A classification–based approach to the optimal control of affine switched systems*

Giorgio Manganini, Luigi Piroddi, and Maria Prandini[1]

Abstract— **This paper deals with the optimal control of discrete–time switched systems, characterized by a finite set of operating modes, each one associated with given affine dynamics. The objective is the design of the switching law so as to minimize an infinite–horizon expected cost, that penalizes frequent switchings. The optimal switching law is computed off–line, which allows an efficient online operation of the control via a state feedback policy. The latter associates a mode to each state and, as such, can be viewed as a classifier. In order to train such classifier–type controller one needs first to generate a set of training data in the form of optimal state–mode pairs. In the considered setting, this involves solving a Mixed Integer Quadratic Programming (MIQP) problem for each pair. A key feature of the proposed approach is the use of a classification method that provides guarantees on the generalization properties of the classifier. The approach is tested on a multi–room heating control problem.**

I. Introduction

In this paper we consider the optimal control of discrete–time switched affine systems, which are hybrid systems where the discrete state identifies the mode of the system and the continuous state is governed by mode–specific affine dynamics. The objective is to compute a state–feedback switching law that minimizes a cost function over an infinite time horizon.

A general framework for the optimal control of switched systems was established by [1] in the context of hybrid systems. For the class of switched affine systems with no continuous input, [2] showed that the optimal control formulation leads to a two-point boundary value problem, and a general solution is difficult to obtain both analytically and numerically. A state–space discretization technique was adopted in [3] to solve the Hamilton–Jacobi–Bellman equations and determine the optimal feedback control law. One way to reduce the deriving computational complexity consists in performing optimal control decisions over a (short) receding horizon, as in Model Predictive Control (MPC) [4]. Alternatively, [5] dealt with positive switched systems and used piecewise co–positive Lyapunov functions to obtain suboptimal switching rules with a guaranteed level of performance, still retaining a full horizon for the decisions.

Optimal feedback laws are designed for the particular case where the continuous component has linear dynamics and the cost function is piecewise-quadratic, under the assumption that the switching sequence has finite length and that the mode sequence is fixed, so that only the switching instants must be optimized [6]. Later, in [7] the approach is extended to affine dynamics and the optimal control problem is solved with both the switching instants and the mode sequence as decision variables. Finally, in [8] an infinite number of switchings is allowed.

A similar setting is studied here, albeit circumventing some of the limitations of the mentioned approaches. In particular, an infinite horizon problem is analyzed that allows for infinite switchings and, at the same time, encompasses switching costs. Combining all these features together appears to be a challenging problem. For example, [7] considers switching costs, but only for finite switching sequences. Switching costs are handled also in [9], using an ad–hoc heuristic, but only for a finite time horizon control problem.

Additionally, the presented approach removes the requirement of the discretization of the state space, which is a well-known source of computational complexity even for relatively small scale systems. Also, differently from most of the works in the literature, we adopt a discrete–time framework. This can be viewed as a way of implicitly enforcing a certain minimum dwell time between consecutive switchings, thus avoiding chattering phenomena.

Similarly to [10], we employ Mixed Integer Quadratic Programming (MIQP) to determine the optimal control policy. However, while in [10] MIQP problems are repeatedly solved on–line to determine the optimal control sequence at each time instant (only the first control is applied every time, according to the receding horizon strategy), MIQP is here used for the off–line computation of the switching law. When online optimization is not viable, [11], [12] suggest a multi–parametric programming approach for solving a finite–horizon hybrid optimal control problem in a state–feedback form. Notice, however, that the solution proposed in these works is not applicable here, since it refers to a continuous control input rather than the mode switching signal.

In order to transfer the computational load off–line, we introduce a classification-based algorithm that (approximately) computes the optimal feedback policy as a map from the state space onto the control input space, given by the set of operating modes for the system. The key idea is to represent such a policy by means of a (multi–class) classifier: each operating mode is viewed as a distinct class and states are instances to be classified. The use of classifiers for representing control policies has been recently suggested in the reinforcement learning literature in combination with policy iteration schemes [13]–[16]. For instance, [13] formulates the policy improvement step as a classification problem. For each state in a training set, a state–action pair is generated based on the estimated value function associated to the current policy. Then, the updated policy is obtained as a classifier trained over the given state-action pairs.

Rather than resorting to a policy iteration scheme, we here perform a direct policy search by first generating a data-set of optimal state–input pairs, and then training a classifier over

*The authors gratefully acknowledge financial support by the European Commission project UnCoVerCPS under grant number 643921.

[1] Giorgio Manganini, Luigi Piroddi, and Maria Prandini are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, via Ponzio 34/5, 20133 Milano, Italy. {giorgio.manganini, luigi.piroddi,maria.prandini}@polimi.it

these data to provide an (approximately) optimal closed-loop control policy. In the case of systems affine in the state and quadratic cost, the data-set generation reduces to solving a convex optimization problem per initial state. To address the classification task, we employ the *Guaranteed Error Machine* (GEM) classifier [17], which provides theoretical guarantees on the probability of classification errors. Moreover, this error probability can be tuned by the user by adequately setting some parameters entering the classifier definition.

