Bounded Verification with On-the-Fly Discrepancy Computation [⋆]

Chuchu Fan and Sayan Mitra

{cfan10,mitras}@illinois.edu
Department of Electrical and Computer Engineering,
University of Illinois at Urbana-Champaign.

Abstract. Simulation-based verification algorithms can provide formal safety guarantees for nonlinear and hybrid systems. The previous algorithms rely on user provided model annotations called discrepancy function, which are crucial for computing reachtubes from simulations. In this paper, we eliminate this requirement by presenting an algorithm for computing piecewise exponential discrepancy functions. The algorithm relies on computing local convergence or divergence rates of trajectories along a simulation using a coarse over-approximation of the reach set and bounding the maximal eigenvalue of the Jacobian over this over-approximation. The resulting discrepancy function preserves the soundness and the relative completeness of the verification algorithm. We also provide a coordinate transformation method to improve the local estimates for the convergence or divergence rates in practical examples. We extend the method to get the input-to-state discrepancy of nonlinear dynamical systems which can be used for compositional analysis. Our experiments show that the approach is effective in terms of running time for several benchmark problems, scales reasonably to larger dimensional systems, and compares favorably with respect to available tools for nonlinear models.

1 Introduction

Verifying and falsifying nonlinear, switched, and hybrid system models using numerical simulations have been studied in detail [11,20,4,17,9]. The bounded time safety verification problem for a given model is parameterized by a time bound, a set of initial states, and a set of unsafe states and it requires one to decide if there exists a behavior of the model that reaches any unsafe set from any initial state. The simulation-based procedure for this problem first generates a set of numerical approximations of the behaviors from a finite sampling of the initial states. Next, by bloating these simulations by an appropriately large factor it computes an over-approximation of the reachable states from the initial set. If this over-approximation proves safety or produces a counter-example, then the algorithm decides, otherwise, it draws more samples of initial states and repeats the earlier steps to compute a more precise over-approximation. With post-processing of the reachtube over-approximations this basic procedure can be utilized to verify termporal precedence [12] and richer classes of properties [7].

[⋆] The results presented here came about from work supported and funded by the National Science Foundation (NSF CSR 1016791) and the Air Force Office of Scientific Research (AFOSR YIP FA9550-12-1-0336).

In order to make this type of procedure sound, the bloating factor should be chosen to be large. Specifically, it should be large enough to make each bloated simulation an over approximation of the reachable states of the system not only from the sampled initial state, but also from a large enough neighborhood of that state so that the union of these neighborhoods cover the entire set of initial states. On the other hand, to make the procedure complete, or at least relatively complete modulo the precision of the machine, it should be possible to make the error due to bloating arbitrarily small for any point in time. These two opposing requirements are captured in the definition of a *discrepancy function* of [11]: For an n-dimensional dynamical system, it is any function $\beta : \mathbb{R}^{2n} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, such that (a) it gives an upper-bound on the distance between any two trajectories $\xi(x,t)$ and $\xi'(x,t)$ of the system $|\xi(x,t) - \xi(x',t)| \leq \beta(x,x',t)$, and (b) it vanishes as x approaches x'. Simply using the Lipschitz constant of the dynamic function gives one such bound, but it grows exponentially with time even for some incrementally stable models [2].

In [11], it is observed that the notion of a contraction metric [23] gives a much tighter bound and it provided heuristics for finding them for some classes of polynomial systems. Sensitivity analysis approaches give strict error bounds for linear systems [9], but for nonlinear models the bounds are less clear. We present a more detailed overview of related work in Section 2. This paper fills this gap by providing a subroutine that computes a local version of the discrepancy function which turns out to be adequate and effective for sound and relatively complete simulation-based verification of more general nonlinear systems. This subroutine, $ComputeLDF$, itself uses a Lipschitz constant and the Jacobian of the dynamic function (the right hand side of the differential equation) and simulations of the system. The Lispchitz constant is used to construct a coarse, one-step over-approximation of the reach set of the system along a simulation. Then it computes an upper bound on the maximum eigenvalue of the symmetric part of the Jacobian over this over approximation, using a theorem from matrix perturbation theory. This gives an exponential bound on the distance between two trajectories, but roughly, the exponent is the best it can be as it is close to the maximum eigenvalue of the linear approximation of the system in the neighborhood. A key advantage of this scheme is that it is not necessary for the underlying system to have any special properties (such as incremental stability). In practice, $ComputeLDF$ has been used successfully in [10,13], [21] to find bloating factors for safety verification of powertrain control model proposed by Toyota Research group as a challenge problem, and the model of a neural circuit in a multicellular organism.

In this paper, we also propose two practical extensions of this approach. First, we show that a linear coordinate transformation can bring about exponential improvements in the estimated distance. Secondly, we propose a technique for computing input-to-state discrepancy functions for analyzing composed systems and systems with bounded nondeterministic inputs. We report the results from a number of experiments performed with a prototype implementation of this approach applied to safety verification.

2 Related Work

Most existing techniques for formal verification rely on performing *reachability analysis*, which is a way of computing an overapproximation of the set of states that the system can reach over a given time horizon [15]. In [1] the authors provide an approach by linearizing the nonlinear system locally, and bounding the linearization error by Lagrange remainders. In [6], Taylor model is employed to analyze nonlinear hybrid systems. A pervasive challenge in such techniques is that the analysis can be too conservative, and faces the curse of dimensionality.

Simulation based verification has been studied in several papers recently [9,1,7,22]. Sensitivity analysis [9,7] is a technique to systematically simulate arbitrary continuous systems and nonlinear systems with inputs. The novelty of their approach consist in the use of the *sensitivity matrix* : a matrix that captures the sensitivity of the system to its initial condition x_0. This is then used to give an upper bound on the distance between two system trajectories. In [18] the authors present a convenient implementation of sensitivity analysis in the Simulink software. However, this approach cannot formally prove safety, as a quadratic error term is ignored when computing this upper bound. In [22], the authors provide methods to overapproximate the distance between trajectories; however, these methods are mainly applicable to affine and polynomial systems. In contrast, in Section 4 and Section 5 we have provided a strict over-approximation of Lipschitz continuous systems with respect to uncertainty in the initial conditions and uncertainty in the input signals.

The idea of discrepancy function can be seen as a generalization of the incremental stability [2]. An incremental Lyapunov function-based approach is used in [17]. The authors construct a finite symbolic model that is approximately bisimilar to the original switched system. Also in [27], incremental Lyapunov is used to compute finite sound abstractions of control systems without stability assumption. In this paper, our approach bypasses the incremental stability requirement by focusing on the bounded time verification, and we address on computing a good distance between trajectories.

Contraction in [23] is defined as the region in which the eigenvalues of the symmetric part of the Jacobian is uniformly negative. Contraction metrics introduced in [23] is also used in [11] to perform sound and relative complete analysis of nonlinear systems. We extend this idea of contraction by proving that given a region S, the maximum eigenvalue of the symmetric part of the Jacobian matrix over compact set S provides an exponential bound on the distance between any two trajectories in S.

