Simulation-guided Contraction Analysis

Ayca Balkan[1] Jyotirmoy V. Deshmukh[2] James Kapinski[2] Paulo Tabuada[1]

Abstract— **We present a simulation-guided approach to perform contraction analysis of nonlinear dynamical systems. The central step in performing contraction analysis is to discover a contraction metric for a given system; however, there is no general technique to do so. Our approach leverages information from concrete system executions to construct a series of iteratively improved candidate contraction metrics. The final candidate metric is verified using nonlinear satisfiability solvers. We present several examples of systems with dynamics given by nonpolynomial functions, including an example from the automotive powertrain control domain.**

I. Introduction

In [1], Lohmiller and Slotine introduce *contraction analysis* as a tool for investigating the incremental stability of a given system. Through this paper, the authors brought contraction analysis to the attention of the control community (See [2] for a historical background of contraction analysis). Here, instead of asking whether system trajectories converge to a known nominal behavior, one asks if the trajectories converge to each other. The key idea in contraction analysis is to consider a *variation* around a given trajectory and reason about the behavior of this variation over time. If the magnitude of the variation diminishes over time, so does the actual distance between the trajectories. Just as a Lyapunov function shows stability of a system, in contraction analysis a *contraction metric* is a certificate for the point-wise decay of the variations along a trajectory.

Contraction analysis is closely related to incremental stability [3], as the existence of a contraction metric establishes incremental stability. In [4], [5], the authors present a procedure to design controllers that produce incrementally stable closed-loop behavior. Recently, there has been work in formalizing the connections between contraction and incremental stability. Simpson-Porco and Bullo [6] present rigorous proofs of various contraction results and highlight the necessary technical assumptions. Forni and Sepulchre [7] use Finsler structures to show contraction by defining Lyapunov functions on the tangent bundle. Another related notion is that of convergence. In [8], Rüffer *et al.* establish the relationship between convergent and incrementally stable systems.

Modern embedded controllers are used in safety-critical applications, which necessitates formal safety guarantees. While a technique based on, *e.g.*, simulations can help increase confidence in a system design, it does not suffice to prove correctness. In general, formally verifying the correctness of a design based on simulations would require an infinite number of simulations since there is an infinite number of initial conditions. This is in contrast to the techniques presented in this paper that either rely on a nonlinear solver to prove correctness or a compactness argument to prove that finitely many simulations suffice. Reachability analysis is one approach used to verify that a safety-critical closed-loop system stays in a desired safe region. Therefore, of particular interest to us is the application of contraction analysis to bounded-time reachability analysis [9], [10], [11].

Existing literature [12], [13] uses *auto-bisimulation* functions to show that for a given finite-time trajectory $\mathbf{x}(t)$ with initial condition $\mathbf{x}_0$, finite-time trajectories that start inside a ball of radius δ centered at $\mathbf{x}_0$ lie within a "tube" of radius δ around $\mathbf{x}(t)$. This allows performing comprehensive verification by exploring a finite number of simulation traces. Typically, auto-bisimulation functions are discovered using sum-of-squares (SoS) optimization-based methods, which restricts them to polynomial dynamical systems. After investigating a number of examples, we observed that discovering polynomial auto-bisimulation functions is a rather difficult task. Hence, in this work, we explore contraction metrics, which, like auto-bisimulation functions can also be used to construct tubes around finite-time simulation traces.

In addition to reachability analysis, incremental stability and contraction have been used for synchronization of complex networks [14], construction of symbolic models for nonlinear control systems [15], and design of nonlinear observers and optimal controllers [16], [4]. Clearly, contraction metrics have diverse applications in design and analysis of control systems. Unfortunately, a similarity with Lyapunov theory persists: just as computing Lyapunov functions for arbitrary systems is an unsolved problem, computing contraction metrics is also challenging.

Aylward et al. [17] use SoS optimization for systems with polynomial dynamics to compute contraction metrics, by reducing the problem to finding a feasible solution of a semidefinite program (SDP). To our knowledge, there is no prior work on finding contraction metrics for nonpolynomial systems. In our experience, SDP solvers can produce contraction metrics that are unsound, *i.e.*, contraction conditions may not be strictly satisfied at all points. Additional care is also required to ensure that the underlying optimization problem is well-conditioned.

In this paper, we propose a simulation-guided technique to find provably correct contraction metrics for arbitrary nonlinear dynamical systems. Our approach is similar to those in [18], [19] for finding Lyapunov functions from system traces; here, the key insight is that simulations can guide the

[1] Ayca Balkan and Paulo Tabuada are with the Department of Electrical Engineering at the University of California, Los Angeles, abalkan@ucla.edu, tabuada@ee.ucla.edu

[2] Jyotirmoy V. Deshmukh and James Kapinski are with Toyota Technical Center, Gardena, CA, jyotirmoy.deshmukh@tema.toyota.com, jim.kapinski@tema.toyota.com.

search algorithm when the system dynamics are complex. This is especially valuable in an industrial setting where system dynamics are often nonpolynomial or even lacking an closed-form representation. On the other hand, in practical settings, system models in rich simulation environments such as Simulink® are typically available, and obtaining high-fidelity simulation traces is standard. As our technique for constructing candidate contraction metrics can work without the knowledge of the system dynamics, it is applicable to a significantly more diverse set of systems compared to those that can be analyzed by SoS techniques.

We now give a brief overview: We formalize the background and notation for contraction analysis in Sec. II. In Sec. III, we show how we can use simulations to discover candidate contraction metrics. We assume that the candidate has a specific template form; for our experiments we pick the form used in [17]. Here, the contraction metric is defined by a matrix, the entries of which are polynomials of some fixed degree but undetermined coefficients. Using simulation traces, we obtain a set of necessary constraints for the contraction metric, each of which is a linear matrix inequality (LMI), and solve these using an SDP solver to obtain a contraction metric *candidate*. Next, we try to identify a system trace for which the contraction conditions are not valid for the current candidate; such a trace is called a *counterexample*. To do this, we frame the search for a counterexample as an optimization problem on the trajectory space, such that if the obtained minimum is negative, then the corresponding trajectory is a counterexample. If found, we use the counterexample to update the set of LMI constraints, and thus to refine the candidate contraction metric. If no counterexample is found, the algorithm terminates.

In general, a global optimizer is incomplete, *i.e.*, there is no guarantee that it will find a global minimum. Thus, after our search procedure terminates, if we have knowledge of the system dynamics, we show how we can verify whether the candidate metric strictly satisfies the contraction metric conditions in Sec. III. For this, we leverage existing tools like Mathematica® [20] and dReal [21]. We also discuss a sampling-based verification procedure as an alternative to formal decision procedures, and provide the necessary conditions required for such a procedure to be sound.

In Sec. V, we illustrate the effectiveness of our technique on benchmark systems with both polynomial and nonpolynomial dynamics. We also provide an example from the automotive domain with four states and highly nonlinear dynamics. Finally, we present our conclusions in Sec. VI.

