# Verification of Annotated Models from Executions


## ABSTRACT

Simulations can help enhance confidence in system designs but they provide almost no formal guarantees. In this paper, we present a simulation-based verification framework for embedded systems described by non-linear, switched systems. In our framework, users are required to annotate the dynamics in each control mode of switched system by something we call a discrepancy function that formally measures the nature trajectory convergence/divergence in the system. Discrepancy functions generalize other measures of trajectory convergence and divergence like Contraction Metrics and Incremental Lyapunov functions. Exploiting such annotations, we present a sound and relatively complete verification procedure for robustly safe/unsafe systems. We have built a tool based on the framework that is integrated into the popular Simulink/Stateflow modeling environment. Experiments with our prototype tool shows that the approach (a) outperforms other verification tools on standard linear and non-linear benchmarks, (b) scales reasonably to larger dimensional systems and to longer time horizons, and (c) applies to models with diverging trajectories and unknown parameters.


## 1. INTRODUCTION

Simulations play an important role in helping designers gain confidence in the correctness of their systems, especially in the context of embedded systems, where verification of designs remains computationally challenging. However, while simulations have proved to be valuable and scalable, they provide almost no formal guarantees about the correctness of designs. Recently, there have been some proposals [10, 15, 8, 6, 20, 12] to obtain formal correctness guarantees from multiple simulation runs of system. In the context of embedded systems, most of them consider continuous dynamical models described by linear differential equations; see Section 2 for a detailed discussion of these papers. In this paper, we consider embedded systems described by a richer class of *switched, non-linear* dynamical system models. These are models with multiple control modes where different (non-linear) physical laws govern the evolution of system state, and a switching/control sequence determines which control mode is visited at different times. Embedded systems with time triggered behavioral changes can be naturally modeled in this framework. We present a framework that formally verifies the safety of such systems based on simulating system designs or executing system implementations.

Our approach, relies on the inherent continuity in the behaviors of such a system — if two executions start close by, then the distance between them at time $t$ ($< T$) can be bounded. However, obtaining such a relationship is computationally intractable for general systems. We, therefore, require users to provide *annotations* on the dynamics that allows us to compute such a relationship. Our use of annotations is inspired by the central role that concepts like loop invariants have played in the verification of software.

We introduce a class of annotations that we call discrepancy functions. We show that discrepancy function is a generalization of other well known measures of trajectory convergence and divergence such as *contraction metric* [17] and *incremental stability* [2] in the control theory literature. Thus, the annotations we require, might be naturally produced by a control theorist during the design process. Moreover, there are proposals to construct such measures automatically in the control theory literature, using sum-of-squares and convex optimization tools [3]. In this paper, we also outline some automated techniques to generate discrepancy function annotations for systems; our method is not guaranteed to succeed, but we used them to analyze some of the examples we consider in our experiments.

We then present an algorithm to verify the safety of a non-linear switched system within bounded time, from simulations using such annotations. Simulation traces or executions of embedded systems observe and record the system state only at discrete points in time. Thus, the gaps in the sampled trace have to be filled in order to reason about the actual trace. Second, the recorded states in the execution/simulation might not be the actual states of the system due to errors introduced from sensor quantizations and from numerical integrators used to generate the simulation trace. Our algorithm accounts for these sources of error, and exploits the annotations to overcome these challenges. We show that our verification algorithm is sound and relatively complete. That is, the answers of safety/unsafety of the system given by our algorithm are correct, and if the system is either robustly safe

or unsafe, our algorithm is guaranteed to terminate; we say a system is robustly safe, if all states in some envelope around the system behaviors are safe. In the situation, when the system is safe but not robustly safe, our algorithm might not terminate.

We have built a prototype tool called Check-Execute-Compare Engine (C2E2) that implements our approach. Our tool is integrated into the Simulink/Stateflow modeling framework, which is widely popular among designers of embedded systems. The annotations for the examples we considered were either obtained from the literature, or using our automated approach. Our experimental analysis shows that the approach successfully analyzes commonly arising non-linear systems, and on standard benchmarks outperforms recent tools for non-linear systems like *Flow*$^*$ [5] and *Ariadne* [4]. Our experiments also demonstrate that the method scales to reasonably large dimensional systems and to longer time horizons. One advantage of such a simulation based approach is that it also allows us to verify executable systems where the system models have unknown parameters. A traditional model checker would fail to analyze such systems because of the incompleteness of the system model.

## 2. RELATED WORK

Safety verification of dynamical and hybrid systems using simulations has been studied in a handful of papers. In [10], the authors construct a Metric Transition System (MTS) using the simulations and then verify the transition system. The approach is sound and is relatively complete only for linear models. An incremental Lyapunov function-based approach is used in [11] for constructing approximate bisimulations. In contrast, our approach does not construct intermediate transition relations but directly checks bounded safety.

In [14], the authors give a bisimulation function for stable systems, which is related to incremental Lyapunov functions (see Section 4). Our discrepancy functions can be seen as an extension of bisimulation functions which covers systems with possibly divergent trajectories. Further, discrepancy functions subsume other notions like contraction metrics, which are not considered in [14]. Sensitivity analysis is the central theme in the paper [8] and the authors provide verification algorithms based on sensitivity. For nonlinear systems this technique does not guarantee completeness and is also possibly unsound. Contraction metrics introduced in [17] generalizes the notion of sensitivity for nonlinear systems and informally, it can be considered as a Lyapunov function for sensitivity. In the current paper, we extend the notion of contraction metric, and hence, give a generalization of sound and relatively complete analysis of sensitivity for nonlinear systems. In a similar spirit to this paper, analysis using control theoretic annotations to Simulink models and code has been also been explored recently in [13].

Other approaches which uses simulations for verifying properties of embedded systems are given in [15, 6, 20, 12]. Simulink-Stateflow simulations are used in [15] for constructing symbolic representation of sets of initial states that behave similarly. Approaches [6, 20] use statistical techniques and heuristics to give probabilistic guarantees about the system. In [9], we developed an algorithm for analyzing distributed hybrid systems where the traces are collected from agents with inaccurately synchronized clocks. Here, the algorithm computes states that are reachable by the model through executions that are consistent with the simulation traces, and did not take advantage of annotations.

## 3. PRELIMINARIES

For a vector $x \in \mathbb{R}^n$, $|x|$ denotes the $\ell^2$ norm For $x_1, x_2 \in \mathbb{R}^n$, $|x_1 - x_2|$ is the Euclidean distance between the points. $B_\delta(x_1) \subseteq \mathbb{R}^n$ denotes the closed ball of radius $\delta$ centered at $x_1$. For a set $S \subseteq \mathbb{R}^n$, $B_\delta(S) = \cup_{x \in S} B_\delta(x)$. A set $S_1$ is a *$\delta$-overapproximation* of $S_2$, if $S_2 \subseteq S_1 \subseteq B_\delta(S_2)$. For a bounded set $S$, a *$\delta$-cover* of $S$ is a finite collection of points $\mathcal{X} = \{x_i\}_{i=1}^m$ such that $S \subseteq \bigcup_{i=1}^m B_\delta(x_i)$. Its diameter $diameter(S) \triangleq \sup_{x_1, x_2 \in S} |x_1 - x_2|$.

For an $n \times n$ matrix $A \in \mathbb{R}^{n \times n}$, the *norm* is defined as $|A| = \max_{x:|x|=1} abs(x^T A x)$. $A$ is *positive semi-definite*, written as $A \succeq 0$, if $\forall x \in \mathbb{R}^n$, $x^T A x \geq 0$. It is *positive definite*, $A \succ 0$, if the previous inequality is strict. It is *negative (semi) definite* if $-A$ is positive (semi) definite.

A continuous function $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is *smooth* if all its higher derivatives and partial derivatives exist and are also continuous. It has a Lipschitz constant $K \geq 0$ if for every $x_1, x_2 \in \mathbb{R}^n$, $|f(x_1) - f(x_2)| \leq K|x_1 - x_2|$. A non-negative function $g : \mathbb{R}^n \to \mathbb{R}$ is a *class $\mathcal{K}$ function* if $g(x) \geq 0$ for $x \neq 0, g(0) = 0$ and $g(x) \to 0$ as $x \to 0$. A class $\mathcal{K}$ function $g$ is called $\mathcal{K}_\infty$ if $g(x) \to \infty$ as $x \to \infty$. For example, the function $f(y_1, y_2) = y_1^2$ belongs to class $\mathcal{K}$ but not to $\mathcal{K}_\infty$. A function $g : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is called a $\mathcal{KL}$ function, if and only if (1) for each $t \in \mathbb{R}$, $g_t(x) \triangleq g(x,t)$ is a $\mathcal{K}$ function and (2) for each $x \in \mathbb{R}^n$, $g_x(t) \triangleq g(x,t) \to 0$ as $t \to \infty$ (see Appendix of [16] for these standard definitions).

A *matrix function* $\mathbf{M} : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{m \times m}$ maps a state $x \in \mathbb{R}^n$ and a time $t$ to a matrix $\mathbf{M}(x,t)$. $\mathbf{M}$ is called an *uniform metric* if $\forall x \in \mathbb{R}^n, \forall t > 0$, $\mathbf{M}(x,t)$ is symmetric, positive definite and $\forall y \in \mathbb{R}^m, \exists\, v_2, v_1$ such that $0 < v_2 < v_1$, $v_2(y^T \cdot y) \leq y^T \mathbf{M}(x,t) y \leq v_1(y^T \cdot y)$. We will sometimes consider uniform metrics which satisfy the above properties over a subset of $\mathbb{R}^n \times \mathbb{R}$.

