Available online at www.sciencedirect.com

SciVerse ScienceDirect

Journal of the Franklin Institute 349 (2012) 2783–2798

Journal of The Franklin Institute

www.elsevier.com/locate/jfranklin

Polynomial level-set method for attractor estimation

Ta-Chung Wang[a,*], Sanjay Lall[b], Matthew West[c]

[a]Institute of Civil Aviation, National Cheng-Kung University, No. 1 University Road, Tainan 701, Taiwan
[b]Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA
[c]Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA

Received 26 October 2011; received in revised form 17 April 2012; accepted 27 August 2012
Available online 6 September 2012

Abstract

In this study, we present a polynomial level-set method for attractor estimation. This method uses the sub-level representation of sets. The problem of flowing these sets under the advection map of a dynamic system is converted to a semi-definite program, which is used to compute the coefficients of the polynomials. The required storage space for describing the result is much less than the mesh-based methods. The characteristics of attractors are used in the algorithm formulations so that the associated numerical error can be reduced. We further address the related problems of constraining the degree of the polynomials. Various numerical examples are used to show the effectiveness of the advection approach.

1. Introduction

The introduction of Lyapunov theory brings the attentions of analyzing invariant sets in control field. In Lyapunov theory, the stability of a system can be justified by testing whether a form of energy of a system is positively invariant. Level surfaces of such form of energy are the boundaries of positively invariant sets. Applications of set invariance have been proposed in many fields such as stability analysis [1–4], controller synthesis [5], robustness analysis [6], disturbance rejection [7], and performance analysis [8].

One example of positively invariant sets of a dynamic system is the attractor. An attractor is the smallest positively invariant set of a dynamic system. Any initial system state starting in the

*Corresponding author. Tel.: +886 6 275 7575x63620; fax: +886 6 238 9940.
E-mail addresses: tachung@mail.ncku.edu.tw (T.-C. Wang), lall@stanford.edu (S. Lall), mwest@illinois.edu (M. West).

0016-0032/$32.00
http://dx.doi.org/10.1016/j.jfranklin.2012.08.014

domain-of-attraction (DoA) of an attractor will eventually move inside the attractor. Therefore, attractors could be used as verification tools to determine system stability. Other than stability verification, the attractor of a chaotic system has also been used for designing secure, private multiuser digital communications systems. For example, Rohde et al. [9] proposed an algorithm based on Poincare's recurrence theorem for chaotic signal detection and estimation using sampled signal within the attractor of the underlying chaotic system. To obtain accurate detection result, a clear knowledge of the attractor is required so that the sampled signal patterns are valid and useful. Similar to the DoA of nonlinear systems, the attractor may be a complicated shape or even a set with non-integer dimension [10,11]. Using the invariant nature of attractors, estimation of an attractor can be carried out by flowing an initial set forward in time and observing how this initial set evolves. The mesh-based methods have been successfully applied to problems about estimating the (un)stable manifold of a point x_0 [11,12]. Although there are many different methods of computing the unstable manifold being proposed in this field, the basic idea remains the same; that is, to grow the (un)stable manifold from a local neighborhood around x_0. These methods differ in how they ensure a good mesh representation being computed during the process. Some methods are based on growing a geodesic circle around x_0 [13,14]. Guckenheimer and Holmes [13] grew the geodesic circles by modifying the vector fields. Krauskopf and Osinga [15], on the other hand, used the hyperplane techniques. Johnson et al. [16] used time parameterization based on trajectory arc length. A new geodesic circle was generated by interpolating the points evolving from the previous circle. However, it was hard to control the quality of the interpolation. The method used by Dellnitz and Junge [11] computes the outer approximation of the manifold by boxes. A sub-division algorithm was then applied to refine the mesh. All these mesh-based methods need fine mesh grids to obtain the result with acceptable resolution and the required storage space grows exponentially with the dimension of the manifold.

In this paper, we present an advection algorithm for the estimation of the attractor. Our approach presented in this paper uses 0-sub-level set of polynomials to represent a set of system states, and employs semi-definite programming (SDP) to perform the computation of advecting sets. The algorithm is iterative and proceeds by advecting the sub-level set of the polynomial under the flow map of polynomial systems. Unlike the mesh-based algorithms, the advection algorithm only needs to store the coefficients of the polynomials, which requires much smaller storage space than mesh-based methods. Using semi-algebraic representations also make it easy to be applied on-line to verify whether a point is in the attractor. Several numerical examples computed using the SOSCODE [17] toolbox for MATLAB are presented in this study.

The structure of this paper is as follows. In Section 2, we present the notations used in this study and provide a brief introduction to the sum-of-squares (SOS) techniques. Section 3 discusses the algorithm for attractor estimation as well as the related theoretical analysis. The discussion of the required computation time can be found in Section 4. Lastly, we conclude the paper in Section 5.

2. Sum-of-squares techniques for set advection

We used $\mathbb{R}[x]$ 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 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 used Σ to denote the set of all SOS polynomials in $\mathbb{R}[x]$. Also, we employed $\mathbb{R}_d[x]$ to denote the set of degree d polynomials in x with real coefficients and used Σ_d to denote the set of all SOS polynomials in $\mathbb{R}_d[x]$.

Suppose $g:\mathbb{R}^n \to \mathbb{R}$ is C^1. Let us define the 0-*sub-level-set* of g to as $\mathcal{C}(g) \subset \mathbb{R}^n$ given by $\mathcal{C}(g) = \{x \in \mathbb{R}^n \mid g(x) \le 0\}$. Further, let us 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. We will use $\mathbb{R}_+$ to represent the set of positive real numbers.

The following lemma is an important tool for formulating the proposed algorithm and is a special case of Putinar's Theorem [18].

