Hybridization Domain Construction using Curvature Estimation*

Thao Dang
CNRS/VERIMAG
Centre Equation, 2 av de Vignate
38610 Gières, France
Thao.Dang@imag.fr

Romain Testylier
VERIMAG
Centre Equation, 2 av de Vignate
38610 Gières, France
Romain.Testylier@imag.fr

ABSTRACT

This paper is concerned with the reachability computation for non-linear systems using hybridization. The main idea of hybridization is to approximate a non-linear vector field by a piecewise-affine one. The piecewise-affine vector field is defined by building around the set of current states of the system a simplicial domain and using linear interpolation over its vertices. To achieve a good time-efficiency and accuracy of the reachability computation on the approximate system, it is important to find a simplicial domain which, on one hand, is as large as possible and, on the other hand, guarantees a small interpolation error. In our previous work [8], we proposed a method for constructing hybridization domains based on the curvature of the dynamics and showed how the method can be applied to quadratic systems. In this paper we pursue this work further and present two main results. First, we prove an optimality property of the domain construction method for a class of quadratic systems. Second, we propose an algorithm of curvature estimation for more general non-linear systems with non-constant Hessian matrices. This estimation can then be used to determine efficient hybridization domains. We also describe some experimental results to illustrate the main ideas of the algorithm as well as its performance.

Categories and Subject Descriptors

G.1 [**Numerical Analysis**]: Initial value problems

General Terms

Algorithms, Reliability, Verification

Keywords

Hybrid systems, formal verification, reachability analysis

*Research supported by ANR project VEDECY.

HSCC'11, April 12–14, 2011, Chicago, Illinois, USA.

1. INTRODUCTION

Hybrid systems are systems that combine continuous and discrete dynamics and can be used as a mathematical model as well as a computation methodology for many important application domains, such as embedded and cyber-physical systems. Reachability analysis is a central problem in verification and synthesis. For these reasons, numerous reachability computation techniques for hybrid systems have been developed (for example, [12, 2, 17, 5, 20, 16, 7, 15, 14, 11]). In particular, the recently-developed techniques specialized for linear dynamical systems [10, 15, 1, 11] can handle high dimensional systems. Nevertheless, reachability computation for non-linear systems is still a challenging problem. *Hybridization* is an approach to analyze non-linear systems by approximating it by a system with simpler dynamics, such as piecewise-constant or piecewise-affine, which can be handled more efficiently. This idea was first proposed in [9] for numerical simulation of large non-linear systems. It was later adapted to reachability computation in [3, 4, 6]. The principle of hybridization is rather simple: in order to analyze the behavior of a system from some states, one constructs a domain, which we call "hybridization domain", around these states, and in this domain the original non-linear derivative function f is approximated by a simpler, such as affine, function l and an error set U which accounts for the approximation error, that is

$$\forall x \in \Delta \ \exists u \in U \text{ s.t. } f(x) = l(x) + u. \tag{1}$$

Then, the system

$$\dot{x} = Ax + u$$

where u is the uncertain input taking values in U can be used as a conservative approximation of the original non-linear system inside Δ. By "conservative approximation", we mean that any trajectory of the original system is also a trajectory of the approximate system. Other properties of this approximation, such as convergence and qualitative behavior preservation, are described in [4]. Various implementations of hybridization have been developed and used for interesting examples, such as biological systems [6, 8]. While affine dynamics can be handled by relatively efficient methods, the application of the hybridization approach to non-linear systems is still limited by the difficulty in determining good hybridization domains so that the error in the approximate dynamics is not too large. Indeed, large errors in the dynamics approximation might reduce significantly the computation speed since this requires small time steps or a large number of small hyrbidization domains. In our previous work [8] we proposed a method for constructing

efficient hybridization domains using the curvature of the dynamics and showed how the method can be applied to quadratic systems. In this paper we pursue this work further and present two main results. First, we prove an optimality property of the domain construction method for a class of quadratic systems. Second, we propose an algorithm of curvature estimation for more general non-linear systems with non-constant Hessian matrices. This estimation can then be used to determine efficient hybridization domains.

Before presenting our results, we summarize the basic principles of hybridization and its use in the computation of reachable sets of non-linear systems. We also emphasize that although this paper addresses only continuous systems, its extension to hybrid systems is direct. Indeed, the underlying techniques we use to handle approximate piecewise-affine systems generated by hybridization are based on the techniques for hybrid systems [2].

2. HYBRIDIZATION: BASIC IDEAS

In this section, we describe affine hybridization where the approximate vector field in each domain is affine. The use of piecewise-affine systems is motivated by a large choice of available methods for their verification (see for instance [2, 5, 16, 15, 11]). However, other classes of functions can be used, such as constant or multi-affine.

2.1 Principle

We consider a non-linear system:

$$\dot{x}(t) = f(x(t)),\ x \in \mathcal{X} \subseteq \mathbb{R}^n. \tag{2}$$

where the function f is Lipschitz over the state space $\mathcal{X}$.

Given a set of all current states of the system, the trajectories from which we want to explore, we construct a domain which is a neighborhood of this set and then assign an affine vector field to that domain. When the trajectories of the system leave the domain, a new domain is created. This way the non-linear system is approximated by a piecewise-affine system.

In this work, to define approximate affine vector fields, we use linear interpolation over simplicial domains. An important advantage of this approximation method is that using the vertices of a simplex, the affine interpolant is uniquely determined, since each simplex has exactly $(n+1)$ vertices.

We denote by $\Delta \subseteq \mathbb{R}^n$ a simplex[1] which is used as a hybridization domain. The approximate affine system associated with Δ is defined as follows:

$$\dot{x}(t) = s(x(t), u(t)) = l(x(t)) + u(t) \tag{3}$$

where $x \in \Delta$, $u(\cdot) \in \mathcal{U}_\mu$; l is an affine map of the form:

$$l(x) = Ax + b$$

where A is a matrix of size $n \times n$ and $b \in \mathbb{R}^n$ such that l interpolates the function f at the vertices of Δ, that is for each vertex v of Δ we have $l(v) = f(v)$. The input u is introduced in order to account for the approximation error. More concretely, let μ be the bound of $||f - l||$, i.e. for all $x \in \Delta$

$$||f(x) - l(x)|| \leq \mu$$

[1]We recall that a simplex in $\mathbb{R}^n$ is the convex hull of $(n+1)$ affinely independent points in $\mathbb{R}^n$.

where $||\cdot||$ is some norm on $\mathbb{R}^n$. In this work we will consider the norm $||\cdot||_\infty$ which is defined as

$$||x||_\infty = \max(|x_1|, \ldots, |x_n|).$$

Thus, the error set U contains all the points $u \in \mathbb{R}^n$ such that $||u|| \leq \mu$. The set $\mathcal{U}_\mu$ is the set of piecewise-continuous input functions of the form $u : \mathbb{R}^+ \to U$.

