Polynomial Level-Set Method for Polynomial System Reachable Set Estimation

Ta-Chung Wang, *Member, IEEE*, Sanjay Lall, *Senior Member, IEEE*, and Matthew West

***Abstract*—In this paper, we present a polynomial level-set method for *advecting* a semi-algebraic set for polynomial systems. This method uses the sub-level representation of sets. The problem of flowing these sets under the *advection map* of a dynamical system is converted to a semi-definite program, which is then used to compute the coefficients of the polynomials. The method presented in this paper does not require either the sets being positively invariant or star-shaped. Hence, the proposed algorithm can describe the behavior of system states both inside and outside the domain of attraction and can also be used to describe more complex shapes of sets. We further address the related problems of constraining the degree of the polynomials. Various numerical examples are presented to show the effectiveness of *advection* approach.**

***Index Terms*—Algebraic/geometric methods, level-set methods, nonlinear systems, semi-definite programming.**

I. Introduction

FOR linear systems, several crucial characteristics such as stability, controllability and observability have been well studied and analyzed. However, for nonlinear systems, these properties are quite difficult to analyze. Several phenomena such as multiple isolated equilibria and limit cycles occur in the presence of nonlinearity, and cannot be analyzed or described by methods used for analyzing linear systems [1].

One of the key problems in nonlinear-systems analysis is the analysis of the reachable set. For a given system and a set of initial system states of interest, the analysis of reachable sets computes all the system states starting from the given initial set. The study of reachable sets enables us to verify whether the system is stable or unstable for a given initial region, and can also be used for safety verification purposes by identifying whether the system state in the initial set will reach an unsafe region [2]–[4]. The most intuitive way of reachability computation is numerical integration. To find all the reachable states, we can exhaustively pick all the points inside the initial set and integrate the trajectories of the system; then we know which system state is reachable. Instead of integrating all the possible initial states, this integration approach can also be done in a set-based manner. That is, we can integrate the initial set and study how it evolves with time. Several techniques such as polytope based approach [3], [5], ellipsoid based approach [6], and level-set based approach [2] have been proposed, and can be categorized on the basis of how they represent the sets. The choice of representation of sets is closely related to the required computational complexity and accuracy. Our method can be viewed as a combination of the above representation methods. We represent a set by using a sub-level-set of polynomials. This enables us to represent a complex set by storing the coefficients of the representing polynomial. Moreover, we flow the level-set by computing the coefficients of the polynomials rather than integrating the mesh points of level-sets [2]. Therefore, the proposed method can *advect* complex shaped sets while maintaining suitable computation burden.

Estimation of the Domain-of-Attraction (DoA) of an equilibrium point is a good example demonstrating the capability of the proposed algorithm. The DoA is the region in which the system exhibits stable behavior around the associated equilibrium point. That is, for any system state inside the DoA, its location converges to the equilibrium point as time increases. This DoA region is often difficult to find and represent computationally. The usual mathematical tools used for the analysis of the DoA are methods based on the Lyapunov stability theorem. The Lyapunov stability theorem studies the behavior of positively invariant sets. Methods based on Lyapunov's theorem give us a sufficient condition for local stability although it is often difficult to find a Lyapunov function that can be used as a certificate for the whole DoA. Several prior approaches have used quadratic functions, for example, [7], [8], and [9]. In particular, the approach of Chesi [9] makes use of semi-definite programming to find a quadratic function whose sub-level-set is a good inner approximation of the DoA. For a system in which the region is complicated, an ellipsoid may not provide a good approximation, and the above methods leave a large unexplored region within the DoA. In [10], parameter-dependent quadratic Lyapunov functions were used to improve the flexibility of the chosen Lyapunov functions. And later, Chesi [11] used the union of several estimated DoAs obtained by parameter-dependent quadratic Lyapunov functions to improve the estimation result. Similar approaches were later applied to non-polynomial systems [12] and uncertain systems [13]. Tan [14] proposed a composite polynomial Lyapunov approach which estimate the DoA through a composition of several polynomial Lyapunov functions. This composite approach generates a very precise estimation of the DoA by solving an optimization over a non-convex solution space.

With recent developments in algebra and Sum-of-Squares (SOS) techniques, it is now possible to solve for a Lyapunov

Manuscript received August 07, 2011; revised March 05, 2012; accepted September 09, 2012. Date of publication May 17, 2013; date of current version September 18, 2013. Recommended by Associate Editor F. Dabbene.

T.-C. Wang is with the Institute of Civil Aviation, National Cheng-Kung University, Tainan 708, Taiwan (e-mail: tachung@mail.ncku.edu.tw).

S. Lall is with Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305 USA.

M. West is with Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2013.2263916

function with a more general polynomial form [15], [16]. Positive definiteness properties are replaced by SOS constraints that can be efficiently solved using convex optimization. The SOSTOOLS [17] toolbox for MATLAB has been developed as an easy computational tool to solve problems that utilize the SOS techniques. This approach has also allowed finding a Lyapunov function within some specified semi-algebraic region [18], [19]. However, while this provides a method to certify an inner approximation to the DoA, it does not immediately provide a way to find it.

One other approach for estimating the DoA is to flow a positively invariant set around the equilibrium point backward in time. This approach *advects* a small positively invariant set around the equilibrium point backward in time. The approach is similar in spirit to the level-set methods [20] that have been used for computation of reachable sets [2], [21]. The key distinction is that most level-set methods represent the function on a mesh; we represent the function as a polynomial. Consequently, the computational requirements may grow more slowly with dimension, if one may fix the required degrees of the polynomials *a priori*. By contrast, a mesh-based method has computational costs that grow exponentially with dimension. The DoA can be estimated more precisely using the *advection* idea [22]. Better estimation of DoA gives us a better insight into the system behavior around the equilibrium point, and can be used for refining the system controller's parameters [23].

In this paper, we present an approach using 0-sub-level-set of polynomials to represent a set of system states, and use Semi-Definite Programming (SDP) to perform the computation of *advecting* sets. To apply the proposed algorithm, the underlying system needs to be a polynomial system. The algorithm is iterative, and proceeds by *advecting* the sub-level-set of the polynomial under the flow map of polynomial systems. In our method, the polynomial used to represent sets does not have to be quadratic or SOS. Therefore, the proposed method can represent more complicated sets than existing methods [7], [8], [10]. Unlike the method used for DoA estimations in our previous work [23], the method presented in this paper is a more generic algorithm that does not require either the sets being positively invariant or star-shaped. The proposed algorithm can describe the behavior of system states both inside and outside the DoA, and can also be used to describe more complex shapes of sets. Several numerical examples computed using the SOSCODE [24] toolbox for MATLAB is presented in this paper.

The structure of this paper is as follows. In Section II, we present the notations used in this paper and provide a simple introduction to the SOS techniques. The detailed analysis of the proposed *advection algorithm* as well as two of its variants is stated in Section III. Section IV contains several numerical examples as well as comparison of accuracies of the proposed algorithms. Discussions of the computation time of the proposed algorithms are also provided. And we conclude the paper in Section V. Several preliminary results can be found in the Appendix section.

II. Sum-of-Squares Representations of Level-Sets

We use $\mathbb{R}[x]$ to represent the ring of polynomials in x with real coefficients. A polynomial $f \in \mathbb{R}[x]$ is called Positive Semi-Definite (PSD) if $f(x) \geq 0$, for all $x \in \mathbb{R}^n$. A polynomial f is called a *Sum-of-Squares* (SOS) if there exist polynomials $g_1, \ldots, g_s \in \mathbb{R}[x]$ such that $f = g_1^2 + g_2^2 + \cdots + g_s^2$. Clearly, if f is SOS, then f is PSD. It is also well-known that the converse is not true. We use Σ^2 to denote the set of all SOS polynomials in $\mathbb{R}[x]$. Also, we use $\mathbb{R}_d[x]$ to denote the set of degree d polynomials in x with real coefficients, and use Σ_d^2 to denote the set of all SOS polynomials in $\mathbb{R}_d[x]$. We will use $\mathbb{R}_+$ to represent the set of nonnegative real numbers and use $\mathbb{Z}_+$ to represent the set of positive integers. For $z \in \mathbb{R}^n$ and $r > 0$, we denote the open ball by $\mathcal{B}_r(z)$ given by

$$\mathcal{B}_r(z) = \{x \in \mathbb{R}^n | \, \|x - z\| < r\}.$$

Avoiding confusion, we use $\mathcal{B}_r$ to represent the open ball with radius r centered at the origin.

Suppose $g : \mathbb{R}^n \to \mathbb{R}$ is C^1. Define the 0-*sub-level-set* of g to be $\mathcal{C}(g) \subset \mathbb{R}^n$ given by $\mathcal{C}(g) = \{x \in \mathbb{R}^n | g(x) \leq 0\}$. The intersection of $\mathcal{C}(g)$ with another set M will be denoted as $\mathcal{C}_M(g)$. Further, define the *variety* of g by $\mathcal{V}(g) \subset \mathbb{R}^n$ given by $\mathcal{V}(g) = \{x \in \mathbb{R}^n | g(x) = 0\}$. The boundary of $\mathcal{C}(g)$ is denoted by $\partial\mathcal{C}(g)$. Then we have $\mathcal{V}(g) \supset \partial\mathcal{C}(g)$, and both $\mathcal{V}(g)$ and $\mathcal{C}(g)$ are closed when g is a continuous function.

Lemma 1 (From [15]): Given $p, q \in \mathbb{R}[x]$, suppose there exist $s_0, s_1 \in \Sigma^2$ such that

$$s_0 - s_1 q + p = 0 \qquad \text{for all } x \in \mathbb{R}^n. \tag{1}$$

Then, $\mathcal{C}(q) \subset \mathcal{C}(p)$. Further, given q and the degree bound of p, s_0, and s_1, the set of coefficients of p, s_0 and s_1 satisfying (1) is the feasible set of a semi-definite program.

Proof: See, for example, [18] or [15]. □

The representation shown in Lemma 1 is one of the simplest cases of Putinar's distinguished representation theorem [25]. The following result is similar. Given $q \in \mathbb{R}[x]$, if there exists $s_0, s_1 \in \Sigma^2$ and $\epsilon > 0$ such that

$$s_0 + s_1 q - p + \epsilon = 0, \tag{2}$$

then $\mathcal{C}(p) \subset \mathcal{C}(q)$.

Note that, usually, we know q and want to find p such that $\mathcal{C}(p)$ and $\mathcal{C}(q)$ approximately represent the same set with some other constraints on p, such as a pre-defined degree or passing through some pre-specified points. We use the above relationships to construct such constraints. This technique is frequently used in the proposed level-set algorithm.

In this paper, we use sub-level-sets of polynomial functions to represent subsets in $\mathbb{R}^n$. The capability of this kind of representation is that by using a small storage space for storing the coefficients of the polynomials, we can represent very complicated sets. Moreover, we can use the union and/or intersection of semi-algebraic sets to further extend the capability of this kind of representation. The following example demonstrates how to utilize the result of Lemma 1 to find a polynomial representation of the intersection of sets.

