Guaranteed control of switched control systems using model order reduction and state-space bisection

Adrien Le Coënt[1], Florian de Vuyst[1], Christian Rey[2,3], Ludovic Chamoin[2], and Laurent Fribourg[4]

1 **CMLA, ENS Cachan & CNRS**
61 Av. du Président Wilson, 94235 Cachan Cedex, France
`adrien.le-coent,devuyst@cmla.ens-cachan.fr`

2 **LMT Cachan, ENS Cachan & CNRS**
61 Av. du Président Wilson, 94235 Cachan Cedex, France
`rey,chamoin@lmt.ens-cachan.fr`

3 **Safran Tech**
1 Rue Geneviève Aubé, 78772 Magny les Hameaux, France
`christian.rey@safran.fr`

4 **LSV, ENS Cachan & CNRS**
61 Av. du Président Wilson, 94235 Cachan Cedex, France
`fribourg@lsv.ens-cachan.fr`

Abstract

This paper considers discrete-time linear systems controlled by a *quantized* law, i.e., a piecewise constant time function taking a finite set of values. We show how to generate the control by, first, applying *model reduction* to the original system, then using a "state-space bisection" method for synthesizing a control at the reduced-order level, and finally computing an upper bound to the deviations between the controlled output trajectories of the reduced-order model and those of the original model. The effectiveness of our approach is illustrated on several examples of the literature.

1998 ACM Subject Classification B.1.2 Control Structure Performance Analysis and Design Aids, B.1.3 Control Structure Reliability, Testing, and Fault-Tolerance, B.2.3 Reliability, Testing, and Fault-Tolerance

Keywords and phrases Model Order Reduction, guaranteed control, stability control, reachability control, error bounding.

Digital Object Identifier 10.4230/OASIcs.SynCoP.2015.XXX

1 Introduction

We are focusing here on switched control systems, a class of hybrid systems recently used with success in various domains such as automotive industry and power electronics. Several strategies have been developped to design control laws for such systems; we use here the invariance analysis [9, 8, 10]. The associated algorithms are very expensive and require a limited state space dimension; we thus use a model order reduction in order to synthesize a controller at the reduced-order level. Two methods are proposed to apply the controller to the full-order system. Offline and online controls are enabled, and the computation of upper bounds of the error induced by the reduction allowed to guarantee the effectiveness of the controller.

Comparison with related work. Model order reduction techniques for hybrid or switched systems are classically used in *numerical simulation* in order to construct at the reduced level trajectories which cannot be computed directly at the original level due to complexity and large size dimension [3].

Han and Krogh make use of model reduction in order to perform *set-based reachability analysis* [15]. They do not construct isolated trajectories issued from isolated points, but (an overapproximation of) the infinite set of trajectories issued from a dense set of initial points. This allows them to perform formal verification of properties such as *safety*. In both approaches, the control is *given* as an input of the problem. In contrast here, the control is *synthesized* using set-based methods in order to achieve by construction properties such as *convergence* and *stability*.

The problem of control synthesis for hybrid and switched systems has been widely studied and various tools exist. The Multi-Parametric Toolbox (MPT 3.0 [17]) for example solves optimal control problems using operations on polytopes. Most approaches make use of Lyapunov or the so-called "multiple Lyapunov functions" to solve the problem of control synthesis for switched systems - see for example [24]. The approximate bisimulation approach abstracts switched systems under the form of a discrete model [14, 12] under certain Lyapunov-based stability conditions. The latter approach has been implemented in PESSOA [18] and CoSyMA [20]. The approach used in this paper avoids using Lyapunov functions and relies on the notion of "(controlled) invariant" [7].

Plan. In Section 2, we give some preliminaries about linear controlled systems and reachability sets. In Section 3, we recall the principles of the state-space bisection method. In Section 4, we explain how to construct a reduced model, apply the state-space bisection method at this level, and compute upper bounds to the error induced at the original level. In Section 5, we propose two methods of control synthesis allowing to synthesize (either offline or online) a controller at the reduced-order level and apply it to the full-order system. In Section 6, we apply our approach to several examples of the literature. We conclude in Section 7.

2 Background

We consider a class of control systems composed of a plant and a controller defined as follows. The plant is a discrete-time linear time invariant (DLTI) system Σ defined by (see [4, 24] for more information on DLTI and sampled switched systems):

$$\Sigma : \begin{cases} x(t+\tau) & = A_d x(t) + B_d u(t), \\ y(t) & = C_d x(t). \end{cases} \tag{1}$$

Here, the state x is an n-vector, the control input u a p-vector, the output y an m-vector, and A_d an $n \times n$-matrix, B_d an $n \times p$-matrix, C_d a $m \times n$ matrix. The real positive constant τ is the time sampling parameter. All the coefficients are reals. The system Σ is obtained from the temporal discretization of the continuous linear time invariant (LTI) system:

$$\begin{cases} \dot{x}(t) & = Ax(t) + Bu(t), \\ y(t) & = Cx(t), \end{cases}$$

with

$$A_d = e^{A\tau}, \quad B_d = \int_0^{\tau} e^{A(\tau - t)} B dt, \quad C_d = C.$$

We consider here the case of a *quantized* control (see, e.g. [22]). This means that the control law $\mathbf{u} \equiv u(\cdot)$ is a *piecewise constant* function that changes its value at each sampling time $0, \tau, 2\tau, \ldots$ Furthermore, $\mathbf{u}$ can take only a *finite* number of vector values. We denote by U the finite set of values taken by $\mathbf{u}$, every element of U is called a *mode*. The problem of (discretized) quantized control is to find a *state-dependent* law $u(t)$ which, at every sampling time, finds the element of U that allows to achieve a given goal, such as the stabilization around an objective point y_{obj}. Actually, the stabilization of such systems cannot be perfect [22], and we thus look for *practical* stability: we do not look for an equilibrium point y_{obj}, but only for a neighborhood of y_{obj} in which we confine the system. We note that the controller in the above model implements a state feedback law and has only discrete-state dynamics.

The entries of the problem are the following:

1. a subset $R_x \subset \mathbb{R}^n$ of the state space, called *interest set*,
2. a subset $R_y \subset \mathbb{R}^m$ of the output space, called *objective set*.

The objective is to find a law $u(\cdot)$ which, for any initial state $x_0 \in R_x$, stabilizes the output y in the set R_y.

Remark: In [10], we have proposed a procedure, called *state-space bisection procedure*[1], in order to achieve a similar goal. The context was simpler since we considered there only the state equation $x(t+\tau) = A_d x(t) + B_d u(t)$ without the output equation $y(t) = C_d x(t)$. Here, the stabilization problem is naturally extended to take into account the output $y(t)$. Note that even if the initial state x_0 belongs to R_x, the corresponding output y_0 does not necessarily belong to R_y.