Lemma 1. *Given $p,\ q \in \mathbb{R}[x]$, suppose there exist $s_0, s_1 \in \Sigma$ such that*

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

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, [19] or [20]. □

The representation shown in Lemma 1 is one of the simplest cases of Schmüdgen's Theorem [21], which states that if $p \in \mathbb{R}[x]$ is strictly positive inside a compact semi-algebraic set, S, generated by $p_1, \ldots, p_m$ as $S = \{x \in \mathbb{R}^n \mid p_i \ge 0,\ i = 1,2,\ldots,m\}$, then

$$p = \Sigma_v p_1^{v_1} \ldots p_m^{v_m} s_v,$$

where $v = (v_1, \ldots, v_m) \in \{0,1\}^m$ and $s_v \in \Sigma$. Putinar [18] later showed that under some additional constraints on p_i, p has a simpler representation as

$$p = s_0 + s_1 p_1 + \cdots + s_m p_m.$$

The gap between Schmüdgen and Putinar's representation was later investigated by Jacobi and Prestel [22]. In the simple case shown in Lemma 1, if $\mathcal{C}(q) \subset \mathcal{C}(p)$ and $\mathcal{C}(q)$ is compact, the representation of p by Eq. (1) is always possible

The following result is similar. Given $q \in \mathbb{R}[x]$, if there exists $s_0,\ s_1 \in \Sigma$, and $\epsilon \in \mathbb{R}_+$, such that

$$s_0 + s_1 q - p + \epsilon = 0,$$

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 pre-defined degree or passing through some pre-specified points. We have used the above-mentioned relationships to construct such constraints. This technique is frequently used in the proposed level-set algorithm.

2.1. Sum-of-squares polynomials as semi-definite programming problems

One of the benefits of the SOS techniques is that it can be formulated as an SDP problem. The following is a standard form of an SDP problem.

$$\begin{aligned} \min \quad & \operatorname{trace} CX \\ \text{s.t.} \quad & \operatorname{trace} A_i X = b_i \text{ for } i = 1, \ldots, m \\ & X \succcurlyeq 0, \end{aligned}$$

where $X \in \mathbb{R}^{n \times n}$ is symmetric. By $X \succcurlyeq 0$, we mean that $z^T X z$ is PSD for all $z \in \mathbb{R}^n$.

Instead of giving the details of the SOS techniques, we use the following example on how an SOS problem can be casted as an SDP problem.

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

$$s_1 = z^T O z, \quad s_2 = z^T P z$$
$$s_3 = z^T Q z, \quad s_4 = z^T R z,$$

where $z = [x\ 1]^T$ and $O, P, Q, R \in \mathbb{R}^{2\times 2}$. It should be noted that if $O, P, Q, R \succcurlyeq 0$, we can decompose s_1, s_2, s_3, s_4 as SOS polynomials [23].

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

$$s_5 - s_1(x-1) + s_2(x+1) + f = 0$$
$$s_3 + s_6(x-1) - f = 0$$
$$s_4 - s_7(x+1) - f = 0.$$

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

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

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

$$f = x^2 - 1 \quad s_1 = \tfrac{1}{2}x^2 + x + \tfrac{1}{2}$$
$$s_2 = \tfrac{1}{2}x^2 - x + \tfrac{1}{2} \quad s_3 = x^2 - 2x + 1$$
$$s_4 = x^2 + 2x + 1 \quad s_5 = 0$$
$$s_6 = 2 \quad s_7 = 2,$$

and $\mathcal{C}(f) = \{x \in \mathbb{R} \mid x+1 \geq 0,\ x-1 \leq 0\}$, 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.

2.2. *Set advection*

Consider the following autonomous system:

$$\dot{x}(t) = f(x), \tag{2}$$

where $f: \mathbb{R}^n \to \mathbb{R}^n$ is locally Lipschitz. The basic local existence and uniqueness theorem [24] states that given an open subset $U \in \mathbb{R}^n$, there exists $c \in \mathbb{R}_+$ such that the autonomous system (2) has a unique solution for any $z \in U$ in the compact time interval $[-c, c]$.

We define the *flow map* $\phi_t(z): \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ to be the local unique solution of

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

For any $t \in \mathbb{R}$ such that $\phi_t(z)$ exists, the map $\phi_t: \mathbb{R}^n \to \mathbb{R}^n$ is continuous, invertible and has a continuous inverse; that is, it is a topological homeomorphism on $\mathbb{R}^n$ [25].

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

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

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 2. Consider that g_1, and g_2 are functions mapping $\mathbb{R}^n$ to $\mathbb{R}$. If $g_2 = A_t g_1$, then $\mathcal{C}(g_2) = \phi_t(\mathcal{C}(g_1))$.

2.3. Time-stepping

To carry out advection, we must use an approximation to the flow map ϕ_h with time step h. 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

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

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

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

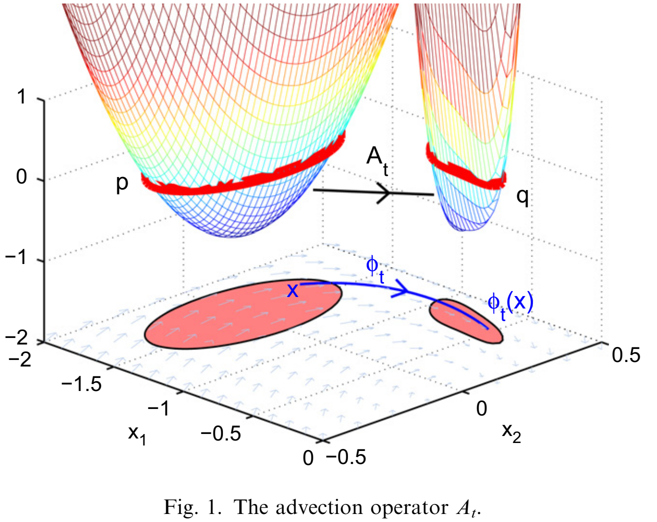

Fig. 1. The advection operator A_t.