3 Background

For a vector $x \in \mathbb{R}^n$, $\|x\|$ is the l^2-norm of x and x_i denotes its i^{th} component. For $\delta \geq 0$, $B_\delta(x) = \{x' \in \mathbb{R}^n \mid ||x' - x|| \leq \delta\}$. For a set $S \subseteq \mathbb{R}^n$, $B_\delta(S) = \cup_{x\in S} B_\delta(x)$. Let $S \oplus B_\delta(0)$ represents the Minkowski sum of S and $B_\delta(0)$. Therefore, $S \oplus B_\delta(0) = B_\delta(S)$. For sets $S_1, S_2 \subseteq \mathbb{R}^n$, $hull(S_1, S_2)$ is their convex hull. The diameter of a compact set S is $dia(S) = \sup_{x_1,x_2 \in S} \|x_1 - x_2\|$.

A continuous function $f : \mathbb{R}^n \to \mathbb{R}$ is *smooth* if all its higher derivatives and partial derivatives exist and are also continuous. It has a Lipschitz constant $L \geq 0$ if for every $x, x' \in \mathbb{R}^n$, $||f(x) - f(x')|| \leq L||x - x'||$. A function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a *class $\mathcal{K}$ function* if it is continuous, strictly increasing, and $f(0) = 0$.

Given a differentiable vector-valued function $f : \mathbb{R}^n \times \mathbb{R}^{\geq 0} \to \mathbb{R}^n$, the *Jacobian* J_f of f is the matrix-valued function of all the first-order partial derivatives of f. Let $f_i, i = 1 \ldots n : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ be the scalar components of f. The Jacobian of f is: $(J_f(x))_{ij} = \frac{\partial f_i(x)}{\partial x_j}$. The *symmetric part of the Jacobian of f* matrix is defined as $\frac{1}{2}(J_f(x) + {J_f}^T(x))$.

For an $n \times n$ matrix A, $\|A\|$ represents the l^2-norm of A: $\|A\| = \sqrt{\lambda_{\max}(A^H A)}$. A^H is the *conjugated transpose* of A, and $\lambda_{\max}(A)$ is the largest eigenvalue of matrix A. When A is a real matrix, we write it as A^T. If $\forall x \in \mathbb{R}^n$, $x^T A x \leq 0$, then we say A is negative-semidefinite, and write $A \preceq 0$. We write $A \preceq B$ if $A - B \preceq 0$.

Safety Verification Problem. Consider an *autonomous dynamical system*:

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

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is a Lipschitz continuous function. A *solution or a trajectory* of the system is a function $\xi : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ such that for any initial point $x_0 \in \mathbb{R}^n$ and at any time $t \geq 0$, $\xi(x_0, t)$ satisfies the differential equation (1).

The *bounded-time safety verification problem* is parameterized by: (a) an n-dimensional dynamical system, that is, the function f defining the right hand side of its differential equation, (b) a compact set $\Theta \subseteq \mathbb{R}^n$ of initial states, (c) an open set $\mathbb{U} \subseteq \mathbb{R}^n$ of unsafe states, and (d) a time bound $T > 0$. A state x in $\mathbb{R}^n$ is *reachable from Θ within a time interval* $[t_1, t_2]$ if there exists an initial state $x_0 \in \Theta$ and a time $t \in [t_1, t_2]$ such that $x = \xi(x_0, t)$. The set of all reachable states in the interval $[t_1, t_2]$ is denoted by $\mathsf{Reach}(\Theta, [t_1, t_2])$. If $t_1 = 0$ then we write $\mathsf{Reach}(t_2)$ when set Θ is clear from the context. Given a bounded-time safety verification problem, we would like to design algorithms for deciding if any reachable state is safe, that is, if $\mathsf{Reach}(T) \cap \mathbb{U} = \varnothing$. If there exists some $\epsilon > 0$ such that $B_\epsilon(\mathsf{Reach}(T)) \cap \mathbb{U} = \varnothing$, we say the system is robustly safe. A sequence of papers [11,12,9] presented algorithms for solving this problem for a broad class of nonlinear dynamical, switched, and hybrid systems. In the remainder of this section, we present an overview of this approach. (Algorithm 1).

Simulations, Reachtubes and Annotations. The algorithm uses simulation oracles that give sampled numerical simulations of the system from individual initial states.

Definition 1. *A* (x_0, τ, ϵ, T)-simulation *of the system described in Equation (1) is a sequence of time-stamped sets* (R_0, t_0), $(R_1, t_1) \ldots, (R_n, t_n)$ *satisfying:*

(1) Each R_i is a compact set in $\mathbb{R}^n$ with $dia(R_i) \leq \epsilon$.
(2) The last time $t_n = T$ and for each i, $0 < t_i - t_{i-1} \leq \tau$, where the parameter τ is called the sampling period.
(3) For each t_i, the trajectory from x_0 at t_i is in R_i, i.e., $\xi(x_0, t_i) \in R_i$, and for any $t \in [t_{i-1}, t_i]$, the solution $\xi(x_0, t) \in hull(R_{i-1}, R_i)$.

Simulation engines generate a sequence of states and error bounds using numerical integration. Libraries like CAPD [5] and VNODE-LP [24] compute such simulations for a wide range of nonlinear dynamical system models and the R_i's are represented by some data structure like hyperrectangles.

Closely related to simulations are *reachtubes*. For a set of states $D \subseteq \mathbb{R}^n$, a (D, τ, T)-*reachtube* of (1) is a sequence of time-stamped sets $(R_0, 0), (R_1, t_1) \ldots, (R_n, t_n)$ satisfying:

(1) Each $R_i \subseteq \mathbb{R}^n$ is a compact set of states.
(2) The last time $t_n = T$ and for each i, $0 \leq t_i - t_{i-1} \leq \tau$.
(3) For any $x_0 \in D$, and any time $t \in [t_{i-1}, t_i]$, the solution $\xi(x_0, t) \in R_i$.

A reachtube is analogous to a simulation from a set of states, but they are much harder to compute. In fact, an algorithm for computing exact reachtubes readily solves the safety verification problem.

The algorithms in [11,19] require the user to decorate the model with annotations called *discrepancy functions* for computing reachtubes.

Definition 2. *A continuous function $\beta : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a* discrepancy function *of the system in Equation (1) if*

(1) for any $x, x' \in \mathbb{R}^n$, and any time $t > 0$, $\|\xi(x,t) - \xi(x',t)\| \leq \beta(x,x',t)$, and
(2) for any t, as $x \to x'$, $\beta(.,.,t) \to 0$,
(3) $\forall \epsilon > 0, \forall x, x' \in \mathbb{R}^n, \exists \delta$ such that for any time t, $\|x - x'\| < \delta \Rightarrow \beta(x,x',t) < \epsilon$.

If the function β meets the two conditions for any pair of states x, x' in a compact set K instead of $\mathbb{R}^n$ in both condition (1) and (3), then it is called a *K-local discrepancy function.*

The annotation β gives an upper bound on the distance between two neighboring trajectories as a function of their initial states and time. Unlike incremental stability conditions [2], the second condition on β does not require the trajectories to converge as time goes to infinity, but only as the initial states converge. Obviously, if the function f has a Lipschitz constant L, then $\beta(x,x',t) = ||x - x'||e^{Lt}$ meets the above criteria. In [11,19] other heuristics have been proposed for finding discrepancy functions. As will be clear from the following discussion, the quality of the discrepancy function strongly influences the performance of the simulation-based verification algorithm. [11,19,20] need user provided discrepancy and simulation engines to give verification of bounded time safety and temporal precedence properties. In this paper, we will present approaches for computing *local discrepancy functions* that unburdens the user from finding these annotations.

