Barrier Certificates Revisited

Liyun Dai, Ting Gan, Bican Xia

LMAM & School of Mathematical Sciences,Peking University

Naijun Zhan

State Key Lab. of Computer Science, Institute of Software, Chinese Academy of Sciences

Abstract

A barrier certificate can separate the state space of a considered hybrid system (HS) into safe and unsafe parts according to the safety property to be verified. Therefore this notion has been widely used in the verification of HSs. A stronger condition on barrier certificates (BCs) means that fewer BCs can be synthesized, as the expressiveness of synthesized BCs is weaker. On the other hand, synthesizing more expressive BCs normally means higher complexity. In [11], Kong et al. investigated how to relax the condition of BCs while still keeping their convexity so that one can synthesize more expressive BCs efficiently using semi-definite programming (SDP). In this paper, we first discuss how to relax the condition of BCs in a general way, while still keeping their convexity. Thus, one can utilize different weaker conditions flexibly to synthesize different kinds of BCs with more expressiveness efficiently using SDP, which gives more opportunities to verify the considered system. We also show how to combine two functions together to form a combined BC in order to prove a safety property under consideration, whereas neither of them may be a BC separately. In fact, the notion of combined BCs is strictly more expressive than that of BCs, so it further brings more chances to verify a considered system. Another contribution of this paper is to investigate how to avoid the unsoundness of SDP based approaches caused by numerical error through symbolic checking.

Key words: Hybrid system, barrier certificate, formal verification, invariant, nonlinear system, semi-definite programming, sum of squares

⋆ This work has been supported partly by "973 Program" under grant No. 2014CB340701, by NSFC under grants 91118007, 91418204, 11290141 and 11271034, by the CAS/SAFEA International Partnership Program for Creative Research Teams, and by CDZ project CAP (GZ 1023).

Email addresses: {dailiyun,gant}@pku.edu.cn, xbc@math.pku.edu.cn (Liyun Dai, Ting Gan, Bican Xia), znj@ios.ac.cn (Naijun Zhan).

URL: http://lcs.ios.ac.cn/~znj/ (Naijun Zhan).

1. Introduction

Embedded systems (ESs) make use of computer units to control physical devices so that the behavior of the controlled devices meets expected requirements. They have become ubiquitous in our daily life, e.g. automotive, aerospace, consumer electronics, communications, medical, manufacturing and so on. ESs are used to carry out highly complex and often critical functions, e.g. ESs are used to monitor and control industrial plants, complex transportation equipment, communication infrastructure, etc. The development process of ESs is widely recognized as a highly complex and challenging task. A thorough validation and verification activity is necessary to enhance the quality of the ESs and, in particular, to fulfill the quality criteria mandated by the relevant standards. Hybrid systems (HSs) are mathematical models with precise mathematical semantics for ESs, wherein continuous physical dynamics are combined with discrete transitions. Based on HSs, rigorous analysis and verification of ESs become feasible, so that errors can be detected and corrected in the very early stage of the design of ESs.

In the past, analysis and verification of HSs are mainly done through directly computing reachable sets, either by model-checking (e.g., [1, 25, 8]) or by decision procedures (e.g., [13]). The basic idea is to partition the state space of a considered system into finitely many equivalent classes, or represent it by finitely many computable sets according to the solutions of the ODE of the system. Since there is only a very small class of ODEs with closed form solutions, the scalability of these approaches is very restricted, only applicable to very specific linear HSs. Recently, there are lots of work based on abstraction and numeric approximation to scale up these approaches, e.g., [4, 2, 5, 26]. In principle, these approaches are quite successful in the falsification of a considered H-S by debugging bugs using bounded model-checking, but have difficulty in proving the error-avoidance of the system. As an alternative, deductive methods have been recently proposed and successfully applied in practice [20, 21, 15]. The most challenging part of a deductive method is how to discover invariants, which hold at all reachable states of the system. For technical reasons, people only consider how to synthesize inductive invariants, which are preserved by all discrete and continuous transitions. In general, a safety property itself is an invariant, but may not be an inductive invariant. Obviously, an inductive invariant is an approximation of the reachable set, which may be discovered according to the ODE, rather than its solutions. The basic idea is as follows: first, predefine a property template (linear or non-linear, depending on the property to be verified); then, encode the conditions of a property to be inductive (discretely and/or continuously) into some constraints on state variables and parameters; finally, find out solutions to the constraints. So, how to define inductive conditions and the power of constraint solving are essential to these approaches.

Many approaches have been proposed following the line discussed above. E.g., in [9, 27], the authors independently proposed different approaches for constructing inductive invariants for linear HSs; Sankaranarayanan et al. presented a computational method to automatically generate algebraic invariants for algebraic HSs in [28, 29], based on the theory of pseudo-ideals over polynomial rings and quantifier elimination; Prajna in [22, 23] provided a new notion of inductive invariants called *barrier certificates* (BC) for verifying the safety of semi-algebraic HSs in the stochastic setting using the technique of sum-of-squares (**SOS**); In [20], Platzer and Clarke extended the idea of *BCs* by considering Boolean combinations of multiple polynomial inequalities; In [6, 31], Gulwani

et al. investigated how to generate inductive invariants with more expressiveness for semi-algebraic HSs through relaxing the inductive conditions by considering inductiveness only on the boundaries of predefined invariant templates; while in [16], Liu et al. considered how to further relax the inductive condition given in [6, 31] and established a first finite complete inductive condition to determine if a polynomial formula is an invariant of a given semi-algebraic HS. In [30], Solth at el. proposed an approach to constructing global inductive invariants from local differential invariants using optimization techniques.

The aforementioned approaches can be classified into two categories: symbolic computation based like [9, 27, 28, 29, 20, 6, 31, 16], and numeric computation based like [22, 23, 30]. In general, the former can synthesize more expressive invariants, but their efficiencies are very low; in contrast, the efficiency of the latter is very high, normally in *polynomial time*, which is the complexity of SDP that is used to solve the resulted constraints derived from the corresponding inductiveness. But the disadvantages of the latter is twofold : the expressiveness of synthesized invariants is restrictive; what is more, they may be unsound because of the numerical error caused by SDP, which means that a synthesized invariant could not be a real one.

In [11], Kong et al. investigated how to synthesize more expressive BCs by proposing *exponential BC condition*, which is a relaxed inductive condition, but still keeps the convexity of *barrier certificate*s. So, more expressive BCs can be synthesized with high efficiency by SDP.

In this paper, firstly, following Kong et al.'s line, in the prerequisite of keeping the convexity of the condition of BCs so that SDP is still applicable, we discuss how to relax the condition of BCs in a general way. Thus, one can utilize different weaker conditions flexibly to synthesize different kinds of BCs with more expressiveness efficiently, which gives more opportunities to verify the considered system. In addition, we consider how to combine two functions together to form a combined BC which can guarantee a safety property under consideration, whereas neither of them can be used as a BC separately. In fact, we can prove that the notion of combined BCs is strictly more expressive than that of BCs. Another contribution of this paper is that we investigate how to avoid the unsoundness of SDP based approaches caused by numerical errors through symbolic checking.

The rest of the paper is organized as follows: Section 2 introduces some basic notions; In Section 3, we discuss how to relax BC conditions, as well as how to combine two functions to form a combined BC, even when none of them can be used as a BC separately; Section 4 is devoted to how to synthesize BCs according to relaxed conditions based on SDP, and how to avoid unsoundness caused by numerical errors through symbolic checking; Section 5 discusses how to extend our approach to more general systems beyond polynomials and/or with inputs; Section 6 provides some case studies as well as experimental results. Finally, we conclude this paper in Section 7.

2. Preliminaries

In this section, we first introduce some basic notions, and then explain the basic idea of barrier certificates (BCs).

In what follows, we use $\mathbb{R}$ to stand for the set of reals, $\mathcal{C}^\omega[\mathbb{R}^n]$ for the set of analytic function from $\mathbb{R}^n$ to $\mathbb{R}$. A subset $A \subseteq \mathbb{R}^n$ is called *semi-algebraic* if there is a quantifier-free polynomial formula φ expressed in Tarski's algebra s.t. $A = \{\mathbf{x} \in \mathbb{R}^n \mid \varphi(\mathbf{x}) \text{ is true}\}$.

2.1. Basic notions

An autonomous continuous dynamical system (CDS) is represented by a differential equation of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{f}$ is a vector function, called *vector field*, whose components are in $\mathcal{C}^\omega[\mathbb{R}^n]$, satisfying local Lipschitz condition [1]. In the context of HSs, a CDS is normally equipped with a domain $D \subseteq \mathbb{R}^n$ constraining its state space and an initial set of states Ξ.

In this paper, we use hybrid automata [1] to model HSs. More models of HSs can be found in [34].

Definition 1 (Hybrid Automata)**.** A *hybrid automaton* (HA) is a tuple $\mathcal{H} \widehat{=} (Q, X, \mathbf{f}, D, E, G, R, \Xi)$, where

- $Q = \{q_1, \ldots, q_m\}$ is a finite set of discrete states (or modes);
- $X = \{x_1, \ldots, x_n\}$ is a finite set of continuous state variables, with $\mathbf{x} = (x_1, \ldots, x_n)$ ranging over $\mathbb{R}^n$;
- $\mathbf{f} : Q \to (\mathbb{R}^n \to \mathbb{R}^n)$ assigns to each mode $q \in Q$ a locally Lipschitz continuous vector field $\mathbf{f}_q$;
- D assigns to each mode $q \in Q$ a mode domain $D_q \subseteq \mathbb{R}^n$;
- $E \subseteq Q \times Q$ is a finite set of discrete transitions;
- G assigns to each transition $e \in E$ a switching guard $G_e \subseteq \mathbb{R}^n$;
- R assigns to each transition $e \in E$ a reset function R_e: $\mathbb{R}^n \to \mathbb{R}^n$;
- Ξ assigns to each $q \in Q$ a set of initial states $\Xi_q \subseteq \mathbb{R}^n$.

The state space of $\mathcal{H}$ is $\mathbb{H} \widehat{=} Q \times \mathbb{R}^n$, the domain of $\mathcal{H}$ is $D_{\mathcal{H}} \widehat{=} \bigcup_{q \in Q}(\{q\} \times D_q)$, and the set of all initial states is denoted by $\Xi_{\mathcal{H}} \widehat{=} \bigcup_{q \in Q}(\{q\} \times \Xi_q)$. The semantics of $\mathcal{H}$ can be characterized by the set of *hybrid trajectories* accepted by $\mathcal{H}$ or the *reachable set* of $\mathcal{H}$.

Definition 2 (Hybrid Time Set)**.** A *hybrid time set* is a sequence of intervals $\tau = \{I_i\}_{i=0}^N$ (N may be ∞) such that:

- $I_i = [\tau_i, \tau_i']$ with $\tau_i \leq \tau_i' = \tau_{i+1}$ for all $i < N$;
- if $N < \infty$, then $I_N = [\tau_N, \tau_N'\rangle$ is a right-closed or right-open nonempty interval (τ_N' may be ∞);
- $\tau_0 = 0$.

Given a hybrid time set, let $\langle\tau\rangle = N$, and $\|\tau\| = \sum_{i=0}^N (\tau_i' - \tau_i)$. Then τ is called *infinite* if $\langle\tau\rangle = \infty$ or $\|\tau\| = \infty$, and *zeno* if $\langle\tau\rangle = \infty$ but $\|\tau\| < \infty$.

Definition 3 (Hybrid Trajectory)**.** A *hybrid trajectory* of $\mathcal{H}$ starting from an initial point $(q_0, \mathbf{x}_0) \in \Xi_{\mathcal{H}}$ is a triple $\omega = (\tau, \alpha, \beta)$, where $\tau = \{I_i\}_{i=0}^N$ is a hybrid time set, and $\alpha = \{\alpha_i : I_i \to Q\}_{i=0}^N$ and $\beta = \{\beta_i : I_i \to \mathbb{R}^n\}_{i=0}^N$ are two sequences of functions satisfying:

(1) *Initial condition*: $\alpha_0[0] = q_0$ and $\beta_0[0] = \mathbf{x}_0$;

[1] Local Lipschitz condition guarantees the existence and uniqueness of the solution of (1) from any initial $\mathbf{x}_0$.

(2) *Discrete transition*: for all $i < \langle\tau\rangle$, $e = \big(\alpha_i(\tau_i'), \alpha_{i+1}(\tau_{i+1})\big) \in E$, $\beta_i(\tau_i') \in G_e$ and $\beta_{i+1}(\tau_{i+1}) = R_e(\beta_i(\tau_i'))$;
(3) *Continuous evolution*: for all $i \leq \langle\tau\rangle$ with $\tau_i < \tau_i'$, if $q = \alpha_i(\tau_i)$, then
  (1) for all $t \in I_i$, $\alpha_i(t) = q$,
  (2) $\beta_i(t)$ is the solution to the differential equation $\dot{\mathbf{x}} = \mathbf{f}_q(\mathbf{x})$ over I_i with initial value $\beta_i(\tau_i)$, and
  (3) for all $t \in [\tau_i, \tau_i')$, $\beta_i(t) \in D_q$.

The set of trajectories starting from an initial state $(q_0, \mathbf{x}_0)$ of $\mathcal{H}$ is denoted by $\mathcal{T}r(\mathcal{H})(q_0, \mathbf{x}_0)$, and the set of all trajectories of $\mathcal{H}$ by $\mathcal{T}r(\mathcal{H})$.

A hybrid trajectory $\omega = (\tau, \alpha, \beta)$ is called *infinite* or *zeno*, if τ is infinite or zeno respectively. An HA $\mathcal{H}$ is called *non-blocking* if for any $(q_0, \mathbf{x}_0) \in \Xi_{\mathcal{H}}$ there exists an infinite trajectory in $\mathcal{T}r(\mathcal{H})(q_0, \mathbf{x}_0)$, and *blocking* otherwise; $\mathcal{H}$ is called *non-zeno* if there exists no zeno trajectory in $\mathcal{T}r(\mathcal{H})$, and *zeno* otherwise.