II. Preliminaries

We consider dynamical systems of the form:

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

where $\mathbf{x}(t) \in \mathcal{X}$ is the state of the system at time $t \in \mathbb{R}_{\geq 0}$, $\mathcal{X}$ is the state space, a compact subset of $\mathbb{R}^n$, and the function $f : \mathcal{X} \to \mathcal{X}$ is continuously differentiable.

We say that $\psi : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathcal{X}$ is a *trajectory* of the system if $\psi(\mathbf{x}_0, t)$ is the solution to the initial value problem for (1), with $\psi(\mathbf{x}_0, 0) = \mathbf{x}_0$. The discrete time *trace* of the system, denoted by ϕ, is a function $\phi : \mathcal{X} \times T \to \mathcal{X}$, where $T = \{t_1, \ldots, t_N\} \subset \mathbb{R}_{\geq 0}$ is a finite set of time instants, N is a positive integer greater than 1, and there exists a trajectory $\psi(\mathbf{x}_0, t)$ such that $\phi(\mathbf{x}_0, t_j) = \psi(\mathbf{x}_0, t_j)$, for $j = 1, 2, \ldots N$. We define the *length* of ϕ as $N - 1$.

Definition 1 (Forward Invariant Set): A set of states $I \subseteq \mathcal{X}$ of the system described in (1) is called *forward invariant* if for all $\mathbf{x}_0 \in I$ and for all $t \geq 0$, $\psi(\mathbf{x}_0, t) \in I$.

Definition 2 (Lyapunov function): Let $\mathbf{0}$ be an equilibrium point of system (1). Given a set $\mathcal{X}$, a function $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is called a Lyapunov function over $\mathcal{X}$ if the following holds:

$$\forall \mathbf{x} \in \mathcal{X} \backslash \{\mathbf{0}\} : \quad V(\mathbf{x}) \ > 0, \ V(\mathbf{0}) = 0 \tag{2}$$

$$\forall \mathbf{x} \in \mathcal{X} : \quad \nabla V(\mathbf{x})^T \cdot f(\mathbf{x}) \ \leq 0. \tag{3}$$

The *sublevel set* $\mathcal{S}_\ell$ of a Lyapunov function $V(\mathbf{x})$ is $\mathcal{S}_\ell = \{\mathbf{x} \mid V(\mathbf{x}) \leq \ell\}$ where $\ell \in \mathbb{R}$. Just as Lyapunov functions can be used to show asymptotic and Lyapunov stability of a system, we now turn to contraction metrics, which can be used to establish incremental stability[1].

We first introduce some terminology. We say that a matrix $P \in \mathbb{R}^{n \times n}$ is negative definite if $\mathbf{y}^T P \mathbf{y} < 0$, for all $\mathbf{y} \in \mathbb{R}^n$. A *matrix function* $M : \mathbb{R}^n \to \mathbb{R}^{m \times m}$ maps a state $\mathbf{x}$ to a matrix $M(\mathbf{x})$. Given a matrix function $A(\mathbf{x})$ defined over a domain $\mathcal{X}$, we write $A(\mathbf{x}) \prec 0$ if $A(\mathbf{x})$ is negative definite for all $\mathbf{x} \in \mathcal{X}$. We next provide a brief theoretical background on contraction analysis.

A. Contraction Analysis

Given system (1), contraction metrics are certificates of the asymptotic convergence of the system trajectories to one another. The idea introduced in [1] is the following: if neighboring trajectories converge to each other, then all trajectories asymptotically converge to a single trajectory. Going from a local convergence to a global convergence result is important because this eliminates the need to directly construct a decreasing distance between every pair of trajectories.

To show the local convergence of trajectories, Slotine et.al show that the *variation*[2] around the trajectories goes to zero. The variation at time t is denoted by $\delta\mathbf{x}(t)$ and satisfies the following differential relation:

$$\dot{\delta\mathbf{x}} = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \delta\mathbf{x}(t). \tag{4}$$

Notice that along a given trajectory $\phi(x_0, t)$ of (1), $\dot{\delta\mathbf{x}} = \left[\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\phi(\mathbf{x}_0,t)}\right] \delta\mathbf{x}$, is the linearization of (1), along this trajectory.

We can prove contraction by showing that the magnitude of the variation decreases around the trajectories of the system. We do this by constructing a suitable (Riemannian) metric such that the norm of the variation induced by this

[1]Note that asymptotic stability of a system implies incremental stability on a compact set [3]; however, given a Lyapunov function, there is no constructive proof in the literature to find the distance between the trajectories as a class $\mathcal{K}$ function of the distance between the initial conditions.

[2]The interested reader can refer to [6] or [7] for a formal definition of the notion of a variation.

metric decreases around the trajectories of (1). The following definitions formalize this:

Definition 3 (Contraction Metric [1]): Given the system in (1), let $J(\mathbf{x})$ be an abbreviation for $\frac{\partial f}{\partial \mathbf{x}}(\mathbf{x})$. Let $\mathcal{C}$ be a forward invariant and connected subset of $\mathcal{X}$. A matrix function $M(\mathbf{x})$ is called a contraction metric if $\forall \mathbf{x} \in \mathcal{C}$ the following conditions hold:

$$M(\mathbf{x}) \succ 0, \tag{5}$$

$$J(\mathbf{x})^T M(\mathbf{x}) + M(\mathbf{x}) J(\mathbf{x}) + \frac{\partial M(\mathbf{x})}{\partial x} f(\mathbf{x}) \prec 0. \tag{6}$$

Here, the region $\mathcal{C}$ is called the *contraction region*.

Definition 4 (Induced Metric): The matrix function $M(\mathbf{x})$ induces a metric d_M on the set $\mathcal{C}$, where $d_M(\mathbf{x}_1, \mathbf{x}_2)$ is defined by:

$$d_M(\mathbf{x}_1, \mathbf{x}_2) = \inf_{\gamma \in \Gamma(\mathbf{x}_1, \mathbf{x}_2)} \int_\gamma \sqrt{\delta \mathbf{x}^T M(\mathbf{x}) \delta \mathbf{x}} \circ \gamma \, \mathrm{d}\gamma, \tag{7}$$

where, $\Gamma(\mathbf{x}_1, \mathbf{x}_2)$ is the set of all piecewise C^1 curves on $\mathcal{C}$ connecting $\mathbf{x}_1$ and $\mathbf{x}_2$.

In general, computing the induced metric, d_M is nontrivial. Using arguments similar to those in [22], $d_M(\mathbf{x}_1, \mathbf{x}_2)$ can be bounded by the Euclidean distance, if $M(\mathbf{x})$ satisfies the following for all $x \in \mathcal{C}$, for some positive c_1 and c_2:

$$c_1 I \preceq M(\mathbf{x}) \preceq c_2 I.$$

When each entry of the matrix function $M(\mathbf{x})$ is a constant, then computing the induced metric is straightforward. In particular, when $M(\mathbf{x})$ is the identity matrix, the induced metric is the usual Euclidean distance metric.