To determine the error set U, one needs to estimate the error bound μ which depends on the domain geometry and size. It is however important to note that determining the exact maximal error μ is difficult. Suppose that we can find an upper bound of μ, denoted by $\overline{\mu}$. Then, we can choose the error set U to be the ball (thats is a hypercube for the infinity norm) centered at the origin and has radius $\overline{\mu}$. This will be discussed in detail later.

To define the reachable set of the system (3), let $\Phi_s(t, x, u(\cdot))$ be the trajectory starting from a state x under input $u(\cdot) \in \mathcal{U}$. The reachable sets of the system (3) from a set of initial states $X_0 \subseteq \mathcal{X}$ during the interval $[0, t]$ is defined as:

$$Reach_s(t, X_0) = \{\, y = \Phi_s(\tau, x, u(\cdot)) \mid \tau \in [0, t], x \in X_0, u(\cdot) \in U_\mu \,\}.$$

The reachable set of the original system can be defined similarly.

It is easy to see that the size of the error set directly impacts the distance between the trajectories of the original and the approximate systems. It is therefore of great interest to construct hybridization domains with a small error bound. An intuitive observation is that to one can achieve a better interpolation accuracy by using smaller hybridization domains. This makes the staying time within this domain shorter, which results in a higher frequency of domain construction and more computation effort. However, when computing the reachable set of a hybrid system, accuracy becomes important since the error accumulation can be drastically aggravated by discrete dynamics in such systems, which induces spurious trajectories the exploration of which might be expensive and lead to erroneous results. In other words, for highly "sensitive" systems, requiring high accuracy can indeed be a way of reducing computation time in the long run. It is also crucial to exploit the structure of the dynamics in order to improve accuracy without causing too much additional computation time. In our previous work [8] we showed that this can be done by considering the relation between the interpolation error and the curvature of the dynamics. This enabled us to construct larger hybridization domains while maintaining the same accuracy. This result was applied to the class of quadratic functions. In this paper we continue this work by extending its application to more general non-linear systems. In addition, the error bound we used in [8] is valid for rather general systems, for more specific systems optimal bounds and domain construction can be considered. This is another problem we address in this paper. Before describing our new contributions, we first recap the result presented in the previous paper [8].

2.2 Hybridization error bound

It is possible to show an error bound that depends on the radius of the smallest containment ball of the simplex. The smallest containment ball of a simplex is the smallest ball that contains the simplex (which should not be confused with its circumcirle). In addition, using the curvature of

the functions f one can map the original space (where the functions f are defined) to an "isotropic" space by shrinking the domain along the direction with small curvature. As a result, in the isotropic space, the new domain becomes more "regular" and smaller. It is then possible to derive a tighter error bound depending on the radius of the smallest containment ball of the new domain. In the opposite direction, using the inverse of this transformation, we can extend a domain along the direction with small curvature while preserving the same error bound.

In order to explain this more formally, we first introduce some notation. From now on we write f^i with $i \in \{1, 2, \ldots, n\}$ to indicate the i^{th} component of f.

For a vector $d \in \mathbb{R}^n$, the first-order directional derivative of f^i with $i \in \{1, \ldots, n\}$ along the vector d is

$$\partial f^i(x, d) = \sum_{j=1}^{n} \frac{\partial f^i}{\partial x_j}(x) d_j$$

The Hessian matrix associated with each function f^i for $i \in \{1, \ldots, n\}$ is:

$$H^i(x) = \begin{pmatrix} \frac{\partial^2 f^i}{\partial x_1^2} & \cdots & \frac{\partial^2 f^i}{\partial x_1 x_n} \\ & \cdots & \\ \frac{\partial^2 f^i}{\partial x_1 x_n} & \cdots & \frac{\partial^2 f^i}{\partial x_n^2} \end{pmatrix}. \quad (4)$$

Then, the second-order directional derivative of f^i along d is defined as:

$$\partial^2 f^i(x, d) = d^T H^i(x) d.$$

Given a set $X \subseteq \mathcal{X}$, let $\gamma_X \in \mathbb{R}$ be the smallest real number such that f satisfies the following condition for all unit vector $d \in \mathbb{R}^n$ and for all $x \in X$:

$$\forall i \in \{1, \ldots, n\} : \max_{x \in X,\ ||d||=1} |\partial^2 f^i(x, d)| \leq \gamma_X. \quad (5)$$

The value γ_X is called the *maximal curvature* of f in X.

Intuitively, we are interested in finding an "isotropic" coordinate transformation so that when mapping a domain back to the original space, we obtain a larger domain while preserving the same interpolation error bound. To this end, we assume the boundedness of the directional curvature of f as follows. Given a set $X \subseteq \mathcal{X}$, we assume that there exists a positive-definite matrix C such that

$$\forall i \in \{1, \ldots, n\}\ \forall x \in X\ \forall d \in \mathbb{R}^n : \\ ||d|| = 1\ \wedge\ |\partial^2 f^i(x, d)| \leq d^T C d. \quad (6)$$

The matrix C corresponds to a bound on the directional curvature of f in the set X, and we call C a *curvature tensor matrix* of f in X. Geometrically speaking, for each $x \in X$ the ellipsoid defined by $d^T C d = 1$ is included in the level set defined by $|\partial^2 f^i(x, d)| = 1$ for all $i \in \{1, \ldots, n\}$.