Another way to interpret hybrid automata is using reachability relation.

Definition 4 (Reachable Set)**.** Given an HA $\mathcal{H}$, the *reachable set* of $\mathcal{H}$, denoted by $\mathcal{R}_{\mathcal{H}}$, consists of those $(q, \mathbf{x})$ for which there exists a finite sequence

$$(q_0, \mathbf{x}_0), (q_1, \mathbf{x}_1), \ldots, (q_l, \mathbf{x}_l)$$

such that $(q_0, \mathbf{x}_0) \in \Xi_{\mathcal{H}}$, $(q_l, \mathbf{x}_l) = (q, \mathbf{x})$, and for any $0 \leq i \leq l-1$, one of the following two conditions holds:

- (*Discrete Jump*): $e = (q_i, q_{i+1}) \in E$, $\mathbf{x}_i \in G_e$ and $\mathbf{x}_{i+1} = R_e(\mathbf{x}_i)$; or
- (*Continuous Evolution*): $q_i = q_{i+1}$, and there exists a $\delta \geq 0$ s.t. the solution $\mathbf{x}(\mathbf{x}_i; t)$ to $\dot{\mathbf{x}} = \mathbf{f}_{q_i}$ satisfies
  · $\mathbf{x}(\mathbf{x}_i; t) \in D_{q_i}$ for all $t \in [0, \delta]$; and
  · $\mathbf{x}(\mathbf{x}_i; \delta) = \mathbf{x}_{i+1}$.

Note that there is a subtle difference between Definition 3 and 4 in how to treat a continuous state $\mathbf{x}$ which terminates a piece of continuous evolution and evokes a discrete jump. Definition 3 is less restrictive because such $\mathbf{x}$ is not required to be inside the mode domain before jump happens. Nevertheless, if all mode domains are assumed to be *closed* sets, then the above two definitions are consistent with each other, that is, $\mathcal{R}_{\mathcal{H}}$ is exactly the set of states that are covered by $\mathcal{T}r(\mathcal{H})$.

One of the major concerning properties of hybrid systems is *safety*. Given an HA $\mathcal{H}$, a safety requirement $\mathcal{S}$ assigns to each mode $q \in Q$ a safe region $S_q \subseteq \mathbb{R}^n$, i.e. $\mathcal{S} = \bigcup_{q \in Q}(\{q\} \times S_q)$. We say that $\mathcal{H}$ satisfies $\mathcal{S}$ if $\mathbf{x} \in S_q$ for all $(q, \mathbf{x}) \in \mathcal{R}_{\mathcal{H}}$. Dually, $\mathcal{S}^u = \bigcup_{q \in Q}(\{q\} \times (D_q - S_q))$ is called the *unsafe set*.

2.2. Barrier certificates

Given an HS $\mathcal{H}$ and a safety property $\mathcal{S}$ (dually, an unsafe set $\mathcal{S}^u$), the problem we considered is whether $\mathcal{R}_{\mathcal{H}} \subseteq \mathcal{S}$ (dually, $\mathcal{R}_{\mathcal{H}} \cap \mathcal{S}^u = \emptyset$). Obviously, it is equivalent to $\forall q \in Q.\mathcal{R}_{\mathcal{H}}\restriction_q \subseteq S_q$ (dually, $\forall q \in Q.\mathcal{R}_{\mathcal{H}}\restriction_q \cap S_q^u = \emptyset$), where $\mathcal{R}_{\mathcal{H}}\restriction_q$ stands for all continuous states of $\mathcal{R}_{\mathcal{H}}$ projected onto q. For this problem on CDSs, Prajna et al. in [22, 23] used the idea of Lyapunov functions for stability analysis in control theory to separate safe states from unsafe states by a barrier function with convexity, called *barrier certificate*. According to their definition, a barrier function $\varphi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n]$ satisfies the following conditions:

i) $\varphi(\mathbf{x}) \leq 0$ for any point $\mathbf{x} \in \Xi$;
ii) $\varphi(\mathbf{x}) > 0$ for any point $\mathbf{x} \in \mathcal{S}^u$; and
iii) $\forall \mathbf{x} \in D.\mathcal{L}_{\mathbf{f}}\varphi(\mathbf{x}) \leq 0$, where $\mathcal{L}_{\mathbf{f}}\varphi(\mathbf{x}) = \frac{\partial \varphi}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x})$ is the Lie derivative of φ with respect to the vector field $\mathbf{f}$.

Trivially to see, the existence of a BC is just a sufficient condition to guarantee the safety property to be verified. Hence, using the approach of Prajna et al., one cannot claim the property does not hold if he/she fails to discover a polynomial BC. Actually, as observed in [11] by Kong et al., if condition iii) is relaxed by

iii') $\mathcal{L}_{\mathbf{f}}\varphi(\mathbf{x}) - \gamma\varphi(\mathbf{x}) \leq 0$, where γ is a real number,

one can synthesize BCs with more expressiveness. Certainly, it is more likely to prove a safety property by using a more expressive BC, as it gives a tighter approximation of the reachable set.

3. Revisiting Barrier Certificate Conditions

In this section, we revisit the condition of BCs so that it can be relaxed in a general way, while allowing BCs to be efficiently synthesized according to the relaxed condition.

3.1. Relaxed barrier certificate conditions for CDSs

First of all, we consider how to relax the condition i)-iii) of BCs given in [22, 23] for CDSs in a general way. To that end, we need to have a principle to justify in what sense a relaxed condition of BCs is reasonable and thus acceptable. An obvious principle is:

Principle of BC (PBC): Given a CDS $\mathcal{D}$ equipped with an initial set Ξ and an unsafe set $\mathcal{S}^u$, a BC should be a real-valued function $\varphi(\mathbf{x})$ such that $\varphi(\mathbf{x}) \leq 0$ for any $\mathbf{x} \in \mathcal{R}_{\mathcal{D}}$, and $\varphi(\mathbf{x}) > 0$ for any point $\mathbf{x} \in \mathcal{S}^u$.

Certainly, if there exists such a function $\varphi(\mathbf{x})$, we can assert that $\mathcal{R}_{\mathcal{D}} \cap \mathcal{S}^u = \emptyset$, and $\phi(\mathbf{x}) \leq 0$ is therefore an invariant. However, such a principle cannot be effectively checked in general, so we have to strengthen the condition to make it effectively checkable, like in [22, 23, 11]. An interesting problem is with which condition more expressive BCs can be synthesized, but the condition is still effectively checkable and satisfies **PBC**. We answer this problem by the following theorem.

Theorem 1 (General Barrier Condition (**GBC**))**.** Given a CDS $\mathcal{D}$ equipped with a domain D, an initial set Ξ and an unsafe set $\mathcal{S}^u$, if there is a function $\varphi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n]$, a function $\psi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}]$ such that

$$\begin{aligned} &\forall \mathbf{x} \in \Xi.\varphi(\mathbf{x}) \leq 0, && (2)\\ &\forall \mathbf{x} \in D.\mathcal{L}_{\mathbf{f}}\varphi(\mathbf{x}) - \psi(\varphi(\mathbf{x})) \leq 0, && (3)\\ &\forall \mathbf{x} \in \mathcal{S}^u.\varphi(\mathbf{x}) > 0, && (4)\\ &\forall \xi.\xi > 0 \Rightarrow \theta(\mathbf{x}(\xi)) \leq 0, \end{aligned}$$

where $\theta(\mathbf{x}(t)) : \mathbb{R} \to \mathbb{R}$ is a function such that

$$\begin{cases} \theta(\mathbf{x}(0)) \leq 0, \\ \mathcal{L}_{\mathbf{f}}\theta(\mathbf{x}) - \psi(\theta(\mathbf{x})) = 0, \end{cases} \tag{5}$$

for any solution $\mathbf{x}(t)$ with initial value $\mathbf{x}(0)$, then $\mathcal{R}_{\mathcal{D}} \cap \mathcal{S}^u = \emptyset$.

Proof. Suppose $\mathbf{x}_0 \in \Xi$ and $\mathbf{x}(t)$ is the corresponding solution of (1) starting from $\mathbf{x}_0$. By (4), we only need to prove that for any function $\varphi(\mathbf{x}(t))$ satisfying (2), (3) and (5),

$$\forall \xi \geq 0.\varphi(\mathbf{x}(\xi)) \leq 0. \tag{6}$$

We will show (6) by contradiction.

Assume $\varphi(\mathbf{x}(\xi')) > 0$ for some $\xi' > 0$. Let $g(\mathbf{x}) = \mathcal{L}_{\mathbf{f}}\varphi(\mathbf{x}) - \psi(\varphi(\mathbf{x}))$, then by (3)

$$\forall \mathbf{x} \in D.g(\mathbf{x}) \leq 0. \tag{7}$$

Since $\frac{d\varphi(\mathbf{x}(t))}{dt} = \frac{\partial \varphi}{\partial \mathbf{x}}\frac{d\mathbf{x}}{dt} = \frac{\partial \varphi}{\partial \mathbf{x}}f(\mathbf{x}) = \mathcal{L}_f\varphi(\mathbf{x})$, we have

$$\begin{cases} \frac{d\varphi(\mathbf{x}(t))}{dt} - \psi(\varphi(\mathbf{x}(t))) - g(\mathbf{x}(t)) = 0, \\ \varphi(\mathbf{x}(0)) = \varphi(\mathbf{x}_0). \end{cases} \tag{8}$$

Let $\theta(\mathbf{x}(t))$ be a function such that

$$\begin{cases} \frac{d\theta(\mathbf{x}(t))}{dt} - \psi(\theta(\mathbf{x}(t))) = 0, \\ \theta(\mathbf{x}(0)) = \varphi(\mathbf{x}_0). \end{cases} \tag{9}$$

for any solution $\mathbf{x}(t)$ with initial value $\mathbf{x}(0)$. Then, set

$$\Theta = \{\xi \ \mid \varphi(\mathbf{x}(\xi)) > \theta(\mathbf{x}(\xi)), \xi \geq 0\}.$$

From (5) we have $\forall \xi > 0.\theta(\mathbf{x}(\xi)) \leq 0$. So Θ is nonempty as $\xi' \in \Theta$ by the assumption. Then there is a real number $\mu \geq 0$ s.t. $\mu = \inf(\Theta)$. Obviously, $\varphi(\mathbf{x}(t))$, $\theta(\mathbf{x}(t))$, $g(\mathbf{x}(t))$, $\frac{d\varphi(\mathbf{x}(t))}{dt}$ and $\frac{d\theta(\mathbf{x}(t))}{dt}$ are analytic functions w.r.t. t. Thus $\varphi(\mathbf{x}(\mu)) = \theta(\mathbf{x}(\mu))$.

If $g(\mathbf{x}(\mu)) < 0$, then $\frac{d\varphi(\mathbf{x}(t))}{dt}|_{t=\mu} < \frac{d\theta(\mathbf{x}(t))}{dt}|_{t=\mu}$. Hence,

$$\exists \nu.\ \nu > \mu \wedge \forall \xi \in (\mu, \nu).\ \frac{d\varphi(\mathbf{x}(t))}{dt}|_{t=\xi} < \frac{d\theta(\mathbf{x}(t))}{dt}|_{t=\xi}.$$

Thus, $\forall \xi \in (\mu, \nu).\varphi(\mathbf{x}(\xi)) < \theta(\mathbf{x}(\xi))$, which contradicts the definition of μ.

So $g(\mathbf{x}(\mu)) = 0$ from (7). Since $\varphi(\mathbf{x}(\mu)) = \theta(\mathbf{x}(\mu))$, from (8) and (9), we have

$$\frac{d\varphi(\mathbf{x}(t))}{dt}|_{t=\mu} = \frac{d\theta(\mathbf{x}(t))}{dt}|_{t=\mu}.$$

Case 1. If $\forall k > 1.\frac{d^k\varphi(\mathbf{x}(t))}{dt^k}|_{t=\mu} = \frac{d^k\theta(\mathbf{x}(t))}{dt^k}|_{t=\mu}$, then $\varphi(\mathbf{x}(\xi)) = \theta(\mathbf{x}(\xi))$ for any $\xi \in \mathbb{R}^+$, since φ, θ are analytic functions. So $\Theta = \emptyset$, which contradicts the definition of μ.

Case 2. Suppose there is $k > 1$ such that

$$\frac{d^k\varphi(\mathbf{x}(t))}{dt^k}|_{t=\mu} < \frac{d^k\theta(\mathbf{x}(t))}{dt^k}|_{t=\mu}, \quad \frac{d^i\varphi(\mathbf{x}(t))}{dt^i}|_{t=\mu} = \frac{d^i\theta(\mathbf{x}(t))}{dt^i}|_{t=\mu},\ \forall i < k,$$

then there is $\nu_1 > \mu$ s.t. $\varphi(\mathbf{x}(\xi)) < \theta(\mathbf{x}(\xi))$ for any $\xi \in (\mu, \nu_1)$, which also contradicts the definition of μ.