### 3.1 Dynamical and Switched System Models

An $n$-dimensional *nonlinear dynamical system* is specified by a differential equation:

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

where the variable $x$ takes values in $\mathbb{R}^n$, $t$ is a non-negative real variable which models time, and $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ is a continuous function. In this context points in $\mathbb{R}^n$ are called *states*. A solution for Equation (1) is a function $\xi : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$, such that for a given initial state $x_0 \in \mathbb{R}^n$, the state of the system at time $t$ is $\xi(x_0, t)$. A solution is also called a *trajectory*. For the vector-valued function $f(x,t) = [f_1(x,t) \cdots f_n(x,t)]^T$, the Jacobian matrix is denoted by $\frac{\partial f}{\partial x}$ and it generalizes differentiation of scalar functions.

A *switched system* is specified by a set of differential equations and a set of *switching signals*. Each switching signal defines the time intervals over which particular differential equation governs the evolution of the system. Let $\mathcal{F} = \{f_i\}_{i \in \mathcal{I}}$ be a

collection of continuous functions indexed by a finite set $\mathcal{I}$, which specify the right hand side of the differential equations. A *switching point* $p$ consists of (a) an element of $\mathcal{I}$, denoted by $p$.mode, and (b) a nonnegative real number denoted by $p$.time which defines the time up to which the system evolves with dynamics $\dot{x} = f_{p.\mathsf{mode}}(x,t)$. A sequence of switching points $\sigma = p_0, p_1, \ldots, p_k$ with $p_0.\mathsf{time} = 0$ and strictly increasing switching times[1] naturally defines a switching signal $\sigma : [0, p_k.\mathsf{time}] \to \mathcal{I}$ as $\sigma(t) = p_i.\mathsf{mode}$ if and only if $t$ is in the half-open interval $[p_{i-1}.\mathsf{time}, p_i.\mathsf{time})$. The execution of a switching system given a switching signal $\sigma = p_1, \ldots, p_k$ is defined as $\xi_\sigma : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ where $\xi_\sigma(x_0, 0) = x_0$ and $\forall t \in (p_{j-1}.\mathsf{time}, p_j.\mathsf{time}), \dot{\xi_\sigma}(x_0,t) = f_{p_j.\mathsf{mode}}(\xi_\sigma(x_0,t), t - p_{j-1}.\mathsf{time})$ and $\xi_\sigma(x_0, p_j.\mathsf{time}) = lim_{t \to p_j.\mathsf{time}^-} \xi_\sigma(x_0,t)$.

We specify a set of switching signals by what we call a *switching interval sequence* which is a finite sequence of the form $\rho = q_1, \ldots, q_k$ where each $q_i$ is of the form $(q_i.\mathsf{mode}, q_i.\mathsf{range})$ with $q_i.\mathsf{mode} \in \mathcal{I}$ and $q_i.\mathsf{range}$ is a closed interval over $\mathbb{R}_{\geq 0}$. The $i^{th}$ element in the sequence is also denoted as $\rho[i]$. The upper and lower bounds of the interval $q_i.\mathsf{range}$ are denoted as $q_i.\mathsf{ub}$ and $q_i.\mathsf{lb}$ respectively. A switching interval sequence defines a set of switching signals $sig(\rho) = \{\sigma |\ \sigma = p_0, \ldots, p_k, p_0.\mathsf{time} = 0, \forall i \in 1 : k, p_i.\mathsf{time} \in q_i.\mathsf{range}\}$. We define a refinement function over a switching interval sequence that gives the finite set of all possible switching interval sequences that are obtained by splitting each of the closed intervals into two closed intervals of equal width. Formally, $refine(\rho) \triangleq \{\rho_1, \ldots, \rho_m\}$ such that $\bigcup_{i=1:m} sig(\rho_i) = sig(\rho)$ and $\forall i = 1 : m, \forall j = 1 : k$, either $\rho_i[j].\mathsf{lb} = \rho[j].\mathsf{lb}; \rho_i[j].\mathsf{ub} = \frac{1}{2}(\rho[j].\mathsf{lb} + \rho[j].\mathsf{ub})$ or $\rho_i[j].\mathsf{lb} = \frac{1}{2}(\rho[j].\mathsf{lb} + \rho[j].\mathsf{ub}); \rho_i[j].\mathsf{ub} = \rho[j].\mathsf{ub}$. The set of executions corresponding to $\rho$ is defined as, $\xi_\rho(x_0,t) = \{\xi_\sigma(x_0,t) | \sigma \in sig(\rho)\}$. The *reachable set* for a state $x_0$ for a switching signal $\sigma$ at a given time $t_0$ is given by $\{\xi_\sigma(x_0,t_0)\}$. Thus, by extension *reachable set* for a switching interval sequence $\rho$ at time $t_0$ is $\bigcup_{\sigma \in sig(\rho)} \{\xi_\sigma(x_0,t_0)\}$. The notion extends naturally for a set of states and for interval durations.

DEFINITION 1. *The system $\mathcal{F}$, with a switching interval sequence $\rho$ is* $\epsilon$-robustly safe upto $T_{bound}$, *for a given initial state $x_0$, an unsafe set $U \subseteq \mathbb{R}^n$, an $\epsilon \geq 0$ and a time bound $T_{bound}$ if for every $t \in [0, T_{bound}]$, $B_\epsilon(\xi_\rho(x_0,t)) \cap U = \emptyset$. It is* robustly safe *if it is $\epsilon$-robustly safe for some $\epsilon > 0$. The definition is extended in the natural way to a set of initial states $\Theta$.*

# 4. TRAJECTORY DISCREPANCY FUNCTION

The bounded safety verification algorithm proposed in Section 6 exploits annotations that have to be provided with the system model. We propose a type of annotation which generalizes several different types of proof certificates that are used in the control theory literature for analyzing trajectories of dynamical systems. We call this general class of annotations discrepancy functions. Informally, a trajectory discrepancy function gives an upper bound on the distance between two trajectories as a function of the distance between their initial states and the time elapsed. In the following subsections, we present three different ways to obtain discrepancy functions with the help of a running example.

[1]In this paper we require strictly increasing switching times, but in the general theory of switched systems the switching times may be nondecreasing [16].

DEFINITION 2. *A smooth function $V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is called a* discrepancy function *for the system (1) if and only if there are functions $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and a uniformly continuous function $\beta : \mathbb{R}^{2n} \times \mathbb{R} \to \mathbb{R}_{\geq 0}$ with $\beta(x_1,x_2,t) \to 0$ as $|x_1 - x_2| \to 0$ such that for any pair of states $x_1, x_2 \in \mathbb{R}^n$:*

$$x_1 \neq x_2 \iff V(x_1,x_2) > 0, \tag{2}$$

$$\alpha_1(|x_1 - x_2|) \leq V(x_1,x_2) \leq \alpha_2(|x_1 - x_2|) \textit{and} \tag{3}$$

$$\forall\, t > 0.\ V(\xi(x_1,t), \xi(x_2,t)) \leq \beta(x_1,x_2,t), \tag{4}$$

*A tuple $(\alpha_1, \alpha_2, \beta)$ satisfying the above conditions is called a* witness *to the discrepancy function.*

The complete annotation for the system (1) is a discrepancy function and its witness. The function $V(x_1,x_2)$ vanishes to zero if and only if the first two arguments are identical. The second condition states that the value of $V(x_1,x_2)$ can be upper and lower-bounded by functions of the $\ell^2$ distance between $x_1$ and $x_2$. These functions are used in algorithm in Figure 1 for computing the tubes that are used in safety verification. The final, and the more interesting, condition requires that the function $V$ applied to solutions of (1) at a time $t$ from a pair of initial states is upper bounded and converges to 0 as the $\ell^2$ norm of $x_1 - x_2$ converges to 0. We note that Definition 2 can be restricted in a natural way to define *local* discrepancy functions which satisfy the above properties over a given subset of $\mathbb{R}^n$.

**Remark :** *[Comparison of Discrepancy Function and Sensitivity Analysis]* Sensitivity $s_x$ for the system given in 1 is given by the differential equation $\dot{s_x} = \frac{\partial f}{\partial x} s_x$ and the value of sensitivity at time $t$ for an initial state $x_0$, denoted as $s_{x_0}(t)$ is obtained by solving this differential equation along the solution of the system 1. It has observed in [8] that $|s_{x_0}(t)| \cdot \epsilon$ is an approximate estimate of $|\xi(x,t) - \xi(x_0,t)|$ where $|x - x_0| = \epsilon$. However this estimate is neither an overapproximation, nor an under-approximation. Compared to sensitivity analysis, our approach requires that $\beta(x_0,x,t)$ gives an overapproximation of $V(\xi(x_0,t), \xi(x,t))$.

In the remainder of this section, we observe that proof certificates routinely used in stability analysis of dynamical systems are in fact discrepancy functions. Proofs of the propositions appear in the Appendix.

## 4.1 Lipschitz dynamics

Consider a system (1) such that $L$ is Lipschitz constant of $f(x,t)$. We observe that for such a system, the function $|x_1 - x_2|$ is a discrepancy function. Lipschitz constants can be computed algorithmically for linear, polynomial, and certain classes of trigonometric functions. For more general classes, empirical techniques can estimate it over closed subsets [23].