The state-space bisection procedure being subject to the so-called *curse of dimensionality*, it will be applied here to a reduced order model $\hat{\Sigma}$ of dimension $n_r < n$ in order to stabilize its output in the objective set R_y. We will show that the control law synthesized on the reduced system $\hat{\Sigma}$ still stabilizes the output of the original system Σ with a tolerance $\varepsilon > 0$. We now introduce some notations required to explain the state-space bisection procedure.

Some notations. We will use $\mathbf{x}(t, x, u)$ to denote the point reached by Σ at time t under mode $u \in U$ from the initial condition x. This gives a transition relation $\to_u^\tau$ defined for x and x' in $\mathbb{R}^n$ by: $x \to_u^\tau x'$ iff $\mathbf{x}(\tau, x, u) = x'$. Given a set $X \subset \mathbb{R}^n$, we define the *successor set* of a set $X \subset \mathbb{R}^n$ of states under mode u as:

$$Post_u(X) = \{x' \mid x \to_u^\tau x' \text{ for some } x \in X\}.$$

The set $Post_u(X)$ is then the result of the affine transformation $A_d X + B_d u$. Likewise, we define the *output successor set* of a set $X \subset \mathbb{R}^n$ of states under mode u as:

$$Post_{u,C}(X) = \{Cx' \mid x \to_u^\tau x' \text{ for some } x \in X\}.$$

An *input pattern* named Pat is defined as a finite sequence of modes. A k*-pattern* is an input pattern of length at most k. The successor set of $X \subset \mathbb{R}^n$ using $Pat \equiv (u_1 \cdots u_k)$ is defined by

$$Post_{Pat}(X) = \{x' \mid x \to_{u_1}^\tau \cdots \to_{u_k}^\tau x',\ x \in X\}.$$

[1] In [9, 8, 10], this method was called “state-space decomposition”, but we use here “state-space bisection” in order to avoid ambiguity with model reduction methods.

The mapping $Post_{Pat}$ is itself an affine transformation. The output successor set of $X \subset \mathbb{R}^n$ using $Pat \equiv (u_1 \cdots u_k)$ is defined by

$$Post_{Pat,C}(X) = \{Cx' \mid x \to^{\tau}_{u_1} \cdots \to^{\tau}_{u_k} x',\ x \in X\}.$$

Given an input pattern Pat of the form $(u_1 \cdots u_m)$, and a set $X \subset \mathbb{R}^n$, the *unfolding of* X *via* Pat, denoted by $Unf_{Pat}(X)$, is the set $\bigcup_{i=0}^{m} X_i$ with:

- $X_0 = X$,
- $X_{i+1} = Post_{u_{i+1}}(X_i)$, for all $0 \le i \le m-1$.

The unfolding thus corresponds to the set of all the intermediate states produced when applying the input pattern Pat to the states of X.

3 Control Synthesis by State-Space Bisection

We now explain the method that we are going to use in order to find a control law $\mathbf{u}$ that stabilizes the state x of the system Σ into a given zone $R_x \subset \mathbb{R}^n$ [10], and makes the output y reach a given zone $R_y \subset \mathbb{R}^n$.

3.1 x-stabilization and y-convergence requirements

Given an interest set R_x and an objective set R_y, we can define the notion of "x-stabilization" and "y-convergence" in this context as follows.

▶ **Definition 1.** Given a system Σ, a set R_x subset of $\mathbb{R}^n$, a set R_y subset of $\mathbb{R}^m$, and a positive integer k, an *x-stabilizing and y-convergent control* for (R_x, R_y, k) with respect to Σ is a function that associates to each $x \in R_x$ a k-pattern Pat such that:

- $Post_{Pat}(\{x\}) \subseteq R_x$,
- $Post_{Pat,C}(\{x\}) \subseteq R_y$.

Note that we use the term "y-convergence" because the output corresponding to the initial state can possibly be outside R_y, whereas the initial state necessarily belongs to R_x. Given a system Σ, an x-stabilizing and y-convergent control guarantees that all the trajectories starting at R_x return to R_x within k steps and that the output is sent into R_y. In order to find an x-stabilizing and y-convergent control, we can adapt the method of "state-space bisection" introduced in [8].

▶ **Definition 2.** Given a system Σ, two sets R_x and R_y respectively subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$, and a positive integer k, a *successful decomposition of* (R_x, R_y, k) *w.r.t.* Σ is a set Δ of the form $\{V_i, Pat_i\}_{i\in I}$, where I is a finite set of indices, every V_i is a subset of R_x, and every Pat_i is a k-pattern such that:

(a) $\bigcup_{i\in I} V_i = R_x$,
(b) for all $i \in I$: $Post_{Pat_i}(V_i) \subseteq R_x$,
(c) for all $i \in I$: $Post_{Pat_i,C}(V_i) \subseteq R_y$.

Remark: Note that in pratice, the condition $Post_{Pat_i,C}(V_i) \subseteq R_y$ is verified by verifying that the bounding box of $Post_{Pat_i,C}(V_i)$ (that is the smallest square box containing $Post_{Pat_i,C}(V_i)$) belongs to R_y. All the set-based operations are carried out with zonotopes. See [1] for the computation of the bounding box of a zonotope and operations realized on zonotopes.

A successful decomposition $\Delta = \{(V_i, Pat_i)\}_{i\in I}$ naturally induces a *state-dependent control* on R_x. The control induced by Δ is defined as follows: consider an initial point x_0

of R_x; we know that $x_0 \in V_i$ for some $i \in I$ (since $R_x = \bigcup_{i \in I} V_i$). One thus applies Pat_i to x_0, which gives a new state x_1 that belongs itself to R_x, and the associated output belongs to R_y (since $Post_{Pat_i}(V_i) \subseteq R_x$ and $Post_{Pat_i,C}(V_i) \subseteq R_y$). The process can then be repeated on x_1, and so on iteratively. Obviously, the induced control is an x-stabilizing and y-convergent control for (R_x, R_y, k). Formally, we have:

▶ Proposition 1. Suppose that Δ is a successful decomposition of (R_x, R_y, k) w.r.t. Σ. Then the control induced by Δ is an x-stabilizing and y-convergent control for (R_x, R_y, k) w.r.t. Σ.

The problem of finding an x-stabilizing and y-convergent controller thus reduces to the problem of finding a successful decomposition. The latter problem can be solved by using the method of *state-space bisection* [8], as explained below.

