Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

MAXIS-G: a software package for computing polyhedral invariant sets for constrained LPV systems

Stefano Miani and Carlo Savorgnan
Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica
Universitá degli Studi di Udine
Via delle Scienze 208, 33100 Udine - Italy
{miani.stefano,carlo.savorgnan}@uniud.it

Abstract— **In this paper, a software package for computing the maximal invariant set of constrained linear parameter varying (LPV) systems is presented. The theoretical details at the basis of the proposed algorithm are first briefly illustrated along with the extensions which allow to deal with systems affected by disturbances and different kinds of uncertainty. Then, the algorithm is presented together with some comments on its actual implementation.**

I. Introduction

In this work, the software package MAXIS-G, developed for the computation of invariant sets for constrained linear parameter varying (LPV) systems is presented. Though this class of systems (and the determination of invariant sets which can be associated to them) has received a great deal of attention in the literature (see [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]), there are few software packages available for the computation of invariant sets.

The software package presented here is a contribution in this area and deals exclusively with polyhedral invariant sets. This interactive presentation is organized as follows: first, in Section II, the theoretical background of the determination of polyhedral invariant sets for the considered class of systems is presented, then, in Section III, the algorithms used for the effective computation of an invariant set are illustrated, and finally, in Section IV, the numerical aspects of the implemented software software as well as some heuristics to reduce the computational load are given. Some examples in section V complete the present work.

II. Notation and preliminaries

In this paper the considered discrete time systems are of the form:

$$x(t+1) = A(w(t))x(t) + B(w(t))u(t) \tag{1}$$

where $x(t) \in \mathcal{R}^n$ and $u(t) \in \mathcal{R}^q$ are respectively the state and the input vectors. $w(t) \in W \in \mathcal{R}^m$ is a time varying parameter belonging to the compact set $W = \{w : \ w_i \geq 0, \ \sum_{i=1}^{s} w_i = 1\}$ where s is finite.

The matrices $A(w(t)) \in \mathcal{R}^{n \times n}$ and $B(w(t)) \in \mathcal{R}^{n \times q}$ satisfy

$$A(w(t)) = \sum_{i=1}^{s} A_i w_i(t), \ B(w(t)) = \sum_{i=1}^{s} B_i w_i(t) \tag{2}$$

where the matrices A_i and B_i are given.

From the control design point of view, different cases can be distinguished depending on the knowledge of $w(t)$. When the current value of $w(t)$ is not available for synthesis purposes, we will talk about **robust** synthesis, otherwise we will talk about **gain scheduling** synthesis. Concerning this latter case, another important distinction can be made depending on which are the values that $w(t)$ attains: if $w(t)$ doesn't belong point-wise to W but it attains values only on the vertices we will talk about **switched gain scheduling** synthesis. We do not consider such distinction for robust synthesis since switched and non switched systems are equivalent in this case (see [15] for more details).

As previously mentioned, the sets dealt with are polyhedral, say whose form is

$$\mathcal{X} = \{x \in R^n \ : \ f_i^T x \leq g_i, \ \ i = 1 \ldots, m\} \tag{3}$$

where f_i are vectors in R^n, the superscript T in f_i^T denotes the transpose and each of the $g_i > 0$ is a scalar value. Denoting by $F \in R^{m \times n}$ the matrix $F = [f_1, f_2, \ldots, f_m]^T$ and by $g \in R^m$ the vector $g = [g_1, g_2, \ldots, g_m]^T$, it is possible to adopt a more compact representation for (3)

$$\mathcal{X} = \{x \in R^n \ : \ Fx \leq g\} \tag{4}$$

The above, is known as constraints representation. Another equivalent way to represent a given polyhedron is the one obtained from its r vertices x_i. Denoting by $X \in R^{n \times r}$ the matrix $X = [x_1, x_2, \ldots, x_r]$, we obtain the vertices representation:

$$\mathcal{X} = \{x : x = X\alpha, \sum_{i=1}^{r} \alpha_i \leq 1, \ \alpha_i \geq 0\}. \tag{5}$$

In the standard terminology $\mathcal{X}$ is referred to as the convex-hull of the vertices.

The algorithms used in the software are based on the concept of pre-image set.