Example 1: Suppose we would like to find a polynomial $f \in \mathbb{R}[x]$ such that $\mathcal{C}(f) = \{x \in \mathbb{R} | x+1 \geq 0, x-1 \leq 0\}$. Let $f = f_0 + f_1 x + f_2 x^2$, $s_5, s_6, s_7 \in \mathbb{R}$ and

$$\begin{aligned} s_1 &= z^T O z & s_2 &= z^T P z \\ s_3 &= z^T Q z & s_4 &= z^T R z, \end{aligned}$$

where $z = [x\ 1]^T$ and $O, P, Q, R \in \mathbb{R}^{2\times 2}$. By examining the relationships between f and $x+1$, and f and $x-1$, we would like $f > 0$ if either $x+1 < 0$ or $x-1 > 0$ and $f < 0$ when $x+1 > 0$ and $x-1 < 0$. Using Lemma 1,we have

$$\begin{aligned} s_5 - s_1(x-1) + s_2(x+1) + f &= 0 \\ s_3 + s_6(x-1) - f &= 0 \\ s_4 - s_7(x+1) - f &= 0. \end{aligned}$$

By comparing the coefficients, we have the following feasibility problem; we need to solve for $O, P, Q, R \succeq 0$, $s_5, s_6, s_7 \geq 0$ such that

$$\begin{aligned} f_0 + s_5 + o_{22} + p_{22} &= 0 \\ f_1 - o_{22} + o_{12} + o_{21} + p_{22} + p_{12} + p_{21} &= 0 \\ f_2 - o_{12} - o_{21} + o_{11} + p_{12} + p_{21} + p_{11} &= 0 \\ p_{11} - o_{11} &= 0 \\ q_{22} - s_6 - f_0 &= 0 \\ q_{12} + q_{21} + s_6 - f_1 &= 0 \\ q_{11} - f_2 &= 0 \\ r_{22} - s_7 - f_0 &= 0 \\ r_{12} + r_{21} - s_7 &= 0 \\ r_{11} - f_2 &= 0, \end{aligned}$$

where we use the lower case letters with subscripts to denote the elements of the matrices. We get the following solution

$$\begin{aligned} f &= x^2 - 1 & s_1 &= \frac{1}{2}x^2 + x + \frac{1}{2} \\ s_2 &= \frac{1}{2}x^2 - x + \frac{1}{2} & s_3 &= x^2 - 2x + 1 \\ s_4 &= x^2 + 2x + 1 & s_5 &= 0 \\ s_6 &= 2 & s_7 &= 2, \end{aligned}$$

and $\mathcal{C}(f) = \{x \in \mathbb{R} | x+1 \geq 0, x-1 \leq 0\}$, as expected. This example shows how we use SOS techniques to find the intersection of several semi-algebraic sets. This technique is also one of the crucial concepts of the proposed algorithm.

One other method to find the intersection/union of semi-algebraic sets is to use R-composition of R-functions [26], [27]. R-compositions have specific form of generating the not/intersection/union of R-functions and has been applied in nonlinear system analysis [28]. However, the generated R-functions may be of non-integer degree and hence will not be suitable for further operations using SOS techniques.

III. Set Advection

In this paper, we consider the following autonomous system

$$\dot{x}(t) = f(x(t)), \tag{3}$$

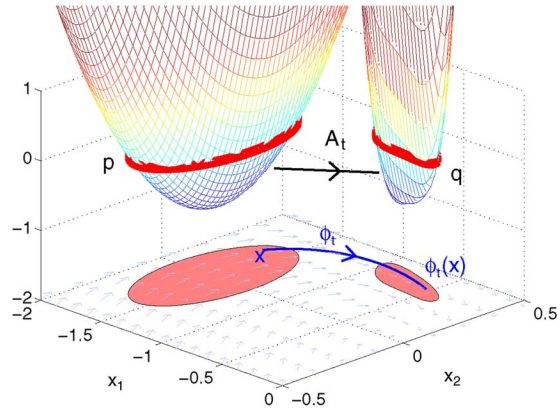

Fig. 1. Advection operator A_t.

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is locally Lipschitz. The basic local existence and uniqueness theorem [29] states that given an open subset $U \in \mathbb{R}^n$, there exists $c \in \mathbb{R}_+$ such that the autonomous system (3) has a unique solution for any $z \in U$ in the compact time interval $[-c, c]$.

We define the *flow map* $\phi_t(z) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ with notation $\phi_t(z)$ where $(z,t) \in \mathbb{R}^n \times \mathbb{R}$ as the unique solution of

$$\begin{aligned} &\frac{\partial \phi_t(z)}{\partial t} = f(\phi_t(z)) \quad \text{for } t \in [-c, c],\ c(z) \in \mathbb{R}_+,\ z \in \mathbb{R}^n \\ &\phi_0(z) = z. \end{aligned}$$

For any $t \in \mathbb{R}$ such that $\phi_t(z)$ exists, the map $z \mapsto \phi_t(z)$ is continuous, invertible and has a continuous inverse; that is, it is a topological homeomorphism on $\mathbb{R}^n$ [1].

Given $t \in \mathbb{R}$, we define the time t *advection operator* $A_t : C(\mathbb{R}^n, \mathbb{R}) \to C(\mathbb{R}^n, \mathbb{R})$ by

$$q = A_t p \quad \text{if} \quad q(x) = p(\phi_{-t}(x)) \text{ for all } x \in \mathbb{R}^n,$$

where $C(X, Y)$ is the set of functions mapping from X to Y. The map A_t is also called the Liouville operator associated with f. A very important property is that it is linear. Fig. 1 shows the concept of the advection operator. We relate the advection operator to the advection of sets in the following remark.

Remark 1: Suppose g_1, g_2 are functions mapping $\mathbb{R}^n$ to $\mathbb{R}$. If $g_2 = A_t g_1$, then $\mathcal{C}(g_2) = \phi_t(\mathcal{C}(g_1))$.

A. Time-Stepping

Performing advection, an approximation to the flow map ϕ_h with time step h should be used. Many such approximations are possible, and are provided by the theory of numerical integration. Here we present two simple approximations. The first-order Taylor approximation to $q = A_h p$ is the map $B_h : C(\mathbb{R}^n, \mathbb{R}) \to C(\mathbb{R}^n, \mathbb{R})$ given by

$$q = B_h p \qquad \text{if } q(x) = p(x) - hDp(x)f(x),$$

where the derivative $Dp(x)$ is a linear map $Dp(x) : \mathbb{R}^n \to \mathbb{R}^n$ at each point x. An alternative approximation is C_h, given by

$$q = C_h p \qquad \text{if } q(x) = p(x - hf(x)).$$

Both the approximations have the following properties: they are both linear maps, and if p and f are polynomials, then so is q. In this paper, we concentrate on B_h, since typically $B_h p$ is a polynomial whose degree is less than that of $C_h p$.

Based on the required accuracy of advection, we might use a higher order Taylor approximation. However, depending on the system dynamics, this will lead to the requirement of using higher degree polynomials in the SOS constraints. The relationship between accuracy and degree of polynomials will be further investigated in future work.

Given an autonomous polynomial system $\dot{x} = f(x)$ and a polynomial p, we are now able to approximately represent $\phi_h(\mathcal{C}(p))$ by using B_h. However, $B_h p$ will usually yield a polynomial with a degree higher than p. Therefore, we should not directly use B_h to do the advection. To deal with this problem, we will use the approximated fitting of the advected set.

Here, we introduce the approximation tolerance parameter $\delta \in \mathbb{R}_+$. Given an autonomous system f and a semi-algebraic set $\mathcal{C}(g)$, we would like to find a polynomial p such that $\mathcal{C}(g) \subset \mathcal{C}(B_{-h}p) \subset \mathcal{C}(g-\delta)$. To analyze the difference between sets, we define

$$\xi(A,B) = \max_{x\in A}\min_{y\in B}\|x-y\|$$

and use

$$H(A,B) = \max\left(\xi(A,B),\xi(B,A)\right)$$

to represent the *Hausdorff metric* [30] of sets A and B. For simplification, we use $H_M(A,B)$ to denote $H(A\cap M, B\cap M)$.

The error of the Taylor series approximated advection sub-level-set is shown in the following theorem. The related propositions and lemmas can be found in the Appendix section.

Theorem 1: Consider a compact set M in which the autonomous system has unique solutions defined for all $x \in M$, $t \in [-h^*, h^*]$ for some $h^* > 0$. For every polynomial p such that $\phi_t(\mathcal{C}(p)) \subset M$ for $t \in [-h^*, h^*]$, there exist $0 < h < h^*$, $q \in \mathbb{R}[x]$, $\eta > 0$, and $\gamma > 0$ such that

$$H_M(\mathcal{C}(p),\mathcal{C}(B_{-h}q)) \le e^{Fh}\gamma + H_M(\mathcal{C}(B_{-h}q-\eta),\mathcal{C}(B_{-h}q+\eta)).$$

where F is a Lipschitz constant of $f(x)$ in M^+, and $M \subset M^+$ such that $\phi_t M \subset M^+$ for $t \in [-h^*, h^*]$. Further suppose $\|\nabla(B_{-h}q(x))\|$ is nonzero for all $x \in (\mathcal{C}(B_{-h}q-\zeta)\backslash\mathcal{C}(B_{-h}q+\zeta))$ for some $\zeta > 0$. Then, for any $\epsilon > 0$, there exist $0 < h < h^*$, $q \in \mathbb{R}[x]$, such that

$$H_M\left(\mathcal{C}(p),\mathcal{C}(B_{-h}q)\right) \le \epsilon.$$

Proof: From Lemma 2, Lemma 3 (both in Appendix A) and the triangle inequality, there exist $\gamma > 0$ and $\eta > 0$ such that

$$\begin{aligned} &H_M\left(\mathcal{C}(p),\mathcal{C}(B_{-h}q)\right)\\ &\quad\le H_M\left(\mathcal{C}(p),\mathcal{C}(A_{-h}q)\right)\\ &\qquad + H_M\left(\mathcal{C}(A_{-h}q),\mathcal{C}(B_{-h}q)\right)\\ &\quad\le e^{Fh}\gamma + H_M\left(\mathcal{C}(B_{-h}q-\eta),\mathcal{C}(B_{-h}q+\eta)\right). \end{aligned}$$

From the discussion of Lemma 2, γ can be minimized by using a high degree polynomial, q. Hence, the first term on the right and side can be made arbitrarily small. Further, suppose $\|\nabla(B_{-h}q(x))\|$ is nonzero for all $x \in (\mathcal{C}(B_{-h}q-\zeta)\setminus\mathcal{C}(B_{-h}q+\zeta))$ for some $\zeta > 0$. Then, from Lemma 3, the second term on the right and side can be made arbitrarily small by choosing a small gap h. Therefore, for any $\epsilon > 0$, there exist $0 < h < h^*$, $q \in \mathbb{R}[x]$, such that

$$H_M\left(\mathcal{C}(p),\mathcal{C}(B_{-h}q)\right) \le \epsilon.$$

□

Theorem 1 indicates that it is possible to find a polynomial q whose Taylor approximated backward advected set $\mathcal{C}_M(B_{-h}q)$ can fit another sub-level-set $\mathcal{C}_M(p)$ to the required accuracy by reducing the step size. It should be noted that the approximation error is upper bounded by $\epsilon > \gamma e^{Fh}$. This means that for a small desired precision ϵ, a much smaller h has to be used.

