BOUNDED VERIFICATION WITH ON-THE-FLY DISCREPANCY COMPUTATION

Chuchu Fan and Sayan Mitra

Coordinated Science Laboratory
1308 West Main Street, Urbana, IL 61801
University of Illinois at Urbana-Champaign

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 2015	

4. TITLE AND SUBTITLE
Bounded Verification with On-the-Fly Discrepancy Computation

5. FUNDING NUMBERS
CAR 1054247 (NSF)
NSF CSR 1016791
AFOSR YIP FA9550-12-1-0336

6. AUTHOR(S)
Chuchu Fan and Sayan Mitra

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, 1308 W. Main St., Urbana, IL, 61801-2307

8. PERFORMING ORGANIZATION REPORT NUMBER
UILU-ENG-15-2201

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Science Foundation, 4201 Wilson Boulevard, Arlington, VA 22230
Air Force Office of Scientific Research, 875 N. Randolph, Ste. 325, Arlington, VA 22203

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

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 functions,” which are crucial for computing reachtubes from simulations. In this report, we eliminate that requirement by presenting an algorithm for computing piece-wise 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.

14. SUBJECT TERMS
Formal verification; Hybrid systems; Reachability; Algorithms

15. NUMBER OF PAGES
24

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Bounded Verification with On-the-Fly Discrepancy Computation*

Chuchu Fan and Sayan Mitra
{cfan10, mitras}@illinois.edu
Coordinated Science Laboratory
University of Illinois at Urbana Champaign
Urbana, IL 61801

February 5, 2015

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 piece-wise 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 [10, 17, 4, 14, 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,

*We gratefully acknowledge the feedback from anonymous referees on a previous draft of this technical report. The results presented here came about from work supported and funded by the National Science Foundation (grant: CAR 1054247 and NSF CSR 1016791) and the Air Force Office of Scientific Research (AFOSR YIP FA9550-12-1-0336).

then the algorithm decides, otherwise, it draws more samples of initial states and repeats the earlier steps to compute more precise over-approximation. With post-processing of the reachtube over-approximations this basic procedure can be utilized to verify termporal precedence [11] and richer classes of properties [7].

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 [10]: 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 [10], it is observed that the notion of a contraction metric [19] gives a much tighter bound and it provided heuristics for finding them for some classes of polynomial systems. Sensitivity analysis approach gives 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 6. 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. 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.

We 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 Background

2.1 Notations

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 $f(0) = 0$.

We denote the transpose of a matrix A by A^T. The *conjugated transpose* of A is the matrix A^H obtained by replacing each entry in A^T with its complex conjugate.

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: $\|A\| = \sqrt{\lambda_{\max}(A^H A)}$. If $\forall x \in \mathbb{R}^n$, $x^T Ax \leq 0$, then we say A is negative-semidefinite, and write $A \preceq 0$. We write $A \preceq B$ if $A - B \preceq 0$.

2.2 Safety Verification Problem

Consider an n-dimensional *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 $t > 0$, $\xi(x_0, t)$ 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 [10, 11, 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. (Figure 1).

2.3 Simulations, Reachtubes and Annotations

The algorithm uses simulation oracles that give sampled numerical simulations of the system from individual initial states.

Definition 2.1. 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 [20] 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 [10, 16] require the user to decorate the model with annotations called *discrepancy functions* for computing reachtubes.

Definition 2.2. 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 pair of states $x, x' \in \mathbb{R}^n$, and any time $t > 0$,

$$\|\xi(x, t) - \xi(x', t)\| \leq \beta(x, x', t), \text{and} \tag{2}$$

(2) for any t, as $x \to x'$, $\beta(., ., t) \to 0$,

If the function β meets the two conditions for any pair of states x, x' in a compact set K 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 [10, 16] 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. [10, 16, 17] 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.

2.4 Verification Algorithm

The simulation-based verification algorithm is shown in Figure 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

1: **Input:**$\Theta, \mathbb{U}, T$
2: $\delta \leftarrow dia(\Theta); \epsilon \leftarrow \epsilon_0; \mathcal{C} \leftarrow \varnothing, \mathcal{R} \leftarrow \varnothing$; //$\epsilon_0$ is a small constant
3: $\mathcal{C} \leftarrow \langle Partition(\Theta, \delta), \delta, \epsilon \rangle$
4: **while** $\mathcal{C} \neq \varnothing$ **do**
5: **for** $(\theta, \delta, \epsilon) \in \mathcal{C}$ **do**
6: $\psi \leftarrow Simulate(\theta, \tau, \epsilon, T)$
7: $\beta \leftarrow ComputeLDF(\psi, J_f, L_f, \delta, \epsilon)$
8: $\mathrm{D} \leftarrow \psi \oplus \beta$
9: **if** $D \cap \mathbb{U} = \varnothing$ **then**
10: $\mathcal{C} \leftarrow \mathcal{C} \backslash \{(\theta, \delta, \epsilon)\}; \mathcal{R} \leftarrow \mathcal{R} \cup D$
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: **end if**
17: **end for**
18: **end while**
19: **return** (SAFE, $\mathcal{R}$)

Figure 1: Verification Algorithm

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 reachtube from $B_\delta(\theta)$ 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 8) 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 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 [10]. In this paper, we will analyze in detail the *ComputeLDF* subroutine which computes a local version of discrepancy function automatically.

The following results from [10] state two key properties of the algorithm. Although in [10] β was defined globally, it is easy to check that the local version still satisfies them.