Definition 2.1: Given the dynamic system (1), the one step **pre-image set** of the set $\mathcal{X}^{(0)} \subset R^n$ is given by all the states that can by brought in one step in $\mathcal{X}^{(0)}$ when a suitable control action is applied. The pre-image set, called $\mathcal{X}^{(1)}$, can be shown to be (see [16], [17])

- in the robust case

$$\mathcal{X}^{(1)} = \{x \in R^n : \ \exists u : F(A_i x + B_i u) \leq g, \ \forall i\} \tag{6}$$

- in the switched gain scheduling case

$$\mathcal{X}^{(1)} = \{x \in R^n : \ \exists u_i : F(A_i x + B_i u_i) \leq g, \ \forall i\} \quad (7)$$

Remark 2.1: It is important to notice the difference between (6) and (7). In the robust case the value of the input is independent of the value of i, while in the switched gain scheduling case the knowledge of such value is actively used in the determination of the value of the input u_i.

Concerning the gain scheduling synthesis, it is worth saying that the present work deals only with the switched gain scheduling case, since up to now no numerical results concerning the computation of pre-image sets for the general gain scheduling case are available.

Other useful definitions which will be used in the sequel are reported next.

Definition 2.2: Given the dynamic system (1), a set $\mathcal{X} \subset \mathcal{R}^n$ is said to be **invariant** if for every value of $x(t)$ there is a proper value of the control $u(t)$ such that $x(t+1) \in \mathcal{X}$.

Note 2.1: The previous definition can be given in this equivalent way: an invariant set is a set that is contained in its pre-image.

A concept similar to invariance, but with possibly stronger requirements, is the concept of contractivity introduced in the following definition.

Definition 2.3: Given the dynamic system (1) and $0 < \lambda \leq 1$, a set $\mathcal{X} \subset \mathcal{R}^n$ is said to be **λ-contractive** if for every value $x(t)$ of the state there is a proper value of the control $u(t)$ such that $x(t+1) \in \lambda\mathcal{X} = \{x \in \mathcal{R}^n : \frac{x}{\lambda} \in \mathcal{X}\}$.

Looking at the definitions (2.2) and (2.3), it can be noticed how these are equivalent when $\lambda = 1$. We can therefore say that an invariant set is 1-contractive. From now on we will speak only about contractive sets.

In [16] it has been shown how it is possible to compute a λ-contractive polyhedron for the robust case. Starting from an initial polyhedral set $\mathcal{X}^{(0)}$ containing the origin , and recursively computing the pre-image sets $\mathcal{X}^{(k)}$ for the dynamic system

$$\tilde{x}(t+1) = \frac{A(w(t))}{\lambda}\tilde{x}(t) + \frac{B(w(t))}{\lambda}\tilde{u}(t) \quad (8)$$

the λ-contractive set for the original system can be calculated as

$$\mathcal{X}^{(\infty)} = \bigcap_{k=0}^{k=\infty} \mathcal{X}^{(k)} \quad (9)$$

An important fact is that $\mathcal{X}^{(\infty)}$ is the largest λ-contractive set contained in $\mathcal{X}^{(0)}$ for system (1). This means that all the other λ-contractive sets are contained in it.

Generally, this recursive technique can be used also in the switched gain scheduling case. Therefore, the following result holds:

Theorem 2.1: Given a polyhedral set $\mathcal{X}^{(0)}$, the maximal λ-contractive set contained in it for the switched system (1) can be computed as $\mathcal{X}^{(\infty)} = \bigcap_{k=0}^{\infty} \mathcal{X}^{(k)}$, where $\mathcal{X}^{(k)}$, $k > 1$, is the pre-image set of $\mathcal{X}^{(k-1)}$ for the switched system (8).

A. λ-contractive sets and Lyapunov functions

A bounded λ-contractive set corresponds to the level set of a polyhedral control Lyapunov function[1] (see [16] and [18]). Therefore, by choosing a bounded $\mathcal{X}^{(0)}$, the recursive approach for the computation a λ-contractive set can return a Lyapunov function.