3.2 Bisection method

We give here a simple algorithm, adapted from [8], called Bisection algorithm. Given two sets R_x and R_y, and a positive integer k, the algorithm, when it succeeds, provides for a successful decomposition Δ of (R_x, R_y, k) w.r.t. Σ of the form $\{V_i, Pat_i\}_{i \in I}$. The input sets R_x and R_y are given under the form of *boxes* of $\mathbb{R}^n$ and $\mathbb{R}^m$ (i.e., cartesian products of n closed intervals for R_x, and cartesian products of m closed intervals for R_y). The subsets V_i, $i \in I$, of R_x are boxes that are obtained by repeated bisection. At the beginning, the Bisection procedure calls sub-procedure Find_Pattern in order to get a k-pattern Pat such that $Post_{Pat}(R_x) \subseteq R_x$ and $Post_{Pat,C}(R_x) \subseteq R_y$. If it succeeds, then it is done. Otherwise, it divides R_x into 2^n sub-boxes $V_1, \dots, V_{2^n}$ of equal size. If for each V_i, Find_Pattern gets a k-pattern Pat_i such that $Post_{Pat_i}(V_i) \subseteq R_x$ and $Post_{Pat_i,C}(V_i) \subseteq R_y$, it is done. If, for some V_j, no such input pattern exists, the procedure is recursively applied to V_j. It ends with success when a successful decomposition of (R_x, R_y, k) is found, or failure when the maximal degree d of bisection is reached. The algorithmic form of the procedure is given in Algorithms 1 and 2. The main procedure Bisection(W, R_x, R_y, D, K) is called with R_x as input value for W, d for input value for D, and k as input value for K; it returns either $\langle \{(V_i, Pat_i)\}_i, True \rangle$ with $\bigcup_i V_i = W$, $\bigcup_i Post_{Pat_i}(V_i) \subseteq R_x$, $\bigcup_i Post_{Pat_i,C}(V_i) \subseteq R_y$ when it succeeds, or $\langle \_, False \rangle$ when it fails. Procedure Find_Pattern(W,R_x,R_y,K) looks for a K-pattern Pat for which $Post_{Pat}(W) \subseteq R_x$ and $Post_{Pat,C}(W) \subseteq R_y$: it selects all the K-patterns by increasing length order until either it finds such an input pattern Pat (output: $\langle Pat, True \rangle$), or none exists (output: $\langle \_, False \rangle$). The correctness of the procedure is stated as follows.

▶ **Theorem 3.** *If Bisection(R_x,R_x,R_y,d,k) returns $\langle \Delta, True \rangle$, then Δ is a successful decomposition of (R_x, R_y, k) w.r.t. Σ.*

In [8], we have developed a tool that implements the Bisection procedure, using zonotopes (see [13]); it is written in Octave [21].

4 Model Order Reduction

Actually, because of the computational cost of the bisection procedure, the application of the bisection method at the full-order level n becomes rapidly intractable (typically for $n \geq 15$). Therefore, it is interesting to apply *projection-based* model order reduction methods (see [3]), then construct decompositions (hence control laws) at the reduced state level of dimension $n_r < n$ rather than at the full-order state level of dimension n. For many

```
Algorithm 1: Bisection($W, R_x, R_y, D, K$)
                                   /* with additional input $\varepsilon_x$ for online version */
Input: A box $W$, a box $R_x$, a box $R_y$, a degree $D$ of bisection, a length $K$ of input
       pattern
Output: $\langle\{(V_i, Pat_i)\}_i, True\rangle$ with $\bigcup_i V_i = W$, $\bigcup_i Post_{Pat_i}(V_i) \subseteq R_x$ and
        $\bigcup_i Post_{Pat_i,C}(V_i) \subseteq R_y$, or $\langle\_, False\rangle$
1  $(Pat, b) := Find\_Pattern(W, R_x, R_y, K)$
2  if $b = True$ then
3      return $\langle\{(W, Pat)\}, True\rangle$
4  else
5      if $D = 0$ then
6          return $\langle\_, False\rangle$
7      else
8          Divide equally $W$ into $(W_1, \dots, W_{2^n})$
9          for $i = 1 \dots 2^n$ do
10             $(\Delta_i, b_i)$ := Bisection($W_i$,$R_x$,$R_y$,$D-1$,$K$)
11         return $(\bigcup_{i=1\dots 2^n} \Delta_i, \bigwedge_{i=1\dots 2^n} b_i)$
```

```
Algorithm 2: Find_Pattern($W, R_x, R_y, K$)
                                   /* with additional input $\varepsilon_x$ for online version */
Input: A box $W$, a box $R_x$, a box $R_y$, a length $K$ of input pattern
Output: $\langle Pat, True\rangle$ with ,$Post_{Pat}(W) \subseteq R_x$,$Post_{Pat,C}(W) \subseteq R_y$ and
        $Unf_{Pat}(W) \subseteq S$, or $\langle\_, False\rangle$ when no input pattern maps $W$ into $R_x$ and
        $CW$ into $R_y$
1  for $i = 1 \dots K$ do
2      $\Pi$ := set of input patterns of length $i$
3      while $\Pi$ is non empty do
4          Select $Pat$ in $\Pi$
5          $\Pi := \Pi \setminus \{Pat\}$
6          if $Post_{Pat}(W) \subseteq R_x$ and $Post_{Pat,C}(W) \subseteq R_y$ /* condition modified for online
           version */ then
7              return $\langle Pat, True\rangle$
8  return $\langle\_, False\rangle$
```

applications and engineering problems, it is observed that the systems can be reduced while being accurate enough. This is due to the underlying regularity of the solutions or, in other words, to the low dimension of the actual trajectory submanifold.

Given a full-order system Σ, an interest set $R_x \subset \mathbb{R}^n$ and an objective set $R_y \subset \mathbb{R}^m$, we will construct a reduced-order system $\hat{\Sigma}$ using a projection π of $\mathbb{R}^n$ to $\mathbb{R}^{n_r}$. If $\pi \in \mathbb{R}^{n\times n}$ is a projection, it verifies $\pi^2 = \pi$, and π can be written as $\pi = \pi_L \pi_R$, where $\pi_L \in \mathbb{R}^{n\times n_r}$, $\pi_R \in \mathbb{R}^{n_r\times n}$ and $n_r = rank(\pi)$. The DLTI system $\hat{\Sigma}$ is defined by the matrices $\hat{A}_d$, $\hat{B}_d$, $\hat{C}_d$, and can be written:

$$\hat{\Sigma} : \begin{cases} \hat{x}(t+\tau) & = \hat{A}_d \hat{x}(t) + \hat{B}_d u(t), \\ y_r(t) & = \hat{C}_d \hat{x}(t), \end{cases}$$

with

$$\hat{A}_d = e^{\hat{A}\tau}, \quad \hat{B}_d = \int_0^\tau e^{\hat{A}(\tau - t)} \hat{B} dt, \quad \hat{C}_d = \hat{C}.$$