Theorem 2.3. *The Algorithm in Fig.1 is sound, that is, if it returns "SAFE" then the system is safe; when it returns "UNSAFE" there exists at least one execution from Θ that is unsafe. The Algorithm is relatively complete, that is, if the system is robustly safe, the algorithm will terminate and return "SAFE". If any executions from Θ is unsafe, it will terminate and return "UNSAFE".*

3 Local discrepancy function

In this section, we present the analysis of *ComputeLDF* algorithm. 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 3.1. 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 *piece-wise exponential discrepancy function* $\beta : C \times C \times [0, T] \to \mathbb{R}_{\geq 0}$ is defined as
$\beta(x_1, x_2, t) =$

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

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.

3.1 ComputeLDF Algorithm

Figure 2 shows the pseudocode for *ComputeLDF* used in Line 7 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 piece-wise 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 3.8). In Lines 8 and 9, 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 3.2). In Lines 10–13 an upper bound on the maximum eigenvalue of the symmetric part of the Jacobian over the set S, is computed as $b[i]$ (Lemma 3.6). Then Δ is updated as $(\Delta + \epsilon) e^{b[i](t_i - t_{i-1})}$ for the next iteration.

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

Figure 2: Algorithm *ComputeLDF*.

3.2 Analysis of ComputeLDF

In this section, we will prove that *ComputeLDF*$(\psi, J_f, L_f, \ \delta, \epsilon)$ returns a piece-wise 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. First, we show in Lemma 3.2 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 3.2. In i^{th} iteration of the loop of *ComputeLDF*, $\mathsf{Reach}(B_\Delta(R_{i-1}), [t_{i-1}, t_i]) \subseteq S$.

Proof. Let $\xi(\theta, t)$ denote the actual trajectory from θ, where θ is the initial state of ψ. By Definition 2.1 for ψ, it is known that $\theta \in R_0$ and $\forall i = 1, \ldots, k, \xi(\theta, t_i) \in R_i$.

For a fixed iteration number i, consider state $x = \xi(\theta, t_{i-1}) \in R_{i-1}$ from Definition 2.1. We know that for any $t \in [t_{i-1}, t_i]$, $\xi(x, t) \in hull(R_{i-1}, R_i)$. Now consider another state $x' \in B_\Delta(R_{i-1})$. Since L_f is the Lipschitz constant of f, using Gronwall's inequality we have that $\|\xi(x, t) - \xi(x', t)\| \leq \|x - x'\| e^{L_f(t - t_{i-1})}$. Since $\|x - x'\| \leq \Delta + \epsilon$, $\|\xi(x, t) - \xi(x', t)\| \leq (\Delta + \epsilon)e^{L_f(t - t_{i-1})}$. Therefore, $\xi(x', t) \in hull(R_{i-1}, R_i) \oplus B_{(\Delta+\epsilon)e^{L_f(t_i - t_{i-1})}}(0) = S$. Because x' is arbitrarily selected from $B_\Delta(R_{i-1})$, the lemma is proved. ∎

Next, using the generalized mean value theorem (Lemma 3.3), we get that in the i^{th} iteration, the computed $b[i]$ in Line 13 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 3.3. For any continuously differentiable vector-valued function $f : \mathbb{R}^n \to \mathbb{R}^n$, and $x, r \in \mathbb{R}^n$,

$$f(x + r) - f(x) = \left(\int_0^1 J_f(x + sr) ds \right) \cdot r, \tag{3}$$

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. The complete proofs of Lemma 3.3 and Theorem 3.4 can be found in the appendix.

Theorem 3.4. *If A and E are $n \times n$ symmetric matrices, then*

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

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

Corollary 3.5. If A and E are $n \times n$ symmetric matrices, then

$$|\lambda_k(A+E) - \lambda_k(A)| \leq \|E\|. \tag{4}$$

Since A is symmetric, $\|A\| = \sqrt{\lambda_{\max}(A^T A)} = \max(|\lambda(A)|)$. From Theorem 3.4, we have $|\lambda_k(A+E) - \lambda_k(A)| \leq \max\{|\lambda_n(E)|, |\lambda_1(E)|\} = \|E\|$. 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 $\|E(x)\|$ over S by each term $e_{ij}(x)$, $|e_{ij}(x)|$ over S. Let ${}_{x\in S}(|e_{ij}(x)|)$ be denoted by $\tilde{e}_{ij}$, then we know , $\|E(x)\| \leq \sqrt{\sum_{i=1}^n \sum_{j=1}^n \tilde{e}_{ij}^2}$. Using Corollary 3.5, we next show in Lemma 3.6 that $b[i]$ calculated in Line 13 bounds the eigenvalues of symmetric part of Jacobian matrix over S.

Lemma 3.6. 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 9 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 3.5 implies that $\lambda_{max}(J_f^T(x) + J_f(x)) - \lambda_{max}(J^T + J) \leq ||E(x)||$. The *error* term computed in Line 12 is the upperbound on $||E(x)||$. Therefore, $\lambda_{max}(J_f^T(x) + J_f(x)) \leq \lambda_{max}(J^T + J) + error$. In Line 13 set $b[i]$ equals to $\lambda_{max}((J^T + J)/2) + error/2$. Thus, $\lambda_{max}(J_f^T(x) + J_f(x)) \leq 2b[i]$, which immediately indicates that $\forall x \in S : J_f^T(x) + J_f(x) \preceq 2b[i]I$. ∎

By Lemma 3.3 and Lemma 3.6, we can prove as in Lemma 3.7 that $b[i]$ calculated in Line 13 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 3.7. 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 $t \in [t_{i-1}, t_i]$, $\|\xi(x_1, t) - \xi(x_2, t)\| \leq \|x_1 - x_2\| e^{b[i](t-t_{i-1})}$.