3.1 Verification Algorithm

The simulation-based verification algorithm is shown in Algorithm 1. It takes as input some finite description of the parameters of a safety verification problem, namely, the function f, the initial set Θ, the unsafe set $\mathbb{U}$, and the time bound T. It has two main data stuctures: The first, $\mathcal{C}$ returned by function $Partition$, is a collection of triples $\langle \theta, \delta, \epsilon \rangle$ such that the union of all the δ-balls around the θ's completely cover the initial set Θ. The second data structure $\mathcal{R}$ incrementally gets the bounded-time reachtube from Θ.

Initially, $\mathcal{C}$ has a singleton cover $\langle \theta_0, \delta_0, \epsilon_0 \rangle$ such that $\delta_0 = dia(\Theta)$, $\Theta \subseteq B_{\delta_0}(\theta_0)$, and ϵ_0 is a small constant for simulation precision.

In the **while**-loop, this verification algorithm iteratively refines the cover of Θ and for each $\langle \theta, \delta, \epsilon \rangle$ in $\mathcal{C}$, computes over-approximations of the reachtube from $B_\delta(\theta)$. The higher-level structure of the algorithm is familiar: if the reachtube from $B_\delta(\theta)$ proves to be safe, i.e., disjoint from $\mathbb{U}$, then the corresponding triple is removed from $\mathcal{C}$ (Line 10). If part of the simulation from θ overlaps with $\mathbb{U}$, then the system is declared to be unsafe (Line 12). Otherwise, a finer cover of $B_\delta(\theta)$ is created, and the corresponding triples with finer parameters are added to $\mathcal{C}$.

Here we discuss the reachtubes computed from discrepancy and simulations. For each $\langle \theta, \delta, \epsilon \rangle$ in $\mathcal{C}$, a $(\theta, \tau, \epsilon, T)$-simulation ψ, which is a sequence of $\{(R_i, t_i)\}$, is generated. Note that ψ contains the trajectory from θ, $\xi(\theta, t), t \in [0, T]$. Then we bloat each R_i by some factor (Line 7) such that the resulting sequence contains the reachtube from $B_\delta(\theta)$. It is shown that this bloated simulation is guaranteed to be an over-approximation of $\mathsf{Reach}(B_\delta(\theta), T)$ and the union of these bloated simulations is an over-approximation of $\mathsf{Reach}(\Theta, T)$. Therefore, the algorithm is sound. Furthermore, the second property of β ensures that the reach set over-approximations become tighter and tighter as we make δ smaller and smaller. Finally it will return "SAFE" for robustly

Algorithm 1: Verification Algorithm

```
Input: $\Theta, \mathbb{U}, T$
1  $\delta \leftarrow dia(\Theta)$; $\epsilon \leftarrow \epsilon_0$; $\mathcal{C} \leftarrow \varnothing$; $\mathcal{R} \leftarrow \varnothing$; // $epsilon_0$ is a small constant
2  $\mathcal{C} \leftarrow \langle Partition(\Theta, \delta), \delta, \epsilon\rangle$
3  while $\mathcal{C} \neq \varnothing$ do
4      foreach $(\theta, \delta, \epsilon) \in \mathcal{C}$ do
5          $\psi = \{(R_i, t_i)\}_{i=1}^{k} \leftarrow Simulate(\theta, \tau, \epsilon, T)$
6          $\beta \leftarrow ComputeLDF(\psi, J_f, L_f, \delta, \epsilon)$
7          $\mathrm{D} \leftarrow \psi \oplus \beta$
8          if $D \cap \mathbb{U} = \varnothing$ then
9              $\mathcal{C} \leftarrow \mathcal{C} \backslash \{(\theta, \delta, \epsilon)\}$
10             $\mathcal{R} \leftarrow \mathcal{R} \cup D$
11         else if $\exists k, R_k \subseteq \mathbb{U}$ then
12             return (UNSAFE, $\mathcal{R}$)
13         else
14             $\mathcal{C} \leftarrow \mathcal{C} \backslash \{(\theta, \delta, \epsilon)\}$
15             $\mathcal{C} \leftarrow \mathcal{C} \cup Partition(\Theta \cap B_\delta(\theta), (\frac{\delta_1}{2}, \ldots, \frac{\delta_N}{2}), \frac{\epsilon}{2})$
16 return (SAFE, $\mathcal{R}$)
```

safe reachtubes or find a counter example and return "UNSAFE". For user defined discrepancy function, the factor is obtained by maximizing $\beta(\theta, \tilde{\theta}, t)$ over $\tilde{\theta} \in B_\delta(\theta)$ and $t \in [t_{i-1}, t_i]$. Indeed this is the approach taken in the algorithm presented in [11]. In this paper, we will analyze in detail the *ComputeLDF* subroutine which computes a local version of discrepancy function automatically. The following results from [11] state two key properties of the algorithm. Although in [11] β was defined globally, it is easy to check that the local version still satisfies them.

Theorem 1. *Algorithm 1 is sound, i.e., if it returns "SAFE" then the system is safe; when it returns "UNSAFE" there exists at least one execution from Θ that is unsafe. It is relatively complete, i.e., if the system is robustly safe, it will terminate and return "SAFE". If any executions from Θ is unsafe, it will terminate and return "UNSAFE".*

4 Local discrepancy function

In this section, we present the *ComputeLDF* algorithm and its analysis. This algorithm computes a special type of local discrepancy in terms of time-varying exponential functions that bound from above the distance between two trajectories starting from a compact neighborhood. Roughly speaking, it computes the rate of trajectory convergence or divergence for an interval of time instances.

Definition 3. *Consider a compact set $C \subseteq \mathbb{R}^n$ and a sequence of time points $0 = t_0 < t_1 < t_2 < \ldots < t_k = T$. For $\forall x_1, x_2 \in C, \forall t \in [0, T]$, a* piecewise exponential discrepancy function *$\beta : C \times C \times [0, T] \to \mathbb{R}_{\geq 0}$ is defined as:*

$$\beta(x_1, x_2, t) = \begin{cases} \|x_1 - x_2\|, & \text{if } t = t_0, \\ \beta(x_1, x_2, t_{i-1}) e^{b[i](t - t_{i-1})}, & \text{if } t \in (t_{i-1}, t_i], \end{cases} \tag{2}$$

Algorithm 2: Algorithm $ComputeLDF$.

```
Input: $\psi = \{(R_i, t_i)\}_{i=1}^{k}$,$J_f$,$L_f$,$\delta$, $\epsilon$
1  $\Delta \leftarrow \delta$,$b \leftarrow$ zeros(k)
2  for $i$ = $1{:}k$ do
3      $\tau \leftarrow t_i - t_{i-1}$
4      $d \leftarrow (\Delta + \epsilon)e^{L_f \tau}$
5      S $\leftarrow hull(R_{i-1}, R_i) \oplus B_d(0)$
6      J $\leftarrow J_f(center(S))$
7      $\lambda \leftarrow \max(eig(J + J^T)/2)$
8      error $\leftarrow$ upper $_{x \in S}\|(J_f(x) + J_f^T(x)) - (J + J^T)\|$
9      $b[i] \leftarrow \lambda$ + error/2
10     $\Delta \leftarrow (\Delta + \epsilon)e^{b[i]\tau}$
11 return $b$
```

where $b[1], \ldots, b[k]$ *are real constants.*

From the definition, we can immediately get that $\beta(x_1, x_2, t) = ||x_1 - x_2||e^{b[i](t-t_{i-1})+\sum_{j=1}^{i-1} b[j](t_j - t_{j-1})}$, $i = 1, \ldots, k$, where t_{i-1} is the largest time point in the sequence before t.