The matrices $\hat{A}$, $\hat{B}$ and $\hat{C}$ are defined as follows:

$$\hat{A} = \pi_R A \pi_L, \quad \hat{B} = \pi_R B, \quad \hat{C} = C \pi_L.$$

Here, $\hat{x}$ is an n_r-vector, u a p-vector, y_r an m-vector, and $\hat{A}_d$ an $n_r \times n_r$-matrix, $\hat{B}_d$ an $n_r \times p$-matrix, $\hat{C}_d$ a $m \times n_r$ matrix. The reduced system is obtained with the change of variable: $\hat{x} = \pi_R x$. Accordingly, $\hat{R}_x = \pi_R R_x \subset \mathbb{R}^{n_r}$ is the projection of R_x. The objective set R_y is kept unchanged. Using the method described in Section 3, one generates a successful decomposition $\hat{\Delta}$ of $(\hat{R}_x, R_y, k)$ w.r.t. $\hat{\Sigma}$ for some given k. This leads to a reduced-order control $\mathbf{u}_{\hat{\Delta}}$. By Theorem 3, $\mathbf{u}_{\hat{\Delta}}$ sends the output y_r in R_y. When this control is applied to the full-order system Σ, this leads to a trajectory $y(t)$. The difference between the two trajectories $y(t)$ and $y_r(t)$ will be denoted by $e(t)$. An upper bound of $\|e(t)\|$ will be computed in order to assess the deviation from R_y. We will denote by ε_y^j an upper bound of this error for $t = j\tau$, and we will denote by ε_y^∞ the maximum value of this bound: $\varepsilon_y^\infty = \sup_{j \geq 0} \varepsilon_y(j\tau)$ (see Appendix for the calculation of these bounds).

5 Reduced Order Control

In this section, we first explain the procedure of control synthesis, then we propose a method to guarantee that the obtained controller sends the output of the full-order system in R_y with a tolerance ε_y^∞.

5.1 Guaranteed offline control

Suppose that we are given a system Σ, an interest set R_x, and an objective set R_y. The procedure first consists in reducing the system Σ of order n to a system $\hat{\Sigma}$ of order $n_r < n$ (see section 4). Here, the classical method of balanced truncation [5, 2, 19, 6] is used to construct π.

We apply the state-space bisection procedure to the reduced-order system $\hat{\Sigma}$, we obtain a successful decomposition $\hat{\Delta}$ of $(\hat{R}_x, R_y, k)$ w.r.t. $\hat{\Sigma}$. Therefore, the procedure returns a successful decomposition $\hat{\Delta}$ of the form $\{\hat{V}_i, Pat_i\}_{i \in I}$ such that:

1. I is a finite set of indices,
2. every $\hat{V}_i$ $(i \in I)$ is an interval product of dimension n_r such that $\bigcup_{i \in I} \hat{V}_i = \hat{R}_x$,
3. every Pat_i $(i \in I)$ is a k-pattern such that for all $i \in I$: $Post_{Pat_i}(\hat{V}_i) \subseteq \hat{R}_x$ and $Post_{Pat_i, \hat{C}}(\hat{V}_i) \subseteq R_y$.

The successful decomposition $\hat{\Delta}$ induces a control $u_{\hat{\Delta}}$ on $\hat{R}_x$. This control $u_{\hat{\Delta}}$ is $\hat{x}$-stabilizing and y_r-convergent for $(\hat{R}_x, R_y, k)$ w.r.t. $\hat{\Sigma}$. Let x_0 be an initial condition in R_x. Let $\hat{x}_0 = \pi_R x_0$ be its projection belonging to $\hat{R}_x$, $\hat{x}_0 = \pi_R x_0$ is the initial condition for the reduced system $\hat{\Sigma}$: $\hat{x}_0$ belongs to $\hat{V}_{i_0}$ for some $i_0 \in I$; thus, after applying Pat_{i_0}, the system is led to a state $\hat{x}_1$; $\hat{x}_1$ belongs to $\hat{V}_{i_1}$ for some $i_1 \in I$; and iteratively, we build, from an initial state $\hat{x}_0$, a sequence of states $\hat{x}_1, \hat{x}_2, \ldots$ obtained by application of the sequence of k-patterns Pat_{i_0}, Pat_{i_1}, ... (steps (1), (2) and (3) of Figure 1).

The sequence of k-patterns is computed for the reduced system $\hat{\Sigma}$, but it can be applied to the full-order system Σ: we build, from an initial point x_0, a sequence of points x_1, x_2,... by application of the k-patterns Pat_{i_0},Pat_{i_1},... (steps (4), (5) and (6) of Figure 1). For

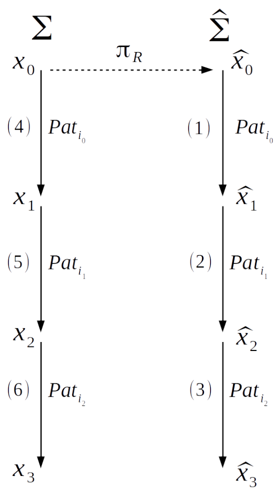

Figure 1 Diagram of the offline procedure for a simulation of length 3.

all $x_0 \in R_x$ and for all $t \geq 0$, the error $\|y(x_0, u, t) - y_r(\pi_R x_0, u, t)\|$ is bounded by ε_y^∞, as defined in Appendix). This leads to the following proposition:

▶ Proposition 2. Let us consider a DLTI system Σ, an interest set R_x, and an objective set R_y. Let $\hat{\Sigma}$ be the projection by balanced truncation of Σ. Let $\hat{\Delta}$ be a successful decomposition of $(\hat{R}_x$,R_y,$k)$ w.r.t. $\hat{\Sigma}$. Then, for all $x_0 \in R_x$, the induced control $u_{\hat{\Delta}}$ applied to the full-order system Σ in x_0 is such that for all $j > 0$, the output of the full-order system $y(t)$ returns to $R_y + \varepsilon_y^\infty$ after at most k τ-steps.

Here, $R_y + \varepsilon_y^\infty$ denotes the set containing R_y with a margin of ε_y^∞. More precisely, if R_y is an interval product of the form $[a_1, b_1] \times \cdots \times [a_m, b_m]$, then $R_y + \varepsilon_y^\infty$ is defined by $[a_1 - \varepsilon_y^\infty, b_1 + \varepsilon_y^\infty] \times \cdots \times [a_m - \varepsilon_y^\infty, b_m + \varepsilon_y^\infty]$.