Both these approximations have the following properties: They are both linear maps, and if p and f are polynomials, then so is q. In this study, we have concentrated on B_h, because typically, B_{hp} is a polynomial whose degree is less than that of C_hp.

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

2.4. The truncation error

Here, we have examined the bound of the truncation error. Only the truncation error for the first-order Taylor approximation is presented. The bound for higher order Taylor approximation can be easily derived using similar methods.

Proposition 3. *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*

$$\|B_hp - A_hp\| \leq \epsilon = K\frac{h^2}{2} \quad \text{for all } x \in M.$$

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

Proof. Using the 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

$$A_hp(x) = p(x) - hDp(x)f(x) + L_f^2p(y)\frac{h^2}{2} = B_hp(x) + L_f^2p(y)\frac{h^2}{2},$$

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

$$\|L_f^2p(y)\|\frac{h^2}{2} = \|L_f^2p(y) - L_f^2p(z) + L_f^2p(z)\|\frac{h^2}{2} \leq Nr\frac{h^2}{2} + \|L_f^2p(z)\|\frac{h^2}{2}.$$

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

There are several ways to deal with the truncation error. For example, the upper bound of $\|L_f^2p(x)\|$ for all $x \in \overline{M}$ can be directly estimated by solving an optimization problem. This allows us to monitor the growth of the truncation error.

Without further constraints, the truncation error may generate $\mathcal{C}(p_{k+1})$ with disconnected sub-level sets outside M. We have used various constraints to deal with this problem.

3. Estimation of the attractors

3.1. Attracting sets

The following provides the definition of the attractor. Similar definitions can be found in [26,24].

Definition 4 (*Attracting Set*). Consider that f is analytic with flow map, ϕ. A closed invariant set $\mathbb{A} \subset \mathbb{R}^n$ is called an *attracting set* under the flow map, ϕ, if there exist some neighborhood U of $\mathbb{A}$ in which $\phi_t(x)$ is defined for all $t > 0$ and $x \in U$ such that

- $\phi_t U \subset U$ for all $t \geq 0$,
- $\bigcap_{t>0} \phi_t U = \mathbb{A}$.

Definition 5 (*Attractor*). The attractor of f is the indecomposable attracting set of f. Moreover, the minimal attractor, $\mathcal{A}$, is the attractor for which no proper subset of $\mathcal{A}$ is an attractor.

For each attractor, its DoA is given by

$$\bigcup_{t\leq 0} \phi_t U,$$

for any open set U such that $\phi_t(x) \in \mathcal{A}$ for $t \to \infty$ for all $x \in U$. For each attractor in a system, there exists an associated DoA. If the DoA for a given attractor is the entire $\mathbb{R}^n$, then the attractor is called the global attractor. Depending on the characteristics of a system, the attractor could be points, curves, or complicated sets. For a globally stable system, the equilibrium point is the minimal attractor and $\mathbb{R}^n$ is the associated DoA.

The polynomial level-set method has been successfully applied to the estimation of DoA [1]. In the estimation of DoA, we advect an positively invariant set backward in time with the help of star-shaped constraints to avoid the problems caused by truncation errors. For estimating the attractor, some modification to the polynomial level-set method are required, since the star-shaped constraint may be too restrictive. For the case where the DoA is a subset of $\mathbb{R}^n$, the level-set method can also be applied if we choose the initial set properly. For simplicity, we will consider the global attractor in this paper. Also, we will only discuss the case where the attractor is bounded.

The proposed algorithm is based on the convergence property given in the following.

Lemma 6. *Consider that f is analytic and S_1 is a connected closed positively invariant set. Let $h \in \mathbb{R}_+$ and define the forwards advection ofS_1 to be S_2, given by*

$$S_2 = \phi_h(S_1).$$

Then $S_1 \supset S_2$, and S_2 is also connected, closed and positively invariant. Further, $\partial S_2 = \phi_h(\partial S_1)$.

Proof. Firstly, we will show $S_2 \subset S_1$. Since S_1 is positively invariant, we have

$$S_2 = \phi_h(S_1) \subset S_1,$$

as desired. To show positive invariance of S_2, notice that for any $t \geq 0$

$$\phi_{-h}(\phi_t(S_2)) = \phi_t(\phi_{-h}(S_2)) = \phi_t(S_1) \subset S_1,$$

since S_1 is positively invariant. Taking ϕ_h of both sides, we have $\phi_t(S_2) \subset S_2$ as desired.

Finally, connectedness, closedness and preservation of the boundary follow because ϕ_h is a topological homeomorphism on $\mathbb{R}^n$. □

Theorem 7. *Consider that f is analytic and $\mathcal{A}$ is a bounded minimal attractor. Also, consider that $S_0 \supset \mathcal{A}$ is connected closed positively invariant and S_0 contains only one minimal attractor, $\mathcal{A}$, and is contained in the DoA of $\mathcal{A}$. Furthermore, suppose $\phi_t(x)$ is defined for all*

$t>0$ and $x \in S_0$. If the sequence $S_0, S_1, S_2, \ldots$ is generated such that

$$S_{i+1} = \phi_h S_i \quad \text{for } i = 0,1,2,\ldots,$$

for some $h \in \mathbb{R}_+$, then the sequence converges to $\mathcal{A}$.