II. Preliminaries and Problem Statement

In this paper we consider the class of hybrid systems, commonly denoted as *switched systems*, with affine dynamics:

$$x(k+1) = A_{\sigma(k)}x(k) + f_{\sigma(k)}, \quad \sigma(k) \in \mathcal{I}, \tag{1}$$

where $x(k) \in \mathbb{R}^n$ is the continuous state, $\sigma(k) \in \mathcal{I} = \{1, 2, \dots, m\}$ is the switching signal that selects the *mode* of the system. Each mode $i \in \mathcal{I}$ identifies a matrix $A_i \in \mathbb{R}^{n\times n}$ and a vector $f_i \in \mathbb{R}^n$ characterizing the system dynamics. The switching signal at time k may depend on the previous mode of the system, identified by $\sigma(k-1)$. To model this, we add the discrete variable $q(k) \in \mathcal{I}$ that specifies the previous active mode:

$$q(k+1) = \sigma(k). \tag{2}$$

Note that (1-2) can be viewed as the equations of a nonlinear *stationary* system of the form:

$$z(k+1) = F(z(k), \sigma(k)), \tag{3}$$

with state $z = [x^T \ \ q]^T$ and control input σ.

With reference to system (1), a control policy is a rule that selects which mode to activate at each time instant, so as to achieve a desired behavior of the controlled system. The control objective can be formulated, *e.g.*, in terms of a discounted cost over the time horizon $[0, L]$:

$$\begin{aligned} J_L(x(0), q(0)) = &\sum_{k=0}^{L-1} \gamma^k g(x(k), q(k), \sigma(k)) \\ &+ \gamma^L g_L(x(L), q(L)), \end{aligned} \tag{4}$$

where $x(0)$ is the initial state and $q(0)$ is the initial mode, $\gamma \in (0, 1)$ is the discount factor, $g(x(k), q(k), \sigma(k)) \in \mathbb{R}_+$ is the cost per stage (assumed stationary for simplicity), and $g_L(x(L), q(L))$ is the terminal cost. In the sequel, the cost per stage includes a state tracking term and a switching cost:

$$\begin{aligned} &g(x(k), q(k), \sigma(k)) = \\ &(x(k) - x_{ref})^T Q(x(k) - x_{ref}) + H_{q(k),\sigma(k)}, \end{aligned} \tag{5}$$

where x_{ref} is some reference set point and $Q = Q^T \succeq 0$ is a given positive semi-definite weighting matrix. Parameter $H_{i,j}$, $i, j \in \mathcal{I}$, identifies the cost incurred for switching from mode i to mode j, and satisfies $H_{i,j} \geq 0$ ($H_{i,j} = 0$ when $i = j$). The desired trade–off between the two objectives can be enforced by suitably setting Q and $H_{i,j}$, $i, j \in \mathcal{I}$. Finally, the terminal cost is set to

$$g_L(x(L), q(L)) = (x(L) - x_{ref})^T Q(x(L) - x_{ref}). \tag{6}$$

If the time horizon length L grows to infinity, the cost function becomes:

$$J_\infty(x(0), q(0)) = \sum_{k=0}^{\infty} \gamma^k g(x(k), q(k), \sigma(k)), \tag{7}$$

and the terminal cost disappears.

The overall cost depends on the sequence $\sigma(0), \dots, \sigma(L-1)$ of control input values. A control sequence is optimal (denoted σ^*) for a given initial state $x(0)$ and mode $q(0)$ if it minimizes $J_L(x(0), q(0))$. A control policy π is a (possibly time-varying) map from the state (x, q) to σ. A control policies can be defined either in open-loop or closed-loop. *Open-loop* policies select the control input ($\sigma(k)$) based on the initial state:

$$\sigma(k) = \pi_{ol}(x(0), q(0), k),$$

i.e. without any direct measurement of the effects of past inputs. On the other hand, *closed-loop* policies select the control input based on the knowledge of the current state:

$$\sigma(k) = \pi_{cl}(x(k), q(k), k).$$

A control policy is optimal (denoted π^*) if it provides the optimal control sequence for all initial states. In particular, it must hold that $\pi^*_{cl}(x, q, 0) = \pi^*_{ol}(x, q, 0)$ for every (x, q), whereas for $k > 0$ the two policies provide the same control input sequence only in the absence of disturbances and model uncertainty. A closed-loop policy implementation of the optimal control sequence is preferable, in order to deal with possible uncertainties affecting the real system.