4.1 ComputeLDF Algorithm

Algorithm 2 shows the pseudocode for $ComputeLDF$ used in Line 6 of the verification algorithm. $ComputeLDF$ takes as input a parameter δ, an error bound for simulation ϵ, the Lipschitz constant L_f, the Jacobian matrix J_f of function f, and a $(\theta, \tau, \epsilon, T)$-simulation $\psi = \{(R_i, t_i)\}, i = 0, 1, \ldots, k$. It computes a piecewise exponential local discrepancy function (LDF) for the compact set $B_\delta(R_0)$ and for the time points $t_0, \ldots, t_k$. and returns it as an array of exponential coefficients b.

The algorithm starts with the initial set $B_\delta(R_0)$ and with $\Delta = \delta$. In each iteration of the **for**-loop it computes exponent $b[i]$ corresponding to the time interval $[t_{i-1}, t_i]$. In the i^{th} iteration, Δ is updated so that $B_\Delta(R_{i-1})$ is an over-approximation of the reachable states from $B_\delta(R_0)$ at t_{i-1} (Lemma 5). In Lines 4-5, a set S is computed by bloating the convex hull $hull(R_{i-1}, R_i)$ by a factor of $d = (\Delta + \epsilon)e^{L_f(t_i - t_{i-1})}$. The set S will later be proved to be a (coarse) over-approximation of the reachtube from $B_\Delta(R_{i-1})$ over the time interval $[t_{i-1}, t_i]$ (Lemma 1). In Lines 6-9 an upper bound on the maximum eigenvalue of the symmetric part of the Jacobian over the set S, is computed as $b[i]$ (Lemma 3). Then Δ is updated as $(\Delta + \epsilon)e^{b[i](t_i - t_{i-1})}$ for the next iteration.

4.2 Analysis of ComputeLDF

In this section, we will prove that $ComputeLDF(\psi, J_f, L_f, \delta, \epsilon)$ returns a piecewise exponential LDF of the system in Equation (1), for the compact neighborhood $B_\delta(R_0)$, and the sequence of the time points in the simulation ψ. We establish some lemmas to prove the main theorem. The complete proof of the lemmas can be found in the technical report [14]. First, we show in Lemma 1 that in the i^{th} iteration of the loop, the computed S is an over-approximation of the set of states that can be reached by the system from $B_\Delta(R_{i-1})$ over the time interval $[t_{i-1}, t_i]$.

Lemma 1. *In the i^{th} iteration of ComputeLDF,* $\mathsf{Reach}(B_\Delta(R_{i-1}),[t_{i-1},t_i]) \subseteq S$.

Lemma 1 shows that using Lipschitz constant and Gronwall's inequality we can get an one step over-approximation of the reachtube.

Next, using the generalized mean value theorem (Lemma 2), we get that in the i^{th} iteration, the computed $b[i]$ in Line 9 is the exponential divergence (if positive) or convergence (negative) rate of the distance between any two trajectories starting from $B_\Delta(R_{i-1})$ over time $[t_{i-1},t_i]$.

Lemma 2. *For any continuously differentiable vector-valued function $f : \mathbb{R}^n \to \mathbb{R}^n$, and $x, r \in \mathbb{R}^n$, $f(x+r) - f(x) = \left(\int_0^1 J_f(x+sr)ds\right) \cdot r$, where the integral is component-wise.*

Next, we will use a well-known theorem that gives bounds on eigenvalues of perturbed symmetric matrices, the proof of which uses the Courant-Fischer minimax theorem.

Theorem 2. *If A and E are $n \times n$ symmetric matrices, then*

$$\lambda_n(E) \le \lambda_k(A+E) - \lambda_k(A) \le \lambda_1(E),$$

where $\lambda_i(\cdot)$ is the i^{th} largest eigenvalue of a matrix.

Corollary 1. *If A and E are $n \times n$ symmetric matrices, then*

$$|\lambda_k(A+E) - \lambda_k(A)| \le \|E\|. \tag{3}$$

Since A is symmetric, $\|A\| = \sqrt{\lambda_{\max}(A^T A)} = \max(|\lambda(A)|)$. From Theorem 2, we have $|\lambda_k(A+E) - \lambda_k(A)| \le \max\{|\lambda_n(E)|, |\lambda_1(E)|\} = \|E\|$. If $E(x)$ is a matrix-valued function: $\mathbb{R}^n \to \mathbb{R}^{n\times n}$ maps a state $x \in \mathbb{R}^n$ to a matrix $E(x)$, and every component of $E(x)$, $e_{ij}(x) : \mathbb{R}^n \to \mathbb{R}$ is continuous over some compact closed set S, then we can get an upper bound of $\|E(x)\|$ over S by compute the upper bound of the absolute value of each term $e_{ij}(x)$, $|e_{ij}(x)|$ over S. Let $\text{upper}_{x\in S}(|e_{ij}(x)|)$ be denoted by $\tilde{e}_{ij}$, then we know $\forall x \in S$, $\|E(x)\| \le \sqrt{\sum_{i=1}^n \sum_{j=1}^n \tilde{e}_{ij}^2}$. Because we assume the system to be Lipschitz continuous, the upper bound of the symmetric part of the Jacobian matrix in Line 8 always exists. Using Corollary 1, we next show in Lemma 3 that $b[i]$ calculated in Line 9 bounds the eigenvalues of the symmetric part of Jacobian matrix over S.

Lemma 3. *In the i^{th} iteration, for $\forall x \in S : J_f^T(x) + J_f(x) \preceq 2b[i]I$.*

Proof. Let S be the set computed in Line 5 and J be the Jacobian evaluated at the center s_0 of S. Consider any point $x \in S$. We define the perturbation matrix $E(x) \equiv J_f^T(x) + J_f(x) - (J^T + J)$. Since $J_f^T(x) + J_f(x)$ and $J^T + J$ are symmetric matrices, Corollary 1 implies that $\lambda_{max}(J_f^T(x) + J_f(x)) - \lambda_{max}(J^T + J) \le ||E(x)||$. The $error$ term computed in Line 8 is the upperbound on $||E(x)||$. Therefore, $\lambda_{max}(J_f^T(x) + J_f(x)) \le \lambda_{max}(J^T + J) + error$. In Line 9 set $b[i]$ equals to $\lambda_{max}((J^T + J)/2) + error/2$. Thus, $\lambda_{max}(J_f^T(x) + J_f(x)) \le 2b[i]$, which immediately indicates that $\forall x \in S$: $J_f^T(x) + J_f(x) \preceq 2b[i]I$.

By Lemma 2 and Lemma 3, we can prove as in Lemma 4 that $b[i]$ calculated in Line 9 is the exponential rate of divergence or convergence of two trajectories starting from $B_\Delta(R_{i-1})$ over the interval $[t_{i-1},t_i]$.

Lemma 4. *In the i^{th} iteration, for any two states $x_1, x_2 \in B_\Delta(R_{i-1})$ at time t_{i-1}, and any time $t \in [t_{i-1}, t_i]$, $\|\xi(x_1,t) - \xi(x_2,t)\| \leq \|x_1 - x_2\| e^{b[i](t-t_{i-1})}$.*

This lemma can be deduced using Lemma 2.

