed in Section 3.4, that when adding an improper with a proper interval, we can get a proper interval, which results in anempty inner-approximation, but this does not prevent us to keep computing the Taylor models that can give non-empty inner-approximations at later times, depending on the behaviour of the Jacobian of the flow.

Sylvie : reflechir a ce qu'on peut en deduire sur le systeme (cycle?), et si on veut que je fasse d'autres experiences ?

4.2 Comparisons to the related work

We provide in this section some elements of comparisons to the experimental results given in the related work [1, 16]. Let us highlight that is difficult to have a fair comparison, as on the one hand, evaluating the compared accuracy of these methods is difficult, and on the other hand we only

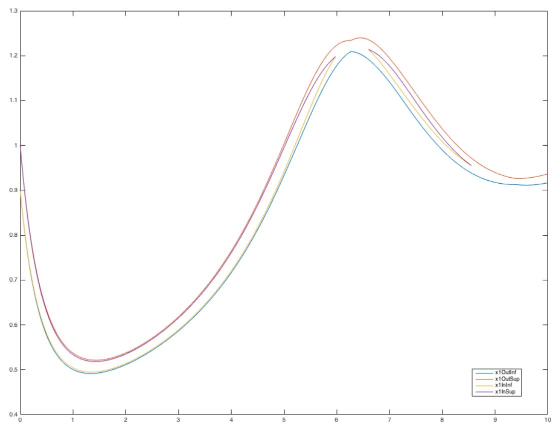

Figure 2: Brusselator (x1), evaluation of the Taylor models (order 4) with affine arithmetic

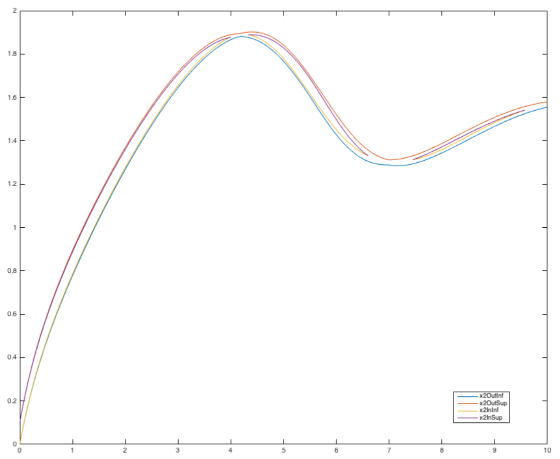

Figure 3: Brusselator (x2), evaluation of the Taylor models (order 4) with affine arithmetic

used a very basic Matlab implementation of our approach, whereas the related work relies on actual implementations, using optimized solvers such as VNode. In order to partially account for this overhead of our preliminary implementation, we will also compare in our experiments the computation time of our inner (and outer-) approximated flowpipes, with the time taken by our implementation for the classical outer-approximation of Taylor model flowpipes.

Among the examples studied in both [1] and [16], we first selected the version of the Brusselator introduced in section 4.1 as a representative of the systems of low degree. We chose as second example the following biological system, because of its higher degree (7), to investigate the way our approach scales.

$$\begin{pmatrix} \dot{x_1} \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} = \begin{pmatrix} -0.4x_1 + 5x_3x_4 \\ 0.4x_1 - x_2 \\ x_2 - 5x_3x_4 \\ 5x_5x_6 - 5x_3x_4 \\ -5x_5x_6 + 5x_3x_4 \\ 0.5x_7 - 5x_5x_6 \\ -0.5x_7 + 5x_5x_6 \end{pmatrix} \tag{30}$$

4.2.1 Comparison to [1]

In [1], the accuracy of computations is measured by the minimum width ratio

$$\gamma_{\min} = \min \frac{\gamma_u(v)}{\gamma_o(v)}, v \in V$$

where V is a set of vectors, and $\gamma_u(v)$ and $\gamma_o(v)$ measure respectively the width of the inner-approximation and outer-approximation in direction $v \in V$. Intuitively, the larger this ratio, the better the approximation. Our method naturally gives inner ranges for the projection of the flow system on its state variables (although we can also directly deduce from our Taylor models an inner-approximation of the range of any linear function of these variables, and also define a joint range, as shown in [6, 7]).

— Deja dit avant, on insiste ou pas?

We thus measure in our case

$$\gamma_{\min} = \min_i \frac{\gamma_u(x_i)}{\gamma_o(x_i)},$$

where the x_i are the state variables of the system. We think this corresponds to the measure that was used for experiments in [1], as they mention that the vectors are selected along the dimensions (axis-aligned).

Comparison on the Brusselator.