In this paper, we focus on the infinite–horizon case, for which the optimal control design and implementation are easier. Indeed, since the system is stationary and the cost function has a time–invariant cost per stage, it turns out that there exists an optimal closed–loop *stationary* control policy, [18]. Denoting such policy as $\bar{\pi}^*_{cl}$, it holds that $\bar{\pi}^*_{cl}(x, q) = \pi^*_{cl}(x, q, k)$, $\forall k \geq 0$. A consequence of this is that $\bar{\pi}^*_{cl}(x, q) = \pi^*_{ol}(x, q, 0)$. In principle, then, one could compute the optimal closed–loop policy by calculating the optimal input at time $k = 0$ according to the open–loop policy for all possible (initial) states.

In practice, the calculation of the optimal open–loop policy is approximated over a finite horizon $[0, L]$. Furthermore, only a finite number of initial states can be considered, so that the desired map must be constructed by generalization over the whole state space based on a finite number of samples. Both these sources of approximation can by controlled by the user, by extending the time horizon length L and exploiting the generalization error guarantees of the GEM classifier.

III. Classification-based two-stage algorithm

In this section we introduce an algorithm to compute off-line a closed-loop control policy $\bar{\pi}^*_{cl}$ that minimizes (7). The computed optimal policy is then stored in memory and applied on–line. $\bar{\pi}^*_{cl}$ is a map from the (hybrid) state to the (discrete) switching signal. For convenience purposes we represent it as a collection of functions of the continuous state component x, indexed by the mode q: $\{\bar{\pi}^*_{cl}(\cdot, q) : \mathbb{R}^n \to \mathcal{I}\}_{q\in\mathcal{I}}$. For a given $q \in \mathcal{I}$, the switching policy $\bar{\pi}^*_{cl}(\cdot, q)$ is a piece–wise constant function of x.

For each $q \in \mathcal{I}$, the algorithm computes the control policy $\bar{\pi}^*_{cl}(\cdot, q)$, in two stages:

1) *Data–set generation stage*: a data-set $\mathcal{E}_N^q$ of (x, σ^*) pairs is generated, where x is drawn from μ_x and σ^* is the optimal switching mode associated to (x, q);
2) *Learning stage*: a classifier is trained over $\mathcal{E}_N^q$ to provide (an approximation of) the optimal closed–loop policy $\bar{\pi}_{cl}^*(\cdot, q)$.

A pseudo-code of the algorithm is provided in Algorithm 1.

Algorithm 1 Classification-based two-stage algorithm

```
Input: $\mu_x$ (initial state distribution)
       $N$ (size of the training set)
for all $q \in \mathcal{I}$ do
    $\mathcal{E}_N^q \leftarrow \varnothing$
    for $i = 1$ to $N$ do
        Draw $x_{(i)} \sim \mu_x$
        $\sigma_{(i)}^* \leftarrow$ DataGen$(x_{(i)}, q)$
        $\mathcal{E}_N^q \leftarrow \mathcal{E}_N^q \cup \{(x_{(i)}, \sigma_{(i)}^*)\}$
    end for
    $\bar{\pi}_{cl}^*(\cdot, q) \leftarrow$ Learn$(\mathcal{E}_N^q)$
end for
Output: Policy $\bar{\pi}_{cl}^*$
```

The next two sub–sections explain in detail the two main stages of the algorithm.

A. Data-set generation: the DATAGEN subroutine

To generate the data–set $\mathcal{E}_N^q$ of training examples associated to mode $q \in \mathcal{I}$, we solve the following optimization problem for any given initial state $x(0) = x_0$ extracted at random according to μ_x:

$$\begin{aligned}
&\min_{\sigma(0),\dots,\sigma(L-1)} J_L(x(0), q(0))\\
&\text{subject to:}\\
&x(k+1) = A_{\sigma(k)}x(k) + f_{\sigma(k)},\ k = 0, \dots, L-1\\
&q(k+1) = \sigma(k),\ k = 0, \dots, L-1\\
&x(0) = x_0, q(0) = q
\end{aligned} \tag{8}$$

where J_L is defined as in (4). Finally, the pair $(x_0, \sigma(0))$ is stored in $\mathcal{E}_N^q$.

If L is sufficiently large, the finite-horizon cost is a good approximation of the infinite horizon cost and the value obtained for the control input at time 0 approximates the optimal one for the infinite horizon case.

This optimization problem can be efficiently solved via MIQP [19]. To this end, it is useful to reformulate system (1) as a Mixed Logical Dynamical (MLD) system [10]. An MLD systems is described by affine dynamic equations subject to linear mixed-integer inequalities involving both continuous and binary variables.