Up to this point all the lemmas were statements about a single iteration of the **for** -loop, next we show that in the i^{th} iteration of the loop, $B_\Delta(R_i)$ used in Lemma 1 and 4 is the reach set from $B_\delta(R_0)$ at time t_i.

Lemma 5. *For $\forall i = 1, \ldots, k$, $\mathsf{Reach}(B_\delta(R_0), [t_i, t_i]) \subseteq B_{\Delta_i}(R_i)$, and $\mathsf{Reach}(B_{\Delta_{i-1}}(R_{i-1}), [t_{i-1}, t_i]) \subseteq hull(R_{i-1}, R_i) \oplus B_{\Delta'_i}(0)$, where Δ_i is Δ after Line 10 is executed in the i^{th} iteration, and $\Delta'_i = \max\{\Delta_i, \Delta_{i-1} + \epsilon\}$.*

Proof. The lemma is proved by induction on i. First, we know that when $i = 1$, $\mathsf{Reach}(B_\delta(R_0), [t_0, t_0]) = B_\delta(R_0) = B_{\Delta_0}(R_0)$. Then we will prove that

$$\mathsf{Reach}(B_\delta(R_0), [t_0, t_1]) \subseteq hull(R_0, R_1) \oplus B_{\max\{\Delta_1, \Delta_0 + \epsilon\}}(0),$$

using Lemma 4.

Assuming that the lemma holds for $i = m-1$, we have $\mathsf{Reach}(B_\delta(R_0), [t_{m-1}, t_{m-1}]) \subseteq B_{\Delta_{m-1}}(R_{m-1})$. And we will prove that the lemma holds for $i = m$ as well.

$\cup_{i=1}^{k}\{hull(R_{i-1}, R_i) \oplus B_{\Delta'_i}(0)\}$ contains the $(B_\delta(R_0), \tau, T)$-reachtube of the system. Line 7 of Algorithm 1 is computed in this way. Now we are ready to prove the main theorem.

Theorem 3. *The items in array b computed by $ComputeLDF$ are the coefficients of a $B_\delta(R_0)$-local piecewise exponential discrepancy function (Definition 3).*

Proof. First of all consider any time $t \in [t_0, t_1]$ and any two states: $x_1, x_2 \in B_\delta(R_0)$. By Lemma 4, $\|\xi(x_1,t) - \xi(x_2,t)\| \leq \|x_1 - x_2\| e^{b[1](t-t_0)}$. Then consider $t \in [t_1, t_2]$. By Lemma 5 we know at time t_1, $\xi(x_1, t_1)$ and $\xi(x_2, t_1)$ are all contained in $B_{\Delta_1}(R_1)$, so we can use Lemma 4 such that for any time $t \in [t_1, t_2]$, $\|\xi(x_1,t) - \xi(x_2,t)\| \leq \|\xi(x_1,t_1) - \xi(x_2,t_1)\| e^{b[2](t-t_1)} \leq \|x_1 - x_2\| e^{b[2](t-t_1) + b[1](t_1 - t_0)}$.

The procedure above can be performed iteratively as follows. For any time $t \in [t_{i-1}, t_i]$, by lemma 5 we know at time t_{i-1}, $\xi(x_1, t_{i-1})$ and $\xi(x_2, t_{i-1})$ are all contained in $B_{\Delta_{i-1}}(R_{i-1})$. By Lemma 4 it follows that

$$\begin{aligned}\|\xi(x_1,t) - \xi(x_2,t)\| &\leq \|\xi(x_1,t_{i-1}) - \xi(x_2,t_{i-1})\| e^{b[i](t-t_{i-1})} \\ &\leq \|x_1 - x_2\| e^{b[i](t-t_{i-1}) + \sum_{j=1}^{i-1} b[j](t_j - t_{j-1})}.\end{aligned}$$

Next we will prove that

$\beta(x_1, x_2, t) \equiv \|x_1 - x_2\| e^{b[i](t-t_{i-1}) + \sum_{j=1}^{i-1} b[j](t_j - t_{j-1})}$ is a valid LDF.

In Lines 6-9, because J is a real matrix, the maximum eigenvalue λ of $(J^T + J)/2$ is bounded. Assume that each component of $E(x) = J_f^T(x) + J_f(x) - J^T - J$ is continuous over the closed set S, then we can find the upper bound of $\|E(x)\|$, so the "error" term is also bounded. Therefore, each $b[i]$ is bounded. So $\forall t \in [t_{i-1}, t_i]$, $i = 1, \ldots, k$, $\exists N < \infty$, such that $e^{b[i](t-t_{i-1}) + \sum_{j=1}^{i-1} b[j](t_j - t_{j-1})}$ is bounded by N from the above.

As $x_1 \to x_2$, obviously,

$$\|x_1 - x_2\| e^{b[i](t-t_{i-1}) + \sum_{j=1}^{i-1} b[j](t_j - t_{j-1})} \to 0.$$

And for any $\epsilon > 0$, $\exists \delta = \epsilon/N > 0$, such that $\forall x_1, x_2 \in B_\delta(R_0)$ and $\|x_1 - x_2\| < \delta$, it follows

$$\|x_1 - x_2\| e^{b[i](t-t_{i-1}) + \sum_{j=1}^{i-1} b[j](t_j - t_{j-1})} < \epsilon/N \cdot N = \epsilon.$$

So $\beta(x_1, x_2, t) = \|x_1 - x_2\| e^{b[i](t-t_{i-1}) + \sum_{j=1}^{i-1} b[j](t_j - t_{j-1})}$ is a $B_\delta(R_0)$-local piecewise discrepancy function and the array b contains the corresponding coefficients.

4.3 Coordinate transformation

In this section, we will discuss the issue that the upper bound of the symmetric part of the Jacobian computed inLines 6-9 may introduce loss in precision. We propose a a strategy to reduce this loss by first performing a coordinate transformation. Consider a simple linear system:

$$\dot{x} = [0 \;\; 3; -1 \;\; 0]x, \tag{4}$$

which has eigenvalues $\pm\sqrt{3}i$ and thus its trajectories oscillate. The symmetric part of the Jacobian is $[0 \;\; 1; 1 \;\; 0]$ with eigenvalues ± 1, which gives the exponentially growing discrepancy with $b = 1$. In what follows, we will see that a tighter bound can be obtained by first taking a linear transformation of x. The following is a coordinate transformed version of Lemma 4. The coordinate transformation matrix P can be any $n \times n$ real invertible matrix, and the condition number of P is $\|P\|\|P^{-1}\|$.

Lemma 6. *In i^{th} iteration of the loop, for any $x_1, x_2 \in B_\Delta(R_{i-1})$, and any $t \in [t_{i-1}, t_i]$,*

$$\|\xi(x_1, t) - \xi(x_2, t)\| \leq K\|x_1 - x_2\| e^{\tilde{\lambda}_{max}(S)(t-t_{i-1})},$$

where $\tilde{\lambda}_{max}(S)$ is the upper bound of $\frac{1}{2}(\widetilde{J_f}^T(x) + \widetilde{J_f}(x))$ over the set S, $\widetilde{J_f}(x) = PJ_f(x)P^{-1}$, and K is the condition number of P.

The proof of Lemma 6 is similar to the proof of Lemma 4 considering the coordinate transformation. This shows that the distance can be bounded in the same way for the transformed system with a (possibly much smaller) $\tilde{\lambda}_{max}(S)$ but with an additional multiplicative cost of $cond(P)$.