Definition 5 (Open Ball): The set $\{\mathbf{y} \in \mathbb{R}^n \,|\, d_M(\mathbf{x}, \mathbf{y}) < \epsilon\}$ is called the open ball of radius ϵ centered at $\mathbf{x} \in \mathbb{R}^n$ with respect to the metric d_M, and is denoted as $\mathcal{B}^\epsilon_M(\mathbf{x})$.

The convergence result is formalized in Theorem 1. A formal proof for this theorem can be found in [7].

Theorem 1: [1], [7] Consider the system equations (1) and a forward invariant contraction region $\mathcal{C}$ with respect to the contraction metric $M(\mathbf{x})$. Let $\mathbf{x}(t)$ be a system trajectory with the initial condition $\mathbf{x}_0$, then all trajectories with initial condition in $\mathcal{B}^\epsilon_M(\mathbf{x}_0) \subseteq \mathcal{C}$, stay in $\mathcal{B}^\epsilon_M(\mathbf{x}(t)) \subseteq \mathcal{C}$ and asymptotically converge to $\mathbf{x}(t)$.

Note that, for linear systems, contraction is equivalent to the asymptotic stability of the system. Therefore finding a Lyapunov function is sufficient for proving contraction. For nonlinear systems, the current methods to identify contraction metrics are based on SoS techniques [17]. Thus, these methods can only address relaxations of the original problem based on the S-procedure to find polynomial contraction metrics on systems with polynomial or polynomialized dynamics.

III. Discovering Contraction Metrics

In this section, we present our algorithm to construct a candidate contraction metric and iteratively refine it using counterexamples. This algorithm is similar in spirit to the algorithm described in [18] for computing Lyapunov functions by using simulations; however, as we will show, significant modifications are required when searching for contraction metrics.

A. Approximating System Dynamics

In this step of our method, we assume that the system dynamics contain complex nonpolynomial expressions. Thus, instead of reasoning over the expressions for the vector field $f(\mathbf{x})$ and symbolic computation of its Jacobian (denoted by $J(\mathbf{x})$), we instead numerically estimate these quantities from discrete-time simulation traces.

Let Φ denote the flow function associating an initial state $\mathbf{x}_0 \in \mathcal{C}$ and a time $t \in \mathbb{R}$ with the solution at time t ($\mathbf{x}(t)$) of the initial value problem defined by $\mathbf{x}(0) = \mathbf{x}_0$ and the ODE in (1). Recall that the $(i,k)^{th}$ entry of the Jacobian matrix $J(\mathbf{x})$ is denoted by $J_{i,k}(\mathbf{x})$, which represents $\frac{\partial f_i}{\partial x_k}(\mathbf{x})$. Let 1_k denote a vector of length n where are all values are 0, except the k^{th} value which is 1. Let $\widetilde{f}(\mathbf{x})$ and $\widetilde{J}(\mathbf{x})$ represent the approximations of $f(\mathbf{x})$ and $J(\mathbf{x})$, respectively, that are defined as follows:

$$\tilde{f}_i(\mathbf{x}) \approx \frac{\Phi(\mathbf{x}, \Delta t) - \mathbf{x}}{\Delta t} \tag{8}$$

$$\widetilde{J}_{i,k}(\mathbf{x}) \approx \frac{\tilde{f}_i(\mathbf{x} + \Delta x \cdot 1_k) - \tilde{f}_i(\mathbf{x})}{\Delta x}, \tag{9}$$

for each $i, k \in \{1, \ldots, n\}$.

In a practical implementation, $\tilde{f}_i(\mathbf{x})$ and $\widetilde{J}_{i,k}(\mathbf{x})$ can be computed in the following way:

1) Select a sample point $\mathbf{x} \in \mathcal{C}$,
2) To to obtain $\widetilde{f}_i(\mathbf{x})$, as per (8), use a simulation tool to obtain one time-step discrete time trace with the fixed step-size Δt from the initial condition $\mathbf{x}$,
3) For each $k \in \{1, \ldots, n\}$, perturb the point $\mathbf{x}$ by Δx along the k^{th} dimension, to obtain the point $\mathbf{x}'_k = \mathbf{x} + \Delta x.1_k$, and,
4) To obtain $\widetilde{J}_{i,k}(\mathbf{x})$, as per (9), use the simulation tool to obtain a single time-step trace (as in the second step), from the initial condition $\mathbf{x}'_k$.

In our implementation, we use discrete time simulation traces with multiple steps. We also allow using an ODE solver with variable time steps. Given a discrete time trace ϕ at times $t_1, \ldots, t_N$, the above approximations can be repeated $N-1$ times at each of the points $\phi(t_1)$ through $\phi(t_{N-1})$, using the time step $\Delta t = t_{i+1} - t_i$ for the calculations in the i^{th} iteration.

B. Computing the Contraction Metric

In what follows, we assume a template form for the metric. We pick $M(\mathbf{x})$ to be a matrix whose entries are polynomials in $\mathbf{x}$ of some fixed degree, with unknown coefficients. Next, we show a procedure to obtain constraints that encode necessary conditions for a contraction metric to be valid on a trace ϕ defined at time instants $t_1, \ldots, t_N$. To generate constraints from a set of traces, we simply repeat the procedure for each trace. Constraint (10) enforces positive definiteness of the contraction metric at each point in ϕ.

$$M(\phi(t_j)) \succ 0, \forall j \in 1 \ldots N \tag{10}$$

The second necessary condition on a contraction metric pertains to its time-derivative. Recall from (6) that this is given by the expression $J(\mathbf{x})^T M(\mathbf{x}) + M(\mathbf{x}) J(\mathbf{x}) + \frac{\partial M(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x})$. At a given point on the trace, we approximate J by $\widetilde{J}$ as shown in the previous section. Since $\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}) = \frac{dM(\mathbf{x}(t))}{dt}|_{t=0}$, it is approximated as

$$\widetilde{D}(\mathbf{x}) := \frac{M(\Phi(\mathbf{x}, \Delta t)) - M(\mathbf{x})}{\Delta t}. \tag{11}$$

The following constraint enforces that the time-derivative of the contraction metric is negative definite at each point in ϕ:

$$\widetilde{J}(\phi(t_j))^T M(\phi(t_j)) + M(\phi(t_j))\widetilde{J}(\phi(t_j)) + \widetilde{D}(\phi(t_j)) \prec 0, \ \forall j \in 1 \ldots N. \tag{12}$$

We call the procedure to take in a set of discrete time traces and a maximum degree ℓ for the polynomials in $M(\mathbf{x})$, and formulate a semidefinite program $\mathcal{P}$ by adding constraints of the form (10) and (12) for each trace as `GenerateConstraints`.