Proof. Let us fix the iteration i and two states $x_1, x_2 \in B_\Delta(R_{i-1})$. From Lemma 3.2 it's can be seen that for any $t \in [t_{i-1}, t_i]$, $\xi(x_1, t) \in S, \xi(x_2, t) \in S$. Define $y(t) \equiv \xi(x_2, t) - \xi(x_1, t)$. For a fixed time t, from Lemma 3.3 we have

$$\begin{aligned} \dot{y}(t) &= \dot{\xi}(x_2, t) - \dot{\xi}(x_1, t) = f(\xi(x_2, t)) - f(\xi(x_1, t)) \\ &= \left(\int_0^1 J_f(\xi(x_1, t) + sy(t)) ds \right) y(t). \end{aligned} \tag{5}$$

Since S is the Minkowski sum of two convex sets $hull(R_{i-1}, R_i)$ and $B_{\Delta e^{L_f(t_i - t_{i-1})}}(0)$, it is also convex. Recall that $\xi(x_1, t)$, $\xi(x_2, t) \in S$, and for any $s \in [0, 1]$, $\xi(x_1, t) + sy(t) \subseteq S$.

Differentiating $\|y(t)\|^2$, we have

$$\begin{aligned}\frac{d\|y(t)\|^2}{dt} &= \dot{y}^T(t)y(t) + y^T(t)\dot{y}(t)\\ &= y^T(t)\left(\int_0^1 J_f^T(\xi(x_1,t)+sy(t))ds\right)y(t)\\ &+ y^T(t)\left(\int_0^1 J_f(\xi(x_1,t)+sy(t))ds\right)y(t).\end{aligned} \tag{6}$$

Using Lemma 3.6, we know

$$\forall x \in S, \quad J_f^T(x) + J_f(x) \preceq 2b[i]I.$$

Thus, we can bound (6)

$$\begin{aligned}\frac{d\|y(t)\|^2}{dt} &\leq y^T(t)\left(\int_0^1 (2b[i]I)ds\right)y(t)\\ &= 2b[i]y^T(t)y(t)\\ &= 2b[i]\|y(t)\|^2.\end{aligned} \tag{7}$$

Integrating both sides over t_{i-1} to any $t \in [t_{i-1}, t_i]$, we have

$$\begin{aligned}&\ln(\|y(t)\|^2) - \ln(\|y(t_{i-1})\|^2) \leq 2b[i](t - t_{i-1})\\ \Rightarrow &\|y(t)\|^2 \leq \|y(t_{i-1})\|^2 e^{2b[i](t-t_{i-1})}\\ \Rightarrow &\|\xi(x_1,t) - \xi(x_2,t)\| \leq \|x_1 - x_2\| e^{b[i](t-t_{i-1})}.\end{aligned}$$

■

Up to this point all the lemmas were statements about a single iteration of the **for** -loop, next we show that in i^{th} iteration of the loop, $B_\Delta(R_i)$ used in Lemma 3.2 and 3.7 is the reach set from $B_\delta(R_0)$ at time t_i.

Lemma 3.8. 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 14 is executed in the i^{th} iteration, and $\Delta_i' = \max\{\Delta_i, \Delta_{i-1} + \epsilon\}$

Proof. In this proof, let $\xi(\theta, \cdot)$ denote the trajectory from θ. From the Definition 2.1 for ψ, we know that $\theta \in R_0$ and $\forall i = 1, \ldots, k, \xi(\theta, t_i) \in R_i$. Let S_i denote S after Line 9 is executed in the i^{th} iteration. The lemma is proved by induction on i. Note that the initial set is $B_\delta(R_0)$, and before the **for**-loop, Δ_0 is set as δ.

When $i = 1$, we already have $\mathsf{Reach}(B_\delta(R_0), [t_0, t_0]) = B_\delta(R_0) = B_{\Delta_0}(R_0)$.

Lemma 3.2 indicates that $\forall t \in [t_0, t_1], \mathsf{Reach}(B_{\Delta_0}(R_0), [t_0, t_1]) \subseteq S$. And consider state $x = \theta \in R_0$, we also know $\xi(x,t) \in hull(R_0, R_1)$ and $\xi(x, t_1) \in R_1$. From Lemma 3.7, it follows that for $\forall x' \in B_{\Delta_0}(R_0), \forall t \in [t_0, t_1]$,

$$\|\xi(x,t) - \xi(x',t)\| \leq \|x - x'\| e^{b[1](t-t_0)}.$$

And at Line 14, $\Delta_1 \leftarrow (\Delta_0+\epsilon)e^{b[1](t_1-t_0)}$. Since $b[1]$ could be positive or negative, $\max_{t\in[t_0,t_1]} \|x-x'\|e^{b[1](t-t_0)} = \max\{\Delta_1, \Delta_0+\epsilon\}$. Therefore,

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

and at time t_1, $\xi(x',t_1)$ is at most Δ_1 distance to $\xi(x,t_1) \in R_1$, so $\mathsf{Reach}(B_\delta(R_0),[t_1,t_1]) = \mathsf{Reach}(B_{\Delta_0}(R_0),[t_1,t_1]) \subseteq B_{\Delta_1}(R_1)$.