The first step in the reformulation is the introduction of the binary input variables $\delta_i(k) \in \{0, 1\}$ to model the choice of the control input at time k. More precisely, we have

$$\begin{aligned}
&\delta_i(k) = 1 \Leftrightarrow \sigma(k) = i, \quad k = 0, \dots, L-1,\ i \in \mathcal{I}\\
&\sum_{i=1}^{m} \delta_i(k) = 1, \quad k = 0, \dots, L-1,
\end{aligned} \tag{9}$$

where condition (9) implements the exclusive-or constraint that the control input can take only one value at any time. System (1) can now be rewritten as:

$$x(k+1) = \sum_{i=1}^{m} \left[A_i x(k) + f_i\right] \delta_i(k).$$

This equation is nonlinear, since it involves products between states and logical inputs. However, introducing the auxiliary continuous variables $z_i(k) \in \mathbb{R}^n$:

$$z_i(k) = \left[A_i x(k) + f_i\right] \delta_i(k), \quad \forall i \in \mathcal{I} \tag{10}$$

and setting $x(k+1) = \sum_{i=1}^{m} z_i(k)$, one can transform constraint (10) into a set of mixed-integer linear inequalities by using the so-called “big-M” approach [10]. More precisely, for each $i \in \mathcal{I}$ we can set:

$$\begin{aligned}
&z_i(k) \le M\delta_i(k)\\
&z_i(k) \ge m\delta_i(k)\\
&z_i(k) \le A_i x(k) + f_i - m(1 - \delta_i(k))\\
&z_i(k) \ge A_i x(k) + f_i - M(1 - \delta_i(k))
\end{aligned} \tag{11}$$

where $M \in \mathbb{R}^n$ and $m \in \mathbb{R}^n$ are an upper and lower bound on the state vector x, respectively.

The switching cost in (5) can be reformulated as a quadratic function of the decision variables, by expressing the state variable $q(k)$ in a binary form, *i.e.* by introducing $q^\delta(k) \in \{0, 1\}^m$ such that $q_i^\delta(k) = 1$ if $q(k) = i$ and 0 otherwise:

$$H_{q(k),\sigma(k)} = \begin{bmatrix} q_1^\delta(k) \\ \vdots \\ q_m^\delta(k) \end{bmatrix}^T \begin{bmatrix} 0 & H_{12} & \dots & H_{1m} \\ H_{21} & 0 & \dots & H_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ H_{m1} & H_{m2} & \dots & 0 \end{bmatrix} \begin{bmatrix} \delta_1(k) \\ \vdots \\ \delta_m(k) \end{bmatrix} \tag{12}$$

In view of expressions (10-12) problem (8) is reformulated as a MIQP, which can be solved via standard solvers like CPLEX [20].

Remark 3.1: The main diagonal of matrix H in (12) contains all zero elements and, therefore, leads to a non–convex quadratic function of the control variables, which cannot be efficiently solved by a MIQP solver. However, the cost function can be convexified by adding a correction term that is constant with respect to the optimization variables and hence does not change the optimal switching sequence (details are omitted for brevity).

B. The GEM classification machine: the LEARN procedure

The Learn procedure consists in training a classifier on the data gathered as explained in the previous subsection. For this purpose, we employ the GEM [17] algorithm, in view of the guarantees it provides on the probability of classification errors. These guarantees can be used to prescribe a desired level of the generalization error and determine the number N of training data to be calculated by the DataGen procedure, or, alternatively, to calculate the error probability associated to a given size N of the training data–set.

Let $x \in \mathbb{R}^d$ be a vector of measured attributes and $y = y(x) \in Y = \{1, \dots, m\}$ the corresponding (discrete) class label. A classifier $\hat{y} = \hat{y}(x)$ provides an estimate for the class y of x. It errs on x if $y(x) \neq \hat{y}(x)$. Differently from most other classifying machines, the GEM returns an augmented class set $Y \cup \{unknown\}$, that includes an $unknown$ label, expressing the inability to classify the sample.

Now, let $\mathcal{E}_N = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ be a data-set of N training samples, where $x_1, \ldots, x_N$ are independently extracted according to a probability distribution μ and $y_i = y(x_i)$. The *probability of error* (or generalization error) of a classifier $\hat{y}_N(\cdot)$ trained on these data is defined as

$$PE(\hat{y}_N) = \mu\left(\hat{y}_N(x) \in Y \ \wedge \ y(x) \neq \hat{y}_N(x)\right),$$

that is the probability that an output is issued and that the output is not correct. Given that $\hat{y}_N(\cdot)$ is derived based on the set $\mathcal{E}_N$ of randomly sampled training data, $PE(\hat{y}_N)$ is a random variable whose distribution depends on the unknown data generation mechanism $(\mu, y(\cdot))$.