C. Iteratively Improved Candidate Contraction Metrics

Note that (10) and (12) are necessary but not sufficient conditions for (5) and (6) to be valid. Hence, we need to determine if the contraction metric obtained in the previous section is valid for all traces in the conjectured contraction region $\mathcal{C}$. Any trace for which the contraction metric candidate is not valid is a counterexample trace, which we add to the set of available simulation traces, and then use `GenerateConstraints` to obtain a revised semidefinite program. The solution of this program provides a refined candidate contraction metric. The pseudo-code for these steps is presented in Algorithm 1.

For ease of exposition, we assume a fixed time-step ODE solver, with a user-provided time-step. The extension to a variable time-step ODE solver is straightforward.

The inputs to Algorithm 1 are the candidate contraction region $\mathcal{C}$, the time step Δt for the ODE solver, a desired length N of the simulation traces, the degree ℓ for the polynomials in $M(\mathbf{x})$, and a function $\Phi_{\mathsf{sim}}(\mathbf{x}, \Delta t, N)$ that provides discrete time simulation traces of length N of the system at fixed time-steps of Δt from the given initial condition $\mathbf{x}$. We now elaborate on the key steps of the algorithm:

Initialization. First, we initialize the algorithm with a set of discrete time simulation traces. These are obtained by: 1.) uniform random sampling of the region $\mathcal{C}$ to obtain a set of initial states (Line 1), and 2.) simulating the system (1) for N steps with the fixed time step Δt (Line 3). For our prototype implementation, we assume the system is given as an ODE, and therefore use the numerical integration tool `ode45` in MATLAB® to generate the simulation traces.

Candidate Construction. Using the simulation traces thus obtained, we formulate a semidefinite program using `GenerateConstraints` from Sec. III-B, and solve it (Line 7–8). To solve the semidefinite program, we use the freely available SeDuMi [23], SDPT3 [24], and YalMIP [25] packages.

Falsifier. A key procedure in Algorithm 1 is the `Falsify` subroutine (Line 13) that obtains counterexamples to the

Algorithm 1: Pseudo-code for obtaining iteratively improved contraction metric candidates

```
Input: $\mathcal{C}$, $\Delta t$, $N$, $\ell$, $\Phi_{\mathsf{sim}}$
Output: $M(\mathbf{x})$
1  Initial_States := Sample($\mathcal{C}$) ; $\tau$:=$\emptyset$
2  foreach $\mathbf{x}_0 \in$ Initial_States do
3      $\tau := \tau \cup \{\Phi_{\mathsf{sim}}(\mathbf{x}_0, \Delta t, N)\}$
4  end
5  $\rho^* := -\infty$
6  while $\rho^* < 0$ do
7      $\mathcal{P}$ := GenerateConstraints($\tau, \ell$)
8      (flag, $M(\mathbf{x})$) := Solve($\mathcal{P}$)
9      if flag = infeasible then
10         return "Refine template: perhaps, increase $\ell$."
11         break
12     end
13     $(\rho^*, \phi^*)$ := Falsify($\Phi_{\mathsf{sim}}, \Delta t, N$)
14     $\tau := \tau \cup \{\phi^*\}$
15 end
16 return "Candidate Contraction Metric $M(\mathbf{x})$"
```

candidate contraction metric. Let τ^N be the shorthand notation for $\{\phi \mid \phi \in \Phi_{\mathsf{sim}}(\mathbf{x}_0, \Delta t, N), \mathbf{x}_0 \in \mathcal{C}\}$, *i.e.*, the set of all discrete time traces of length N and fixed time-step Δt that start from some state $\mathbf{x}_0$ in $\mathcal{C}$ and stay in $\mathcal{C}$ at all times. Further, let $\lambda_{max}(M)$ and $\lambda_{min}(M)$, denote the largest eigenvalue and the smallest eigenvalue of a given symmetric matrix M respectively, and let $\widetilde{R}(\mathbf{x})$ be the shorthand notation for $\widetilde{J}(\mathbf{x})^T M(\mathbf{x}) + M(\mathbf{x})\widetilde{J}(\mathbf{x}) + \widetilde{D}(\mathbf{x})$. In this procedure, we use a global optimizer to solve the optimization problem shown in (13).

$$\min_{\phi \in \tau^N} \min_{j \in 1 \ldots N} \left(-\lambda_{max}\left(\widetilde{R}\left(\phi\left(t_j\right)\right)\right), \lambda_{min}\left(M\left(\phi\left(t_j\right)\right)\right)\right) \tag{13}$$

Essentially, the outer $\min$ quantifies over the set of all traces in τ^N, and the inner $\min$ is the minimum of two quantities: negative of the largest eigenvalue of $\widetilde{R}(\mathbf{x})$ and the smallest eigenvalue of $M(\mathbf{x})$ at any such point. If the first quantity is negative, *i.e.*, if the largest eigenvalue of $\widetilde{R}(\mathbf{x})$ is positive, it violates the condition in Eq. (6). If the second quantity is negative, *i.e.*, if the smallest eigenvalue of $M(\mathbf{x})$ is negative, it violates the condition in Eq. (5). Thus, if the minimum ρ^* is negative, then the solution of the optimization problem, *i.e.*, the trace ϕ^* is a counterexample to the candidate contraction metric. In this case, we add ϕ^* to the set of traces τ (Line 14) and go back to Line 7. If $\rho^* > 0$, then the algorithm terminates and outputs a contraction metric candidate $M(\mathbf{x})$.

Remark 1: In our implementation of the `Falsify` procedure, we use a global optimizer based on the Nelder-Mead or the downhill simplex method. This optimizer takes as input a seed point to commence the optimization. We found it useful to perform uniform random sampling of the region $\mathcal{C}$ to obtain a set of initial seeds for the optimizer. As Nelder-Mead based optimizers can converge to non-stationary points, seeding the search from a sufficient number of points in the search space seems to provide better results.

Finally we note that if `Solve`$(\mathcal{P})$ indicates that the semidefinite problem to be solved is infeasible, then we halt the algorithm. This can happen due to several reasons: 1.) $\mathcal{C}$ is not a contracting region, or, 2.) the contraction metric does not have an appropriate template form, and a potential solution is to increase the degree ℓ of the polynomials in $M(\mathbf{x})$, or, 3.) numerical problems related to ill-conditioned constraints have been encountered. Such a non-result can also provide valuable insight to the designer.

As the global optimization step is inherently inexhaustive and lacking formal guarantees, we must verify that the candidate contraction metric obtained using Algorithm 1 strictly satisfies (5) and (6). In the next section, we show how we can exploit techniques from nonlinear satisfiability solvers to verify the contraction metric candidate computed by Algorithm 1.

IV. Verification for white-box systems

To ensure validity of the candidate contraction metric produced by Algorithm 1, we need to check if it satisfies the conditions in Eq. (5–6) for all $\mathbf{x} \in \mathcal{C}$. In other words, we need to check that $M(\mathbf{x}) \succ 0$ and $R(\mathbf{x}) \prec 0$, for all $\mathbf{x} \in \mathcal{C}$. (Recall that $R(\mathbf{x})$ is shorthand for the expression $(J(\mathbf{x})^T M(\mathbf{x}) + M(\mathbf{x})J(\mathbf{x}) + \frac{\partial M(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}))$.