Assuming that the lemma holds for $i = m-1$, we have $\mathsf{Reach}(B_\delta(R_0),[t_{m-1},t_{m-1}]) \subseteq B_{\Delta_{m-1}}(R_{m-1})$. Next we prove the lemma holds for $i = m$ as well. Consider state $x = \xi(\theta,t_{m-1}) \in R_{m-1}$, $\forall t \in [t_{m-1},t_m]$, by definition it follows that $\xi(x,t) \in hull(R_{m-1},R_m)$ and $\xi(x,t_m) \in R_m$. $\forall x' \in B_{\Delta_{m-1}}(R_{m-1}), \forall t \in [t_{m-1},t_m]$, from Lemma 3.7

$$\|\xi(x,t)-\xi(x',t)\| \le \|x-x'\|e^{b[m](t-t_{m-1})}.$$

Note at Line 14, $\Delta_m \leftarrow (\Delta_{m-1}+\epsilon)e^{b[m](t_m-t_{m-1})}$. Therefore, $\mathsf{Reach}(B_{\Delta_{m-1}}(R_{m-1}),[t_{m-1},t_m]) \subseteq hull(R_{m-1},\ R_m) \oplus B_{\max\{\Delta_m,\Delta_{m-1}+\epsilon\}}(0)$. And at time t_m, $\xi(x',t_m)$ is at most Δ_m distance to $\xi(x,t_m) \in R_m$. Hence, $\mathsf{Reach}(B_{\Delta_{m-1}}(R_{m-1}),[t_m,t_m]) \subseteq B_{\Delta_m}(R_m)$. Recall that $\mathsf{Reach}(B_\delta(R_0),[t_{m-1},t_{m-1}]) \subseteq B_{\Delta_{m-1}}(R_{m-1})$, thus $\mathsf{Reach}(B_\delta(R_0),[t_m,t_m]) \subseteq B_{\Delta_m}(R_m)$. ■

$\cup_{i=1}^k \{hull(R_{i-1},R_i) \oplus B_{\Delta_i'}(0)\}$ contains the $(B_\delta(R_0),\tau,T)$-reachtube of the system. Line 8 of the algorithm in Figure 1 is computed in this way. Now we are ready to prove the main theorem.

Theorem 3.9. *The items in array b computed by ComputeLDF are the coefficients of a $B_\delta(R_0)$-local piece-wise exponential discrepancy function (Definition 3.1).*

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 3.7, $\|\xi(x_1,t)-\xi(x_2,t)\| \le \|x_1-x_2\|e^{b[1](t-t_0)}$. Then consider $t \in [t_1,t_2]$. By Lemma 3.8 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 3.7 such that for any time $t \in [t_1,t_2]$, $\|\xi(x_1,t)-\xi(x_2,t)\| \le \|\xi(x_1,t_1)-\xi(x_2,t_1)\|e^{b[2](t-t_1)} \le \|x_1-x_2\|e^{b[2](t-t_1)+b[1](t_1-t_0)}$.

The procedure above can be performed iteratively as follows. For any time $t \in [t_{i-1},t_i]$, by lemma 3.8 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 3.7 it follows that

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

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 10–13, 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, , 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 piece-wise discrepancy function and the array b contains the corresponding coefficients. ■

3.3 Coordinate transformation

In this section, we will discuss the issue that the upper bound of the symmetric part of the Jacobian computed in Lines 10–13 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:

$$\dot{x} = \begin{bmatrix} 0 & 3 \\ -1 & 0 \end{bmatrix} x, \tag{8}$$

which has eigenvalues $\pm\sqrt{3}i$ and thus its trajectories oscillate. The symmetric part the of the Jacobian is $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ 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 linear transformation of x. The following is a coordinate transformed version of Lemma 3.7. 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 3.10. 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.

Proof. Let $z(t) = \xi(x_2, t) - \xi(x_1, t)$ and $y(t) = Pz(t)$ From (5) get:

$$\begin{aligned} \dot{y}(t) &= P\dot{z}(t) \\ &= P\left(\int_0^1 J_f(\xi(x_1, t) + sz(t))ds\right) z(t) \\ &= P\left(\int_0^1 J_f(\xi(x_1, t) + sz(t))ds\right) P^{-1}y(t) \\ &= \left(\int_0^1 \widetilde{J_f}(\xi(x_1, t) + sz(t))ds\right) y(t). \end{aligned}$$

Since for all $x \in S, \widetilde{J_f}^T(x) + \widetilde{J_f}(x) \preceq \tilde{\lambda}_{\max}(S)I$ and $\forall s \in [0, 1], \xi(x_1, t) + sz(t) \subseteq S$, we have

$$\frac{d\|y(t)\|^2}{dt} \leq 2\tilde{\lambda}_{\max}(S)\|y(t)\|^2,$$

which leads to: $\forall t \in [t_{i-1}, t_i]$

$$\|y(t)\| \leq \|y(t_{i-1})\| e^{\tilde{\lambda}_{\max}(S)(t-t_{i-1})}. \tag{9}$$

Substituting (9) in $z(t) = P^{-1}y(t)$:

$$\begin{aligned} \|z(t)\| &\leq \|P^{-1}\|\|y(t)\| \\ &\leq \|P^{-1}\|\|y(t_{i-1})\| e^{\tilde{\lambda}_{max}(S)(t-t_{i-1})} \\ &\leq \|P^{-1}\|\|P\|\|z(t_{i-1})\| e^{\tilde{\lambda}_{max}(S)(t-t_{i-1})} \\ &= cond(P)\|z(t_{i-1})\| e^{\tilde{\lambda}_{max}(S)(t-t_{i-1})}. \end{aligned} \tag{10}$$

■

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 $cond(P)$.

