Journal of Automated Reasoning manuscript No.
(will be inserted by the editor)

Verifying safety and persistence in hybrid systems using flowpipes and continuous invariants.

Andrew Sogokon · Paul B. Jackson ·
Taylor T. Johnson

Received: date / Accepted: date

Abstract We describe a method for verifying the temporal property of persistence in non-linear hybrid systems. Given some system and an initial set of states, the method establishes that system trajectories always eventually evolve into some specified target subset of the states of one of the discrete modes of the system, and always remain within this target region. The method also computes a time-bound within which the target region is always reached. The approach combines flowpipe computation with deductive reasoning about invariants and is more general than each technique alone. We illustrate the method with a case study showing that potentially destructive stick-slip oscillations of an oil-well drill eventually die away for a certain choice of drill control parameters. The case study demonstrates how just using flowpipes or just reasoning about invariants alone can be insufficient and shows the richness of systems that one can handle with the proposed method, since the systems features modes with non-polynomial ODEs. We also propose an alternative method for proving persistence that relies solely on flowpipe computation.

Keywords persistence verification · safety verification · ordinary differential equations · hybrid systems · metric temporal logic · flowpipes · positively invariant sets

This material is based upon work supported by the UK Engineering and Physical Sciences Research Council under grants EPSRC EP/I010335/1 and EP/J001058/1, the National Science Foundation (NSF) under grant numbers CNS 1464311 and CCF 1527398, the Air Force Research Laboratory (AFRL) through contract number FA8750-15-1-0105, and the Air Force Office of Scientific Research (AFOSR) under contract number FA9550-15-1-0258.

A. Sogokon
GHC 9116, 5000 Forbes Ave., Pittsburgh, PA 15213, United States of America
E-mail: asogokon@cs.cmu.edu

P.B. Jackson
2.12 Informatics Forum, 10 Crichton Street, Edinburgh EH8 9AB, United Kingdom
E-mail: Paul.Jackson@ed.ac.uk

T.T. Johnson
Room 300, 1025 16th Ave. S., Nashville, TN 37212, United States of America
E-mail: taylor.johnson@vanderbilt.edu

1 Introduction

Hybrid systems combine discrete and continuous behaviour and provide a very general framework for modelling and analysing the behaviour of systems such as those implemented in modern embedded control software. Although a number of tools and methods have been developed for verifying properties of hybrid systems, most are geared towards proving bounded time safety properties, often employing set reachability computations based on constructing over-approximating enclosures of the reachable states of ordinary differential equations (e.g. [8,21,17,34]). Methods capable of proving unbounded time safety properties often rely (explicitly or otherwise) on constructing *continuous invariants* (e.g. [57,38], and referred to in short as *invariants*); these invariants may be thought of as a generalization of *positively invariant sets* (see e.g. [6]) and are analogous to inductive invariants used in computer science to reason about the correctness of discrete programs using Hoare logic.

We argue that a combined approach employing bounded time reachability analysis and reasoning about invariants can be effective in proving *persistence* and *safety* properties in non-polynomial (non-linear) hybrid systems. We illustrate the combined approach using a detailed case study with non-polynomial ODEs for which neither approach individually was sufficient to establish the desired safety and persistence properties.

Methods for bounded time safety verification cannot in general be applied to prove safety for all time and their accuracy tends to degrade for large time bounds, especially for non-linear systems. Verification using invariants, while a powerful technique that can prove strong properties about non-linear systems, relies on the ability to find invariants that are sufficient for proving the unbounded time safety property. In practice, many invariants for the system can be found which fall short of this requirement, often for the simple reason that they do not include all the initial states of the system. We show how a combined approach employing both verification methods can, in some cases, address these limitations.

Overview

We (**I**) show that bounded time safety verification based on flowpipe construction can be naturally combined with invariants to verify persistence and unbounded time safety properties, addressing some of the limitations of each verification method when considered in isolation. (**II**) To illustrate the approach, we consider a simplified torsional model of a conventional oil well drill string that has been the subject of numerous studies by Navarro-López et al. [49]. (**III**) We illustrate an alternative approach to proving persistence properties which does not require directly reasoning about invariants and only requires flowpipe computation. (**IV**) We discuss some of the challenges that currently stand in the way of fully automatic verification using this approach. Additionally, we provide a readable overview of the methods employed in the verification process and the obstacles that present themselves when these methods are applied in practice.

2 Safety and Persistence for Hybrid Automata

2.1 Preliminaries

A number of formalisms exist for specifying hybrid systems. The most popular framework at present is that of hybrid automata [2,29], which are essentially discrete transition systems in which each discrete state represents an operating mode inside which the system evolves continuously according to an ODE under some evolution constraint. Additionally, transition guards and reset maps are used to specify the discrete transition behaviour (i.e. switching) between the operating modes. A sketch of the syntax and semantics of hybrid automata is as follows.

Definition 1 (Hybrid automaton [39]) Formally, a hybrid automaton is given by $(Q, \mathit{Var}, f, \mathit{Init}, \mathit{Inv}, T, G, R)$, where

- $Q = \{q_0, q_1, \ldots, q_k\}$ is a *finite* set of discrete states (modes),
- $\mathit{Var} = \{x_1, x_2, \ldots, x_n\}$ is a *finite* set of continuous variables,
- $f : Q \times \mathbb{R}^n \to \mathbb{R}^n$ gives the vector field defining continuous evolution in each mode,
- $\mathit{Init} \subset Q \times \mathbb{R}^n$ is the set of initial states,
- $\mathit{Inv} : Q \to 2^{\mathbb{R}^n}$ gives the *mode invariants* constraining evolution in discrete states,
- $T \subseteq Q \times Q$ is the transition relation,
- $G : T \to 2^{\mathbb{R}^n}$ gives the guard conditions for enabling transitions,
- $R : T \times \mathbb{R}^n \to 2^{\mathbb{R}^n}$ gives the reset map.

A *hybrid state* of the automaton is of the form $(q, \boldsymbol{x}) \in Q \times \mathbb{R}^n$. A *hybrid time trajectory* is a sequence (which may be finite or infinite) of intervals $\tau = \{I_i\}_{i=0}^{N}$, for which $I_i = [\tau_i, \tau_i']$ for all $i < N$ and $\tau_i \leq \tau_i' = \tau_{i+1}$ for all i. If the sequence is finite, then either $I_N = [\tau_N, \tau_N']$ or $I_N = [\tau_N, \tau_N')$. Intuitively, one may think of τ_i as the times at which discrete transitions occur. An *execution* (or a *run* or *trajectory*) of a hybrid automaton defined to be $(\tau, q, \boldsymbol{x})$, where τ is a hybrid time trajectory, $q : \langle\tau\rangle \to Q$ (where $\langle\tau\rangle$ is defined to be the set $\{0, 1, \ldots, N\}$ if τ is finite and $\{0, 1, \ldots\}$ otherwise) and $\boldsymbol{x} = \{x^i : i \in \langle\tau\rangle\}$ is a collection of differentiable functions $x^i : I_i \to \mathbb{R}^n$ such that $(q(0), x^0(0)) \in \mathit{Init}$, for all $t \in [\tau_i, \tau_i')$, $\dot{x}^i(t) = f(q(i), x^i(t))$ and $x^i(t) \in \mathit{Inv}(q(i))$. For all $i \in \langle\tau\rangle \setminus \{N\}$ it is also required that transitions respect the guards and reset maps, i.e. $e = (q(i), q(i+1)) \in T$, $x^i(\tau_i') \in G(e)$ and $x^{i+1}(\tau_{i+1}) \in R(e, x^i(\tau_i'))$.

We consider MTL[1] formulas satisfied by trajectories. The satisfaction relation is of form $\rho \models^p \phi$, read as "*trajectory ρ at position p satisfies temporal logic formula ϕ*", where a position p on a trajectory is identified by a pair of form (i_p, t_p) where $i_p \leq N$ and time $t_p \in I_{i_p}$. We use the MTL modality $\Box_I \phi$ which states that formula ϕ always holds in time interval I in the future. Formally, this can be defined as $\rho \models^p \Box_I \phi \equiv \forall p' \geq p \,\text{s.t.}\, (t_{p'} - t_p) \in I.\, \rho \models^{p'} \phi$, where $p' \geq p \equiv i_{p'} > i_p \vee (i_{p'} = i_p \wedge t_{p'} \geq t_p)$. Similarly we can define the modality $\Diamond_I \phi$ which states that formula ϕ eventually holds at some time in the time interval I in the future. An MTL formula is valid for a given hybrid automaton if it is satisfied by all trajectories of that automaton starting at position $(0, 0)$. For clarity when writing MTL formulas, we assume trajectories are not restricted to start in *Init* states and instead introduce *Init* predicates into the formulas when we want restrictions.

Alternative formalisms for hybrid systems, such as *hybrid programs* [56], enjoy the property of having a compositional semantics and can be used to verify properties of systems by verifying properties of their parts in a theorem prover [58,22]. Other formal modelling frameworks for hybrid systems, such as *Hybrid CSP* [37], have also found application in theorem provers [77,79].

[1] Metric Temporal Logic; see e.g. [35].

2.2 Bounded Time Safety and Eventuality

The *bounded time safety verification problem* (with some finite time bound $t > 0$) is concerned with establishing that given an initial set of states $\text{Init} \subseteq Q \times \mathbb{R}^n$ and a set of safe states $\text{Safe} \subseteq Q \times \mathbb{R}^n$, the state of the system may not leave Safe within time t along any valid trajectory τ of the system. In the absence of closed-form solutions to the ODEs, this property may be established by verified integration, i.e. by computing successive over-approximating enclosures (known as *flowpipes*) of the reachable states in discrete time steps. Bounded time reachability analysis can be extended to full hybrid systems by also computing/over-approximating the discrete reachable states (up to some finite bound on the number of discrete transitions).

A number of bounded time verification tools for hybrid systems have been developed based on verified integration using interval enclosures. For instance, *iSAT-ODE*, a verification tool for hybrid systems developed by Eggers et al. [17] relies on the verified integration tool *VNODE-LP* by Nedialkov [52] for computing the enclosures. Other examples include *dReach*, a reachability analysis tool for hybrid systems developed by Kong et al. [34], which uses the *CAPD* library [32]. Over-approximating enclosures can in practice be very precise for small time horizons, but tend to become conservative when the time bound is large (due to the so-called *wrapping effect*, which is a problem caused by the successive build-up of over-approximation errors that arises in interval-based methods; see e.g. [53]). An alternative verified integration method using *Taylor models* was introduced by Makino and Berz (see [5, 53]) and can address some of these drawbacks, often providing tighter enclosures of the reachable set. Implementations of the method have been reported in *COSY INFINITY*, a scientific computing tool by Makino and Berz [43]; *VSPODE*, a tool for computing validated solutions to parametric ODEs by Lin and Stadtherr [36]; and in *Flow**, a bounded time verification for hybrid systems developed by Chen et al. [8].