Checking whether $M(\mathbf{x}) \succ 0$ for all $\mathbf{x} \in \mathcal{C}$, is equivalent to checking if $\forall \mathbf{x} \in \mathcal{C}$, the smallest eigenvalue of $M(\mathbf{x})$ is positive. We can convert this validity query into checking the satisfiability of its negation:

$$\exists \mathbf{x} : (\mathbf{x} \in \mathcal{C}) \wedge (\lambda_{min}(M(\mathbf{x})) < 0). \tag{14}$$

Similarly, to check the negative definiteness of $R(\mathbf{x})$, it is sufficient to check that the largest eigenvalue of $R(\mathbf{x})$ is negative over all $\mathbf{x}$. Equivalently, we wish to ascertain that the query in (15) is unsatisfiable.

$$\exists \mathbf{x} : (\mathbf{x} \in \mathcal{C}) \wedge (\lambda_{max}(R(\mathbf{x})) > 0). \tag{15}$$

We can use any decision procedure for deciding queries over polynomial functions of real-valued variables to decide Query (14). Following the decidability of queries over real closed fields established by Tarski [26], Query (14) is decidable as long as we choose $M(\mathbf{x})$ to be a matrix function with entries that are polynomial functions. In our implementation, we typically rely on the `Reduce` function in the Wolfram Mathematica® tool. [3]

Query (15) can be much more difficult to solve because the expression for $R(\mathbf{x})$ includes the vector field, which could contain nonpolynomial expressions. While the language of queries allowed in Mathematica® includes nonpolynomial expressions, in our experience, it cannot efficiently handle such queries, especially when the size of the query or the number of independent variables in the query is large.

In this section, we propose two different approaches to decide $R(\mathbf{x}) \prec 0$. The first is based on utilizing interval constraint propagation (ICP)-based solvers for nonlinear theories and the second is based on a sample-and-check scheme.

[3]In Mathematica, we formulate this query as $\exists \mathbf{x}, \mathbf{y} : (\mathbf{x} \in \mathcal{C}, \mathbf{y} \in \mathbb{R}^n) \wedge (\mathbf{y}^T M(\mathbf{x})\mathbf{y} < 0)$

A. Verification with ICP-based SMT solvers

In an interval constraint propagation (ICP)-based technique, we assume that individual variables appearing in a query take values from an interval ($\subseteq \mathbb{R}$). Such interval domains for the variables define a hyperbox over which the ICP-based solver looks for a satisfying solution to a given query. The technique works by pruning the interval domains while ensuring that any value that may be consistent with a set of constraints is not removed. ICP is typically combined with a branch and bound algorithm, which helps obtain quick but approximate results for deciding satisfiability of nonlinear constraints.

For example, dReal [21] and iSat [27] are such solvers, and we focus on using dReal for our validation problems. dReal supports various nonlinear elementary functions in the framework of δ-complete decision procedures, and returns "unsat" or "δ-sat" for a given query, where δ is a precision value specified by the user. When the answer is "unsat", the meaning is that a δ-strengthening of the original query remains unsatisfiable. Further, dReal produces a proof of unsatisfiability; when it returns "δ-sat", it gives an interval of size δ, which contains points that may possibly satisfy the query.

To decide Query (15) in dReal, we make use of the characteristic polynomial[4] of $R(\mathbf{x})$, which we denote by $C_R(\mathbf{x}, \lambda)$. Instead of checking for the largest eigenvalue, we formulate the stronger query (16) that checks if *any* eigenvalue of $R(\mathbf{x})$ is positive. Unsatisfiabilty of Query (16) shows negative definiteness of $R(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{C}$.

$$\exists \mathbf{x}, \lambda : (\mathbf{x} \in \mathcal{C}) \wedge (\lambda > 0) \wedge (C_R(\mathbf{x}, \lambda) = 0) \tag{16}$$

Though we have used Mathematica® to check positive definiteness of $M(\mathbf{x})$, note that we can also formulate a similar query for checking positive definiteness of $M(\mathbf{x})$ using dReal. If both queries (14,16) are unsatisfiable, this means that the candidate contraction metric $M(\mathbf{x})$ is in fact a valid contraction metric over $\mathcal{C}$.

B. Verification using Dense Sampling

There are two limitations in using ICP-based solvers like dReal or decision procedure-based solvers like Mathematica®. First, the computational complexity of queries increases exponentially with the number of variables (here, this is the number of state variables). The difficulty is also tied to the actual system dynamics; we have observed that with an increase in the number of nonpolynomial terms, the time required to decide the queries also increases. The second difficulty is fundamental; transcendental extensions of the theory of real closed fields are undecidable [28]. Thus, for arbitrary system dynamics, the query $R(\mathbf{x}) \prec 0$ would be a sentence in a theory that is undecidable.

Hence, we now discuss an alternative approach to verify $R(\mathbf{x}) \prec 0$. The basic scheme is very simple: perform uniform random sampling of states in $\mathcal{C}$, and for each sampled state

[4]Recall that the characteristic polynomial of a square matrix A is a univariate polynomial (in λ) defined by the expression $\det(\lambda I - A)$. Here I is the identify matrix of the same dimension as A.

$\mathbf{x}_\delta$, check the eigenvalues of $R(\mathbf{x}_\delta)$. The result we present shows how such a sample-and-check procedure can be sound. In other words, we show how we can establish conditions on the sampling density that allow us to conclude that $R(\mathbf{x}) \prec 0$ for all $\mathbf{x} \in \mathcal{C}$ by verifying the eigenvalues of $R(\mathbf{x})$ at only *finitely many* $\mathbf{x}_\delta \in \mathcal{C}$.

Definition 6 (δ-Sampling): Given a $\delta \in \mathbb{R}_{\geq 0}$, a δ-sampling of set $\mathcal{C}$ is a finite set $\mathcal{C}_\delta$ such that the following holds: $\mathcal{C}_\delta \subset \mathcal{C}$, and, for any $\mathbf{x} \in \mathcal{C}$, there exists a $\mathbf{x}_\delta \in \mathcal{C}_\delta$ such that $\|\mathbf{x} - \mathbf{x}_\delta\| < \delta$.

The core intuition is that under certain assumptions on $f(\mathbf{x})$, one can show that the eigenvalues of the state dependent matrix $R(\mathbf{x})$, do not change "too fast" with changes in $\mathbf{x}$. We start by the following Lemma from [29].

Lemma 2: Let A and B be Hermitian matrices. Then we have the following eigenvalue stability inequality[5] for each i:

$$|\lambda_i(A) - \lambda_i(B)| \leq \|A - B\|_F, \tag{17}$$

where $\lambda_i : \mathbb{R}^{n\times n} \to \mathbb{R}$, denotes a function that maps a matrix to its i^{th} largest eigenvalue, and $\|(\cdot)\|_F$ denotes the Frobenius-norm [6] of a given matrix.