Let $\tilde{J} = PJP^{-1}$ the real Jordan form which looks like:

$$\left[\begin{array}{cc|c|cc} \lambda_1 & \epsilon & 0 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 & 0 \\ \hline 0 & 0 & \lambda_2 & 0 & 0 \\ \hline 0 & 0 & 0 & \lambda_3 & c \\ 0 & 0 & 0 & -c & \lambda_3 \end{array}\right]$$

where $2\epsilon < \lambda_1$ and $\lambda_1, \lambda_2, \lambda_3 \pm ci$ are the eigenvalues of J. There could be several more blocks like $\begin{bmatrix} \lambda_1 & \epsilon \\ 0 & \lambda_1 \end{bmatrix}$,λ_2 and $\begin{bmatrix} \lambda_3 & c \\ -c & \lambda_3 \end{bmatrix}$ in general. We use the matrix P as the coordinate transformation matrix for $J_f(x)$. In this approach the eigenvalues of $\frac{1}{2}(\tilde{J} + \tilde{J}^T)$ are $\lambda_1 + \frac{\epsilon}{2}, \lambda_2, \lambda_3$, which preserve the original eigenvalues to some extent.

In the previous example (8), 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 $P = \begin{bmatrix} 1 & 3 \\ -\sqrt{3} & \sqrt{3} \end{bmatrix}$ as the coordinate transformation matrix, $\tilde{J} = PJP^{-1} = \begin{bmatrix} 0 & \sqrt{3} \\ -\sqrt{3} & 0 \end{bmatrix}$, 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 $\prod cond(P[i])$.

Input: ψ,J_f,L_f,δ, ϵ,step
$\Delta \leftarrow \delta$,$b \leftarrow$ zeros(k), $K \leftarrow$ zeros(ceil(k/step))
for j = 1:step:k **do**
  $[V, D]$ = JordanDecom(average(R_j,...,$R_{j+step-1}$))
  $K(ceil(k/step))) \leftarrow cond(V)$
  for i = j:j+step-1 **do**
    $\tau \leftarrow t_i - t_{i-1}$
    $d \leftarrow (\Delta + \epsilon)e^{L_f \tau}$
    S $\leftarrow hull(R_{i-1}, R_i) \oplus B_d(0)$
    J $\leftarrow V J_f(center(S))V^{-1}$
    $\lambda \leftarrow \max(eig(J + J^T)/2)$
    error $\leftarrow$ upper $_{x \in S} \|V((J_f(x) + J_f^T(x)) - (J + J^T)) * V^{-1}\|$
    $b[i] \leftarrow \lambda +$ error$/2$
    $\Delta \leftarrow (\Delta + \epsilon)e^{b[i]\tau}$
  end for
  $\Delta \leftarrow K(ceil(k/step)))\Delta$
end for
return b, K

Figure 3: Algorithm *ComputeLDF* to coordinate transformation.

4 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 [17], 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 [16]. However,[17] still assumes that the user provides the IS-discrepancy for the smaller modules. In this section, we will show the approach used in 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.

4.1 Defining Local IS Discrepancy

Consider a dynamical system with inputs:

$$\dot{x} = f(x, u) \tag{11}$$

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 (11). Here $\mathcal{U}$ is the set $\{u | u : [0, \infty) \to \mathbb{R}^p\}$ of all input signals.

Definition 4.1. 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 $t \in \mathbb{R}_{\geq 0}$:

$$\|\xi(x, t) - \xi(x', t)\| \leq \beta(\|x - x'\|, t) + \int_0^t \gamma(\|u(s) - u'(s)\|) ds. \tag{12}$$

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 4.2. 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:

$\alpha(x_1, x_2, u_1, u_2, t) =$

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

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

4.2 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 9 S should be computed as in Lemma 4.3, 2) in Line 14 Δ should be updated as in Lemma 4.5. 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 3.2. Let $l = dia(\mathcal{I})$.

Lemma 4.3. 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 $\Delta' = (\Delta + \epsilon)(e^{L_f \tau_i}) + l L_f e^{L_f \tau_i} \tau_i$, $\tau_i = t_i - t_{i-1}$.

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 4.5. Before proving it, we need another generalized form of mean value theorem:

Lemma 4.4. 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 3.3. ∎

Lemma 4.5. Consider the i^{th} iteration of the loop for a dynamic system (11). 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,

$$\begin{aligned} \|\xi(x,t) - \xi(x',t)\| &\leq \|x - x'\| e^{a(t-t_{i-1})} \\ + \quad M e^{a(t-t_{i-1})} &\int_{t_{i-1}}^{t} \|u_1(\tau) - u_2(\tau)\| d\tau, \end{aligned} \tag{13}$$

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 $M = \max_{u \in \mathcal{U}(\mathcal{I})} (\|J_u(\xi(x,t), u)\|)$.