III. Algorithms

Based on the previous results we are now going to show in more detail which are the algorithms used in the software for calculating a contractive set for (1). To avoid storing all the pre-image sets and intersecting them all at the end, the intersection between $\mathcal{X}^{(k)}$ and $\mathcal{X}^{(k+1)}$ is performed at each step.

Algorithm 3.1: **Robust case**

1) Set the initial polyhedra $X^{(0)} = \{x : F^{(0)}x \leq g^{(0)}\}$ and the parameters λ and ε such that $\varepsilon \geq 0$ and $0 < \lambda \leq \lambda + \varepsilon \leq 1$. $k = 0$;
2) compute the expanded set $\mathcal{Q}^{(k)} \subset \mathcal{R}^{(n+m)}$:
$$\mathcal{Q}^{(k)} = \{(x,u) : F^{(k)}[A_i x + B_i u] \leq \lambda g^{(k)}, \ \forall i\};$$
3) compute the projection of $\mathcal{Q}^{(k)}$ on R^n :
$$\mathcal{P}^{(k)} = \{x : \ \exists u \ \ : \ (x,u) \in \mathcal{Q}^{(k)}\};$$
4) compute the polyhedron
$$\mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} \bigcap \mathcal{P}^{(k)}$$
and let the matrices $F^{(k+1)}$ and $g^{(k+1)}$ be those associated to the constraints representation of the set $\mathcal{X}^{(k+1)}$, say $\mathcal{X}^{(k+1)} = \{x : \ F^{(k+1)}x \leq g^{(k+1)}\}$
5) check if $\mathcal{X}^{(k+1)}$ is $(\lambda+\varepsilon)$-contractive. In the affirmative case set $\mathcal{P} = \mathcal{X}^{(k+1)}$ and stop, else set $k = k+1$ and go to 2).

Differently from section (II), the variable ε has been introduced. This tolerance, that leads to produce a $(\lambda+\varepsilon)$-contractive set $\mathcal{P}$, is necessary to bypass a possibly infinite computation. As a matter of fact, it may happen that without introducing ε, starting form a certain value of k, the following iterations of the algorithm do not change $\mathcal{X}^{(k)}$ significantly. Therefore, relaxing the condition in step 5), the stop criterion is met more easily.

To accommodate the switched gain scheduling case the previous algorithm must be slightly modified as follows.

Algorithm 3.2: **Switched gain scheduling case**

1) Set the initial polyhedra $X^{(0)} = \{x : F^{(0)}x \leq g^{(k)}\}$ and the parameters λ and ε such that $\varepsilon \geq 0$ and $0 < \lambda \leq \lambda + \varepsilon \leq 1$. $k = 0$;
2) compute the expanded sets $\mathcal{Q}_i^{(k)} \subset \mathcal{R}^{(n+m)}$:
$$\mathcal{Q}_i^{(k)} = \{(x,u) : F^{(k)}[A_i x + B_i u] \leq \lambda g^{(k)}\};$$
3) compute the projection of $\mathcal{Q}_i^{(k)}$ on R^n :
$$\mathcal{P}_i^{(k)} = \{x : \ \exists u \ \ : \ (x,u) \in \mathcal{Q}_i^{(k)}\};$$

[1] the term control refers to the fact that the function considered is a Lyapunov function only for a properly chosen value of the control input

4) compute the polyhedron

$$\mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} \bigcap_{i=1\ldots s} \mathcal{P}_i^{(k)}$$

and let the matrices $F^{(k+1)}$ and $g^{(k+1)}$ be those associated to the constraints representation of the set $\mathcal{X}^{(k+1)}$, say $\mathcal{X}^{(k+1)} = \{x : \ F^{(k+1)}x \leq g^{(k+1)}\}$

5) check if $\mathcal{X}^{(k+1)}$ is $(\lambda+\varepsilon)$-contractive. In the affirmative case set $\mathcal{P} = \mathcal{X}^{(k+1)}$ and stop, else set $k = k+1$ and go to 2).

For a better understanding of the two algorithms it is important to underline where the pre-image set is calculated. In algorithm (3.1) such set is represented by $\mathcal{P}^{(k)}$, while in algorithm (3.2) the pre-image set corresponds to the intersection $\bigcap_{i=1\ldots s} \mathcal{P}_i^{(k)}$.