Let $f_i(\mathbf{x})$, $M(\mathbf{x})$, $J(\mathbf{x})$ and $\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}}$ be Lipschitz continuous functions over the compact set $\mathcal{C} \subset \mathbb{R}^n$, and let L be the matrix whose $(i,j)^{th}$ element is the Lipschitz constant [7] of the $(i,j)^{th}$ element of $R(\mathbf{x})$. Next, we use Lemma (2), along with these Lipschitz continuity assumptions to make the argument that to show $R(\mathbf{x}) < 0$ on $\mathcal{C}$ one only needs to show $R(\mathbf{x})$ is negative definite on a sufficiently dense δ-sampling of $\mathcal{C}$.

Theorem 3: Given a contraction metric $M(\mathbf{x})$ over $\mathcal{C}$, where $\frac{d}{dt}\big|_{t=0}(\delta\mathbf{x}^T M(\mathbf{x})\delta\mathbf{x} = \delta\mathbf{x}^T R(\mathbf{x})\delta\mathbf{x}$, and a δ-sampling of $\mathcal{C}$, i.e. $\mathcal{C}_\delta$, such that $\forall\, \mathbf{x}_\delta \in \mathcal{C}_\delta$, the following holds: if $\lambda_{max}(R(\mathbf{x}_\delta)) < -\delta\|L\|_F$, then $R(\mathbf{x}) \prec 0$ for all $\mathbf{x} \in \mathcal{C}$.

Proof:

We prove the result by contradiction. Assume $\lambda_{max}(R(\mathbf{x}_\delta)) < -\delta\|L\|_F$ for all $\mathbf{x}_\delta \in \mathcal{C}_\delta$, and there exists $\mathbf{x} \in \mathcal{C}$ with $R(\mathbf{x}) \not\prec 0$.

Recall that $R(\mathbf{x}) = J(\mathbf{x})^T M(\mathbf{x}) + M(\mathbf{x})J(\mathbf{x}) + \frac{\partial M(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x})$. Further, note that $\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}) = \frac{\partial M(\mathbf{x})}{\partial x_1} f_1(\mathbf{x}) + \ldots + \frac{\partial M(\mathbf{x})}{\partial x_n} f_n(\mathbf{x})$. As $f_i(\mathbf{x})$, $J(\mathbf{x})$, $M(\mathbf{x})$, and $\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}}$ are Lipschitz continuous on the compact set $\mathcal{C}$, the element $R_{ij}(\mathbf{x})$ of $R(\mathbf{x})$ is also Lipschitz continuous on $\mathcal{C}$, for some constant L_{ij}. This follows from the fact that on a compact set, addition, multiplication and composition of Lipschitz-continuous functions is Lipschitz-continuous. The Lipschitz continuity of each element of $R(\mathbf{x})$ can be expressed as $|R_{ij}(\mathbf{x} + \Delta\mathbf{x}) - R_{ij}(\mathbf{x})| \leq L_{ij}|\Delta\mathbf{x}|$. From this, we can conclude that:

$$\|R(\mathbf{x} + \Delta\mathbf{x}) - R(\mathbf{x})\|_F \leq \|L\|_F \cdot |\Delta\mathbf{x}| \tag{18}$$

[5]Note that in [29], instead of the Frobenius norm, the 2-norm is used, however since $\|A\|_2 \leq \|A\|_F$, the lemma follows.

[6]$\|A\|_F = \sum_i \sum_j |A_{ij}|^2$

[7]Note that, to identify a Lipschitz constant of a given function over a specified compact domain, a Lipschitz constant can be posited and then checked by formulating an appropriate satisfiability query to an SMT solver.

By application of Lemma (2), we have:

$$|\lambda_i(R(\mathbf{x} + \Delta\mathbf{x})) - \lambda_i(R(\mathbf{x}))| \leq \|R(\mathbf{x} + \Delta\mathbf{x}) - R(\mathbf{x})\|_F, \tag{19}$$

for each i. Combining (19) and (18), we have:

$$|\lambda_i(R(\mathbf{x} + \Delta\mathbf{x})) - \lambda_i(R(\mathbf{x}))| \leq \|L\|_F \cdot |\Delta\mathbf{x}|. \tag{20}$$

Thus, $\forall\, \mathbf{x} \in \mathcal{C}$, there exists some $\mathbf{x}_\delta \in \mathcal{C}_\delta$ such that:

$$|\lambda_i((R(\mathbf{x}_\delta)) - \lambda_i(R(\mathbf{x}))| < \|L\|_F\delta, \tag{21}$$

for each i. By assumption, $R(\mathbf{x}) \not\prec 0$, which implies:

$$-\lambda_i(R(\mathbf{x})) < 0, \tag{22}$$

for some i. As $\lambda_{max}(R(\mathbf{x}_\delta)) < -\delta||L||_F$, we also have:

$$\lambda_i(R(\mathbf{x}_\delta)) < -\delta\|L\|_F, \tag{23}$$

for all i. Adding (22) and (23), we have

$$\lambda_i(R(\mathbf{x}_\delta)) - \lambda_i(R(\mathbf{x})) < -\delta\|L\|_F. \tag{24}$$

By the triangle inequality, this yields

$$|\lambda_i(R(\mathbf{x}_\delta)) - \lambda_i(R(\mathbf{x}))| > \delta\|L\|_F, \tag{25}$$

for some i. We can then see that (21) and (25) are a contradiction. ∎

To verify $R(\mathbf{x}) \prec 0$, using Theorem 3, first we pick a $\delta > 0$ randomly, and check if $\lambda_{max}(R(\mathbf{x}_\delta)) < -\delta\|L\|_F$. If that is the case, then $R(\mathbf{x}) \prec 0$. If not, we decrease δ and repeat the same procedure.

Even though this method does not offer a cure for the curse of dimensionality, it does help in cases where solvers based on decision procedures fail because of their inability to reason over complex nonpolynomial expressions.

Both verification approaches are capable of generating counterexamples when any of the queries is found to be satisfiable. Hence, if the verification phase fails, we can repeat Algorithm 1 with the counterexample trace added to the set of initial traces.

V. Example Case Studies

In this section, we demonstrate the effectiveness of our method on several examples. In each example, given a nonlinear dynamical system and a region of interest $\mathcal{C}$, we prove that $\mathcal{C}$ is a contraction region by constructing a contraction metric valid on this set and then verifying it with Mathematica® or dReal. For all of these examples, the method in [18], which is also based on simulation traces, can be used to determine forward invariant sets in the region of interest, however since this paper is not focusing on finding Lyapunov functions, for some of the examples we borrow the forward invariance results from the existing literature.