Case 3. Suppose there is $k > 1$ such that

$$\frac{d^k\varphi(\mathbf{x}(t))}{dt^k}|_{t=\mu} > \frac{d^k\theta(\mathbf{x}(t))}{dt^k}|_{t=\mu}, \quad \frac{d^i\varphi(\mathbf{x}(t))}{dt^i}|_{t=\mu} = \frac{d^i\theta(\mathbf{x}(t))}{dt^i}|_{t=\mu},\ \forall i < k.$$

Since $\varphi(\mathbf{x}(\mu)) = \theta(\mathbf{x}(\mu))$, then

$$\psi(\varphi(\mathbf{x}(\mu))) = \psi(\theta(\mathbf{x}(\mu))).$$

Since $\frac{d\varphi(\mathbf{x}(t))}{dt}|_{t=\mu} = \frac{d\theta(\mathbf{x}(t))}{dt}|_{t=\mu}$, then

$$\begin{aligned}\frac{d\psi(\varphi(\mathbf{x}(t)))}{dt}|_{t=\mu} &= \frac{d\psi(y)}{dy}|_{y=\varphi(\mathbf{x}(\mu))}\frac{d\varphi(\mathbf{x}(t))}{dt}|_{t=\mu} = \frac{d\psi(y)}{dy}|_{y=\theta(\mathbf{x}(\mu))}\frac{d\theta(\mathbf{x}(t))}{dt}|_{t=\mu}\\ &= \frac{d\psi(\theta(\mathbf{x}(t)))}{dt}|_{t=\mu},\end{aligned}$$

i.e., $\frac{d\psi(\varphi(\mathbf{x}(t)))}{dt}|_{t=\mu} = \frac{d\psi(\theta(\mathbf{x}(t)))}{dt}|_{t=\mu}$. If $\frac{d^2\varphi(\mathbf{x}(t))}{dt^2}|_{t=\mu} = \frac{d^2\theta(\mathbf{x}(t))}{dt^2}|_{t=\mu}$, then

$$\begin{aligned}\frac{d^2\psi(\varphi(\mathbf{x}(t)))}{dt^2}|_{t=\mu} &= \frac{d^2\psi(y)}{dy^2}|_{y=\varphi(\mathbf{x}(\mu))}(\frac{d\varphi(\mathbf{x}(t))}{dt})^2|_{t=\mu} + \frac{d\psi(y)}{dy}|_{y=\varphi(\mathbf{x}(\mu))}\frac{d^2\varphi(\mathbf{x}(t))}{dt^2}|_{t=\mu}\\ &= \frac{d^2\psi(y)}{dy^2}|_{y=\theta(\mathbf{x}(\mu))}(\frac{d\theta(\mathbf{x}(t))}{dt})^2|_{t=\mu} + \frac{d\psi(y)}{dy}|_{y=\theta(\mathbf{x}(\mu))}\frac{d^2\theta(\mathbf{x}(t))}{dt^2}|_{t=\mu}\\ &= \frac{d^2\psi(\theta(\mathbf{x}(t)))}{dt^2}|_{t=\mu},\end{aligned}$$

i.e., $\frac{d^2\psi(\varphi(\mathbf{x}(t)))}{dt^2}|_{t=\mu} = \frac{d^2\psi(\theta(\mathbf{x}(t)))}{dt^2}|_{t=\mu}$. By induction, for all $i < k$ we have that

$$\frac{d^i\psi(\varphi(\mathbf{x}(t)))}{dt^i}|_{t=\mu} = \frac{d^i\psi(\theta(\mathbf{x}(t)))}{dt^i}|_{t=\mu}.$$

For $i < k$, computing the $(i-1)$-th derivatives of the two sides of the first formulae of (8) and (9), we have

$$\begin{cases}\frac{d^i\varphi(\mathbf{x}(t))}{dt^i}|_{t=\mu} - \frac{d^{i-1}\psi(\varphi(\mathbf{x}(t)))}{dt^{i-1}}|_{t=\mu} - \frac{d^{i-1}g(\mathbf{x}(t))}{dt^{i-1}}|_{t=\mu} = 0,\\ \frac{d^i\theta(\mathbf{x}(t))}{dt^i}|_{t=\mu} - \frac{d^{i-1}\psi(\theta(\mathbf{x}(t)))}{dt^{i-1}}|_{t=\mu} = 0.\end{cases} \tag{10}$$

Since,

$$\frac{d^k\varphi(\mathbf{x}(t))}{dt^k}|_{t=\mu} > \frac{d^k\theta(\mathbf{x}(t))}{dt^k}|_{t=\mu}, \quad \frac{d^i\varphi(\mathbf{x}(t))}{dt^i}|_{t=\mu} = \frac{d^i\theta(\mathbf{x}(t))}{dt^i}|_{t=\mu}, \ \forall i < k,$$

and for all $i < k$,

$$\frac{d^i\psi(\varphi(\mathbf{x}(t)))}{dt^i}|_{t=\mu} = \frac{d^i\psi(\theta(\mathbf{x}(t)))}{dt^i}|_{t=\mu},$$

then from (10) we have that

$$\frac{d^{k-1}g(\mathbf{x}(t))}{dt^{k-1}}|_{t=\mu} > 0, \quad \frac{d^{i-1}g(\mathbf{x}(t))}{dt^{i-1}}|_{t=\mu} = 0, \ \forall i < k-1.$$

Thus, there is a $\delta > \mu$ s.t. $\forall \xi \in (\mu, \delta).g(\mathbf{x}(\xi)) > 0$, which contradicts the definition of $g(\mathbf{x})$. This completes the proof. □

From now on, we call φ in Theorem 1 a *barrier certificate* of $\mathcal{D}$.

Remark 1. • The application of Theorem 1 includes the following two steps:
i) fix a function ψ which satisfies condition (5) first;
ii) similar to the work in [11], synthesize BC according to the resulting conditions of (2)-(4) by instantiating ψ with the function obtained in the first step.
• All existing BC conditions can be uniformly represented in our general condition with specific functions ψ satisfying condition (5). For instance, *convex condition* in [23] and

differential invariant in [20] correspond to $\psi(\theta) = 0$, while *exponential condition* in [11] corresponds to $\psi(\theta) = \alpha\theta$, where $\alpha \in \mathbb{R}$.

The following lemma indicates that we can find so many classes of functions ψ different from existing ones, satisfying condition (5). Thus, from these we can construct a class of relaxed conditions of BCs by **GBC**, that can be used to generate BCs with different expressiveness.

Lemma 1. Given

$$\begin{cases} \theta(0) \leq 0, \\ \dot{\theta} - \psi(\theta) = 0, \end{cases} \tag{11}$$

where ψ is a first order continuous differentiable function with respect to θ such that the solution $\theta(t)$ of (11) is well-defined for all $t \geq 0$. If $\psi(0) = 0$, then for any $t \geq 0$, $\theta(t) \leq 0$.

Proof. (By contradiction). Assume there is a solution θ of (11) with $\theta(0) \leq 0$, but $\theta(t^*) > 0$ for some $t^* > 0$. It is easy to see $\theta(t)$ is continuous at t^*. So, there must exist $t_0 > 0, \delta > 0$, s.t. $\theta(t_0) = 0$, and for all $t_0 < t \leq t_0 + \delta, \theta(t) > 0$.

Fix a real number $\epsilon > 0$. Obviously, $\frac{\partial\psi(\theta)}{\partial\theta}$ is bounded whenever $\theta \in [-\epsilon, \epsilon]$, since ψ is a first order continuous differentiable function. Thus, $\dot{\theta} = \psi(\theta)$ w.r.t. θ satisfies the local Lipschitz condition in $[t_0, t_0 + \delta] \times [-\epsilon, \epsilon]$. Hence, $\dot{\theta} = \psi(\theta)$ with $\theta(t_0) = 0$ has a unique solution over $[t_0, t_0 + \delta]$. On the other hand, it is easy to see that $\theta(t) = 0$ is indeed a solution of (11) as $\psi(0) = 0$. This implies that $\forall t \in [t_0, t_0 + \delta].\theta(t) = 0$, which contradicts the assumption. □

From Lemma 1, in order to find a function ψ satisfying condition (5) in Theorem 1, we just need to ensure ψ to be a first order continuous differentiable function w.r.t. θ and $\psi(0) = 0$. Obviously, $\psi(\theta) = 0$ in [23] and [20] and $\psi(\theta) = \alpha\theta$ in [11] both satisfy the conditions. The following lemma clearly indicates that more general ψ do exist.

Lemma 2. Let

$$\begin{cases} \dot{\theta} - \alpha\theta - \beta\theta^2 = 0, \\ \theta(0) = \theta_0 \leq 0. \end{cases} \tag{12}$$

If $\alpha < 0 \wedge (\beta > 0 \vee (\beta < 0 \wedge \frac{\alpha}{\beta} + \theta_0 > 0))$, then $\theta(t)$ is well defined for $t \geq 0$. Besides, $\forall t > 0.\, \theta(t) \leq 0$.

Proof. (12) has an explicit solution $\theta(t) = \frac{\alpha}{\beta}(\frac{1}{1 - \frac{\theta_0}{\frac{\alpha}{\beta}+\theta_0} e^{\alpha t}} - 1)$. It is easy to see that if $\alpha < 0 \wedge (\beta > 0 \vee (\beta < 0 \wedge \frac{\alpha}{\beta} + \theta_0 > 0))$, then $1 - \frac{\theta_0}{\frac{\alpha}{\beta}+\theta_0} e^{\alpha t}$ will never vanish for $t \geq 0$. Thus, $\theta(t)$ is well defined for $t \geq 0$. Since $\psi(\theta) = \alpha\theta + \beta\theta^2$ is a continuous differentiable function w.r.t. θ and $\psi(0) = 0$, by Lemma 1, the conclusion holds. □

Remark 2. The above lemma tells us that a class of quadratic functions ψ can be used to construct relaxed BC conditions in Theorem 1. One can flexibly select an appropriate relaxed condition from the above class by setting suitable values to α and β according to the following rules, which is illustrated in Fig. 1:

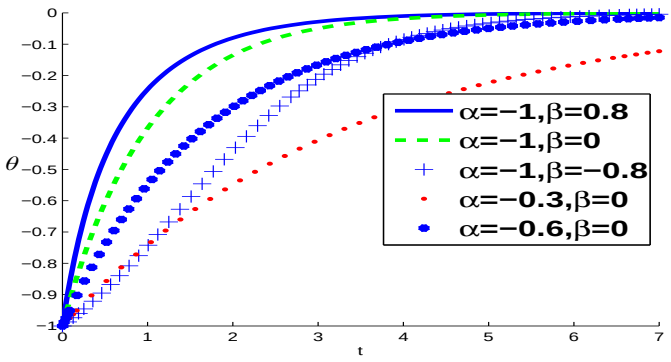

Fig. 1. Solutions of (12) with $\theta_0 = -1$ on different values of α, β.

- the smaller the value of α is, the tighter synthesized BCs are, and vice versa;
- the greater the value of β is, the tighter synthesized BCs are, and vice versa.

The following example clearly indicates that one can synthesize some interesting BCs with some relaxed condition from the above class, which cannot be discovered using the existing approaches.

Example 1. Consider a CDS $\mathcal{D}_1$ as follows:

$$\begin{cases} \dot{x_1} = x_1^2 - 2x_1 + x_2, \\ \dot{x_2} = x_1 + x_2^2 - 2x_2, \end{cases}$$

with $\Xi = \{(x_1, x_2) \mid 0.01 - x_1^2 - x_2^2 \geq 0\}, \mathcal{S}^u = \{(x_1, x_2) \mid x_1^2 + x_2^2 - 0.25 \geq 0\}$.

By Theorem 1, we can check whether $\varphi = x_1^2 + x_2^2 - 0.04$ is a BC w.r.t. $\psi(\theta) = -\theta + 2\theta^2$ as follows: Let $g_0 = 0.01 - x_1^2 - x_2^2$, $g_1 = x_1^2 + x_2^2 - 0.25$. Obviously, $-\varphi - g_0 = 0.03 > 0$, $\varphi - g_1 = 0.21 > 0$ and $-\mathcal{L}_f(\varphi) - \varphi + 2\varphi^2 = 2x_1^4 - 2x_1^3 + 4x_1^2x_2^2 + 2.84x_1^2 - 4x_1x_2 + 2x_2^4 - 2x_2^3 + 2.84x_2^2 + 0.0432$ is an **SOS**, so the condition of Theorem 1 is satisfied.

On the other hand, we can show that there is no BC φ with $deg(\varphi) \leq 2$ that can be synthesized by the condition given in [11]. Suppose there is a BC satisfying the condition of [11], i.e., with the form

$$\varphi = a_{20}x_1^2 + a_{11}x_1x_2 + a_{02}x_2^2 + a_{10}x_1 + a_{01}x_2 + a_{00}$$

w.r.t. $\psi(\theta) = \alpha\theta$, where $\alpha, a_{20}, a_{11}, a_{02}, a_{10}, a_{01}, a_{00} \in \mathbb{R}$. Let $L = -\mathcal{L}_f(\varphi) + \alpha\varphi$, so L should be **SOS**. From Ξ and $\mathcal{S}^u$, it follows that not all of a_{20}, a_{11}, a_{02} are equal to 0. Suppose $a_{20} \neq 0$, then L has a monomial $-2a_{20}x_1^3$. Consider the value of L over the set $\{(\xi, 0) \mid a_{20}\xi > 0\}$, which will become negative when $|\xi|$ becomes large enough. Similarly, we can derive a contradiction when $a_{02} \neq 0$. If $a_{11} \neq 0$ and $a_{20} = a_{02} = 0$, it is easy to see that the degree of L is 3, which is impossible to be an **SOS**, that contradicts the assumption. □

The above example indicates that more BCs can be synthesized with **GBC**.

3.2. Combined barrier certificates

Given a CDS $\mathcal{D}$ equipped with D, Ξ and $\mathcal{S}^u$, suppose $\varphi(\mathbf{x})$ is a BC satisfying Theorem 1 w.r.t. $\psi(\mathbf{x})$. Clearly, $\{\mathbf{x} \mid \varphi(\mathbf{x}) \leq 0\}$ is an over-approximation of $\mathcal{R}_{\mathcal{D}}$, while $\{\mathbf{x} \mid \varphi(\mathbf{x}) > 0\}$ is an over-approximation of $\mathcal{S}^u$. In many cases we cannot find such a single BC to over-approximate the reachable set, but it can be achieved by combining several functions together. We call the combination of these functions a *combined BC*. Actually, a similar problem on differential invariants has been discussed in [20, 6, 28, 16]. In the literature, e.g. in [20], the combination of different differential invariants (BCs) is achieved in a trivial way, i.e., each of them satisfies BC condition separately. But here, we would like to consider deep combination that could be more useful in practice, i.e., none of which could satisfy BC condition, however, their combination can be used as a differential invariant whenever the given condition is satisfied.

Below, we discuss how to combine two functions together to form a combined BC. For easing discussion, let us fix the aforementioned CDS $\mathcal{D}$.