To choose the coordinate transformation matrix, one approach that produces good empirical results is making the Jacobian matrix at the center point a real Jordan form. Let S be the set computed in Line 5 and J is the Jacobian evaluated at the center s_0 of S. Let $\tilde{J} = PJP^{-1}$ the real Jordan form and use the matrix P as the coordinate transformation matrix for $J_f(x)$. Contraction matrix [23] introduces more general coordinate transformation. However, there are no general methods to compute it for nonlinear systems. Choosing a constant matrix as transformation is an implementable approach, and Lemma 6 applies to any invertible matrix.

In the previous example (4), the Jacobian matrix is constant, and the discrepancy function without coordinate transformation is:

$$\|\xi(x_1, t) - \xi(x_2, t)\| \leq \|x_1 - x_2\| e^{t-t_1}.$$

If we use $P = [1 \;\; 3; -\sqrt{3} \;\; \sqrt{3}]$ as the coordinate transformation matrix, $\tilde{J} = PJP^{-1} = [0 \;\; \sqrt{3}; -\sqrt{3} \;\; 0]$, and the discrepancy function with coordinate transformation is

$$\|\xi(x_1, t) - \xi(x_2, t)\| \leq \sqrt{3}\|x_1 - x_2\|.$$

In practice, the coordinate transformation can be made for longer time interval $[t_{i-k}, t_i]$, where $k > 2$, to reduce the multiplicative error term $\prod cond(P[i])$.

5 Local Input-State Discrepancy

Large and complex models of dynamical system are created by composing smaller modules or subsystems. Consider a dynamical system A consisting of several interacting subsystems $A_1, \ldots, A_N$, that is, the input signals of a subsystem A_i are driven by the outputs (or states) of some another component A_j. Let's say that each A_i is n-dimensional which makes A nN-dimensional. One way of achieving scalable verification of A is to exploit this compositional structure and somehow analyze the component A_i's to infer properties of A.

In [20], the notion of input-to-state (IS) discrepancy was introduced to address the problem of finding annotations for large models. It is shown that if we can find input-to-state (IS) discrepancy functions for the individual component A_i, then we can construct a reduced N-dimensional model M such that the executions of M serve as the discrepancy of the overall system. Thus, from IS-discrepancy for the smaller A_i models and simulations of the N-dimensional system M, we are able to verify A. This has the beneficial side-effect that if the A_i's are rewired in a new topology, then only the reduced model changes [19]. However,[20] still assumes that the user provides the IS-discrepancy for the smaller modules. In this section, we will show the approach used in the previous section can be used to get IS discrepancy function for Lipschitz continuous nonlinear subsystems A_i. Furthermore, it gives an over-approximation of the reachsets with nondeterministic bounded inputs.

5.1 Defining Local IS Discrepancy

Consider a dynamical system with inputs:

$$\dot{x} = f(x, u) \tag{5}$$

where $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ is Lipschitz continuous. For a given input signal which is a integrable function $\upsilon : [0, \infty) \to \mathbb{R}^p$, and an initial state $x_0 \in \mathbb{R}^n$, a solution (or trajectory) of the system is a function $\xi : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ such that $\xi(x_0, 0) = x_0$ and for any time $t \geq 0$, $\dot{\xi}(x,t) = f(\xi(x,t), \upsilon(t))$.

First, we give the original definition of IS discrepancy function for the system in (5). Here $\mathcal{U}$ is the set $\{u | u : [0, \infty) \to \mathbb{R}^p\}$ of all input signals.

Definition 4. *A pair of uniformly continuous functions $\beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and $\gamma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is called C-local input-to-state discrepancy if*

(1) β is of class $\mathcal{K}$ with respect to its first argument and γ is also of class $\mathcal{K}$,
(2) for any pair of initial states $x, x' \in C$, any pair of input signals $u, u' \in \mathcal{U}$, and $t \in \mathbb{R}_{\geq 0}$:$\|\xi(x,t) - \xi(x',t)\| \leq \beta(\|x - x'\|, t) + \int_0^t \gamma(\|u(s) - u'(s)\|) ds$.

For a bounded, compact set $\mathcal{I} \subseteq \mathbb{R}^p$. A family of bounded time input signals over $\mathcal{I}$ is the set $\mathcal{U}(\mathcal{I}) = \{u | u : [0, T) \to \mathcal{I}\}$ of integrable functions. We denote $\mathsf{Reach}(K, \mathcal{U}(\mathcal{I}), [t_1, t_2])$ as the reachable states of the system from compact set K with input set $\mathcal{U}(\mathcal{I})$ over $[t_1, t_2]$. Next, we introduce an inductive definition of IS discrepancy for inputs over compact neighborhoods.

Definition 5. *Consider compact sets $K \in \mathbb{R}^n, \mathcal{I} \in \mathbb{R}^p$ and a sequence of time points $0 = t_0 < t_1 < t_2 < \ldots < t_k = T$. For any pair of initial states $x_1, x_2 \in K$, any pair of input signals $u_1, u_2 \in \mathcal{U}(\mathcal{I})$, the $(K, \mathcal{U}(\mathcal{I}))$-local IS discrepancy function $\alpha : K^2 \times \mathcal{U}(\mathcal{I})^2 \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is defined as:*

$$\alpha(x_1,x_2,u_1,u_2,t) = \begin{cases} \|x_1 - x_2\|, & \textit{if } t = t_0, \\ \alpha(x_1, x_2, u_1, u_2, t_{i-1})e^{a[i](t-t_{i-1})} + & \\ M[i]e^{a[i](t-t_{i-1})} \int_{t_{i-1}}^{t} \|u_1(\tau) - u_2(\tau)\| d\tau & \textit{if } t \in (t_{i-1}, t_i] \end{cases}$$

where $a[1], \ldots, a[k], M[1], \ldots, M[k]$ are real constants.

5.2 Algorithm for Local IS Discrepancy

The approach to find $(K,\mathcal{U}(\mathcal{I}))$-local IS discrepancy function is similar to $ComputeLDF$ algorithm, which also uses a **for** -loop to compute the coefficients $a[i]$ and $M[i]$. The only changes are 1) in Line 5 S should be computed as in Lemma 7, 2) in Line 10 Δ should be updated as in Lemma 9. Next we illustrate this process in more detail. First, we use Lipschitz constant to get a coarse over-approximation of $\mathsf{Reach}(K,\mathcal{U}(\mathcal{I}), [t_{i-1}, t_i])$ parallel to Lemma 1. Let $l = dia(\mathcal{I})$.

Lemma 7. *In i^{th} iteration of the* **for** *-loop,* $\mathsf{Reach}(B_\Delta(R_{i-1}), \mathcal{U}(\mathcal{I}), [t_{i-1}, t_i]) \subseteq S$, *where $S = hull(R_{i-1}, R_i) \oplus B_{\Delta'}(R_i)$ and $\Delta' = (\Delta + \epsilon)(e^{L_f \tau_i}) + lL_f e^{L_f \tau_i}\tau_i$, $\tau_i = t_i - t_{i-1}$.*

Two trajectories starting from $x_1, x_2 \in \mathbb{R}^n$ at t_{i-1}, with $u_1, u_2 \in \mathcal{U}(\mathcal{I})$ as inputs respectively, their distance at time t, $\|\xi(x_1,t) - \xi(x_2,t)\| \leq \|x_1 - x_2\|(e^{L_f(t-t_{i-1})}) + L_f e^{L_f(t-t_{i-1})} \cdot \int_{t_{i-1}}^{t} \|u_1(\tau) - u_2(\tau)\| d\tau$. The lemma directly follows this inequality.