We provide a summary of our results in Table I. For each example system, we provide the degree of the polynomials in $M(\mathbf{x})$, the computation time for the contraction metric candidate, the time required for verification, and the tool used. We start by trying to construct a constant contraction metric and gradually increase the degree of the polynomials in the contraction metric until we obtain a feasible solution. Due to space restrictions we do not provide the contraction

TABLE I

Example	Degree of $M(\mathbf{x})$	Number of Traces	Time Taken (secs) Constr-uction	Verifi-cation	Verification Tool
Sec. V-A	2	1408	317	<1	Mathematica
Sec. V-B	0	704	181	3	Mathematica
Sec. V-C	2	621	276	54	dReal
Sec. V-D	2	2670	390	445	dReal
Sec. V-E	4	2828	567	33	dReal
Sec. V-F	0	752	467	7449	dReal

metrics for all of the presented examples, but make them available at [30].

We remark that except for the the first example system, all system dynamics contain either transcendental, exponential, or nonpolynomial algebraic expressions. Thus, a direct application of the technique in [17] is not possible. It could be argued that a nonlinear change of coordinates might transform the system into polynomial form, and then SoS techniques could be applied. However, the purpose of our examples is to demonstrate the automation that our technique provides, as it can be directly used on the system description as presented. Thus, we avoid the need for "polynomializing" transformations that are likely to require human ingenuity and increase state-space dimension.

In what follows, we use $\mathcal{B}^1(\mathbf{0})$ as shorthand for the set $\{\mathbf{x} \mid \mathbf{x}^T\mathbf{x} < 1\}$, *i.e.*, the unit ball with respect to the Euclidean norm, centered at the origin. Also, recall that given a function $V(\mathbf{x})$, we use $\mathcal{S}_\ell$ to denote the set $\{\mathbf{x} \mid V(\mathbf{x}) < \ell\}$. Let $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$ denote the largest sublevel set of $V(\mathbf{x})$ that lies within $\mathcal{B}^1(\mathbf{0})$, *i.e.*, $\forall \ell$ such that $\mathcal{S}_\ell \subseteq \mathcal{B}^1(\mathbf{0})$, $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0})) \supseteq \mathcal{S}_\ell$.

A. Time Reversed Van der Pol Oscillator

Consider the time-reversed Van der Pol Oscillator dynamics:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix}-x_2\\ (x_1^2+1)x_2 - x_1\end{bmatrix}. \tag{26}$$

The time-reversed Van der Pol Oscillator has an equilibrium point at the origin and its basin of attraction includes $\mathcal{B}^1(\mathbf{0})$. Therefore, we attempt to construct a contraction metric over $\mathcal{B}^1(\mathbf{0})$. As the dynamics of the Van der Pol Oscillator are polynomial, the conditions $M(\mathbf{x}) \succ 0$ and $R(\mathbf{x}) \prec 0$, for all $\mathbf{x} \in \mathcal{B}^1(\mathbf{0})$ can be verified using Mathematica. The resulting contraction metric shows that the system is incrementally stable in $\mathcal{B}^1(\mathbf{0})$.

B. Normalized Pendulum

Consider a standard pendulum system with normalized parameters:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix}x_2\\ -\sin x_1 - x_2\end{bmatrix}. \tag{27}$$

Here x_1, x_2 are the angular position and velocity respectively. Consider the Lyapunov function $V(\mathbf{x}) = \mathbf{x}^T P\mathbf{x}$ for this system, where P is as given in [18]:

$$P = \begin{bmatrix}100 & 24\\ 24 & 92\end{bmatrix}.$$

We conjecture that $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0})) = \{\mathbf{x} | \mathbf{x}^T P\mathbf{x} \le 71\}$ is a contraction region. By definition, $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$ is forward invariant. For the dynamics in (27), the contraction metric $M(\mathbf{x})$ with constant entries can be used to show that $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$ is a contraction region, and hence establish incremental stability of the system on $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$. For reference, the resulting $M(\mathbf{x})$ is:

$$\begin{bmatrix}1.7564 & 1.1046\\ 1.1046 & 2.6616\end{bmatrix}. \tag{28}$$

C. Whirling Pendulum

Next, consider the dynamics of a whirling pendulum [31]:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix}x_2\\ -\frac{k_f}{m_b}x_2 + w^2\sin(x_1)\cos(x_1) - \frac{g}{l_p}\sin(x_1)\end{bmatrix}. \tag{29}$$

Here, x_1, x_2 are the angular position and velocity respectively. We set k_f, m_b, w, l_p to 1, and g to 10. The the dynamics include transcendental terms similar to the previous example. A Lyapunov function for the whirling pendulum system is provided in [31]: $V(\mathbf{x}) = 0.33445x_2^2 + 1.4615\sin(x_1)^2 + 1.7959\cos(x_1)^2 - 6.689\cos(x_1) + 4.8931$, and we conjecture that $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0})) = \{\mathbf{x} | V(\mathbf{x}) \le 0.052985\}$ is a contraction region.

Even though the dynamics of this example are similar to the previous example, the search for a constant contraction metric was not successful, but a contraction metric with polynomials of degree 2 could be obtained. The existence of this metric proves that $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$ is a contraction region.

D. System with Saturation

Consider the following 2-dimensional system from [31]:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix}x_2\\ -\frac{x_1+x_2}{\sqrt{1+(x_1+x_2)^2}}\end{bmatrix}. \tag{30}$$

A Lyapunov function was provided in [31] as $V(\mathbf{x}) = -3.9364 + 3.9364\sqrt{(1+x_1^2)} + 0.0063889x_1^2 + 0.010088x_1x_2 + 2.0256x_2^2$, and we conjecture that $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0})) = \{\mathbf{x} \mid V(\mathbf{x}) < 1.6365\}$ is a contraction region. We find a contraction metric over $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$, where the degree of polynomials in $M(\mathbf{x})$ is 2.

E. System with Exponential Dynamics

Consider the system from [32]:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix}-x_1 + x_2 + 0.5(e^{x_1}-1)\\ -x_1 - x_2 + x_1x_2 + x_1\cos(x_1)\end{bmatrix}. \tag{31}$$

We search for a contraction metric over $\mathcal{B}^1(\mathbf{0})$. Note that, this region has been shown to be forward invariant in [32]. We found a contraction metric with second degree of polynomials in $M(\mathbf{x})$ that is valid over $\mathcal{B}^1(\mathbf{0})$, thereby establishing incremental stability over $\mathcal{B}^1(\mathbf{0})$.

F. Powertrain Control System

We consider the air/fuel ratio controller with the dynamics given in [18], [33] shifted so that the equilibrium point is at the origin. The system has 4 state variables, where 2 of these are states of the plant and the other 2 are of the controller. The dynamics contain nonpolynomial expressions such as ratios of polynomials and square root terms. For complete details, please see [18], [33]. We search for a contraction metric for the region $\mathcal{X} = \{\mathbf{x} | \mathbf{x}^T\mathbf{x} \leq 0.01^2\}$. This region corresponds to the set of initial states in [18]. A Lyapunov function $V(\mathbf{x})$ for this system was obtained using the technique in [18]. From this, we find the set $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0})) = \{\mathbf{x} \mid V(\mathbf{x}) < 0.0027\}$ inside $\mathcal{X}$ that is forward invariant, and thus a valid contraction region. We constructed and verified a constant contraction metric for this system that is valid over $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$. Note that the constructed contraction metric allows us to perform reachability analysis over the set of initial states in $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$. Hence, it can serve as a valuable tool to evaluate the performance of the air/fuel ratio controller. For example, the designer can take advantage of the safety verification tools as in [10], [11] and use the obtained contraction metric to test whether the states that start in a given subset of $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$ enter a certain unsafe region in $\mathcal{S}_{max}(\mathcal{B}^1(\mathbf{0}))$.