Lemma 3. $\{\mathbf{x} \mid \chi(\mathbf{x}) \leq 0\}$ is an over approximation of $\mathcal{R}_{\mathcal{D}}$ for a CDS $\mathcal{D}$ where $\chi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n]$, if there is a function $\psi_1(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}]$ such that

$$\forall \mathbf{x} \in \Xi.\ \chi(\mathbf{x}) \leq 0, \tag{13}$$

$$\forall \mathbf{x} \in D.\ \mathcal{L}_f\chi(\mathbf{x}) - \psi_1(\chi(\mathbf{x})) \leq 0, \tag{14}$$

$$\forall \xi.\, \xi > 0 \Rightarrow \theta(\mathbf{x}(\xi)) \leq 0, \text{ where } \theta(\mathbf{x}(t)) : \mathbb{R} \to \mathbb{R} \text{ is any function such that}$$

$$\begin{cases} \mathcal{L}_f\theta(\mathbf{x}) - \psi_1(\theta(\mathbf{x})) = 0, \\ \theta(\mathbf{x}(0)) \leq 0. \end{cases} \tag{15}$$

Proof. It can be proved similarly to Theorem 1. □

Lemma 4. If $\mathcal{A} = \{\mathbf{x} \mid \chi(\mathbf{x}) \leq 0\}$ is an over approximation of the reachable set $\mathcal{R}_{\mathcal{D}}$ and there are functions $\varphi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n], \psi_2(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}]$ such that

$$\forall \mathbf{x} \in \Xi.\ \varphi(\mathbf{x}) \leq 0, \tag{16}$$

$$\forall \mathbf{x} \in D \cap \mathcal{A}.\ \mathcal{L}_f\varphi(\mathbf{x}) - \psi_2(\varphi(\mathbf{x})) \leq 0, \tag{17}$$

$$\forall \mathbf{x} \in \mathcal{S}^u \cap \mathcal{A}.\ \varphi(\mathbf{x}) > 0, \tag{18}$$

$$\forall \xi.\, \xi > 0 \Rightarrow \theta(\mathbf{x}(\xi)) \leq 0, \text{ where } \theta(\mathbf{x}(t)) : \mathbb{R} \to \mathbb{R} \text{ is any function such that}$$

$$\begin{cases} \mathcal{L}_f\theta(\mathbf{x}) - \psi_2(\theta(\mathbf{x})) = 0, \\ \theta(\mathbf{x}(0)) \leq 0, \end{cases} \tag{19}$$

for any solution $\mathbf{x}(t)$ with initial value $\mathbf{x}(0)$, then $\mathcal{R}_{\mathcal{D}} \cap \mathcal{S}^u = \emptyset$, where $\varphi(\mathbf{x}), \chi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n]$ and $\psi_2(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}]$.

Proof. Since $\mathcal{A}$ is an over approximation of the reachable set $\mathcal{D}$, we just need to consider a new CDS whose vector field is the same as $\mathcal{D}$'s, but with the domain $D \cap \mathcal{A}$ and the unsafe set $\mathcal{S}^u \cap \mathcal{A}$. Thus, by Theorem 1, the claim is trivially true. □

Theorem 2. If there exist $\chi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n]$ and $\psi_1(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}]$ satisfying (13)-(15) and there exist $\varphi(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n], \psi_2(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}]$ satisfying (16)-(19), then $\mathcal{R}_{\mathcal{D}} \cap \mathcal{S}^u = \emptyset$.

Proof. It is straightforward by Lemmas 3 and 4. □

We will call the pair (χ, φ) a *combined BC*.

Obviously, a single BC defined in Theorem 1 can be seen as a specific combined BC by letting $\chi = 0$. In addition, actually, it is easy to prove that a combined BC forms a combined differential invariant.

Corollary 1. $\chi \leq 0 \wedge \varphi \leq 0$ in Theorem 2 is a *differential invariant* (the definition can be found in [20]) of $\mathcal{D}$, which can guarantee its safety.

The following example witnesses that the notion of combined BCs does give more power to the verification of CDSs as well as HSs.

Example 2. Consider the following CDS $\mathcal{D}_2$

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 - x_1x_2 \\ 2x_1^2 - x_2 \end{bmatrix}$$

with $\Xi = \{\mathbf{x} \in \mathbb{R}^2 \mid x_1^2 + (x_2 + 2)^2 \leq 1\}$ and $\mathcal{S}^u = \{\mathbf{x} \in \mathbb{R}^2 \mid x_1^2 + (x_2 - 1)^2 \leq 0.09\}$.

To prove its safety, by Theorem 2, we can synthesize a combined BC (χ, φ), see Fig. 2, in which $\chi(\mathbf{x}) = 0$ is denoted by the dash line and $\varphi(\mathbf{x}) = 0$ is denoted by the solid line (their mathematical representations can be found in the appendix). In fact, we can prove $\chi(\mathbf{x}) \leq 0 \wedge \varphi(\mathbf{x}) \leq 0$ is indeed a differential invariant according to the definition given in [20], which can guarantee the unsafe set unreachable.

Moreover, we can prove that neither $\chi(\mathbf{x})$ nor $\varphi(\mathbf{x})$ is a BC in the sense of Theorem 1. Furthermore, using the same values of α, β and the degree bound as used in synthesizing the combined BC $(\chi(\mathbf{x}), \varphi(\mathbf{x}))$, we cannot obtain any single BC in the sense of Theorem 1. □

From the above example, the reader may be still skeptical whether to synthesize a polynomial with higher degree which is a BC of the considered system. But Theorem 3 clearly indicates that the notion of combined BCs is strictly more expressive than that of BCs. To the end, we need to prove the following lemma first.

Lemma 5. Let $S_1 = x < 0 \wedge y < 0$, $S_2 = x > 1 \vee y > 1$. Then, there is no $f \in \mathbb{R}[x, y]$ such that $\forall (a, b) \in S_1.f(a, b) \geq 0$ and $\forall (a, b) \in S_2.f(a, b) < 0$.

Proof. Suppose there is a polynomial $f \in \mathbb{R}[x, y]$ which satisfies the condition. Obviously, $f \not\equiv 0$. Let $f(x, y) = \sum_{i=0}^{N} x^i h_i(y)$, where $h_i(y) \in \mathbb{R}[y]$, $h_N \not\equiv 0$. For each $i \in \mathbb{N}$, obviously, $f(-i, y)$ is continuous, $f(-i, y) < 0$ if $y > 1$, and $f(-i, y) \geq 0$ if $y < 0$. Thus, there is a $0 \leq b_i \leq 1$ such that $f(-i, b_i) = 0$ by the Mean Value Theorem. Thus, there is a subsequence $\{b_{i_k}\}$ such that $\lim_{k \to +\infty} b_{i_k} = \xi$ with $0 \leq \xi \leq 1$. Let $h(x) = f(x, \xi)$. $h(x)$ is a univariate polynomial, and $\lim_{x \to -\infty} h(x) = \lim_{i \to +\infty} h(-i) = 0$, so it follows $h(x) \equiv 0$. Hence, $h_i(\xi) = 0$ for $0 \leq i \leq N$, as $h(x) = f(x, \xi) = \sum_{i=0}^{N} x^i h_i(\xi) \equiv 0$. Thus,

$$f = \sum_{i=0}^{N} c_i x^i (y - \xi) k_i(y) = (y - \xi) \sum_{i=0}^{N} c_i x^i k_i(y),$$

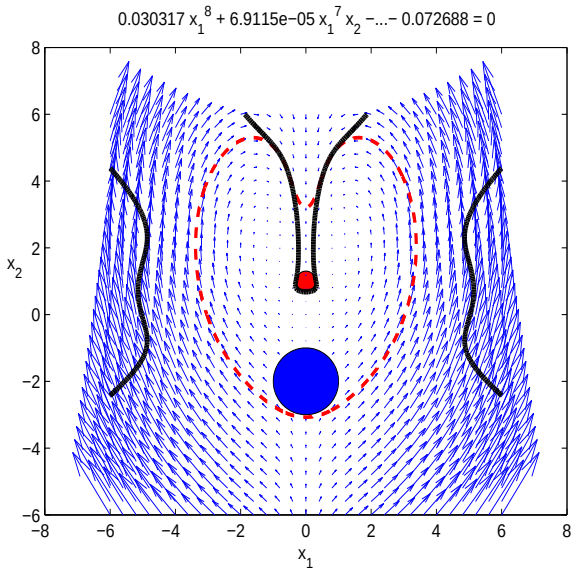

Fig. 2. A combined BC for Example 2

where $k_i(y)$ is a polynomial in $\mathbb{R}[y]$. It is easy to see that $(2,\xi) \in S_2$, then $f(2,\xi) < 0$, but $f(2,\xi) = \sum_{i=0}^{N} c_i 2^i(\xi - \xi)k_i(\xi) = 0$, which is a contradiction. □

Now, we can conclude that

Theorem 3. The notion of combined BCs is more expressive than that of BCs.

Proof. We prove this theorem by considering the following CDS $\mathcal{D}$:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -x \\ -y \end{bmatrix}$$

with $\Xi = \{(x,y) \in \mathbb{R}^2 \mid x < 0 \wedge y < 0\}$ and $\mathcal{S}^u = \{(x,y) \in \mathbb{R}^2 \mid x > 1 \vee y > 1\}$.

From Lemma 5 there is no polynomial which can separate the reachable set $\mathcal{R}_{\mathcal{D}}$ from the unsafe set $\mathcal{S}^u$, which means that there is no a BC in the sense of Theorem 1. On the other hand, let $\chi(x,y) = x$,$\varphi(x,y) = y$, $\psi(\theta) = -\theta + \theta^2$. It obtains $\mathcal{L}_f\chi(x,y) = -x$, $\mathcal{L}_f\varphi(x,y) = -y$. Thus, $\mathcal{L}_f\chi(x,y) - \psi(\chi(x,y)) = -x^2$. It's easy to check that the conditions (13)-(15) are satisfied. Similarly, it is easy to check that the conditions (16),(17) and (19) hold. Furthermore, from $\mathcal{S}^u \cap \mathcal{A} = \{(x,y) \mid x \leq 0 \wedge y > 1\}$ and $\varphi(x,y) = y$, it follows that (18) holds. Thus there exists a *combined BC*.

In fact, for any given $0 \leq a, b < 1$, the pair of $\chi(x,y) = x - a$ and $\varphi(x,y) = y - b$ is a *combined BC*. □

3.3. Relaxed barrier certificate conditions for HSs

As discussed in [11], the principle of the condition of BCs $\Phi(\mathbf{x})$ for an HS $\mathcal{H} = (Q, X, f, D, E, G, R, \Xi)$ w.r.t. a given unsafe set S^u should satisfy the following conditions:

- $\Phi(\mathbf{x})$ consists of a set of functions $\{\varphi_q(\mathbf{x}) \mid q \in Q\}$, each $\varphi_q(\mathbf{x})$ is a BC for CDS $\dot{\mathbf{x}} = \mathbf{f}_q$ equipped with the domain D_q, initial set Ξ_q and unsafe set S_q^u;
- all the discrete transitions starting from every mode $q \in Q$ have to be taken into account in the BC condition so that $\Phi(\mathbf{x})$ can construct a global inductive invariant of $\mathcal{H}$.

Based on the discussions about BC conditions for CDSs as well as the above principle, we can accordingly revisit the condition of BCs for HSs by the following theorem:

Theorem 4. Given an HS $\mathcal{H} = (Q, X, f, D, E, G, R, \Xi)$ and an unsafe set S^u, if there exists a set of non-negative real numbers $\{c_e \mid e \in E\}$, and a set of functions $\{\varphi_q(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n] \mid q \in Q\} \cup \{\psi_q(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}] \mid q \in Q\}$ s.t.

$$\forall q \in Q \;\; \forall \mathbf{x} \in \Xi_q.\; \varphi_q(\mathbf{x}) \leq 0, \tag{20}$$

$$\forall q \in Q \;\; \forall \mathbf{x} \in D_q.\; \mathcal{L}_{\mathbf{f}_q}\varphi_q(\mathbf{x}) - \psi_q(\varphi_q(\mathbf{x})) \leq 0, \tag{21}$$

$$\forall q \in Q \;\; \forall \mathbf{x} \in S_q^u.\; \varphi_q(\mathbf{x}) > 0, \tag{22}$$

$$\begin{aligned} &\forall q \in Q \;\; \forall \xi.\; \xi > 0 \Rightarrow \theta_q(\mathbf{x}(\xi)) \leq 0, \\ &\qquad \text{where } \theta_q(\mathbf{x}(t)) : \mathbb{R} \to \mathbb{R} \text{ is any function such that} \\ &\begin{cases} \mathcal{L}_{\mathbf{f}_q}\theta_q(\mathbf{x}) - \psi_q(\theta_q(\mathbf{x})) = 0, \\ \theta_q(\mathbf{x}(0)) \leq 0, \end{cases} \end{aligned} \tag{23}$$

$$\forall e \in E \;\; \forall \mathbf{x} \in G(e).\;\; \mathbf{x}' = R(e)(\mathbf{x}).\; c_e\varphi_{S(e)}(\mathbf{x}) - \varphi_{T(e)}(\mathbf{x}') \geq 0, \tag{24}$$

then $\mathcal{R}_\mathcal{H} \cap S^u = \emptyset$, where $S(e)$ and $T(e)$ stand for the source and target modes of jump e, respectively.