Proof. let $y(t) = \xi(x',t) - \xi(x,t)$ and $v(t) = u_2(t) - u_1(t)$. For a fixed time t, using Lemma 4.4

$$\begin{aligned} \dot{y}(t) &= \dot{\xi}(x',t) - \dot{\xi}(x,t) \\ &= f(\xi(x',t), u_2(t)) - f(\xi(x,t), u_1(t)) \\ &= \left(\int_0^1 J_x(\xi(x,t) + sy(t), u_2(t))ds\right) y(t) \\ &+ \left(\int_0^1 J_u(\xi(x,t), u_1(t) + \tau v(t))d\tau\right) v(t). \end{aligned}$$

We write $J_x(\xi(x,t) + sy(t), u_2(t))$ as J_x and $J_u(\xi(x,t), u_1(t) + \tau v(t))$ as J_u. Then the differentiating $\|y(t)\|^2$ with respect to t:

$$\begin{aligned} &\frac{d}{dt}\|y(t)\|^2 = y^T(t)\left(\int_0^1 (J_x^T + J_x)ds\right) y(t) \\ &+ v^T(t)\left(\int_0^1 J_u^T d\tau\right) y(t) + y^T(t)\left(\int_0^1 J_u d\tau\right) v(t) \\ &\leq y^T(t)\left(\int_0^1 J_x^T + J_x ds\right) y(t) + y^T(t) y(t) \\ &+ \left(\left(\int_0^1 J_u d\tau\right) v(t)\right)^T \left(\left(\int_0^1 J_u d\tau\right) v(t)\right). \end{aligned} \tag{14}$$

Recall that $\lambda_{max}(S)$ is the upperbound of the eigenvalues of the symmetric part of J_x over S, so $J_x^T + J_x \preceq 2\lambda_{max}(S) I$. Therefore, (14) becomes:

$$\frac{d}{dt}\|y(t)\|^2 \leq (2\lambda_{max}(S) + 1)\|y(t)\|^2 + \| \left(\int_0^1 J_u d\tau\right) v(t)\|^2.$$

Let $2a = 2\lambda_{max}(S) + 1$, $M = \max_{u \in \mathcal{U}(\mathcal{I})} (\|J_u(\xi(x,t),u)\|)$, then equation (14) becomes

$$\frac{d}{dt}\|y(t)\|^2 \leq 2a\|y(t)\|^2 + M\|v(t)\|^2. \tag{15}$$

Integrating each side from t_{i-1} to t where $t < t_i$, we have:

$$\|y(t)\|^2 \leq e^{2a(t-t_{i-1})}\left(\|y(t_{i-1})\|^2 + \int_{t_{i-1}}^{t} M\|v(\tau)\|^2 d\tau\right). \tag{16}$$

It follows that,

$$\|y(t)\| \leq e^{a(t-t_{i-1})}\|y(t_{i-1})\| + Me^{a(t-t_{i-1})}\int_{t_{i-1}}^{t} \|v(\tau)\| d\tau.$$

■

Using Lemma 4.5 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 4.6. *The items in array a and M are a coefficients of the* $(K,\mathcal{U}(\mathcal{I}))$*-local IS discrepancy function for the system* (11).

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, [17] 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 (11) with the input u being chosen nondeterministically in some compact set.

5 Experimental Evaluation

We have implemented the verification algorithm of Figure 1 and the *ComputeLDF* subroutine both with and without coordinate transformation in Matlab. The implementation and the examples are available from [12]. 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 3.2). 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.

5.1 Comparison with other tools

We compare the performance of our algorithm with two other tools, namely, Flow* [6] and HyCreate [22], 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 [21] 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 [13].

We have implemented verification algorithmwith and without coordinate transformation. Columns (#SimO) and (LDFO(s)) show the number of simulations and running time of our algorithm (Figure 2) without coordinate transformation. In comparison, Columns (#Sim) and (LDF) are the results with coordinate transformation. Coordinate transformation provides tighter bounds, so the number of simulations and running time decrease under the same environment (i.e. same initial sets and unsafe sets). 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. 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 "N/A" 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* 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.

5.2 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 2.

From the first 5 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 4 and 6-9 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 10-12, 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

	example	dim	δ	$\mathbb{U}$	#Sim	LDF(s)	#SimO	LDFO(s)	flow*(s)	HyCreate(s)
1	Vanderpol	2	0.5	x>2.0	9	0.378	61	2.01	11.2	2.776
2	Brusselator	2	0.5	x>1.3	21	1.01	85	2.79	11.8	1.84
3	Jet Engine	2	0.4	x>2.0	5	0.353	61	1.97	8.74	5.54
4	Robot arm	4	0.5	x>2.5	81	4.66	1159	47.9	169	>300
5	CoupledVanderpol	4	0.5	x>2.5	41	2.21	1353	54.2	93	49.8
6	Sinusoidal Tracking	6	0.5	x>10	185	13.2	753	97.0	258	>300
7	Lorenz Attractor	3	0.02	x>1e4	570	13.99	3105	72.0	53.4	N/A
8	Fixed-wing UAV (T=10)	7	3	x> 39	321	20.8	N/A	N/A	N/A	N/A
9	Helicopter	28	0.02	x>4	585	67.7	N/A	N/A	N/A	N/A
10	Fixed-wing UAV (T=50)	7	3	x> 39	321	99.8	N/A	N/A	N/A	N/A
11	Fixed-wing UAV (T=100)	7	3	x> 39	321	196	N/A	N/A	N/A	N/A

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: running time of our implementation (with coordinate transformation) in seconds; #SimO: number of simulations using algorithm in Figure 2; LDFO: running time of algorithm in Figure 2(without coordinate transformation)in seconds.

