Control Synthesis of Nonlinear Sampled Switched Systems using Euler’s Method

A. Le Coënt∗  F. De Vuyst  L. Chamoin
CMLA, CNRS & ENS Paris-Saclay  LMT, CNRS & ENS Paris-Saclay
lecoent@cmla.ens-cachan.fr

L. Fribourg
LSV, CNRS, INRIA & ENS Paris-Saclay

In this paper, we propose a symbolic control synthesis method for nonlinear sampled switched systems whose vector fields are one-sided Lipschitz. The main idea is to use an approximate model obtained from the forward Euler method to build a guaranteed control. The benefit of this method is that the error introduced by symbolic modeling is bounded by choosing suitable time and space discretizations. The method is implemented in the interpreted language Octave. Several examples of the literature are performed and the results are compared with results obtained with a previous method based on the Runge-Kutta integration method.

1 Introduction

As said in [10], in the methods of symbolic analysis and control of hybrid systems, the way of representing sets of state values and computing reachable sets for systems defined by ordinary differential equations (ODEs) is fundamental (see, e.g., [2, 14]). An interesting approach appeared recently, based on the propagation of reachable sets using guaranteed Runge-Kutta methods with adaptive step size control (see [6, 17]). In [10] such guaranteed integration methods are used in the framework of sampled switched systems.

Given an ODE of the form \( \dot{x}(t) = f(t, x(t)) \), and a set of initial values \( X_0 \), a symbolic (or “set-valued”) integration method consists in computing a sequence of approximations \( \{t_n, \tilde{x}_n\} \) of the solution \( x(t; x_0) \) with \( x_0 \in X_0 \) such that \( \tilde{x}_n \simeq x(t_n; x_{n-1}) \). Symbolic integration methods extend classical numerical integration methods which correspond to the case where \( X_0 \) is just a singleton \( \{x_0\} \). The simplest numerical method is Euler’s method in which \( t_{n+1} = t_n + h \) for some step-size \( h \) and \( \tilde{x}_{n+1} = \tilde{x}_n + h f(t_n, \tilde{x}_n) \); so the derivative of \( x \) at time \( t_n \), \( f(t_n, x_n) \), is used as an approximation of the derivative on the whole time interval. This method is very simple and fast, but requires small step-sizes \( h \). More advanced methods coming from the Runge-Kutta family use a few intermediate computations to improve the approximation of the derivative. The general form of an explicit \( s \)-stage Runge-Kutta formula of the form \( \tilde{x}_{n+1} = \tilde{x}_n + h \sum_{i=1}^{s} b_i k_i \) where \( k_i = f(t_n + c_i h, \tilde{x}_n + h \sum_{j=1}^{i-1} a_{ij} k_j) \) for \( i = 2, 3, ..., s \). A challenging question is then to compute a bound on the distance between the true solution and the numerical solution, i.e.: \( \| x(t_n; x_{n-1}) - \tilde{x}_n \| \). This distance is associated to the local truncation error of the numerical method. In [10], such a bound is computed using the Lagrange remainders of Taylor expansions. This is achieved using affine arithmetic [27] (by application of the Banach’s fixpoint theorem and Picard-Lindelöf operator, see [23]). In the end, the Runge-Kutta based method of [10] is an elaborated method that requires the use of affine arithmetic, Picard iteration and computation of Lagrange remainder.

∗ corresponding author

© A. Le Coënt, F. De Vuyst, L. Chamoin & L. Fribourg
This work is licensed under the Creative Commons Attribution License.
In contrast, in this paper, we use ordinary arithmetic (instead of affine arithmetic) and a basic Euler scheme (instead of Runge-Kutta schemes). We need neither estimate Lagrange remainders nor perform Picard iteration in combination with Taylor series. Our simple Euler-based approach is made possible by having recourse to the notion of one-sided Lipschitz (OSL) function [12]. This allows us to bound directly the global error, i.e. the distance between the approximate point \( \tilde{x}(t) \) computed by the Euler scheme and the exact solution \( x(t) \) for all \( t \geq 0 \) (see Theorem 1).

**Plan.** In Section 2 we give details on related work. In Section 3 we state our main result that bounds the global error introduced by the Euler scheme in the context of systems with OSL flows. In Section 4, we explain how to apply this result to the synthesis of symbolic control of sampled switched systems. We give numerical experiments and results in Section 5 for five examples of the literature, and compare them with results obtained with the method of [10]. We give final remarks in Section 6.

## 2 Related work

Most of the recent work on the symbolic (or set-valued) integration of nonlinear ODEs is based on the upper bounding of the Lagrange remainders either in the framework of Taylor series or Runge-Kutta schemes [2, 5, 6, 8, 9, 10, 20, 24, 26]. Sets of states are generally represented as vectors of intervals (or “rectangles”) and are manipulated through interval arithmetic [22] or affine arithmetic [27]. Taylor expansions with Lagrange remainders are also used in the work of [2], which uses “polynomial zonotopes” for representing sets of states in addition to interval vectors. None of these works uses the Euler scheme nor the notion of one-sided Lipschitz constant.

In the literature on symbolic integration, the Euler scheme with OSL conditions is explored in [12, 18]. Our approach is similar but establishes an analytical result for the global error of Euler’s estimate (see Theorem 1) rather than analyzing, in terms of complexity, the speed of convergence to zero, the accuracy and the stability of Euler’s method.

In the control literature, OSL conditions have been recently applied to control and stabilization [1, 7], but do not make use of Euler’s method. To our knowledge, our work applies for the first time Euler’s scheme with OSL conditions to the symbolic control of hybrid systems.