The main algorithm is designed to utilize the results of Theorem 1. The objective is to convert the relationship of $H_M(\mathcal{C}(p),\mathcal{C}(A_{-h}q)) \le \epsilon$ into algebraic constraints. Notice that $A_{-h}q$ is usually not a polynomial. Therefore, $B_{-h}q$ will be used. From Proposition 1 (in Appendix A), given a compact M, there exists $\epsilon > 0$ such that $|A_{-h}q - B_{-h}q| \le \epsilon$. Hence, we have

$$\mathcal{C}(B_{-h}q+\epsilon) \subset \mathcal{C}(A_{-h}q) \subset \mathcal{C}(B_{-h}q-\epsilon).$$

We will use this result to bound the Hausdorff distance between $\mathcal{C}(A_{-h}q)$ and $\mathcal{C}(p)$. This gives us the main algorithm.

Before presenting the main algorithm, we introduce another parameter, $\beta > 0$, which serves two purposes. First, we would like β to be chosen such that $\mathcal{C}(g-\beta)$ contains all the system trajectories starting from $\mathcal{C}(p)$ for $t \in [-h, 0]$. Using the result of Proposition 1, the compact set $\mathcal{C}(g-\beta)$ can then be used to estimate the truncation error of $B_{-h}p$. Second, we can use β to ensure that $\mathcal{C}(p)$ has no spurious parts outside $\mathcal{C}(g-\beta)$ so that the truncation error outside $\mathcal{C}(g-\beta)$ will not cause us any trouble.

From the distinguished representation theorem [25], in order to ensure the existence of the solutions, we will need a bounding set so that the associated constraints on the solution p are specified in a compact set. Therefore, we introduce a compact set M into our advection algorithm. The compact set M is also used to isolate the region in which we are interested, and to help remove the erroneous components of $\mathcal{C}(p)$ due to truncation errors. To achieve this, M should contain all the system trajectories starting from $\mathcal{C}(g-\beta)$ for $t \in [-h, 0]$. To do so, we usually choose M as a much bigger set than $\mathcal{C}(g-\beta)$. The relationship between all the above parameters can be clearly seen in Fig. 2.

B. Set Advection Algorithm

Consider a compact semi-algebraic set $\mathcal{C}(g)$ and a polynomial autonomous system $\dot{x} = f(x)$ with the flow map, ϕ. Select the approximation tolerance $\delta \in \mathbb{R}_+$. Also select the time step $h \in \mathbb{R}_+$, the parameter $\beta > \delta$ and a compact semi-algebraic set, $M = \{x \in \mathbb{R}^n | q(x) \le 0\}$, such that the relationships of M, $\mathcal{C}(g)$, $\mathcal{C}(g-\beta)$, and $\mathcal{C}(g-\delta)$ have the following characteristics:

- M contains only the component of $\mathcal{C}(g-\beta)$ which contains S;
- $x \in \mathcal{C}_M(g-\beta)$ for all x such that $x - hf(x) \in \mathcal{C}_M(g-\delta)$;

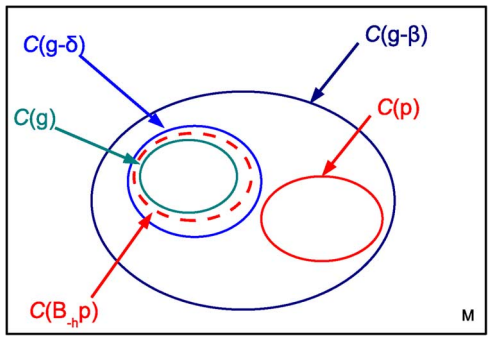

Fig. 2. Level-set algorithm parameters.

- $\phi_t(x) \in \mathcal{C}_M(g-\beta)$ for all x such that $x - hf(x) \in \mathcal{C}_M(g-\delta)$ for $t \in [-h, 0]$.
- $\phi_t(\mathcal{C}_M(g-\beta)) \subseteq M$ for all $t \in [-h, 0]$.

Please refer to Fig. 2 for graphical representations of these related sets. We would like to find the semi-algebraic set $\mathcal{C}(p)$ whose backward advected set $\mathcal{C}(B_{-h}p)$ is approximately $\mathcal{C}(g)$. And we use $\mathcal{C}(g)$ and $\mathcal{C}(g-\delta)$ to define how precise we want for the approximation. The set M is used to capture all the system trajectories starting from $\mathcal{C}(g-\beta)$. This can be easily done by using a ball surrounding the sets of interest. $\mathcal{C}(g-\beta)$ is used for truncation error estimation and has to contain the system trajectory starting from $\mathcal{C}(g-\delta)$. Since we are using first order Taylor approximation, $\mathcal{C}(g-\beta)$ also needs to contain these first-order approximated trajectories. If we were to use higher order Taylor approximations, these conditions would have to be modified accordingly. Since we will be using a small time-step approximation of the advected sets, the above conditions are not too hard to satisfy.

Then, pick small positive numbers κ and ζ and the degrees $d, d_r \in \mathbb{Z}_+$ for the polynomials. Solve for $p \in \mathbb{R}_d[x]$, $s_1, \ldots, s_{18} \in \Sigma^2_{d_r}$ and $\eta \in \mathbb{R}_+$ from the following optimization problem

$$\begin{aligned}
&\min\ \eta \\
&\text{s.t. } s_1 - s_2 g - s_3 q + B_{-h}p + \eta = 0 \\
&\quad s_4 + s_5(g-\delta) - s_6 q - B_{-h}p + \eta + \kappa = 0 \\
&\quad s_7 + s_8(g-\beta) - s_9 q - p + \zeta = 0 \\
&\quad s_{10} + s_{11} g - s_{12}(g-\delta) - \langle D(B_{-h}p), Dg\rangle = 0 \\
&\quad s_{13} - s_{14}(g-\beta) - s_{15} q + L_f^2 p \frac{h^2}{2} - \eta = 0 \\
&\quad s_{16} - s_{17}(g-\beta) - s_{18} q - L_f^2 p \frac{h^2}{2} - \eta = 0. \qquad (4)
\end{aligned}$$

The last two constraints in (4) show that

$$\left\| L_{f^2} p(x) \frac{h^2}{2} \right\| \leq \eta \text{ for all } x \in \mathcal{C}_M(g-\beta).$$

From the selection of M, β, δ, h and from Proposition 1, the truncation error

$$\|A_{-h}p(x) - B_{-h}p(x)\| \leq \eta \qquad (5)$$

for all x such that $x - hf(x) \in \mathcal{C}_M(g-\delta)$. The third constraint shows that p is positive for $x \in M \setminus \mathcal{C}(g-\beta)$. Therefore, the only possible parts of $\mathcal{C}(p)$ come from either inside of $\mathcal{C}(g-\beta)$ or outside of M. Thus, we have $\phi_{-h}(\mathcal{C}(p) \cap \mathcal{C}(g-\beta)) = \phi_{-h}(\mathcal{C}_M(p))$. The components of $\mathcal{C}(A_{-h}p)$ generated from $C(p)$ outside M will not be considered since we are concerned only with the sub-level-sets in M. From (5) and the selection of M, h, and the first two constraints in (4), we have

$$\begin{aligned}
\mathcal{C}_M(g) &\subset \mathcal{C}_M(B_{-h}p + \eta) \\
&\subset \phi_{-h}\left(\mathcal{C}_M(p)\right) \\
&\subset \mathcal{C}_M(B_{-h}p - \eta) \\
&\subset \mathcal{C}_M(g-\delta).
\end{aligned}$$

Therefore, the above algorithm tries to solve for a polynomial p such that

$$H\left(\mathcal{C}_M(g), \phi_{-h}\left(\mathcal{C}_M(p)\right)\right) \leq H_M\left(\mathcal{C}(g), \mathcal{C}(g-\delta)\right).$$

The fourth constraint is used to ensure that the gradient of $B_{-h}p$ is nonzero in $\mathcal{C}(g-\delta) \setminus \mathcal{C}_M(g)$. Then there are no spurious components or disconnected sub-level-sets of $\mathcal{C}(B_{-h}p)$ in the region of $\mathcal{C}_M(g-\delta) \setminus \mathcal{C}_M(g)$.

To apply algorithm (4), many parameters have to be chosen. One parameter that must be chosen is d, degree of the desired approximations; how large a d is necessary depends on the complexity of the system and its advected sets. The cost of computation grows polynomially with d; hence, typically, d is chosen as small as possible. Once the degree has been chosen, the map B_{-h} is a matrix, operating on the coefficients of the polynomials. The selection of d_r is usually based on the complexity of the system as well as the number of d. Since Lemma 1 does not tell us the degree of polynomials used in the representations, we usually will start by choosing d_r as an even number slightly bigger than d and increase d_r when necessary.

Another important parameter is δ, which we think of as follows. The above algorithm finds a degree d polynomial p such that, $\phi_{-h}\mathcal{C}(p) \subset \mathcal{C}(g-\delta)$, and $\phi_{-h}\mathcal{C}(p) \supset \mathcal{C}(g)$. Hence, the parameter δ may be thought of as a tolerance that allows for the constraint that p is required to have with degree d or less. We use the result in Proposition 3 (in Appendix A) to find the appropriate δ for the desired tolerance.

One other important parameter is $\beta > 0$. From the last two constraints, $\mathcal{C}(g-\beta)$ is used for estimating the truncation error. If β is chosen too large, the estimation of truncation error may be too conservative and it would be difficult to get a solution. β also appears in the third constraint, which ensures that $\mathcal{C}(p)$ does not have spurious parts outside a certain region away from $\mathcal{C}(g)$. Based on the above discussion, we chose β so that $\mathcal{C}(g-\beta)$ barely covers all the system trajectories starting from $\mathcal{C}(g)$ for $t \in [0, h]$. The value of β can be estimated from the upper bound of $\|\dot{g}\|$ along $\mathcal{V}(g)$, by using SOS techniques [17]. Without this constraint, we are likely to get a result with multiple disconnected sub-level-sets.

The last two parameters are κ and ζ. The use of these two parameters comes from (2). We can simply choose two small positive numbers for these two parameters.

Using the result of Theorem 1, it is possible to find an h such that algorithm (4) has a solution. The following theorem shows that the solution $\mathcal{C}_M(p)$ gives an approximated solution of $\mathcal{C}_M(A_h g)$.

Theorem 2: Consider a compact semi-algebraic set $\mathcal{C}(g)$ and a polynomial autonomous system $\dot{x} = f(x)$ with the flow map, ϕ. Then by solving (4), the Hausdorff distance between the true advected set and the approximated one grows exponentially with respect to the time-step h.