PROPOSITION 1. *For system (1) with Lipschitz constant $L \in \mathbb{R}_{\geq 0}$ for the function $f(x,t)$, $V(x_1,x_2) \triangleq |x_1 - x_2|$ is a discrepancy function with witness $(\alpha_1, \alpha_2, \beta)$ where $\alpha_1(|x_1 - x_2|) = \alpha_2(|x_1 - x_2|) = |x_1 - x_2|$ and $\beta(x_1,x_2,t) = e^{Lt}|x_1 - x_2|$.*

**Example 1** A second order differential equation modeling the dynamics of current in an RLC circuit is given by: $\frac{d^2 i}{dt^2} +$

$\frac{R}{L}\frac{di}{dt} + \frac{i}{LC} = 0$. For particular choice of the $R$, $L$, and $C$ parameters the system can be written as:

$$\begin{pmatrix}\dot{x}\\ \dot{y}\end{pmatrix} = \begin{pmatrix}0 & 1\\ -\frac{1}{LC} & -\frac{R}{L}\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} = \begin{pmatrix}0 & 1\\ -2 & -2\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix}. \quad (5)$$

The system shows damped oscillations and eventually stabilizes to the origin. Considering an operating region in the state space of the system, say $R = [-10, 10] \times [-10, 10]$, the Lipschitz constant over this interval can be computed by maximizing $\frac{|A_1[x_1,y_1]^T - A_1[x_2,y_2]^T|}{|[x_1-x_2, y_1-y_2]^T|}$ by over $(x_1, y_1)$ and $(x_2, y_2)$ in $R$, where $A_1$ is the matrix in Equation (5). Thus, in this case the Lipschitz constant is equal to the matrix norm $|A_1|$ which can be computed using any algorithm for finding eigenvalues of $A_1$. For this numerical example, we found the Lipschitz constant to be $L = 2.9208$ using MATLAB. It follows that, the discrepancy function is given as $V(x_1, y_1, x_2, y_2) = |[x_1 - x_2, y_1 - y_2]^T|$ and $\beta(x_1, y_1, x_2, y_2, t) = e^{2.9208t}|[x_1 - x_2, y_1 - y_2]^T|$. □

## 4.2 Contraction metrics

DEFINITION 3 (DEFINITION 2 FROM [17]). *A uniform metric $\mathbf{M} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^{n\times n}$ is called a* contraction metric *for the system (1) if there exists a constant $\beta_M \in \mathbb{R}_{\geq 0}$ such that*

$$\frac{\partial f}{\partial x}^T \mathbf{M}(x,t) + \mathbf{M}(x,t)\frac{\partial f}{\partial x} + \dot{\mathbf{M}}(x,t) + \beta_M \mathbf{M}(x,t) \preceq 0. \quad (6)$$

THEOREM 2 (THEOREM 2 FROM [17]). *For system (1) with a contraction metric $\mathbf{M}$, the distance between the trajectories changes exponentially with time, i.e., $\exists\ k \geq 1, \gamma > 0$ such that, $\forall x_1, x_2 \in \mathbb{R}^n$, $\delta x^T \cdot \delta x \leq k\delta x_0^T \cdot \delta x_0 e^{-\gamma t}$, where $\delta x_0 = x_1 - x_2$ and $\delta x = \xi(x_1, t) - \xi(x_2, t)$.*

PROPOSITION 3. *For system (1) with a contraction metric $\mathbf{M}$, $V(x_1, x_2) \triangleq (x_1 - x_2)^T(x_1 - x_2)$ is a discrepancy function with witness $(\alpha_1, \alpha_2, \beta)$ where $\alpha_1(|x_1 - x_2|) = \alpha_2(|x_1 - x_2|) = (|x_1 - x_2|)^2$ and $\beta(x_1, x_2, t) = k(|x_1 - x_2|)^2 e^{-\gamma t}$ where $k, \gamma$ are from Theorem 2.*

**Example 2** Continuing with the system in Example 1, we compute the Jacobian $\frac{\partial f}{\partial x} = \begin{pmatrix}0 & 1\\ -2 & -2\end{pmatrix}$ and observe that the matrix function $\mathbf{M}(x,t) \triangleq \begin{pmatrix}2.5 & 0.5\\ 0.5 & 0.75\end{pmatrix}$ is a uniform metric. Evaluating the right hand side of equation (6) with $\beta_M = 0.5$, we obtain $\frac{\partial f}{\partial x}^T \mathbf{M}(x,t) + \mathbf{M}(x,t)\frac{\partial f}{\partial x} + \dot{\mathbf{M}}(x,t) + \beta_M \mathbf{M}(x,t) = \begin{pmatrix}-0.75 & 0.25\\ 0.25 & -1.625\end{pmatrix} \prec 0$. Hence, $\mathbf{M}(x,t)$ is a contraction metric for Example 1. From Theorem 2, we have that $\exists k \geq 1, \gamma > 0$, such that $\delta x^T \delta x \leq k\delta x_0^T \delta x e^{-\gamma t}$. From Proposition 3, the function $V(x_1, y_1, x_2, y_2) = (x_1 - x_2, y_1 - y_2)^T(x_1 - x_2, y_1 - y_2)$ is a discrepancy function with witness $\alpha_1(|(x_1, y_1) - (x_2, y_2)|) = \alpha_2(|(x_1, y_1) - (x_2, y_2)|) = ((x_1 - x_2)^2 + (y_1 - y_2)^2)$ and $\beta(x_1, y_1, x_2, y_2, t) = k((x_1 - x_2, y_1 - y_2)^T(x_1 - x_2, y_1 - y_2)e^{-\gamma t}$. □

## 4.3 Incremental Stability

Incremental stability is a notion used in control theory [2] which formalizes the property that the Euclidean distance between two trajectories is bounded by a $\mathcal{KL}$ function of the distance between their initial states and time. This is stronger than what is required in a discrepancy function which may not converge to zero as time goes to infinity.

DEFINITION 4. *The system (1) is* incrementally stable *if there is a $\mathcal{KL}$ function $\beta$ such that for any two initial states $x_1$ and $x_2$ in $\mathbb{R}^n$, $|\xi(x_1, t) - \xi(x_2, t)| \leq \beta(|x_1 - x_2|, t)$.*

THEOREM 4 (THEOREM 1 FROM [2]). *If system (1) is incrementally stable then there exists a smooth function $V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ and $\exists \alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and $\alpha \in \mathcal{K}$, such that for every pair of states $x_1, x_2 \in \mathbb{R}^n$*

$$\alpha_1(|x_1 - x_2|) \leq V(x_1, x_2) \leq \alpha_2(|x_1 - x_2|), \text{and} \quad (7)$$

$$V(\xi(x_1, t), \xi(x_2, t)) - V(x_1, x_2) \leq \int_0^t -\alpha(|\xi(x_1, \tau) - \xi(x_2, \tau)|)d\tau. \quad (8)$$

*The function $V$ is called an* incremental Lyapunov function.

PROPOSITION 5. *For system (1) with incremental Lyapunov function $V$, then a the function $V$ is a discrepancy function with witness $(\alpha_1, \alpha_2, \beta)$ where $\alpha_1$ and $\alpha_2$ are from Equation 7 and $\beta(x_1, x_2, t) = V(x_1, x_2) + \int_0^t -\alpha(|\xi(x_1, \tau) - \xi(x_2, \tau)|)d\tau$ where $\alpha$ is from Equation 8.*

**Example 3** Continuing with the system in Example 1, here we start with a candidate quadratic incremental Lyapunov function $V((x_1, y_1), (x_2, y_2)) \triangleq (x_1 - x_2, y_1 - y_2)^T P(x_1 - x_2, y_1 - y_2)$, with $\alpha_1 \triangleq 0.375(x_1 - x_2)^2 + 0.375(y_1 - y_2)^2 + 0.5(x_1 - x_2)(y_1 - y_2)$, $\alpha_2 \triangleq 1.25(x_1 - x_2)^2 + 1.25(y_1 - y_2)^2 + 0.5(x_1 - x_2)(y_1 - y_2)$ and $\alpha \triangleq (x_1 - x_2)^2 + (y_1 - y_2)^2$, with $P = \begin{pmatrix}1.25 & 0.25\\ 0.25 & 0.375\end{pmatrix}$. First we check that $V$ is indeed an incremental Lyapunov function by computing $\dot{V}((x_1, y_1), (x_2, y_2))$ which turns out to be $(x_1 - x_2, y_1 - y_2)^T[A^T P + PA](x_1 - x_2, y_1 - y_2) = -\alpha(x_1, y_1, x_2, y_2)$. Since $\alpha > 0$ whenever $(x_1, y_1) \neq (x_2, y_2)$, $(x_1, y_1) \neq (0, 0)$ and $(x_2, y_2) \neq (0, 0)$. Integrating both sides, we get $V(\xi(x_1, y_1, t), \xi(x_2, y_2, t)) - V(x_1, y_1, x_2, y_2) \leq \int_0^t -\alpha(\xi(x_1, y_1, \tau), \xi(x_2, y_2, \tau))\ d\tau$. Hence $V$ is a discrepancy function with witness $(\alpha_1, \alpha_2, \beta)$ where $\beta((x_1, y_1), (x_2, y_2), t) = V((x_1, y_1), (x_2, y_2)) + \int_0^t -\alpha(\xi(x_1, y_1, t), \xi(x_2, y_2, t))d\tau$. □

**Example 4** Consider a two dimensional linear system: $\dot{x} = -x$; $\dot{y} = \frac{-y_0}{100}$, with initial state $(x_0, y_0)$. A trajectory of the system is given by $\xi((x_0, y_0), t) = [x_0 e^{-t}, y_0 \frac{(1-t)}{100}]$. The system converges to origin. In this example, the distance between two trajectories from different initial states decreases linearly and not exponentially. It can be verified that the function $V((x_1, y_1), (x_2, y_2)) = (x_1 - x_2)^2 + (y_1 - y_2)^2$ is an incremental Lyapunov function. The Jacobian $\frac{\partial f}{\partial x}$ is $\begin{pmatrix}-1 & 0\\ 0 & 0\end{pmatrix}$

and hence a uniform metric $\mathbf{M}(x,t)$ that satisfies Definition 3 does not exist. □

As seen in Example 4, incremental Lyapunov function exists for systems which contraction metric does not exist. However, the guarantees given by contraction metric is that the distance between trajectories decreases exponentially, which is much stronger guarantee than what is obtained by incremental stability. If the contraction metric is a constant, then one can derive equivalence between contraction metric and incremental stability. The common trait between contraction metrics and incremental Lyapunov functions is that these techniques are useful only for proving convergence of trajectories. This is not necessary for a discrepancy function. For more details regarding these two concepts, an interested reader is referred to [17, 2].

Our last example of this section illustrates that for unstable systems over a bounded time horizon, we can still obtain discrepancy functions, even though trajectories from neighboring states actually diverge with time.