We now derive from C a isotropic transformation as follows. Since C is a symmetric matrix with real entries, we can write its eigen-decomposition:

$$C = S \Xi S^T = S \begin{pmatrix} \xi_1 & 0 & \ldots & 0 \\ 0 & \xi_2 & \ldots & 0 \\ & & \ldots & \\ 0 & 0 & \ldots & \xi_n \end{pmatrix} S^T$$

where ξ_j with $j \in \{1, \ldots, n\}$ are the eigenvalues of C, and S is an orthonormal square matrix containing the eigenvectors of C, that is

$$S = [v_1 v_2 \ldots v_n].$$

Let $\xi_{max}(C)$ and $\xi_{min}(C)$ be the largest and smallest eigenvalues of the matrix C. Then, we consider the linear transformation defined by the matrix

$$T = S W_s S^T = S \begin{pmatrix} \sqrt{\frac{\xi_1}{\xi_{max}(C)}} & \ldots & 0 \\ & \ldots & \\ 0 & \ldots & \sqrt{\frac{\xi_n}{\xi_{max}(C)}} \end{pmatrix} S^T \quad (7)$$

and denote $\widehat{x} = Tx$. We call this an "isotropic" transformation. Indeed, the linear transfomation associated with the matrix T transform an ellipsoid in to a sphere, as shown in Figure 1. This can be seen later in Section 3. Let $\widehat{X}$ denote the set resulting from applying the above linear transformation to X, that is,

$$\widehat{X} = \{Tx \mid x \in X\}.$$

Figure 1: Illustration of the transformation to an isotropic space: The ellipse is a level set of the directional curvature and the curvature is small along its semimajor axis. The isotropic transformation T "shortens" the triangle on the left along the directions in which f has small curvature

The following theorem [19] shows an error bound which depends on the radius of the smallest containement ball in the isotropic space.

THEOREM 1. *Let l be the affine function that interpolates the functions f over the simplex Δ. Then, for all $x \in \Delta$*

$$||f(x) - l(x)|| \leq \gamma_\Delta \frac{r_c^2(\widehat{\Delta})}{2}.$$

where γ_Δ is the maximal curvature of f in Δ and $r_c(\widehat{\Delta})$ is the radius of the smallest containement ball of the transformed simplex $\widehat{\Delta}$.

The transformation of Δ to $\widehat{\Delta}$ can shorten an edge of Δ by a factor of up to $\sqrt{\frac{\xi_{max}(C)}{\xi_{min}(C)}}$ if this edge is aligned with the eigenvector corresponding to ξ_{min}. In the worst case, its length remains unchanged if it is aligned with the eigenvector corresponding to ξ_{max}.

Using this theorem we can determine a simplicial domain that guarantees a desired error bound ε. Let P be the set of current states of the system, and $\widehat{P}$ be the polytope resulting from applying the transformation T to P. The hybridization domain construction using an isotropic transformation is summarized in Algorithm 1.

Algorithm 1 Hybridization domain construction using isotropic transformation

Compute the transformation matrix T.

Determine the maximal radius r_c of the smallest containement ball corresponding to the desired error bound ε.

Let B be the ball of radius r_c the centroid of which coincides with that of the polytope $\widehat{P}$.

Construct in the isotropic space an equilateral simplex $\widehat{\Delta}$ that is circumscribed by the ball B. Such simplices have the largest volume for the given radius r_c of the smallest containement ball.

Use the inverse transformation T^{-1} to map the simplex $\widehat{\Delta}$ back to the original space. This results in a larger simplex Δ while guaranteeing the desired error bound.

Indeed, in this algorithm, the shape of the simplex is fixed while its orientation can be freely chosen. This offers a posibility of further optimization, such as reducing the frequency of domain constructions by orienting the simplex according to the dynamics of the system so that its trajectories stay in the domain for a longer time.

In the previous work [8], we also showed an effective application of this approach to quadratic systems where the Hessian matrices are constant and hence the maximal curvature γ_Δ can be straightforwardly computed. As mentioned earlier, in this paper we pursue this work and describe two new results.

The remainder of the paper is organized as follows. We first present the optimality property of the domain construction with isotropic transformation for quadratic systems. We then consider more general non-linear systems with non-constant Hessian matrices and describe an algorithm for constructing domains using a curvature estimation. We also demonstrate the interest of the algorithm by means of some examples and report preliminary experimental results on the computational efficiency of the algorithm.

3. OPTIMAL DOMAINS FOR QUADRATIC FUNCTIONS

We show a class of quadratic functions f for which the domain construction based on equilateral simplices in an isotropic space is optimal. This optimality property is stated as follows: given an error tolerance ε, the computed simplex Δ has the largest volume and, in addition, the error between f and its linear interpolation over Δ is not greater than ε.

Let each quadratic function f^i be written as

$$f^i(x) = x^T H^i x + (m^i)^T x + p^i$$

where H^i is a real-valued matrix of size $n \times n$, $m^i \in \mathbb{R}^n$ and $p^i \in \mathbb{R}$. Note that we use the same notation H^i here because the Hessian matrix of f^i is exactly H^i. For every $i \in \{1, \ldots, n\}$, we define the interpolation error function as

$$e^i(x) = f^i(x) - l^i(x)$$

which is also a quadratic function. We now study this error function and seek its maxima.

The error function can be expressed as:

$$e^i(x) = (w^i)^T x + q^i + x^T H^i x$$

where $w^i \in \mathbb{R}^n$ and $q^i \in \mathbb{R}$. Note that the level sets of this function form a family of conics with a common center, denoted by c^i. Indeed, they are ellipsoids if $det(H^i) > 0$ and hyperboloids if $det(H^i) < 0$. We now derive the error in a neighborhood of this common center. Let $\delta_x \in \mathbb{R}^n$ be a deviation from the common center c^i, then for every $i \in \{1, \ldots, n\}$

$$\begin{aligned} e^i(c^i + \delta_x) &= (w^i)^T(c^i + \delta_x) + q^i - (c^i + \delta_x)^T H^i (c^i + \delta_x) \\ &= [(w^i)^T c + q^i - (c^i)^T H_i c^i] + \\ &\quad (w^i)^T \delta_x - 2\delta_x^T H^i c^i - \delta_x^T H^i \delta_x \\ &= e^i(c^i) + (w^i)^T \delta_x - 2(c^i)^T H^i \delta_x - \delta_x^T H^i \delta_x. \end{aligned}$$