Proof: Let F be the Lipschitz constant of f in M. Using *Gronwall-Bellman inequality* [1] and follow similar arguments to the proof of Lemma 2, we have,

$$H\left(\phi_h\left(\mathcal{C}_M(g)\right), \mathcal{C}_M(p)\right) \leq e^{Fh} H\left(\mathcal{C}_M(g), \phi_{-h}\left(\mathcal{C}_M(p)\right)\right) \tag{6}$$

The advection algorithm (4) solves for p such that

$$H\left(\mathcal{C}_M(g), \phi_{-h}\left(\mathcal{C}_M(p)\right)\right) \leq H_M\left(\mathcal{C}(g), \mathcal{C}(g-\delta)\right). \tag{7}$$

Therefore,

$$H\left(\phi_h\left(\mathcal{C}_M(g)\right), \mathcal{C}_M(p)\right) \leq e^{Fh} H_M\left(\mathcal{C}(g), \mathcal{C}(g-\delta)\right). \tag{8}$$

Therefore, the Hausdorff distance between the true advected set and the approximated one grows exponentially with respect to the time-step h. □

Theorem 2 shows that by solving (4), we can get a polynomial p whose advected sub-level-set, $\phi_{-h}(\mathcal{C}_M(p))$, is approximately $\mathcal{C}_M(g)$. The error comes from two different parts. The first part is from the error due to η and the second is from the error caused by δ. The optimization problem tries to minimize the truncation error, and then augment $B_{-h}p$ with the bound of the truncation error to enforce

$$\mathcal{C}_M(g) \subset \phi_{-h}\left(\mathcal{C}_M(p)\right) \subset \mathcal{C}_M(g-\delta).$$

The error can be reduced by using smaller values of δ. However, we may need to use higher degree polynomials or higher order Taylor approximation to achieve the desired result.

Although the selection of M, β and h seems complicated, the difficulty can be greatly reduced by using a small time step h and picking a large semi-algebraic set M. Picking smaller time steps also helps reducing the growth of the approximation error. The relationship between time step size and growth of approximation error is clearly shown in the result of Theorem 2.

We would like to generate a sequence of semi-algebraic sets $S_i = \{x \in \mathbb{R}^n | g_i(x) \leq 0\}$ such that $S_{i+1} \cap M \approx \phi_h S_i \cap M$. We use (4) repeatedly by letting $g = g_{i-1}$, $p = g_i$ for each iteration to generate such a sequence. The accumulated error is analyzed in the following corollary. Here we will consider the positive time step case. The case with negative time steps can be derived similarly.

Corollary 1: Consider an initial compact semi-algebraic set $\mathcal{C}(g)$ and a polynomial autonomous system $\dot{x} = f(x)$ with flow map ϕ. Pick $q_0 \in \mathbb{R}[x]$ such that $S_0 \subset \mathcal{C}(q_0)$. Let $M = \mathcal{C}(q)$ be a bounding compact set containing the sub-level-sets of interest. Then, for each step i, solve for g_{i+1} such that $\mathcal{C}_M(A_{-h}g_{i+1}) \approx \mathcal{C}_M g_i$ by solving (4).

Let $g_1, \ldots, g_m$ be the solutions of (4) after each iteration, and let $T = \sum_{i=1}^{m} h_i$. Also let $h_{\max} = \max_{i=1,\ldots,m} h_i$. Then $H(\phi_T(\mathcal{C}_M(g_0)), \mathcal{C}_M(g_m)) \to 0$ as $h_{\max} \to 0$.

Proof: For simplicity, let

$$\gamma_i = H_M\left(\mathcal{C}(g_{i-1}), \mathcal{C}(g_{i-1}-\delta_i)\right).$$

Also, let F_i be a Lipschitz constant of f in $\mathcal{C}_M(g_{i-1}-\beta_i)$ and let $F_{\max} = \max_{i=1,\ldots,m} F_i$. From the result of Theorem 2, we have

$$H_M\left(\phi_{h_1}\left(\mathcal{C}(g_0)\right), \mathcal{C}(g_1)\right) \leq e^{F_{\max} h_{\max}} \gamma_1.$$

Similarly, we have

$$\begin{aligned}
&H_M\left(\phi_{h_1+h_2}\left(\mathcal{C}(g_0)\right), \mathcal{C}(g_2)\right) \\
&\quad \leq H\left(\phi_{h_1+h_2}\left(\mathcal{C}(g_0)\right), \phi_{h_2}\left(\mathcal{C}(g_1)\right)\right) \\
&\qquad + H_M\left(\phi_{h_2}\left(\mathcal{C}(g_1)\right), \mathcal{C}(g_2)\right) \\
&\quad \leq e^{F_2 h_2} H_M\left(\phi_{h_1}\left(\mathcal{C}(g_0)\right), \mathcal{C}(g_1)\right) \\
&\qquad + e^{F_2 h_2} H_M\left(\mathcal{C}(g_1), \mathcal{C}(g_1-\delta_2)\right) \\
&\quad \leq e^{(F_1 h_1 + F_2 h_2)} \gamma_1 + e^{F_2 h_2} \gamma_2 \\
&\quad \leq e^{F_{\max} 2 h_{\max}} \gamma_1 + e^{F_{\max} h_{\max}} \gamma_2.
\end{aligned}$$

By repeating the above process, we have

$$H_M\left(\phi_T\left(\mathcal{C}(g_0) \cap\right), \mathcal{C}(g_m)\right) \leq \sum_{i=1}^{m} e^{F_{\max}(m-i+1) h_{\max}} \gamma_i.$$

Now from the discussion of Proposition 2 (in Appendix A) and Proposition 3, we can select δ_i such that

$$H_M\left(\mathcal{C}(g_{i-1}), \mathcal{C}(g_{i-1}-\delta_i)\right) \leq C h_i^2,$$

for $i = 1, \ldots, m$, $C \in \mathbb{R}_+$. By selecting such δ_i, we have

$$\begin{aligned}
\gamma_i &= H_M\left(\mathcal{C}(g_{i-1}), \mathcal{C}(g_{i-1}-\delta_i)\right) \\
&\leq C h_i^2 \\
&\leq C h_{\max}^2.
\end{aligned}$$

Hence,

$$H_M\left(\phi_T\left(\mathcal{C}(g_0)\right), \mathcal{C}(g_m)\right) \leq C h_{\max}^2 \sum_{i=1}^{m} e^{F_{\max}(m-i+1) h_{\max}}. \tag{9}$$

It can be verified that

$$\lim_{h_{\max} \to 0} \sum_{i=1}^{m} e^{F_{\max}(m-i+1) h_{\max}} h_{\max} = \frac{e^{F_{\max} T} - 1}{F_{\max}}.$$

Therefore,

$$\lim_{h_{\max} \to 0} H_M(\phi_T(\mathcal{C}(g_0)), \mathcal{C}(g_m)) \leq \lim_{h_{\max} \to 0} \frac{e^{F_{\max} T} - 1}{F_{\max}} C h_{\max} = 0.$$

□

The above discussion gives us the information of how good the approximations are. From the above discussion, it can be seen that the Hausdorff distance between the approximated set and the actual advected set is upper bounded by the right hand side of (9). To reduce the upper bound, in each iteration, we choose δ_i, using Proposition 3, so that the Hausdorff distance between $\mathcal{C}_M(g_{i-1})$ and $\mathcal{C}_M(g_{i-1} - \delta_i)$ is within the desired level, Ch_i^2, for some pre-specified value of C. And then we

can use (9) to compute the upper bound of the approximation error. An example of estimating the upper bound is presented in Section IV.

C. Simplified Advection Algorithm

Using the result of Corollary 1, we can have more control over the truncation error. In order to estimate the truncation error, the algorithm increases the problem size to a large extent. From our experience, if the system coordinate is scaled such that $\mathcal{B}_1 \supset \mathcal{C}(g)$, the truncation error usually does not cause any problem. Hence, with proper scaling of the coordinate and without the last three constraints in (4), the simplified optimization problem is enough to provide a fast and reliable solution.

The compact semi-algebraic set $M = \{x \in \mathbb{R}^n | q(x) \leq 0\}$ in (4) is simply used for making the semi-algebraic sets compact to ensure the existence of the solutions. From our experience, dropping the q terms still gives us adequate flexibility to represent the solutions. Furthermore, without the q terms, the problem size is much smaller, and the solution can be directly used without taking the intersection with M. Moreover, from our experience, dropping the κ_i and ζ_i terms usually does no harm to the solution. In the following, we present the simplified version of the algorithm without q, κ_i and ζ_i terms. Once again, it is our experience that in order to reduce the problem caused by the truncation error, the system coordinate should be scaled such that $\mathcal{B}_1 \supset \mathcal{C}(g_i)$.

The simplified version of the complete advection algorithm is as follows. Given an initial compact set $S_0 = \{x \in \mathbb{R}^n | g_0(x) \leq 0\}$, we would like to approximately find the advected set $S_i = \{x \in \mathbb{R}^n | g_i(x) \leq 0\}$ such that $S_i \approx \{x \in \mathbb{R}^n | \phi_{-h}(x) \in S_{i-1}\}$. Pick time step $h \in \mathbb{R}_+$, the tolerance parameter $\delta_i \in \mathbb{R}_+$, and positive integers d, d_r. Pick $\beta_i \in \mathbb{R}_+$ such that $\phi_h(x) \in \mathcal{C}(g_{i-1} - \beta_i)$ for all $x \in \mathcal{C}(g_{i-1} - \delta_i)$. Solve, using SDP, the following feasibility problem for $g_i \in \mathbb{R}_d[x]$, and $s_1, \ldots, s_6 \in \Sigma^2_{d_r}$.

$$\begin{aligned} s_1 - s_2 g_{i-1} + B_{-h} g_i &= 0 \\ s_3 + s_4(g_{i-1} - \delta_i) - B_{-h} g_i &= 0 \\ s_5 + s_6(g_{i-1} - \beta_i) - g_i &= 0. \end{aligned} \tag{10}$$

It should be noted that in the simplified algorithm, we are no longer estimating the bound of the truncation error. Therefore, the growth of approximation error cannot be properly predicted. However, we still can estimate the estimation error after we have acquired the solution using the result of Proposition 1. More discussion of the usage of the approach will be presented in Section IV.

D. Approximate Fitting Advection Algorithm

The previous algorithm (10) gives us a low degree representation with specific bounds of the approximation error. This is done through the second and third constraints in the algorithm. Sometimes, we do not need a result with that much precision. In this case, we could search for a best fixed degree fitting of the advected set.

The following approach is inspired by observing the case of the best quadratic fitting.

Corollary 2: Consider a compact semi-algebraic set, Y, whose interior contains the origin. Suppose there exists a positively definite matrix $Q \in \mathbb{R}^{n \times n}$, and $\alpha \in \mathbb{R}+$ that solve the following optimization problem

$$\begin{aligned} &\min\ \alpha \\ &\text{s.t. } S_\alpha \subset Y \subset S \\ &\text{where } S = \{x \in \mathbb{R}^n \mid x^T Q x - 1 \leq 0\} \text{ and} \\ &\qquad S_\alpha = \{x \in \mathbb{R}^n \mid x^T Q x - 1 + \alpha \leq 0, \alpha > 0\}. \end{aligned}$$

Then S gives the best quadratic fitting of Y in the sense of relative radial error.

Proof: Clearly $S_\alpha \subset S$. Let $v_1 \in \partial(S)$ and let $v_2 = k v_1$, $k \in [0, 1]$ be a point lying on the boundary of S_α. Then we have $\alpha = 1 - k^2$.