**Example 5** Consider the system $\dot{x} = 1$; $\dot{y} = \frac{y_0}{100}$, where $(x_0, y_0)$ is the initial state of the trajectory. The closed form solution of the above system is given as $x(t) = x_0 + t, y(t) = y_0(1 + \frac{t}{100})$. For two trajectories starting from $(x_1, y_1)$ and $(x_2, y_2)$, the distance between the trajectories after time $t$ is given as $\sqrt{(x_1 - x_2)^2 + (1 + \frac{t}{100})^2(y_1 - y_2)^2}$. Observe that the function $V(x_1, y_1, x_2, y_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ with $\alpha_1 = \alpha_2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ and $\beta(x_1, y_1, x_2, y_2, t)$ $= \sqrt{(x_1 - x_2)^2 + (1 + \frac{t}{100})^2(y_1 - y_2)^2}$ satisfies all the required conditions by Definition 2 and hence is a discrepancy function . □

## 5. COMPUTING ANNOTATIONS

Our verification algorithm relies on annotations, namely the discrepancy functions of the dynamical system and its witness. We expect the users to derive these annotations through some means—possibly using existing control theoretic techniques. In Section 4 we showed how knowledge about a system in terms of Lipschitz constants, contraction metrics, and incremental Lyapunov functions can be exploited for obtaining annotations. In this section, we briefly discuss the related but orthogonal issue of obtaining discrepancy functions for both linear and nonlinear dynamical systems. The strategy discussed here restricts the search to *exponential discrepancy functions* which grow or shrink exponentially with time.

DEFINITION 5. *For system (1) a discrepancy function of the system $V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is said to be an* exponential discrepancy function *if $\exists \gamma \in \mathbb{R}$ such that:*

$$\frac{\partial V(x_1,x_2)}{\partial x_1} f(x_1,t) + \frac{\partial V(x_1,x_2)}{\partial x_2} f(x_2,t) \leq \gamma V(x_1,x_2). \quad (9)$$

Since $V(\xi(x_1,t),\xi(x_2,t)) \leq e^{\gamma_1 t} V(x_1,x_2)$, it follows that $V$ satisfies Equation (4) with $\beta(x_1,x_2,t) = e^{\gamma_1 t} V(x_1,x_2)$. Thus an exponential discrepancy function $V$ is indeed a special type of discrepancy function.

PROPOSITION 6. *Consider a linear system $\dot{x} = Ax$. Suppose two $n \times n$ matrices $P$ and $Q$ satisfy the (Lyapunov-like) equation $A^T P + PA = Q$ and $P$ is positive definite and $Q$ is either positive definite or negative semi-definite. Then, $V(x_1,x_2) \triangleq (x_1 - x_2)^T P (x_1 - x_2)$ is an exponential discrepancy function.*

PROOF. The left hand side of Equation (9) evaluates to $(x_1 - x_2)^T Q (x_1 - x_2)$. If $Q$ is negative-semi definite, then Equation (9) is trivially satisfied with $\gamma = 0$. If $Q$ is negative definite, then a value of $\gamma < 0$ is guaranteed to exist such that $(x_1 - x_2)^T Q (x_1 - x_2) \leq \gamma (x_1 - x_2)^T P (x_1 - x_2)$. Otherwise $Q$ is positive definite, and since $P$ is also positive definite, a $\gamma > 0$ is guaranteed to exist, such that $(x_1 - x_2)^T Q (x_1 - x_2) \leq \gamma (x_1 - x_2)^T P (x_1 - x_2)$. □

The hypothesis of Proposition 6 is satisfied by linear systems that are Lyapunov stable, globally asymptotically stable (refer to [16] for formal definitions) and for systems whose trajectories diverge. For such linear systems, exponential discrepancy functions can be computed completely automatically.

The underlying principle for finding exponential discrepancy functions for nonlinear systems is similar. The key step is to identify classes of systems for which Equations (2), (3), and (9) can be encoded and effectively solved as convex optimization problems. We demonstrate the procedure for finding quadratic discrepancy functions encoded as linear matrix inequalities (LMIs) which are solved using existing software tools such as the sum-of-squares toolbox [21]. Similar procedures may be instantiated for finding polynomials discrepancy functions of higher degree.

Consider that a candidate quadratic form $V(x_1,x_2) \triangleq (x_1 - x_2)^T P (x_1 - x_2)$, where $P$ is a $n \times n$ matrix. Plugging this in Equation (9) we get: $(f(x_1,t) - f(x_2,t))^T P (x_1 - x_2) + (x_1 - x_2)^T P (f(x_1,t) - f(x_2,t)) < \gamma V(x_1,x_2)$. We denote the left hand side of this expression by $R(x_1,x_2,t)$ and proceed as follows: (1) Express the constraints $V(x_1,x_2) > 0$ and $R(x_1,x_2,t) \leq 0$ as LMIs and check feasibility. (2) If feasible, then search for appropriate $\gamma \leq 0$ such that $R(x_1,x_2,t) \leq \gamma V(x_1,x_2)$, again encoding this as an LMI. (3) If the check in step (1) is *infeasible* , then express the constraints $V(x_1,x_2) > 0$ and $R(x_1,x_2,t) > 0$ as LMIs and check feasibility. (4) If feasible, then search for appropriate $\gamma \geq 0$ such that $R(x_1,x_2,t) \leq \gamma V(x_1,x_2)$ encoding it as an LMI. (5) If step (3) returns *infeasible* , then the search for a quadratic discrepancy function fails.

## 6. SAFETY VERIFICATION FROM DISCREPANCY FUNCTIONS AND SIMULATIONS

In this section, we present an algorithm which uses simulations and annotations for checking safety of switched systems. First, we define the notion of simulations we use, then we present a subroutine for verification of dynamical systems (Section 6.2), and finally in Section 6.3 we use this subroutine in the algorithm for verification of switched models.

### 6.1 Simulation Traces for Dynamical Systems

Given the model of a dynamical system, an initial state $x_0$ and a period of integration $h > 0$ (also called the sampling

period), a standard ODE solver approximates the values of $\xi(x_0, 0), \xi(x_0, h), \xi(x_0, 2h), \ldots$ by numerically integrating the right hand side of the differential equation. In our algorithms and implementations, we rely on a validated ODE solver (for example VNODE-LP [19]), which computes a sequence of regions $R_1, R_2, \ldots$ such that for each $t \in [j \cdot h, (j+1) \cdot h], \xi(x_0, t) \in R_j$. If the diameter of $R_i$ is greater than the desired simulation error bound $\epsilon$, then we reduce the value of $h$ adaptively until the diameter becomes less than $\epsilon$. Features of such simulation traces are stated in the definition below.

DEFINITION 6. *For given a dynamical system described by Equation (1), an initial state $x_0$, a time bound $T$, error bound $\epsilon > 0$, and time step $\tau > 0$, a $(x_0, T, \epsilon, \tau)$-simulation trace is a finite sequence $\psi = (R_0, t_0), (R_1, t_1), \ldots (R_k, t_k)$ where $R_0 = \{x_0\}$, $\forall i > 0$, $R_i \subseteq \mathbb{R}^n$, $t_i \in \mathbb{R}_{\geq 0}$ and*

*(1)* $\forall\, i \in \{0, 1, \ldots k\}, t_{i+1} - t_i \leq \tau, t_0 = 0,$ *and* $t_k = T,$

*(2)* $\forall\, i > 0, \forall t \in [t_{i-1}, t_i], \xi(x_0, t) \in R_i,$ *and*

*(3)* $\forall\, i > 0, diameter(R_i) \leq \epsilon.$

It follows that the union of the sequence of regions in a $(x_0, T, \epsilon, \tau)$-trace contains states of the dynamical system that can be reached from $x_0$ in $T$ time. We will identify this union with the name of the simulation trace $\psi$.

## 6.2 Simulation to Verification: Dynamical Systems

In this section, we will present an algorithm for checking safety of a dynamical system using annotations. For the remainder of this subsection, we fix a differential equation of the form given in Equation (1), an annotation $V$ for it, and a witness $(\alpha_1, \alpha_2, \beta)$ for that annotation. For any state $x \in \mathbb{R}^n$, we define $B^V_\delta(x) \triangleq \{y \mid V(x, y) \leq \delta\}$. For a subset $W \subseteq \mathbb{R}^n$ $B^V_\delta(W) \triangleq \cup_{x \in W} B^V_\delta(x)$. $B^{\alpha_1}_\delta(x)$ and $B^{\alpha_2}_\delta(x)$ are defined analogously. It follows from Equation 3 of Definition 2 that for any state $x$,

$$B^{\alpha_2}_\delta(x) \subseteq B^V_\delta(x) \subseteq B^{\alpha_1}_\delta(x). \tag{10}$$

We are ready to describe the algorithm in detail (see Figure 1). It takes the following inputs: (a) initial partitioning parameter $\delta_0$, (b) the dynamical system model specified by the function $f$, (c) $\Theta$, which is an $\omega$-approximation of the *bounded* set of initial states, (d) $U$ is an *open* set of unsafe states, (e) an annotation $V$ for $f$ and its witness, and (f) $T_{bound}$ is a time bound. We claim that the algorithm returns "SAFE" if all the executions starting from $\Theta$ are safe up to $T_{bound}$, "UNSAFE" if $\exists x_0 \in \Theta$ such that all executions starting from $B_{(\omega+\delta)}(x_0)$ are unsafe and, it returns "$\omega$-TOO LARGE" otherwise.