We first consider the Brusselator system, with an initial set taken in [1], defined by $x_1 \geq 0.9$, $x_2 \geq 0$, $x_1 + x_2 - 1 \leq 0$, which we can project on $1 \geq x_1 \geq 0.9$, $0.1 \geq x_2 \geq 0$. Naturally, this outer-approximation of the initial set is quite inaccurate, which will result in a lower quality of the inner-approximation that must be taken into account in the comparison to the results of [1]. In our approach, we actually can consider initial sets that are not given as intervals, for instance given as zonotopes (in the present case, we could use a zonotopic outer-approximation of the simplex), but we did not investigate this here.

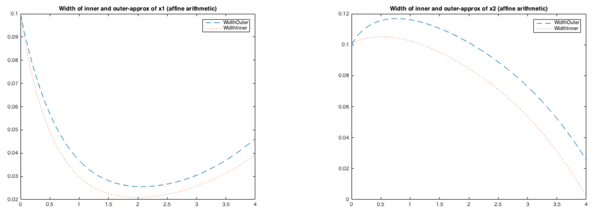

Figure 4: Width of the inner- and outer-approximations of x1 and x2 with time (order 4, h=0.02)

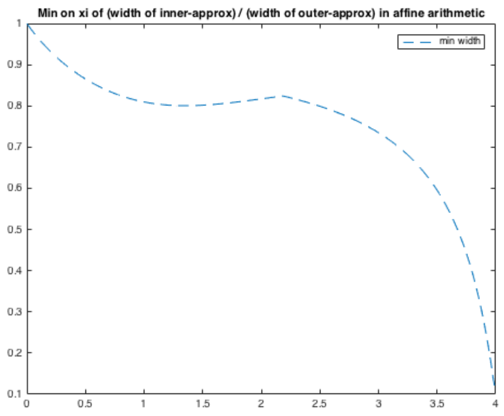

Figure 5: Evolution of $\gamma_{\min}$ with time

In [1], the authors study the result for T = 3 and T = 4. We choose, as they do, 4th order Taylor models, but take a larger time step $h = 0.05$ in order to keep comparable (though much larger here) analysis time. In this case, our implementation until $T = 4$ takes a total of 776 seconds (to compute both outer and inner approximations), where [1] takes 89 seconds. Our implementation is thus almost an order of magnitude larger, however we believe this is very much due to our naive implementation with matlab and the affine arithmetic of Intlab. This can be for instance quantified by measuring the time that our implementation of the classical outer-approximated Taylor Model takes on the same example with the same parameters: it takes here 213 seconds, so that the overhead due to inner-approximation is really of the same order of magnitude (intuitively, the time from outer to inner-approximation with opur approach should indeed approximately be multiplied by the dimension of the Jacobian).

On Figure 5, we see that at $t = 3$, the relative width of the inner-approximation over the outer-approximation is of order 0.7, which is equal to the value given in Table 1 of [1]. However, this ration decreases quickly, mostly due to the x_2 component, and at $t = 4$, we get a ratio very close to 0.1, instead of 0.55 as in [1]! Indeed, $t = 4$ is a time at which, as already seen in Section 4.1, our inner-approximation of variable x_2 is temporarily of lower quality. We thus got on this example, with the same parameters for the Taylor models, worse inner-approximation quality for a longer time of our implementation. However, our approach allows us here to reduce the order of our models and take larger time steps for very close quality of results on γ: taking order 3 Taylor models and a timestep of 0.1, we now get a

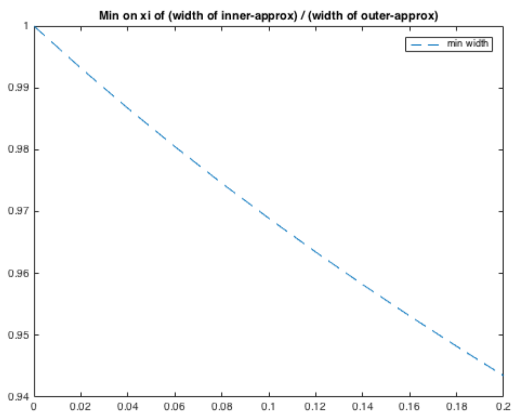

Figure 6: Evolution of $\gamma_{\min}$ with time on the biological system

computation time of 93 seconds, thus comparable now to [1]. Further decreasing the precision starts degrading the quality of results.

Comparison on the biological system.

We now consider the biological system, with initial condition $\boldsymbol{x}_0 \in [0.1, 0.1175] \times ... \times [0.1, 0.1175]$, which is an outer-approximation of the simplex taken in [1], and we compute inner and outer approximated flowpipes for time in [0,0.2], with order 3 Taylor Models and a stepsize of 0.02. The computation completes in 122 seconds, and we get as a measure of quality of the approximation $\gamma(t = 0.2) \approx 0.943$, which is this time much better than the $\gamma(t = 0.2) = .25$ obtained in 632 seconds in [1], in spite of using a larger step size of 0.02 (0.002 in [1]). This seems to confirm that our approaches scales very well to high dimensional systems. Again, the 122 seconds for the inner-approximation flowpipe must also be compared to the 39.7 seconds that our implementation of the classical outer-approximated Taylor Model takes with the same parameters.