Since $S_\alpha \subset Y \subset S$, there exists a $\tilde{k} \in [k, 1]$ such that $v_3 = \tilde{k} v_1 \in \partial(Y)$. Then we have

$$\begin{aligned} \|v_1 - v_3\|^2 &= (1 - \tilde{k}^2)\|v_1\|^2 \\ &\leq (1 - k^2)\|v_1\|^2 \\ &= \alpha \|v_1\|^2. \end{aligned}$$

Therefore, by minimizing α, we are minimizing the relative radial distance between the boundaries of S and Y. □

We extend the quadratic result to a more general case to get the following algorithm.

$$\begin{aligned} &\min\ \alpha \\ &\text{s.t. } g_i(\gamma_i) = \epsilon_i \\ &\qquad s_1 - s_2 g_{i-1} + B_{-h} g_i = 0 \\ &\qquad s_3 + s_4 g_{i-1} - B_{-h} g_i - \alpha = 0 \\ &\qquad s_5 + s_6(g_{i-1} - \beta_i) - g_i = 0 \\ &\qquad \deg(g_i) \leq d,\ \alpha \geq 0,\ \gamma_i \in \phi_h \mathcal{C}(g_{i-1}),\ \epsilon_i < 0,\ \beta_i > 0 \end{aligned} \tag{11}$$

The second and third constraints are the best fitting constraints, and the fourth constraint ensures $\mathcal{C}(g_i)$ does not have spurious parts outside $\mathcal{C}(g_{i-1} - \beta_i)$. This constraint also helps in controlling the truncation error of the Taylor approximation. Although, theoretically, the first constraint is not required, it helps to avoid getting a semi-algebraic set with a very shallow 0-sub-level-set which may cause numerical problems.

E. Implementation of the Proposed Algorithms

The steps of implementing the complete algorithm is shown in Algorithm 1. To implement the main advection algorithm, we need to firstly find the value of δ in (4), so that the backward advected set of the next time instant is close enough to the current set. The value of δ is found by using the results of Proposition 2 and Proposition 3. From the definition of $\delta_{\max}$ in Proposition 3, if r is small, $\delta_{\max}$ can be approximated as $r\underline{g}^2/\bar{g}$, where $\underline{g} = \min_{x \in \mathcal{V}(g)} \|Dg(x)\|$ and $\bar{g} = \max_{x \in \mathcal{V}(g)} \|Dg(x)\|$. The first step of Algorithm 1 shows the optimization problem for finding $\bar{g}$. The equality constraint used here is

$$s_1 + p_1 g + \|Dg\|^2 - \bar{g}^2 = 0,$$

where s_1 is SOS and p_1 is a polynomial. This ensures that whenever $x \in \mathcal{V}(g)$, we have $\|Dg\|^2 \leq \bar{g}^2$. Therefore, by minimizing

$\bar{g}$, we get $\bar{g} = \max_{x\in\mathcal{V}(g)} \|Dg(x)\|$. $\underline{g}$ is found using similar approaches. After we have found $\delta_{\max}$, we can pick $\delta = \delta_{\max}$ in the main algorithm.

In the optimization problem of the complete advection algorithm, we need to decide the values of κ, β, and ζ in (4). As described before, κ and ζ can be chosen as some small positive numbers. β is used to form the set bounding the system trajectories starting from $\mathcal{C}(g)$, and its value can be approximately set by time step h and $\bar{g}_f = \max_{x\in\mathcal{V}(g)} \|L_f g\|$. Step 4 shows the optimization problem for finding $\bar{g}_f$. One possible selection of β is shown in step 5. Polynomial q is used to form the bounding compact set $M = \mathcal{C}(q)$ shown in Fig. 2. One simple selection is to pick a large ball covering the sets. Then, by solving (4), we get the advected set $\mathcal{C}(p)$. The simplified algorithm and the approximate fitting algorithm can be implemented similarly.

Algorithm 1 The complete advection algorithm.

Require: $d, d_r, d_s, d_p \in \mathbb{Z}_+$: The degrees of polynomials;

$r > 0$: The approximation tolerance;

$\kappa, \zeta > 0$: Arbitrary small positive numbers;

$h > 0$: The step size;

g: The polynomial whose 0-sub-level-set is to be advected;

q: The polynomial whose 0-sub-level-set satisfies the conditions shown in Fig. 2.

Ensure: $p \in \mathbb{R}_d[x]$ such that $H(\mathcal{C}_M(g), \phi_{-h}(\mathcal{C}_M(p))) \le r$.

1) Solve for $s_1 \in \Sigma^2_{d_s}$, $p_1 \in \mathbb{R}_{d_p}[x]$, and $\bar{g}$ from the following optimization problem:

$$\begin{aligned} &\min\ \bar{g}^2 \\ &\text{s.t. } s_1 + p_1 g + \|Dg\|^2 - \bar{g}^2 = 0. \end{aligned}$$

2) Solve for $s_2 \in \Sigma^2_{d_s}$, $p_2 \in \mathbb{R}_{d_p}[x]$, and $\underline{g}$ from the following optimization problem:

$$\begin{aligned} &\max\ \underline{g}^2 \\ &\text{s.t. } s_2 + p_2 g + \underline{g}^2 - \|Dg\|^2 = 0. \end{aligned}$$

3) Let $\delta = \delta_{\max} = r\underline{g}^2/\bar{g}$.
4) Solve for $s_3 \in \Sigma^2_{d_s}$, $p_3 \in \mathbb{R}_{d_p}[x]$, and $\bar{g}_f$ from the following optimization problem:

$$\begin{aligned} &\min\ \bar{g}_f^2 \\ &\text{s.t. } s_3 + p_3 g + \|L_f g\|^2 - \bar{g}_f^2 = 0. \end{aligned}$$

5) Let $\beta = 2h\bar{g}_f$.
6) Solve for $p \in \mathbb{R}_d[x]$, $s_1, \ldots, s_{18} \in \Sigma^2_{d_r}$ and $\eta \in \mathbb{R}_+$ from the optimization problem (4).

IV. Numerical Examples of Advection Algorithms

Example 2: Consider the following dynamical system

$$\begin{aligned} \dot{x} &= y \\ \dot{y} &= -0.2x - 0.1y. \end{aligned}$$

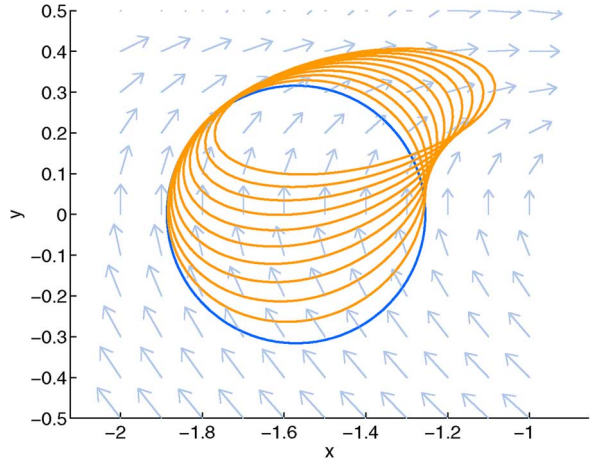

Fig. 3. Successive iterates of the advection algorithm.

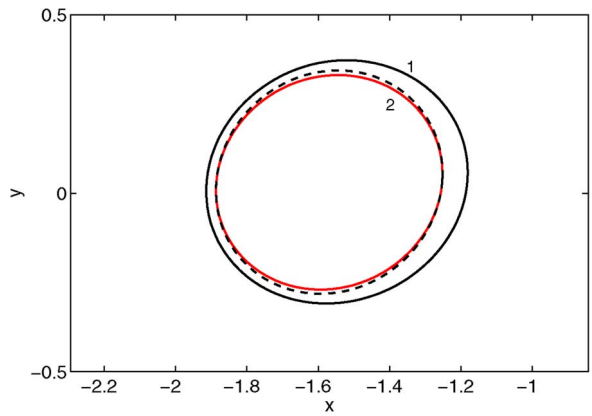

Fig. 4. Truncation error estimation. Curve 1 is the expanded set to cover all the reachable points. Curve 2 is the advection algorithm result. Dashed curve is the simulated result.

Suppose we are interested in the reachable sets from the initial set

$$S_0 = \left\{(x, y) \in \mathbb{R}^2 \mid \left(x + \frac{\pi}{2}\right)^2 + y^2 \le 0.1\right\}.$$

Apply the advection algorithm, repeatedly; we can get the estimated advected set at different time steps. Fig. 3 shows the result of first ten steps with time step 0.1 using the simplified algorithm. The boundary of the initial set is the ball $\mathcal{B}_{\sqrt{0.1}}(-(\pi/2), 0)$.

We can also estimate the truncation error. After getting the advection set $S_1 = \{(x, y) \in \mathbb{R}^2 | \phi_{-0.1}(x, y) \in S_0\}$, the estimated truncation error is less than 0.29. We could expand S_1 accordingly so that the expanded S_1 is guaranteed to cover all the possible points starting from S_0. The result is shown in Fig. 4. The solid curve 2 is the boundary of the advected set S_1, and the solid curve 1 is the boundary of the expanded S_1. The dashed curve shows the result by tracing 100 points from the boundary of S_0. As can be seen in Fig. 4, with careful selection of scaling and time step, we could get a reasonable estimation of the advected set. The estimated truncation error does help us find a guaranteed bound of the reachable set. It can also be observedthat the use of Lipschitz constant to form the bound may be conservative. The use one-sided Lipschitz constant [31], [32] may give us a better bound and will be investigated in the future.

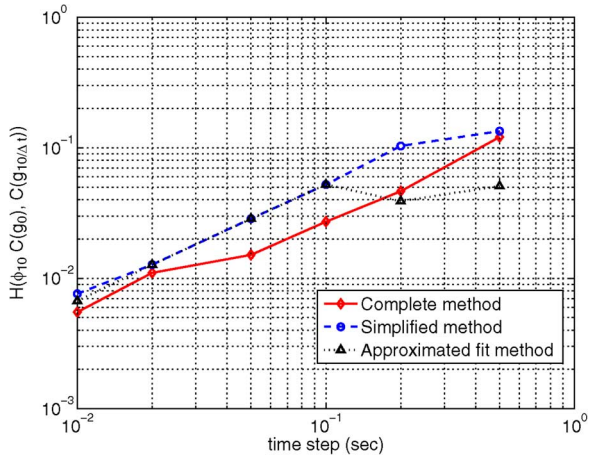

Fig. 5. Comparison of the accuracy of the three different algorithms.

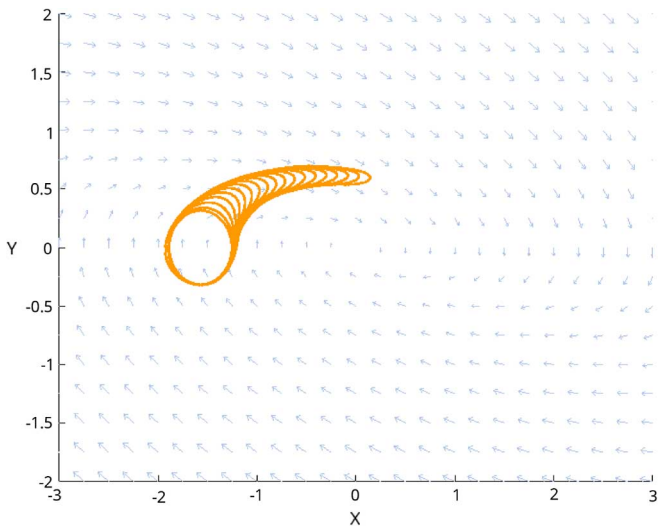

Fig. 6. Reachable sets of example 2.