Proof. From Lemma 6, we have

$$S_0 \supset S_1 \supset S_2 \supset \cdots.$$

By Definition 5, every point in the DoA eventually moves inside an arbitrary ϵ-neighborhood of $\mathcal{A}$. Since S_0 is located inside the DoA of $\mathcal{A}$, from Lemma 6, the boundary points of S_i will move inside an arbitrary ϵ-neighborhood of $\mathcal{A}$ for i sufficiently large. Therefore, the sequence $S_1, S_2, S_3, \ldots$ converges to the minimal attractor $\mathcal{A}$. $\square$

In our previous work [1], we used the star-shaped constraints to reduce the problem generated from the truncation error. In the case of estimating the attractor, star-shaped constraints may be restrictive because the shape of an attractor is usually not a star-shaped set. Therefore, we should drop the star-shaped constraints for the algorithm. The main purpose of the star-shaped constraint is to ensure that the truncation error does not generate disconnected sub-level-sets. In the case of attractor estimation, similar property can be enforced by using $S_{i+1} \subseteq S_i$. We have the following forward advection algorithm.

3.2. Forward advection algorithm without star-shaped constraint

Given a polynomial g_{i-1} such that $\mathcal{A} \subset \mathcal{C}(g_{i-1})$, and $\mathcal{C}(g_{i-1})$ is connected, closed and positively invariant, we computed a polynomial g_i such that $\mathcal{C}(A_{-h}g_i)$ is approximately $\mathcal{C}(g_{i-1})$ as follows.

$\gamma \in \mathcal{A}$, $\alpha>0$, and positive integers $d,\tilde{d}$ are picked. Then, using SDP, the following feasibility problem for $g_i \in \mathbb{R}_d[x]$, $s_1,\ldots,s_6 \in \Sigma_{\tilde{d}}$ is solved:

$$\begin{aligned} &g_i(\gamma) = -1 \\ &s_1 - s_2 g_{i-1} + B_{-h} g_i = 0 \\ &s_3 + s_4 g_{i-1} - B_{-(h-\alpha)} g_i = 0 \\ &s_5 + s_6 g_{i-1} - g_i = 0. \end{aligned} \tag{3}$$

Here, an important parameter α is introduced. The above-mentioned algorithm finds a degree d polynomial g_i such that $\phi_{-h}\mathcal{C}(g_i) \supset \mathcal{C}(g_{i-1})$, and $\phi_{-h-\alpha}\mathcal{C}(g_i) \subset \mathcal{C}(g_{i-1})$. Hence the parameter α may be considered as a tolerance that allows for the constraint that g_i is required to have degree d or less. The second and the third constraints are these approximation constrains for g_i. The fourth constraint ensures that $\mathcal{C}(g_i) \subseteq \mathcal{C}(g_{i-1})$.

To guarantee that $\mathcal{C}(g_i)$ contains the attractor, we use the following relationships. Consider that the truncation error is e. We can obtain $\overline{e}$ as the estimated upper bound of $|e|$ around $\mathcal{C}(g_{i-1})$ using SDP and Proposition 3. We have

$$\mathcal{C}(A_{-h}(g_i - \overline{e})) = \mathcal{C}(B_{-h}g_i + e - \overline{e}) \supseteq \mathcal{C}(B_{-h}g_i) \supseteq \mathcal{C}(g_{i-1}) \supseteq \mathcal{A}.$$

Hence, $\phi_h \mathcal{C}(A_{-h}(g_i - \overline{e})) = \mathcal{C}(g_i - \overline{e}) \supseteq \phi_h \mathcal{C}(g_{i-1}) \supseteq \mathcal{A}$. By subtracting g_i with $\overline{e}$, the generated semi-algebraic set is guaranteed to contain the attractor.

3.3. Finding the initial semi-algebraic set

Before applying the proposed method, we need to start the algorithm with a positively invariant set. This can be done by searching for a feasible Lyapunov function in $\mathbb{R}^n \backslash \mathcal{A}$. Let $p \in \mathbb{R}[x]$ be chosen such that $\mathcal{A} \subset \mathcal{C}(p)$. Then, the following convex feasibility problem is solved.

Find $V \in \mathbb{R}[x]$ and $s_0, s_1 \in \Sigma$ such that

$$\begin{aligned} &DV(x)x > 0 \quad \text{for all } x \neq 0 \\ &V(x) > 0 \quad \text{for all } x \neq 0 \\ &V(0) = 0 \\ &DV(x)f(x) + s_0 + s_1 p = 0 \quad \text{for all } x \neq 0. \end{aligned}$$

Then, the smallest γ sub-level set of V that contains $\mathcal{C}(p)$ gives us the smallest initial estimation of the attractor. It should be noted that this method uses the star-shaped constraint, which is the first constraint. Therefore, to apply the above method, we need first make some coordinate transformation to place the origin inside the initial attracting set.

3.4. Stopping condition

By using the proposed level-set method, one can successfully propagate the system states forward in time. However, a stopping criterion is still needed to terminate the iterations. To detect the convergence of the advected sets, the closeness of two semi-algebraic sets is analyzed. The following shows that the closeness of two semi-algebraic sets can be estimated by using scaled sets.

Proposition 8. *Given $g \in \mathbb{R}[x]$, solve $s_0 \in \Sigma, h_o \in \mathbb{R}[x], b \in \mathbb{R}_+$ of the following optimization problem.*

$$\begin{aligned} &\max b \\ &\text{s.t.} \quad \|Dg_k\|^2 - b^2 = s_0 + h_0 g_k. \end{aligned}$$

Then $\|Dg_k\| \geq b$ for all $x \in \mathcal{V}g_k$.

Proof. Given any $x_0 \in \mathcal{V}(g_k)$, the right-hand side of the constraint is PSD. Therefore, $\|Dg_k\| \geq b$. □

Remark 9. Suppose $\mathcal{C}(g_1) \supset \mathcal{C}(g_2)$ and $q \in \mathbb{R}[x]$ is constructed such that $\mathcal{C}(q) \subset \mathcal{C}(g_1)$ and the normal distance between the boundaries of $\mathcal{C}(g_1)$ and $\mathcal{C}(q)$ is less than ϵ. Then, if $\mathcal{C}(q) \subset \mathcal{C}(g_2) \subset \mathcal{C}(g_1)$, the normal distance between the boundaries of $\mathcal{C}(g_1)$ and $\mathcal{C}(g_2)$ is less than ϵ.