Because flowpipes provide an over-approximation of the reachable states at a given time, verified integration using flowpipes can also be used to reason about *liveness* properties such as *eventuality*, i.e. when a system is guaranteed to eventually enter some *target set* having started off at some point in an initial set. The bounded time safety and eventuality properties may be more concisely expressed by using MTL notation, i.e. by writing $\text{Init} \to \Box_{[0,t]}\, \text{Safe}$, and $\text{Init} \to \Diamond_{[0,t]}\, \text{Target}$, where Init describes the initial set of states, $\text{Safe} \subseteq Q \times \mathbb{R}^n$ is the set of safe states and $\text{Target} \subseteq Q \times \mathbb{R}^n$ is the target region which is to be eventually attained.

Remark 1 The bounded time eventuality properties we will consider are more restrictive than the general (unbounded time) case. For instance, consider a continuous 2-dimensional system governed by $\dot{x}_1 = x_2, \dot{x}_2 = 0$ and confined to evolve in the region where $x_2 > 0$. If one starts this system inside a state where $x_1 = 0$, it will eventually evolve into a state where $x_1 = 1$ by following the solution, however one may not put a finite bound on the time for this to happen. Thus, while $x_1 = 0 \to \Diamond_{[0,\infty)}\, x_1 = 1$ is true for this system the bounded time eventuality property $x_1 = 0 \to \Diamond_{[0,t]}\, x_1 = 1$, will not hold for any finite $t > 0$.

2.3 Unbounded Time Safety

A safety property for unbounded time may be more concisely expressed using an MTL formula:

$$\text{Init} \to \Box_{[0,\infty)}\, \text{Safe}.$$

A proof of such a safety assertion is most commonly achieved by finding an appropriate *invariant*, $I \subseteq Q \times \mathbb{R}^n$, which contains no unsafe states (i.e. $I \subseteq \text{Safe}$) and such that the state of the system may not escape from I into an unsafe state along any valid trajectory of the system. Invariance is a special kind of safety assertion and may be written as I $\rightarrow$ $\Box_{[0,\infty)}$ I. A number of techniques have been developed for proving invariance properties for continuous systems without the need to compute solutions to the ODEs [63,56,74,38, 24,68].

2.4 Combining Unbounded Time Safety with Eventuality to Prove Persistence

In linear temporal logic, a *persistence* property [44] states that a formula is ‘eventually always’ true. For instance, using persistence one may express the property that a system starting in any initial state always eventually reaches some target set and then always stays within this set. Using MTL notation, we can write this as:

$$\text{Init} \rightarrow \Diamond_{[0,\infty)} \Box_{[0,\infty)} \text{Target}.$$

Persistence properties generalize the concept of stability. With stability one is concerned with showing that the state of a system always converges to some particular equilibrium point. With persistence, one only requires that the system state eventually becomes always trapped within some set of states.

Here we are concerned with a slightly stronger form of persistence, where one ensures that the target set is always reached within some specified time t:

$$\text{Init} \rightarrow \Diamond_{[0,t]} \Box_{[0,\infty)} \text{Target}.$$

We observe that a way of proving this is to find a set $I \subseteq \text{Target}$ such that:

1. $\text{Init} \rightarrow \Diamond_{[0,t]} I$ holds, and
2. I is an invariant for the system.

This fact can be stated more formally as a rule of inference:

$$\text{(Persistence)}\ \frac{\text{Init} \rightarrow \Diamond_{[0,t]} I \qquad I \rightarrow \Box_{[0,\infty)} I \qquad I \rightarrow \text{Target}}{\text{Init} \rightarrow \Diamond_{[0,t]} \Box_{[0,\infty)} \text{Target}}\ .$$

Previous Sections 2.2 and 2.3 respectively surveyed how the eventuality premise $\text{Init} \rightarrow \Diamond_{[0,t]} I$ and invariant premise $I \rightarrow \Box_{[0,\infty)} I$ can be established by a variety of automated techniques. In Section 5 we explore automation challenges further and remark on ongoing work addressing how to automatically generate suitable invariants I.

2.5 Using Persistence to Prove Safety

Finding appropriate invariants to prove unbounded time safety as explained above in Section 2.3 can in practice be very difficult. It might be the case that invariants $I \subseteq \text{Safe}$ for the system can be found, but also ensuring that $\text{Init} \subseteq I$ is infeasible. Nevertheless it might be the case that one of these invariants I is always eventually reached by trajectories starting in Init and all those trajectories are contained within Safe. In such cases, Safe is indeed a safety property of the system when starting from any point in Init. More precisely, if one can find an invariant I as explained above in Section 2.4 to show the persistence property: $\mathrm{Init} \to \Diamond_{[0,t]} \Box_{[0,\infty)} \mathrm{Safe}$, and further one can show for the same time bound t that: $\mathrm{Init} \to \Box_{[0,t]} \mathrm{Safe}$, then one has: $\mathrm{Init} \to \Box_{[0,\infty)} \mathrm{Safe}$. As a result, one may potentially utilize invariants I that were by themselves insufficient for proving the safety property. This approach to safety verification can be summarized in the following rule:

$$(\mathrm{Safety})\ \frac{\mathrm{Init} \to \Box_{[0,t]} \mathrm{Safe} \qquad \mathrm{Init} \to \Diamond_{[0,t]} I \qquad I \to \Box_{[0,\infty)} I \qquad I \to \mathrm{Safe}}{\mathrm{Init} \to \Box_{[0,\infty)} \mathrm{Safe}} .$$

Remark 2 The problem of showing that a state satisfying $\Box_{[0,\infty)} \mathrm{Safe}$ is reached in finite time t, while ensuring that the formula $\Box_{[0,t]} \mathrm{Safe}$ also holds (i.e. states satisfying $\neg\mathrm{Safe}$ are avoided up to time t) is sometimes called a *reach-avoid problem* [78].

Even if one's goal is to establish bounded (rather than unbounded) time safety properties, this inference scheme could still be of use, as it could significantly reduce the time bound t needed for bounded time reachability analysis. In practice, successive over-approximation of the reachable states using flowpipes tends to become conservative for large values of t. In highly non-linear systems one can realistically expect to compute flowpipes only for very modest time bounds (e.g. in chaotic systems flowpipes are guaranteed to 'blow up', but invariants may still sometimes be found). Instead, it may in some cases be possible to prove the safety property by computing flowpipes up to some small time bound, after which the system can be shown to be inside an invariant that implies the safety property for all times thereafter.

3 An example persistence verification problem

Stick-slip oscillations are commonly encountered in mechanical engineering in the context of modelling the effects of dynamic friction. Informally, the phenomenon manifests itself in the system becoming "stuck" and "unstuck" repeatedly, which results in unsteady "jerky" motions. In engineering practice, stick-slip oscillations can often degrade performance and cause failures when operating expensive machinery [51]. Although the problem of demonstrating absence of stick-slip oscillations in a system is primarily motivated by safety considerations, it would be misleading to call this a *safety verification problem*. Instead, the problem may broadly be described as that of demonstrating that the system (in finite time) enters a state in which no stick-slip motion is possible and remains there indefinitely. Using MTL one may write:

$$\mathrm{Init} \to \Diamond_{[0,t]} \Box_{[0,\infty)} \mathrm{Steady},$$

where Steady describes the states in which harmful oscillations cannot occur. The formula may informally be read as saying that "from any initial configuration, the system will eventually evolve within time t into a state region where it is always steady".

As an example of a system in which eventual absence of stick-slip oscillations is important, we consider a model of a simplified conventional oil well drill string (due to Navarro-López et al. [49]). The system can be characterized in terms of the following variables: φ_r, the angular displacement of the top rotary system; φ_b, the angular displacement of the drilling bit; $\dot{\varphi}_r$, the angular velocity of the top rotary system; and $\dot{\varphi}_b$, the angular velocity of the drilling bit. The continuous state of the system $\boldsymbol{x}(t) \in \mathbb{R}^3$ can be described in terms of these variables, i.e. $\boldsymbol{x}(t) = (\dot{\varphi}_r,\ \varphi_r - \varphi_b,\ \dot{\varphi}_b)^T$. The system has two control parameters:

W_{ob} giving the weight applied on the drilling bit, and $u = T_m$ giving the surface motor torque. The dynamics is governed a non-linear system of ODEs $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$, given by:

$$\dot{x}_1 = \frac{1}{J_r}\Big(- (c_t + c_r)x_1 - k_t x_2 + c_t x_3 + u\Big), \tag{1}$$

$$\dot{x}_2 = x_1 - x_3, \tag{2}$$

$$\dot{x}_3 = \frac{1}{J_b}\Big(c_t x_1 + k_t x_2 - (c_t + c_b)x_3 - T_{f_b}(x_3)\Big). \tag{3}$$

The term $T_{f_b}(x_3)$ denotes the friction modelling the bit-rock contact and is responsible for the non-polynomial non-linearity. It is given by

$$W_{ob}R_b\Big(\mu_{c_b} + (\mu_{s_b} - \mu_{c_b})e^{-\frac{\gamma_b}{\nu_f}|x_3|}\Big)\mathrm{sgn}(x_3),$$

where $\mathrm{sgn}(x_3) = \frac{x_3}{|x_3|}$ if $x_3 \neq 0$ and $\mathrm{sgn}(x_3) \in [-1, 1]$ if $x_3 = 0$. Constants used in the model [49] are as follows: $c_b = 50\,\mathrm{Nms/rad}$, $k_t = 861.5336\,\mathrm{Nm/rad}$, $J_r = 2212\,\mathrm{kg\,m}^2$, $J_b = 471.9698\,\mathrm{kg\,m}^2$, $R_b = 0.155575\,\mathrm{m}$, $c_t = 172.3067\,\mathrm{Nms/rad}$, $c_r = 425\,\mathrm{Nms/rad}$, $\mu_{cb} = 0.5$, $\mu_{sb} = 0.8$, $\gamma_b = 0.9$, $\nu_f = 1\,\mathrm{rad/s}$. Even though at first glance the system looks like a plain continuous system with a single set of differential equations, it is effectively a hybrid system with at least 3 modes, where the drilling bit is: “rotating forward” ($x_3 > 0$), “stopped” ($x_3 = 0$), and “rotating backward” ($x_3 < 0$). A sub-mode of the stopped mode models when the drill bit is stuck. In this sub-mode, the torque components on the drill bit due to c_t, c_b and k_t are insufficient to overcome the static friction $W_{ob}R_b\mu_{c_b}$, and $\mathrm{sgn}(x_3)$ is further constrained so as to ensure $\dot{x}_3 = 0$.