Theorem 1 in [17] provides a formal expression for the probability distribution $F_{PE}(\epsilon) := \mu^N\left\{PE(\hat{y}_N) \leq \epsilon\right\}$, where $1 - \epsilon$, with $\epsilon \in (0, 1)$, is the accuracy level:

$$F_{PE}(\epsilon) \geq \sum_{i=k}^{N-1} \binom{N-1}{i} \epsilon^i (1-\epsilon)^{N-1-i}, \qquad (13)$$

where the right hand side is a Beta$(k, N-k)$ distribution, that represents the confidence over $PE(\hat{y}_N) \leq \epsilon$ and can be expressed as

$$\text{Beta}(k, N-k) = 1 - \delta, \quad \delta \in (0, 1).$$

Following [21], we can compute the *sample complexity*, *i.e.* a lower bound on the minimum number of data N that are needed for expression (13) to hold, as a function of ϵ, δ, and k. Given $\delta \in (0, 1)$, $\epsilon \in (0, 1)$, and the nonnegative integer k, if N satisfies the inequality

$$N \geq 1 + \frac{1}{\epsilon}\left(k - 1 + \log\frac{1}{1-\delta} + \sqrt{2(k-1)}\log\frac{1}{1-\delta}\right),$$

then, (13) holds. Interestingly, the derived bound for N is independent of the state space dimension n.

The GEM algorithm takes as input the training data $\mathcal{E}_N$ and the tuning parameter $k \leq N-1$. It constructs an ordered sequence of (hyper)ellipsoidal regions $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_r\}$ which constitute a (sub)partition of the space of the training example. Each of these regions $\mathcal{R}_j$ has an associated label $\ell_j \in Y$, such that the union of these regions corresponds to that part of the input space where the GEM issues an answer, whereas in the remaining (uncovered) part of the space the machine returns the label $unknown$.

More in detail, starting from x_1, the algorithm constructs region $\mathcal{R}_1$, that contains x_1 and extends until it touches another datum x_j, $j \neq 1$, such that $y_j \neq y_1$. All instances included into R_1 share the same label as x_1 and are thus removed from the training set, while the instances on the boundary $\Omega(\mathcal{R}_1)$ of the region are marked as "active" points and added to a set Q. If $|Q| \leq k$ and the training data-set is not empty, then the active point farthest from x_1 is selected as the new base instance, and a new region is constructed. Ensuring that $|Q| \leq k$ is the key property that guarantees that (13) holds [17]. For more details on the GEM algorithm refer to [17].

Remark 3.2: The specific nature of the considered problem, which deals with affine switching systems, may induce a special structure in the switching regions. More precisely, in the absence of switching costs these regions are conical [7]. This can be exploited to tailor the classification method.

IV. Numerical example

In the following the proposed classification-based control design methodology is tested on a modified version of a benchmark multi-room heating control problem described in [22]. The problem concerns the simultaneous temperature regulation in n rooms, each room being endowed with a heater, with the constraint that at most one heater at a time can be active. A switching control strategy must be designed to decide at each time step which room should be heated, depending on the temperature values in all the rooms. With respect to the original benchmark description, a deterministic setting is here adopted and an energy–related cost associated to the on/off switching of the heaters is included.

The n-room heating system can be modeled as a switched system with continuous state component $x = (x_1, \ldots, x_n) \in X = \mathbb{R}^n$ representing the (average) temperature in each room. The switching signal $\sigma \in \mathcal{I} = \{1, \ldots, n+1\}$ has $n+1$ command options, namely turning on the heater of the ith room ($\sigma = i$), $i = 1, \ldots, n$, or turning them all off ($\sigma = n+1$).

The average temperature in room i is ruled by the following difference equation, obtained by Euler discretization of the corresponding continuous–time dynamics with constant time step Δt:

$$\begin{aligned} x_i(k+1) = x_i(k) + [b_i(x_a - x_i(k)) + c_i h_i(\sigma(k)) \qquad (14) \\ + \sum_{j=1,\ldots,n; j\neq i} a_{ij}(x_j(k) - x_i(k))]\Delta t, \ i = 1, \ldots, n, \end{aligned}$$

where x_a is the ambient temperature (assumed constant), and $h_i(k)$ is a boolean function equal to 1 when room i is heated, and 0 otherwise. Parameters a_{ij}, b_i and c_i in (14) are non-negative constants representing the heat exchange coefficients between room i and room j (a_{ij}), the heat loss rate of room i to the ambient (b_i) and the heat rate supplied by the heater in room i (c_i), all normalized with respect to the average thermal capacity of room i. The parameters are set as follows: $\Delta t = 1/30$, $x_a = 6$, $b_i = 0.25$ and $c_i = 12$ for $i = 1 \ldots n$, $a_{ij} = a_{ji} = 0.33$, for $i = 1, \ldots, n-1$, $j = i+1$.

The control problem can be formulated as in Section II, where the objective is to track the (constant) reference state x_{ref}, while weighting the cost penalty component associated with changing the heated room. Accordingly, we adopt the infinite horizon cost (7), with a discount factor $\gamma = 0.95$. The initial state distribution μ_x is uniform over the domain $[15.25, 24.25]^n$. A prediction horizon of length $L = 10$ is used for building the training data-set according to the approach in Section III-A. The control performance is evaluated over the same look-ahead horizon. The one-step cost function defined in (5) is employed with $Q = I_{n\times n}$ and $x_{i,ref} = 19^\circ$ for all rooms, whereas the switching cost $H_{i,j}$ represents a tuning parameter.