Remark 3.1: When no inputs are present ($m = 0$) the algorithm can still be used. To handle such situation it is sufficient to skip the projection step (the "expansion" step produces polyhedra that belong to $\mathcal{R}^n$).

A. Constrained systems

Most interesting synthesis problems in general have to cope with state an input constraints. Concerning state constraints only, if $\mathcal{X}^{(0)}$ is the set given by the constraints on the state variables, then algorithms (3.1) and (3.2) produce as output (if any) the maximal set of initial states such that $x(t)$ respects the constraints when a proper feedback control is used.

Some small changes on the algorithms are required when input constraints are also to be dealt with. Assume the input constraint is of the form $u(t) \in \mathcal{U}$, where $\mathcal{U}$ is a polyhedral set with constraints representation $\mathcal{U} = \{u : \ \ F_u u \leq g_u\}$. In this case it is sufficient to notice that the pre-image set for system (1) becomes

$$\mathcal{X}^{(1)} = \{x \in R^n : \ \exists u \in \mathcal{U} : F(A_i x + B_i u) \leq g, \ \forall i\} \tag{10}$$

in the robust case and

$$\mathcal{X}^{(1)} = \{x \in R^n : \ \exists u_i \in \mathcal{U} : F(A_i x + B_i u_i) \leq g, \ \forall i\} \tag{11}$$

in the switched gain scheduling case. This added conditions can be easily incorporated in the proposed algorithms by replacing the computation of the expanded set $\mathcal{Q}^{(k)}$ by

$$\mathcal{Q}^{(k)} = \{(x,u) : \begin{cases} F^{(k)}[A_i x + B_i u], \ \leq \lambda g^{(k)}, \ \forall i, \\ F_u u \leq g_u \end{cases} \tag{12}$$

in the robust case and by

$$\mathcal{Q}_i^{(k)} = \{(x,u) : \begin{cases} F^{(k)}[A_i x + B_i u], \ \leq \lambda g^{(k)}, \\ F_u u \leq g_u \end{cases} \tag{13}$$

in the gain scheduling case.

Another interesting kind of constraints which can be dealt with are mixed input and state variables constraints of the form $C_x x + C_u u \leq g_{xu}$. These constraints are also accommodated in the software, but for brevity we will not give here any further detail.

B. Systems with disturbances

Another practical issue for synthesis problems is the presence of disturbances in the system. A well established way of modeling the disturbance action for linear systems is the following

$$x(t+1) = A(w(t))x(t) + B(w(t))u(t) + Ed(t) \tag{14}$$

where $d(t) \in \mathcal{D} \subset \mathcal{R}^p$ represents the unknown disturbance input and $E \in \mathcal{R}^{(n\times p)}$ is the disturbance matrix. $\mathcal{D}$ is a bounded set containing the origin. To take into account also the disturbance effect, some small modifications need once again to be introduced, precisely the expansion set has to be changed to accomplish a worst case procedure:

$$\mathcal{Q}^{(k)} = \{(x,u) : F^{(k)}[A_i x + B_i u] \leq \lambda(g^{(k)} - \max_{d\in\mathcal{D}} F^{(k)}Ed), \ \forall i\} \tag{15}$$

for the robust case and

$$\mathcal{Q}_i^{(k)} = \{(x,u) : F^{(k)}[A_i x + B_i u] \leq \lambda(g^{(k)} - \max_{d\in\mathcal{D}} F^{(k)}Ed)\} \tag{16}$$

for the switched gain scheduling case.

The newly introduced expression $\max_{d\in\mathcal{D}}$ has to be intended component-wise. If the value of $g^{(k)} - \max_{d\in\mathcal{D}} F^{(k)}Ed$ turns out to be not positive, the algorithm is stopped because there is no invariant set with the required characteristics.

C. Continuous-time case: Euler discretization

Continuous-time synthesis problems (even in the presence of state/input constraints and disturbances) for systems of the form

$$\dot{x} = A^c(w(t))x(t) + B^c(w(t))u(t) \tag{17}$$

(c stands for continuous) can be handled by the techniques already presented by using the discrete-time Euler Approximating System (EAS for short), say

$$x(t+1) = (I + \tau A^c(w(t)))\,x(t) + \tau B(w(t))u(t) \tag{18}$$

where I is the identity matrix in $R^{n\times n}$ and $\tau > 0$ is a discretization parameter. We refer the reader to [19] for the technical details concerning the approximation of the maximal domain of attraction for input/state constrained dynamic systems and to [20] for the use of EAS discretization schemes for suboptimal control problem solution.