Once the drill is in operation, so-called *stick-slip oscillations* can cause damage when the bit repeatedly becomes stuck and unstuck due to friction in the bottom hole assembly. In the model this behaviour would correspond to the system entering a state where $x_3 = 0$ repeatedly. The objective is to verify the eventual absence of stick-slip oscillations in the system initialized at the origin (i.e. at rest) for some given choice of the control parameters W_{ob} and u. Previous work by Navarro-López and Carter [49] explored modelling the simplified model of the drill as a hybrid automaton and simulated the resulting models in Stateflow and Modelica.

Simulations, such as those obtained in [49], using different models and control parameters for the drill can suggest stick-slip oscillations or their absence (illustrated in Fig. 1) in a particular model, however the task of verifying their eventual absence cannot be adequately addressed with simulation alone. In practice however, simulation is incredibly useful in providing some degree of confidence in the overall result, which is very important to know before attempting verification.

A simulation of the system with a concrete choice for the control parameters $W_{ob} = 50,000$ N and $u = 6,000$ Nm, shown as a trajectory in the 3-dimensional state space in Fig 6a, suggests that the system does not exhibit stick-slip oscillations, because the trajectory is observed to start at the origin, escape the surface ($x_3 = 0$)[2] and stabilize around a point where the angular velocity of the drilling bit is positive ($x_3 > 0$).

4 Verifying Persistence

The property of interest, i.e. the eventual absence of stick-slip oscillation that we observe in the simulation, may be phrased as the following formula in metric temporal logic:

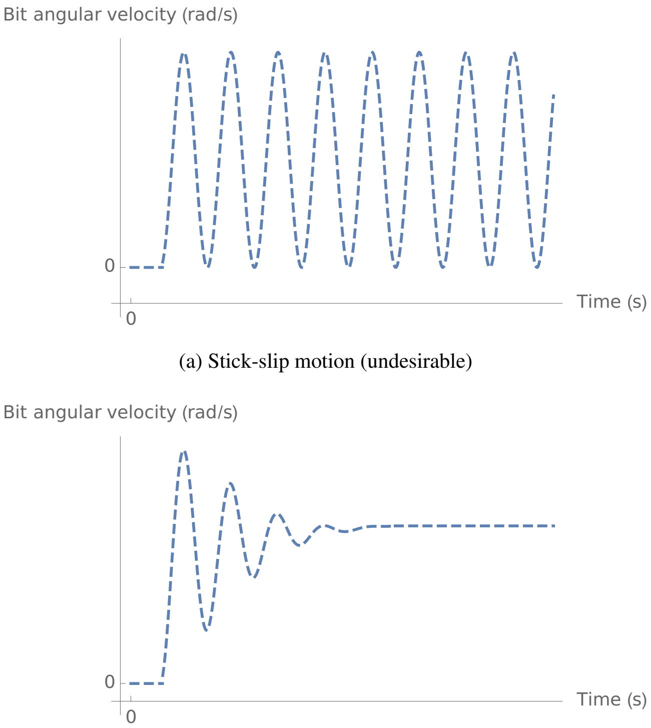
[2] The system exhibits *sliding behaviour* on a portion of this surface known as the *sliding set*. See [49].

$x_1 = 0 \wedge x_2 = 0 \wedge x_3 = 0 \rightarrow \Diamond_{[0,t]} \Box_{[0,\infty)} x_3 > 0$, which informally asserts that the system initialized at the origin will *eventually* (diamond modality) enter a state where it is *always* (box modality) the case that $x_3 > 0$. In the following sections we describe a method for proving this assertion. Following our approach, we break the problem down into the following two sub-problems:

1. Finding an appropriate invariant I in which the property $\Box_{[0,t]} x_3 > 0$ holds. For this we employ continuous/positive invariants, discussed in the next section.
2. Proving that the system reaches a state in the set I in finite time when initialized at the origin, i.e. $x_1 = 0 \wedge x_2 = 0 \wedge x_3 = 0 \rightarrow \Diamond_{[0,t]} I$.

4.1 Continuous Invariant

Finding continuous invariants that are sufficient to guarantee a given property is in practice remarkably difficult. Methods for automatic continuous invariant generation have been reported by numerous authors [63,75,27,68,67,38,80,23,65,70], but in practice often result in "coarse" invariants that cannot be used to prove the property of interest, or require an



(a) Stick-slip motion (undesirable)

(b) Stabilization (desired behaviour)

Figure 1: Simulations can exhibit stabilization with positive bit angular velocity and stick-slip bit motion.

unreasonable amount of time due to their reliance on expensive real quantifier elimination algorithms.

In order to craft a suitable invariant for our system manually, we will first consider simplified linear dynamics governing the motion in the mode where the drill bit is spinning with sufficiently-positive angular velocity x_3 that the non-linearity of the bit-rock friction $T_{f_b}(x_3)$ is negligible. The linearization is obtained by replacing the non-linear dynamic friction term T_{f_b} with $T_{c_b} = W_{ob}R_b\mu_{cb}$, the Coulomb friction torque:

$$\dot{x}_3 = \frac{1}{J_b}\Big(c_t x_1 + k_t x_2 - (c_t + c_b)x_3 - W_{ob}R_b\mu_{cb}\Big),$$

where $\dot{x}_1$ and $\dot{x}_2$ are the same as in (1) and (2), respectively. This linearization results in an affine system with an equilibrium $\boldsymbol{x}_*$ at

$$\begin{aligned} x_{*1} &= \frac{u - \mu_{cb}R_b W ob}{c_b + c_r}, \\ x_{*2} &= \frac{c_b u + c_r \mu_{cb} R_b W_{ob}}{c_b k_t + c_r k_t}, \\ x_{*3} &= \frac{u - \mu_{cb}R_b W_{ob}}{c_b + c_r}. \end{aligned}$$

By applying a simple transformation that moves the equilibrium to the origin, i.e.

$$(x_1, x_2, x_3) \mapsto (x_1 + x_{*1}, x_2 + x_{*2}, x_3 + x_{*3}),$$

one obtains the following (purely linear) system:

$$\begin{aligned} \dot{x}_1 &= \frac{1}{J_r}\Big(-(c_r + c_t)x_1 - k_t x_2 + c_t x_3\Big), \\ \dot{x}_2 &= x_1 - x_3, \\ \dot{x}_3 &= \frac{1}{J_b}\Big(c_t x_1 + k_t x_2 - (c_b + c_t)x_3\Big), \end{aligned}$$

which may be written down in matrix form as

$$\dot{\boldsymbol{x}} = \begin{pmatrix} \frac{-(c_r+c_t)}{J_r} & \frac{-k_t}{J_r} & \frac{c_t}{J_b} \\ 1 & 0 & -1 \\ \frac{c_t}{J_b} & \frac{k_t}{J_b} & \frac{-(c_b+c_t)}{J_r} \end{pmatrix} \boldsymbol{x}.$$

One popular technique for testing stability of linear systems of the form

$$\dot{\boldsymbol{x}} = A\boldsymbol{x}$$

involves solving the so-called Lyapunov equation [33]

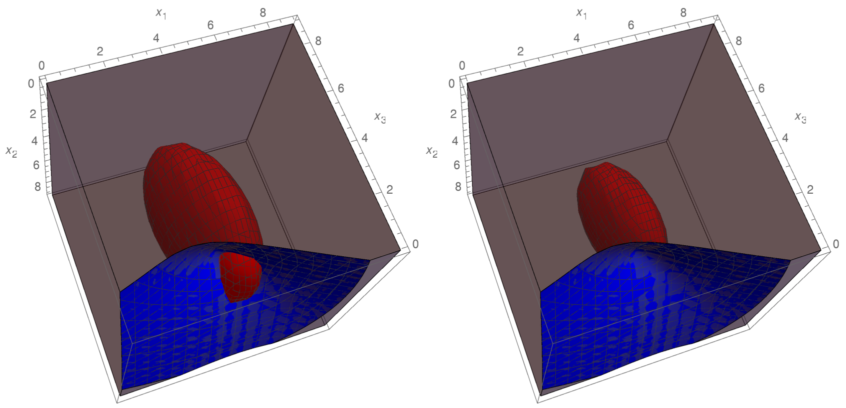
$$AX + XA^T + Q = 0.$$

One may conclude that the linear system is stable if one finds positive definite matrices X and Q satisfying the equation. In practice, one is required to choose Q to be some particular positive definite matrix and then search for the matrix X. For example, if one has the means of generating (pseudo-)random numbers, one may generate a square matrix R with random entries. From this, provided that R is invertible, one may obtain a positive definite matrix Q with random entries by setting $Q = R^T R$. One may further use the solution X to the Lyapunov equation to obtain a (necessarily quadratic) Lyapunov function for the linear system, which is given by $\boldsymbol{x}^T X\boldsymbol{x}$. This can be readily turned into a Lyapunov function for the original affine linearization with an equilibrium at $\boldsymbol{x}_*$, i.e.

$$V(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x}_*)^T X(\boldsymbol{x} - \boldsymbol{x}_*). \tag{4}$$

A concrete example of a Lyapunov function obtained by following this method is [3]

$$\begin{aligned} V(\boldsymbol{x}) = {} & 50599.6 - 14235.7x_1 + 1234.22x_1^2 - 4351.43x_2 + 342.329x_1x_2 + 288.032x_2^2 \\ & - 3865.81x_3 + 367.657x_1x_3 + 18.2594x_2x_3 + 241.37x_3^2. \end{aligned}$$



(a) $V(\boldsymbol{x}) \leq 2400$. (b) $V(\boldsymbol{x}) \leq 1400$.

Figure 2: Ellipsoidal sub-level sets $V(\boldsymbol{x}) \leq k$ (in red) and surfaces where $\dot{V} \equiv \nabla V \cdot f(\boldsymbol{x}) = 0$ (in red.) Region where $\dot{V} < 0$, i.e. the derivative of V along the solutions of the original non-linear system is negative is shaded above the blue surface increasing along axis x_3).

The sub-level sets of V are guaranteed to define positively invariant sets [6] (i.e. trapping regions from which solutions cannot escape) of the affine linearization; however, this need *not* be true of the original non-linear dynamics. A sufficient condition for ensuring that the sub-level set $V(\boldsymbol{x}) \leq k$, where $k > 0$, is positively invariant under the flow of an autonomous (perhaps non-linear) system $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ is [4]

$$\forall\, \boldsymbol{x} \in \mathbb{R}^3.\ V(\boldsymbol{x}) = k \to \nabla V \cdot f(\boldsymbol{x}) < 0.$$

In our case the system is 3-dimensional and autonomous, hence we can visualize both the surfaces defined by $V = k$ and regions where the rate of change of V along the solutions is negative , i.e. $\dot{V} \equiv \nabla V \cdot f(\boldsymbol{x}) < 0$, as 3-dimensional geometric objects (Fig. 2).