Proof. Suppose $\mathcal{R}_\mathcal{H} \cap S^u \neq \emptyset$, then there must exist a hybrid trajectory (τ, α, β), where $\tau = \{[t_i, t_{i+1}]\}_{i=0}^{m-1}$, $\alpha = (q_1, q_2, \cdots, q_m)$, where $t_0 = 0$ and $t_m = T$, s.t. $\beta_{q_1}(t_0) \in \Xi_{q_1}$ and $\beta_m(T) \in S_{q_m}^u$. Since $\beta_{q_1}(0) \in \Xi_{q_1}$, we have $\varphi_{q_1}(\beta_{q_1}(0)) \leq 0$ from condition (20). So, it is easy to know $\varphi_{q_1}(\beta_{q_1}(t_1)) \leq 0$ from condition (21) and (23). Moreover, from condition (24), it follows $\varphi_{q_2}(\beta_{q_2}(t_1)) \leq c_e\varphi_{q_1}(\beta_{q_1}(t_1)) \leq 0$, i.e. $\varphi_{q_2}(\beta_{q_2}(t_1)) \leq 0$. Repeating the above arguments, we can get $\varphi_{q_m}(\beta_{q_m}(T)) \leq 0$. On the other hand, $\varphi_{q_m}(\beta_{q_m}(T)) > 0$ because $\beta_{q_m}(T) \in S_{q_m}^u$ from the assumption, which is a contradiction. □

Similarly, based on Theorem 2 and Theorem 4, we can define the condition of combined BCs for HSs as follows:

Theorem 5. Given an HS $\mathcal{H} = (Q, X, f, D, E, G, R, \Xi)$ and an unsafe set S^u, if there exists a set of non-negative real numbers $\{c_{e,1}, c_{e,2}, c_{e,3}, c_{e,4} \mid e \in E\}$, a set of **SOS** polynomials $\{\delta_q \mid q \in Q\}$, and a set of functions $\{\varphi_q(\mathbf{x}), \chi_q(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}^n] \mid q \in Q\} \cup$

$\{\psi_{q,1}(\mathbf{x}), \psi_{q,2}(\mathbf{x}) \in \mathcal{C}^\omega[\mathbb{R}] \mid q \in Q\}$ s.t.

$$\forall q \in Q \;\; \forall \mathbf{x} \in \Xi_q.\ \chi_q(\mathbf{x}) \leq 0, \tag{25}$$

$$\forall q \in Q \;\; \forall \mathbf{x} \in D_q.\ \mathcal{L}_{\mathbf{f}_q}\chi_q(\mathbf{x}) - \psi_{q,1}(\chi_q(\mathbf{x})) \leq 0, \tag{26}$$

$$\forall q \in Q \;\; \forall \mathbf{x} \in \Xi_q.\ \varphi_q(\mathbf{x}) \leq 0, \tag{27}$$

$$\forall q \in Q \;\; \forall \mathbf{x} \in D_q.\ \mathcal{L}_{\mathbf{f}_q}\varphi_q(\mathbf{x}) - \psi_{q,2}(\varphi_q(\mathbf{x})) - \delta_q\chi_q \leq 0, \tag{28}$$

$$\forall q \in Q \;\; \forall \mathbf{x} \in S^u_q.\ \varphi_q(\mathbf{x}) > 0, \tag{29}$$

$$\forall q \in Q \;\; \forall \xi.\ \xi > 0 \Rightarrow \theta_q(\mathbf{x}(\xi)) \leq 0,$$
where $\theta_q(\mathbf{x}(t)) : \mathbb{R} \to \mathbb{R}$ is any function such that
$$\begin{cases} \mathcal{L}_{\mathbf{f}_q}\theta_q(\mathbf{x}) - \psi_{q,1}(\theta_q(\mathbf{x})) = 0, \\ \theta_q(\mathbf{x}(0)) \leq 0, \end{cases} \tag{30}$$

$$\forall q \in Q \;\; \forall \xi.\ \xi > 0 \Rightarrow \theta'_q(\mathbf{x}(\xi)) \leq 0,$$
where $\theta'_q(\mathbf{x}(t)) : \mathbb{R} \to \mathbb{R}$ is any function such that
$$\begin{cases} \mathcal{L}_{\mathbf{f}_q}\theta_q(\mathbf{x}) - \psi_{q,2}(\theta'_q(\mathbf{x})) = 0, \\ \theta'_q(\mathbf{x}(0)) \leq 0, \end{cases} \tag{31}$$

$$\forall e \in E \;\; \forall \mathbf{x} \in G(e).\ \mathbf{x}' = R(e)(\mathbf{x}) \Rightarrow c_{e,1}\varphi_{S(e)}(\mathbf{x}) + c_{e,3}\chi_{S(e)}(\mathbf{x}) - \varphi_{T(e)}(\mathbf{x}') \geq 0, \tag{32}$$

$$\forall e \in E \;\; \forall \mathbf{x} \in G(e).\ \mathbf{x}' = R(e)(\mathbf{x}) \Rightarrow c_{e,2}\varphi_{S(e)}(\mathbf{x}) + c_{e,4}\chi_{S(e)}(\mathbf{x}) - \chi_{T(e)}(\mathbf{x}') \geq 0, \tag{33}$$

then $\mathcal{R}_{\mathcal{H}} \cap S^u = \emptyset$, where $S(e)$ and $T(e)$ stand for the source and target modes of the jump e, respectively.

Proof. The proof is similar to Theorem 4. □

4. Discovering Relaxed Barrier Certificates by SDP

Theorems 1&2 (resp. Theorems 4&5) provide relaxed conditions which can guarantee a function (a pair of functions) to be a (combined) BC for a CDS (resp. an HS), but these theorems do not provide any constructive method for synthesizing (combined) BCs. However, like in the previous work e.g. [9, 22, 23, 30, 11], (combined) BCs can be synthesized in a standard way from these conditions by exploiting SDP techniques [18, 19]. For self-containedness, we introduce SDP and the synthesis algorithm. But we mainly emphasize on how to apply symbolic checking to avoid the unsoundness of these approaches based on SDP because of numerical errors. In the literature, some attempts by using hybrid numeric-symbolic method to avoid the unsoundness of SDP have been investigated due to Wu, Yang et al. [32, 14, 33].

4.1. Semi-definite programming

We use Sym_n to denote the set of $n \times n$ real symmetric matrices, and $deg(f)$ the highest total degree of a given polynomial f.

Definition 5 ((Positive) semidefinite matrices)**.** A matrix $M \in Sym_n$ is called *positive definite (semidefinite)*, denoted by $M \succ 0$ ($M \succeq 0$), if $\mathbf{x}^T M\mathbf{x} > 0$ ($\mathbf{x}^T M\mathbf{x} \geq 0$) for all $\mathbf{x} \in \mathbb{R}^n$.

Definition 6 (Inner product)**.** The *inner product* of two matrices $A = (a_{ij}), B = (b_{ij}) \in \mathbb{R}^{n\times n}$, denoted by $\langle A, B\rangle$, is defined by $Tr(A^T B) = \sum_{i,j=1}^{n} a_{ij}b_{ij}$.

Definition 7 (Semidefinite programming (SDP))**.** The standard (primal) and dual forms of a SDP are respectively given in the following:

$$p^* = \inf_{X\in Sym_n} \langle C, X\rangle \text{ s.t. } X \succeq 0,\ \langle A_j, X\rangle = b_j \quad (j = 1,\ldots,m) \tag{34}$$

$$d^* = \sup_{y\in\mathbb{R}^m} \mathbf{b}^T\mathbf{y} \text{ s.t. } \sum_{j=1}^{m} y_j A_j + S = C,\ S \succeq 0, \tag{35}$$

where $C, A_1, \ldots, A_m, S \in Sym_n$ and $\mathbf{b} \in \mathbb{R}^m$.

There are many efficient algorithms to solve SDP such as interior-point methods. We present a basic path-following algorithm to solve (34) in Algorithm 1.

Definition 8 (Interior point for SDP)**.**

$$intF_p = \{X : \langle A_i, X\rangle = b_i\ (i = 1,\ldots,m),\ X \succ 0\},$$
$$intF_d = \left\{(\mathbf{y}, S) : S = C - \sum_{i=1}^{m} A_i y_i \succ 0\right\},$$
$$intF = intF_p \times intF_d.$$

Obviously, $\langle C, X\rangle - \mathbf{b}^T\mathbf{y} = \langle X, S\rangle \geq 0$ for all $(X, \mathbf{y}, S) \in intF$. Especially, we have $d^* \leq p^*$. So the soul of interior-point methods to compute p^* is to reduce $\langle X, S\rangle$ incessantly and meanwhile guarantee $(X, \mathbf{y}, S) \in intF$.

Algorithm 1: `Interior_Point_Method`

input : C, A_j, b_j $(j = 1,\ldots,m)$ as in (34) and a threshold c
output: p^*

1 Given a $(X, \mathbf{y}, S) \in intF$ with $XS = \mu I$;
`/* ` μ `is a positive constant and` I `is the identity matrix. */`
2 **while** $\mu > c$ **do**
3 $\quad \mu = \gamma\mu$;
`/* ` γ `is a fixed positive constant less than one */`
4 $\quad$ use **Newton iteration** to solve $(X, \mathbf{y}, S) \in intF$ with $XS = \mu I$;
5 **end**

4.2. Symbolic checking

It should be noted that because of the error caused by numeric computation in SDP, in particular, a threshold c upon which SDP depends, it may happen that the (combined) BCs computed by SDP are not real ones, or some real (combined) certificates satisfying

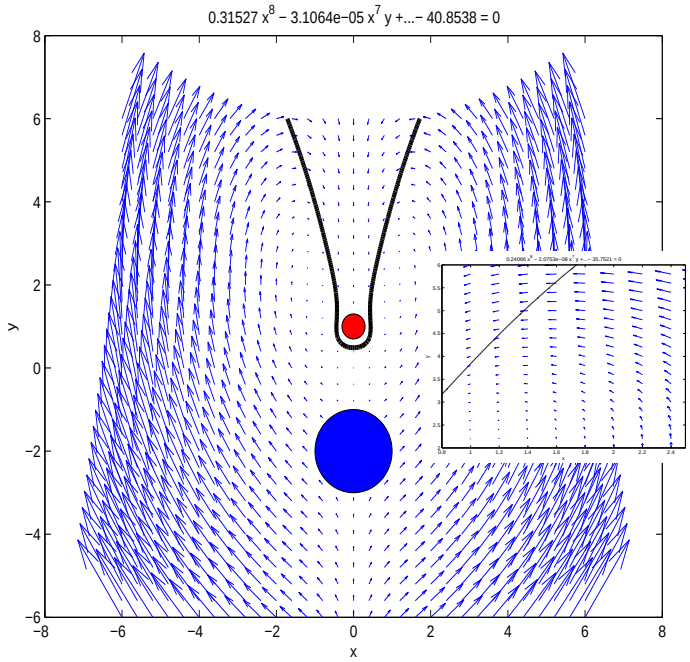

Fig. 3. A false BC of Example 2 due to numerical errors

the condition cannot be computed or are determined as false ones. For example, considering Example 2, if we encode the condition derived from Theorem 1 as an SDP, then call SOSTOOLS [2] [24], and obtain the output is:

" feasratio: 1.0000; pinf: 0; dinf: 0; numerr: 0".

This indicates that the tool does discover a BC. However, after showing the result in Fig. 3, it is easy to find that the black line in Fig. 3 does not satisfy condition (3), as some vectors cross it into the area which contains unsafe set.

So, we have to take the numerical error into account when using these SDP tools. Our experience is:

- The larger the size of matrix X is, the larger the error due to SDP is. Thus, it is more likely to obtain a false (combined) BC;
- The higher the degree of undetermined polynomials as predefined templates of BCs is, the larger the error due to SDP is;
- The degree of a combined BC synthesized by Theorem 2 is normally lower than that of a standard BC by Theorem 1.

It is absolutely necessary to guarantee the soundness of the approaches to the verification of HSs. But the approaches based on SDP to synthesizing (combined) BCs according to these relaxed conditions may be unsound because of the error caused by numeric computation. Below, we advocate to apply symbolic computation techniques to check if the synthesized (combined) BCs are real ones, which is hinted by our previous work [3].

[2] SOSTOOLS is of version v2.04 with MATLAB R2011b.

Problem 1. For $f \in \mathbb{R}[\mathbf{x}]$, if $\forall \mathbf{x} \in \mathbb{R}^n . f(\mathbf{x}) \geq 0$?

Checking the constraints in Theorems 1&2&4&5 are obviously instances of Problem 1. A lot of work has been done on Problem 1. We proposed an exact method based on an improved Cylindrical Algebraic Decomposition(CAD) algorithm in [7], and implemented a tool called `CADpsd` for the checking. The `CADpsd` returns **True** when the input polynomial is positive semidefinite and **False** otherwise.

Remark 3. One may doubt the efficiency of the above symbolic checking since the complexity of CAD is $O(2^{2^n})$ in general, where n is the number of variables. However, please note that Problem 1 is a special case of quantifier elimination. One of the main contributions of [7] is an improved algorithm for solving Problem 1. Although the improved algorithm cannot be proved with a lower complexity theoretically, it has been shown that it does avoid many heavy resultant computation. So, in practice, especially in the case where the number of variables is greater than 2, `CADpsd` is much faster than any general CAD tool. Please see [7] for the detail. In our experience, `CADpsd` can finish checking in few seconds when $deg(f)$ is no larger than 6 and the number of variables in f is less than 5, which is enough for many problems.

4.3. Algorithms

Now, we can sketch the basic steps of the algorithm to construct (combined) BCs using SDP as follows:

Step 1: predefine parametric polynomial templates with a degree bound as possible candidates of (combined) BCs;
Step 2: derive constraints on the parameters of these parametric polynomial templates according to the considered relaxed barrier condition;
Step 3: reduce all the constraints on the parameters to an SDP;
Step 4 apply some SDP solver to solve the resulting SDP problem and obtain instantiations of these parameters.

In the above procedure, for most of the constraints on parameters, we only need to consider how to reduce $p \geq 0$ ($p \leq 0$) to $p = \delta$ ($-p = \delta$), where p is a polynomial and δ is an undetermined **SOS** polynomial. In the literature, there is a lot of work on this, please refer to [9, 22, 23, 30, 11, 3] for the detail.