Since c is the common center of the family of conics corresponding to the error function, c satisfies the following

$$w^i - 2H^i c^i = 0.$$

Then,

$$\begin{aligned} (w^j)^T \delta_x &= 2(H^i c^i)^T \delta_x \\ &= 2(c^i)^T H^i \delta_x. \end{aligned}$$

It then follows from the above that

$$e^i(c^i + \delta_x) = e^i(c^i) - \delta_x^T H^i \delta_x.$$

We also observe that, the symmetric matrix H^i can be decomposed as

$$H^i = S W^T D W S^T$$

where D is a diagonal matrix with entries $\sigma_j \in \{-1, 0, +1\}$; W is a diagonal matrix whose entries on the diagonal are the square roots of the absolute values of the eigenvalues ξ_j of H^i; S is an orthonormal matrix containing the eigenvectors of H^i. We define a linear transformation

$$T^i = W^T S^T.$$

LEMMA 1. *Using the transformation $\widehat{\delta_x} = T^i \delta_x$, the term $\delta_x^T H^i \delta_x$ in the error $e^i(c^i + \delta_x)$ can be transformed into a quadratic form*

$$\delta_x^T H^i \delta_x = \sum_{j=1}^{n} \sigma_1^i \widehat{\delta_x}_j^{\,2}$$

where for all $j \in \{1, \ldots, n\}$ $\sigma_j^i \in \{-1, 0, +1\}$.

PROOF. Using $\delta_x = T^{-1}\widehat{\delta_x}$ and $H^i = S W^T D W S^T$, we obtain after some straightforward calculation:

$$\begin{aligned} \delta_x^T H^i \delta_x &= (T^{-1}\widehat{\delta_x})^T S W D W^T S^T T^{-1} \widehat{\delta_x} \\ &= (\widehat{\delta_x})^T D \widehat{\delta_x}. \end{aligned}$$

Therefore,

$$\begin{aligned} \delta_x^T H^i \delta_x &= \widehat{\delta_x}^T \begin{pmatrix} \sigma_1 & 0 & \ldots & 0 \\ 0 & \sigma_2 & \ldots & 0 \\ & & \ldots & \\ 0 & 0 & \ldots & \sigma_n \end{pmatrix} \widehat{\delta_x} \\ &= (\sigma_1 \widehat{\delta_x}_1^{\,2} + \sigma_2 \widehat{\delta_x}_2^{\,2} + \ldots + \sigma_n \widehat{\delta_x}_n^{\,2}). \end{aligned}$$

where $\forall j \in \{1,\ldots,n\} : \sigma_j \in \{-1,0,+1\}$. In other words, using the linear transformation T^i we transforms the matrix H^i into a diagonal matrix D which has only entries 0, $+1$ and -1 on the diagonal. ■

Again, we can see that the interpolation error in the new space (resulting from the transformation T) is isotropic, that is it does not depend on the direction of $\widehat{\delta_x}$.

We identify a class of quadratic systems such that for every function f^i, σ^i_j are all equal to either $+1$ or -1. In the isotropic space the level sets of the error are spheres with a common center. The circumsphere of $\widehat{\Delta}$ is the level set of value zero (due to interpolation over the vertices). Hence, the maximal value of $|e^i(x)|$ is achieved at c^i and is directly related to the square of the radius of the circumsphere of $\widehat{\Delta}$.

Using the above reasoning, we can determine the maximal value of every error function $|e^i(x)|$ ($i \in \{1,\ldots,n\}$). Let f^i be the function that corresponds to the largest value. We then take the associated matrix T^i to be the isotropic transformation T for domain construction purposes. Note that in this case, the circumsphere radius is also the radius of the smallest containement ball of Δ in the Theorem 1. For a given fixed circumsphere radius (corresponding to an error tolerance), an equilateral simplex has the optimal shape because it has the largest volume.

The number of entries $+1$ of D is called the positive index of inertia of H^i, and the number of entry 1 is called the negative index of inertia. According to Sylvester's law of inertia, the positive and negative indices of H^i are also the number of positive and negative eigenvalues of H^i.

THEOREM 2. *For a class of quadratic functions f such that all the Hessian matrices have either only positive eigenvalues or only negative eigenvalues, the domain construction method based on equilateral simplices in the isotropic space is optimal.*

When this condition on the eigenvalues is not satisfied, the theorem no longer holds, that is starting from equilateral simplices in the isotropic space does not yield an optimal construction. For example, in 2 dimensions, in the case where the number of σ_j equal to $+1$ is equal to that of σ_j equal to -1 (which implies that the error is a harmonic function of $\widehat{\delta_x}$), the maximal error is not achieved at the common center but on the boundary of the simplex. Investigating the optimality conditions for the remaining cases is part of our undergoing work.

4. ESTIMATING CURVATURE TENSORS

We now present the second contribution of this paper, which involves an effective application of the domain construction algorithm to more general functions with non-constant Hessian matrices.

It can be seen from Theorem 1 that to obtain a good interpolation error bound, we need to know an accurate curvature tensor of f as defined in (6). We observe that, given a simplex Δ, by definition, for each $i \in \{1,\ldots,n\}$ the eigenvalues of the Hessian matrix H^i are inside the interval $[-\gamma_X, \gamma_X]$ where γ_X is the maximal curvature of f inside X. Hence, the error bound is determined by the maximal eigenvalue $\xi_{max}(C)$ of the matrix C. Note additonally that $r_c(\widehat{\Delta})$ depends on $|det(T)|^{1/n}$ where $det(T)$ is the determinant of the transformation matrix defined in (7).

Therefore, we want to find a positive-definite matrix C that satisfies the condition (6) in the definition of curvature tensor and, in addition, makes $|\xi_{max}(C)||det(T)|^{2/n}$ as small as possible. To do so, we formulate this problem as solving the following constrained optimization problem:

$$\begin{aligned} &\min |\xi_{max}(C)||det(T)|^{2/n} \\ &\text{s.t. } \forall i \in \{1,\ldots,n\}\ \forall x \in X\ \forall d \in \mathbb{R}^n : \\ &\quad ||d|| = 1\ \wedge\ |\partial^2 f^i(x,d)| \leq |d^T C^i d|. \end{aligned}$$

Again, we express C in its eigen-decomposition form. Let $\xi_1, \xi_2, \ldots, \xi_n$ be the eigenvalues in increasing order of C, that is $0 < \xi_1 \leq \xi_2 \leq \ldots \leq \xi_n$, and $v^1, v^2, \ldots, v^n$ be the corresponding eigenvectors. From now on we use superscripts to denote eigenvectors since subscripts will be used to denote their coordinates. Thus,

$$C = S \Xi S^T$$

where

$$\Xi = diag(\xi_1, \xi_2, \ldots, \xi_n)$$

Therefore, mimimizing over all possible matrices C satisfying (6) is equivalent to minimizing over all possible ξ_i and all possible orthogonal matrices S.