Next we give a one step IS discrepancy function in Lemma 9. Before proving it, we need another generalized form of mean value theorem:

Lemma 8. *For any continuous and differentiable function $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$, $f(x + r, u + w) - f(x,u) =$ $\left(\int_0^1 J_x(x + sr, u + w) ds\right) r + \left(\int_0^1 J_u(x, u + \tau w) d\tau\right) w$, where $J_x = \frac{\partial f(x,u)}{\partial x}$ and $J_u = \frac{\partial f(x,u)}{\partial u}$ are the Jacobian matrices of f with respect to x and u.*

Proof. The lemma follows by writing $f(x + r, u + w) - f(x,u) = f(x + r, u + w) - f(x, u + w) + f(x, u + w) - f(x, u)$ and then invoking Lemma 2.

Lemma 9. *Consider the i^{th} iteration of the loop for a dynamic system (5). Let $x, x' \in B_\Delta(R_{i-1})$, and $\xi(x,t)$, $\xi(x',t)$ be the trajectories starting from x and x' with input $u_1(t), u_2(t) \in \mathcal{U}(\mathcal{I})$ respectively, where $t \in [t_{i-1}, t_i]$. Then,*

$$\|\xi(x,t) - \xi(x',t)\| \leq \|x - x'\| e^{a(t-t_{i-1})} + M e^{a(t-t_{i-1})} \int_{t_{i-1}}^{t} \|u_1(\tau) - u_2(\tau)\| d\tau, \tag{6}$$

where $a = \lambda_{max}(S) + \frac{1}{2}$, $\lambda_{max}(S)$ is the upperbound of the eigenvalues of the symmetric part of J_x over S, and $M = \max_{u \in \mathcal{U}(\mathcal{I})} (\|J_u(\xi(x,t), u)\|)$.

To prove Lemma 9, we will use Lemma 8 and the detailed proof can be found in [14].

Using Lemma 9 to get the coefficients $a[i]$ and $M[i]$ in each time interval $[t_{i-1}, t_i], i = 1 \ldots, k$, we will have:

Theorem 4. *The items in array a and M are a coefficients of the $(K,\mathcal{U}(\mathcal{I}))$-local IS discrepancy function for the system* (5).

This theorem enables us to compute the $(K,\mathcal{U}(\mathcal{I}))$-local IS discrepancy function for each subsystem A_i. Although in the original definition we assume the IS discrepancy function is valid for any input signals $u_1, u_2 \in \mathcal{U}$, in practice A_i can only take A_j's outputs or states as inputs, which is bounded. Thus, [20] can still use $(K,\mathcal{U}(\mathcal{I}))$-local IS discrepancy function computed by this approach. Furthermore, the $(K,\mathcal{U}(\mathcal{I}))$-local IS discrepancy function here can over-approximate the reachset of the systems in (5) with the input u being chosen nondeterministically in some compact set.

6 Experimental Evaluation

We have implemented the verification Algorithm 1 and the $ComputeLDF$ subroutine with coordinate transformations in Matlab. The implementation and the examples are available from [14]. For simulation we use Matlab's built-in ODE solver. The Jacobian matrix, an upper bound of the Lipschitz constant are given as inputs. In addition, the function to do the term-wise maximization of the error matrix is also given as inputs (see Section 4.2). We use the absolute error for ODE solver as the error bounds for simulation. The results presented here are based on experiments performed on an Intel Xeon V2 desktop computer.

Comparison with other tools. We compare the performance of our algorithm with two other tools, namely, Flow* [6] and HyCreate [26], for safety verification problem of nonlinear dynamical systems. We use seven benchmarks which are shown in Table 1 with time bound $T = 10s$. Flow* uses Taylor models for approximating reachtubes from a set of initial states. Currently, it returns "Safe" or "Unknown", but not "Unsafe". HyCreate uses the face-lifting approach of [8] and provides a intuitive interface for creating models.

Vanderpol, CoupledVanderpol, JetEngine, and Brusselator are commonly used, low-dimensional, nonlinear benchmarks. Sinusoidal tracking [25] is a 6 dimensional nonlinear designed as a frequency estimator. The Lorenz Attractor (row 7) is a well known chaotic dynamical system. Robot arm is a 4 dimensional nonlinear system described in [3]. The Helicopter is a high dimension linear model of a helicopter system from [16]. In row 10 and 11, we increase the time bound of the fixed-wing model to $T = 50$ and $T = 100$ respectively and the results show that the algorithm scales reasonably for longer time horizons. Columns (#Sim) and (LDF) are the results of the proposed algorithm $ComputeLDF$ with coordinate transformation. More results of the algorithm without coordinate transformation can be found in [14].

Flow* and HyCreate generate a single over-approximation of the reachtube from the initial set independent of the safety property. While our algorithm will refine the initial sets when the reachtube intersects with the unsafe set. In all of these benchmarks, we make the unsafe set close to the reachtubes, to make the models safe yet it needs a lot of refinements to arrive at that conclusion. Overall, the proposed approach with coordinate transformation outperformed others in terms of the running time, especially in high dimensional benchmarks. The "TO" in the table means the algorithm timed out at 30 minutes. Of course, our implementation requires the users to give the symbolic expression of the Jacobian matrix and term-wise maximization functions, while Flow*

Table 1: Safety verification for benchmark examples. dim: dimension of the model; δ: diameter of the initial set; $\mathbb{U}$: unsafe set; #Sim: number of simulations with coordinate transformation; LDF: runtime of our implementation (with coordinate transformation) in secs.

	example	dim	δ	$\mathbb{U}$	#Sim	LDF(s)	flow*(s)	HyCreate(s)
1	Vanderpol	2	0.5	x>2.0	9	0.378	11.2	2.776
2	Brusselator	2	0.5	x>1.3	21	1.01	11.8	1.84
3	Jet Engine	2	0.4	x>2.0	5	0.353	8.74	5.54
4	Robot arm	4	0.5	x>2.5	81	4.66	169	>300
5	CoupledVanderpol	4	0.5	x>2.5	41	2.21	93	49.8
6	Sinusoidal Tracking	6	0.5	x>10	185	13.2	258	>300
7	Lorenz Attractor	3	0.02	x>1e4	570	13.99	53.4	TO
8	Fixed-wing UAV (T=10)	7	3	x> 39	321	20.8	TO	TO
9	Helicopter	28	0.02	x>4	585	67.7	TO	TO
10	Fixed-wing UAV (T=50)	7	3	x> 39	321	99.8	TO	TO
11	Fixed-wing UAV (T=100)	7	3	x> 39	321	196	TO	TO

and HyCreate just needs the differential equations. Moreover, our implementation currently handles only nonlinear dynamical systems, and both Flow* and HyCreate can handle hybrid systems.

Properties of LDF. We explore the behavior of the algorithm with respect to changes in the relative positions of the initial set and the unsafe set. We use the nonlinear model of the Robot arm system. We fix the point $[1.5, 1.5, 0, 0]$ as the center of the initial set and $T = 10$ seconds as the time bound, and vary the diameter of the initial set (δ) and the unsafe set ($\mathbb{U} : \theta > c$), where θ is the angle of the arm. The number of simulations used by the algorithm with coordinate transformation (#Sim), the diameter of the reach tube at the final time T (dia), and the total running time (RT) are shown in Table 7.