[3] The Lyapunov function in fact has rational coefficients, but their representation is too bulky to be given here exactly.

[4] Here ∇ denotes the *gradient* of V, i.e. the vector of partial derivatives $(\frac{\partial V}{\partial x_1}, \dots, \frac{\partial V}{\partial x_n})$.

By inspecting the 3-dimensional plots in Fig. 2, we see that the surface corresponding to $V(\boldsymbol{x}) = 2400$ in Fig. 2a intersects with the regions where the rate of change of V along the solutions of the system is negative (part of the red ellipsoid above the blue surface) and positive (part of the red ellipsoid visible below the blue surface). This suggests that there are solutions of the system that escape from the red ellipsoid. On the other hand, in Fig. 2b we observe that the surface $V(\boldsymbol{x}) = 1400$ appears to be entirely contained in the region above (increasing along x_3) the blue surface, suggesting that everywhere on the surface, the rate of change of the function V along the solutions of the system is negative.

Remark 3 There is a significant body of existing work devoted to the problem of estimating domains of attraction using Lyapunov functions. Forsman [20] developed a method for maximizing the invariant sub-level set of polynomial Lyapunov functions using Gröbner basis computations; this method applies only to polynomial systems of ODEs. The method of Davison and Kurak [13] seeks to maximize the hypervolume enclosed by the Lyapunov sub-level sets. Vannelli and Vidyasagar [76] reported a powerful method for domain of attraction estimation which uses rational functions as Lyapunov function candidates. More recently, Goubault et al. [25] studied the problem of generating non-polynomial Lyapunov functions and positive invariants using so-called *Darboux polynomials*.

In order to prove the continuous invariance property for $I \equiv V(\boldsymbol{x}) \leq 1400$, it is sufficient to show that the following holds:

$$\forall\, \boldsymbol{x} \in \mathbb{R}^3.\; V(\boldsymbol{x}) = 1400 \rightarrow \nabla V \cdot f(\boldsymbol{x}) < 0. \tag{5}$$

Unfortunately, in the presence of non-polynomial terms[5] a first order sentence will in general not belong to a decidable theory [66], although there has recently been progress in broadening the scope of the popular *CAD* real quantifier elimination algorithm (originally due to Collins [11])[6] to work with restricted classes of non-polynomial problems (by Strzeboński [73]).

In practice, this conjecture is easily proved in under 5 seconds using MetiTarski, an automatic theorem prover, developed by L.C. Paulson and co-workers at the University of Cambridge, designed specifically for proving universally quantified first order conjectures featuring transcendental functions, such as sin,cos, $\ln$, $\exp$, etc. The interested reader may find more details about the MetiTarski system in [1,55].

Remark 4 Although Wolfram's *Mathematica* 10 computer algebra system also provides some functionality for proving first-order conjectures featuring non-polynomial expressions using its `Reduce[]` function, we were unable (on our system[7]) to prove conjecture (5) this way after over an hour of computation, after which the Mathematica kernel crashed.

The automatic proof of conjecture (5) obtained using MetiTarski (provided we trust the system) establishes that $V(\boldsymbol{x}) \leq 1400$ defines a positively invariant set, and thus we are guaranteed that solutions initialized inside this set remain there at all future times. In order to be certain that no outgoing discrete transitions of the hybrid system are possible when the system is evolving inside $V(\boldsymbol{x}) \leq 1400$, we further require a proof of the following conjecture featuring only polynomial terms:

$$\forall\, \boldsymbol{x} \in \mathbb{R}^3.\; V(\boldsymbol{x}) \leq 1400 \rightarrow x_3 > 0. \tag{6}$$

[5] E.g. those featured in the right-hand side of the ODE, i.e. $f(\boldsymbol{x})$.

[6] The interested reader may find a superbly readable elementary introduction to the CAD algorithm in [31] and a good overview of current state of the art in [12].

[7] Intel i5-2520M CPU @ 2.50GHz, 4GB RAM, running Arch Linux kernel 4.2.5-1.

An automatic proof of this conjecture may be obtained using an implementation of a decision procedure for first-order real arithmetic. Currently, this functionality is available in e.g. *Mathematica*, *QEPCAD-B*, *Reduce/Redlog* and other packages.

Remark 5 As of 2018, there is yet no formally verified implementation of any decision procedure for first-order real arithmetic, or its universally/existentially-quantified fragment. The existing implementations of CAD and other algorithms found in modern computer algebra systems comprise thousands of lines of code into which the user has presently no choice but to put his/her trust [12]. MetiTarski is likewise exposed to potential bugs in the implementations with which it interfaces. Some of the recent efforts in verified real arithmetic are described in [10,40,45].

4.2 Verified Integration

In order to show that the system does indeed enter the positively invariant ellipsoid $V(\boldsymbol{x}) \leq 1400$ in finite time, it is not sufficient to observe this in a simulation (as in Fig. 6b), which is why we use a tool employing *verified integration* based on Taylor models. *Flow** (implemented by Chen et al. [8]) is a bounded time safety verification tool for hybrid systems that computes Taylor models to analyse continuous reachability.

Remark 6 An earlier implementation of Taylor models for verified integration of nonlinear (and even parametric) ODEs was reported by Lin and Stadtherr [36] in a tool called *VSPODE*.

The tool works by computing successive over-approximations (flowpipes) of the reachable set of the system, which are internally represented using Taylor models (but which may in turn be over-approximated by a bounding hyper-box and easily rendered).

Fig. 3 shows the bounding boxes of solution enclosures computed from the point initial condition at the origin using Flow* with adaptive time steps and Taylor models of order 13, a time bound of 12.7 and the same control parameters used in the simulation (i.e. $u = 6,000$ Nm, $W_{ob} = 50,000$ N). We observe that once solutions escape to the region where $x_3 > 0$, they maintain a positive x_3 component for the duration of the time bound.

The last flowpipe computed by Flow* for this problem can be bounded inside the hyper-rectangle $\mathrm{BoundBox}$ characterized by the formula

$$\mathrm{BoundBox} \equiv \frac{39}{10} \leq x_1 \leq 4 \wedge \frac{51}{10} \leq x_2 \leq \frac{26}{5} \wedge \frac{7}{2} \leq x_3 \leq \frac{37}{10}.$$

Once more, using a decision procedure for real arithmetic, we can check that the following sentence is true:

$$\forall\, \boldsymbol{x} \in \mathbb{R}^3.\ \mathrm{BoundBox} \to V(\boldsymbol{x}) \leq 1400.$$

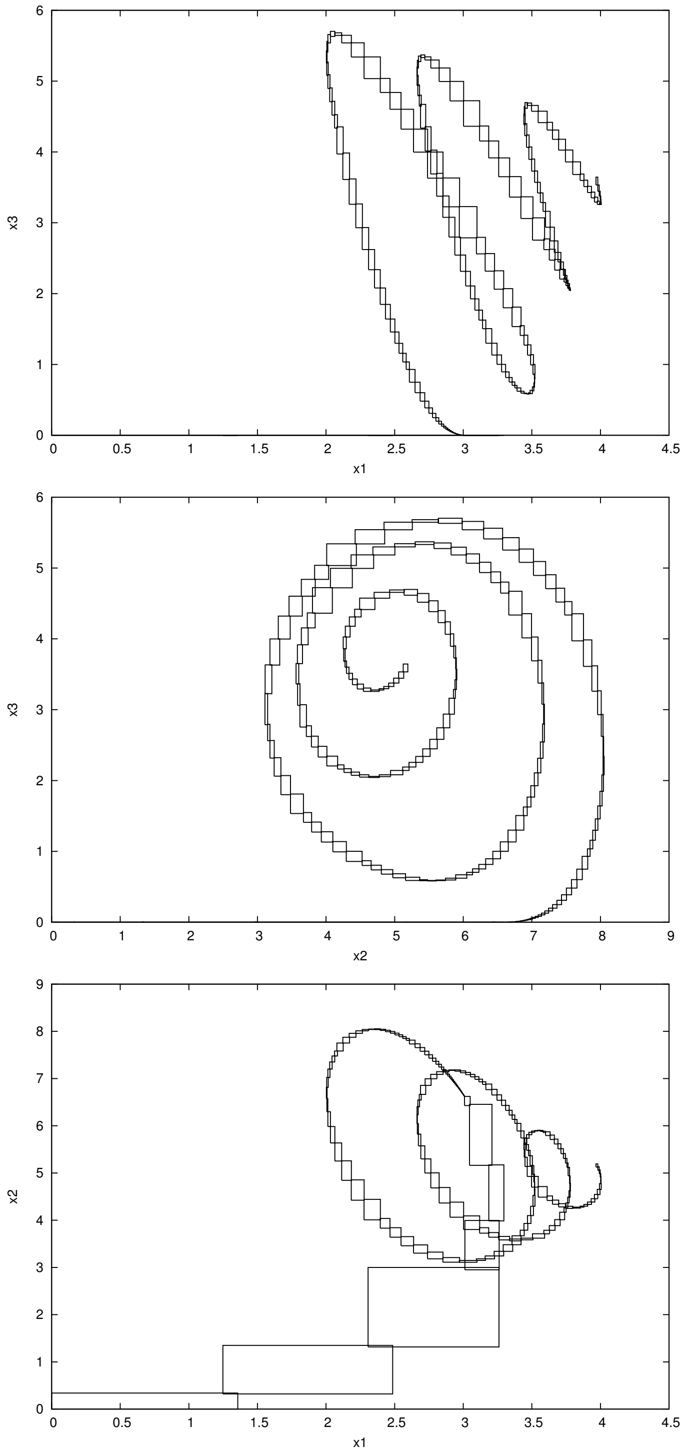


Figure 3: Verified integration up to time $t = 12.7$ from a point initial condition at the origin.