VI. Conclusions

We present a technique that uses simulation traces to construct contraction metrics for nonlinear dynamical systems. The main idea is to iteratively improve candidate contraction metrics using a combination of two iterative loops to refute intermediate candidates. In the inner loop, a global optimizer is used in conjunction with semidefinite programming solvers to rapidly iterate over a space of feasible contraction metrics. In the outer loop, formal verification is performed using decision procedures for nonlinear arithmetic.

We demonstrate the effectiveness of our technique on several nonpolynomial dynamical systems. We also present an example from the automotive domain that cannot be handled with existing methods, but can be effectively analyzed using our technique. In the future, we plan to improve the scalability to higher dimensions and interface this method with a safety verification tool (*e.g.*, [10], [11]).

References

[1] W. Lohmiller and J. J. E. Slotine, "On contraction analysis for nonlinear systems," *Automatica*, vol. 34, no. 6, pp. 683 – 696, 1998.

[2] J. Jouffroy, "Some ancestors of contraction analysis," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, Dec 2005, pp. 5450–5455.

[3] D. Angeli, "A Lyapunov approach to incremental stability properties," *IEEE Trans. on Automatic Control*, vol. 47, no. 3, pp. 410–421, 2002.

[4] M. Zamani and P. Tabuada, "Backstepping design for incremental stability," *IEEE Trans. on Automatic Control*, vol. 56, no. 9, pp. 2184–2189, Sept 2011.

[5] M. Zamani, N. van de Wouw, and R. Majumdar, "Backstepping controller synthesis and characterizations of incremental stability," *Systems & Control Letters*, vol. 62, no. 10, pp. 949 – 962, 2013.

[6] J. W. Simpson-Porco and F. Bullo, "Contraction theory on riemannian manifolds," *Systems & Control Letters*, vol. 65, pp. 74–80, 2014.

[7] F. Forni and R. Sepulchre, "A differential lyapunov framework for contraction analysis," *IEEE Trans. on Automatic Control*, vol. 59, no. 3, pp. 614–628, March 2014.

[8] B. S. Rüffer, N. van de Wouw, and M. Mueller, "Convergent systems vs. incremental stability," *Systems & Control Letters*, vol. 62, no. 3, pp. 277 – 285, 2013.

[9] A. A. Julius and G. J. Pappas, "Trajectory based verification using local finite-time invariance," in *Proceedings of the 12th International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 223–236.

[10] P. S. Duggirala, S. Mitra, and M. Viswanathan, "Verification of annotated models from executions," in *Proc. of Conference on Embedded Software*, 2013, pp. 26:1–26:10.

[11] Z. Huang and S. Mitra, "Proofs from simulations and modular annotations," in *17th International Conference on Hybrid Systems: Computation and Control.*, 2014.

[12] A. Girard and G. Pappas, "Approximate bisimulations for nonlinear dynamical systems," in *Proc. of IEEE Conf. on Decision and Control*, 2005, pp. 684–689.

[13] J. Kapinski, A. Donzé, F. Lerda, H. Maka, S. Wagner, and B. H. Krogh, "Control software model checking using bisimulation functions for nonlinear systems," in *Proc. of IEEE Conf. on Decision and Control*, 2008, pp. 4024–4029.

[14] W. Wang and J.-J. E. Slotine, "On partial contraction analysis for coupled nonlinear oscillators," *Biological Cybernetics*, vol. 92, no. 1, pp. 38–53, 2005.

[15] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508 – 2516, 2008.

[16] W. Lohmiller and J.-J. E. Slotine, "Nonlinear process control using contraction theory," *AIChE*, vol. 46, no. 3, pp. 588–596, 2000.

[17] E. M. Aylward, P. A. Parrilo, and J.-J. E. Slotine, "Stability and robustness analysis of nonlinear systems via contraction metrics and sos programming," *Automatica*, vol. 44, no. 8, pp. 2163–2170, Aug. 2008.

[18] J. Kapinski, J. V. Deshmukh, S. Sankaranarayanan, and N. Arechiga, "Simulation-guided lyapunov analysis for hybrid dynamical systems," in *Proc. of Hybrid Systems: Computation and Control*, 2014.

[19] U. Topcu, P. Seiler, and A. Packard, "Local stability analysis using simulations and sum-of-squares programming," *Automatica*, vol. 44, pp. 2669–2675, 2008.

[20] S. Wolfram, *The Mathematica Book*, 5th ed. Wolfram Media, Incorporated, 2003.

[21] S. Gao, S. Kong, and E. Clarke, "dreal: An smt solver for nonlinear theories over the reals," in *Proc. of Conf. on Automated Deduction*, 2013, pp. 208–214.

[22] J. Jouffroy and T. I. Fossen, "A Tutorial on Incremental Stability Analysis using Contraction Theory," *Modeling, Identification and Control*, vol. 31, no. 3, pp. 93–106, 2010.

[23] J. F. Sturm, "Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 625–653, 1999.

[24] K.-C. Toh, M. J. Todd, and R. H. Tütüncü, "Sdpt3—a matlab software package for semidefinite programming, version 1.3," *Optimization methods and software*, vol. 11, no. 1-4, pp. 545–581, 1999.

[25] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in MATLAB," in *Proc. of Computer Aided Control System Design*, 2004. [Online]. Available: http://control.ee.ethz.ch/~{}joloef/yalmip.php

[26] A. Tarski, *A decision method for elementary algebra and geometry*. Springer, 1998.

[27] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert, "Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure," *JSAT*, vol. 1, no. 3-4, pp. 209–236, 2007.

[28] J. Robinson, *The collected works of Julia Robinson*. American Mathematical Soc., 1996, vol. 6.

[29] T. Tao, *Topics in random matrix theory*. American Mathematical Soc., 2012, vol. 132.

[30] http://www.cyphylab.ee.ucla.edu/contraction_data.

[31] A. Papachristodoulou and S. Prajna, "Analysis of non-polynomial systems using the sum of squares decomposition," in *Positive Polynomials in Control*, ser. Lecture Notes in Control and Information Science, 2005, vol. 312, pp. 23–43.

[32] G. Chesi, "Estimating the domain of attraction for non-polynomial systems via lmi optimizations," *Automatica*, vol. 45, no. 6, pp. 1536–1541, Jun. 2009.

[33] X. Jin, J. V. Deshmukh, J. Kapinski, K. Ueda, and K. Butts, "Powertrain control verification benchmark," in *Proc. of Hybrid systems: computation and control*, 2014, pp. 253–262.