Remark: Here, we ensure that $y(x_0, u, t)$ is in $R_y + \varepsilon_y^\infty$ at the end of every input pattern, but an easy improvement is to ensure that $y(x_0, u, t)$ stays in a given *safety set* $S_y \supset R_y$ at every step of time τ. Indeed, as explained in [8], we can ensure that the unfolding of the output trajectory stays in S_y. In order to guarantee that $y(x_0, u, t)$ stays in S_y, we just have to make sure that $y_r(\pi_R x_0, u, t)$ stays in the reduced safety set $S_y - \varepsilon_y^\infty$. We thus have to add the condition: "and $Unf_{Pat}(CW) \subset S_y - \varepsilon_y^\infty$" in the line 6 of Algorithm 2.

5.2 Guaranteed Online control

Up to this point, the procedure of control synthesis consists in computing a complete sequence of input patterns on the reduced order model $\hat{\Sigma}$ for a given initial state x_0, and applying the input pattern sequence to the full-order model Σ. The control law is thus computed offline. However, using the bisection method applied to the reduced system $\hat{\Sigma}$, we can use the decomposition $\hat{\Delta}$ online as follows: Let x_0 be the initial state in R_x and $\hat{x}_0 = \pi_R x_0$ its projection belonging to $\hat{R}_x$ (step (1) of Figure 2); $\hat{x}_0$ belongs to $\hat{V}_{i_0}$ for some $i_0 \in I$; we can thus apply the associated input pattern Pat_{i_0} to the full-order system Σ, which yields a state $x_1 = Post_{Pat_{i_0}}(x_0)$ (step (2) of Figure 2); the corresponding output is sent to $y_1 = Post_{Pat_{i_0},C}(x_0) \in R_y + \varepsilon_y^{\ell_0}$; in order to continue to step (3), we have to guarantee that $\pi_R Post_{Pat_i}(x)$ belongs to $\hat{R}_x$ for all $x \in R_x$ and for all $i \in I$. As explained below, this is possible using the computation of an upper bound to the error $\|\pi_R Post_{Pat_i}(x) - Post_{Pat_i}(\pi_R x)\|$

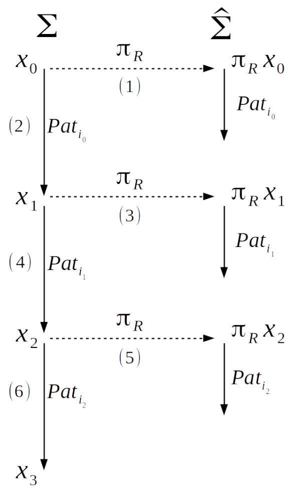

Figure 2 Diagram of the online procedure for a simulation of length 3.

and a reinforcement of the procedure for taking into account this error.

Let ε_x^j be an upper bound to $\|\pi_R Post_{Pat}(x) - Post_{Pat}(\pi_R x)\|$, j being the length of the input pattern Pat (see Appendix for the calculation of this bound). We modify the Algorithms 1 and 2 by adding a new input $\varepsilon_x = (\varepsilon_x^1, \dots, \varepsilon_x^k)$, k being the maximal length of the input patterns. With such an additional input, we define an ε-decomposition as follows:

▶ **Definition 4.** Given a system Σ, two sets R_x and R_y respectively subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$, a positive integer k, and a vector of errors $\varepsilon_x = (\varepsilon_x^1, \dots, \varepsilon_x^k)$, an ε-decomposition of $(R_x, R_y, k, \varepsilon_x)$ w.r.t. Σ is a set Δ of the form $\{V_i, Pat_i\}_{i \in I}$, where I is a finite set of indices, every V_i is a subset of R_x, and every Pat_i is a k-pattern such that:

(a') $\bigcup_{i \in I} V_i = R_x$,

(b') for all $i \in I$: $Post_{Pat_i}(V_i) \subseteq R_x - \varepsilon_x^{|Pat_i|}$,

(c') for all $i \in I$: $Post_{Pat_i,C}(V_i) \subseteq R_y$.

Note that condition (b') is a strenghtening of condition (b) of definition 2. Accordingly, line 6 of Algorithm 2 is modified as follows:

```
6   if $Post_{Pat}(W) \subseteq R_x - \varepsilon_x^i$ and $Post_{Pat,C}(W) \subseteq R_y$ then
```

The computation of an ε-decomposition with the modified algorithms enables to guarantee that the projection $\pi_R x$ of the full-order system state always stays in $\hat{R}_x$. We can thus perform the online control as follows:

Since $Post_{Pat_{i_0}}(\hat{V}_{i_0}) \subseteq \hat{R}_x - \varepsilon_x^{\ell_0}$ and $\pi_R x_0 \in \hat{V}_{i_0}$, we have $Post_{Pat_{i_0}}(\pi_R x_0) \in \hat{R}_x - \varepsilon_x^{\ell_0}$; thus $\pi_R x_1 = \pi_R Post_{Pat_{i_0}}(x_0)$ belongs to $\hat{R}_x$, because $\varepsilon_x^{\ell_0}$ is a bound of the maximal distance between $Post_{Pat_{i_0}}(\pi_R x_0)$ and $\pi_R Post_{Pat_{i_0}}(x_0)$; since $\pi_R x_1$ belongs to $\hat{R}_x$, it belongs to V_{i_1} for some $i_1 \in I$; we can thus compute the input pattern Pat_{i_1}, and we can thus reapply the procedure and compute an input pattern sequence Pat_{i_0},Pat_{i_1},...

We finally have the proposition:

▶ Proposition 3. Let us consider a DLTI system Σ, an interest set R_x, and an objective set R_y. Let $\hat{\Sigma}$ be the projection by balanced truncation of Σ. Let $\hat{\Delta} = \{\hat{V}_i, Pat_i\}_{i \in I}$ be an ε-decomposition for ($\hat{R}_x$,R_y,k,ε_x) w.r.t. $\hat{\Sigma}$, ε_x being defined as above. Then:

$$\forall x \in R_x, \exists i \in I : \quad \pi_R Post_{Pat_i}(x) \in \hat{R}_x \quad \wedge \quad Post_{Pat_i,C}(x) \in R_y + \varepsilon_y^{|Pat_i|}.$$

Using the decomposition $\hat{\Delta}$, we can perform an online control as explained above. We have:

▶ Proposition 4. Let us consider a DLTI system Σ, an interest set R_x, and an objective set R_y. Let $\hat{\Sigma}$ be the projection by balanced truncation of Σ. Let $\hat{\Delta}$ be a ε-decomposition for $(\hat{R}_x,R_y,k,\varepsilon_x)$ w.r.t. $\hat{\Sigma}$. Then, for all $x_0 \in R_x$, the induced online control $u_{\hat{\Delta}}$ yields an output sequence of points y_1,y_2,... which belong to $R_y + \varepsilon_y^{\ell_0}$,$R_y + \varepsilon_y^{\ell_1}$,... where ℓ_0, ℓ_1, ... are the lengths of the input patterns successively applied.