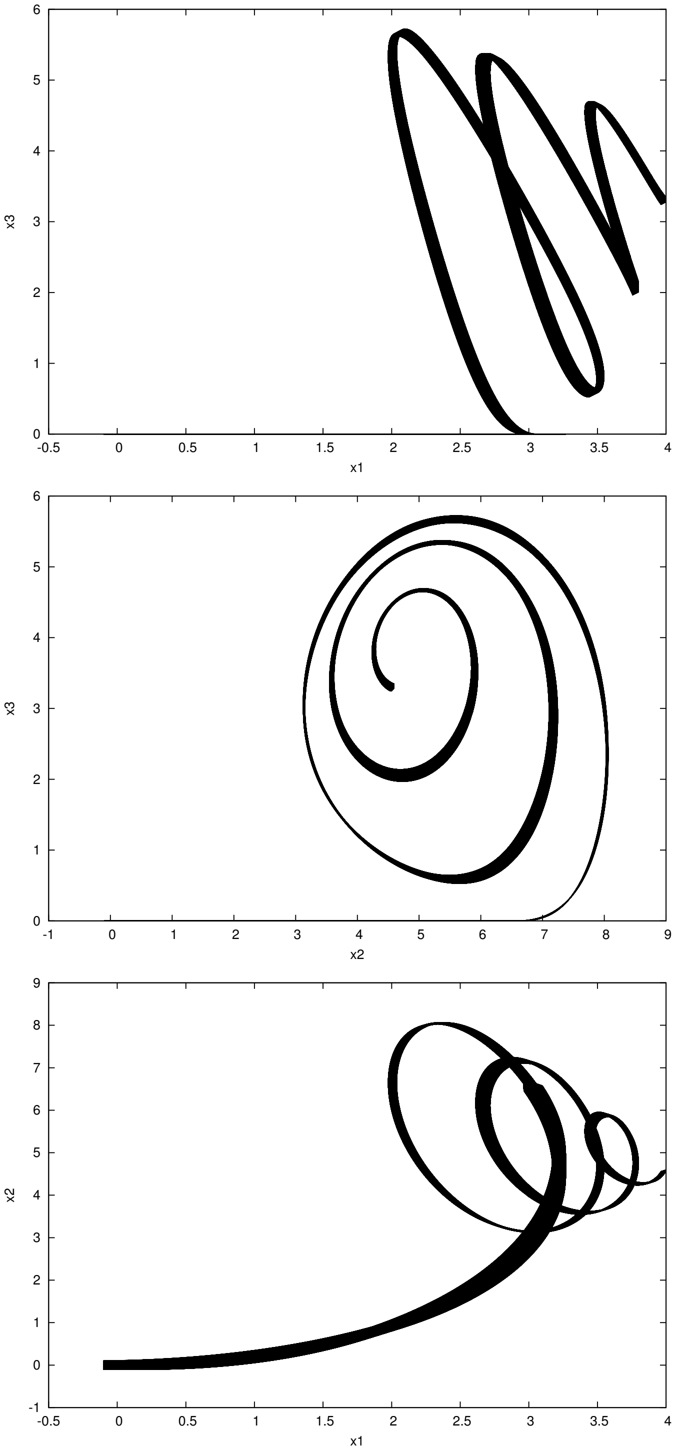


Figure 4: Verified integration up to time $t = 12.2$ from an interval initial condition.

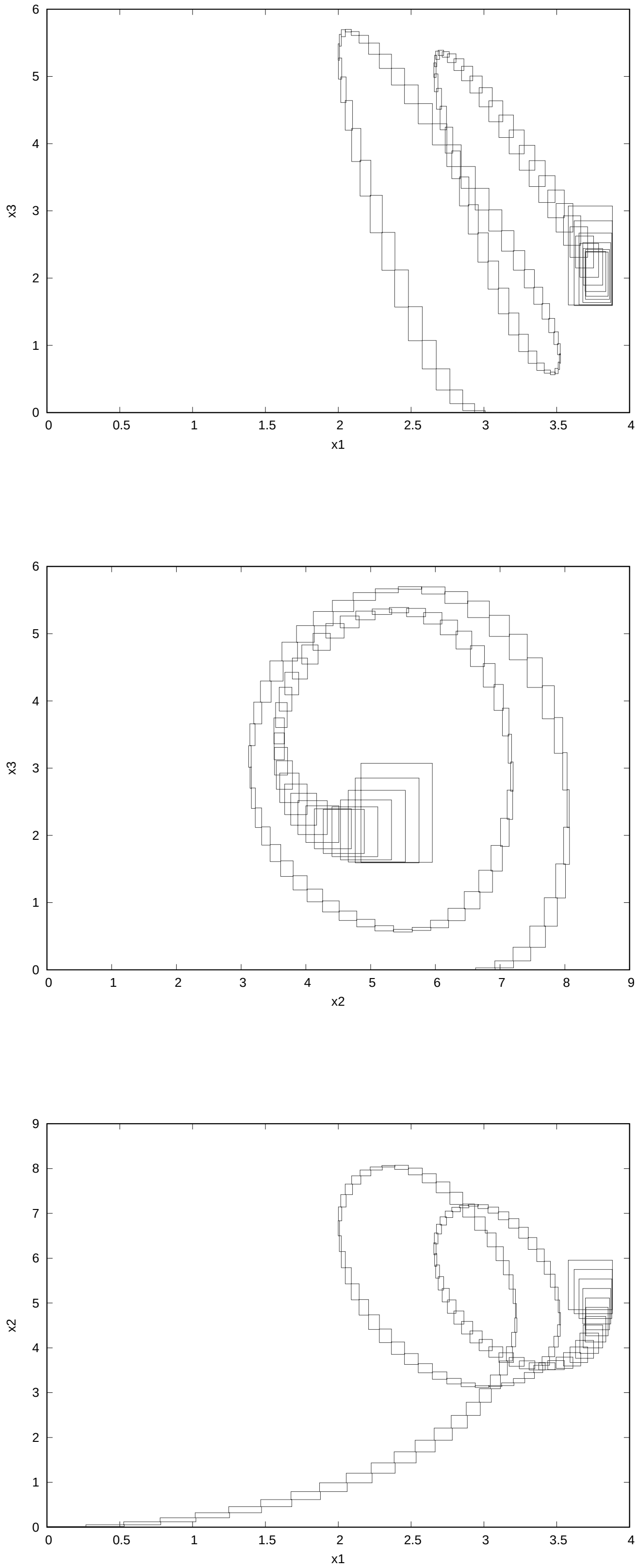


Figure 5: Verified integration with affine disturbance input from a point initial condition at the origin.

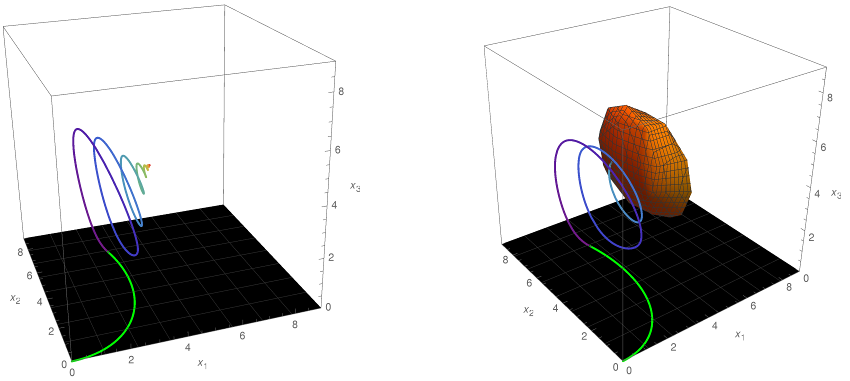
If we are able to establish the following facts:

1. $I \to \Box_{[0,\infty)}\, I$ (I is a continuous invariant),
2. $I \to \text{Steady}$ (inside I, there are no harmful oscillations), and
3. $\text{Init} \to \Diamond_{[0,t]}\, I$ (the system enters the region I in finite time),

then we can conclude that $\text{Init} \to \Diamond_{[0,t]}\, \Box_{[0,\infty)}\, \text{Steady}$ is also true and the system does not exhibit harmful stick-slip oscillations when started inside Init. By taking Init to be the origin $x_1 = 0 \wedge x_2 = 0 \wedge x_3 = 0$, I to be the positively invariant sub-level set $V(\boldsymbol{x}) \leq 1400$ and Steady to be $x_3 > 0$, we are able to conclude the temporal property:

$$x_1 = 0 \wedge x_2 = 0 \wedge x_3 = 0 \to \Diamond_{[0,t]}\, \Box_{[t,\infty)}\, x_3 > 0.$$

Verified integration using Taylor models also allows us to consider *sets* of possible initial conditions, rather than initial points. This is useful when there is uncertainty about the system's initial configuration; however, in practice this comes with a significant performance overhead for verified integration. Nevertheless, we were able to compute flowpipes for 12.2 time units from an interval initial condition $x_1 \in [-0.1, 0.1]$, $x_2 \in [-0.1, 0.1]$ and $x_3 = 0$ (since it is reasonable to assume the bit to be at rest initially). The result of this verified integration is illustrated in Fig. 4.



(a) Simulation showing stabilization with positive bit angular velocity.

(b) Simulation showing eventual entry into an ellipsoidal invariant.

Figure 6: Simulation of the hybrid system initialized at the origin with $W_{ob} = 50,000$ N and $u = 6000$ Nm. The trajectory is contained by the flowpipes shown in Fig. 3 and is observed to enter the positively invariant ellipsoid $V(\boldsymbol{x}) \leq 1400$, illustrating the persistence property of eventual absence of stick-slip oscillations.

Remark 7 Another way of increasing confidence in the robustness of verification results using flowpipes is to introduce bounded disturbance into the ODEs in the Flow* model, e.g. by adding a bounded interval $d_i \subseteq \mathbb{R}$ to each f_i in the right-hand side of the ODEs, i.e. setting $\dot{x}_i = f_i(\boldsymbol{x}) + d_i$ for each $i = 1, 2, 3$. However, this makes verified integration significantly more coarse and slow, even for relatively small disturbances. For example, in Fig. 5 one can observe the flowpipes from a point initial condition becoming more conservative and 'blowing up' (after about 9 time units) for a disturbance $d_i = [-0.000001, 0.000001]$ added to the right-hand side in the positive bit spinning mode (even as the Taylor model order is increased from 13 to 14).

5 Flowpipes for proving persistence

With the above-described invariant approach, one identifies an invariant subset I of the target region, a subset from which trajectories never exit once they have entered. As an alternative to using continuous invariants, in some cases it is possible to prove persistence properties by relying entirely on bounded time reachability analysis using flowpipes. For example, it is sufficient to find some subset of the target region $S \subseteq \mathrm{Target}$ with the property that all trajectories starting from S eventually return to S in some bounded time while never passing outside of the target region. This property of always eventually returning in bounded time is one that can be checked by flowpipe computations. Many such S will not be invariants, and we have the freedom to choose perhaps very simple representations of sets S; interval boxes, for example. Formally, this reasoning can be packaged into an alternative rule:

$$(\mathrm{Persistence}_2)\ \frac{\mathrm{Init} \to \Diamond_{[0,t]}\, S \qquad S \to \Diamond_{[\tau,\tau]}\, S \qquad S \to \Box_{[0,\tau]}\, \mathrm{Target}}{\mathrm{Init} \to \Diamond_{[0,t]}\, \Box_{[0,\infty)}\, \mathrm{Target}}\ .$$

where $t \geq 0$ and $\tau > 0$. All premises of this rule can be addressed by computing flowpipes.

Remark 8 In practice, numerical simulations can provide reasonable candidates for the time τ at which trajectories return back into S in the above rule.