IV. Numerics

In this section a deeper insight on the implementation of the algorithms is given. Before giving any detail we make a coarse analysis of which are the hardest points from a computational point of view. Since the two algorithms are similar, we can look at them together considering each step. Basically, the algorithms manipulate the matrices representing the various sets involved and iterate on such representations until a stopping condition is met. Unfortunately, the number of steps needed for the program termination together with the dimensions of such matrices is not known a priori and this prevents a worst case analysis even if the state dimension is restricted to $n = 3$. Due to this reason, an estimate of the computational complexity will be reported just for the first three steps.

- step 1) is trivial since only some initializations are performed;
- step 2) is also quite simple since the expansion of $\mathcal{X}^{(k)}$ corresponds to s matrix multiplications, precisely those needed to form the products

$$F^{(K)}\left[\begin{array}{cc} A_i & B_i \end{array}\right]$$

  The number of flops involved depends on the dimension of the matrix $F^{(K)}$ (say on its number of rows $n_{F^{(K)}}$, on the number of system vertices s and on the state and input dimensions, n and q respectively. The computational complexity is $\mathcal{O}(n_{F^{(K)}}n(n+q))$.
- step 3) is the most critical one since it requires the projection of the expanded sets on the state space. This operation can be carried out by using the Fourier-Motzkin elimination method (see [21] for details) that iteratively combines $q+1$ inequalities of $\mathcal{Q}^{(K)}$ (or $\mathcal{Q}_i^{(K)}$ in the switched case), to obtain a projected constraint on the n dimensional space. The method used in the current implementation is the basic one whose worst case complexity is $\mathcal{O}(n_{F^{(K)}}{}^{2^q})$ (see [22] for a general overview on polyhedral set projection). The set of the projected inequalities can form a redundant representation (say a representation in which not all the inequalities are necessary to describe the polyhedron) of $\mathcal{P}^{(k)}$ (or $\mathcal{P}_i^{(k)}$ in the switched case). Eliminating a redundant inequality corresponds to a standard linear programming (LP) problem whose complexity clearly depends on the dimensions of the data and are in general not computable a-priori.
- step 4) requires an intersection to be performed. This operation can also be carried by a sequence of LP problems (when we intersect two polyhedrons described by respectively j and k inequalities, to have a non-redundant representation of the polyhedron given by the intersection, $j+k$ LP problems need to be executed).
- step 5) can be dealt with by executing the steps from 2) to 4), where $\lambda+\varepsilon$ instead of λ is used, and check if the obtained $\mathcal{X}^{(k+1)} \subseteq \lambda\mathcal{X}^{(k+2)}$ (this means checking if all the inequalities of $\lambda\mathcal{X}^{(k+2)}$ are redundant with respect to $\mathcal{X}^{(k+1)}$). Another way of carrying out the contractivity test is checking if for every $x(t) \in \mathcal{X}^{(k+1)}$ there exists a control input value $u(t)$ (or a discrete set of control input values $u_i(t)$ for the gain scheduling switched case) such that $x(t+1) \in (\lambda+\varepsilon)\mathcal{X}^{(k+1)}$. The latter solution is a bad choice since it is really time consuming (implementing it would amount to enumerate the vertices of $\mathcal{X}^{(k+1)}$ and check the existence of a control input for everyone of them).