Theoretically, the complete advection algorithm should give the solution with the best accuracy. From our numerical experience, when the step size is not too large, all the three algorithms give similar results. Fig. 5 shows the comparison of accuracy of each algorithm. The horizontal axis is the fixed time step size, h, used in the simulation. {The vertical axis is the Hausdorff metric between $\phi_{10}\mathcal{C}(g_0)$ and $\mathcal{C}(g_m)$, where $mh = 10$.

From Fig. 5, we can see that the complete algorithm does have better results than the simplified algorithm. However, the differences are not very significant when the step size is less than 0.02. The approximate fitting algorithm works exceptionally well even for a larger step size. However, this may be only a lucky result since the shape of the approximated shape happens to be of elliptic shape. If the user chooses a low degree polynomial to fit the advected set, the result may be much worse than the simplified algorithm. The theoretical analysis for the approximate fitting algorithm will be further analyzed in the future.

If we are interested in finding the reachable set from system points in the initial set, we can take a union of all the semi-algebraic-sets at all time steps using the SOS techniques stated in Section II. Fig. 6 shows the result of the union of the first few steps.

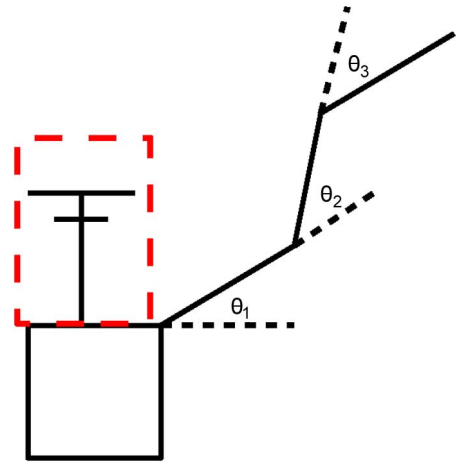

Fig. 7. Solar panel deployment problem.

Example 3: This example is a solar panel deployment problem for a satellite. The solar panels are folded during the launch procedure. When the satellite reaches the designated position, it starts to deploy the solar panels. Fig. 7 shows a satellite with three solar panels connected in serial by hinges. The hinges of the solar panels have torsional springs installed with latches to lock the solar panels in the deployed states. The torsional spring is made of tape springs so that the exerted moment is minimum at both the folded position and the deployed position.

In this example, we consider only the system states that exist during the deployment process. Therefore, the effect of the latch is not considered here. For demonstration, we consider the following one panel system

$$\dot{x} = y$$
$$\dot{y} = 2\left(-\frac{0.4}{\pi} - \frac{\pi^2}{4}\right)x + 2\left(-\pi - \frac{0.4}{\pi^2}\right)x^2 - 2x^3 - 0.1y,$$

where $x = \theta$ is the deflection angle of the solar panel and $y = \dot{\theta}$. Here we added a small damping to reflect the friction of the hinge. Suppose, again, that we are interested in the reachable sets from the initial set

$$S_0 = \left\{(x, y) \in \mathbb{R}^2 \mid \left(x + \frac{\pi}{2}\right)^2 + (y - \sqrt{0.1})^2 \leq 0.1\right\},$$

which contains all the possible configurations of the push spring that will initiate the deployment process when the satellite reaches its desired position. In this example, we use degree 4 polynomials to represent the advected sets. Fig. 8 shows some of the advected sets with $h = 0.05$. As can be seen in this figure, the advected set flows under the system's vector field. We can then analyze the union of these sets to see whether the system states will pass through some undesired configurations.

Example 4: In this example, we study the flow of the initial set

$$S_0 = \left\{(x, y) | (x + 1)^2 + (y - 1)^2 \leq 0.2\right\},$$

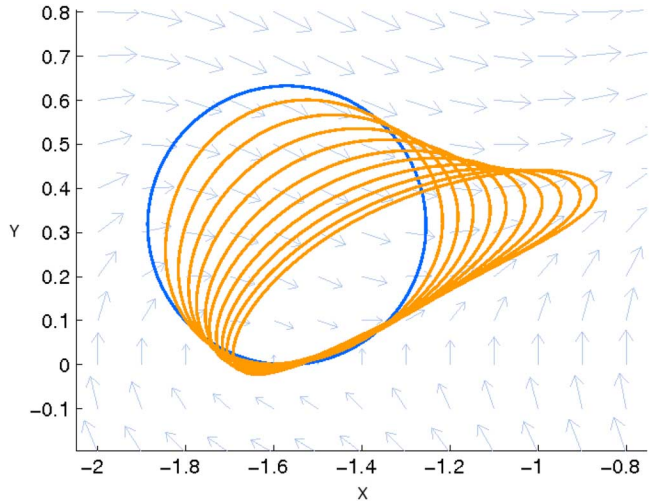

Fig. 8. Some advected sets of example 3.

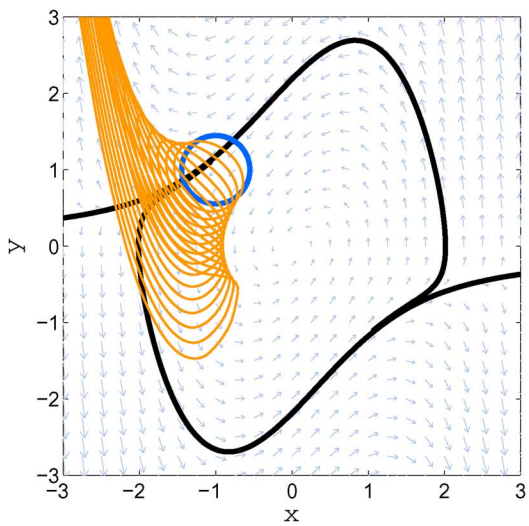

Fig. 9. Some advected sets of example 4.

in a Van-der Pol oscillator system in backward time. The system dynamics can be formulated as

$$\dot{x} = -y$$
$$\dot{y} = x - y(1 - x^2).$$

The initial set covers a portion of the domain of attraction of the system, and a portion in which the states will diverge. Since the system states change rapidly in the unstable region, we use second order approximation in the advection algorithm. Fig. 9 shows the result of the first 28 time steps with step size 0.02. Only the boundaries of sets with even-numbered iterates are shown in this figure. As can be seen, the advection algorithm successfully captures the variation of the initial set with time.

If we place the initial set around the origin, we can use the advection algorithm in backward time to get an estimation of the DoA around the origin. The left hand side of Fig. 10 shows some iteration results of using the advection algorithm for DoA estimation. The right hand side of Fig. 10 shows the comparison of our result with some other existing methods. As can be seen in Fig. 10, the result generated by the proposed method is much more precise than other approaches. Composite polynomial Lyapunov function is also capable of generating competitive results of estimating the DoA of the Van-der Pol system. However, our approach is still slightly better than the composite polynomial Lyapunov approach [14].

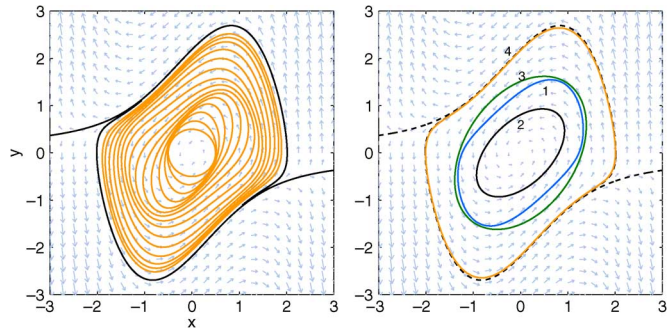

Fig. 10. Van der Pol oscillator. The left figure shows the sequence of iterations. The right figure shows the 40th iterate as Curve 4 along with some other results. Curve 1 is the result using the method from [19]. Curve 2 is the result of [8] and Curve 3 is the result of [7].

TABLE I
TIMING OF THREE ALGORITHMS FOR EXAMPLE 4

Algorithm	Dim. of A	T_1(sec.)	T_2(sec.)
Complete	396×5392	0.333	2.086
Simplified	199×2044	0.302	0.673
Approx. fit	199×2044	0.256	0.444

A. Computation Time

Computation of the advection algorithm can be divided into two parts. The first part is to prepare the matrices A, C and vector b for the SDP solver. The second part is to solve the SDP using current available SDP solvers.

Table I shows the timing results and problem sizes of Example 4. In this example, the degree of the solution polynomial is 8 and the degree of SOS polynomials is 10. In this table, T_1 is used to represent the time used to compute the matrices of the associated SDP problem and T_2 is used to represent the time used to solve the SDP problem using SeDuMi [33]. The example is solved on a machine with an Intel i5 M540 CPU and 6 GB memory.

As can be seen in Table I, simplified algorithm and the approximated fitting algorithm have similar timing results. The complete algorithm has much larger size of matrix A and thus the computation time is significantly higher than the other two algorithms. The time used for solving the SDP relies on the capability of the SDP solver. SeDuMi [33] is a fast and reliable solver for small to medium sized problems. For large SDP problems, there are some existing SDP solvers that utilize the power of parallel computing and show good performance for large sized problems. With the capabilities of the current SDP solvers, we are able to apply the proposed method for systems with 4 state variables. For systems with more than 4 state variables, a more powerful SDP solver is needed. Also the time required for doing the advection is not fast enough for real-time applications. However, once the advected set is computed, we can use the computed set for further analysis purposes.

It should be noted that the proposed algorithm is an iterative approach. Therefore, in order to explore the entire reachable set, many iterations are required. Hence, the total computation time would be several times of the computation time listed in Table I. Other existing approaches solve the estimation problem by solving a single optimization problem and hence consume less computational time. However, existing estimation approaches are based on the invariant property of the stable sets and can only be used for estimating the DoA. On the other hand, the proposed approach does not require such invariant-set property and can estimate unstable reachable sets such as the case shown in Fig. 9.

V. Conclusions

In this paper, we advect sets forward and backward in time to explore the DoA or reachable sets of a polynomial system. A level-set algorithm is proposed to advect subsets of the state space using semi-definite programming. The sets are represented as semi-algebraic sets, i.e., sub-level-sets of polynomials. The proposed algorithm generates a polynomial whose 0-sub-level-set approximately represents the advected set. The algorithm is also proved to be convergent. From the derivation of this level-set algorithm, the algorithm not only works for two-dimensional systems, but also for higher dimensional systems.

Although Putinar's theorem provides the theoretical support for the existence of the solutions, it does not give us the upper bounds of the degrees of polynomials used in such representations. Hence, when the algorithm fails, it might be due to insufficient degree in either the sum-of-squares polynomials or the polynomial used to represent the advected set. Adding the degree incorrectly will lead to an increase in problem size while still not yielding a feasible solution. A more careful study on the sufficiency of the degree is required in the future.