## 3 Sampled switched systems with OSL conditions

### 3.1 Control of switched systems

Let us consider the nonlinear switched system

\[
\dot{x}(t) = f_{\sigma(t)}(x(t))
\]  

(1)

defined for all \( t \geq 0 \), where \( x(t) \in \mathbb{R}^n \) is the state of the system, \( \sigma(\cdot) : \mathbb{R}^+ \rightarrow U \) is the switching rule. The finite set \( U = \{1, \ldots, N\} \) is the set of switching modes of the system. We focus on sampled switched systems: given a sampling period \( \tau > 0 \), switchings will occur at times \( \tau, 2\tau, \ldots \). The switching rule \( \sigma(\cdot) \) is thus constant on the time interval \([((k-1)\tau, k\tau)\) for \( k \geq 1 \). For all \( j \in U \), \( f_j \) is a function from \( \mathbb{R}^n \) to \( \mathbb{R}^n \). We make the following hypothesis:

\[(H0)\quad \text{For all } j \in U, \ f_j \text{ is a locally Lipschitz continuous map.}\]
As in [15], we make the assumption that the vector field $f_j$ is such that the solutions of the differential equation [1] are defined, e.g., by assuming that the support of the vector field $f_j$ is compact. We will denote by $\phi_\sigma(t;x^0)$ the solution at time $t$ of the system:

$$
\begin{align*}
\dot{x}(t) &= f_{\sigma(t)}(x(t)), \\
x(0) &= x^0.
\end{align*}
$$

(2)

Often, we will consider $\phi_\sigma(t;x^0)$ on the interval $0 \leq t < \tau$ for which $\sigma(t)$ is equal to a constant, say $j \in U$. In this case, we will abbreviate $\phi_\sigma(t;x^0)$ as $\phi_j(t;x^0)$. We will also consider $\phi_\sigma(t;x^0)$ on the interval $0 \leq t < k\tau$ where $k$ is a positive integer, and $\sigma(t)$ is equal to a constant, say $j_k$, on each interval $[(k' - 1)\tau, k'\tau)$ with $1 \leq k' \leq k$; in this case, we will abbreviate $\phi_\sigma(t;x^0)$ as $\phi_\pi(t;x^0)$, where $\pi$ is a sequence of $k$ modes (or “pattern”) of the form $\pi = j_1 \cdot j_2 \cdots j_k$.

We will assume that $\phi_\sigma$ is continuous at time $k\tau$ for all positive integer $k$. This means that there is no “reset” at time $k'\tau$ $(1 \leq k' \leq k)$; the value of $\phi_\sigma(t;x^0)$ for $t \in [(k' - 1)\tau, k\tau]$ corresponds to the solution of $\dot{x}(u) = f_{j_k}(x(u))$ for $u \in [0, \tau]$ with initial value $\phi_\sigma ((k' - 1)\tau;x^0)$.

Given a “recurrence set” $R \subset \mathbb{R}^n$ and a “safety set” $S \subset \mathbb{R}^n$ which contains $R (R \subseteq S)$, we are interested in the synthesis of a control such that: starting from any initial point $x \in R$, the controlled trajectory always returns to $R$ within a bounded time while never leaving $S$. We suppose that sets $R$ and $S$ are compact. Furthermore, we suppose that $S$ is convex.

We denote by $T$ a compact overapproximation of the image by $\phi_j$ of $S$ for $0 \leq t \leq \tau$ and $j \in U$, i.e., $T$ is such that

$$T \supseteq \{ \phi_j(t;x^0) \mid j \in U, 0 \leq t \leq \tau, x^0 \in S \}.$$

The existence of $T$ is guaranteed by assumption (H0). We know furthermore by (H0) that, for all $j \in U$, there exists a constant $L_j > 0$ such that:

$$
\| f_j(y) - f_j(x) \| \leq L_j \| y - x \| \quad \forall x, y \in S.
$$

(3)

Let us define $C_j$ for all $j \in U$:

$$
C_j = \sup_{x \in S} L_j \| f_j(x) \| \quad \text{for all} \quad j \in U.
$$

(4)

We make the additional hypothesis that the mappings $f_j$ are one-sided Lipschitz (OSL) [12]. Formally:

(H1) For all $j \in U$, there exists a constant $\lambda_j \in \mathbb{R}$ such that

$$
\langle f_j(y) - f_j(x), y - x \rangle \leq \lambda_j \| y - x \|^2 \quad \forall x, y \in T,
$$

where $\langle \cdot, \cdot \rangle$ denotes the scalar product of two vectors of $\mathbb{R}^n$.

Remark 1. Constants $\lambda_j$, $L_j$ and $C_j$ ($j \in U$) can be computed using (constrained) optimization algorithms. See Section 5 for details.

### 3.2 Euler approximate solutions

Given an initial point $\tilde{x}^0 \in S$ and a mode $j \in U$, we define the following “linear approximate solution” $\tilde{\phi}_j(t;\tilde{x}^0)$ for $t$ on $[0, \tau]$ by:

$$
\tilde{\phi}_j(t;\tilde{x}^0) = \tilde{x}^0 + tf_j(\tilde{x}^0).
$$

(5)
Note that formula \([5]\) is nothing else but the explicit forward Euler scheme with “time step” \(t\). It is thus a consistent approximation of order 1 in \(t\) of the exact solution of \([1]\) under the hypothesis \(\dot{x}_0 = x_0\).

More generally, given an initial point \(x_0 \in S\) and pattern \(\pi\) of \(U^k\), we can define a “(piecewise linear) approximate solution” \(\tilde{\phi}_\pi(t;\tilde{x}_0)\) of \(\phi_\pi\) at time \(t \in [0,k\tau]\) as follows:

- \(\tilde{\phi}_\pi(0;\tilde{x}_0) = \tilde{x}_0\) if \(\pi = \emptyset\), \(k = 0\) and \(t \in [0,\tau]\), and
- \(\tilde{\phi}_\pi(\tau;\tilde{x}_0) = \frac{\tilde{x}_0}{\tau}\) if \(\pi = \emptyset\), \(k = 1\) and \(t \in [0,\tau]\), and
- \(\tilde{\phi}_\pi(t;\tilde{x}_0) = \tilde{f}_j(\tilde{x}_0) + \tilde{z}(t)\) with \(\tilde{z} = \tilde{\phi}_\pi((k-1)\tau;\tilde{x}_0)\), if \(k \geq 2\), \(t \in [0,\tau]\), \(\pi = j \cdot \pi'\) for some \(j \in U\) and \(\pi' \in U^{k-1}\).

We wish to synthesize a guaranteed control \(\sigma\) for \(f_j\), using the approximate functions \(\tilde{\phi}_\pi\). We define the closed ball of center \(x \in \mathbb{R}^n\) and radius \(r > 0\), denoted \(B(x,r)\), as the set \(\{x' \in \mathbb{R}^n \mid \|x' - x\| \leq r\}\).