To evaluate the performance of the policy obtained by the proposed algorithm (denoted $\bar{\pi}^*_{GEM}$ in the following), we compared it with a standard MPC policy (referred as $\bar{\pi}^*_{MPC}$). In the MPC approach, a closed–loop policy is computed by using a time receding horizon strategy, solving on–line at every time instant k the optimization problem (8), with the initial state given by $x(k)$ and $q(k)$, and a time horizon of $k + L$. Notice that the data used to train the GEM classifier are obtained by solving the same optimization problem, so that the performance of the GEM–based controller can be put in direct relation with that of the MPC controller, and the approximation and generalization properties of the former can be evaluated.

To illustrate the results, we first make reference to the 2-room case. In this case, $N = 3000$ training state–control input pairs have been generated for modes $q = 1, 2, 3$, while the confidence and accuracy parameters for the GEM classifiers have been set to $\delta = 10^{-5}$ and $1 - \epsilon = 0.97$, respectively. No $unknown$ regions resulted after the GEM training. Figures 1-2 show the system trajectories starting from the initial state $x(0) = [17\ 17]^T$ and mode $q(0) = 3$, under both the control policies $\bar{\pi}^*_{MPC}$ and $\bar{\pi}^*_{GEM}$. In particular, Figure 1 refers to the case with no switching costs ($H_{i,j} = 0, \forall i, j \in \mathcal{I}$), while in Figure 2 we assign a high cost penalty to a switch occurrence ($H_{i,j} = 50, \forall i, j \in \mathcal{I}, i \neq j$). Notice that in the absence of switching costs the optimal policy is independent of the current mode. Conversely, its dependence on the mode increases with the switching cost.

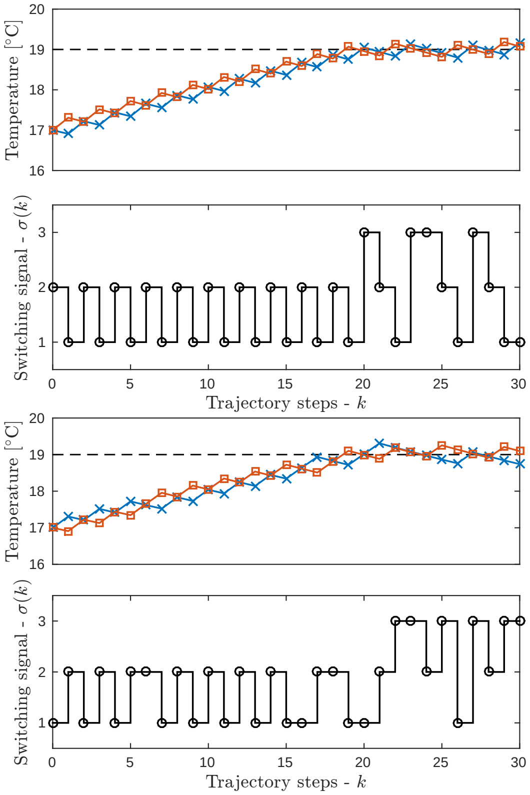

Fig. 1. Continuous state trajectories (red and blue lines represent the 1^{st} and 2^{nd} room, respectively) and switching sequences starting from $x(0) = [17\ 17]^T$ and $q(0) = 3$, under policies $\bar{\pi}^*_{MPC}$ (top) and $\bar{\pi}^*_{GEM}$ (bottom), in the absence of switching costs.

The policy $\bar{\pi}^*_{GEM}$ computed by the proposed algorithm shows a similar behavior with respect to $\bar{\pi}^*_{MPC}$, though it generates a slightly different switching sequence. Not surprisingly, better tracking performances are obtained when the switching cost is absent (Figure 1), which allows both policies to alternate the heating in the two rooms. On the other hand, when the switching cost becomes significant, both policies have to exercise a trade–off between the conflicting goals, and the number of switching occurrences is greatly reduced. For the reader's reference, the performance of the MPC policy is equal to $J = 37.0980$ and $J = 174.5605$, in the two examined conditions respectively, which are only 0.3% and 0.12% better than the corresponding values obtained with $\bar{\pi}^*_{GEM}$.