We now consider

4.2.2 Comparison to [16]

Comparison on the Brusselator.

In [16] they consider the Brusselator system for an initial value $X = [0.3, 0.4] \times [0.5, 0.7]$, a time frame in [0,1.1], and a time step h=0.05. We compute the inner and outer-approximations, with order 4 Taylor Models, evaluated in AA, in 118.12 seconds. In Figure 7, we represent the inner and outer-approximations obtained for x_1 and x_2. In Figure 8, we represent the maximum distance between points in the inner and the outer approximations, for each variable x_1 and x_2. This distance is somehow comparable to parameter ϵ_M which is used to estimate the precision in [16].

— Sylvie. Continuer

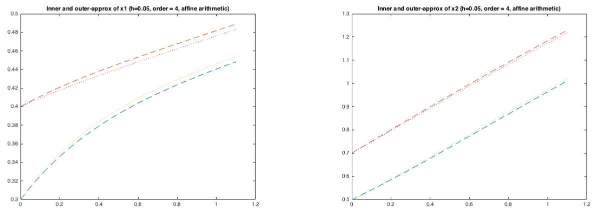

Figure 7: Inner and outer approximations (order 4) for x_1 and x_2

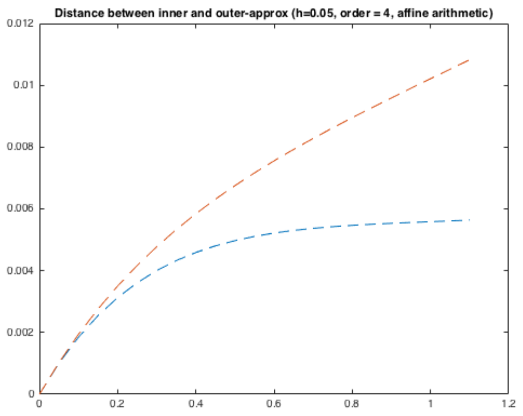

Figure 8: Inner and outer approximations (order 4) for x1 and x2

— De ce que je comprends, leur ϵ_M est la précision avec laquelle ils donnent une surapprox de la frontiere du reachable set. Leur precision c'est la taille (en chaque coordonnee) des boites qui font cette surapprox. Donc au pire ils sont a ϵ_M de la frontiere, dans cette collection de boites, dans chaque direction. Apres ils trouvent un polytope maximal à l'interieur de cette approximation de frontiere par collection de boites. Donc ben difficile a comparer...Par rapport au vrai reachable set ils seront en general plus ou moins a ϵ_M de la, mais peut-etre bcp plus, peut-etre bcp moins s'ils ont de la chance... Eux ils font avec $\epsilon_M = 0.001$ et $\epsilon_M = 0.0002$; je n'ai pas trouve a quel ordre ils utilisent VNODE (j'ai souvenir d'un ordre 20 que je ne trouve plus?). La a cause du x_1 pour une fois, on est en gros potentiellement 10 fois moins bons, mais il y a aussi la distance de la surapprox au vrai reachable set. On pourrait aussi utiliser un modele de Taylor surapprox (voire VNODE?) a ordre bcp plus eleve pour affiner cette distance? Ou alors integrer le brusselator, numeriquement au hasard sur pas mal d'etats initiaux et voir la distance? (un peu comme eux avec leurs simus RK4?). On peut essayer aussi au moins a l'ordre 5 voir si on ameliore cette distance? Eux meme avec VNODE, ils mettent 55s pour le premier ϵ_M et 410s pour le deuxieme, on devrait pouvoir se comparer agreablement vu qu'on a du Matlab, et plein de choses non optimisees?

Comparison on the biological system.

4.3 Other examples

Integrable systems.
To make precision estimates of our method.

$$\dot{u} = u^2$$

with $u(0) = u_0$. Then $u = \frac{u_0}{1-u_0 t}$
The logistic equation :

$$\dot{u} = \lambda u(1-u)$$

with $u(0) = u_0$. Then $u(t) = \frac{u_0 e^{\lambda t}}{1-u_0+u_0 e^{\lambda t}}$.

— PAS MAL D'AUTRES DANS HTTP: //WWW.MATH.UMN.EDU/~OLVER/AM_/ODZ.PDF MAIS LES DEUX AU DESSUS PARAISSENT SIMPLES ET RAISONNABLES, SURTOUT LA LOGISTIC EQUATION?

$$\dot{y} = \frac{a}{\sqrt{y}} + y$$

with $y(a) = 1$ Solution is :

$$y(t) = \left(exp\left(\frac{3t}{2} - \frac{3a}{2} + log(a+1)\right) - a\right)^{\frac{2}{3}}$$

(from Mathworks - can find $y(0)$ and translate it into polynomial system?)