In line 4, $\delta$-Partition$(S)$ returns a $\delta$-cover of $S$. As $S = \Theta$ is bounded, $\delta$-Partition$(S)$ returns a finite cover. A simulation trace $\psi$ is obtained in line 8, which is a finite sequence $(R_0, t_0), (R_1, t_1), \ldots (R_k, t_k)$ described in Section 6.1. The tube $T$ constructed in line 9 is guaranteed to contain all the states reachable within time $T_{bound}$ from states in $B_\delta(x_0)$ based on the properties of $\beta$ and the accuracy of the simulation. The

1: **Input:** $\delta_0$, $\langle f, \Theta, U\rangle$, $\omega$, $V(x_1, x_2)$, $(\alpha_1, \alpha_2, \beta)$, $T_{bound}$
2: $S \leftarrow \Theta$; $\delta \leftarrow \delta_0$; $\mathcal{R} \leftarrow \emptyset$
3: **while** $S \neq \emptyset$ **do**
4: $\quad\mathcal{X} \leftarrow \delta$-Partition$(S)$
5: $\quad$**for** $x_0 \in \mathcal{X}$ **do**
6: $\quad\quad\epsilon \leftarrow \sup\{\beta(y, x_0, t) \mid y \in B_\delta(x_0),\ 0 \leq t \leq T_{bound}\}$
7: $\quad\quad\epsilon' \leftarrow \sup\{\beta(y, x_0, t) | y \in B_{(\omega+\delta)}(x_0),\ 0 \leq t \leq T_{bound}\}$
8: $\quad\quad\psi \leftarrow$ simulate$(x_0, T_{bound}, \epsilon, \tau)$
9: $\quad\quad T \leftarrow B^V_\epsilon(\psi)$
10: $\quad\quad$**if** $T \cap U = \emptyset$ **then**
11: $\quad\quad\quad S \leftarrow S \setminus B_\delta(x_0)$; $\mathcal{R} \leftarrow \mathcal{R} \cup T$
12: $\quad\quad$**else if** $\exists i.\ B^V_{\epsilon'}(R_i) \subseteq U$ **then**
13: $\quad\quad\quad$**return** (UNSAFE, $\mathcal{R}$, $max\{2\epsilon + \epsilon'\}$)
14: $\quad\quad$**else if** $\exists i.\ R_i \subseteq U$ **and** $B^V_{\epsilon'}(R_i) \not\subseteq U$ **then**
15: $\quad\quad\quad$**return** ($\omega$-TOO LARGE, $\mathcal{R}$, $max\{2\epsilon + \epsilon'\}$)
16: $\quad\quad$**end if**
17: $\quad$**end for**
18: $\quad\delta \leftarrow \delta/2$; $\tau \leftarrow \tau/2$
19: **end while**
20: **return** (SAFE, $\mathcal{R}$, $max\{2\epsilon + \epsilon'\}$)

**Figure 1: Algorithm 1: Safety verification of dynamical systems.**

algorithm returns the result a tuple, where the first element is either SAFE or UNSAFE or $\omega$-TOO LARGE. The second element is the collection of all the tubes computed in line 9 that are safe, denoted as $\mathcal{R}$. The third element $max\{2\epsilon + \epsilon'\}$ is the maximum value of $2\epsilon + \epsilon'$ encountered during the verification of all safe tubes. This value gives the upper bound on the approximation of the reachable set computed by the union of tubes in line 9.

We establish the soundness and relative completeness of the algorithm using a some intermediate propositions.

PROPOSITION 7. *If $x \in B_\delta(x_0)$ then $\forall t \in [0, T_{bound}], \xi(x, t) \in B^V_\epsilon(\xi(x_0, t))$, where $\epsilon = sup\{\beta(x, x_0, t) | 0 \leq t \leq T_{bound}, x \in B_\delta(x_0)\}$.*

PROPOSITION 8. *For $x \in B_\delta(x_0)$, for any $t \in [0, T_{bound}]$, $\xi(x, t) \in B^V_\epsilon(\psi)$, where $\psi$ is the union of regions obtained from a $(x_0, T_{bound}, \epsilon, \tau)$-simulation trace.*

THEOREM 9. *Algorithm 1 is sound. That is, if it returns "SAFE" then all the executions from $\Theta$ are safe, and when it returns "UNSAFE" there exists at least one execution from the initial set, that is unsafe.*

PROOF. Observe that from Proposition 8, the set of all states reachable within time $T_{bound}$ from some state in $B_\delta(x_0)$ is contained in the tube $T$ computed in line 9. Thus, if $T \cap U = \emptyset$, then all executions from the set $B_\delta(x_0)$ are safe, and so are eliminated from consideration in future iterations (line 11). Hence, if the algorithms exits the while loop because $S = \emptyset$, we can conclude that the system is safe.

On the other hand, if there is some $R_i$ in the simulation trace $\psi$ such that $B^V_{\epsilon'}(R_i) \subseteq U$, then it means that $\exists x_0 \in \Theta$ such

that all executions starting from $B_{(\omega+\delta)}(x_0)$ are unsafe. Since $\Theta$ is an $\omega$-approximation of the initial states, and $x_0$ is an element in $\delta$-partitioning of $\Theta$, it follows that there exists at least one state from the set of initial states in $B_{(\omega+\delta)}(x_0)$. Hence, there exists at least one execution from the initial set that is unsafe. Thus, the decision in line 13 is correct. □

PROPOSITION 10. *The collection of safe tubes $\mathcal{R}$ returned by Algorithm 1 in line 20 is utmost $max\{2\epsilon + \epsilon'\}$-approximation of the set of reachable states from the initial set of states.*

THEOREM 11. *For $\omega = 0$, Algorithm 1 is a complete procedure for robust safety verification of dynamical systems. That is, if all the executions from $\Theta$ are robustly safe, then the it will terminate and return "SAFE". If any of the executions from $\Theta$ is unsafe, then it will terminate and return "UNSAFE".*

PROOF. Suppose that all the executions from $\Theta$ are $\epsilon_0$-robustly safe. From the proof of Theorem 9, we can conclude two things. First, line 13 is never executed, since the system is safe. Second, when a set of initial states in eliminated from consideration (line 11), those states are safe. Now, since $V \leq \alpha_2, \alpha_2 \in \mathcal{K}_\infty$, $\exists \delta'$ such that whenever $|x - y| < \delta', V(x, y) < \epsilon_0/4$. Also, since $\beta$ is uniformly continuous in $x$ when $t \in [0, T_{bound}]$, we have that $\exists \delta_1$, such that $\forall y \in B_{\delta_1}(x), t \in [0, T_{bound}]$, we have $|\beta(x, y, t)| < \delta'$. Thus $\exists \delta_1$ such that the value of $\epsilon$ in line 6 is $< \epsilon_0/4$.

We also have that $\alpha_1 \leq V, \alpha_1 \in \mathcal{K}_\infty$, thus $\exists \delta''$ such that $\alpha_1(\delta'') < \epsilon_0/2$. Thus $B^V_{\delta''}(x) \subset B_{\epsilon_0/2}(x)$. Let $\delta_2$ be the value of $\delta$ in the algorithm such that the value of $\epsilon$ in line 6 is $< \delta''$. If the $\delta$ in the algorithm reaches a value $< \min\{\delta_1, \delta_2\}$, The tubes computed in line 9 are

$$\begin{aligned} T &= B^V_{\delta''}(\psi) \\ &\subseteq B^{\alpha_1}_{\delta''}(\psi) \\ &\subset B_{\epsilon_0/2}(B_{\epsilon_0/4}(\xi(x_0, t))) \\ &\subset B_{3\epsilon_0/4}(\xi(x_0, t)) \end{aligned}$$

Since the system is robustly safe, all the tubes would be safe and hence the algorithm will terminate and prove that system is safe.

Suppose $\langle f, \Theta, U\rangle$ is unsafe. Since $U$ is open, there is a initial state $y_0$, time $t \leq T_{bound}$, and $\epsilon_0 > 0$, such that $B_{\epsilon_0}(\xi(y_0, t)) \subseteq U$. Now, since $\xi(\cdot)$ is continuous, there are $\tau_0$ and $\delta_0$ such that for any $x_0 \in B_{\delta_0}(y_0)$, there is an $i$ such that $|\xi(x_0, i\tau_0) - \xi(y_0, t)| < \epsilon_0/4$. Further, $V(x_0, y_0, t) \to 0$ as $|x_0 - y_0| \to 0$, hence $\exists \delta_1$, such that for every $x_0 \in B_{\delta_1}(y_0), \beta(x_1, x_2, t) < \epsilon_0/4$. Finally, since we can simulate with arbitrary precision, and $\delta$ and $\tau$ are decreasing, we will eventually generate a simulation trace, for which line 13 will be reached. □

We remark that in the case when the dynamical system has some execution from $\Theta$ that is safe but not robustly safe, Algorithm 1 may not terminate. Also, note that relative completeness of the above algorithm is achieved if $\delta$ and $\tau$ can be made arbitrarily small. However, even if we have finite precision arithmetic, relative completeness is achieved if the system is robustly safe with $\epsilon_0$ greater than the the granularity of the precision.

1: **Input:** $\langle \mathcal{F}, \Theta, U\rangle$, $\rho$, $\{V_i\}$, $\{(\alpha_{1,i}, \alpha_{2,i}, \beta_i)\}$, $T_{bound}$, $\delta_0$
2: $SwitchingSet \leftarrow \rho$; $\delta \leftarrow \delta_0$
3: **for all** $\rho \in SwitchingSet$ **do**
4: $S \leftarrow \Theta$; $\omega \leftarrow 0$; $\mathcal{R} \leftarrow \emptyset$
5: **for all** $i = 1 : k$, where $\rho = q_1, q_2, \ldots, q_k$ **do**
6: $(result, \mathcal{R}, \omega) \leftarrow CheckDS(\delta, \langle f_{\rho[i].\mathsf{mode}}, S, U\rangle, \omega, \rho[i].\mathsf{ub} - \rho[i-1].\mathsf{lb}, )$
7: **if** $result =$ SAFE **then**
8: $S \leftarrow Project(\mathcal{R}, [\rho[i].\mathsf{ub}, \rho[i].\mathsf{lb}])$
9: **else if** $result =$ UNSAFE **then**
10: **return** UNSAFE
11: **else if** $result = \omega$-TOO LARGE **then**
12: $SwitchingSet \leftarrow SwitchingSet \backslash \{\rho\} \cup refine(\rho)$
13: $\delta \leftarrow \delta/2$ ; GOTO Line 3
14: **end if**
15: **end for**
16: **end for**
17: **return** SAFE

**Figure 2: Algorithm 2: Safety verification of switching system for switching interval sequence $\rho$**

## 6.3 Simulation to Verification: Switched Systems

In this section, we will present an algorithm for verifying switched systems with respect to a set of switching signals that uses Algorithm 1. As before, $\mathcal{F} = \{f_i\}_{i \in \mathcal{I}}$ specifies a collection of dynamical subsystems, $\rho = q_1, q_2, \ldots, q_k$ specifies a collection of switching signals, $\Theta$ is a *bounded* set of initial set of states, $T_{bound}$ is the time bound, and an *open* unsafe region $U$. Our algorithm will decide if an unsafe state is reached within $T_{bound}$ and it uses annotations discrepancy function $V_i(\cdot)$ and witnesses $(\alpha_{1,i}, \alpha_{2,i}, \beta_i)$ for each of the subsystems in $\mathcal{I}$.