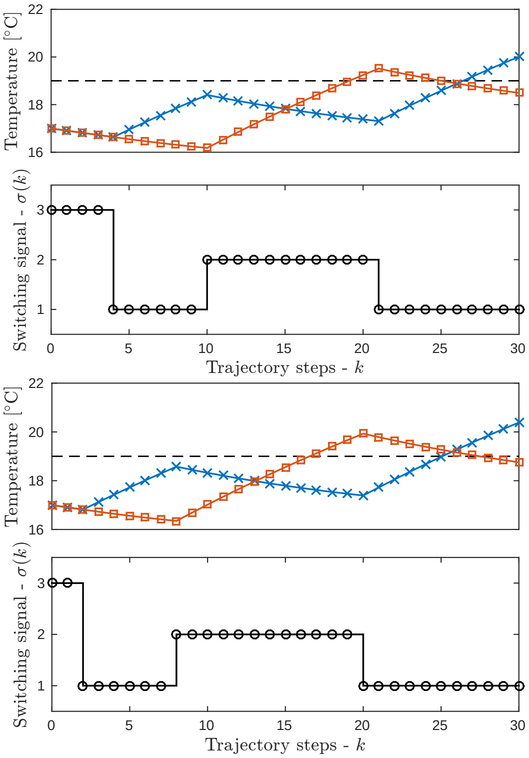

Fig. 2. Continuous state trajectories (red and blue lines represent the 1^{st} and 2^{nd} room, respectively) and switching sequences starting from $x(0) = [17\ 17]^T$ and $q(0) = 3$, under policies $\bar{\pi}^*_{MPC}$ (top) and $\bar{\pi}^*_{GEM}$ (bottom), in the presence of large switching costs.

The performance loss due to the approximation introduced by the GEM classifier in the definition of the policy $\bar{\pi}^*_{GEM}$ has been further tested in larger case instances. A Monte Carlo estimate of the expected cost

$$\mathop{\mathbb{E}}_{\substack{x(0)\sim\mu_x \\ q(0)\sim\mu_q}} [J_\infty(x(0), q(0))]$$

with μ_q uniform, is evaluated for the n-room scenario, when $n = 1, \dots, 4$, for both control policies: the simulations are performed following each policy for 50 steps, starting from 100 initial states drawn from the distribution μ_x, with initial mode $q(0) = q = n + 1$, and no switching cost is considered. $N = 15000$ training input pairs are generated for the computation of the policy $\bar{\pi}^*_{GEM}$. Figure 3 shows the relative performance loss of the policy $\bar{\pi}^*_{GEM}$ with respect to $\bar{\pi}^*_{MPC}$ as a function of the accuracy of the GEM machine. Apparently, this dependence is essentially linear, showing a graceful degradation of the performance as ϵ is increased. Notice also that, while equation (13) provides a lower bound for the probability that $PE(\bar{\pi}^*_{GEM}) \leq \epsilon$, the practical application of the GEM classifier typically leads to better performances.

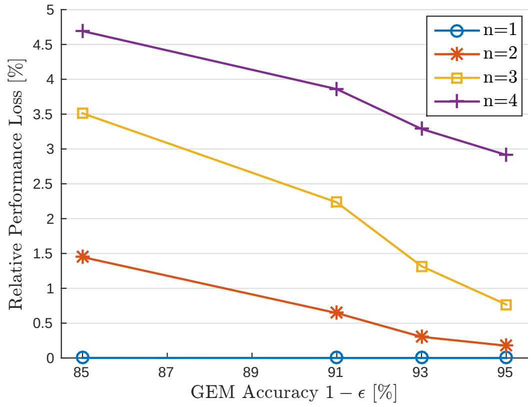

Fig. 3. Performance loss of policy $\bar{\pi}^*_{GEM}$ with respect to $\bar{\pi}^*_{MPC}$, as a function of the accuracy of the GEM machine.

Finally, based on the same parameter settings, a computational analysis of the proposed algorithm is shown in Figure 4. Clearly, the $\bar{\pi}^*_{GEM}$ policy results in an extremely fast on–line computational time and depends almost linearly on the problem size, whereas the MPC policy involves a much higher computational cost in the on–line phase. Indeed, the complexity of Algorithm (1) resides in the off-line phase, and depends essentially on the construction of the data–set of training examples generated for learning the GEM classifier. We remark that the number of samples N required by the algorithm in order to calculate the policy $\bar{\pi}^*_{GEM}$ is independent of the state dimension and can be tuned by the user so as to select the level accuracy of the GEM classifier.

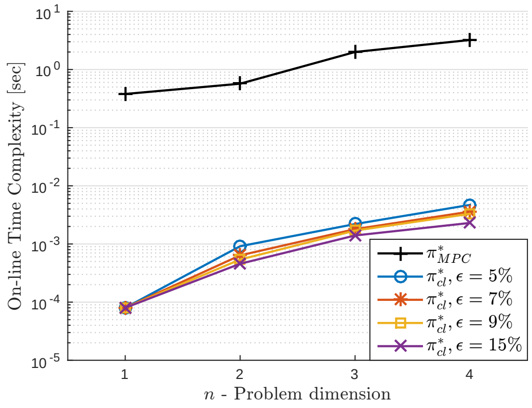

Fig. 4. On–line time complexity for the $\bar{\pi}^*_{MPC}$ and $\bar{\pi}^*_{GEM}$ policies.