To use the result of Remark 9, the user specifies an ϵ and generate the scaled semi-algebraic set to test whether the result of the next advected set is close enough to stop the iteration.

The scaled version of the $\mathcal{C}(g_1)$ can be constructed approximately using Proposition 8 in the following way. Suppose $\|Dg_i(x)\| > b > 0$ for all $x \in \mathcal{V}(g_i)$. Let $x \in \mathcal{V}(g_i)$; then, the normal vector of g_i at x is

$$n_x = \frac{Dg_i}{\|Dg_i\|}.$$

For any $x \in \mathcal{V}(g_i)$ and a small $\delta \in \mathbb{R}_+$, there exists $\alpha_x > 0$ such that $x - \alpha_x n_x \in \mathcal{V}(g_i + \delta)$. Equivalently, we have $g_i(x - \alpha_x n_x) = -\delta$. The value of α_x is then the normal distance between the boundary of $\mathcal{C}(g_i)$ and $\mathcal{C}(g_i+\delta)$. We can approximate this relationship by first-order Taylor approximation as

$$g_i(x) - \langle Dg_i(x), \alpha_x n_x \rangle \approx -\delta$$
$$\alpha_x \|Dg_i(x)\| \approx \delta$$
$$\alpha_x \approx \frac{\delta}{\|Dg_i(x)\|}.$$

Now, we would like $\alpha_x \leq \epsilon$ for all $x \in \mathcal{V}g_i$. Let $\delta = \epsilon b$. Then, the normal distance between $\mathcal{V}(g_i+\delta)$ and $\mathcal{V}(g_i)$ is approximately less than ϵ because

$$\alpha_x \approx \frac{\delta}{\|Dg_i(x)\|} = \frac{\epsilon b}{\|Dg_i(x)\|} \leq \epsilon.$$

It should be noted that the above-mentioned approximate method fails if the lower bound of $\|Dg_i(x)\|$ is zero. In such case, we could expand $\mathcal{C}(g_i)$ by adding a small positive value γ from g_i and then apply the above process again.

3.5. Examples of attractor estimation

Example 2. Consider the following dynamical system,

$$\dot{x} = y$$
$$\dot{y} = x - y - x^3.$$

This system is derived from Duffing's equation [24] without external forcing. The origin is an unstable equilibrium point. The actual minimal attractor for the system is the other two equilibrium points, (1,0),(−1,0). Fig. 2 shows the result of this example. The first few iteration results are shown as thin solid curves. The thick solid curve indicates the result after 100 iterations. The iteration stops because the decrement rate of the estimated attractor is too slow. Several system trajectories are also shown as dashed curves.

The solution obtained after 100 iterations is

$$\begin{aligned} g = {} & -641 + 2x - 5y + 49{,}039y^2 + 115y^3 + 397{,}279y^4 + 1{,}022y^5 + 64{,}226y^6 \\ & -41{,}291xy - 148xy^2 - 248{,}650xy^3 - 362xy^4 - 11{,}562xy^5 + 10{,}969x^2 \\ & +17x^2y - 102{,}496x^2y^2 - 18x^2y^3 + 10{,}911x^2y^4 + 12x^3 + 83{,}659x^3y \\ & +51x^3y^2 + 98{,}766x^3y^3 - 20{,}780x^4 + 4x^4y + 66{,}980x^4y^2 - 13x^5 \\ & -40{,}639x^5y + 10{,}000x^6. \end{aligned}$$

It can be noticed that this system contains two minimal attractors, −1 and 1. In this case, the level-set method does not give us the minimal attractor since the initial set contains both the minimal attractors. With a more cleverly chosen initial set, S_0, such that S_0 is contained inside one of the minimal attractor's DoA, the proposed algorithm will give us the estimation of the attractor instead of giving us the estimation of the collection of the attractors.

Example 3. The Lorenz system [10] is a classical system of a chaotic attractor.

$$\dot{\overline{x}} = \sigma(\overline{y} - \overline{x})$$
$$\dot{\overline{y}} = \overline{x}(\rho - \overline{z}) - \overline{y}$$
$$\dot{\overline{z}} = \overline{x}\,\overline{y} - \beta \overline{z}.$$

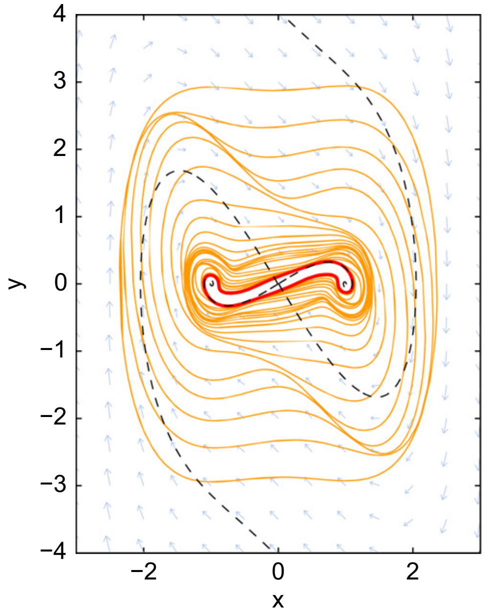

Fig. 2. The result of Example 6.

σ is called the Prandtl number and ρ is the Rayleigh number. The most interesting case is probably the combination of $(\sigma,\rho,\beta)=(10,8/3,28)$, where the attractor is butterfly shaped. The chaotic behavior of the system makes it difficult to obtain a description of the attractor. In this example, we will try to get an estimation of the butterfly shaped attractor of the system.

First, a few coordinate changes are made by letting

$$50x=\overline{x}$$
$$50y=\overline{y}$$
$$50z=(\overline{z}-25)$$

so that the attractor is located at the center of the unit box.