Given a positive real \(\delta\), we now define the expression \(\delta_j(t)\) which, as we will see in Theorem 1, represents (an upper bound on) the error associated to \(\tilde{\phi}_j(t;\tilde{x}_0)\) (i.e. \(\|\tilde{\phi}_j(t;\tilde{x}_0) - \phi_j(t;\tilde{x}_0)\|\)).

**Definition 1.** Let \(\delta\) be a positive constant. Let us define, for all \(0 \leq t \leq \tau\), \(\delta_j(t)\) as follows:

- if \(\lambda_j < 0\):
  \[
  \delta_j(t) = \left(\delta^2 e^{\lambda_j t} + \frac{C_j^2}{\lambda_j^2} \left(\tau^2 + \frac{2t}{\lambda_j} + \frac{2}{\lambda_j^2}(1 - e^{\lambda_j t})\right)\right)^{\frac{1}{2}}
  \]

- if \(\lambda_j = 0\):
  \[
  \delta_j(t) = \left(\delta^2 e^0 + C_j^2(-\tau^2 + 2(e^t - 1))\right)^{\frac{1}{2}}
  \]

- if \(\lambda_j > 0\):
  \[
  \delta_j(t) = \left(\delta^2 e^{3\lambda_j t} + \frac{C_j^2}{3\lambda_j^2} \left(-\tau^2 - \frac{2t}{3\lambda_j} + \frac{2}{9\lambda_j^2}(e^{3\lambda_j t} - 1)\right)\right)^{\frac{1}{2}}
  \]

Note that \(\delta_j(t) = \delta\) for \(t = 0\). The function \(\delta_j(\cdot)\) depends implicitly on two parameters: \(\delta \in \mathbb{R}\) and \(j \in U\). In Section 4, we will use the notation \(\delta_j(\cdot)\) where the parameters are denoted by \(\delta'\) and \(j\).

**Theorem 1.** Given a sampled switched system satisfying (H0-H1), consider a point \(\tilde{x}_0\) and a positive real \(\delta\). We have, for all \(x_0 \in B(\tilde{x}_0,\delta)\), \(t \in [0,\tau]\) and \(j \in U\):

\(\phi_j(t;\tilde{x}_0) \in B(\tilde{\phi}_j(t;\tilde{x}_0),\delta_j(t))\).

**Proof.** Consider on \(t \in [0,\tau]\) the differential equations

\[
\frac{dx(t)}{dt} = f_j(x(t))
\]

and

\[
\frac{d\tilde{x}(t)}{dt} = f_j(\tilde{x}_0).
\]

with initial points \(x_0 \in S, \tilde{x}_0 \in S\) respectively. We will abbreviate \(\phi_j(t;\tilde{x}_0)\) (resp. \(\tilde{\phi}_j(t;\tilde{x}_0)\)) as \(x(t)\) (resp. \(\tilde{x}(t)\)). We have

\[
\frac{d}{dt}(x(t) - \tilde{x}(t)) = \left(f_j(x(t)) - f_j(\tilde{x}_0)\right),
\]
then
\[
\frac{1}{2} \frac{d}{dt} (\|x(t) - \bar{x}(t)\|^2) = \langle f_j(x(t)) - f_j(x^0), x(t) - \bar{x}(t) \rangle \\
= \langle f_j(x(t)) - f_j(\bar{x}(t)) + f_j(\bar{x}(t)) - f_j(x^0), x(t) - \bar{x}(t) \rangle \\
= \langle f_j(x(t)) - f_j(\bar{x}(t)), x(t) - \bar{x}(t) \rangle + \langle f_j(\bar{x}(t)) - f_j(x^0), x(t) - \bar{x}(t) \rangle \\
\leq \langle f_j(x(t)) - f_j(\bar{x}(t)), x(t) - \bar{x}(t) \rangle + \|f_j(\bar{x}(t)) - f_j(x^0)\| \|x(t) - \bar{x}(t)\|.
\]

The last expression has been obtained using the Cauchy-Schwarz inequality. Using (H1) and (3), we have
\[
\frac{1}{2} \frac{d}{dt} (\|x(t) - \bar{x}(t)\|^2) \leq \lambda_j \|x(t) - \bar{x}(t)\|^2 + \|f_j(\bar{x}(t)) - f_j(x^0)\| \|x(t) - \bar{x}(t)\| \\
\leq \lambda_j \|x(t) - \bar{x}(t)\|^2 + L_j \|\bar{x}(t) - x^0\| \|x(t) - \bar{x}(t)\| \\
\leq \lambda_j \|x(t) - \bar{x}(t)\|^2 + L_j t \|f_j(x^0)\| \|x(t) - \bar{x}(t)\|.
\]

Using (4) and a Young inequality, we then have
\[
\frac{1}{2} \frac{d}{dt} (\|x(t) - \bar{x}(t)\|^2) \leq \lambda_j \|x(t) - \bar{x}(t)\|^2 + C_j t \|x(t) - \bar{x}(t)\| \\
\leq \lambda_j \|x(t) - \bar{x}(t)\|^2 + C_j t \frac{1}{2} \left( \alpha \|x(t) - \bar{x}(t)\|^2 + \frac{1}{\alpha} \right)
\]
for all \(\alpha > 0\).