	δ	$\mathbb{U}$	saftey	#Sim	dia	RT(s)
1	0.6	θ >3	safe	17	5.6e-3	0.948
2	0.4	θ >3	safe	9	3.6e-3	0.598
3	0.3	θ >3	safe	9	2.6e-3	0.610
4	0.2	θ >3	safe	5	1.8e-3	0.444
5	0.1	θ >3	safe	1	1.5e-3	0.271
6	0.2	θ >2.5	safe	9	1.7e-3	0.609
7	0.2	θ >2.18	safe	17	1.4e-3	0.933
8	0.2	θ >2.17	safe	29	1.0e-3	1.429
9	0.2	θ >2.15	safe	161	9.2e-4	6.705
10	0.2	θ >2.14	unsafe	45	N/A	1.997
11	0.2	θ >2.13	unsafe	35	N/A	1.625
12	0.2	θ >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;

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.

	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.

6 Related Work

Simulation based verification has been studied in several papers recently [9, 1, 7, 18]. In [9] the authors introduce a general simulation based method for proving safety of arbitrary continuous systems. The novelty of their approach consist in the use of sensitivity analysis, where the *sensitivity matrix* with respect to initial state x_0 at time t is defined as $s_{x_0} \triangleq \frac{\partial \xi(x_0,t)}{\partial x_0}$. It is shown that $\dot{s}_{x_0}(t) = J_f(x_0,t)s_{x_0}(t)$ and $s_{x_0}(t)$ can be solved by efficient solvers. Then $\|s_{x_0}(t)\|\delta$ is used to bound the distance $\|\xi(x,t) - \xi(x_0,t)\|$ for $x \in B_\delta(x_0)$ at time t. It is shown that this upperbound

holds for linear time varying systems. For general nonlinear systems, $\|s_{x_0}(t)\|\delta$ has a quadratic error term with respect to δ that requires further analysis. Thus, this technique is sound for linear system but does not provide any formal guarantees for nonlinear systems ([9], page 13). In [7] this technique is extended to nonlinear systems subject to disturbances as inputs and uncertainty in the initial conditions to obtained an approximation that ignores the higher order terms. In contrast, in Section 3 and Section 4 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. In [18], the authors provide several approaches to capture the upperbound of the distance between two trajectories for linear systems and some polynomial systems.

In [15] the authors present a convenient implementation of sensitivity analysis in the Simulink software. Again, the trajectory sensitivity matrix can only be used as a linear approximation for a *perturbed trajectory* , instead of over-approximation of the reachset. In [1] the authors provide a different approach by linearizing the nonlinear system locally, and bounding the linearization error by Lagrange remainders. The original definition of discrepancy function can be seen as a generalization of the incremental stability [2]. The incremental Lyapunov function can be used as discrepancy function when a system is incrementally stable. An incremental Lyapunov function-based approach is used in [14]. Here the authors go much further and construct a finite symbolic model that is approximately bisimilar to the original switched system. Our approach bypasses the incremental stability requirement by focusing on bounded time analysis.

Contraction in [19] is defined as the region in which the eigenvalues of the symmetric part of the Jacobian is uniformly negative. The authors use "virtual displacement" to get the result, while we get the upperbound of the eigenvalues of the symmetric part of the Jacobian directly from the generalized mean value function. Contraction metrics introduced in [19] is also used in [10] to perform sound and relative complete analysis of nonlinear systems.

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 piece-wise 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.

In the future, we plan on using more rigorous ODE solvers like [5] and embedding the algorithm in verification tools like C2E2 [10] for safety verification of hybrid systems.

References

[1] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *CDC 2008. 47th IEEE Conference on*, pages 4042–4048. IEEE, 2008.

[2] D. Angeli. A lyapunov approach to incremental stability properties. *IEEE Transactions on Automatic Control*, 47(3):410–421, 2002.

[3] D. Angeli, E. D. Sontag, and Y. Wang. A characterization of integral input-to-state stability. *Automatic Control, IEEE Transactions on*, 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*. Springer, 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. Springer, 2013.

[7] T. Dang, A. Donzé, O. Maler, and N. Shalev. Sensitive state-space exploration. In *CDC 2008. 47th IEEE Conference on*, pages 4049–4054. IEEE, 2008.

[8] T. Dang and O. Maler. Reachability analysis via face lifting. In *HSCC*, pages 96–109. Springer, 1998.

[9] A. Donzé and O. Maler. Systematic simulation using sensitivity analysis. In *HSCC*, pages 174–189. Springer, 2007.

[10] P. S. Duggirala, S. Mitra, and M. Viswanathan. Verification of annotated models from executions. In *Proceedings of the Eleventh ACM International Conference on Embedded Software*, page 26. IEEE Press, 2013.

[11] 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 2014*, pages 215–229. Springer, 2014.

[12] 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.

[13] 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. Springer, 2011.

[14] A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *Automatic Control, IEEE Transactions on*, 55(1):116–126, 2010.

[15] Z. Han and P. J. Mosterman. Towards sensitivity analysis of hybrid systems using simulink. In *HSCC*, pages 95–100. ACM, 2013.

[16] Z. Huang, C. Fan, A. Mereacre, S. Mitra, and M. Z. Kwiatkowska. Invariant verification of nonlinear hybrid automata networks of cardiac cells. In *CAV 2014.*, pages 373–390. Springer, 2014.

[17] 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.

[18] A. A. Julius and G. J. Pappas. Trajectory based verification using local finite-time invariance. In *HSCC*, pages 223–236. Springer, 2009.

[19] W. Lohmiller and J.-J. E. Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998.