The hardest part is how to reduce the constraints that contain ψ, χ, ψ_q, $\psi_{q,i}$, or χ_q, as they may contain the product of two or more parametric polynomials after replacement, which result in non-linear expressions on parameters, that cannot be seen as an SDP any more. For instance, let $\psi = \theta + \theta^2$, and $\theta = ax_1 + bx_2$ be a template of BCs. By Theorem 1, the constraint derived from condition (3) will contain expression $(ax_1+bx_2)+(ax_1+bx_2)^2$, which cannot be reduced to an SDP directly.

To address this issue, we explore the iterative approach proposed in [22] which can handle a constraint containing the product of two parametric polynomials. Thus, we implemented Algorithm 2 below based on the idea of the iterative approach for dealing with the following problem.

Problem 2. Suppose $\Xi, \mathcal{S}^u, \mathbf{f}, \psi$ are given, where ψ satisfies (5). Our goal is to find a φ which satisfies (2)- (4).

Algorithm 2: `Iterative Algorithm for Problem 2`

input : $\Xi, \mathcal{S}^u, \mathbf{f}, \psi(\theta) = \sum_{i=0}^{s} a_i \theta^i$, where $\psi(\theta)$ satisfies (5) and an iteration step number N

output: θ' a possible solution for the Problem 2

$\theta' = 0$;
$j = 0$;
while $j \leq N$ **do**
  $\psi' = \sum_{i=0}^{s} a_i \theta \theta'^{i-1}$;
  Use an SDP tool to solve the resulting Problem 2 by replacing ψ with ψ';
  if *the SDP solver returns a result* **then**
    Denote the result by θ';
  end
  else
    break;
  end
  $j = j + 1$;
end
return θ';

The basic idea of Algorithm 2 is as follows: In order to avoid occurrences of non-linear expressions in the parameters contained in the template of θ, which is caused as ψ contains non-linear expressions in θ, we first initialize $\theta' = 0$, and use $\psi' = \sum_{i=0}^{s} a_i \theta \theta'^{i-1}$ to approximate ψ. Thus, ψ' has no non-linear expression in θ any more. Then we can call an SDP solver to the resulted SDP problem. If the solver gives an answer, we repeat the procedure by refining the approximation of ψ' in order to obtain a more likely correct solution to **Problem 2**. The refining procedure (the while loop) terminates whenever either the number of iteration reaches the upper bound N or no solution to **Problem 2** can be found using the refined ψ'. For example, in case the while loop terminates just after the first iteration, the returned solution is $\theta' = 0$, which is impossible to be a solution to **Problem 2**.

5. Beyond polynomials and with inputs

In this section, we discuss how to extend our approach to more general systems including non-polynomial vector fields and/or CDSs with inputs.

5.1. Beyond polynomials

Hybrid systems with non-polynomial functions such as reciprocal function $\frac{1}{x}$, exponential function e^x, logarithm function $\ln(x)$, trigonometric functions $\sin(x)$ and $\cos(x)$, and their compositions are very common in practice, e.g., in [35], a real-world example of a lunar lander is given. How to verify such systems is a challenge. By combining with the approach in [17], it is easy to extend our approach to the verification of these systems. We will show the idea by the following example.

Example 3. Consider a CDS $\mathcal{D}_3$ with elementary function as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \exp^{-x^2} + y - 1 \\ -\sin^2(x) - 1 \end{bmatrix} \tag{36}$$

with $\Xi = \{(x,y) \in \mathbb{R}^2 | (x+1)^2 + y^2 - 0.09 \leq 0\}$, $D = \{(x,y) \in \mathbb{R}^2 | -2 \leq x \leq 2 \wedge -2 \leq y \leq 2\}$, $S^u = \{(x,y) \in \mathbb{R}^2 | (x-1.5)^2 + (y+1)^2 - 0.09 \leq 0\}$. This is no longer a semialgebraic system as the ODE contains transcendental expressions $\exp^{-x^2}$ and $\sin^2(x)$. According to the approach given in [17], we can abstract $\mathcal{D}_3$ to a polynomial one $\widetilde{\mathcal{D}}_3$ by variable transformation, so that the safety verification of $\mathcal{D}_3$ is reduced to the corresponding one of $\widetilde{\mathcal{D}}_3$, via the following two steps:

Step 1: Reduction of an elementary ODE to a polynomial ODE by variable transformation. In this example, for e^{-x^2} and $sin^2(x)$, we respectively introduce two fresh variables v_1 and v_2, and establish a replacement equation

$$\begin{cases} v_1 = \exp^{-x^2}, \\ v_2 = \sin(x). \end{cases} \tag{37}$$

Then, to differentiate the two sides of (37) and simplify it, we obtain

$$\begin{cases} \dot{v_1} = -2x \exp^{-x^2} \dot{x} = -2xv_1\dot{x} = -2xv_1(v_1 + y - 1), \\ \dot{v_2} = \cos(x)\dot{x}, \end{cases} \tag{38}$$

which still contains non-polynomial expression $\cos(x)$. So, we have to introduce another fresh variable v_3, and put $v_3 = \cos(x)$ into the replacement equation and obtain

$$\begin{cases} v_1 = \exp^{-x^2}, \\ v_2 = \sin(x), \\ v_3 = \cos(x). \end{cases} \tag{39}$$

To differentiate the two sides of (39) and simplify it again, we obtain a new additional ODE, which only contains polynomial expressions. By merging the new ODE into that of $\mathcal{D}_3$, we establish the vector field of $\widetilde{\mathcal{D}}_3$ as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v_1} \\ \dot{v_2} \\ \dot{v_3} \end{bmatrix} = \begin{bmatrix} v_1 + y - 1 \\ v_2^2 - 1 \\ -2xv_1(v_1 + y - 1) \\ v_3(v_1 + y - 1) \\ -v_2(v_1 + y - 1) \end{bmatrix}. \tag{40}$$

Step 2: Deriving the initial, domain and unsafe sets of $\widetilde{\mathcal{D}}_3$ from the respective initial, domain and unsafe sets of $\mathcal{D}_3$ by using the replacement equation. In this example, we

can derive the following additional constraints on v_1, v_2, v_3 from known ranges of the functions, trigonometric relations and Taylor expansions:

$$const(v_1, v_2, v_3) = 0 \leq v_1 \leq 1 \wedge -1 \leq v_2 \leq 1 \wedge -1 \leq v_3 \leq 1 \wedge x^2 - \frac{1}{6}x^4 \leq xv_2 \leq x^2 \\ \wedge 1 - \frac{1}{2}x^2 \leq v_3 \leq 1 - \frac{1}{2}x^2 + \frac{1}{24}x^4 \wedge 1 - x^2 \leq v_1 \leq 1 - x^2 + \frac{1}{2}x^4.$$

Thus, the initial, domain and unsafe sets of $\widetilde{\mathcal{D}_3}$ can be given as follow:

$$\widetilde{\Xi_0} = \{(x, y, v_1, v_2, v_3) \in \mathbb{R}^5 \mid \ const(v_1, v_2, v_3) \wedge (x, y) \in \Xi_0\},$$
$$\widetilde{D} = \{(x, y, v_1, v_2, v_3) \in \mathbb{R}^5 \mid \ const(v_1, v_2, v_3) \wedge (x, y) \in D\},$$
$$\widetilde{S^u} = \{(x, y, v_1, v_2, v_3) \in \mathbb{R}^5 \mid \ const(v_1, v_2, v_3) \wedge (x, y) \in S^u\}.$$

To verify $\mathcal{R}_{\widetilde{\mathcal{D}_3}} \cap \widetilde{S^u} = \emptyset$, we apply the results reported in the previous sections and obtain a BC $\varphi = 0.32873x^3 + 0.74066x^2y - 1.5285x^2 - 1.8105xy^2 - 2.3281xy + 5.7603x + 3.0577y^3 + 8.9834y^2 + 7.193y - 3.1476$. On the other hand, according to the results in [17], we have that $\varphi(x, y, v_1, v_2, v_3) \in \mathbb{R}[x, y, v_1, v_2, v_3]$ is a BC of $\widetilde{\mathcal{D}_3}$ implies that $\varphi(x, y, \exp^{-x^2}, \sin(x), \cos(x))$ is a BC of $\mathcal{D}_3$, which guarantees the safety of $\mathcal{D}_3$.

Obviously, the above idea can be easily extended to hybrid systems with non-polynomial expressions.

5.2. With inputs

In the previous subsection, we discussed how to extend our approach to elementary systems. In this subsection, we shall argue that the same idea can be applied to extend our approach to systems with inputs. Here, we just consider CDSs with inputs, as hybrid systems with inputs can be handled similarly. Actually, a similar idea has been used in synthesizing the standard BCs for CDSs with inputs, see [30].

Definition 9. A CDS with input can be represented by the following form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}),$$

where $\mathbf{x} \in \mathbb{R}^n$ is a vector of variables, $\mathbf{u} \in U \subseteq \mathbb{R}^m$ is an input, and $\mathbf{f} \in \mathbb{R}[\mathbf{x}, \mathbf{u}]$ is a vector of polynomials over $\mathbf{x}, \mathbf{u}$, in which, $U \subseteq \mathbb{R}^m$ is called *an input set*, which is compact.

Suppose $\mathbf{u}(t) = (u_1(t), \cdots, u_m(t))$, then for every $u_i(t)$ there is a lower bound ul_i and a upper bound up_i since U is a compact set. Then regard each component of $\mathbf{u}$ as a fresh variable, and the constraint for u_i, $i = 1, \cdots, m$ is $ul_i \leq u_i \leq up_i$. Thus, the problem of relaxed BCs of a CDS with inputs is naturally reduced to that of an autonomous CDS.

6. Experimental Results

In this section, we demonstrate our approach with some examples.

Example 4 (Adapted from [12])**.** Consider a CDS $\mathcal{D}_4$ as follows:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 - x_1x_2 \\ 2x_1^2 - x_2 \end{bmatrix}$$

with $\Xi = \{\mathbf{x} \in \mathbb{R}^2 \mid x_1^2 + (x_2+2)^2 \leq 1\}$ and $\mathcal{S}^u = \{\mathbf{x} \in \mathbb{R}^2 \mid x_1^2 + (x_2 - 5.2)^2 \leq 0.81\}$.

No polynomial BCs can be discovered using the existing approaches in the verification of $\mathcal{D}_4$, except for the one in [12] with which a polynomial BC with degree 8 was discovered. By setting $\alpha = -4$ and $\beta = 1.5$, using the corresponding relaxed condition **GBC**, it is easy to synthesize a polynomial barrier certificate with degree 6, see Fig. 4 (also see the appendix for its mathematical representation). Furthermore, let $\beta = 0$, the relaxed condition is degenerated to the case considered in [12]. But unfortunately, we can not synthesize any BC from the conditions, see Fig. 5. □

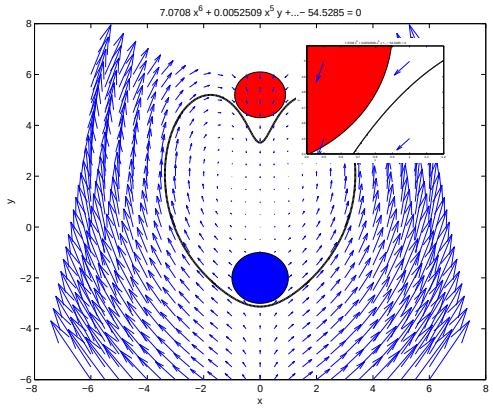

Fig. 4. $\alpha = -4, \beta = 1.5$

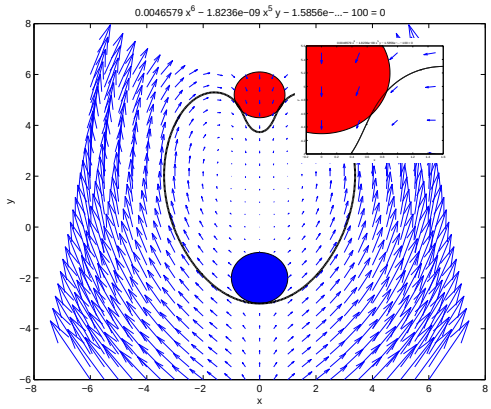

Fig. 5. $\alpha = -4, \beta = 0$

Example 5. Consider the following CDS $\mathcal{D}_5$

$$\begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = 2x_1 - x_2 - x_1^2x_2 - x_1^3 \end{cases}$$

with $\Xi = \{x \in \mathbb{R}^2 \mid (x_1+1)^2+(x_2-2)^2 \leq 0.16\}$ and $\mathcal{S}^u = \{x \in \mathbb{R}^2 \mid (x_1-1)^2+x_2^2 \leq 0.04\}$. Let $g_0 = 0.16 - (x_1+1)^2 - (x_2-2)^2, g_1 = 0.04 - (x_1-1)^2 - x_2^2$. In order to prove $\mathcal{R}_{\mathcal{D}_5} \cap \mathcal{S}^u = \emptyset$, according to Theorem 1, using the above procedure, we can obtain the following polynomials:

$$
\begin{aligned}
\varphi &= -0.91253x_1^2 + 0.40176x_1x_2 + 1.3603x_1 + 0.13922x_2^2 - 1.0308x_2 - 0.27657,\\
\chi &= 0.19394x_1^4 + 0.29363x_1^3x_2 - 0.1696x_1^3 + 0.091674x_1^2x_2^2 - 0.2317x_1^2x_2 - 1.3805x_1^2\\
&\quad +0.056453x_1x_2^3 - 0.14904x_1x_2^2 + 0.096278x_1x_2 + 1.7932x_1 + 0.070488x_2^4\\
&\quad -0.063002x_2^3 + 0.48804x_2^2 - 1.1726x_2 - 0.38201\\
\delta &= 0.1956x_1^4 + 0.23674x_1^3x_2 - 0.13109x_1^3 + 0.14603x_1^2x_2^2 - 0.16935x_1^2x_2 + 1.0686x_1^2\\
&\quad +0.35005x_1x_2^3 - 0.29307x_1x_2^2 - 0.5897x_1x_2 - 1.8943x_1 + 0.26073x_2^4 - 0.23047x_2^3\\
&\quad +0.027813x_2^2 + 0.64131x_2 + 1.7118,\\
u_1 &= 0.47292x_1^4 + 0.03761x_1^3x_2 - 0.15676x_1^3 + 0.45935x_1^2x_2^2 + 0.13126x_1^2x_2 + 0.26007x_1^2\\
&\quad +0.0766x_1x_2^3 - 0.02395x_1x_2^2 + 0.045239x_1x_2 + 0.068505x_1 + 0.33983x_2^4 + 0.17729x_2^3\\
&\quad +0.4338x_2^2 + 0.054172x_2 + 0.37428\\
u_2 &= 0.45008x_1^4 + 0.0064431x_1^3x_2 - 0.14066x_1^3 + 0.48519x_1^2x_2^2 + 0.18081x_1^2x_2 + 0.31882x_1^2\\
&\quad +0.045636x_1x_2^3 - 0.030792x_1x_2^2 + 0.0463x_1x_2 + 0.022898x_1 + 0.3829x_2^4 + 0.24085x_2^3\\
&\quad +0.48187x_2^2 + 0.10909x_2 + 0.37734\\
u_3 &= 0.5497x_1^4 - 0.035471x_1^3x_2 + 0.073809x_1^3 + 0.66023x_1^2x_2^2 - 0.085302x_1^2x_2 + 0.34888x_1^2\\
&\quad -0.020016x_1x_2^3 + 0.55526x_1x_2^2 + 0.032773x_1x_2 - 0.10637x_1 + 0.81332x_2^4 - 0.055596x_2^3\\
&\quad +0.49761x_2^2 + 0.25765x_2 + 0.93038\\
\psi_1(\theta) &= \psi_2(\theta) = -4\theta + 2\theta^2,
\end{aligned}
$$

where δ,u_1,u_2, $u_3-\chi-u_1g_0$, $-\mathcal{L}_f(\chi)+\psi_1(\chi)$, $-\varphi-u_2g_0$, $-\mathcal{L}_f(\varphi)+\psi_2(\varphi)+\delta\chi$, $\varphi-u_2g_1$ are positive polynomials. □