- In the case \(\lambda_j < 0\):
  
  For \(t > 0\), we choose \(\alpha > 0\) such that \(C_j t \alpha = -\lambda_j\), i.e. \(\alpha = -\frac{\lambda_j}{C_j t}\). It follows, for all \(t \in [0, \tau]\):

  \[
  \frac{1}{2} \frac{d}{dt} (\|x(t) - \bar{x}(t)\|^2) \leq \frac{\lambda_j}{2} \|x(t) - \bar{x}(t)\|^2 - \frac{C_j t}{2\alpha} = \frac{\lambda_j}{2} \|x(t) - \bar{x}(t)\|^2 - \frac{(C_j t)^2}{2\alpha}.
  \]

  We thus get:

  \[
  \|x(t) - \bar{x}(t)\|^2 \leq \|x^0 - x^0\|^2 e^{\lambda_j t} + \frac{C_j^2}{\lambda_j^2} \left( t^2 + \frac{2t}{\lambda_j} + \frac{2}{\lambda_j^2} \left( 1 - e^{\lambda_j t} \right) \right).
  \]

- In the case \(\lambda_j > 0\):
  
  For \(t > 0\), we choose \(\alpha > 0\) such that \(C_j t \alpha = \lambda_j\), i.e. \(\alpha = \frac{\lambda_j}{C_j t}\). It follows, for all \(t \in [0, \tau]\):

  \[
  \frac{1}{2} \frac{d}{dt} (\|x(t) - \bar{x}(t)\|^2) \leq \frac{3\lambda_j}{2} \|x(t) - \bar{x}(t)\|^2 + \frac{C_j t}{2\alpha} = \frac{3\lambda_j}{2} \|x(t) - \bar{x}(t)\|^2 + \frac{(C_j t)^2}{2\lambda_j}.
  \]

  We thus get:

  \[
  \|x(t) - \bar{x}(t)\|^2 \leq \|x^0 - x^0\|^2 e^{3\lambda_j t} + \frac{C_j^2}{3\lambda_j^3} \left( -t^2 - \frac{2t}{3\lambda_j} + \frac{2}{9\lambda_j^2} (e^{3\lambda_j t} - 1) \right).
  \]
• In the case $\lambda_j = 0$:
  For $t > 0$, we choose $\alpha = \frac{1}{C_j}$. It follows:
  \[ \frac{d}{dt} \left( \| x(t) - \bar{x}(t) \|^2 \right) \leq \| x(t) - \bar{x}(t) \|^2 + C_j t^2 \]
  We thus get:
  \[ \| x(t) - \bar{x}(t) \|^2 \leq \| x^0 - \bar{x}^0 \|^2 e^t + C_j^2 (-t^2 - 2t + 2(\epsilon^2 - 1)) \]
  In every case, since by hypothesis $x^0 \in B(\bar{x}^0, \delta)$ (i.e. $\| x^0 - \bar{x}^0 \|^2 \leq \delta^2$), we have, for all $t \in [0, \tau]$:
  \[ \| x(t) - \bar{x}(t) \| \leq \delta_j(t). \]
  It follows: $\phi_j(t; x^0) \in B(\tilde{\phi}_j(t; \bar{x}^0), \delta_j)$ for $t \in [0, \tau]$.

\[ \square \]

**Remark 2.** In Theorem 2, we have supposed that the step size $h$ used in Euler’s method was equal to the sampling period $\tau$ of the switching system. Actually, in order to have better approximations, it is often convenient to take a fraction of $\tau$ as for $h$ (e.g., $h = \frac{\tau}{10}$). Such a splitting is called “sub-sampling” in numerical methods. See Section 5 for details.

**Corollary 1.** Given a sampled switched system satisfying (H0-H1), consider a point $\bar{x}^0 \in S$, a real $\delta > 0$ and a mode $j \in U$ such that:

1. $B(\bar{x}^0, \delta) \subseteq S$.
2. $B(\tilde{\phi}_j(t; \bar{x}^0), \delta_j(t)) \subseteq S$, and
3. $\frac{d}{dt}(\delta_j(t)) > 0$ for all $t \in [0, \tau]$.

Then we have, for all $x^0 \in B(\bar{x}^0, \delta)$ and $t \in [0, \tau]$: $\phi_j(t; x^0) \in S$.

**Proof.** By items 1 and 2, $B(\tilde{\phi}_j(t; \bar{x}^0), \delta_j(t))$ for $t = 0$ and $t = \tau$. Since $\delta_j(\cdot)$ is convex on $[0, \tau]$ by item 3, and $S$ is convex, we have $B(\tilde{\phi}_j(t; \bar{x}^0), \delta_j(t)) \subseteq S$ for all $t \in [0, \tau]$. It follows from Theorem 1 that $\phi_j(t; x^0) \in B(\tilde{\phi}_j(t; \bar{x}^0), \delta_j(t)) \subseteq S$ for all $1 \leq t \leq \tau$.

\[ \square \]

**Remark 3.** Condition 3 of Corollary 1 on the convexity of $\delta_j(\cdot)$ on $[0, \tau]$ can be established again using an optimization function (see Section 5).

## 4 Application to control synthesis

Consider a point $\bar{x}^0 \in S$, a positive real $\delta$ and a pattern $\pi$ of length $k$. Let $\pi(k')$ denote the $k'$-th element (mode) of $\pi$ for $1 \leq k' \leq k$. Let us abbreviate the $k'$-th approximate point $\tilde{\phi}_\pi(k' \tau; \bar{x}^0)$ as $\tilde{x}^k$ for $k' = 1, \ldots, k$, and let $\tilde{x}^k = \bar{x}^0$ for $k' = 0$. It is easy to show that $\tilde{x}^k_\pi$ can be defined recursively for $k' = 1, \ldots, k$, by:
\[ \tilde{x}^k_\pi = \tilde{x}^{k-1} + \tau f_j(\tilde{x}^{k-1}) \] with $j = \pi(k')$.

Let us now denote by $\delta^k_\pi$ (an upper bound on) the error associated to $\tilde{x}^k_\pi$, i.e. $\| \tilde{x}^k_\pi - \phi_\pi(k' \tau; \bar{x}^0) \|$. Using repeatedly Theorem 1, $\delta^k_\pi$ can be defined recursively as follows:

For $k' = 0$: $\delta^k_\pi = \delta$, and for $1 \leq k' \leq k$: $\delta^{k'}_\pi = \delta^{k'}_j(\tau)$ where $\delta'$ denotes $\delta^{k'-1}$, and $j$ denotes $\pi(k')$.

Likewise, for $0 \leq t \leq k \tau$, let us denote by $\delta_\pi(t)$ (an upper bound on) the global error associated to $\tilde{\phi}_\pi(t; \bar{x}^0)$ (i.e. $\| \tilde{\phi}_\pi(t; \bar{x}^0) - \phi_\pi(t; \bar{x}^0) \|$). Using Theorem 1, $\delta_\pi(t)$ can be defined itself as follows:
Corollary 2. Given a switched system satisfying (H0-H1), consider a point \( \tilde{x}^0 \in \mathcal{S} \), a positive real \( \delta \) and a pattern \( \pi \) of length \( k \) such that, for all \( 1 \leq k' \leq k \):

1. \( B((\tilde{x}^0_k, \pi), \delta^k_{\pi}) \subseteq \mathcal{S} \)
2. \( \frac{d^2(\delta^j(t))}{dt^2} > 0 \) for all \( t \in [0, \tau] \), with \( j = \pi(k') \) and \( \delta' = \delta^k_{\pi} \).