4.4 Conclusion

Resultats preliminaires qui peuvent ne pas sembler que positif mais a priori essentiellement dus a la presente implem

— - COMPARAISON SUR / SOUS-APPROX => SOUS-APPROX COMPETITIVE EN COMPLEXITE PAR RAPPORT AUX MODELES DE TAYLOR CLASSIQUES
- BON PASSAGE A L'ECHELLE SUR DEGRE 7 ?
- IA / AA LAISSANT ETENDRE QUE IMPLEM AA UTILISEE TRES SOUS-OPTIMALE - UN DES INTERETS EST QU'ON PEUT RAFFINER LA PRECISION (SUR ET SOUS-APPROXIMEE) EN MONTANT EN ORDRE A LA DEMANDE, DU STYLE ALGO A PAS OU ORDRE ADAPTATIF PAR CETTE ESTIMATION DIRECTE DE L'ERREUR!

5. CONCLUSION AND FUTURE WORK

A natural future extension of the present work, that we are interested in, is the inner-approximation of reachable sets in presence of guards and constraints, so as to handle general hybrid systems. Our approach allows us to inner-approximate not only the variables as demonstrated here, but also the projection on whatever function of these variables. Also, the symbolic information included in our model if we have a relational information in the evaluation of the Taylor model coefficients, such as is the case when we evaluate them with affine arithmetic, will allow us to use existing work on the inner-approximation of joint range of functions and constraint solving.

— AJOUTER REF?

In that respect, our current matlab prototype, while producing interesting results for the continuous case, does not allow us to handle this symbolic information in a satisfying way, and some engineering improvement is needed and planned.

- combination with backward inner-approximated analyses ?
- abstract model-checking ?

6. REFERENCES

[1] X. Chen, S. Sankaranarayanan, and E. Abraham. Under-approximate flowpipes for non-linear continuous systems. In *Proc. of Formal Methods in Computer-Aided Design (FMCAD'14)*, pages 59–66. IEEE/ACM, 2014.

[2] J. L. D. Comba and J. Stolfi. Affine arithmetic and its applications to computer graphics. *Proceedings of SIBGRAPI*, 1993.

[3] A. Girard, C. L. Guernic, and O. Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. In *Hybrid Systems: Computation and Control, 9th International Workshop, HSCC 2006, Santa Barbara, CA, USA, March 29-31, 2006, Proceedings*, pages 257–271, 2006.

[4] A. Goldsztejn, D. Daney, M. Rueher, and P. Taillibert. Modal intervals revisited: a mean-value extension to generalized intervals. In *QCP'05*, 2005.

[5] A. Goldsztejng and W. Hayes. Rigorous inner approximation of the range of functions. In *Proceedings of the 12th GAMM - IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics*, SCAN '06. IEEE Computer Society, 2006.

[6] E. Goubault, M. Kieffer, O. Mullier, and S. Putot. General inner approximation of vector-valued functions. *Reliable Computing*, 18:117–143, 2013.

[7] E. Goubault, O. Mullier, S. Putot, and M. Kieffer. Inner approximated reachability analysis. In *17th International Conference on Hybrid Systems: Computation and Control (part of CPS Week), HSCC'14, Berlin, Germany, April 15-17, 2014*, pages 163–172, 2014.

[8] E. Goubault and S. Putot. Under-approximations of computations in real numbers based on generalized affine arithmetic. In *SAS*, pages 137–152, 2007.

[9] L. Jaulin, M. Kieffer, and O. Didrit. *Applied interval analysis : with examples in parameter and state estimation, robust control and robotics.* Springer, 2001.

[10] E. Kaucher. Interval analysis in the extended interval space IR. *Comput. (Supplementum) 2*, 1980.

[11] C. Le Guernic. *Reachability Analysis of Hybrid Systems with Linear Continuous Dynamics.* PhD thesis, University Joseph Fourier, 2009.

[12] R. E. Moore. *Interval analysis.* 1966.

[13] N. S. Nedialkov, K. R. Jackson1, and G. F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Appl. Math. Comput.*, 105(1):21–68, Oct. 1999.

[14] M. Neher, K. R. Jackson, and N. S. Nedialkov. On taylor model based integration of odes. *SIAM J. Numer. Anal*, 45:2007.

[15] S. Rump. INTLAB - INTerval LABoratory. In *Developments in Reliable Computing*, pages 77–104. Kluwer Academic Publishers, 1999.

[16] B. Xue, Z. She, and A. Easwaran. Under-approximating backward reachable sets by polytopes. In *CAV*, May 2016.

[17] P. Zgliczynski. c^1 lohner algorithm. *Foundations of Computational Mathematics*, 2(4):429–465, 2002.