Example 1 (Damped oscillator) Simple harmonic motion can be observed in a mass m (kg) suspended from an spring with stiffness constant k (Figure 7a), which obeys Hooke's law, i.e. the linear relationship $F = -kx$, where F (N) is the force required to displace the mass by x (m) from the point of equilibrium. From Newton's second law, i.e. $F = m\ddot{x}$, the motion of the mass on a spring is governed by the differential equation: $m\ddot{x} + kx = 0$, more commonly written as $\ddot{x} + \omega^2 x = 0$, where $\omega = \sqrt{\frac{k}{m}}$ is the frequency of oscillation.

(a) Harmonic oscillator.

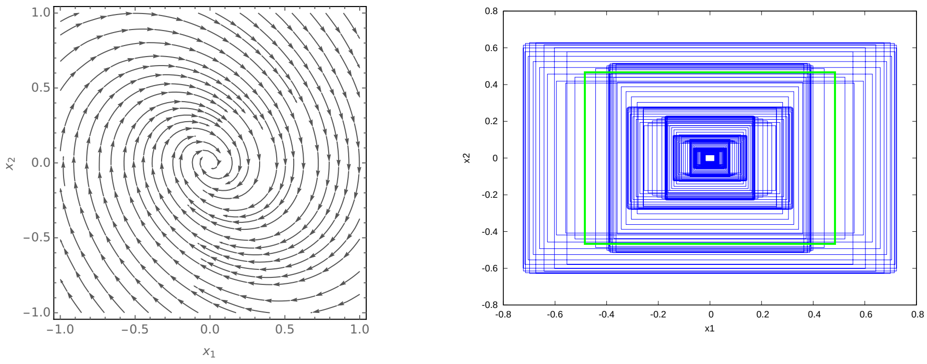
(b) Damped oscillator.

Figure 7: Oscillator model.

If one introduces damping (with viscous damping coefficient c) into the system (illustrated in Figure 7b), the differential equation becomes $\ddot{x} + 2d\omega\dot{x} + \omega^2 x = 0$, where $d = \frac{c}{2\sqrt{km}}$ is known as the *damping factor*. By setting $x_1 = x$ and $x_2 = \dot{x}$, one may write:

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -\omega^2 x_1 - 2d\omega x_2.$$

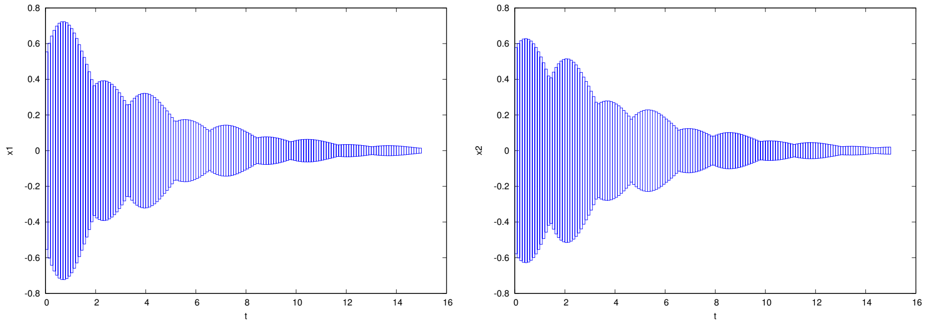
Setting $\omega = 1$ and $d = 0.25$, the phase portrait of the resulting stable system is illustrated in Fig. 8a. By selecting S to be the hyperbox $[-0.5, 0.5] \times [-0.5, 0.5]$ and computing flowpipes from S, one may observe the bounding boxes of the flowpipes first expanding but then eventually converging to inside of S. Bounding boxes at successive time steps are shown in Fig. 8b where the initial bounding box is highlighted in green. Fig. 9a and Fig. 9b show how the bounding box dimensions vary over time. This eventual convergence to within S in can be used to e.g. establish the property $S \to \Diamond_{[4,4]} S$.



(a) Phase portrait.

(b) Converging flowpipes from S.

Figure 8: Damped oscillator.



(a) Bounds on the x_1 component.

(b) Bounds on the x_2 component.

Figure 9: Bounds on flowpipes from an initial box S.

We should note that this approach is in practice highly sensitive to the amount of damping, i.e. it is often easier to show that flowpipes return into the initial set when there is a greater rate of convergence towards the stable equilibrium. This can already be observed in the damped oscillator model. For example, after decreasing the damping factor d in the model to 0.05, one may observe a much slower convergence of flowpipe enclosures, which forces one to consider larger time bounds in order to apply this technique. By further decreasing d from 0.05 to 0.005, one is no longer able to show that flowpipe enclosures return into the original set using the same Taylor model orders and time bound (e.g. 10 and 15 respectively (see Appendix, Fig. 10).

Remark 9 While in principle this alternative technique could be applied to show persistence of our drill-string example, our experiments in this application have so far have not been successful: the flowpipe computations are much more expensive fail to produce eventually convergent enclosures with the time bounds and Taylor model parameters that we tried.

6 Outlook and Challenges to Automation

Correctness of reachability analysis tools based on verified integration is soundness critical to the overall verification approach, which makes for a strong case in favour of using formally verified implementations. At present few are available, e.g. recent work by Immler [30] which presented a formally verified continuous reachability algorithm based on adaptive Runge-Kutta methods. Verified implementations of Taylor model-based reachability analysis algorithms for continuous and hybrid systems would clearly be very valuable. One alternative to over-approximating reachable sets of continuous systems using flowpipes is based on simulating the system using a finite set of sampling trajectories and employs *sensitivity analysis* to address the coverage problem. This technique was explored by Donzé and Maler in [14]. A similar approach employing *matrix measures* has more recently been studied by Maidens and Arcak [41,42], as well as Fan et al. [19,18] (the latter exhibiting particularly encouraging results).

As an alternative to using verified integration, a number of deductive methods are available for proving eventuality properties in continuous and hybrid systems (e.g. [57,72]). These approaches can be much more powerful since they allow one to work with more general classes of initial and target regions that are necessarily out of scope for methods based on verified integration (e.g. they can work with initial sets that are unbounded, disconnected, etc). Making effective use of deductive verification tools currently in existence typically requires significant input and expertise on part of the user (finding the right invariants being one of the major stumbling blocks in practice), in stark contrast to the near-complete level of automation offered by tools based on verified integration. Methods for automatic continuous invariant generation are crucial to the mechanization of the overall verification approach. Progress on this problem would be hugely enabling for non-experts and specialists alike, as it would relieve them from the task of manually constructing appropriate invariants, which often requires intuition and expertise. Work in this area is ongoing (see e.g. [57,38,70]).

A no less important problem than verification itself concerns the correct *formal modelling* of systems with discontinuous ODEs, such as the drill string model, using hybrid automata. This problem was studied in the work of Navarro-López and Carter [49]. A different approach, applicable to discontinuous ODEs that are piecewise *polynomial*, has been pursued more recently in [71].

7 Related Work

Combining elements of qualitative and quantitative reasoning[8] to study the behaviour of dynamical systems has previously been explored in the case of planar systems by Nishida et al. [54]. The idea of combining bounded time reachability analysis with qualitative analysis in the form of discrete abstraction was investigated by Clarke et al. in [9].

Similar ideas to ours have been employed by Carter [7] and Navarro-López in [50], where the concept of *deadness* is introduced and used as a way of disproving liveness properties. Intuitively, deadness is a formalization of an idea that inside certain regions the system cannot be live, i.e. some desired property may never become true as the system evolves inside a “deadness region”. These ideas were used in a case study [7, Chapter 5] also featuring the drill system studied in [49], but with a different set of control parameters and in which the verification objective was to prove the existence of a *single trajectory* for which the drill

[8] e.g numerical solution computation with “qualitative” features, such as invariance of certain regions.

eventually gets "stuck", which is sufficient to disprove the liveness (oscillation) property. In this regard, *inner approximating* (i.e. under-approximating) flowpipes of reachable sets of ODEs (such as those explored in the recent work of Goubault and Putot [26]) provide a natural tool for studying deadness questions, whereas over-approximating flowpipes that we use in our work are more appropriate for persistence, safety, and eventuality.

Region stability is similar to our notion of persistence [59], which requires all trajectories to eventually reach some region of the state space. Sound and complete proof rules for establishing region stability have been explored and automated [61], as have more efficient encodings of the proof rule that scale better in dimensionality [46]. However, all algorithms we are aware of for checking region stability require linear or simpler (timed or rectangular) ODEs [59,61,60,46,15,62]. Strong attractors are basins of attraction where every state in the state space eventually reaches a region of the state space [59]. Some algorithms do not check region stability, but actually check stronger properties such as strong attraction, that imply region stability [59]. In contrast to these works, our method checks the weaker notion of persistence for non-linear ODEs.

She and Ratschan studied methods of proving set eventuality in continuous systems under constraints using Lyapunov-like functions [64]. Duggirala and Mitra also employed Lyapunov-like function concepts to prove inevitability properties in hybrid systems [16]. Möhlmann et al. developed Stabhyil [48], which can be applied to non-linear hybrid systems and checks classical notions of Lyapunov stability, which is a strictly stronger property than persistence. In [47] Möhlmann et al. extended their work and applied similar ideas, using information about (necessarily invariant) sub-level sets of Lyapunov functions to terminate reachability analysis used for safety verification. Prabhakar and Soto have explored abstractions that enable proving stability properties without having to search for Lyapunov functions, albeit these are not currently applicable to non-linear systems [62]. In summary, in contrast to other works listed above, our approach enables proving persistence properties in conjunction with safety properties for non-linear, non-polynomial hybrid systems and does not put restrictions on the form or the type of the invariant used in conjunction with bounded time reachability analysis.

8 Conclusion

We explored a combined technique for safety and persistence verification employing continuous invariants and reachable set computation based on constructing flowpipes. The approach was illustrated on a model of a simplified oil well drill string system studied by Navarro-López et al. [49,51], where the verification objective is to prove absence of damaging stick-slip oscillations. The system was useful in highlighting many of the existing practical challenges to applying and automating the proposed verification method. Many competing approaches already exist for verifying safety in hybrid systems, but these rarely combine different methods for reachability analysis and deductive verification, which our approach combines. We demonstrate that a combination of different approaches can be more practically useful than each constituent approach taken in isolation.

Acknowledgements The authors wish to thank the anonymous reviewers for their careful reading and valuable suggestions for improving this work and extend special thanks to Dr. E.M. Navarro-López for pointing out the highly relevant work on *deadness* [50] before it appeared in print.