Then we have, for all \( x^0 \in B(\tilde{x}^0, \delta) \) and \( t \in [0, k\tau] \): \( \phi_{\pi}(t; x^0) \in \mathcal{S} \).

Proof. By induction on \( k \) using Corollary 1.

The statement of Theorem 2 is illustrated in Figure 1 for \( k = 2 \). From Theorem 2, it easily follows:

![Figure 1: Illustration of Theorem 2](image)

Corollary 2. Given a switched system satisfying (H0-H1), consider a positive real \( \delta \) and a finite set of points \( \tilde{x}_1, \ldots, \tilde{x}_m \) of \( \mathcal{S} \) such that all the balls \( B(\tilde{x}_i, \delta) \) cover \( \mathcal{R} \) and are included into \( \mathcal{S} \) (i.e. \( \mathcal{R} \subseteq \bigcup_{i=1}^{m} B(\tilde{x}_i, \delta) \subseteq \mathcal{S} \)). Suppose furthermore that, for all \( 1 \leq i \leq m \), there exists a pattern \( \pi_i \) of length \( k_i \) such that:

1. \( B((\tilde{x}_i)^k_{\pi_i}, \delta^k_{\pi_i}) \subseteq \mathcal{S} \), for all \( k' = 1, \ldots, k_i - 1 \)
2. \( B((\tilde{x}_i)^k_{\pi_i}, \delta^k_{\pi_i}) \subseteq \mathcal{R} \).
3. \( \frac{d^2(\delta^j(t))}{dt^2} > 0 \) with \( j = \pi_i(k') \) and \( \delta' = \delta^k_{\pi_i} \), for all \( k' \in \{1, \ldots, k_i\} \) and \( t \in [0, \tau] \).

These properties induce a control \( \sigma \) which guarantees

---

1Given an initial point \( x \in \mathcal{R} \), the induced control \( \sigma \) corresponds to a sequence of patterns \( \pi_i, \pi_2, \ldots \) defined as follows: Since \( x \in \mathcal{R} \), there exists a a point \( \tilde{x}_1 \) with \( 1 \leq i_1 \leq m \) such that \( x \in B(\tilde{x}_1, \delta) \); then using pattern \( \pi_{i_1} \), one has: \( \phi_{\pi_{i_1}}(k_1, \tau; x) \in \mathcal{R} \). Let \( x' = \phi_{\pi_{i_1}}(k_1, \tau; x) \); there exists a point \( \tilde{x}_2 \) with \( 1 \leq i_2 \leq m \) such that \( x' \in B(\tilde{x}_2, \delta) \), etc.
• (safety): if \( x \in R \), then \( \phi_\sigma(t; x) \in S \) for all \( t \geq 0 \), and

• (recurrence): if \( x \in R \) then \( \phi_\sigma(k\tau; x) \in R \) for some \( k \in \{k_1, \ldots, k_m\} \).

Corollary 2 gives the theoretical foundations of the following method for synthesizing \( \sigma \) ensuring recurrence in \( R \) and safety in \( S \):

- we (pre-)compute \( \lambda_j, L_j, C_j \) for all \( j \in U \);
- we find \( m \) points \( \tilde{x}_1, \ldots, \tilde{x}_m \) of \( S \) and \( \delta > 0 \) such that \( R \subseteq \bigcup_{i=1}^{m} B(\tilde{x}_i, \delta) \subseteq S \);
- we find \( m \) patterns \( \pi_i (i = 1, \ldots, m) \) such that conditions 1-2-3 of Corollary 2 are satisfied.

A covering of \( R \) with balls as stated in Corollary 2 is illustrated in Figure 2. The control synthesis method based on Corollary 2 is illustrated in Figure 3 (left) together with an illustration of method of [10] (right).

---

5  Numerical experiments and results

This method has been implemented in the interpreted language Octave, and the experiments performed on a 2.80 GHz Intel Core i7-4810MQ CPU with 8 GB of memory.

The computation of constants \( L_j, C_j, \lambda_j \ (j \in U) \) are realized with a constrained optimization algorithm. They are performed using the “sqp” function of Octave, applied on the following optimization problems:

- Constant \( L_j \):

\[
L_j = \max_{x, y \in S, x \neq y} \frac{||f_j(y) - f_j(x)||}{||y - x||}
\]

- Constant \( C_j \):

\[
C_j = \max_{x \in S} L_j ||f_j(x)||
\]

- Constant \( \lambda_j \):

\[
\lambda_j = \max_{x, y \in T, x \neq y} \frac{\langle f_j(y) - f_j(x), y - x \rangle}{||y - x||^2}
\]
Likewise, the convexity test $\frac{d^2(\delta_j(t))}{dt^2} > 0$ can be performed similarly.

Note that in some cases, it is advantageous to use a time sub-sampling to compute the image of a ball. Indeed, because of the exponential growth of the radius $\delta_j(t)$ within time, computing a sequence of balls can lead to smaller ball images. It is particularly advantageous when a constant $\lambda_j$ is negative. We illustrate this with the example of the DC-DC converter. It has two switched modes, for which we have $\lambda_1 = -0.014215$ and $\lambda_2 = 0.142474$. In the case $\lambda_j < 0$, the associated formula $\delta_j(t)$ has the behavior of Figure 4 (a). In the case $\lambda_j > 0$, the associated formula $\delta_j(t)$ has the behavior of Figure 4 (b). In the case $\lambda_j < 0$, if the time sub-sampling is small enough, one can compute a sequence of balls with reducing radius, which makes the synthesis easier.