After obtaining an invariant set covering the attractor, we start using the forward advection algorithm to estimate the attractor. We stop the algorithm after 316 iterations because the decrement rate is less than desired. The rounded solution is as follows:

$$\begin{aligned} g = & -6-2xy^4z+21z-7x^5z+38z^2-4892y^2z+397y^2+4366y^4 \\ & +8369y^6-970xy-30{,}911xy^3+11{,}027y^2z^2-13{,}481x^2z+xy^4 \\ & +16{,}309xyz-56{,}320xyz^2+32{,}093xyz^3-58{,}767xy^5+608x^2 \\ & +77{,}863x^2y^2-5x^2y^3+254{,}344x^2y^4+7418y^2z^3-88{,}675x^3y \\ & +9x^3y^2-3x^2yz^2-x^2yz^3-680{,}076x^3y^3+39{,}852x^4-9x^4y \\ & -78{,}951x^2z^3+1{,}009{,}580x^4y^2+3x^5-77{,}0871x^5y+248{,}581x^6 \\ & -49{,}319x^2z^4-24{,}907xy^3z^2-697z^3+7464z^5+64{,}744x^2z^2 \\ & -24{,}900y^4z+21{,}137y^2z^4-2xy^2z+164{,}574xy^3z+6562xyz^4 \\ & -412{,}196x^2y^2z+42{,}644x^2y^2z^2+3x^2yz+2xy^2z^2+xy^2z^3 \end{aligned}$$

$$
\begin{aligned}
&-256{,}376x^4z+245{,}304x^4z^2+2x^3z^3-15x^3y^2z+15x^4yz\\
&-213{,}402x^3yz^2-2x^3z+496{,}094x^3yz+8x^2y^3z+10{,}471z^6\\
&+15{,}529y^4z^2-255z^4+x^3z^2.
\end{aligned}
$$

The result is then transformed back to the original coordinate, and is plotted in Fig. 3. The shape of the estimated attractor is shown by thin curves that represent several $x-z$ cross-sections of the set. The thick solid curves are the system trajectories which are used to

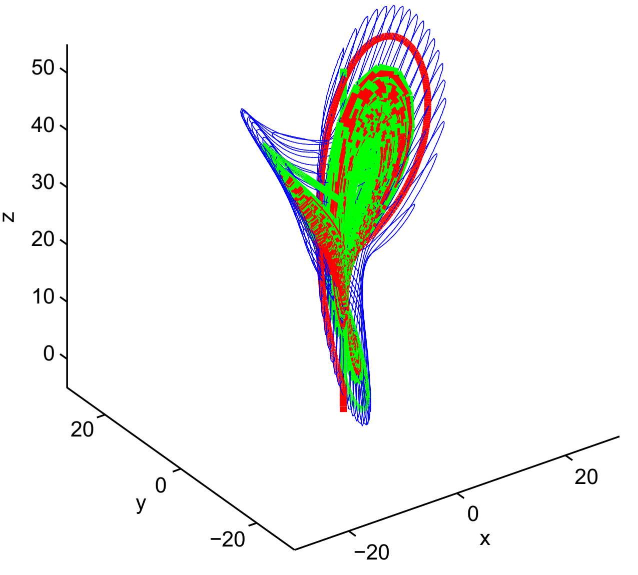

Fig. 3. Three-dimensional view of the estimation of the Lorenz attractor.

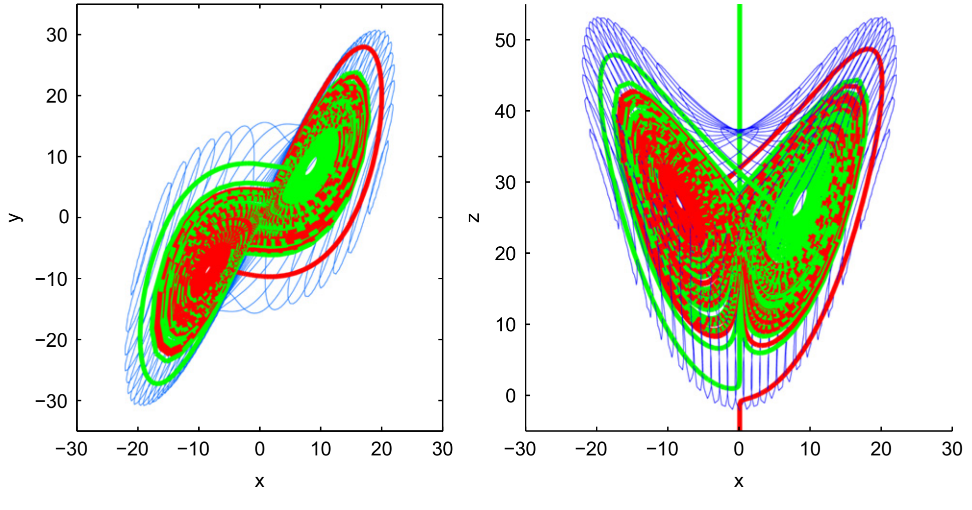

Fig. 4. The $x-y$ and $x-z$ view of the estimation of the Lorenz attractor.

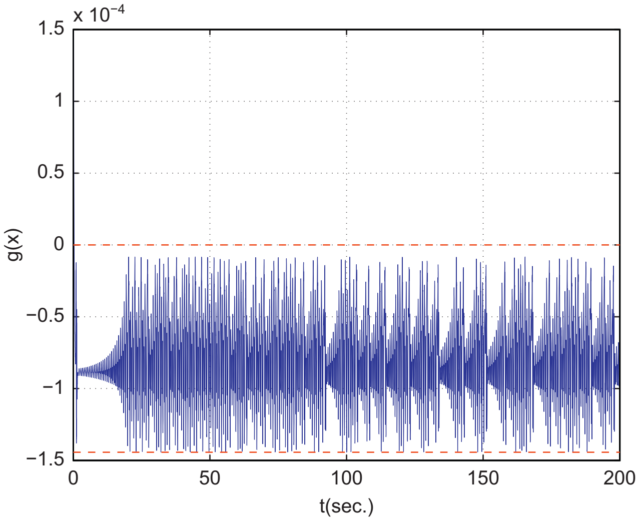

Fig. 5. Checking the invariant property of the estimated attractor. The upper and lower bounds of $g(x)$ in the attractor are shown with dashed lines.

show the shape of the attractor. Fig. 4 shows the $x-y$ and $x-z$ plots of the result. The chaotic behavior of the attractor makes it hard to continue the advection process. However, the proposed method successfully shows the butterfly shape of the attractor.