A. Speeding up the algorithm

MAXIS-G uses some techniques to make the computation of the invariant sets faster:

- we have already seen how step 5) can be carried out by repeating steps from 2) to 4). This consideration leads to the possibility of incorporate the redundancy test of the iteration k with the steps 2)-4) of the iteration $k+1$ with small overheads;
- the intersection of two polyhedra $\mathcal{X}_1$ and $\mathcal{X}_2$ has to be carried out intersecting every constraint of $\mathcal{X}_1$ w.r.t. $\mathcal{X}_2$. A fast intersection test (e.g. made w.r.t. a reduced complexity polyhedron that contains $\mathcal{X}_2$) can be preliminarily performed to quickly eliminate some of the redundant constraints;
- in step 4) we have seen that $\mathcal{X}^{(k+1)}$ is defined as the intersection of $\mathcal{X}^{(k)}$ and $\mathcal{P}^{(k)}$. It's easy to figure out how some of the constraints of $\mathcal{X}^{(k+1)}$ are new (say generated from the expansion-projection step in the iteration k) and some of them are "inherited" from $\mathcal{X}^{(k)}$. In steps 2) and 3) it can be observed how every constraint of $\mathcal{P}^{(k)}$ is generated from only a certain number of constraints in $\mathcal{X}^{(k)}$, let's say j. Every j-tuplet in $\mathcal{X}^{(k+1)}$ that is inherited from $\mathcal{X}^{(k)}$ produces a constraint that has already been generated in the previous algorithm iteration. Avoiding such redundant calculation drastically reduces the computational load.

It is worth noting that all the algorithms are implemented in C++ in MAXIS-G by using two different polyhedron representation:

- double-description representation (the informations about vertices and inequalities are stored);
- constraints representation (no information about the vertices of the polyhedron are stored).

B. Software availability

MAXIS-G is available under GPL2 license and can be freely downloaded [23].

The software package is composed by two parts:

- a command line program that can be used to execute all the calculations (like writing the input files and calculate the invariant sets);
- a simple interface that has been designed to ease the input and output operations within the Matlab™environment.

together with a short user manual.

V. Examples and software performance

A. Maximum invariant set for a system with disturbance

Consider the 2-dimensional uncertain system with disturbance

$$x(t+1) = [w(t)A_1 + (1-w(t))A_2]x(t) + Bu(t) + Ed(t) \quad (19)$$

with

$$A_1 = \begin{bmatrix} 1.3 & 2.1 \\ 0 & 1.5 \end{bmatrix};\ A_2 = \begin{bmatrix} 1.3 & 1 \\ 0 & 1.5 \end{bmatrix} \quad (20)$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix};\ E = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (21)$$

The disturbance is bounded as $|d(t)| \leq 0.1$ and $x(t)$ is constrained to the closed polyhedron given by $\|x(t)\|_\infty \leq 1$. Suppose that the uncertain variable attains only two values

$w(t) = \{0,1\}$. We want to study which are the advantages given by the knowledge of $w(t)$ when synthesizing a control. MAXIS-G can be used to calculate the maximum invariant set contained in $\|x(t)\|_\infty \leq 1$ in the robust and the switched gain scheduling case. In the former case we find a polyhedron with 12 vertices and in the latter a polyhedron with 8 vertices.

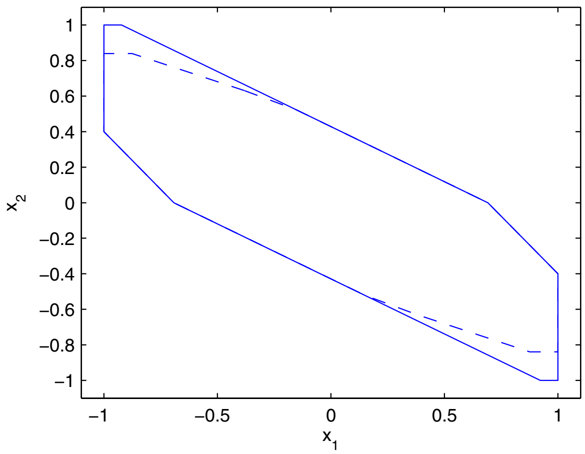

Fig. 1. Maximum invariant set for system (19).

Figure 1 shows the result of such computation: the dashed line represents the robust case, the solid line the switched gain scheduling case. The calculation time is 0.008 s when using the double-description representation and 0.09 s when using the constraints representation[2].