In the following, we give the results obtained with our Octave implementation of this Euler-based...
5.1 Four-room apartment

We describe a first application on a 4-room 16-switch building ventilation case study adapted from [21]. The model has been simplified in order to get constant parameters. The system is a four room apartment subject to heat transfer between the rooms, with the external environment, the underfloor, and human beings. The dynamics of the system is given by the following equation:

$$\frac{dT_i}{dt} = \sum_{j \in \mathcal{N}^* \setminus \{i\}} a_{ij}(T_j - T_i) + \delta_i b_i(T^{s}_i - T^4_i) + c_i \max\left(0, \frac{V_i - V^{s}_i}{V_i - V^*_{\bar{V}_i}}\right) (T_u - T_i), \quad \text{for } i = 1, \ldots, 4.$$

The state of the system is given by the temperatures in the rooms $T_i$, for $i \in \mathcal{N} = \{1, \ldots, 4\}$. Room $i$ is subject to heat exchange with different entities stated by the indices $\mathcal{N}^* = \{1, 2, 3, 4, u, o, c\}$. We have $T_0 = 30, T_c = 30, T_u = 17, \delta_i = 1$ for $i \in \mathcal{N}$. The (constant) parameters $T^{s}_i, V^{*}_i, \bar{V}_i, a_{ij}, b_i, c_i$ are given in [21]. The control input is $V_i (i \in \mathcal{N})$. In the experiment, $V_1$ and $V_4$ can take the values 0V or 3.5V, and $V_2$ and $V_3$ can take the values 0V or 3V. This leads to a system of the form (1) with $\sigma(t) \in U = \{1, \ldots, 16\}$, the 16 switching modes corresponding to the different possible combinations of voltages $V_i$. The sampling period is $\tau = 30s$. Compared simulations are given in Figure 5. On this example, the Euler-based method works better than DynIBEX in terms of CPU time.

|                  | Euler | DynIBEX |
|------------------|-------|---------|
| $R$              | $[20, 22]^2 \times [22, 24]^2$ | $[22, 24]^2$ |
| $S$              | $[19, 23]^2 \times [21, 25]^2$ | $[21, 25]^2$ |
| $\tau$           | No    | Yes     |
| Time subsampling | Yes   | Yes     |
| Complete control | Yes   | Yes     |
| $\max_{j=1,\ldots,16} \lambda_j$ | $-6.30 \times 10^{-3}$ | $4.18 \times 10^{-6}$ |
| $\max_{j=1,\ldots,16} C_j$ | $4.18 \times 10^{-6}$ | $4.18 \times 10^{-6}$ |
| Number of balls/tiles | 4096 | 252 |
| Pattern length   | 1     | 1       |
| CPU time         | 63 seconds | 249 seconds |

Table 1: Numerical results for the four-room example.

5.2 DC-DC converter

This linear example is taken from [3] and has already been treated with the state-space bisection method in a linear framework in [13].

The system is a boost DC-DC converter with one switching cell. There are two switching modes depending on the position of the switching cell. The dynamics is given by the equation $\dot{x}(t) = A_{\sigma(t)}x(t) + B_{\sigma(t)}$ with $\sigma(t) \in U = \{1, 2\}$. The two modes are given by the matrices:

$$A_1 = \begin{pmatrix} -\frac{R}{L} & 0 \\ 0 & -\frac{1}{C} \frac{1}{V_i}\end{pmatrix}, \quad B_1 = \begin{pmatrix} \frac{V_i}{L} \\ 0 \end{pmatrix}$$
Figure 5: Simulation of the four-room case study with our synthesis method (left) and with the synthesis method of [10] (right).

\[
A_2 = \begin{pmatrix}
-\frac{1}{x_c} (r_l + r_0 r_c) & -\frac{1}{x_c} r_0 r_c \\
\frac{1}{x_c} r_0 r_c & 0
\end{pmatrix}, \quad B_2 = \begin{pmatrix}
u_s \\
0
\end{pmatrix}
\]

with \(x_c = 70, x_l = 3, r_c = 0.005, r_l = 0.05, r_0 = 1, v_s = 1\). The sampling period is \(\tau = 0.5\). The parameters are exact and there is no perturbation. We want the state to return infinitely often to the region \(R\), set here to \([1.55, 2.15] \times [1.0, 1.4]\), while never going out of the safety set \(S = [1.54, 2.16] \times [0.99, 1.41]\). On this example, the Euler-based method fails while DynIBEX succeeds rapidly.

|                | Euler | DynIBEX |
|----------------|-------|---------|
| \(R\)          | \([1.55, 2.15] \times [1.0, 1.4]\) | \([1.54, 2.16] \times [0.99, 1.41]\) |
| \(S\)          | \([1.54, 2.16] \times [0.99, 1.41]\) | \(0.5\) |
| \(\lambda_1\)  | \(-0.014215\) | \(\text{No}\) |
| \(\lambda_2\)  | \(0.142474\) | \(\text{Yes}\) |
| \(C_1\)        | \(6.7126 \times 10^{-5}\) | |
| \(C_2\)        | \(2.6229 \times 10^{-2}\) | |
| Number of balls/tiles | \(x\) | \(48\) |
| Pattern length  | \(x\) | \(6\) |
| CPU time        | \(x\) | \(<1\) second |

Table 2: Numerical results for the DC-DC converter example.

5.3 Polynomial example

We consider the polynomial system taken from [19]:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} = \begin{pmatrix}
-x_2 - 1.5x_1 - 0.5x_1^3 + u_1 \\
x_1 + u_2
\end{pmatrix}.
\]
The control inputs are given by $u = (u_1, u_2) = K_{\sigma(t)}(x_1, x_2)$, $\sigma(t) \in U = \{1, 2, 3, 4\}$, which correspond to four different state feedback controllers $K_1(x) = (0, -x_2^3 + 2), K_2(x) = (0, -x_2), K_3(x) = (2, 10), K_4(x) = (-1.5, 10)$. We thus have four switching modes. The disturbances are not taken into account. The objective is to visit infinitely often two zones $R_1$ and $R_2$, without going out of a safety zone $S$.

|          | Euler          | DynIBEX       |
|----------|----------------|---------------|
| $R_1$    | $[-1, 0.65] \times [0.75, 1.75]$ |               |
| $R_2$    | $[-0.5, 0.5] \times [-0.75, 0.0]$ |               |
| $S$      | $[-2.0, 2.0] \times [-1.5, 3.0]$ |               |
| $\tau$  | 0.15           |               |
| Time subsampling | $\tau/20$ |               |
| Complete control | Yes | Yes |
| $\lambda_1$ | -1.5 |               |
| $\lambda_2$ | -1.0 |               |
| $\lambda_3$ | $-1.1992 \times 10^{-8}$ |           |
| $\lambda_4$ | $-5.7336 \times 10^{-6}$ |           |
| $C_1$    | 641.37        |               |
| $C_2$    | 138.49        |               |
| $C_3$    | 204.50        |               |
| $C_4$    | 198.64        |               |
| Number of balls/tiles | 16 & 16 | 1 & 1 |
| Pattern length  | 8       | 7          |
| CPU time | 29 & 4203 seconds | <0.1 & 329 seconds |

Table 3: Numerical results for the polynomial example example.