The advantage of such an online control is that the estimated errors $\varepsilon_y^{\ell_0}$,$\varepsilon_y^{\ell_1}$,... are dynamically computed, and are smaller than the static bound ε_y^{∞} used in the offline control. The price to be paid is the strenghtening of the $\hat{x}$-stabilization condition (b') of definition 4.

6 Case Studies

6.1 Distillation Column

We consider a linearized model of a distillation column system [25] written under the form of a DLTI system (1). The state $x = (x_1, x_2, \ldots, x_{11})^\top$ of the system is of dimension 11: $x_1, x_2, \ldots, x_{10}$ correspond to the composition of the most volatile component in the different stages of the column, and x_{11} corresponds to the pressure at the top of the column. The perturbation of input feed ω is neglected. The control variable $u \in U = \{0, 1\}$ corresponds to the state turned on (1) or turned off (0) of the reheater. The output y is of dimension 1 and corresponds to the composition of the most volatile component in the bottom product. The system is reduced from $n = 11$ to $n_r = 2$. The sampling time is set to $\tau = 100$ s.

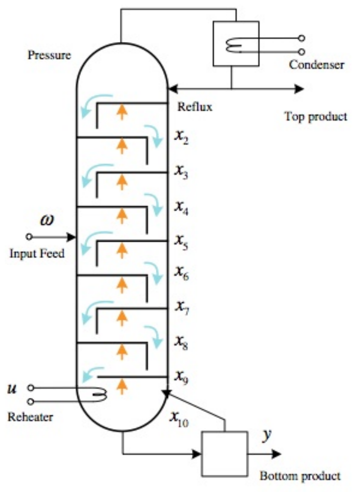

Figure 3 The 11-order distillation column system

The matrices A,B, and C are the following:

$$A = 10^{-2} \times \begin{bmatrix} -1.4 & -0.43 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.95 & -1.38 & 0.46 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.05 \\ 0 & 0.95 & -1.41 & 0.63 & 0 & 0 & 0 & 0 & 0 & 0 & 0.02 \\ 0 & 0 & 0.95 & -1.58 & 1.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.95 & -3.12 & 1.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.02 & -3.52 & 2.2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2.02 & -4.22 & 2.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.02 & -4.82 & 3.7 & 0 & 0.02 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2.02 & -5.72 & 4.2 & 0.05 \\ 2.55 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2.55 & -1.85 \end{bmatrix},$$

$B = [0\ 0\ 0\ 0\ 0.01\ 0\ 0\ 0\ 0\ 0\ 0]^T$ and $C = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0.01\ 0]^T$.

The interest set is $R_x = [0,1]^{11}$. The objective set is $R_y = [0.0015, 0.0025]$. The two methods presented in this paper give the results of Figure 4. The circles in the simulations correspond to the end of input patterns. The simulations have been performed with MINIMATOR (an Octave code available at https://bitbucket.org/alecoent/minimator_red) on a 2.80 GHz Intel Core i7-4810MQ CPU with 8 GB of memory. The offline and online methods led to the same decomposition because of the contractive behaviour of the system. The decompositions were obtained in 106 seconds.

Figure 4 shows simulations of the offline and online methods. The initial point $x_0 = (0.55)^{11}$ is in R_x. Note that the corresponding output $y_0 = Cx_0 = 5.5 \times 10^{-3}$ lies outside R_y. For the offline method (on the left), the output y_r of the reduced system (continuous blue) is sent into R_y (dashed blue), and the output y of the full-order system (continuous red) is sent in $R_y + \varepsilon_y^\infty$ (dashed red). For the online method (on the right), the output y_r of the reduced system (continuous blue) is sent into R_y (dashed blue), and the output y of the full-order system (continuous red) is sent in $R_y + \varepsilon_y^{\ell_i}$ (dashed red).

Both methods are thus efficient, the offline method is guaranteed but implies a relatively pessimistic tolerance. The online method is very efficient and the tolerances are much more satisfying. A shift can however appear between the full-order model and the reduced order model, this is an unavoidable consequence of the fact that the reduced order model does not represent the dynamic of the model as accurately as the full-order model.

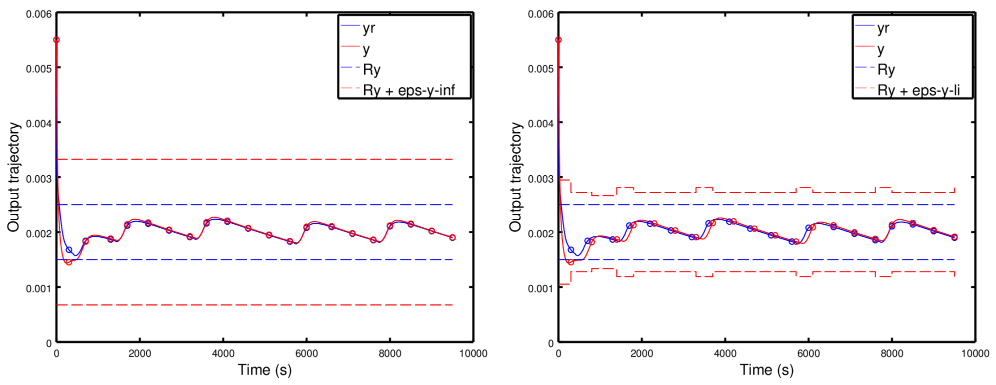

Figure 4 Simulation of $y(t) = Cx(t)$ and $y_r(t) = \hat{C}\hat{x}(t)$ from the initial condition $x_0 = (0.55)^{11}$. Left: guaranteed offline control; right: guaranteed online control.

6.2 Square Plate

We consider here the problem of controlling the central node temperature of a metal square plate discretized by finite elements. This example is taken from [16]. The square plate is subject to the heat equation: $\dfrac{\partial T}{\partial t}(x,t) - \alpha^2 \Delta T(x,t) = 0$. After discretization, the system is written under the form of a DLTI system (1). The plate is insulated along three edges, the right edge is open. The left half of the bottom edge is connected to a heat source. The exterior temperature is set to 0 °C, the temperature of the heat source is either 0 °C (mode 0) or 1 °C (mode 1). The heat transfers with the exterior and the heat source are modelled by a convective transfer. The full-order system state corresponds to the nodal temperatures. The output is the temperature of the central node. The system is reduced from $n = 897$

to $n_r = 2$. The interest set is $R_x = [0, 0.15]^{897}$, the objective set $R_y = [0.06, 0.09]$. The sampling time is set to $\tau = 8$ s. The geometry of the system and the decomposition obtained with the offline procedure are given in Figure 5. The decompositions were obtained in 5 seconds. Simulations of the offline and online methods are given in Figure 6. We notice that the trajectory y (resp. y_r) exceeds the objective set R_y (resp. $R_y + \varepsilon_y^{\ell_i}$) during the application of the second pattern, yet the markers corresponding to the end of input patterns do belong to objective sets.