Appendix A
Preliminaries and Mathematical Background

The following proposition discusses the truncation error caused by the first-order Taylor approximation.

Proposition 1: Consider a polynomial $p \in \mathbb{R}[x]$. Suppose $h \in \mathbb{R}_+$ and $M \subset \mathbb{R}^n$ is a compact set such that the underlying analytic autonomous system, $\dot{x} = f(x)$, with flow map $\phi_t(x)$, has unique solutions defined for any initial condition in M and $t \in [0, h]$. Then, there exists $K, \epsilon \in \mathbb{R}_+$ such that

$$\|B_h p - A_h p\| \leq \epsilon = K\frac{h^2}{2}, \text{ for all } x \in M.$$

That is, the error is proportional to the square of the step size.

Proof: Using Lie-derivative and Taylor's Theorem, for any point $x \in M$, there exists a point $y = \phi_t(x)$ where $t \in [0, h]$ such that

$$\begin{aligned} A_h p(x) &= p(x) - hDp(x)f(x) + L_f^2 p(y)\frac{h^2}{2} \\ &= B_h p(x) + L_f^2 p(y)\frac{h^2}{2}, \end{aligned}$$

where $L_f^2 p$ is the second order Lie-derivative of p. Let $\bar{M}$ be a bigger compact set such that $\bar{M} \supset M$ and $\phi_t(x) \in \bar{M}$, for all $x \in M$, $t \in [0, h]$. Pick a point $z \in M$, and let $r = \sup_{x,y\in\bar{M}} \|x - y\|$. Since the system is analytic, there exists a Lipschitz constant N for $L_f^2 p$ in $\bar{M}$. Then, we have

$$\begin{aligned} \left\|L_f^2 p(y)\right\| \frac{h^2}{2} &= \left\|L_f^2 p(y) - L_f^2 p(z) + L_f^2 p(z)\right\| \frac{h^2}{2} \\ &\leq Nr\frac{h^2}{2} + \left\|L_f^2 p(z)\right\| \frac{h^2}{2}. \end{aligned}$$

Hence, we have the result with $\epsilon = (Nr + \|L_f^2 p(z)\|)h^2/2$. □

The following result shows the continuity property of $\mathcal{C}(g-\delta)$ with respect to the change in δ.

Proposition 2: Given a polynomial $g \in \mathbb{R}[x]$, then $\mathcal{C}(g - \delta)$ is upper semicontinuous in δ, and $H(\mathcal{C}(g), \mathcal{C}(g - \delta)) \to 0$ as $\delta \to 0^+$.

Proof: $\mathcal{C}(g - \delta)$ is a set valued function of δ. Its graph is a set of points $(\delta, x) \in \mathbb{R} \times \mathbb{R}^n$ such that $\delta \in \mathbb{R}$ and $x \in \mathcal{C}(g - \delta)$. Since g is a polynomial, its epigraph is closed. The graph of $\mathcal{C}(g - \delta)$ is its epigraph rotated, and is closed for $\delta \in \mathbb{R}$. Hence, the function $\mathcal{C}(g - \delta)$ is upper semicontinuous on $\mathbb{R}$ [34], [35].

Since $\xi(\mathcal{C}(g), \mathcal{C}(g - \delta)) = 0$ when $\delta \in \mathbb{R}_+$. Therefore $H(\mathcal{C}(g), \mathcal{C}(g - \delta)) = \xi(\mathcal{C}(g - \delta), \mathcal{C}(g))$ for $\delta \in \mathbb{R}_+$. By the property of upper semicontinuity, $H(\mathcal{C}(g), \mathcal{C}(g - \delta)) \to 0$ as $\delta \to 0^+$. □

Proposition 2 shows that we can approximate $\mathcal{C}(g)$ arbitrarily well by choosing δ arbitrarily close to 0 from above. This property will be used for getting a low degree polynomial $p \in \mathbb{R}[x]$ such that $\mathcal{C}(B_{-h}p)$ approximates $\mathcal{C}(g)$ in the sense of Hausdorff metric. To ascertain how small δ must be to achieve a desired value of $H(\mathcal{C}(g), \mathcal{C}(g - \delta))$, we use the result of the following proposition.

Proposition 3: Consider a polynomial $g \in \mathbb{R}[x]$ such that $\mathcal{C}(g)$ is compact. Let $\bar{g} = \max_{x\in\mathcal{V}(g)} \|Dg(x)\|$ and assume $\bar{g} \neq 0$. Given $r \in \mathbb{R}_+$, let

$$\delta_{\max} = -\max_{x\in\mathcal{V}(g)} \min_{l\in[0,r]} g\left(x - l\frac{Dg(x)}{\bar{g}}\right), \tag{12}$$

Then, for any $\delta \in [0, \delta_{\max}]$, we have

$$H\left(\mathcal{C}(g), \mathcal{C}(g + \delta)\right) \leq r$$

Proof: From the definition of $\delta_{\max}$, we know $\delta_{\max} \geq 0$. For any $\delta \in [0, \delta_{\max}]$, it is clear that

$$\mathcal{C}(g + \delta) \subset \mathcal{C}(g).$$

Hence,

$$\xi\left(\mathcal{C}(g + \delta), \mathcal{C}(g)\right) = 0.$$

We only need to show

$$\xi\left(\mathcal{C}(g), \mathcal{C}(g + \delta)\right) \leq r.$$

Note that the maximum distance occurs at $x \in \mathcal{V}(g)$. In the following segment we will concentrate on the points on the boundary.

For any $x \in \mathcal{V}(g)$, there exist $\tilde{l} \in [0, r]$ such that

$$g\left(x - \tilde{l}\frac{Dg(x)}{\bar{g}}\right) = -\delta.$$

Taking $y = x - \tilde{l}(Dg(x)/\bar{g})$, we have $y \in \mathcal{C}(g+\delta)$ and

$$\|x-y\| = \tilde{l}\left\|\frac{Dg}{\bar{g}}\right\| \le \tilde{l} \le r,$$

and the result follows. □

Note that Proposition 2 shows the continuity property for a bigger set while Proposition 3 tries to find a smaller set. We could first try an expanded set, $\mathcal{C}(g-\epsilon)$, $\epsilon > 0$, and then use the result of Proposition 3 to find the $\delta_{\max}$ such that $H(\mathcal{C}(g-\epsilon), \mathcal{C}(g-\epsilon+\delta_{\max})) < r$. If $\delta_{\max} > \epsilon$, then we have

$$H\left(\mathcal{C}(g-\epsilon), \mathcal{C}(g)\right) < H\left(\mathcal{C}(g-\epsilon), \mathcal{C}(g-\epsilon+\delta_{\max})\right) < r.$$

If it happens that $\bar{g} = 0$, one could add a small number to g so that $\bar{g}$ is nonzero then follow similar approaches to find ϵ.

Using Proposition 2 and Proposition 3, we can find a $\delta > 0$ such that $H(\mathcal{C}(g), \mathcal{C}(g-\delta)) \le r$ for a given distance $r \in \mathbb{R}_+$.

The following results are for the positive time step case. The negative time step case has similar results and is thus omitted here.

Proposition 4: Suppose $M \subset \mathbb{R}^n$ is compact, $p \in \mathbb{R}[x]$ and $h > 0$ is such that ϕ_h is well defined on M. Then for all $\epsilon > 0$ there exists a polynomial g_ϵ with $0 < (g_\epsilon - A_h p)(x) < \epsilon$ for all $x \in M$.

Proof: $A_h p + (\epsilon/2)$ is a continuous function. From *Stone-Weierstrass Theorem*, there exists a polynomial g_ϵ such that $|g_\epsilon - (A_h p + (\epsilon/2))| < \epsilon/2$ for all $x \in M$. Hence, $0 < g_\epsilon - A_h p < \epsilon$ for all $x \in M$. □

From Proposition 4, we know that the value of an advected polynomial $A_h p$ can be well approximated by another polynomial g. We would also want to know whether the advected sub-level-set can be well approximated by another polynomial. The following lemma shows the result.

Lemma 2: Let M be a compact set and $h^* > 0$ be such that ϕ_t is well defined on M for all $t \in [-h^*, h^*]$. Let p be a polynomial such that

$$\phi_t\left(\mathcal{C}(p)\right) \subset M \text{ for all } t \in [-h^*, h^*]$$

Then, for all $\delta > 0$ and for all $h \in [-h^*, h^*]$, there exists a polynomial $q \in \mathbb{R}[x]$ such that

$$H_M\left(\mathcal{C}(A_{-h}q), \mathcal{C}(p)\right) \le \delta$$

Proof: From Proposition 4, for any $\epsilon \in \mathbb{R}_+$, there exist $q \in \mathbb{R}[x]$, $0 < h < h^*$ such that

$$0 < q - A_h p < \epsilon \text{ for all } x \in M.$$

Hence,

$$H_M\left(\mathcal{C}(q), \mathcal{C}(A_h p)\right) \le H_M\left(\mathcal{C}(q), \mathcal{C}(q-\epsilon)\right).$$

For simplicity, let $\gamma = H_M(\mathcal{C}(q), \mathcal{C}(q-\epsilon))$.

Let $M^+ \supset M$ be such that $\phi_t M \subset M^+$ for $t \in [-h^*, h^*]$, and let F be a Lipschitz constant of $f(x)$ in M^+. For any $x \in \mathcal{C}(q)$, $y \in \mathcal{C}(A_h p)$, we have

$$\begin{aligned}|\phi_{-h}x - \phi_{-h}y| &\le |x-y| + \int_0^{-h} |f\left(x(\tau)\right) - f\left(y(\tau)\right)|\, d\tau \\ &\le |x-y| + \int_0^{-h} F\,|x(\tau) - y(\tau)|\, d\tau.\end{aligned}$$

Using *Gronwall-Bellman* inequality, we have $|\phi_{-h}x - \phi_{-h}y| \le e^{Fh}|x-y|$.

Using the uniqueness of the solutions, for any $z \in \mathcal{C}(A_{-h}q)$, there exists $y \in \mathcal{C}(A_h p)$ such that

$$|z - \phi_{-h}y| \le e^{Fh}\gamma.$$

Let $w = \phi_{-h}y$. Then, for any $z \in \mathcal{C}(A_{-h}q)$, there exists $w \in \mathcal{C}(p)$ such that

$$|z - w| \le e^{Fh}\gamma.$$

Using this procedure, it can be shown that

$$H_M\left(\mathcal{C}(A_{-h}q), \mathcal{C}(p)\right) \le e^{Fh}\gamma.$$

Here, $e^{Fh}\gamma$ is the upper bound of the error between p and $A_{-h}q$ in M. From Proposition 2, γ can be made arbitrarily small such that $e^{Fh}\gamma \le \delta$. □

Lemma 2 shows that the advected sub-level-set can indeed be well approximated by the sub-level-set of a polynomial. However, since $\delta \ge e^{Fh}\gamma$, to achieve the desired bound δ, a much smaller h is required.

Now we would like to know whether the result holds for the Taylor series approximated advection. The result is shown as follows.