Example 6. Consider an HS with two modes in Fig. 6, in which the CDSs at q_1 and q_2 are respectively $\dot{\mathbf{x}} = \mathbf{f}_1(\mathbf{x})$ and $\dot{\mathbf{x}} = \mathbf{f}_2(\mathbf{x})$, where

$$
\mathbf{f}_1(\mathbf{x}) = \begin{cases} x_2 \\ -x_1 - x_3 \\ x_1 + (2x_2 + 3x_3)(1 + x_3^2), \end{cases} \qquad \mathbf{f}_2(\mathbf{x}) = \begin{cases} x_2 \\ -x_1 - x_3 \\ -x_1 - 2x_2 - 3x_3, \end{cases}
$$

$\Xi_{q_1} = \{\mathbf{x} \in \mathbb{R}^3 \mid x_1^2 + x_2^2 + x_3^2 \leq 0.01\}$, $\Xi_{q_2} = \emptyset$, $D_{q_1} = x_1^2 + 0.01x_2^2 + 0.01x_3^2 \leq 1.01$, $D_{q_2} = x_1^2 + x_2^2 + x_3^2 \geq 0.03 \wedge x_1^2 \leq 5.1^2$, $g_1 = 0.99 \leq x_1^2 + 0.01x_2^2 + 0.01x_3^2 \leq 1.01$, and $g_2 = 0.03 \leq x_1^2 + x_2^2 + x_3^2 \leq 0.05$. All resets are identity.

The proof obligation is to verify $|x_1| \leq 3.2$ at q_2. To the end, we synthesize BCs at each mode first (their mathematical representations are given in the appendix). Setting $\psi(\theta) = -0.2\theta + \theta^2$, we need to verify the following five conditions :

$$
\begin{cases}
c_1 = -\varphi_2 - u_{23}g_{11} - u_{24}g_{12} \geq 0, \\
c_2 = -\chi_2 - u_{21}g_{11} - u_{22}g_{12} \geq 0, \\
c_3 = -\mathcal{L}_f(\chi_2) + \psi(\chi_2) - u_{41}D_2 - u_{41}D_{21} \geq 0, \\
c_4 = -\mathcal{L}_f(\varphi_2) + \psi(\varphi_2) - \delta_2\chi_2 - u_{51}D_2 - u_{52}D_{21} \geq 0, \\
c_5 = \varphi_2 - U_2 - 0.00001 \geq 0,
\end{cases}
$$

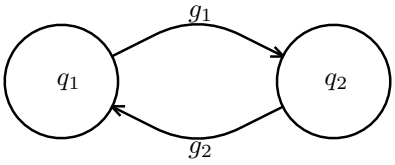

Fig. 6. An HS with two modes

	Exp. cond.		Our method		
	Degree	Time(s)	Degree	Time(s)	
				Synthesis	Symb. checking
E.g. 2	×	×	8	36.023	10.766
E.g. 4	8	1.132	6	2.717	0.226
E.g. 5	6	1.516	4	4.658	0.180
E.g. 6	4	1.387	2	4.260	20.472

Fig. 7. Experimental data.

by SDP. In which, $g_{11} = 1.01 - x_1^2 - 0.01x_2^2 - 0.01x_3^2$, $g_{12} = x_1^2 + 0.01x_2^2 + 0.01x_3^2 - 0.99$, $D_2 = x_1^2 + x_2^2 + x_3^2 - 0.03$, $D_{21} = 26.01 - x_1^2$, $U_2 = x_1^2 - 10.24$, and u_{21}, u_{22}, u_{23}, u_{24}, u_{41}, u_{42}, u_{51}, u_{52}, δ_2 are **SOS** synthesized in the first step. □

All the experimental results on all examples given in this paper are summarised as in Table 7, in which the label × means that no BC can be synthesized with the corresponding method. All the results listed are computed on a 64-bit Intel(R) Core(TM) i5 CPU 650 @ 3.20GHz with 4GB RAM memory and Ubuntu 12.04 GNU/Linux.

By comparing with the approach reported in [11] (see Table 7), our approach can synthesize more BCs, in particular, with lower degree, but our approach takes a little more time. However, our approach is still very efficient. In addition, symbolic checking can make our approach to avoid unsoundness caused by the error due to numeric computation in SDP.

7. Concluding Remarks

To summarize, the contributions of this paper include:

- relaxation of the condition of BCs in a general way, so that one can utilize weaker conditions flexibly to synthesize various kinds of BCs with more expressiveness, which gives more opportunities to verify the considered system;
- a method to combining two functions together to form a combined BC in order to prove a safety property under consideration, whereas neither of them could be used as a BC separately;

- an approach to synthesizing certificates according to the general relaxed condition by SDP. In particular, we discussed how to apply symbolic checking to avoid the unsoundness of our approach caused by the error of numeric computation in SDP;
- extension of our approach to non-polynomials and/or with inputs;
- experimental results indicating that our approach can indeed discover more certificates and give more opportunities to verify a considered HS.

For future work, we plan to combine more than two functions to form a combined BC in a general way, and give more functions ψ satisfying condition (5) and establish a library for them. In addition, it deserves to investigate how to recover the error caused by the numeric computation in SDP by some symbolic computation techniques, and how to combine with SMT solvers for non-linear arithmetic [10].

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[2] X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *CAV'13*, volume 8044 of *LNCS*, pages 258–263. Springer, 2013.

[3] L. Dai, B. Xia, and N. Zhan. Generating non-linear interpolants by semidefinite programming. In *CAV'13*, volume 8044 of *LNCS*, pages 364–380. Springer, 2013.

[4] A. Eggers, N. Ramdani, N. Nedialkov, and M. Fränzle. Improving the SAT modulo ODE approach to hybrid systems analysis by combining different enclosure methods. *Software and Systems Modeling*, pages 1–28, 2012.

[5] S. Gao, S. Kong, and E. M. Clarke. dReal: An SMT solver for nonlinear theories over the reals. In *CADE'13*, volume 7898 of *LNCS*, pages 208–214. Springer, 2013.

[6] S. Gulwani and A. Tiwari. Constraint-based approach for analysis of hybrid systems. In *CAV'08*, volume 5123 of *LNCS*, pages 190–203. Springer, 2008.

[7] J. Han, Z. Jin, and B. Xia. Proving inequalities and solving global optimization problems via simplified CAD projection. *CoRR*, abs/1205.1223, 2012.

[8] T. A. Henzinger and P.-H. Ho. Algorithmic analysis of nonlinear hybrid systems. In *CAV'95*, volume 939 of *LNCS*, pages 225–238. Springer, 1995.

[9] M. Jirstrand. Invariant sets for a class of hybrid systems. In *CDC'98*, volume 4, pages 3699–3704. IEEE, 1998.

[10] D. Jovanovic and L. M. de Moura. Solving non-linear arithmetic. In *IJCAR'12*, volume 7364 of *LNCS*, pages 339–354. Springer, 2012.

[11] H. Kong, F. He, X. Song, W. Hung, and M. Gu. Exponential-condition-based barrier certificate generation for safety verification of hybrid systems. In *CAV'13*, volume 8044 of *LNCS*, pages 242–257. Springer, 2013.

[12] H. Kong, X. Song, D. Han, M. Gu, and J. Sun. A new barrier certificate for safety verification of hybrid systems. *The Computer Journal*, 2013.

[13] G. Lafferriere, G. J. Pappas, and S. Yovine. Symbolic reachability computation for families of linear vector fields. *Journal of Symbolic Computation*, 32(3):231–253, 2001.

[14] W. Lin, M. Wu, Z. Yang, and Z. Zeng. Exact safety verification of hybrid systems using sums-of-squares representation. *SCIENCE CHINA Information Sciences*, 57(5):1–13, 2014.

[15] J. Liu, J. Lv, Z. Quan, N. Zhan, H. Zhao, C. Zhou, and L. Zou. A calculus for hybrid CSP. In *APLAS'10*, volume 6461 of *LNCS*, pages 1–15. Springer, 2010.
[16] J. Liu, N. Zhan, and H. Zhao. Computing semi-algebraic invariants for polynomial dynamical systems. In *EMSOFT'11*, pages 97–106. ACM Press, 2011.
[17] J. Liu, N. Zhan, H. Zhao, and L. Zou. Abstraction of elementary hybrid systems by variable transformation. In *FM'15*, volume 9109 of *LNCS*, pages 360–377. Springer, 2015.
[18] P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Inst. of Tech., 2000.
[19] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96:293–320, 2003.
[20] A. Platzer and E. Clarke. Computing differential invariants of hybrid systems as fixedpoints. In *CAV'08*, volume 5123 of *LNCS*, pages 176–189. Springer, 2008.
[21] A. Platzer and E. M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. *Formal Methods in System Design*, 35(1):98–120, 2009.
[22] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *HSCC'04*, volume 2993 of *LNCS*, pages 477–492. Springer, 2004.
[23] S. Prajna, A. Jadbabaie, and G. J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Transactions on Automatic Control*, 52(8):1415–1428, 2007.
[24] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. Sostools and its control applications. In *Positive polynomials in control*, pages 273–292. Springer, 2005.
[25] A. Puri and P. Varaiya. Decidability of hybrid systems with rectangular differential inclusions. In *CAV'94*, volume 818 of *LNCS*, pages 95–104. Springer, 1994.
[26] S. Ratschan and Z. She. Safety verification of hybrid systems by constraint propagation-based abstraction refinement. *ACM Trans. Embedded Comput. Syst.*, 6(1), 2007.
[27] E. Rodríguez-Carbonell and A. Tiwari. Generating polynomial invariants for hybrid systems. In *HSCC'05*, volume 3414 of *LNCS*, pages 590–605. Springer, 2005.
[28] S. Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In *HSCC'10*, pages 221–230. ACM Press, 2010.
[29] S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Constructing invariants for hybrid systems. In *HSCC'04*, volume 2993 of *LNCS*, pages 539–554. Springer, 2004.
[30] C. Sloth, G. J. Pappas, and R. Wisniewski. Compositional safety analysis using barrier certificates. In *HSCC'12*, pages 15–24. ACM Press, 2012.
[31] A. Taly and A. Tiwari. Deductive verification of continuous dynamical systems. In *FSTTCS'09*, volume 4 of *LIPIcs*, pages 383–394, 2009.
[32] M. Wu and Z. Yang. Generating invariants of hybrid systems via sums-of-squares of polynomials with rational coefficients. In *SNC'11*, pages 104–111. ACM, 2011.
[33] Z. Yang, W. Lin, and M. Wu. Exact safety verification of hybrid systems based on bilinear SOS representation. *ACM Trans. Embedded Comput. Syst.*, 14(1):16:1–16:19, 2015.
[34] N. Zhan, S. Wang, and H. Zhao. Formal modelling, analysis and verification of hybrid systems. In *Unifying Theories of Programming and Formal Engineering Methods*, volume 8050 of *LNCS*, pages 207–281. Springer, 2013.
[35] 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 *FM'14*, volume 8442 of *LNCS*, pages 733–748. Springer, 2014.