References

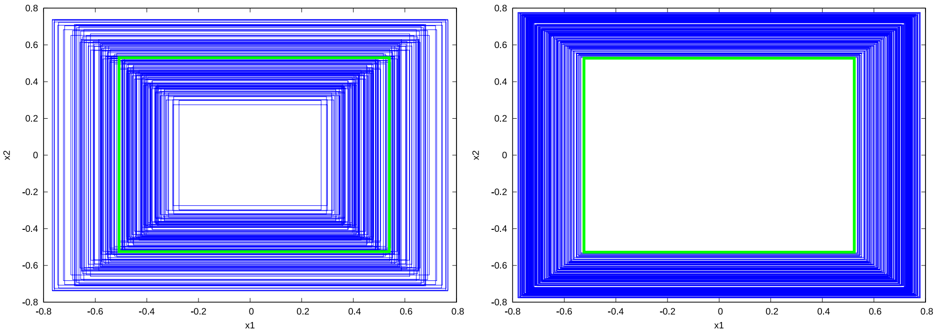
1. B. Akbarpour and L. C. Paulson. MetiTarski: An automatic theorem prover for real-valued special functions. *Journal of Automated Reasoning*, 44(3):175–205, 2010.
2. R. Alur, C. Courcoubetis, T. A. Henzinger, and P. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, editors, *Hybrid Systems*, volume 736 of *LNCS*, pages 209–229. Springer, 1992.
3. C. Baier and C. Tinelli, editors. *Tools and Algorithms for the Construction and Analysis of Systems - 21st International Conference, TACAS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume 9035 of *LNCS*. Springer, 2015.
4. A. Bemporad, A. Bicchi, and G. C. Buttazzo, editors. *Hybrid Systems: Computation and Control, 10th International Workshop, HSCC 2007, Pisa, Italy, April 3-5, 2007, Proceedings*, volume 4416 of *LNCS*. Springer, 2007.
5. M. Berz and K. Makino. Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models. *Reliable Computing*, 4(4):361–369, 1998.
6. F. Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.
7. R. A. Carter. *Verification of liveness properties on hybrid dynamical systems*. PhD thesis, University of Manchester, School of Computer Science, 2013.
8. X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In Sharygina and Veith [69], pages 258–263.
9. E. M. Clarke, A. Fehnker, Z. Han, B. H. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *International Journal of Foundations of Computer Science*, 14(4):583–604, 2003.
10. C. Cohen and A. Mahboubi. Formal proofs in real algebraic geometry: from ordered fields to quantifier elimination. *Logical Methods in Computer Science*, 8(1), 2012.
11. G. E. Collins. Hauptvortrag: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In H. Barkhage, editor, *Automata Theory and Formal Languages, 2nd GI Conference, Kaiserslautern, May 20-23, 1975*, volume 33 of *LNCS*, pages 134–183. Springer, 1975.
12. J. H. Davenport and M. England. Recent advances in real geometric reasoning. In F. Botana and P. Quaresma, editors, *Automated Deduction in Geometry - 10th International Workshop, ADG 2014, Coimbra, Portugal, July 9-11, 2014, Revised Selected Papers*, volume 9201 of *LNCS*, pages 37–52. Springer, 2014.
13. E. Davison and E. Kurak. A computational method for determining quadratic Lyapunov functions for non-linear systems. *Automatica*, 7(5):627 – 636, 1971.
14. A. Donzé and O. Maler. Systematic simulation using sensitivity analysis. In Bemporad et al. [4], pages 174–189.
15. P. S. Duggirala and S. Mitra. Abstraction refinement for stability. In *2011 IEEE/ACM International Conference on Cyber-Physical Systems, ICCPS 2011, Chicago, Illinois, USA, 12-14 April, 2011*, pages 22–31. IEEE Computer Society, 2011.
16. P. S. Duggirala and S. Mitra. Lyapunov abstractions for inevitability of hybrid systems. In T. Dang and I. M. Mitchell, editors, *Hybrid Systems: Computation and Control (part of CPS Week 2012), HSCC'12, Beijing, China, April 17-19, 2012*, pages 115–124. ACM, 2012.
17. A. Eggers, N. Ramdani, N. S. Nedialkov, and M. Fränzle. Improving the SAT modulo ODE approach to hybrid systems analysis by combining different enclosure methods. *Software and System Modeling*, 14(1):121–148, 2015.
18. C. Fan, J. Kapinski, X. Jin, and S. Mitra. Locally optimal reach set over-approximation for nonlinear systems. In *2016 International Conference on Embedded Software, EMSOFT 2016, Pittsburgh, Pennsylvania, USA, October 1-7, 2016*, pages 6:1–6:10. ACM, 2016.
19. C. Fan, J. Kapinski, X. Jin, and S. Mitra. Simulation-driven reachability using matrix measures. *ACM Transactions on Embedded Computing Systems*, 17(1):21:1–21:28, 2018.
20. K. Forsman. Construction of Lyapunov functions using Gröbner bases. In *Proceedings of the 30th IEEE Conference on Decision and Control*, volume 1, pages 798–799. IEEE, Dec. 1991.
21. G. Frehse, C. L. Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. SpaceEx: Scalable verification of hybrid systems. In G. Gopalakrishnan and S. Qadeer, editors, *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *LNCS*, pages 379–395. Springer, 2011.
22. N. Fulton, S. Mitsch, J. Quesel, M. Völp, and A. Platzer. KeYmaera X: an axiomatic tactical theorem prover for hybrid systems. In A. P. Felty and A. Middeldorp, editors, *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*, volume 9195 of *LNCS*, pages 527–538. Springer, 2015.

23. K. Ghorbal and A. Platzer. Characterizing algebraic invariants by differential radical invariants. In E. Ábrahám and K. Havelund, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8413 of *LNCS*, pages 279–294. Springer, 2014.
24. K. Ghorbal, A. Sogokon, and A. Platzer. A hierarchy of proof rules for checking positive invariance of algebraic and semi-algebraic sets. *Computer Languages, Systems & Structures*, 47:19–43, 2017.
25. E. Goubault, J. Jourdan, S. Putot, and S. Sankaranarayanan. Finding non-polynomial positive invariants and Lyapunov functions for polynomial systems through Darboux polynomials. In *American Control Conference, ACC 2014, Portland, OR, USA, June 4-6, 2014*, pages 3571–3578. IEEE, 2014.
26. E. Goubault and S. Putot. Forward inner-approximated reachability of non-linear continuous systems. In G. Frehse and S. Mitra, editors, *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control, HSCC 2017, Pittsburgh, PA, USA, April 18-20, 2017*, pages 1–10. ACM, 2017.
27. S. Gulwani and A. Tiwari. Constraint-based approach for analysis of hybrid systems. In Gupta and Malik [28], pages 190–203.
28. A. Gupta and S. Malik, editors. *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, NJ, USA, July 7-14, 2008, Proceedings*, volume 5123 of *LNCS*. Springer, 2008.
29. T. A. Henzinger. The theory of hybrid automata. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, July 27-30, 1996*, pages 278–292. IEEE Computer Society, 1996.
30. F. Immler. Verified reachability analysis of continuous systems. In Baier and Tinelli [3], pages 37–51.
31. M. Jirstrand. Cylindrical algebraic decomposition - an introduction. Technical Report 1807, Linköping University, Automatic Control, 1995.
32. T. Kapela, M. Mrozek, P. Pilarczyk, D. Wilczak, and P. Zgliczyński. CAPD - a rigorous toolbox for computer assisted proofs in dynamics. Technical report, Jagiellonian University, Krakow, Poland, 2010. Online `http://capd.ii.uj.edu.pl/`.
33. H. K. Khalil. *Nonlinear Systems*. Prentice Hall, third edition, 2002.
34. S. Kong, S. Gao, W. Chen, and E. M. Clarke. dreach: δ-reachability analysis for hybrid systems. In Baier and Tinelli [3], pages 200–205.
35. R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.
36. Y. Lin and M. A. Stadtherr. Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57(10):1145–1162, 2007.
37. J. Liu, J. Lv, Z. Quan, N. Zhan, H. Zhao, C. Zhou, and L. Zou. A calculus for hybrid CSP. In K. Ueda, editor, *Programming Languages and Systems - 8th Asian Symposium, APLAS 2010, Shanghai, China, November 28 - December 1, 2010. Proceedings*, volume 6461 of *LNCS*, pages 1–15. Springer, 2010.
38. J. Liu, N. Zhan, and H. Zhao. Computing semi-algebraic invariants for polynomial dynamical systems. In S. Chakraborty, A. Jerraya, S. K. Baruah, and S. Fischmeister, editors, *Proceedings of the 11th International Conference on Embedded Software, EMSOFT 2011, part of the Seventh Embedded Systems Week, ESWeek 2011, Taipei, Taiwan, October 9-14, 2011*, pages 97–106. ACM, 2011.
39. J. Lygeros, K. H. Johansson, S. N. Simić, J. Zhang, and S. S. Sastry. Dynamical properties of hybrid automata. *IEEE Transactions on Automatic Control*, 48(1):2–17, 2003.
40. A. Mahboubi. Programming and certifying a CAD algorithm in the Coq system. In T. Coquand, H. Lombardi, and M. Roy, editors, *Mathematics, Algorithms, Proofs, 9.-14. January 2005*, volume 05021 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.
41. J. N. Maidens and M. Arcak. Trajectory-based reachability analysis of switched nonlinear systems using matrix measures. In *53rd IEEE Conference on Decision and Control, CDC 2014, Los Angeles, CA, USA, December 15-17, 2014*, pages 6358–6364. IEEE, 2014.
42. J. N. Maidens and M. Arcak. Reachability analysis of nonlinear systems using matrix measures. *IEEE Transactions on Automatic Control*, 60(1):265–270, Jan 2015.
43. K. Makino and M. Berz. COSY INFINITY version 9. *Nuclear Instruments and Methods in Physics Research Section A*, 558(1):346–350, 2006.
44. Z. Manna and A. Pnueli. A hierarchy of temporal properties. In C. Dwork, editor, *Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing, Quebec City, Quebec, Canada, August 22-24, 1990*, pages 377–410. ACM, 1990.
45. É. Martin-Dorel and P. Roux. A reflexive tactic for polynomial positivity using numerical solvers and floating-point computations. In Y. Bertot and V. Vafeiadis, editors, *Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs, CPP 2017, Paris, France, January 16-17, 2017*, pages 90–99. ACM, 2017.