For Euler and DynIBEX, the table indicates two CPU times corresponding to the reachability from $R_1$ to $R_2$ and vice versa. On this example, the Euler-based method is much slower than DynIBEX.

### 5.4 Two-tank system

The two-tank system is a linear example taken from [16]. The system consists of two tanks and two valves. The first valve adds to the inflow of tank 1 and the second valve is a drain valve for tank 2. There is also a constant outflow from tank 2 caused by a pump. The system is linearized at a desired operating point. The objective is to keep the water level in both tanks within limits using a discrete open/close switching strategy for the valves. Let the water level of tanks 1 and 2 be given by $x_1$ and $x_2$ respectively. The behavior of $x_1$ is given by $\dot{x}_1 = -x_1 - 2$ when the tank 1 valve is closed, and $\dot{x}_1 = -x_1 + 3$ when it is open. Likewise, $x_2$ is driven by $\dot{x}_2 = x_1$ when the tank 2 valve is closed and $\dot{x}_2 = x_1 - x_2 - 5$ when it is open. On this example, the Euler-based method works better than DynIBEX in terms of CPU time.

### 5.5 Helicopter

The helicopter is a linear example taken from [11]. The problem is to control a quadrotor helicopter toward a particular position on top of a stationary ground vehicle, while satisfying constraints on the relative velocity. Let $g$ be the gravitational constant, $x$ (reps. $y$) the position according to $x$-axis (resp. $y$-axis), $\dot{x}$ (resp. $\dot{y}$) the velocity according to $x$-axis (resp. $y$-axis), $\phi$ the pitch command and $\psi$ the roll command. The possible commands for the pitch and the roll are the following: $\phi, \psi \in \{-10, 0, 10\}$. 
Since each mode corresponds to a pair \((\phi, \psi)\), there are nine switched modes. The dynamics of the system is given by the equation:
\[
\dot{X} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} X + \begin{pmatrix} 0 \\ g \sin(-\phi) \\ 0 \\ g \sin(\psi) \end{pmatrix}
\]
where \(X = (x \ x \ y \ \dot{y})^\top\). Since the variables \(x\) and \(y\) are decoupled in the equations and follow the same equations (up to the sign of the command), it suffices to study the control for \(x\) (the control for \(y\) is the opposite). On this example again, the Euler-based method works better than \(\text{DynIBEX}\) in terms of CPU time.

### 5.6 Analysis and comparison of results

Our method presents the advantage over the work of [10] that no numerical integration is required for the control synthesis. The computations just require the evaluation of given functions \(f_j\) and (global error) functions \(\delta_j\) at sampling times. The synthesis is thus \textit{a priori} cheap compared to the use of numerical integration schemes (and even compared to exact integration for linear systems). However, most of the computation time is actually taken by the search for an appropriate radius \(\delta\) of the balls \(B_i (1 \leq i \leq m)\) that cover \(R\), and the search for appropriate patterns \(\pi_i\) that make the trajectories issued from \(B_i\) return to \(R\).

We observe on the examples that the resulting control strategies synthesized by our method are quite different from those obtained by the Runge-Kutta method of [10] (which uses in particular rectangular tiles instead of balls). This may explain why the experimental results are here contrasted: Euler’s method works better on 3 examples and worse on the 2 others. Besides the Euler method fails on one example.

|                | Euler | DynIBEX |
|----------------|-------|---------|
| \(R\)          | \([-1.5, 2.5] \times [-0.5, 1.5]\) |         |
| \(S\)          | \([-3, 3] \times [-3, 3]\)         |         |
| \(\tau\)       | 0.2   |         |
| Time subsampling| \(\tau/10\) |         |
| Complete control| Yes   | Yes     |
| \(\lambda_1\)  | 0.20711 |         |
| \(\lambda_2\)  | -0.50000 |        |
| \(\lambda_3\)  | 0.20711 |         |
| \(\lambda_4\)  | -0.50000 |        |
| \(C_1\)        | 11.662 |         |
| \(C_2\)        | 28.917 |         |
| \(C_3\)        | 13.416 |         |
| \(C_4\)        | 32.804 |         |
| Number of balls/tiles | 64 | 10       |
| Pattern length | 6 | 6         |
| CPU time       | 58 seconds | 246 seconds |

Table 4: Numerical results for the two-tank example.
(DC-DC converter) while DynIBEX succeeds on all of them. Note however that our Euler-based implementation is made of a few hundreds lines of interpreted code Octave while DynIBEX is made of around five thousands of compiled code C++.

6 Final Remarks

We have given a new Euler-based method for controlling sampled switched systems, and compared it with the Runge-Kutta method of [10]. The method is remarkably simple and gives already promising results. In future work, we plan to explore the use of the backward Euler method instead of the forward Euler method used here (cf: [4]). We plan also to give general sufficient conditions ensuring the convexity of the error function \( \delta_j(\cdot) \); this would allow us to get rid of the convexity tests that we perform so far numerically for each pattern.

Acknowledgement. We are grateful to Antoine Girard, Jonathan Vacher, Julien Alexandre dit Sandretto and Alexandre Chapoutot for numerous helpful discussions. This work has been partially supported by Federative Institute Farman (ENS Paris-Saclay and CNRS FR3311).

References

[1] M. Abbaszadeh & H.J. Marquez (2010): Nonlinear observer design for one-sided Lipschitz systems. In: Proceedings of the American Control Conference (ACC), IEEE, pp. 799–806, doi:10.1109/ACC.2010.5530715.