For each $\ell \in \{1, \ldots, k\}$, Algorithm 2 makes subroutine calls to Algorithm 1 (subroutine $CheckDS$ in line 6) which attempts to verify that the dynamical system $\dot{x} = f_{q_\ell.\mathsf{mode}}(x, t)$ is safe for the duration $[q_{\ell-1}.\mathsf{lb}, q_\ell.\mathsf{ub}]$. In the process, Algorithm 1 also computes an $\omega$-overapproximation of the reachable states during $q_\ell.\mathsf{range}$ and this is used as $\omega$-approximate initial set for the dynamics $f_{q_{\ell+1}.\mathsf{mode}}$. If Algorithm 2 fails to prove safety or unsafety with respect to $\Theta$ and $\rho$, then it refines the switching interval sequence $\rho$ and the cover of the initial set $\Theta$. The algorithm is given in Figure 2.

The soundness of Algorithm 2 follows from soundness of Algorithm 1 and Theorem 10.

THEOREM 12. *The algorithm in Figure 2 is sound, i.e., when it returns "SAFE" the system is safe, and when it returns "UNSAFE" the system is unsafe.*

PROOF. The proof is based on Theorem 9 and Theorem 10. From Theorem 10, we know that the order of approximation computed by Algorithm 1 in the duration $\rho[i].\mathsf{range}$ is sound. Observe that algorithm 2 returns "SAFE" only when all the subroutine calls return "SAFE" and thus it follows from Theorem 9 that the system is safe in each of the dynamical systems. Notice that algorithm 2 returns "UNSAFE" only when at least one of the subroutine call returns "UNSAFE" and

thus, because of Theorem 10 and Theorem 9, it follows that the system is indeed unsafe. □

THEOREM 13. *If the system $\langle \mathcal{F}, \Theta, U \rangle$ with the set of switching signals $sig(\rho)$ is robustly safe then the Algorithm 2 terminates and returns "SAFE". On the other hand, if the system is unsafe, then algorithm will terminate and return "UNSAFE"*

PROOF. If the system is safe, then it follows from Theorem 9 that line 10 is never executed. Thus, the subroutine call to verification of dynamical system would either return "SAFE", in which case, the algorithm continues, or it would return "$\omega$-TOO LARGE" in which case, we refine $\rho$ and decrease value of $\delta$. Thus, the value of $2\epsilon + \epsilon'$ computed for each of the modes decreases to zero. This ensures that $\omega \to 0$. Since the system is robustly safe, each of the dynamical systems is also robustly safe, and thus $\exists \omega > 0$ such that the each of the dynamical systems would return "SAFE" and hence the algorithm in Figure 2 would return "SAFE".

If the system is unsafe, then it follows from Theorem 9 that line 7 is never executed. Thus, the subroutine call would either return "UNSAFE" in which case, the system is unsafe or it would return "$\omega$-TOO LARGE" in which case we refine $\rho$ and decrease value of $\delta$. This ensures that $\omega \to 0$. Observe that algorithm 1 would return unsafe only when the condition in line 12 is satisfied. Now since $\delta \to 0$, it implies that $\epsilon \to 0$ and $\epsilon' \to 0$ and thus $B^V_{\epsilon'}(R_i) \subseteq U$ is satisfied for some $R_i$. Hence the algorithm will return "UNSAFE". □

We remark that in the case where the system is safe, but not robustly safe, then the above algorithm need not terminate. Further, if the algorithm infers that the system is unsafe, then we can generate a set of traces that correspond to the unsafe behavior. We will conclude this section by pointing out that we are making some computational assumptions about the sets $\Theta, U$, and of the functions $V$ that ensure that all the steps in the algorithms 1 and 2 are computable. We do not explicitly state these assumptions to ensure that our discussion in this paper highlights the key ideas without getting bogged down by technicalities. On a similar note, one can also use $\alpha_1$ and $\alpha_2$ in line 9, instead of $V$ without losing the soundness and completeness guarantees; however, since these are bonded overapproximations, using them might affect the number of iterations needed to prove safety.

**Example 6** Continuing on Example 1, we now compare the performance of the three discrepancy functions defined in Section 4 for the verification procedure described in Figure 1. We begin with the initial conditions as $S = 3 \leq x \leq 5 \wedge y = 0$. The unsafe state if given as $1 < t < 1.2 \wedge x > 3$ and initial value of $\delta$ as 0.1. The number of executions required for proving safety is 70, 10 and 10 if we use discrepancy function as Lipschitz constant, contraction metric and incremental Lyapunov function respectively. In comparison with other annotations, Lipschitz constant requires more number of simulations as it estimates that the distance between executions would increase exponentially, which is a coarse overapproximation. This shows that the time taken to verify a system using simulations depends on the annotation provided by the user. □

# 7. EXPERIMENTAL EVALUATION

For demonstrating the effectiveness of our annotation-based verification Algorithm we have built a tool called Check-Execute-Compare Engine (C2E2). Our tool is integrated into the Simulink Stateflow (SLSF) modeling framework, which is popular among designers of embedded systems. The system verified by C2E2 is described as a Simulink/Stateflow model, which is (automatically) translated into a hybrid automaton according to the semantics described in [18], and VNODE-LP was used to generate simulators for each of the modes of the hybrid automaton. VNODE-LP integration engine that uses finite precision interval arithmetic and Taylor models for generating simulations described in Section 6.1. The tool is developed in C++ and uses GNU Linear Programing Kit (GLPK) to check the distance of simulation traces from the unsafe set. Observe that the algorithm presented in Section 6 requires arbitrary precision arithmetic for relative completeness, however in practice, we use finite precision arithmetic which requires the robustness to be greater than the granularity of precision. In practice we terminate if the partitioning reaches $10^{-7}$ and terminate by saying that the system is not robust with respect to sensitive perturbations.

Our evaluation has four parts: First, we verify a benchmark suite consisting of natural linear and nonlinear dynamical system models. Second, we verify several adaptive control examples where the executable system has a reference model with unknown parameters. Third, we evaluate scalability by increasing the time horizon for verification and the number of dimensions of two parameterized models. Finally, we study the effect of the initial partitioning.

## 7.1 Benchmarks and comparison with other tools

We compare the performance of C2E2 against $Flow^*$ [5] and Ariadne [4] using a benchmark suite consisting of linear and nonlinear models. Ariadne uses faithful geometric representation of sets for computing reachable set and $Flow^*$ uses Taylor Models. Results of comparing the performance with Breach are discussed later in this section. In Table 7, we report the time taken by $Flow^*$ and C2E2 for computing reach set and checking safety with respect to the unsafe set, and the time taken by Ariadne for computing the reachable states. One limitation of $Flow^*$ is that we can only specify unsafe sets as rectangles and not as arbitrary polyhedral constraints. $Flow^*$ running time depends on the remainder error and the order of the Taylor models used. For the results reported here, we use adaptive orders for Taylor models, and set the remainder error to a value such that increasing it by a factor of 2 would change the verification result from "SAFE" to "UNKNOWN". The parameters for Ariadne is set to the same values as that for $Flow^*$. Table 7 show that C2E2 outperforms the other tools in all examples, and in most cases it is faster by at least an order of magnitude.

Breach [7] is another tool against which we evaluated C2E2 . Breach is a toolbox developed in MATLAB with a GUI, making a fair comparison difficult. We ran Breach on Vanderpol coupled oscillator, Sinusoidal tracking, and tank examples, and measured the "wall clock" running time. In all cases C2E2 was faster; we don't report these numbers because of inaccuracies inherent in such numbers. Breach, like $Flow^*$,

| Benchmark | Vars. | TH | Refs. | Sims. | C2E2 (sec) | *Flow*$^*$ (sec) | Ariadne (sec) |
|---|---|---|---|---|---|---|---|
| Moore-Greitzer Jet Engine [3] | 2 | 10 | 12 | 36 | 1.56 | 10.54 | 56.57 |
| Brussellator | 2 | 10 | 33 | 115 | 5.262 | 16.77 | 72.75 |
| VanDerPol | 2 | 10 | 5 | 17 | 0.75 | 8.93 | 98.36 |
| Coupled VanDerPol [3] | 4 | 10 | 10 | 62 | 1.43 | 90.96 | 270.61 |
| Sinusoidal Tracking [22] | 6 | 10 | 12 | 84 | 3.68 | 48.63 | 763.32 |
| Linear Adaptive | 3 | 10 | 8 | 16 | 0.47 | NA | NA |
| Nonlinear Adaptive | 2 | 10 | 16 | 32 | 1.23 | NA | NA |
| NonLin. Adpt. + Disturbance | 3 | 10 | 22 | 48 | 1.52 | NA | NA |

**Table 1: Experimental Results for benchmark examples. Vars: Number of Variables, TH: Time Horizon for Verification, Refs: Number of Refinements, Sims: Total number of simulation traces required for proving safety.**

does not allow polyhedral unsafe sets. Also Breach is neither sound nor complete for nonlinear models, but inaccuracies in the verification results for the examples were not observed.