[20] N. Nedialkov. VNODE-LP: Validated solutions for initial value problem for ODEs. Technical report, McMaster University, 2006.

[21] B. B. Sharma and I. N. Kar. Design of asymptotically convergent frequency estimator using contraction theory. *Automatic Control, IEEE Transactions on*, 53(8):1932–1937, 2008.

[22] B. Stanley and C. Marco. Computing reachability for nonlinear systems with hycreate. In *Demo and Poster Session, HSCC*.

A Appendix: Proofs of Lemmas

Proof of Lemma 3.3:

In this proof, the i's in subscript correspond the the i^{th} components of the vector functions. For any $t \in [0,1], i \in \{1,\ldots,n\}$, we define $g_i(t) := f_i(x+tr)$. Then we have

$$f_i(x+r) - f_i(x) \quad = \quad g_i(1) - g_i(0) = \int_0^1 \frac{dg_i(t)}{dt} dt. \tag{17}$$

Using the chain rule of gradient, we have

$$\begin{aligned} \frac{dg_i(t)}{dt} &= \nabla f_i(u)|_{u=x+tr} \cdot \frac{d(x+tr)}{dt} \\ &= \nabla f_i(u)|_{u=x+tr} \cdot r = \sum_{j=1}^{n} \left.\frac{\partial f_i(u)}{\partial u_j}\right|_{u=x+tr} r_j, \end{aligned} \tag{18}$$

where $\nabla f_i(u) = [\frac{\partial f_i(u)}{\partial u_1}, \frac{\partial f_i(u)}{\partial u_2}, \ldots, \frac{\partial f_i(u)}{\partial u_n}]$ is the gradient of function f_i. Substituting (18) in (17), we have:

$$\begin{aligned} f_i(x+r) - f_i(x) &= \int_0^1 \left(\sum_{j=1}^{n} \left.\frac{\partial f_i(u)}{\partial u_j}\right|_{u=x+sr} r_j \right) ds \\ &= \sum_{j=1}^{n} \left(\int_0^1 \left.\frac{\partial f_i(u)}{\partial u_j}\right|_{u=x+sr} ds \right) r_j. \end{aligned}$$

Since $J_f(x+sr)$ is the matrix consisting of the components of $\left.\frac{\partial f_i(u)}{\partial u_j}\right|_{u=x+sr}$, the lemma holds.

Proof of Theorem 3.4.

This theorem is established by the minimax characterization of the eigenvalues. Let $\tilde{A} = A + E$, and let $\lambda_i(A), \lambda_i(E), \lambda_i(\tilde{A})$ denote the eigenvalues of A, E and $\tilde{A}$ respectively , where all three sets are arranged in non-increasing order. By the maxmin therorem we have

$$\lambda_k(\tilde{A}) = \min_{\dim \mathcal{V} = n-k+1} \left(\max_{0 \neq v \in \mathcal{V}} \rho_{\tilde{A}}(v) \right)$$

Which can also be written as

$$\begin{aligned} \lambda_k(\tilde{A}) &= \min \max (x^T \tilde{A} x) \\ x^T x = 1, p_i^T x = 0 (i = 1, 2, 3, \ldots, k-1) & \end{aligned}$$

Hence, if we take any particular set of p_i, we have for all corresponding x,

$$\lambda_k(\tilde{A}) \leq \max(x^T \tilde{A} x) = \max(x^T A x + x^T E x). \tag{19}$$

If $U^T A U = \Lambda = \mathrm{diag}(\lambda_i(A))$ and U is the orthogonal matrix, then if we take $p_i = U e_i$ the relations to be satisfied are

$$0 = p_i^T x = e_i^T y (i = 1, 2, \ldots, k-1)$$

With this choice of the p_i then the first $k-1$ components of y are zero, and from equation (19) we have

$$\lambda_k(\tilde{A}) \leq \max(x^T A x + x^T E x) \leq \max(\sum_{i=k}^{n} \lambda_i(A) y_i^2 + x^T E x) \tag{20}$$

However,

$$\sum_{i=k}^{n} \lambda_i(A) y_i^2 \leq \lambda_k(A) \tag{21}$$

while

$$x^T E x \leq \lambda_1(E) \tag{22}$$

for any x. Hence the expression in brackets of equation (20) is not greater than $\lambda_k(A)+\lambda_1(E)$ for any x corresponding to this choice of the p_i. Therefore its maximum is not greater than $\lambda_k(A) + \lambda_1(E)$ and we have

$$\lambda_k(\tilde{A}) \leq \lambda_k(A) + \lambda_1(E) \tag{23}$$

Since $A = \tilde{A}+(-E)$ and the eigenvalues of $-E$ in non-increasing order are $-\lambda_n(E), -\lambda_{n-1}(E), \ldots, -\lambda_1(E)$, and application of the result we have just proved gives

$$\lambda_k(A) \leq \lambda_k(\tilde{A}) + (-\lambda_n(E)) \quad \text{or} \quad \lambda_k(\tilde{A}) \geq \lambda_k(A) + \lambda_n(E) \tag{24}$$

Thus we have

$$\lambda_k(A) + \lambda_n(E) \leq \lambda_k(A + E) \leq \lambda_k(A) + \lambda_1(E)$$

The relations (23) and (24) imply that when E is added to A all of its eigenvalues are changed by an amount which lies between the smallest and greatest of the eigenvalues of E. Note that we are not concerned here specifically with small perturbations and the results are not affected by multiplicities in the eigenvalues of A, E and $A + E$.