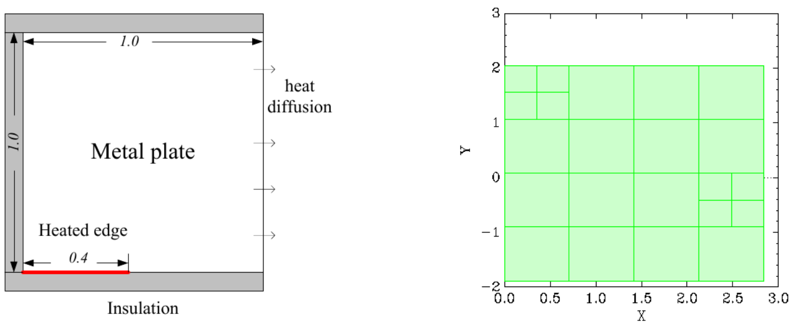

Figure 5 Geometry of the square plate (left) and decomposition of $\hat{R}_x = \pi_R R_x$ in the plane $(\hat{x}_1, \hat{x}_2)$ with the offline procedure (right).

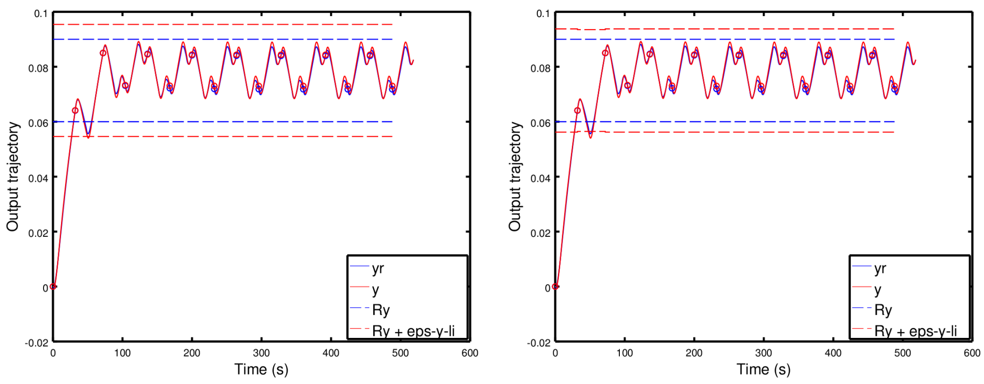

Figure 6 Simulation of $y(t) = Cx(t)$ and $y_r(t) = \hat{C}\hat{x}(t)$ from the initial condition $x_0 = (0)^{897}$. Left: guaranteed offline control; right: guaranteed online control.

7 Final Remarks

Two methods have been proposed to synthesize controllers for switched control systems using model order reduction and the state-space bisection procedure. An offline and an online use are enabled, both uses are efficient but they present different advantages. The offline method allows to obtain the same behaviour as the reduced-order model, but the associated bound is more pessimistic, and the controller has to be computed before the use of the real system. The online method leads to less pessimistic bounds but implies a behaviour slightly different from the reduced-order model, and the limit cycles may be different from those computed on the reduced system. The behaviour of the full-order system is thus less known, but its use can be performed in real time.

There are still some open questions associated to the methods proposed here. During a real online use, only the output y of the system Σ is known, this implies that a reconstruction of the reduced state $\hat{x}$ has to be performed online, either by reconstructing the full-order state x and projecting it, or by reconstructing directly the projected state. Until now, the reconstruction is supposed exact. Our future work will be devoted to the online reconstruction of $\hat{x}$, and this will be done with the use of extended Kalman filters [23, 11].

Acknowledgement. This work is supported by Institut Farman FR3311 through the project SWITCHDESIGN, and by the French National Research Agency (ANR) through the "iCODE Institute project" funded by the IDEX Paris-Saclay, ANR-11-IDEX-0003-02.

References

1 M. Althoff, O. Stursberg & M. Buss: *Verification of uncertain embedded systems by computing reachable sets based on zonotopes.*

2 A. Antoulas & D. C. Sorensen (2001): *Approximation of large-scale dynamical systems: an overview. International Journal of Applied Mathematics and Computer Science* 11(5), pp. 1093–1121.

3 A. Antoulas, D. C. Sorensen & S. Gugercin (2000): *A survey of model reduction methods for large-scale systems. Contemporary Mathematics* 280, pp. 193–219.

4 E. Asarin, O. Bournez, D. Thao, O. Maler & A. Pnueli (2000): *Effective synthesis of switching controllers for linear systems. Proceedings of the IEEE* 88(7), pp. 1011–1025, 10.1109/5.871306.

5 P. Benner, J.-R. Li & T. Penzl (2008): *Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems. Numerical Linear Algebra with Applications* 15(9), pp. 755–777.

6 P. Benner & A. Schneider (2010): *Balanced truncation model order reduction for LTI systems with many inputs or outputs.* In: *Proceedings of the 19th international symposium on mathematical theory of networks and systems–MTNS*, 5.

7 F. Blanchini (1999): *Set invariance in control. Automatica* 35(11), pp. 1747 – 1767.

8 L. Fribourg, U. Kühne & R. Soulat (2014): *Finite Controlled Invariants for Sampled Switched Systems. Formal Methods in System Design* 45(3), pp. 303–329, 10.1007/s10703-014-0211-2.

9 L. Fribourg & R. Soulat (2013): *Control of Switching Systems by Invariance Analysis: Application to Power Electronics.* Wiley-ISTE. 144 pages.

10 L. Fribourg & R. Soulat (2013): *Stability Controllers for Sampled Switched Systems.* In Parosh Aziz Abdulla & Igor Potapov, editors: *Proceedings of the 7th Workshop on Reachability Problems in Computational Models (RP'13), Lecture Notes in Computer Science* 8169, Springer, Uppsala, Sweden, pp. 135–145, 10.1007/978-3-642-41036-9_13.

11 B. P. Gibbs (2011): *Advanced Kalman filtering, least-squares and modeling: a practical handbook.* John Wiley & Sons.

12 A. Girard, G. Pola & P. Tabuada (2010): *Approximately Bisimilar Symbolic Models for Incrementally Stable Switched Systems. Automatic Control, IEEE Transactions on* 55(1), pp. 116–126, 10.1109/TAC.2009.2034922.

13 Antoine Girard (2005): *Reachability of Uncertain Linear Systems Using Zonotopes.* In: *HSCC, Lecture Notes in Computer Science* 3414, Springer, pp. 291–305.