The invariant property of the result can be easily verified by evaluating the value of g along system trajectories. The simulated result is shown in Fig. 5. After the trajectory enters the attractor, the largest value along the trajectory is -8.2546×10^{-6}. Therefore, the estimated attractor is indeed an invariant set. The global lower bound of g is -1.4342×10^{-4} which can be easily found by SOS techniques.

4. Computational cost

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

There are different strategies to compute the matrices of the equivalent SOS SDP problem. SOSCODE [17] and SOSTOOLS [23] are two MATLAB toolboxes available for solving SOS problems. SOSTOOLS relies on the MATLAB symbolic toolbox and has a friendly user interface which makes it a very good tool for initial testing purposes. SOSTOOLS also hides most of the inner subroutines from the users and carries out numerous conversion works. This feature makes the users to have fewer controls on the formulation of the SOS problems. SOSCODE supplies users with necessary subroutines to compute the required transformation matrices. However, users need to have a clear knowledge of the details of the preparation process. The benefits of SOSCODE are that the users have more control of the matrices-generating process and can apply more sophisticated constraints on the SOS problems. The advection algorithms are better suited for using SOSCODE.

Table 1
Timing and memory usage of different examples.

Example	Dim. of A	Size of A (KBytes)	T_1 (s)	T_2 (s)	No. of Coeff. in g
(2)	$529 \times 10{,}101$	944	1.552	5.2	66
(3)	$2908 \times 91{,}168$	12,050	29.23	150	84

In the following tables, T_1 represents the time used to compute the matrices of the associated SDP problem and T_2 represents the time used to solve the SDP problem using SeDuMi [27]. The following tables are the timing information generated on a machine with one 3.4 GHz Pentium 4 CPU and 2 GB memory, running Fedora Core 5.

Table 1 shows some numerical information on the computational costs of the two examples used in this study. As the dimension and degree of the SOS polynomials increase, the required time to carry out the advections also increase. It must be noted that the SDP solver usually does not scale very well as the size of the problem increases. For problems with high dimensions and degrees, most of the time needed for computation is spent on solving the SDP. From our experience, increasing the degree of g does not have a great impact on the computational time. However, increasing the degrees of the SOS polynomials does have a great impact on the loading of the computation.

One way to speed up the computation speed is to use the full-degree ordering of the monomials. A degree d full-degree polynomial, $y(x)$, with m variables is the linear combination of monomials with degrees in each variable not greater than d. That is

$$y = \sum a_i x_1^{d_{i_1}} x_2^{d_{i_2}} \dots x_m^{d_{im}} \quad 0 \le d_{i_j} \le d.$$

Then, using lexicographic ordering of the monomials, one could represent the polynomial as a vector of the coefficients. The position of the coefficients in the vector can be easily calculated by

$$i = \sum_{j=1}^{m} d_{i_j} d^{m-j}.$$

By using the full-degree representation, the transformation matrices exhibit highly structured features, which let them to be easily constructed without excessive monomial comparisons. The derivation process is straightforward and interested readers should be able to do it without problem. Several examples have shown that the code using the full-degree representation runs more than ten times faster than that using SOSCODE.

On the other hand, using the full-degree representation makes the size of the SDP much larger. This could greatly slow down the SDP solving process. Therefore, a post processing is required to remove the extra terms of monomials that have combined degree greater than d. Fortunately, post processing requires very few computational power. Some speed comparisons between the SOSCODE and the full degree code for solving Example 3 are listed in Table 2.

In Table 2, T_1 represents the time required for SOSCODE to generate the matrices and T_1^* represents the time required when using full-degree representation code. After post processing, the generated SDPs from both the approaches are found to be identical. Therefore, the amount of time required to solve the SDPs are the same for both the approaches. It can be

Table 2
Speed comparison between SOSCODE and full-degree code.

Deg. of g	Deg. of Ref.	Dim. of A	T_1 (s)	T_1^* (s)	T_2 (s)
6	16	$2908 \times 91{,}168$	29.23	2.75	150
8	18	$3991 \times 166{,}534$	56.93	4.20	374
10	20	$5314 \times 288{,}875$	105.30	5.75	946

clearly seen in Table 2 that when the problem gets larger, the benefits of using the full-degree code are more prominent.

The time used for solving the SDP relies on the capability of the SDP solver. SeDuMi [27] 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. Another approach is to take advantage of the special structure of the generated SDP problem and develop a specialized solver. This could be a good topic for future research.

From the above-mentioned discussion, it can be concluded that the time required for carrying out the advection is not fast enough for real-time applications. However, once the advected set is computed, it can be very easily used to test whether a point is in the attractor.

5. Conclusions

In this study, a level-set algorithm to advect invariant subsets of the state space using SDP has been presented. The sets have been represented as semi-algebraic sets. The proposed algorithm generates a polynomial whose 0-sub-level-set approximately represents the advected set. An algorithm has been proposed for estimating the minimal attractor of a system. This level-set algorithm not only works for two-dimensional systems, but also works for higher dimensional systems.