Notice that, by the definition of the matrix T,

$$|det(T)| = |det(C)|^{1/2} = |(\Pi_{j=1}^n \xi_j)|^{1/2}$$

The objective function can therefore be written as:

$$|\xi_{max}(C)||det(T)|^{2/n} = |\xi_n|\ |(\Pi_{j=1}^n \xi_j)|^{1/n}.$$

On the other hand, the constraint (6) can be written as:

$$\begin{aligned} &\forall i \in \{1,\ldots,n\}\ \forall x \in X\ \forall d \in \mathbb{R}^n : \\ &\quad ||d|| = 1\ \wedge\ |\partial^2 f^i(x,d)| \leq |d^T S\, \Xi\, S^T d|. \end{aligned}$$

This problem might not have a solution or it might have a solution with some eigenvalue equal to 0, which makes C singular. In the following we consider another approach, which involves approximating C by making the error bound as small as possible while respecting the constraint (6).

Since the error bound depends on the maximal eigenvalue of C and the product of the eigenvalues of C, we estimate C by determining successively its eigenvalues ξ_j and eigenvectors v^j such that each ξ_j is made as small as possible while satisfying the condition (6).

More precisely, in the first step we determine ξ_n such that

$$\forall x \in X\ \forall i \in \{1,\ldots,n\}\ d \in \mathbb{R}^n :\ ||d|| = 1 \wedge \xi_n \geq |\partial^2 f^i(x,d)|.$$

We can find ξ_n by solving the following n optimization problems

$$\xi_{n,i} = max_{x \in X\ \wedge\ ||d||=1} |\partial^2 f^i(x,d)|, i \in \{1,\ldots,n\}$$

Then we take the largest among the computed maximal values:

$$\xi_n = max_{i \in \{1,\ldots,n\}} \xi_{n,i}.$$

In other words, by (5), ξ_n is exactly the largest curvature of f in X. The corresponding eigenvector v^n is chosen as

the unit vector d along which the second-order directional derivative of the corresponding f^i is the largest.

To determine the remaining $(n-1)$ eigenvalues, let $(n-1)$ vectors

$$v^1, \ldots, v^{n-1}$$

in $\mathbb{R}^n$ form an orthonormal basis for the orthogonal complement of $span\{v^n\}$ in $\mathbb{R}^n$. Then, any vector $d \in \mathbb{R}^n$ can be expressed as:

$$d = \sum_{j=1}^{n-1} \alpha_j v^j + \beta v^n \tag{8}$$

where $\alpha = (\alpha_1, \ldots, \alpha_{n-1}) \in \mathbb{R}^{n-1}$ and $\beta \in \mathbb{R}$.

Let

$$S^{n-1} = [v^1 v^2 \ldots v^{n-1}]$$

and Ξ^{n-1} be the diagonal matrix with $\xi_1, \xi_2, \ldots, \xi_{n-1}$ as its diagonal elements. Note that the superscripts here indicate that these matrices are used to compute the eigenvalue ξ_{n-1}. We define then

$$C^{n-1} = S^{n-1}\Xi^{n-1}(S^{n-1})^T.$$

Then, it is not hard to see that

$$d^T C d = \alpha^T C^{n-1}\alpha + \xi_n \beta^2.$$

The condition (6) becomes

$$\forall i \in \{1, \ldots, n\},\ \forall \alpha \in \mathbb{R}^{n-1},\ \forall \beta \in \mathbb{R},\ \forall x \in X : \\ \alpha^T C^{n-1}\alpha \geq |\partial^2 f^i(x, d)| - \xi_n \beta^2. \tag{9}$$

We denote the right hand side as a function of α and β:

$$w^i(\alpha, \beta) = |\partial^2 f^i(x, d)| - \xi_n \beta^2,$$

and

$$\eta^i(\alpha) = max_{\beta \in \mathbb{R}} w^i(\alpha, \beta). \tag{10}$$

The condition (9) is then equivalent to

$$\forall i \in \{1, \ldots, n\},\ \forall \alpha \in \mathbb{R}^{n-1},\ \forall x \in X : \\ \alpha^T C^{n-1}\alpha \geq \eta^i(\alpha) \tag{11}$$

LEMMA 2. *If β maximizes $w^i(\alpha, \beta)$ in (10) then β satisfies*

$$\frac{\partial^2 f^i(x, d)}{|\partial^2 f^i(x, d)|} \partial g^i(x, d) = \xi_n \beta. \tag{12}$$

where

$$g^i(x) = \partial f^i(x, v^n)$$

Intuitively, $\partial g^i(x, d)$ is the first-order directional derivative of g^i with respect to the vector d, and $g^i(x) = \partial f^i(x, v^n)$ is the first-order directional derivative of f^i with respect to the vector v^n.

PROOF. To solve (10), we consider the critical points of $w^i(\alpha, \beta)$ with respect to β. To express the partial derivative of $w^i(\alpha, \beta)$, we first rewrite the second-order directional derivative of f^i as the following sum:

$$\partial^2 f^i(x, d) = \sum_{jk} H^i_{jk}(x) d_j d_k$$

where H^i_{jk} is the element of the j^{th} line and the k^{th} column of the Hessian matrix of the function f^i.