B. Lyapunov function for an uncertain system

Consider the 2-dimensional continuous-time uncertain system proposed in [11].

$$\dot{x}(t) = [w(t)A_1 + (1 - w(t))A_2]x(t) \tag{22}$$

with

$$A_1 = \begin{bmatrix} 0 & 1 \\ -2 & -1 \end{bmatrix};\ A_2 = \begin{bmatrix} 0 & 1 \\ -(2+\sigma) & -1 \end{bmatrix} \tag{23}$$

and $0 \leq w(t) \leq 1$ for all $t \geq 0$. For the above system it can be shown that the maximum value of σ that assures stability is 7. The goal of this example is calculating a Lyapunov function for the greatest value of σ. By using MAXIS-G with the initial set $\mathcal{X}^{(0)} = \{x : \|x\|_\infty \leq 1\}$ and discretizing the system with $\tau = 2.5 \times 10^{-4}$. Using $\lambda = 1 - 1 \times 10^{-9}$ and $\varepsilon = 9 \times 10^{-10}$ a polyhedral Lyapunov function can be computed when $\sigma = 6.97$. The computed polyhedron is really complex since it is given by over 12000 constraints (see figure 2). The calculation time is 22.5 s when using the double-description representation while it is greater than one hour when using the constraints representation.

[2]All the tests on MAXIS-G have been performed on a P4 2.66 GHz computer. The measurements have been made by using the utility *time*.

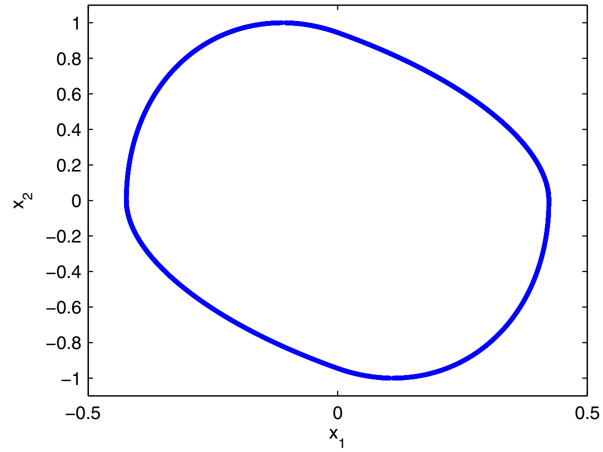

Fig. 2. Lyapunov function for system (22).

C. A software comparison

So far we have not been able to find any comparable software. In a search for comparison results, the work [24] can be cited. In such work the authors, which did not seem to be aware of the present developments, proposed some results on invariant set calculation for autonomous systems. The example we will take into account, which is the same as in the cited paper, is the following. Consider the discrete time system characterized by the matrices

$$A_1 = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix};\ A_2 = \begin{bmatrix} 1 & 0.2 \\ 0 & 1 \end{bmatrix} \tag{24}$$

$$B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix};\ B_2 = \begin{bmatrix} 0 \\ 1.5 \end{bmatrix} \tag{25}$$

This system can be stabilized by the state feedback $u = [-0.3\ -0.1]x$. The calculation of the maximal invariant set in the presence of the constraints $\|x_i\|_1 \leq 10$, $\|u\|_1 \leq 1$ produces a polyhedron bounded by 30 inequalities. In [24] such set has been calculated in 14.4 s by using Matlab$^{\text{TM}}$6.5 on a P4 2 GHz computer. The calculation time obtained by using MAXIS-G is 0.008 s when using the double-description representation and 0.087 s when using the constraints representation.

VI. Conclusions

A software for the computation of polyhedral invariant sets for discrete time systems has been presented, paying attention on the algorithms used and on how they cope with disturbances, uncertainties. It has been shown also how we can use such algorithm for continuous time systems and to calculate Lyapunov functions.

References

[1] J. Daafouz and J. Bernussou. Parameter dependent Lyapunov functions for discrete time systems with time varying parametric uncertainties. *Systems & Control Letters*, 43(5):355–359, 2001.

[2] R. K. Brayton and C. H. Tong. Constructive stability and asymptotic stability of dynamical systems. *IEEE Trans. Circuits and Systems*, 27(11):1121–1130, April 1980.