There are several open problems about the level-set method. Although Schmüdgen's theorem provides the theoretical support for the existence of solutions, it does not give us the upper bound of the degree of the polynomials used in such representation. Hence, when the algorithm fails, it might be due to insufficient degree in the SOS polynomials or the polynomial used to represent the advected set. Incorrect addition of the degree will lead to increase in the problem size and one may not be able to yield a feasible solution. A more careful study regarding the sufficiency of the degree is required in the future work. The computation process relies on the power of the SDP solver used. The fast growth of the computation time for an SDP solver when the problem size increases, makes it difficult for the algorithm to be applied for higher dimensional systems. Thus, determining whether it is possible to utilize the SOS structure of the algorithm to speed up the solution process would be a good direction for future research.

References

[1] T.-C. Wang, S. Lall, T.-Y. Chiou, Polynomial method for pll controller optimization, Sensors 11 (7) (2011) 6575–6592, http://dx.doi.org/10.3390/s110706575. URL ⟨http://www.mdpi.com/1424-8220/11/7/6575/⟩.

[2] M. Margaliot, G. Langholz, Necessary and sufficient conditions for absolute stability: the case of second-order systems, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 50 (2) (2003) 227–234, http://dx.doi.org/10.1109/TCSI.2002.808219.
[3] M. Margaliot, Stability analysis of switched systems using variational principles: an introduction, Automatica 42 (12) (2006) 2059–2077.
[4] R. Shorten, F. Wirth, O. Mason, K. Wulff, C. King, Stability criteria for switched and hybrid systems, SIAM Review 49 (4) (2007) 545–592.
[5] M. Mahmood, P. Mhaskar, On constructing constrained control Lyapunov functions for linear systems, IEEE Transactions on Automatic Control 56 (5) (2011) 1136–1140, http://dx.doi.org/10.1109/TAC.2011.2114470.
[6] D. Mayne, S. Rakovi, R. Findeisen, F. Allgower, Robust output feedback model predictive control of constrained linear systems, Automatica 42 (7) (2006) 1217–1222.
[7] A. Polyakov, A. Poznyak, Invariant ellipsoid method for minimization of unmatched disturbances effects in sliding mode control, Automatica 47 (7) (2011) 1450–1454, http://dx.doi.org/10.1016/j.automatica.2011.02.013.
[8] F. Blanchini, S. Miani, M. Sznaier, Robust performance with fixed and worst-case signals for uncertain time-varying systems, Automatica 33 (12) (1997) 2183–2189, http://dx.doi.org/10.1016/S0005-1098(97)00136-2.
[9] G.Rohde, J. Nichols, F. Bucholtz, Chaotic signal detection and estimation based on attractor sets: applications to secure communications, Chaos 18(1) 013114 (2008), http://dx.doi.org/10.1063/1.2838853.
[10] E. Lorenz, Deterministic nonperiodic flows, Journal of Atmospheric Science 357 (1963) 130–141.
[11] M. Dellnitz, O. Junge, Set oriented numerical methods for dynamical systems, in: Handbook of Dynamical Systems II: Toward Applications. World Scientific, 2002, pp. 221–264.
[12] B. Krauskopf, H. Osinga, E. Doedel, M. Henderson, J. Guckenheimer, A. Vladimirsky, M. Dellnitz, O. Junge, A survey of methods for computing (un)stable manifolds of vector fields, International Journal of Bifurcation and Chaos 15 (3) (2005) 763–792.
[13] J. Guckenheimer, P. Holmes, Dynamical Systems: Some Computational Problems, Kluwer Academic Publishers, 1993.
[14] B. Krauskopf, H.M. Osinga, The Lorenz manifold as a collection of geodesic level sets, Nonlinearity 17 (1) (2004) C1–C6.
[15] B. Krauskopf, H. Osinga, Computing geodesic level sets on global (un)stable manifolds of vector fields, SIAM Journal on Applied Dynamical Systems 4 (2) (2003) 546–569.
[16] M. Johnson, M. Jolly, I. Kevrekidis, Two-dimensional invariant manifolds an global bifurcations: some approximation and visualization studies, Numerical Algorithms 14 (1–3) (1997) 125–140.
[17] S. Lall, M. Peet, T. Wang, SOSCODE : Sum of Square Optimization Toolbox for MATLAB 2005. Available by email: tachung@mail.ncku.edu.tw.
[18] M. Putinar, Positive polynomials on compact semi-algebraic sets, Indiana University Mathematics Journal 42 (3) (1993) 969–984.
[19] P. Parrilo, Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization, Ph.D. Thesis, California Institute of Technology (2000).
[20] P.A. Parrilo, S. Lall, Semidefinite programming relaxations and algebraic optimization in control, European Journal of Control 9 (2–3) (2003) 307–321.
[21] K. Schmüdgen, The *K*-moment problem for compact semi-algebraic sets, Mathematische Annalen 289 (1991) 203–206.
[22] T. Jacobi, A. Prestel, Distinguished representations of strictly positive polynomials, Journal für die reine und Angewandte Mathematik 532 (2001) 223–235.
[23] S. Prajna, A. Papachristodoulou, P.A. Parrilo, Introducing SOSTOOLS: A general purpose sum of squares programming solver, in: Proceedings of the 41st IEEE Conference on Decision and Control, 2002, pp. 741–746.
[24] J. Guckenheimer, P. Holmes, Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields, Springer-Verlag, New York Inc., 1983.
[25] H. Khalil, Nonlinear Systems, 3rd ed., Prentice-Hall Inc., 2000.
[26] S. Wiggins, Introduction to Applied Nonlinear Dynamical Systems and Chaos, 2nd ed., Springer Science+Business Media, Inc., 2003.
[27] J.F. Sturm, SeDuMi: A MATLAB Toolbox for Optimization over Symmetric Cones. Software, User's Guide and Benchmarks. Version 1.03 (September 1999). URL ⟨http://sedumi.mcmaster.ca⟩.