Then,

$$\begin{aligned} \frac{\partial}{\partial \beta}(\partial^2 f^i(x, d)) &= \sum_{jk} H^i_{jk}(x)(d_j \frac{\partial d_k}{\partial \beta} + d_k \frac{\partial d_j}{\partial \beta}) \\ &= \sum_{jk} H^i_{jk}(x)(d_j v^n_k + d_k v^n_j) \quad \text{(from (8))} \\ &= 2\sum_{jk} H^i_{jk}(x) d_j v^n_k \\ &= 2\partial(\partial f^i(x, v^n), d) \\ &= 2\partial g^i(x, d) \quad \text{(from (12))} \end{aligned}$$

In addition,

$$\frac{\partial}{\partial \beta}(\xi_n \beta^2) = 2\xi_n \beta.$$

Note that the critical points that satisfy $\partial^2 f^i(x, d) = 0$ are not among the maxima of $w^i(\alpha, \beta)$. It then follows from the above that the critical points that are candidates to be among the maxima satisfy:

$$\frac{\partial^2 f^i(x, d)}{|\partial^2 f^i(x, d)|} \partial g^i(x, d) = \xi_n \beta.$$

This establishes the proof of the lemma. ■

To determine β, we consider two cases:

- If $\partial^2 f^i(x, d) > 0$, the equation (12) can be rewritten as:

  $$\sum_{j,k} H^i_{jk} v^n_j d_k = \xi_n \beta.$$

  It then follows from (8) that

  $$\sum_{j,k} H^i_{jk} v^n_j (\sum_{1}^{n-1} \alpha_j v^j_k + \beta v^n_k) = \xi_n \beta.$$

  Note that ξ_n and v^n are now known, from the above we can determine β as a function of α and $v^1, v^2, \ldots, v^{n-1}$. From now on, for clarity, we denote by β^n the value of β satisfying the above, since this value corresponds to the eigenvalue ξ_n and the eigenvector v^n computed in the first step. Hence, if $\partial^2 f^i(x, d) > 0$

  $$\beta^n = \frac{(v^n)^T H^i (\sum_{j=1}^{n-1} \alpha_j v^j)}{\xi_n - (v^n)^T H^i v^n}.$$

- If $\partial^2 f^i(x, d) < 0$, similarly we obtain

  $$\beta^n = \frac{-(v^n)^T H^i (\sum_{j=1}^{n-1} \alpha_j v^j)}{\xi_n + (v^n)^T H^i v^n}.$$

With the above β^n, $\eta^i(\alpha)$ in (10) can be determined. Hence, the condition (9) becomes

$$\alpha^T C^{n-1}\alpha \geq u(\alpha) = |\partial^2 f^i(x, d)| - \xi_n (\beta^n)^2 \tag{13}$$

where $d = \sum_1^{n-1} \alpha_j v^j + \beta^n v^n$.

We now come to the same problem as the initial but with only $(n-1)$ eigenvalues to determine. We can repeat this procedure to determine all the eigenvalues. More precisely,

we determine ξ_{n-1} by solving the following optimization problem over the variables $z = \sum_{j=1}^{n-1} \alpha_j v^j$ and x.

$$\begin{gathered}\xi_{n-1,i} = \max(|\partial^2 f^i(x, z + \beta^n v^n)| - \xi_n(\beta^n)^2) \\ \text{s.t. } x \in X \ \wedge \\ z \in \mathbb{R}^n \ \wedge \\ ||z + \beta^n v^n|| = 1.\end{gathered}$$

Then, ξ_{n-1} is determined as the largest of all $\xi_{n-1,i}$.

To reduce further to the problem with $(n-2)$ eigenvalues, we write:

$$d^T C d = \alpha^T C^{n-2} \alpha + \xi_n(\beta^n)^2 + \xi_{n-1}(\beta^{n-1})^2.$$

where α is now a vector in $\mathbb{R}^{n-2}$.

Then, the sequence of optimization problems can be formulated as follows. For $k = n-1, n-2, \ldots, 1$

$$\begin{gathered}\xi_{n-k,i} = \max |\partial^2 f^i(x, z + \sum_{j=n-k+1}^{n} \beta^j v^j)| - \sum_{j=k+1}^{n} \xi_j(\beta^j)^2 \\ \text{s.t. } x \in X \ \wedge \\ z \in \mathbb{R}^n \ \wedge \\ ||z + \textstyle\sum_{j=k+1}^{n} \beta^j v^j|| = 1\end{gathered} \tag{14}$$

Then, ξ_{n-k} is the largest value of all $\xi_{n-k,i}$.

In the above procedure, in order to proceed from the computation of the eigenvector ξ_{n-k} to that of ξ_{n-k-1} we need to compute the corresponding eigenvector v^{n-k}. As an example, in the step $k = n-1$, we obtain the solution $d = z + \beta^n v^n$ of the optimization problem. Let denote this d by d^{n-1} and we want to compute the corresponding eigenvector v^{n-1}. To this end, we use the following scheme. Indeed, at each step k, v^k is made orthogonal to the previous $v^{k+1}, \ldots, v^n$ by substracting the projection of d^k in the directions of $v^{k+1}, \ldots, v^n$.

$$u^k = d^k - \sum_{j=k+1}^{n} \frac{(v^j)^T d^k}{(v^j)^T v^j} v^j$$

Then we determine the eigenvector v^k as:

$$v^k = \frac{u^k}{||u^k||}.$$

Such n vectors v^k span the same subspace as n vectors d^k.

Finally, we construct the matrix C as follows:

$$C = S \begin{pmatrix} \xi_1 & 0 & \ldots & 0 \\ 0 & \xi_2 & \ldots & 0 \\ & & \ldots & \\ 0 & 0 & \ldots & \xi_n \end{pmatrix} S^T$$

where ξ_j with $j \in \{1, \ldots, n\}$ are the computed eigenvalues, and $S = [v^1 v^2 \ldots v^n]$ is an orthonormal matrix containing the computed eigenvectors.

5. REACHABILITY COMPUTATION USING HYBRIDIZATION

Once the curvature tensor matrix is estimated, we can compute from it an isotropic transformation T. This can then be used to create hybridization domains for reachability computation. The reachability computation accuracy depends on the precision of the curvature tensor approximation, since the latter is directly related to the error bound that is used to define the input set U. In the curvature estimation described in the previous section, the optimization problems are solved over all $x \in X$, that is the computed estimate is valid for the whole set X. The estimation precision can be improved by using a dynamical curvature estimation that is invoked each time a new hybridization domain needs to be created. In this case, the optimization domains can be chosen as a neighborhood of the current states of the system.