[3] A. P. Molchanov and E. S. Pyatnitskiĭ. Lyapunov functions that define necessary and sufficient conditions for absolute stability of nonlinear nonstationary control systems. I. *Automat. Remote Control*, 47(3):344–354, 1986.
[4] N. E. Barabanov. Lyapunov indicator of discrete inclusion. I. *Automat. Remote Control*, 49(2):152–157, 1988.
[5] E. G. Gilbert and K. K. Tan. Linear systems with state and control constraints: the theory and the applications of the maximal output admissible sets. *IEEE Trans. Automat Control*, 36(9):1008–1020, 1991.
[6] Y. Ohta, H. Imanishi, L. Gong, and H. Haneda. Computer generated Lyapunov functions for a class of nonlinear systems. *IEEE Trans. Circuits Systems I Fund. Theory Appl.*, 40(5):343–354, 1993.
[7] F. Blanchini. Ultimate boundedness control for discrete-time uncertain system via set-induced Lyapunov functions. *IEEE Trans. Automat Control*, 39(2):428–433, 1994.
[8] Rakovic S. V., Kerrigan E. C., Kouramas K. I., and Mayne D. Q. Invariant approximations of the minimal robust positively invariant set. *IEEE Transactions on Automatic Control*, 2005. In press.
[9] Y. Ohta and M. Tsuji. A generalization of piecewise linear lyapunov functions. In *Proceedings of the* 42^{nd} *Dec. and Control Conference*, Maui, HI, USA, 2003.
[10] M. Johansson and A. Rantzer. Computation of piecewise quadratic Lyapunov functions for hybrid systems. *IEEE Trans. Automat Control*, 43(4):555–559, 1998.
[11] A. L. Zelentsowsky. Nonquadratic lyapunov functions for robust stability analysis of linear uncertain system. *IEEE Trans. Automat Control*, 39(1):135–138, 1994.
[12] G. Chesi, A. Garulli, A. Tesi, and A. Vicino. Homogeneous Lyapunov functions for systems with structured uncertainties. *Automatica J. IFAC*, 39(6):1027–1035, 2003.
[13] F. Blanchini and S. Miani. A universal class of smooth functions for robust control. *IEEE Trans. Autom. Control*, 44(3):641–647, 1999.
[14] Sun Z. and Ge S. S. Analysis and synthesis of switched linear control systems. *Automatica J. IFAC*, 41(2):181–195, 2005.
[15] F. Blanchini, S. Miani, and C. Savorgnan. Stability results for continuous and discrete time linear parameter varying systems. In *Proc.* 16^{th} *IFAC world congress*, Prague, Czech Republic, 2005.
[16] F. Blanchini. Non-quadratic lyapunov function for robust control. *Automatica J. IFAC*, 31(3):451–461, 1995.
[17] F. Blanchini and S. Miani. Stabilization of LPV systems: state feedback, state estimation and duality. *SIAM J. Control Optim.*, 42(1):76–97, 2003.
[18] F. Blanchini. The gain scheduling and the robust state feedback stabilization problems. *IEEE Trans. Automat Control*, 45(11):2061–2070, 2000.
[19] F. Blanchini and S. Miani. Constrained stabilization of continuous-time linear systems. *Systems & Control Letters*, 28(2):95–102, 1996.
[20] F. Blanchini, S. Miani, and F. A. Pellegrino. Suboptimal receding horizon control for continuous-time systems. *IEEE Trans. Autom. Control*, 48(6):1081–1086, 2003.
[21] G. M. Ziegler. *Lectures on Polytopes*. Springer-Verlag, Berlin, 1995.
[22] Jones C. N., Kerrigan E. C., and Maciejowski J. M. Equality set projection: A new algorithm for the projection of polytopes in halfspace representation. Technical report, Department of Engineering, University of Cambridge, http://www-control.eng.cam.ac.uk/Homepage/publication.php?id=1, March 2004. CUED/F-INFENG/TR.463.
[23] C. Savorgnan. *MAXIS-G*. http://www.diegm.uniud.it/savorgnan or http://www.diegm.uniud.it/smiani, 2005.
[24] B Pluymers, J.A. Rossiter, J. Suykens, and B. De Moor. The efficient computation of polyhedral invariant sets for linear systems with polytopic uncertainty. In *Proceedings of the American Control Conference*, pages 804–809, Portland, 2005.