V. Conclusions

A classification-based approach has been proposed for the optimal control of discrete-time switched affine systems. The proposed method operates in two steps. First, a number of initial states is drawn from a uniform distribution over the state space, and an (approximately) optimal control action is associated to each of them. The latter is calculated by solving a MIQP problem. Then, a GEM classifier is trained on the obtained data-set of state-control pairs. Precise bounds can be derived on the generalization capabilities of the classifier, which indirectly affect the control performance. Some approximation must be accepted, since only approximately optimal control solutions can be obtained in practice due to horizon truncation. However, some simulation experiments on a benchmark problem reveal that the difference with respect to a standard MPC policy is small.

References

[1] M. S. Branicky, V. S. Borkar, and S. K. Mitter, "A unified framework for hybrid control: Model and optimal control theory," *Automatic Control, IEEE Transactions on*, vol. 43, no. 1, pp. 31–45, 1998.

[2] R. H. Middleton, P. Colaneri, E. Hernandez-Vargas, and F. Blanchini, "Continuous-time optimal control for switched positive systems with application to mitigating viral escape," in *Proceedings of the 8th IFAC symposium on nonlinear control systems*, 2010, pp. 266–271.

[3] A. Rantzer and M. Johansson, "Piecewise linear quadratic optimal control," *Automatic Control, IEEE Transactions on*, vol. 45, no. 4, pp. 629–637, 2000.

[4] E. A. Hernandez-Vargas, P. Colaneri, and R. H. Middleton, "Switching strategies to mitigate hiv mutation," *Control Systems Technology, IEEE Transactions on*, vol. 22, no. 4, pp. 1623–1628, 2014.

[5] E. Hernandez-Vargas, P. Colaneri, R. Middleton, and F. Blanchini, "Discrete-time control for switched positive systems with application to mitigating viral escape," *International Journal of Robust and Nonlinear Control*, vol. 21, no. 10, pp. 1093–1111, 2011.

[6] A. Giua, C. Seatzu, and C. Van Der Mee, "Optimal control of switched autonomous linear systems," in *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, vol. 3. IEEE, 2001, pp. 2472–2477.

[7] C. Seatzu, D. Corona, A. Giua, and A. Bemporad, "Optimal control of continuous-time switched affine systems," *Automatic Control, IEEE Transactions on*, vol. 51, no. 5, pp. 726–741, 2006.

[8] D. Corona, A. Giua, and C. Seatzu, "Stabilization of switched systems via optimal control," in *Proc. 16th IFAC World Congress*, 2005.

[9] X. Ding, Y. Wardi, D. Taylor, and M. Egerstedt, "Optimization of switched-mode systems with switching costs," in *American Control Conference, 2008.* IEEE, 2008, pp. 3965–3970.

[10] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[11] A. Bemporad, F. Borrelli, and M. Morari, "Piecewise linear optimal controllers for hybrid systems," in *American Control Conference*, vol. 2, 2000, pp. 1190–1194.

[12] F. Borrelli, M. Baotić, A. Bemporad, and M. Morari, "Dynamic programming for constrained optimal control of discrete-time linear hybrid systems," *Automatica*, vol. 41, no. 10, pp. 1709–1721, 2005.

[13] M. G. Lagoudakis and R. Parr, "Reinforcement learning as classification: Leveraging modern classifiers," in *Proc. of the 20th International Conference on Machine Learning*, Washington DC, USA, 2003, pp. 424–431.

[14] I. Rexakis and M. G. Lagoudakis, "Directed exploration of policy space using support vector classifiers," in *Proc. of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, Paris, France, April 11-15 2011, pp. 112–119.

[15] A. Lazaric, M. Ghavamzadeh, and R. Munos, "Analysis of a classification-based policy iteration algorithm," in *Proc. of the 27th Int. Conf. on Machine Learning*, Haifa, Israel, 2010.

[16] J. Langford and B. Zadrozny, "Relating reinforcement learning performance to classification performance," in *In 22nd Int. Conf. on Machine Learning (ICML)*, Bonn, Germany, 2005.

[17] M. C. Campi, "Classification with guaranteed probability of error," *Machine Learning*, vol. 80, no. 1, pp. 63–84, July 2010.

[18] D. P. Bertsekas and S. E. Shreve, *Stochastic Optimal Control: The Discrete Time Case*. Academic Press, 1978.

[19] R. Fletcher and S. Leyffer, "Numerical experience with lower bounds for MIQP branch-and-bound," *SIAM J. on Optimization*, vol. 8, no. 2, pp. 604–616, February 1998.

[20] "IBM ILOG CPLEX Optimizer," 2010. [Online]. Available: http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/

[21] T. Alamo, R. Tempo, and A. Luque, "On the sample complexity of randomized approaches to the analysis and design under uncertainty," in *Proc. of the American Control Conference (ACC)*, Baltimore (MD), USA, June 2010, pp. 4671–4676.

[22] A. Fehnker and F. Ivančić, "Benchmarks for hybrid systems verifications," in *Hybrid Systems: Computation and Control*, ser. LNCS 2993, R. Alur and G. J. Pappas, Eds. Springer Verlag, April 2004, pp. 326–341.