14 Antoine Girard (2010): *Synthesis using approximately bisimilar abstractions: state-feedback controllers for safety specifications.* In: *Proceedings of the 13th ACM international conference on Hybrid systems: computation and control*, ACM, pp. 111–120.

15 Z. Han & B. H. Krogh (2004): *Reachability analysis of hybrid systems using reduced-order models.* In: *American Control Conference*, IEEE, pp. 1183–1189.

16 Z. Han & B. H. Krogh (2006): In: *Hybrid Systems: Computation and Control*, *Lecture Notes in Computer Science* 3927.

17 M. Herceg, M. Kvasnica, C.N. Jones & M. Morari (2013): *Multi-Parametric Toolbox 3.0.* In: *Proc. of the European Control Conference*, Zürich, Switzerland, pp. 502–510.

18 M. Mazo Jr., A. Davitian & P. Tabuada (2010): *PESSOA: A Tool for Embedded Controller Synthesis.* In Tayssir Touili, Byron Cook & Paul Jackson, editors: *Computer Aided Verification*, *Lecture Notes in Computer Science* 6174, Springer Berlin Heidelberg, pp. 566–569, 10.1007/978-3-642-14295-6_49.

19 B. C. Moore (1981): *Principal component analysis in linear systems: Controllability, observability and model reduction. IEEE Transaction on Automatic Control* 26(1).

20 S. Mouelhi, A. Girard & G. Goessler (2013): *CoSyMA: a tool for controller synthesis using multi-scale abstractions.* In: *HSCC'13 - 16th International Conference on Hybrid systems: computation and control*, ACM, Philadelphie, United States, pp. 83–88, 10.1145/2461328.2461343.

21 *Octave Web Page.* http://www.gnu.org/software/octave/.

22 B. Picasso & A. Bicchi (2007): *On the Stabilization of Linear Systems Under Assigned I/O Quantization. IEEE Trans. Automat. Contr.* 52(10), pp. 1994–2000.

23 K. Reif, S. Günther, E. Yaz Sr. & R. Unbehauen (1999): *Stochastic stability of the discrete-time extended Kalman filter. IEEE Transactions on Automatic Control* 44(4), pp. 714–728.

24 Paulo Tabuada (2009): *Verification and control of hybrid systems: a symbolic approach.* Springer Science & Business Media.

25 D. Tong, W. Zhou, A. Dai, H. Wang, X. Mou & Y. Xu (2014): *H_∞ model reduction for the distillation column linear system. Circuits Syst Signal Process.*

Appendix: Model order reduction and error bounding

Error bounding for the output trajectory

Here, a scalar *a posteriori* error bound for e is given (adapted from [15]). An upper bound of the Euclidean norm of the error over all possible initial conditions and controls can be formalized as the solution of the following optimal control problem:

$$\varepsilon_y(t) = \sup_{u\in U, x_0\in R_x} \|e(x_0,u,t)\| = \sup_{u\in U, x_0\in R_x} \|y(x_0,u,t) - y_r(\pi_R x_0,u,t)\|$$

Since the full-order and reduced systems are linear, the error bound can be estimated as $\varepsilon_y(t) \leq \varepsilon^{x_0=0}(t) + \varepsilon^{u=0}(t)$ where $\varepsilon_y^{x_0=0}$ is the error of the zero-state response, given by

$$\varepsilon_y^{x_0=0}(t) = \max_{u\in U} \|u\| \cdot \|e(x_0=0,u,t)\| = \max_{u\in U} \|u\| \cdot \|y(x_0=0,u,t) - y_r(x_0=0,u,t)\|, \quad (2)$$

and $\varepsilon_y^{u=0}$ is the error of the zero-input response, given by

$$\varepsilon_y^{u=0}(t) = \sup_{x_0\in R_x} \|e(x_0,u=0,t)\| = \sup_{x\in R_x} \|y(x_0,u=0,t) - y_r(\pi_R x_0,u=0,t)\|. \quad (3)$$

Using some algebraic manipulations, one can find a precise bound for $\varepsilon_y^{x_0=0}$ and $\varepsilon_y^{u=0}$ (see [15]). We have:

$$\varepsilon_y^j = \varepsilon_y(j\tau) = \|u(\cdot)\|_\infty^{[0,j\tau]} \int_0^{j\tau} \| \begin{bmatrix} C & -\hat{C} \end{bmatrix} \begin{bmatrix} e^{tA} & \\ & e^{t\hat{A}} \end{bmatrix} \begin{bmatrix} B \\ \hat{B} \end{bmatrix} \| dt \; + \sup_{x_0\in R_x} \| \begin{bmatrix} C & -\hat{C} \end{bmatrix} \begin{bmatrix} e^{j\tau A} & \\ & e^{j\tau\hat{A}} \end{bmatrix} \begin{bmatrix} x_0 \\ \pi_R x_0 \end{bmatrix} \|. \quad (4)$$

The bound $\varepsilon_y^\infty = \sup_{j\geq 0} \varepsilon_y(j\tau)$ is computable when the modulus of the eigenvalues of $e^{\tau A}$ and $e^{\tau\hat{A}}$ is strictly inferior to one, which we suppose here.

Error bounding for the state trajectory

We recall and introduce some notations, j being the length of the input pattern Pat tested by the bisection method:

$$Post_{Pat}(x) = e^{j\tau A}x + \int_0^{j\tau} e^{A(j\tau-t)}Bu(t)dt,$$

$$Post_{Pat}(\pi_R x) = e^{j\tau\hat{A}}\pi_R x + \int_0^{j\tau} e^{\hat{A}(j\tau-t)}\hat{B}u(t)dt,$$

Using an approach similar to the construction of the bounds (2) and (3), we obtain the following bound, which depends on the length j of the input pattern Pat:

$$\|\pi_R Post_{Pat}(x) - Post_{Pat}(\pi_R x)\| \leq \varepsilon_x^j, \quad (5)$$

with

$$\varepsilon_x^j = \|u(\cdot)\|_\infty^{[0,j\tau]} \int_0^{j\tau} \| \begin{bmatrix} \pi_R & -I_{n_r} \end{bmatrix} \begin{bmatrix} e^{tA} & \\ & e^{t\hat{A}} \end{bmatrix} \begin{bmatrix} B \\ \hat{B} \end{bmatrix} \| dt \; + \sup_{x_0\in R_x} \| \begin{bmatrix} \pi_R & -I_{n_r} \end{bmatrix} \begin{bmatrix} e^{j\tau A} & \\ & e^{j\tau\hat{A}} \end{bmatrix} \begin{bmatrix} x_0 \\ \pi_R x_0 \end{bmatrix} \|. \quad (6)$$