[2] Matthias Althoff (2013): Reachability Analysis of Nonlinear Systems Using Conservative Polynomialization and Non-convex Sets. In: Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control, HSCC ‘13, ACM, New York, NY, USA, pp. 173–182, doi:10.1145/2461328.2461358.

[3] A Giovanni Beccuti, Georgios Papafotiou & Manfred Morari (2005): Optimal control of the boost dc-dc converter. In: Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC’05. 44th IEEE Conference on, IEEE, pp. 4457–4462, doi:10.1109/CDC.2005.1582864.
Control synthesis of switched systems using Euler method

[4] W.-J. Beyn & J. Rieger (1998): The implicit Euler scheme for one-sided Lipschitz differential inclusions. *Discr. and Cont. Dynamical Systems* B(14), pp. 409–428, doi:10.3934/dcdsb.2010.14.409

[5] Olivier Bouissou, Alexandre Chapoutot & Adel Djoudi (2013): Enclosing Temporal Evolution of Dynamical Systems Using Numerical Methods. In: NASA Formal Methods, LNCS 7871, Springer, pp. 108–123, doi:10.1007/978-3-642-38088-4_8

[6] Olivier Bouissou, Samuel Mimram & Alexandre Chapoutot (2012): HySon: Set-Based Simulation of Hybrid Systems. In: Rapid System Prototyping, IEEE, doi:10.1109/RSP.2012.6380694

[7] Xiushan Cai, Zhenyun Wang & Leipo Liu (2015): Control Design for One-side Lipschitz Nonlinear Differential Inclusion Systems with Time-delay. *Neurocomput.* 165(C), pp. 182–189, doi:10.1016/j.neucom.2015.03.008

[8] Xin Chen, Erika Abraham & Sriram Sankaranarayanan (2012): Taylor Model Flowpipe Construction for Non-linear Hybrid Systems. In: *IEEE 33rd Real-Time Systems Symposium*, IEEE Computer Society, pp. 183–192, doi:10.10116/j.neucom.2015.03.008

[9] Xin Chen, Erika Ábrahám & Sriram Sankaranarayanan (2013): Flow*: An analyzer for non-linear hybrid systems. In: *Computer Aided Verification*, Springer, pp. 258–263, doi:10.1007/978-3-642-39799-8_18

[10] A. Le Coënt, J. Alexandre dit Sandretto, A. Chapoutot & L. Fribourg (2016): Control of nonlinear switched systems based on validated simulation. In: 2016 International Workshop on Symbolic and Numerical Methods for Reachability Analysis (SNR), pp. 1–6, doi:10.1007/978-3-642-38088-4_8

[11] Jerry Ding, Eugene Li, Haomiao Huang & Claire J Tomlin (2011): Reachability-based synthesis of feedback policies for motion planning under bounded disturbances. In: *Robotics and Automation (ICRA)*, 2011 IEEE International Conference on, IEEE, pp. 2160–2165, doi:10.1109/ICRA.2011.5980268

[12] Tzanko Donchev & Elza Farkhi (1998): Stability and Euler Approximation of One-sided Lipschitz Differential Inclusions. *SIAM J. Control Optim.* 36(2), pp. 780–796, doi:10.1137/S0363012995293694

[13] Laurent Fribourg, Ulrich Kühne & Romain Soulat (2014): Finite controlled invariants for sampled switched systems. *Formal Methods in System Design* 45(3), pp. 303–329, doi:10.1007/s10703-014-0211-2

[14] Antoine Girard (2005): Reachability of uncertain linear systems using zonotopes. In: *Hybrid Systems: Computation and Control*, Springer, pp. 291–305, doi:10.1007/978-3-540-31954-2_19

[15] Antoine Girard, Giordano Pola & Paulo Tabuada (2010): Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE Transactions on Automatic Control* 55(1), pp. 116–126, doi:10.1109/TAC.2009.2032329

[16] Ian A Hiskens (2001): Stability of limit cycles in hybrid systems. In: *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, IEEE, doi:10.1109/HICSS.2001.926280

[17] Fabian Immler (2015): Verified reachability analysis of continuous systems. In: *Tools and Algorithms for the Construction and Analysis of Systems*, Springer, pp. 37–51, doi:10.1007/978-3-662-46681-0_3

[18] Frank Lempio (1995): *Set-Valued Interpolation, Differential Inclusions, and Sensitivity in Optimization*. In: Recent Developments in Well-Posed Variational Problems, Kluwer Academic Publishers, pp. 137–169, doi:10.1007/978-94-015-8472-8_6

[19] Kyoko Makino & Martin Berz (2009): Rigorous Integration of Flows and ODEs Using Taylor Models. In: *Proceedings of the 2009 Conference on Symbolic Numeric Computation*, SNC ’09, ACM, New York, USA, pp. 79–84, doi:10.1145/1577190.1577206

[20] Pierre-Jean Meyer (2015): Invariance and symbolic control of cooperative systems for temperature regulation in intelligent buildings. Thèse, Université Grenoble Alpes.

[21] Ramon Moore (1966): *Interval Analysis*. Prentice Hall.
[23] Nedialko S. Nedialkov, K. Jackson & Georges Corliss (1999): Validated solutions of initial value problems for ordinary differential equations. Appl. Math. and Comp. 105(1), pp. 21 – 68, doi:10.1016/S0096-3003(98)10083-8

[24] J. Alexandre dit Sandretto & A. Chapoutot (2015): Validated Solution of Initial Value Problem for Ordinary Differential Equations based on Explicit and Implicit Runge-Kutta Schemes. Research Report, ENSTA ParisTech.

[25] Julien Alexandre dit Sandretto & Alexandre Chapoutot (2015): Dynibex library. Http://perso.ensta-paristech.fr/ chapoutot/dynibex/.

[26] Julien Alexandre dit Sandretto & Alexandre Chapoutot (2016): Validated explicit and implicit runge-kutta methods. Reliable Computing 22, pp. 79–103.

[27] J. Stolfi & L. H. de Figueiredo (1997): Self-Validated Numerical Methods and Applications. Brazilian Mathematics Colloquium monographs, IMPA/CNPq.