The reachability computation algorithm using hybridization is summarized by Algorithm 2 where P is a polytope containing all the initial states of the system. In each iteration, we first estimate the curvature tensor within a zone containing the current set P^k. This matrix is then used to construct the simplicial domain Δ. We perform the reachability computation from P^k under the approximate linear interpolating dynamics defined within Δ. This generates the polytope P^{k+1}. If this polytope contains points outside the current domain Δ we retrieve the previous polytope P^k and construct around P^k a new hybridization domain. Note that when the polytope P^k becomes too large to be included in Δ, splitting is required.

Algorithm 2 Reachability computation using hybridization

Input: Initial polytope P, interpolation error tolerance ε

$P^0 = P$, $k = 0$
$C = CurvatureEstimation(P^k)$
$\Delta = DomainConstruct(P^k, C, \varepsilon)$
while $k \leq k_{max}$ **do**
  $P^{k+1} = Reach_l(P^k, \Delta)$
  if $P^{k+1} \cap \bar{\Delta} \neq \emptyset$ **then**
    /* $\bar{\Delta}$ *is the complement of* Δ */
    $\Delta = DomainConstruct(P^k, \varepsilon)$
  else
    $k = k + 1$
  end if
end while

It is worth mentioning that in this algorithm we use polytopes to represent reachable sets. However, the proposed hybridization and domain construction methods can be combined with the algorithms using other set representations (such as [15, 11]).

6. EXPERIMENTAL RESULTS

We implemented the domain construction algorithm and tested in on various examples. For non-linear optimization we use the publicly available NLopt library [13] which provides a common interface for a number of different optimization algorithms. For the computation of reachable sets of the approximate piecewise-affine systems, we used the algorithms implemented in the tool d/dt [2].

We first illustrate the interest of the algorithm by a number of experiments on a 2-dimensional system, the dynamics of which is described as follows:

$$\dot{x}_1 = x_2 - x_1^3 + x_1 x_2^2 \tag{15}$$
$$\dot{x}_2 = x_1^3 \tag{16}$$

This example is adapted from the one used in the study

of stabilization of systems with uncontrollable linearization in [18] (page 346). The initial set is a small box $[0.5, 0.51] \times [0.5, 0.51]$. The error tolerance is equal to 0.5.

Figure 2 shows the reachable set computed using a hybridization without isotropic transformation. The hybridization domains are chosen to be equilateral and oriented along the local evolution direction of the dynamics. We can see that without using an isotropic transformation, the domains are small and thus a lot of domains were created.

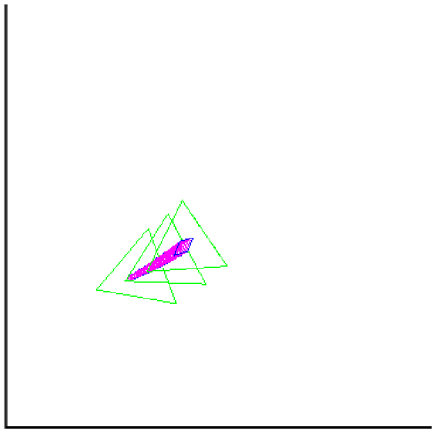

Figure 2: Domains constructed without isotropic transformation.

Figure 3 shows the reachable set computed using a hybridization with an isotropic transformation. In this experiment, the curvature estimation was done dynamically when each domain is created. We can see that the domains are larger for the same accuracy and less domains are needed.

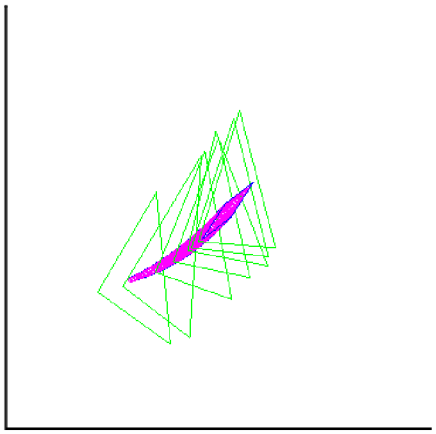

Figure 3: Domains constructed with the curvature tensor dynamically estimated over large zones.

The reachability computation can be further improved by using smaller optimization domains around the reachable sets. This in general requires some rough approximation of the reachable set within a number of next iterations. This is illustrated by the reachable set shown in Figure 4.

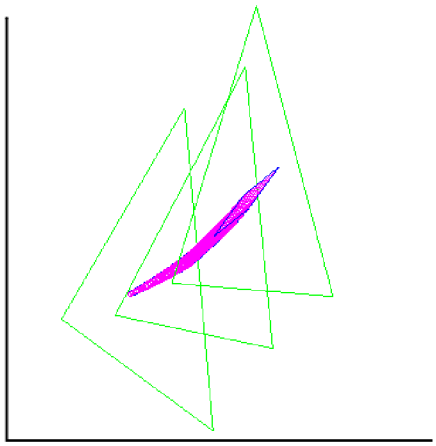

Figure 4: Domains constructed with the curvature tensor dynamically estimated over small zones.

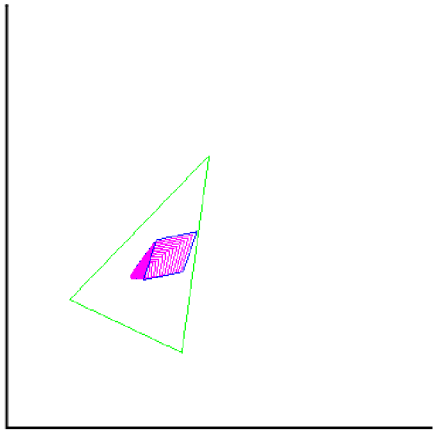

Figure 5: Reachability computation with a large error bound.

In order to illustrate the effect of error bounds, we fix the radius of the smallest containment ball in the isotropic space and perform two experiments: the first one with the curvature tensor estimated over a large zone and the second over a smaller zone. We observe that the hybridization domains in the two experiments are the same. However, a high curvature bound was computed in the first experiment and thus the corresponding error bound is large, which results in a large input set. This causes the system to expand fast, as one can see in Figure 6. On the other hand, with a better curvature estimate in the second experiment, the error bound is smaller and the reachable set computation is more accurate (see Figure 6).