46. C. Mitrohin and A. Podelski. Composing stability proofs for hybrid systems. In U. Fahrenberg and S. Tripakis, editors, *Formal Modeling and Analysis of Timed Systems - 9th International Conference, FORMATS 2011, Aalborg, Denmark, September 21-23, 2011. Proceedings*, volume 6919 of *LNCS*, pages 286–300. Springer, 2011.
47. E. Möhlmann, W. Hagemann, and O. E. Theel. Hybrid tools for hybrid systems - proving stability and safety at once. In S. Sankaranarayanan and E. Vicario, editors, *Formal Modeling and Analysis of Timed Systems - 13th International Conference, FORMATS 2015, Madrid, Spain, September 2-4, 2015, Proceedings*, volume 9268 of *LNCS*, pages 222–239. Springer, 2015.
48. E. Möhlmann and O. E. Theel. Stabhyli: a tool for automatic stability verification of non-linear hybrid systems. In C. Belta and F. Ivančić, editors, *Proceedings of the 16th international conference on Hybrid systems: computation and control, HSCC 2013, April 8-11, 2013, Philadelphia, PA, USA*, pages 107–112. ACM, 2013.
49. E. M. Navarro-López and R. Carter. Hybrid automata: an insight into the discrete abstraction of discontinuous systems. *International Journal of Systems Science*, 42(11):1883–1898, 2011.
50. E. M. Navarro-López and R. Carter. Deadness and how to disprove liveness in hybrid dynamical systems. *Theoretical Computer Science*, 642(C):1–23, Aug. 2016.
51. E. M. Navarro-López and R. Suárez. Practical approach to modelling and controlling stick-slip oscillations in oilwell drillstrings. In *Control Applications, 2004. Proceedings of the 2004 IEEE International Conference on*, volume 2, pages 1454–1460. IEEE, 2004.
52. N. S. Nedialkov. Interval Tools for ODEs and DAEs. In *12th GAMM - IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2006), Duisburg, Germany*, 26–29 Sept 2006.
53. M. Neher, K. R. Jackson, and N. S. Nedialkov. On Taylor model based integration of ODEs. *SIAM Journal on Numerical Analysis*, 45(1):236–262, 2007.
54. T. Nishida, K. Mizutani, A. Kubota, and S. Doshita. Automated phase portrait analysis by integrating qualitative and quantitative analysis. In T. L. Dean and K. R. McKeown, editors, *Proceedings of the 9th National Conference on Artificial Intelligence, Anaheim, CA, USA, July 14-19, 1991, Volume 2.*, pages 811–816. AAAI Press / The MIT Press, 1991.
55. L. C. Paulson. MetiTarski: Past and future. In L. Beringer and A. P. Felty, editors, *Interactive Theorem Proving - Third International Conference, ITP 2012, Princeton, NJ, USA, August 13-15, 2012. Proceedings*, volume 7406 of *LNCS*, pages 1–10. Springer, 2012.
56. A. Platzer. Differential dynamic logic for hybrid systems. *Journal of Automated Reasoning*, 41(2):143–189, 2008.
57. A. Platzer and E. M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. In Gupta and Malik [28], pages 176–189.
58. A. Platzer and J. Quesel. KeYmaera: A hybrid theorem prover for hybrid systems (system description). In A. Armando, P. Baumgartner, and G. Dowek, editors, *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008.
59. A. Podelski and S. Wagner. Model checking of hybrid systems: From reachability towards stability. In J. P. Hespanha and A. Tiwari, editors, *Hybrid Systems: Computation and Control, 9th International Workshop, HSCC 2006, Santa Barbara, CA, USA, March 29-31, 2006, Proceedings*, volume 3927 of *LNCS*, pages 507–521. Springer, 2006.
60. A. Podelski and S. Wagner. Region stability proofs for hybrid systems. In J. Raskin and P. S. Thiagarajan, editors, *Formal Modeling and Analysis of Timed Systems, 5th International Conference, FORMATS 2007, Salzburg, Austria, October 3-5, 2007, Proceedings*, volume 4763 of *LNCS*, pages 320–335. Springer, 2007.
61. A. Podelski and S. Wagner. A sound and complete proof rule for region stability of hybrid systems. In Bemporad et al. [4], pages 750–753.
62. P. Prabhakar and M. G. Soto. Abstraction based model-checking of stability of hybrid systems. In Sharygina and Veith [69], pages 280–295.
63. S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In R. Alur and G. J. Pappas, editors, *Hybrid Systems: Computation and Control, 7th International Workshop, HSCC 2004, Philadelphia, PA, USA, March 25-27, 2004, Proceedings*, volume 2993 of *LNCS*, pages 477–492. Springer, 2004.
64. S. Ratschan and Z. She. Providing a basin of attraction to a target region of polynomial systems by computation of Lyapunov-like functions. *SIAM Journal of Control and Optimization*, 48(7):4377–4394, July 2010.
65. R. Rebiha, A. V. Moura, and N. Matringe. Generating invariants for non-linear hybrid systems. *Theoretical Computer Science*, 594:180–200, 2015.

66. D. Richardson. Some undecidable problems involving elementary functions of a real variable. *Journal of Symbolic Logic*, 33(4):514–520, 12 1968.
67. S. Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In K. H. Johansson and W. Yi, editors, *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2010, Stockholm, Sweden, April 12-15, 2010*, pages 221–230. ACM, 2010.
68. S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Constructing invariants for hybrid systems. *Formal Methods in System Design*, 32(1):25–55, 2008.
69. N. Sharygina and H. Veith, editors. *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, volume 8044 of *LNCS*. Springer, 2013.
70. A. Sogokon, K. Ghorbal, P. B. Jackson, and A. Platzer. A method for invariant generation for polynomial continuous systems. In B. Jobstmann and K. R. M. Leino, editors, *Verification, Model Checking, and Abstract Interpretation - 17th International Conference, VMCAI 2016, St. Petersburg, FL, USA, January 17-19, 2016. Proceedings*, volume 9583 of *LNCS*, pages 268–288. Springer, 2016.
71. A. Sogokon, K. Ghorbal, and T. T. Johnson. Operational models for piecewise-smooth systems. *ACM Transactions on Embedded Computing Systems*, 16(5):185:1–185:19, 2017.
72. A. Sogokon and P. B. Jackson. Direct formal verification of liveness properties in continuous and hybrid dynamical systems. In N. Bjørner and F. S. de Boer, editors, *FM 2015: Formal Methods - 20th International Symposium, Oslo, Norway, June 24-26, 2015, Proceedings*, volume 9109 of *LNCS*, pages 514–531. Springer, 2015.
73. A. W. Strzeboński. Cylindrical decomposition for systems transcendental in the first variable. *Journal of Symbolic Computation*, 46(11):1284–1290, 2011.
74. A. Taly and A. Tiwari. Deductive verification of continuous dynamical systems. In R. Kannan and K. N. Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2009, December 15-17, 2009, IIT Kanpur, India*, volume 4 of *LIPIcs*, pages 383–394. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2009.
75. A. Tiwari. Generating box invariants. In M. Egerstedt and B. Mishra, editors, *Hybrid Systems: Computation and Control, 11th International Workshop, HSCC 2008, St. Louis, MO, USA, April 22-24, 2008. Proceedings*, volume 4981 of *LNCS*, pages 658–661. Springer, 2008.
76. A. Vannelli and M. Vidyasagar. Maximal Lyapunov functions and domains of attraction for autonomous nonlinear systems. *Automatica*, 21(1):69 – 80, 1985.
77. S. Wang, N. Zhan, and L. Zou. An improved HHL prover: An interactive theorem prover for hybrid systems. In M. J. Butler, S. Conchon, and F. Zaïdi, editors, *Formal Methods and Software Engineering - 17th International Conference on Formal Engineering Methods, ICFEM 2015, Paris, France, November 3-5, 2015, Proceedings*, volume 9407 of *LNCS*, pages 382–399. Springer, 2015.
78. B. Xue, A. Easwaran, N. Cho, and M. Fränzle. Reach-avoid verification for nonlinear systems based on boundary analysis. *IEEE Transaction on Automatic Control*, 62(7):3518–3523, 2017.
79. H. Zhao, M. Yang, N. Zhan, B. Gu, L. Zou, and Y. Chen. Formal verification of a descent guidance control program of a lunar lander. In C. B. Jones, P. Pihlajasaari, and J. Sun, editors, *FM 2014: Formal Methods - 19th International Symposium, Singapore, May 12-16, 2014. Proceedings*, volume 8442 of *LNCS*, pages 733–748. Springer, 2014.
80. H. Zhao, N. Zhan, and D. Kapur. Synthesizing switching controllers for hybrid systems by generating invariants. In *Theories of Programming and Formal Methods - Essays Dedicated to Jifeng He on the Occasion of His 70th Birthday*, pages 354–373, 2013.

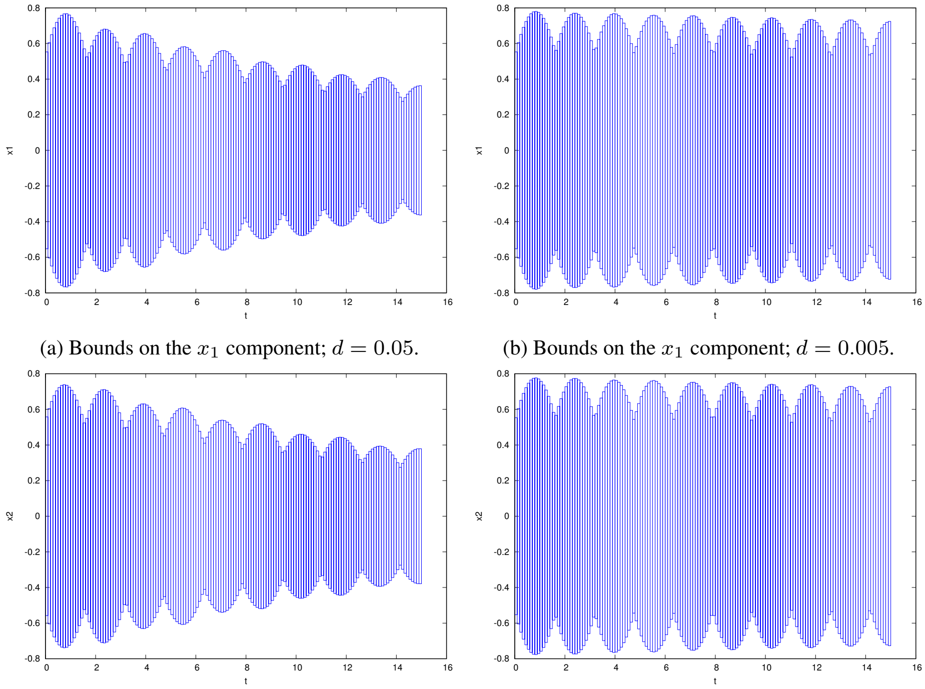
9 Appendix A.



(a) $d = 0.05$.

(b) $d = 0.005$.

Figure 10: Flowpipes from an initial box S (in green) around a stable equilibrium of an oscillator under different damping. Taylor model order: 10; time bound: 15. Flowpipe convergence is easier to show for larger damping factors.



(a) Bounds on the x_1 component; $d = 0.05$.

(b) Bounds on the x_1 component; $d = 0.005$.

(c) Bounds on the x_2 component; $d = 0.05$.

(d) Bounds on the x_2 component; $d = 0.005$.

Figure 11: Bounds on flowpipes from an initial box S.