A. The details of Examples

The polynomials synthesized in Example 2 are:

$\varphi = 0.030317x_1^8 + (6.9115e-05)x_1^7x_2 - (3.6889e-05)x_1^7 + 0.090347x_1^6x_2^2 - 0.11095x_1^6x_2 - 0.75683x_1^6 - (9.0598e-05)x_1^5x_2^3 - 0.00017438x_1^5x_2^2 - (5.4845e-05)x_1^5x_2 + (7.291e-05)x_1^5 - 0.30715x_1^4x_2^4 + 1.0445x_1^4x_2^3 - 1.5458x_1^4x_2^2 + 0.57141x_1^4x_2 - 0.26344x_1^4 - (6.3369e-05)x_1^3x_2^5 + 0.00010503x_1^3x_2^4 + 0.00038237x_1^3x_2^3 - 0.00036159x_1^3x_2^2 - 0.00010184x_1^3x_2 + (9.2214e-05)x_1^3 + 0.03383x_1^2x_2^6 + 0.33103x_1^2x_2^5 - 2.9864x_1^2x_2^4 + 2.0938x_1^2x_2^3 - 0.12636x_1^2x_2^2 + 0.79519x_1^2x_2 - 0.62237x_1^2 + 1.4962e-05x_1x_2^7 - 0.00014241x_1x_2^6 + 0.00048485x_1x_2^5 - 0.00065416x_1x_2^4 + 0.00014521x_1x_2^3 + 0.00040002x_1x_2^2 - 0.00031516x_1x_2 + (6.343e-05)x_1 - 0.0043261x_2^8 + 0.05803x_2^7 - 0.29525x_2^6 + 0.80728x_2^5 - 1.2538x_2^4 + 1.2862x_2^3 - 0.76567x_2^2 + 0.29172x_2 - 0.072688,$

$\chi = 9.8484x_1^6 + 0.001271x_1^5x_2 + 13.4422x_1^4x_2^2 - 31.2496x_1^4x_2 - 85.8767x_1^4 - 0.0031705x_1^3x_2^2 - 0.012227x_1^3x_2 - 0.0042103x_1^3 + 5.396x_1^2x_2^4 - 28.4976x_1^2x_2^3 - 46.3212x_1^2x_2^2 + 87.5486x_1^2x_2 - 44.1755x_1^2 + 0.0049683x_1x_2^2 - 0.0058767x_1x_2 - 0.0020784x_1 + 0.46783x_2^6 - 4.0071x_2^5 + 6.1875x_2^4 + 37.296x_2^3 - 100x_2^2 + 2.0932x_2 - 12.8904,$

$\delta = 0.014034x_1^8 + (3.0608e-06)x_1^7x_2 - (5.813e-06)x_1^7 + 0.0021473x_1^6x_2^2 - 0.013483x_1^6x_2 - 0.0064165x_1^6 + (2.5531e-06)x_1^5x_2^3 + (2.7689e-05)x_1^5x_2^2 - (1.0371e-05)x_1^5x_2 - (2.9253e-06)x_1^5 + 0.02322x_1^4x_2^4 - 0.008259x_1^4x_2^3 + 0.0095319x_1^4x_2^2 + 0.014437x_1^4x_2 + 0.02625x_1^4 + (1.126e-05)x_1^3x_2^5 - (3.9758e-05)x_1^3x_2^4 + (3.4426e-05)x_1^3x_2^3 - (1.2647e-05)x_1^3x_2^2 + (8.5785e-06)x_1^3x_2 - (5.7495e-07)x_1^3 + 0.00051658x_1^2x_2^6 - 0.010421x_1^2x_2^5 + 0.032928x_1^2x_2^4 - 0.041062x_1^2x_2^3 + 0.030059x_1^2x_2^2 - 0.030394x_1^2x_2 + 0.013862x_1^2 + (5.2631e-07)x_1x_2^7 - (4.7302e-06)x_1x_2^6 + (1.3673e-05)x_1x_2^5 - (1.3654e-05)x_1x_2^4 + (8.3569e-07)x_1x_2^3 - (1.6543e-06)x_1x_2^2 + (1.3272e-05)x_1x_2 - (8.3958e-06)x_1 + 0.00013121x_2^8 - 0.0015345x_2^7 + 0.0076214x_2^6 - 0.019749x_2^5 + 0.02836x_2^4 - 0.023961x_2^3 + 0.01575x_2^2 - 0.010837x_2 + 0.0044092,$

$u_1 = 33.1703x_1^4 - 0.0080558x_1^3x_2 - 0.014014x_1^3 + 31.8846x_1^2x_2^2 - 22.7751x_1^2x_2 + 33.5594x_1^2 + 0.002164x_1x_2^3 - 0.0059715x_1x_2^2 - 0.037073x_1x_2 + 0.020061x_1 + 10.479x_2^4 + 6.0815x_2^3 + 19.5851x_2^2 - 18.8795x_2 + 24.5699,$

$u_2 = 0.579x_1^8 + (7.0204e-06)x_1^7x_2 - (1.8771e-05)x_1^7 + 0.61572x_1^6x_2^2 - 0.43594x_1^6x_2 + 0.39633x_1^6 + (4.0635e-06)x_1^5x_2^3 + (5.4444e-06)x_1^5x_2^2 - (9.0679e-06)x_1^5x_2 + (4.1779e-05)x_1^5 + 0.5972x_1^4x_2^4 - 0.446x_1^4x_2^3 + 0.8667x_1^4x_2^2 - 0.48811x_1^4x_2 + 0.57967x_1^4 + (2.0738e-06)x_1^3x_2^5 + (4.4963e-06)x_1^3x_2^4 - (3.9037e-06)x_1^3x_2^3 - (1.5008e-05)x_1^3x_2^2 - (6.0762e-05)x_1^3x_2 + (3.4303e-05)x_1^3 + 0.42761x_1^2x_2^6 - 0.20453x_1^2x_2^5 + 0.45199x_1^2x_2^4 - 0.55762x_1^2x_2^3 + 0.80255x_1^2x_2^2 - 0.18571x_1^2x_2 + 0.36852x_1^2 + (5.1943e-06)x_1x_2^7 - (5.229e-06)x_1x_2^6 + (4.0646e-06)x_1x_2^5 - (8.2704e-06)x_1x_2^4 + (1.2324e-05)x_1x_2^3 - (3.1593e-06)x_1x_2^2 - (8.7493e-06)x_1x_2 + (3.0456e-06)x_1 + 0.18043x_2^8 + 0.1527x_2^7 + 0.11373x_2^6 + 0.090147x_2^5 + 0.40667x_2^4 - 0.26137x_2^3 + 0.68588x_2^2 - 0.38649x_2 + 0.50807,$

$u_3 = 0.82691x_1^8 + (6.8463e-06)x_1^7x_2 + (7.824e-06)x_1^7 + 0.66339x_1^6x_2^2 + 0.69976x_1^6x_2 + 0.71112x_1^6 + (2.8381e-06)x_1^5x_2^3 + (2.1678e-05)x_1^5x_2^2 - (2.269e-05)x_1^5x_2 - (2.7155e-05)x_1^5 + 0.67426x_1^4x_2^4 + 0.31337x_1^4x_2^3 + 1.0011x_1^4x_2^2 + 0.41117x_1^4x_2 + 0.94169x_1^4 + (5.177e-06)x_1^3x_2^5 + (2.4687e-05)x_1^3x_2^4 + (4.9193e-05)x_1^3x_2^3 - (8.5264e-05)x_1^3x_2^2 - (9.261e-05)x_1^3x_2 - 0.00018356x_1^3 + 0.55287x_1^2x_2^6 + 0.26734x_1^2x_2^5 + 0.43798x_1^2x_2^4 - 0.42762x_1^2x_2^3 + 0.90269x_1^2x_2^2 + 0.47337x_1^2x_2 + 0.73082x_1^2 + (6.4342e-06)x_1x_2^7 + (1.7535e-05)x_1x_2^6 + (3.191e-05)x_1x_2^5 + (2.9182e-05)x_1x_2^4 + (7.677e-05)x_1x_2^3 + (2.0006e-05)x_1x_2^2 - (3.1684e-05)x_1x_2 - (6.6486e-06)x_1 + 0.45107x_2^8 - 0.15576x_2^7 + 0.12208x_2^6 - 0.29031x_2^5 + 0.39853x_2^4 - 0.57126x_2^3 + 0.29347x_2^2 - 0.33244x_2 + 0.43254,$

$\psi_1(\theta) = \psi_2(\theta) = -4\theta + 2\theta^2.$

The polynomials synthesized in Example 4 are :

$\varphi = 9.8484x_1^6 + 0.001271x_1^5x_2 + 13.4422x_1^4x_2^2 - 31.2496x_1^4x_2 - 85.8767x_1^4 - 0.0031705x_1^3x_2^2 - 0.012227x_1^3x_2 - 0.0042103x_1^3 + 5.396x_1^2x_2^4 - 28.4976x_1^2x_2^3 - 46.3212x_1^2x_2^2 + 87.5486x_1^2x_2 - 44.1755x_1^2 + 0.0049683x_1x_2^2 - 0.0058767x_1x_2 - 0.0020784x_1 + 0.46783x_2^6 - 4.0071x_2^5 + 6.1875x_2^4 + 37.296x_2^3 - 100x_2^2 + 2.0932x_2 - 12.8904,$

$\chi = 0, \delta = 0, u1 = 0,$

$u_2 = 9.8484x_1^6 + 0.001271x_1^5x_2 + 13.4422x_1^4x_2^2 - 31.2496x_1^4x_2 - 85.8767x_1^4 - 0.0031705x_1^3x_2^2 - 0.012227x_1^3x_2 - 0.0042103x_1^3 + 5.396x_1^2x_2^4 - 28.4976x_1^2x_2^3 - 46.3212x_1^2x_2^2 + 87.5486x_1^2x_2 - 44.1755x_1^2 + 0.0049683x_1x_2^2 -$

$0.0058767x_1x_2 - 0.0020784x_1 + 0.46783x_2^6 - 4.0071x_2^5 + 6.1875x_2^4 + 37.296x_2^3 - 100x_2^2 + 2.0932x_2 - 12.8904$,
$u_3 = 9.8484x_1^6 + 0.001271x_1^5x_2 + 13.4422x_1^4x_2^2 - 31.2496x_1^4x_2 - 85.8767x_1^4 - 0.0031705x_1^3x_2^2 - 0.012227x_1^3x_2 - 0.0042103x_1^3 + 5.396x_1^2x_2^4 - 28.4976x_1^2x_2^3 - 46.3212x_1^2x_2^2 + 87.5486x_1^2x_2 - 44.1755x_1^2 + 0.0049683x_1x_2^2 - 0.0058767x_1x_2 - 0.0020784x_1 + 0.46783x_2^6 - 4.0071x_2^5 + 6.1875x_2^4 + 37.296x_2^3 - 100x_2^2 + 2.0932x_2 - 12.8904$,
$\psi_1 = 0, \psi_2(\theta) = -4\theta + 1.5\theta^2$.

The polynomials synthesized in Example 6 are: $\varphi_2 = 1.6165x_1^2 - 0.20569x_1x_2 + (0.19824e-1)x_1x_3 + (0.95436e-5)x_1 + (0.54446e-1)x_2^2 + (0.69996e-3)x_2x_3 - (0.16916e-6)x_2 + (0.9101e-1)x_3^2 + (0.1511e-7)x_3 - 9.6424$
$\chi_2 = (0.89818e-1)x_1^2 - (0.82739e-1)x_1x_2 + (0.21192e-1)x_1x_3 - (0.15224e-8)x_1 + (0.54928e-2)x_2^2 + (0.84123e-2)x_2x_3 + (0.1277e-8)x_2 + (0.35173e-1)x_3^2 + (0.27238e-9)x_3 - 5.3973$
$\delta_2 = 5.5914x_1^2 - 0.21067x_1x_2 - (0.24733e-1)x_1x_3 + (0.87702e-5)x_1 + 0.20573x_2^2 - (0.52174e-1)x_2x_3 + (0.28769e-6)x_2 + 0.22449x_3^2 + (0.87144e-7)x_3 + 0.29484$
$u_{21} = 1.5356x_1^2 + (0.13731e-1)x_1x_2 - (0.19249e-2)x_1x_3 - (0.10079e-6)x_1 + 0.66295x_2^2 - (0.64549e-1)x_2x_3 - (0.63485e-7)x_2 + 0.39611x_3^2 - (0.66953e-8)x_3 + 2.6867$
$u_{22} = 0.73288x_1^2 - (0.22775e-2)x_1x_2 + (0.27401e-2)x_1x_3 - (0.51154e-7)x_1 + 0.59472x_2^2 - (0.55279e-1)x_2x_3 - (0.48206e-7)x_2 + 0.34978x_3^2 - (0.74061e-8)x_3 + 0.60632$
$u_{23} = 2.0821x_1^2 + (0.40593e-1)x_1x_2 - (0.50855e-2)x_1x_3 - (0.8427e-4)x_1 + 0.61146x_2^2 - (0.90046e-2)x_2x_3 - (0.83808e-5)x_2 + 0.14389x_3^2 - (0.1148e-5)x_3 + 4.5124$
$u_{24} = 1.0004x_1^2 + (0.1131e-1)x_1x_2 - (0.22779e-2)x_1x_3 - (0.288e-4)x_1 + 0.517x_2^2 - (0.80914e-2)x_2x_3 - (0.11074e-4)x_2 + (0.83205e-1)x_3^2 - (0.82264e-6)x_3 + 0.70099$
$u_{41} = (0.43056e-3)x_1^2 - (0.29796e-4)x_1x_2 + (0.10489e-3)x_1x_3 + (0.59287e-11)x_1 + (3.8141e-6)x_2^2 + (0.95752e-5)x_2x_3 + (0.26518e-12)x_2 + (0.39903e-4)x_3^2 + (0.51833e-12)x_3 + (0.36e-2)$
$u_{42} = (0.56936e-2)x_1^2 + (0.53069e-2)x_1x_2 + (0.35737e-2)x_1x_3 - (0.75779e-10)x_1 + (0.20039e-2)x_2^2 + (0.26891e-2)x_2x_3 + (0.29818e-10)x_2 + (0.16505e-2)x_3^2 - (0.16159e-10)x_3 + 0.52902$
$u_{51} = (0.28447e-1)x_1^2 - (0.28324e-2)x_1x_2 - (0.37952e-3)x_1x_3 + (0.125e-6)x_1 + (0.52784e-3)x_2^2 - (0.86183e-4)x_2x_3 - (0.63026e-8)x_2 + (0.88611e-4)x_3^2 - (0.86257e-9)x_3 + (0.11143e-1)$
$u_{52} = (0.12129)x_1^2 - (0.13405e-1)x_1x_2 - (0.15008e-2)x_1x_3 - (0.82285e-6)x_1 + (0.47845e-2)x_2^2 - (0.73624e-3)x_2x_3 - (0.20348e-6)x_2 + (0.52816e-3)x_3^2 + (0.21178e-7)x_3 + 3.5079$