For the adaptive control system, the $n$-tank system and the nonlinear navigation system we computed the annotations, and for all the other examples the annotations were obtained from the papers [3, 22] in which the systems appeared.

## 7.2 Systems models with unknown parameters

One prominent advantage of our approach is that it supports verification of executable systems where the reference model has unknown parameters. An illustrative example is the linear plant: $\dot{x} = 1; \dot{y} = \theta + u$, where $\theta$ is an unknown parameter and $u$ is the control input designed to drive $y$ to zero. Following a standard adaptive control technique, a new variable $\hat{\theta}$—an estimator for $\theta$—is introduced giving the new dynamics for $y$: $\dot{x} = 1$; $\dot{y} = -\sigma y + \theta - \hat{\theta}$; $\dot{\hat{\theta}} = \gamma y$ $\sigma, \gamma > 0$ are constants chosen by the designer. For the new system, $V((x_1, y_1, \hat{\theta}_1), (x_2, y_2, \hat{\theta}_2)) \triangleq \frac{1}{2}(x_1 - x_2)^2 + \frac{\gamma}{2}(y_1 - y_2)^2 + \frac{1}{2}(\hat{\theta}_1 - \hat{\theta}_2)^2$ is an incremental Lyapunov function $\dot{V} = -\gamma\sigma(y_1 - y_2)^2 < 0$ is used as an annotation.

An example of a nonlinear system with unknown parameters. Consider the system: $\dot{x} = \theta x + u$ where $u$ is the control input. Similar to the earlier example, we introduce $\hat{\theta}$ and define the input $u = -\hat{\theta}x - x$ such that the closed system becomes: $\dot{x} = (\theta - \hat{\theta})x - x$; $\dot{\hat{\theta}} = x^2$ The Lyapunov function $\frac{1}{2}x^2 + \frac{1}{2}(\theta - \hat{\theta})^2$ establishes stability of the system. For obtaining the discrepancy function, we come up with a quartic (fourth degree) discrepancy function . A modified version of this example introduces unknown disturbance inputs. The details of this are omitted owing to limited space.

## 7.3 Scalability with time horizon and continuous dimensions

To check the scalability of the approach with time horizon, we verified the VanDerPol and the Nonlinear Adaptive control benchmarks for time horizons 10, 20 and 40. The verification times of the first were 0.741, 1.662 and 4.373 seconds and for the latter were 1.245, 2.604 and 6.075 seconds respectively. These numbers suggest that the verification time scales roughly linearly with the time horizon for these stable systems.

Table 3(a) shows the scaling of the verification times with the number of continuous variables. We consider the a switched variant of nonlinear version of the $n$ interconnected tank system of [1] and a switched nonlinear model with $n/4$ vehicles and each vehicle having 4 continuous variables. For $n$-tank the initial value of $\delta$ is large and this triggers several refinements and for $n$-vehicles the initial value of $\delta$ is small and this decrease (or eliminates) the need for refinements. There are two key observations. First, as the number of refinements increase, the size of the cover increases proportional to the number of continuous variables in the system. This is evident in the case of $n$-tanks, where for the $n = 12$, 1 refinement step required checking of 16 executions, whereas in the case of $n = 24$, 4 refinement steps required 100 executions. Second, as dimension increases, a smaller value of initial partitioning parameter $\delta$ increases the size of the cover exponentially. This is evident in the case of $n$-vehicles, where adding each new vehicle increased the number of simulations by a factor of 2.

## 7.4 Dependence on Initial set partition

Table 3(b) shows the verification times for different $\delta$-coverings of the initial set. If the value of $\delta$ is too small, then C2E2 generates a large initial partitioning and hence increases the number of simulations. On the other hand, if the initial $\delta$ is too large, then C2E2 needs to perform many refinements, and hence, takes more time. The value of optimum value of $\delta$ clearly depends on that robustness of the system and the relative distance of simulations from unsafe set. Observe that in Table 3(b), the partitioning with $\delta = 0.2$ takes less time for verification than $\delta = 0.5$ and $\delta = 0.1$. Searching for the optimal value of $\delta$ is an interesting direction of future work.

# 8. CONCLUSIONS AND FUTURE WORK

In this paper we present a class of annotations for dynamical systems which are a generalization of other existing well studied notions of trajectory convergence and divergence studied in Control theory such as Lipschitz constant, Contraction Metrics and Incremental Lyapunov functions. Further, we showed that such annotations can also be obtained for systems that exhibit divergent behavior. Verification algorithm for switching systems that exploit simulations and such an-

| Benchmark | Refs. | Sims. | Time |
|---|---|---|---|
| 12-Tanks | 1 | 16 | 2.744 |
| 18-Tanks | 4 | 76 | 15.238 |
| 24-Tanks | 4 | 100 | 22.126 |
| 30-Tanks | 4 | 124 | 28.824 |
| 12-Vehicles | 0 | 32 | 5.477 |
| 16-Vehicles | 0 | 64 | 12.238 |
| 20-Vehicles | 0 | 128 | 25.144 |
| 24-Vehicles | 0 | 256 | 54.236 |

(a) Scalability of verification of for $n$-Tank and $n$-vehicle systems.

| Benchmark | $\delta$ | Refs. | Sims. | Time |
|---|---|---|---|---|
| Nonlin. Adpt. | 0.5 | 16 | 32 | 1.01 |
| Nonlin. Adpt. | 0.2 | 9 | 20 | 0.91 |
| Nonlin. Adpt. | 0.05 | 5 | 13 | 0.58 |
| Nonlin. Adpt. | 0.01 | 0 | 26 | 1.32 |
| VanDerPol | 0.5 | 30 | 96 | 3.84 |
| VanDerPol | 0.2 | 5 | 17 | 0.75 |
| VanDerPol | 0.05 | 8 | 32 | 1.44 |
| VanDerPol | 0.01 | 0 | 120 | 5.87 |

(b) Dependence of running time on initial state covers.

**Figure 3: Scalability and Initial state covers. Refs: Number of refinements, Sims: Number of simulation traces, Time: Running time of C2E2 in seconds.**

notations was presented and was shown to be sound and relatively complete. This is the first simulation-based verification algorithm for nonlinear switched systems that is sound and complete. We have developed a tool C2E2 that implements this algorithm and is integrated to the Simulink/Stateflow modeling framework. Preliminary experimental results demonstrate the promise of this verification approach.

There are many interesting directions of future work. One of the immediate directions is to investigate the techniques for computing an optimal initial partitioning to reduce the verification time. Another interesting direction is automatic generation of these annotations by using techniques like Taylor expansion and Lagrangian remainders. Also, existing works in literature try to generate bisimulations from relations over state space, it is worth investigating whether such bisumulations can be generated from these annotations.

# 9. REFERENCES


[1] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *C.D.C.*, 2008.

[2] D. Angeli. A lyapunov approach to incremental stability properties. *IEEE T. A. C.*, 2000.

[3] E. M. Aylward, P. A. Parrilo, and J.-J. E. Slotine. Stability and robustness analysis of nonlinear systems via contraction metrics and sos programming. *Automatica*, 2008.

[4] A. Balluchi, A. Casagrande, P. Collins, A. Ferrari, T. Villa, and A. Sangiovanni-Vincentelli. Ariadne: a framework for reachability analysis of hybrid automata. In *M.T.N.S.*, 2006.

[5] X. Chen, E. Abraham, and S. Sankaranarayanan. Taylor model flowpipe construction for non-linear hybrid systems. In *R.T.S.S.*, 2012.

[6] E. Clarke and P. Zuliani. Statistical model checking for cyber-physical systems. In *A.T.V.A.* 2011.

[7] A. Donzé. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *C.A.V.* 2010.

[8] A. Donzé and O. Maler. Systematic simulation using sensitivity analysis. *H.S.C.C*, 2007.

[9] P. S. Duggirala, T. Johnson, A. Zimmerman, and S. Mitra. Static and dynamic analysis of timed distributed traces. In *R.T.S.S.*, 2012.

[10] A. Girard. Verification using simulation. In *H.S.C.C.*, 2006.

[11] A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE T. A. C*, 2010.

[12] Z. Huang and S. Mitra. Computing bounded reach sets from sampled simulation traces. In *H.S.C.C.*, 2012.

[13] R. Jobredeaux, T. Wang, and E. Feron. Autocoding control software with proofs i: Annotation translation. In *D.A.S.C.*, 2011.

[14] A. A. Julius, G. E. Fainekos, M. Anand, I. Lee, and G. J. Pappas. Robust test generation and coverage for hybrid systems. In *H.S.C.C.*, 2007.

[15] A. Kanade, R. Alur, F. Ivancic, S. Ramesh, S. Sankaranarayanan, and K. Shashidhar. Generating and analyzing symbolic traces of simulink/stateflow models. In *C. A. V.* 2009.

[16] D. Liberzon. *Switching in Systems and Control*. 2003.

[17] W. Lohmiller and J. J. E. Slotine. On contraction analysis for non-linear systems. *Automatica*, 1998.

[18] K. Manamcheri, S. Mitra, S. Bak, and M. Caccamo. A step towards verification and synthesis from simulink/stateflow models. In *H.S.C.C*, 2011.

[19] N. Nedialkov. Vnode-lp: Validated solutions for initial value problem for odes. Technical report, 2006.

[20] T. Nghiem, S. Sankaranarayanan, G. Fainekos, F. Ivancic, A. Gupta, and G. Pappas. Monte-carlo techniques for falsification of temporal properties of non-linear hybrid systems. In *H.S.C.C*, 2010.

[21] S. Prajna, A. Papachristodoulou, P. Seiler, and P. Parrilo. Sostools: Sum of squares optimization toolbox for matlab. 2004.

[22] B. Sharma and I. Kar. Design of asymptotically convergent frequency estimator using contraction theory. *Automatic Control, IEEE Transactions on*, 2008.

[23] G. Wood and B. Zhang. Estimation of the lipschitz constant of a function. *Journal of Global Optimization*, 1996.