In order to evaluate the performance of the algorithm, we performed a set of experiments on some polynomial systems (of degree 4) which are randomly generated. We report in the following the average computation times of 100 itera-

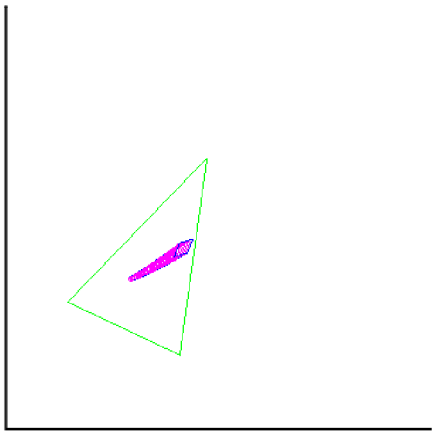

Figure 6: Reachability computation with a smaller error bound.

Dimension	Total time (s)	Optimisation time (s)
2	0.53	0.05
3	0.96	0.63
4	7.87	7.01
5	57.05	48.22
6	90.77	80.78
7	302.5	269.22

Figure 7: Computation times for polynomial systems.

tions for systems up to 7 dimensions. For each dimension, we tested 4 systems. In these experiments, for each system the curvature tensor matrix was estimated only once. The reason we did not go beyond 7 dimensions is that the optimisation took a lot of computation time (while the computation time for treating approximate piecewise affine systems is much less), as shown in Figure 7. Indeed, for a n-dimensional system, to estimate the curvature tensor matrix, we need to compute n eigenvalues, each of which requires solving n constrained optimization problems with $2n$ variables. Indeed, the curvature tensor estimation can be done a-priori over a large analysis zone and such a global estimate can be used for the whole reachability computation process. This alone can significantly improve the accuracy of the reachable set approximation, compared to the domain construction without isotropic transformation, as shown in the above 2-dimensional example. In order to include dynamic curvature estimation, we need more performant optimisation tools, such as those for specific classes of systems.

To sum up, our preliminary experiments demonstrated the interest of the proposed domain construction algorithm in terms of accuracy improvements. The practical efficiency of the algorithm is still limited by the required non-linear optimization. Beside exploiting more performant optimization algorithms, we plan to tackle this problem by exploring other methods for computing isotropic transformations without estimating the curvature tensor matrix.

7. CONCLUSION

In this paper we extended the curvature-based domain construction method to non-linear systems with non-constant Hessian matrices. In addition, we proved an optimality property of the domain construction for a class of quadratic systems. We demonstrated the effectiveness, in particular in terms of accuracy improvement, of the method on various examples. Future work directions include considering the optimal domain construction problem for larger classes of systems. Finding more efficient methods for computing curvature tensor matrices and isotropic transformations is also part of our future work.

8. REFERENCES

[1] M. Althoff, O. Stursberg, and M. Buss. Reachability Analysis of Nonlinear Systems with Uncertain Parameters using Conservative Linearization. *CDC'08*, 2008.

[2] E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate reachability analysis of piecewise linear dynamical systems. *HSCC'00*, LNCS 1790, 21-31, 2000.

[3] E. Asarin, T. Dang, and A. Girard. Reachability Analysis of Nonlinear Systems Using Conservative Approximation. *HSCC'03*, LNCS 2623, pp 20-35, Springer, 2003.

[4] E. Asarin, T. Dang, and A. Girard. Hybridization Methods for the Analysis of Nonlinear Systems. *Acta Informatica* **43**, 451-476, 2007.

[5] A. Chutinan and B.H. Krogh. Computational Techniques for Hybrid System Verification. *IEEE Trans. on Automatic Control* **48**, 64-75, 2003.

[6] T. Dang, C. Le Guernic and O. Maler. Computing Reachable States for Nonlinear Biological Models. *Computational Methods in Systems Biology CMSB'09*, LNCS 5688, pp 126-141, 2009.

[7] T. Dang. Approximate Reachability Computation for Polynomial Systems. *HSCC'06*, LNCS 3927, pp 138-152, 2006.

[8] T. Dang, O. Maler, and R. Testylier. Accurate hybridization of nonlinear systems. *HSCC'10*, ACM, pp 11-20, 2010.

[9] J. Della Dora, A. Maignan, M. Mirica-Ruse, and S. Yovine. Hybrid computation. Proceedings International Symposium on Symbolic and Algebraic Computation ISSAC, 2001.

[10] A. Girard. Reachability of Uncertain Linear Systems using Zonotopes. *HSCC'05*, LNCS 3414, 291-305, 2005.

[11] A. Girard, C. Le Guernic and O. Maler. Efficient Computation of Reachable Sets of Linear Time-invariant Systems with Inputs. *HSCC'06*, LNCS 3927, 257-271 2006.

[12] M.R. Greenstreet and I. Mitchell, Reachability Analysis Using Polygonal Projections. *HSCC'99* LNCS 1569, 103-116, 1999.

[13] S.G. Johnson. The NLopt nonlinear optimization package. http://ab-initio.mit.edu/nlopt

[14] M. Kloetzer and C. Belta, Reachability analysis of multi-affine systems. *HSCC Hybrid Systems: Computation and Control*, vol 3927 in LNCS, 348-362, Springer, 2006.

[15] A. Kurzhanskiy and P. Varaiya, Ellipsoidal Techniques for Reachability Analysis of Discrete-time Linear Systems, *IEEE Trans. Automatic Control* **52**, 26-38, 2007.

[16] M. Kvasnica, P. Grieder, M. Baotic and M. Morari. Multi-Parametric Toolbox (MPT) *HSCC'04*, LNCS 2993, pp 448-462, Springer, 2004.

[17] I. Mitchell and C. Tomlin. Level Set Methods for Computation in Hybrid Systems, *HSCC'00*, LNCS 1790, 310-323, 2000.

[18] S. Sastry. Nonlinear Systems: Analysis, Stability, and Control. Springer-Verlag, 1999.

[19] J.R. Shewchuk. What Is a Good Linear Element? Interpolation, Conditioning, and Quality Measures. Eleventh International Meshing Roundtable (Ithaca, New York), pp 115-126, Sandia National Laboratories, September 2002.

[20] A. Tiwari and G. Khanna. Nonlinear Systems: Approximating Reach Sets. *HSCC'04*, LNCS 2993, pp 600-614, 2004.

[21] P. Varaiya, Reach Set computation using Optimal Control, *KIT Workshop, Verimag, Grenoble*, 377-383, 1998.