Lemma 3: Consider a compact set M in which the autonomous system has unique solutions defined for all $x \in M$, $t \in [-h^*, h^*]$, for some $h^* > 0$. For every polynomial q such that $\phi_t(\mathcal{C}(q)) \subset M$ for $t \in [-h^*, h^*]$, there exist $0 < h < h^*$, and a bound $\eta > 0$ such that

$$H_M\left(\mathcal{C}(A_{-h}q), \mathcal{C}(B_{-h}q)\right) \le H_M\left(\mathcal{C}(B_{-h}q - \eta), \mathcal{C}(B_{-h}q + \eta)\right).$$

Further suppose $\|\nabla(B_{-h}q(x))\|$ is nonzero for all $x \in (\mathcal{C}(B_{-h}q - \zeta) \setminus \mathcal{C}(B_{-h}q + \zeta))$ for some $\zeta > 0$. Then, for any $\epsilon > 0$, there exist $0 < h < h^*$, and $\eta > 0$ such that

$$H_M\left(\mathcal{C}(A_{-h}q), \mathcal{C}(B_{-h}q)\right) \le \epsilon.$$

Proof: From Proposition 1,

$$|A_{-h}q(x) - B_{-h}q(x)| \le \left(Nr + \left|L_f^2 q(z)\right|\right)\frac{h^2}{2} \text{ for all } x \in M$$

where N, r, z are as defined in Proposition 1. Let $\eta = (Nr + |L_f^2 q(z)|)h^2/2$. Then

$$H_M\left(\mathcal{C}(A_{-h}q), \mathcal{C}(B_{-h}q)\right) \leq H_M\left(\mathcal{C}(B_{-h}q - \eta), \mathcal{C}(B_{-h}q + \eta)\right).$$

Note that η is a function of h and can be made arbitrarily small by picking a small h.

Now suppose $\|\nabla(B_{-h}q(x))\|$ is nonzero for all $x \in (\mathcal{C}(B_{-h}q - \eta) \setminus \mathcal{C}(B_{-h}q + \eta))$. From Proposition 2, there exist $\eta_u > 0$ such that

$$H_M\left(\mathcal{C}(B_{-h}q - \eta_u), \mathcal{C}(B_{-h}q)\right) \leq \frac{\epsilon}{2}.$$

From Proposition 3, since the gradient is nonzero, there exists $\eta_l > 0$ such that

$$H_M\left(\mathcal{C}(B_{-h}q + \eta_l), \mathcal{C}(B_{-h}q)\right) \leq \frac{\epsilon}{2}.$$

Now pick $\eta_l < \zeta$ and let $\eta = \min(\eta_u, \eta_l)$. Then, using triangle inequality, we have

$$H_M\left(\mathcal{C}(A_{-h}q), \mathcal{C}(B_{-h}q)\right) \leq \epsilon.$$

□

REFERENCES

[1] H. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River: Prentice-Hall Inc., 2000.

[2] I. M. Mitchell and C. J. Tomlin, "Level set methods for computation in hybrid systems," *Hybrid Systems: Computation and Control*, vol. 1790/2000, pp. 310–323, 2000, Springer Verlag.

[3] M. Greenstreet, "Verifying safety properties of differential equations," in *Proc. Conf. Computer Aided Verification*, 1996, pp. 277–287.

[4] A. Chutinan and B. Krogh, "Computational techniques for hybrid system verification," *IEEE Trans. Automat. Control*, vol. 48, no. 1, pp. 64–75, Jan. 2003.

[5] E. Asarin, O. Bournez, T. Dang, and O. Maler, "Approximate reachability analysis of piecewise-linear dynamical systems," *Hybrid Systems: Computation and Control*, vol. 1790/2000, pp. 21–31, 2000, Springer.

[6] A. Kurzhanskiy and P. Varaiya, "Ellipsoidal techniques for reachability analysis," *Hybrid Systems: Computation and Control*, vol. 1790/2000, pp. 202–214, 2000, Springer.

[7] E. J. Davison and E. M. Kurak, "A computational method for determining quadratic Lyapunov functions for non-linear systems," *Automatica*, vol. 7, pp. 627–636, 1971.

[8] A. Levin, "An analytical method of estimating the domain of attraction for polynomial differential equations," *IEEE Trans. Automat. Control*, vol. 39, no. 12, pp. 2471–2475, 1994.

[9] G. Chesi, A. Tesi, and A. Vicino, "On optimal quadratic Lyapunov functions for polynomial systems," in *Proc. 15th Int. Symp. Mathemat. Theory Networks Syst.*, 2002.

[10] G. Chesi, "Estimating the domain of attraction for uncertain polynomial systems," *Automatica* vol. 40, no. 11, pp. 1981–1986, 2004 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109804002018

[11] G. Chesi, "Estimating the domain of attraction via union of continuous families of Lyapunov estimates," *Syst. Control Lett.* vol. 56, no. 4, pp. 326–333, 2007 [Online]. Available: http://www.sciencedirect.com/science/article/B6V4X-4MFJT56-3/2/2f661958 8e11808a3f4c64628873fac6

[12] G. Chesi, "Estimating the domain of attraction for non-polynomial systems via LMI optimizations," *Automatica* vol. 45, no. 6, pp. 1536–1541, 2009 [Online]. Available: http://www.sciencedirect.com/science/article/B6V21-4VVW4TS-5/2/3dcb0694ad29f5bfc2d47da7381aa934

[13] G. Chesi, *Homogeneous Polynomial Forms for Robustness Analysis of Uncertain Systems*. Berlin: Springer, 2009.

[14] W. Tan and A. Packard, "Stability region analysis using polynomial and composite polynomial Lyapunov functions and sum-of-squares programming," *IEEE Trans. Automat. Control*, vol. 53, no. 2, pp. 565–571, Mar. 2008.

[15] P. A. Parrilo and S. Lall, "Semidefinite programming relaxations and algebraic optimization in control," *E. J. Control*, vol. 9, no. 2–3, pp. 307–321, Apr. 2003.

[16] A. Papachristodoulou and S. Prajna, "On the construction of Lyapunov functions using the sum of squares decomposition," in *Proc. 41st IEEE CDC*, 2002.

[17] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing sostools: A general purpose sum of squares programming solver," in *Proc. 41st IEEE CDC*, December 2002.

[18] P. Parrilo, "Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization," Ph.D. dissertation, California Institute of Technology, , 2000.

[19] Z. W. Jarvis-Wloszek, "Lyapunov Based Analysis and Controllers Synthesis for Polynomial Systems Using Sum-of-Squares Optimization," Ph.D. dissertation, Univ. California, Berkeley, 2003.

[20] J. A. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge: Cambridge Univ. Press, 1999.

[21] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. : Springer Verlag, 2002.

[22] T. Wang, S. Lall, and M. West, "Polynomial level-set methods for nonlinear dynamical systems analysis," in *Proc. Allerton Conf. Communication, Control, Computing*, 2005.

[23] T.-C. Wang, S. Lall, and T.-Y. Chiou, "Polynomial method for PLL controller optimization," *Sensors* vol. 11, no. 7, pp. 6575–6592, 2011 [Online]. Available: http://www.mdpi.com/1424-8220/11/7/6575/

[24] S. Lall, M. Peet, and T. Wang, SOSCODE: Sum of Squares Optimization Toolbox for MATLAB 2005 [Online]. Available: http://www.stanford.edu/lall/

[25] M. Putinar, "Positive polynomials on compact semi-algebraic sets," *Indiana University Mathematics Journal*, vol. 42, no. 3, pp. 969–984, 1993.

[26] A. Balestrino, A. Caiti, E. Crisostomi, and S. Grammatico, "R-composition of Lyapunov functions," in *Proc. 17th Mediterranean Conf. Control Automat.*, June 2009, pp. 126–131.

[27] A. Balestrino, A. Caiti, E. Crisostomi, and S. Grammatico, "Stability analysis of dynamical systems via r-functions," in *Proc. IEEE Eur. Control Conf.* , 2009.

[28] A. Balestrino, A. Caiti, E. Crisostomi, and S. Grammatico, "Stability of linear differential inclusions via r-functions," in *Proc. 8th IFAC Symp. Nonlinear Control Syst.*, 2009.

[29] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, Bifurcations of Vector Fields*. Berlin: Springer-Verlag, 1983.

[30] S. Wiggins, *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, 2nd ed. Berlin: Springer Sci. Business Media Inc., 2003.

[31] E. Hairer, G. Wanner, and S. P. Nørsett, *Solving Ordinary Differential Equations I, II*. Berlin: Springer, 2008.

[32] T. Donchev and E. Farkhi, "Stability and euler approximation of one-sided lipschitz differential inclusions," *SIAM J. Control Optimizat.*, vol. 36, no. 2, pp. 780–796, 1998.

[33] J. F. Sturm, SeDuMi: A MATLAB Toolbox for Optimization Over Symmetric Cones. Software, User's Guide and Benchmarks. Version 1.03 Sep. 1999 [Online]. Available: http://sedumi.mcmaster.ca

[34] A. F. Filippov, *Differential Equations With Discontinuous Righthand Sides*. New York: Kluwer Academic, 1988.

[35] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*. Berlin: Springer-Verlag, 1998.

Ta-Chung Wang (S'02–M'07) received the B. S. and M. S. degrees in mechanical engineering from the National Taiwan University, Taipei, Taiwan, in 1995 and 2000, respectively, and the Ph.D. degree in aeronautics and astronautics from Stanford University, Stanford, CA. in 2007.

Since 2007, he is an Assistant Professor of the Institute of Civil Aviation, National Cheng-Kung University, Tainan, Taiwan. He teaches courses in multivariate analysis, electronics and engineering mathematics. His research interests include operations research in the aviation industry, solving large problems using parallel approaches, and nonlinear system analysis.

Sanjay Lall (M'92–SM'06) received the Ph.D. in engineering and B.A. in mathematics from the University of Cambridge, Cambridge, U.K.

He is an Associate Professor of Electrical Engineering and Associate Professor of Aeronautics and Astronautics at Stanford University, Stanford, CA, USA. Previously, he has held positions at the California Institute of Technology and the Massachusetts Institute of Technology. His research focuses on the development of advanced engineering methodologies for the design of control systems.

Professor Lall received the George S. Axelby Outstanding Paper Award from the IEEE Control Systems Society, the NSF Career award, Presidential Early Career Award for Scientists and Engineers (PECASE), and the Stanford University Graduate Service Recognition Award.

Matthew West received the B.Sc. in pure and applied mathematics from the University of Western Australia, Crawley WA, Australia, in 1996, and the Ph.D. in control and dynamical systems from the California Institute of Technology, Pasadena, CA, USA, in 2004.

He was an Assistant Professor in the Department of Mathematics at the University of California, Davis, CA, USA, from 2003 to 2004, and an Assistant Professor in the Department of Aeronautics and Astronautics at Stanford University, Stanford, CA, USA, from 2004 to 2007. He joined the Department of Mechanical Science and Engineering at the University of Illinois at Urbana-Champaign, Urbana, IL, USA, as an Assistant Professor in 2008, and he is currently an Associate Professor in this department. His research interests include time integration of deterministic and stochastic systems, distributed computation, and particle methods for simulation and estimation.

Dr. West is a recipient of the NSF CAREER award.