# APPENDIX

## Proofs from Section 4

PROOF. of Proposition 1: It is easy to check that the function $V(x_1, x_2) = |x_1 - x_2|$ satisfies the first two conditions in Definition 2 with $\alpha_1(|x_1 - x_2|) = \alpha_2(|x_1 - x_2|) = |x_1 - x_2|$. We fix a pair of initial states $x_1, x_2 \in \mathbb{R}^n$ and will show that for any $t \in \mathbb{R}_{\geq 0}$, $V(\xi(x_1, t), \xi(x_2, t)) \leq e^{Lt}|x_1 - x_2|$. For this we take the derivative of the function $V$ with respect to $t$:

$$\begin{aligned}
&\frac{d}{dt}(V(\xi(x_1,t),\xi(x_2,t))) = \frac{d}{dt}|\xi(x_1,t) - \xi(x_2,t)| \\
= \;& \frac{(\xi(x_1,t) - \xi(x_2,t))}{|\xi(x_1,t) - \xi(x_2,t)|}(\frac{d}{dt}\xi(x_1,t) - \frac{d}{dt}\xi(x_2,t)) \\
\leq \;& (\frac{d}{dt}\xi(x_1,t) - \frac{d}{dt}\xi(x_2,t)) \\
= \;& (f(\xi(x_1,t),t) - f(\xi(x_2,t),t)) \\
\leq \;& L|\xi(x_1,t) - \xi(x_2,t)| = L\,V(\xi(x_1,t),\xi(x_2,t)).
\end{aligned}$$

We have that for any pair of initial states $x_1, x_2$, $\dot{V} \leq L\,V$ which leads to the bound on $V$ which grows exponentially with time $t$:

$V(\xi(x_1,t),\xi(x_2,t)) \leq e^{Lt}V(\xi(x_1,0),\xi(x_2,0)) = e^{Lt}|x_1 - x_2|$.

Hence $\beta(x_1, x_2, t) = e^{Lt}V(x_1, x_2)$. □

PROOF. of Proposition 3: It is clear that the function $V(x_1, x_2)$ satisfies the first two conditions in Definition 2 with $\alpha_1(|x_1 - x_2|) = \alpha_2(|x_1 - x_2|) = (|x_1 - x_2|)^2 = (x_1 - x_2)^T(x_1 - x_2)$. From Theorem 2, we have that

$$\exists k, \gamma > 0, \forall x_1, x_2 \in \mathbb{R}^n, \delta x^T \delta x \leq k\delta x_0^T \delta x_0 e^{-\gamma t}$$

where $\delta x = \xi(x_1, t) - \xi(x_2, t)$ and $\delta x_0 = x_1 - x_2$. Hence, $\beta(x_1, x_2, t) = k(x_1 - x_2)^T(x_1 - x_2)e^{-\gamma t}$. □

PROOF. of Proposition 5: Equation (7) of Theorem 4 gives $\alpha_1$ and $\alpha_2$ for Definition 2. Further, from Equation 8, we have that

$$\begin{aligned}
V(\xi(x_1,t),\xi(x_2,t)) \leq\; & V(x_1,x_2) \\
& + \int_0^t -\alpha(|\xi(x_1,\tau) - \xi(x_2,\tau)|)d\tau, \qquad (11)
\end{aligned}$$

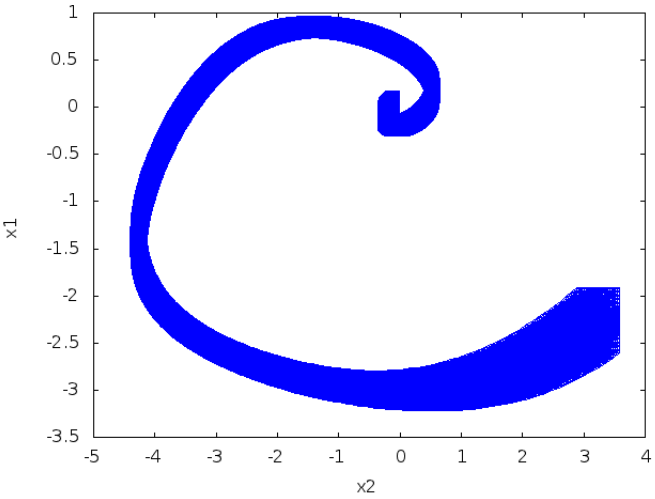
where $\alpha$ is the function obtained from Equation (8). It can be observed that since $\alpha \in \mathcal{K}$, and $V$ is a trajecory metric with witness $(\alpha_1, \alpha_2, \beta)$ where $\beta(x_1, x_2, t) = V(x_1, x_2) + \int_0^t -\alpha(|\xi(x_1, \tau) - \xi(x_2, \tau)|)d\tau$. □

## Proofs from Section 6

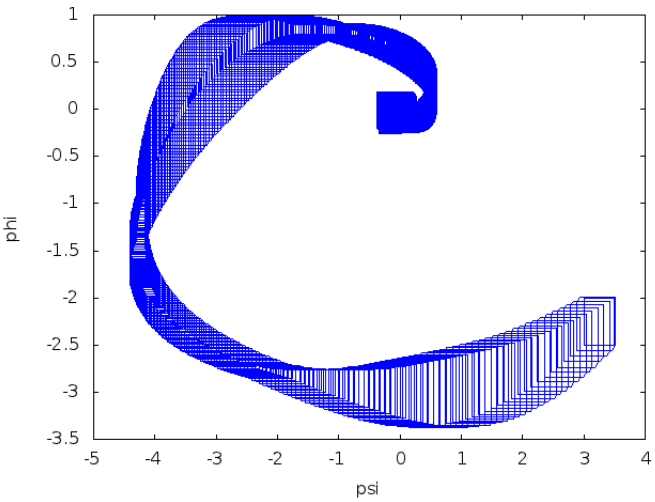
PROOF. of Proposition 7: From Definition 2, we know that $V(\xi(x,t),\xi(x_0,t)) \leq \beta(x, x_0, t) \leq \epsilon$. Hence $\xi(x,t) \in B_\epsilon(\xi(x_0,t))$. □

PROOF. of Proposition 8: Since $\psi$ is a $(x_0, T_{bound}, \epsilon, \tau)$-simulation trace, from Definition 6, we have that $\xi(x_0, t) \in \rho$. From Proposition 7, we have that $\forall x \in B_\delta(x_0), \xi(x,t) \in B_\epsilon^V(\xi(x_0,t))$ and hence $\xi(x,t) \in B_\epsilon^V(\psi)$. □

PROOF. of Proposition 10: Let the set of initial states be $I$. Observe that $\Theta$ is an $\omega$-approximation of the initial set of states $I \subseteq \cup_{x\in\mathcal{X}} B_{\omega+\delta}(x)$. Thus, the $\epsilon'$ bloating of the simulations, i.e. $\cup_{x\in\mathcal{X}} B_{\epsilon'}^V(\psi)$ contains all the set of reachable states from $I$. Further, since the tubes in line 9 are $B_\epsilon^V(\psi)$ and the width of each of the region in $\psi$ is utmost $\epsilon$, we can conclude that the collection of tubes is utmost $max\{2\epsilon + \epsilon'\}$-approximation of the reachable set of states from $I$. □




(a) 2D representation of reachable set by C2E2.

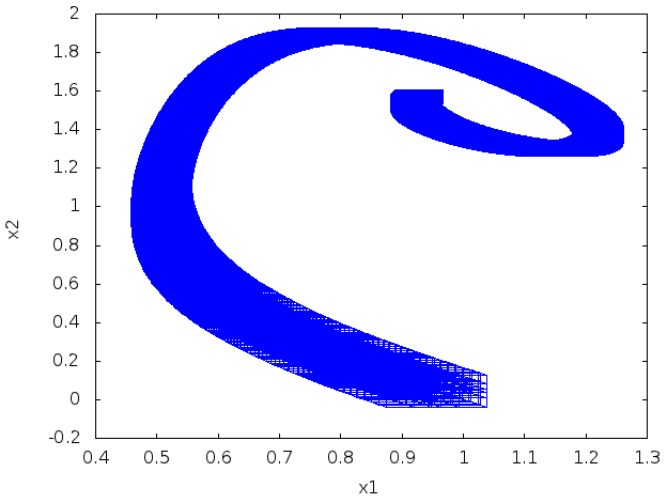



(b) 2D representation of reachable set by $Flow^*$.

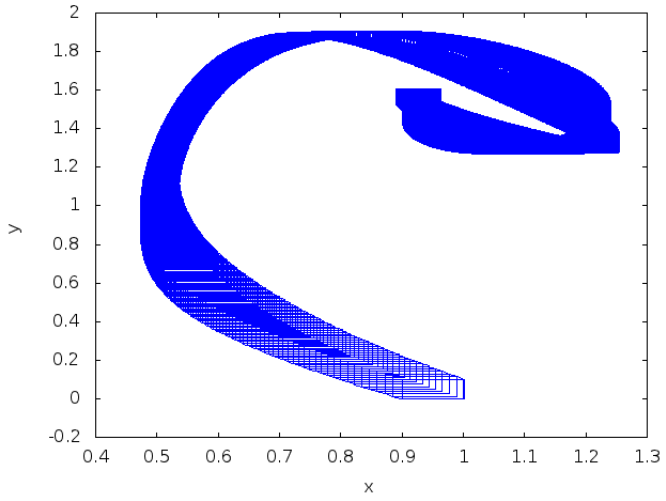
**Figure 4: Graphical representation of reachable set for Moore Greitzer Jet Engine**

### Plotting the Set of Reachable States by $Flow^*$ and C2E2

We now show the 2D representation of the set of reachable states plotted by $Flow^*$ and C2E2 . We consider two of the benchmark examples Brussellator and Moore Greitzer Jet Engine.




(a) 2D representation of reachable set by C2E2.




(b) 2D representation of reachable set by $Flow^*$.

**Figure 5: Graphical representation of reachable set for Brussellator**