From the first 4 rows in the Table, we see the expected behavior that for a fixed unsafe set, the diameter of the Reachtube decreases with decreasing δ. This corresponds to the property that the discrepancy function $\beta(x, x', t)$ goes to 0 as the initial points $x \to x'$, and therefore the error in the reachability computation decreases monotonically with the diameter of the initial set. Rows 3 and 5-6 show that if we fix the size of the initial set, then as the unsafe set comes closer to the actual reachtube, the number of simulations increases and therefore the running time increases until the system becomes unsafe. As more refinements are made by the algorithm, the accuracy (measured by the diameter of the reachtube) improves. Similar trend is seen in rows 7-8, the algorithm will need more refinements to find a counter example that shows unsafe behavior, if the unsafe set is close to the boundary of the reachtube.

Next, we explore the behavior of the algorithm (with coordinate transformation) with large initial sets. We use the 7 dimensional model of a fixed-wing UAV. The initial sets are defined as balls with different radii around a center point $[30, 980, 0, 125, 0, 0, 30.4]$ and δ in the first column is the diameter of the initial sets. The unsafe set is defined as

$H > c$, where H is the thrust of UAV. The time horizon is fixed at $T = 10$ seconds. As shown in Table 3, our algorithm can handle large initial set and high dimension systems. Although it may need many simulations (24001 covers), the algorithm terminates in 30 mins. All the results of this table are safe.

7 Conclusions and Future Work

In this paper, we present an algorithm *ComputeLDF* to compute local discrepancy functions, which is an upperbound of the distance between trajectories starting from an initial set. The algorithm computes the rate of trajectory convergence or divergence for small time intervals and gives the rate as coefficients of a continuous piecewise exponential function. The local discrepancy we compute satisfies the definition of discrepancy function, so the verification algorithm using *ComputeLDF* as a subroutine is sound and relatively complete. We also provide a coordinate transformation method to improve the estimation of rates. Furthermore, we extend the algorithm to compute input-to-state discrepancy functions. *ComputeLDF* has been successfully used to safety verify several complex nonlinear systems like powertrain control system. In the future, we plan on using more rigorous ODE solvers like [5] and embedding the algorithm in verification tools like C2E2 [11] for safety verification of hybrid systems.

	δ	$\mathbb{U}$	saftey	#Sim	dia	RT(s)
1	0.6	$\theta > 3$	safe	17	5.6e-3	0.948
2	0.3	$\theta > 3$	safe	9	2.6e-3	0.610
3	0.2	$\theta > 3$	safe	5	1.8e-3	0.444
4	0.1	$\theta > 3$	safe	1	1.5e-3	0.271
5	0.2	$\theta > 2.5$	safe	9	1.7e-3	0.609
6	0.2	$\theta > 2.15$	safe	161	9.2e-4	6.705
7	0.2	$\theta > 2.14$	unsafe	45	N/A	1.997
8	0.2	$\theta > 2.1$	unsafe	1	N/A	0.267

Table 2: Safety verification for a robot arm with different initial states and unsafe sets. safety: safety result returned by verification algorithm;

	δ	$\mathbb{U}$	#Sim	RT(s)
1	50	$H > 400$	24001	1518
2	46	$H > 400$	6465	415
3	40	$H > 400$	257	16.33
4	36	$H > 400$	129	8.27
5	20	$H > 400$	1	0.237

Table 3: Safety verification for a fixed-wing UAV with large initial sets.

References

1. M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *CDC*, pages 4042–4048, 2008.
2. D. Angeli. A lyapunov approach to incremental stability properties. *IEEE Trans. Autom. Control*, 47(3):410–421, 2002.
3. D. Angeli, E. D. Sontag, and Y. Wang. A characterization of integral input-to-state stability. *IEEE Trans. Autom. Control*, 45(6):1082–1097, 2000.

4. Y. Annpureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan. *S-TaLiRo: A tool for temporal logic falsification for hybrid systems*. 2011.
5. CAPD. Computer assisted proofs in dynamics. urlhttp://www.capd.ii.uj.edu.pl/, 2002.
6. X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *CAV*, pages 258–263, 2013.
7. T. Dang, A. Donzé, O. Maler, and N. Shalev. Sensitive state-space exploration. In *CDC*, pages 4049–4054, 2008.
8. T. Dang and O. Maler. Reachability analysis via face lifting. In *HSCC*, pages 96–109. 1998.
9. A. Donzé and O. Maler. Systematic simulation using sensitivity analysis. In *HSCC*, pages 174–189. 2007.
10. P. S. Duggirala, C. Fan, S. Mitra, and M. Viswanathan. Meeting a powertrain verification challenge (to appear in cav 2015).
11. P. S. Duggirala, S. Mitra, and M. Viswanathan. Verification of annotated models from executions. In *EMSOFT*, page 26, 2013.
12. P. S. Duggirala, L. Wang, S. Mitra, M. Viswanathan, and C. Muñoz. Temporal precedence checking for switched models and its application to a parallel landing protocol. In *FM*, pages 215–229. 2014.
13. C. Fan, P. S. Duggirala, S. Mitra, and M. Viswanathan. Progress on powertrain verification challenge with c2e2. *ARCH*, 2015.
14. C. Fan and S. Mitra. Bounded verification with on-the-fly discrepancy computation (full version). available at `http://web.engr.illinois.edu/~cfan10/research.html`.
15. G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In S. Q. Ganesh Gopalakrishnan, editor, *CAV*, 2011.
16. G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In *CAV*, pages 379–395, 2011.
17. A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE Trans. on Autom. Control*, 55(1):116–126, 2010.
18. Z. Han and P. J. Mosterman. Towards sensitivity analysis of hybrid systems using simulink. In *HSCC*, pages 95–100, 2013.
19. Z. Huang, C. Fan, A. Mereacre, S. Mitra, and M. Z. Kwiatkowska. Invariant verification of nonlinear hybrid automata networks of cardiac cells. In *CAV*, pages 373–390, 2014.
20. Z. Huang and S. Mitra. Proofs from simulations and modular annotations. In *In 17th International Conference on Hybrid Systems: Computation and Control*, Berlin, Germany. ACM press.
21. M. Islam, R. DeFrancisco, C. Fan, R. Grosu, S. Mitra, S. A. Smolka, et al. Model checking tap withdrawal in c. elegans. *arXiv preprint arXiv:1503.06480*, 2015.
22. A. A. Julius and G. J. Pappas. Trajectory based verification using local finite-time invariance. In *HSCC*, pages 223–236. 2009.
23. W. Lohmiller and J.-J. E. Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998.
24. N. Nedialkov. VNODE-LP: Validated solutions for initial value problem for ODEs. Technical report, McMaster University, 2006.
25. B. B. Sharma and I. N. Kar. Design of asymptotically convergent frequency estimator using contraction theory. *IEEE Trans. Autom. Control*, 53(8):1932–1937, 2008.
26. B. Stanley and C. Marco. Computing reachability for nonlinear systems with hycreate. In *Demo and Poster Session, HSCC*.
27. M. Zamani, G. Pola, M. Mazo, and P. Tabuada. Symbolic models for nonlinear control systems without stability assumptions. *IEEE Trans. Autom. Control*, 57(7):1804–1809, 2012.