2017 25th Mediterranean Conference on Control and Automation (MED)
July 3-6, 2017. Valletta, Malta

Regularized Moving-Horizon Piecewise Affine Regression using Mixed-Integer Quadratic Programming*

Vihangkumar V. Naik, Manas Mejari, Dario Piga and Alberto Bemporad

Abstract— **This paper presents a novel two-stage regularized moving-horizon algorithm for *PieceWise Affine* (PWA) regression. At the first stage, the training samples are processed iteratively, and a *Mixed-Integer Quadratic-Programming* (MIQP) problem is solved to find the sequence of active modes and the model parameters which best match the training data, within a relatively short time window in the past. According to a moving-horizon strategy, only the last element of the optimal sequence of active modes is kept, and the next sample is processed by shifting forward the estimation horizon. A regularization term on the model parameters is included in the cost of the formulated MIQP problem, to partly take into account also the past training data outside the considered time horizon. At the second stage, linear multi-category discrimination techniques are used to compute a polyhedral partition of the regressor space based on the estimated sequence of active modes.**

I. INTRODUCTION

Piecewise Affine (PWA) systems are heterogeneous systems which exhibit both continuous and discrete dynamics. PWA models are simple and flexible model structures and thanks to their universal approximation properties, any nonlinear function can be modeled with arbitrary accuracy by a PWA map [8]. Furthermore, due to the equivalence between PWA models and several classes of hybrid models [13], available tools for modelling, analysis and control of hybrid systems can be also applied to PWA systems [4], [6].

PWA regression is an NP-hard problem [15], where both the regressor space partition and the submodel parameters have to be estimated from a set of training data. Several algorithms/heuristics for PWA regression and for the identification of hybrid systems, have been proposed in the last two decades (see the survey papers [12], [20]). Among these algorithms, we mention the set-membership approaches [5], [19], the sparse-optimization based approaches [1], [18] and the mixed-integer programming method [22]. In the latter approach, the estimation of hinging-hyperplane ARX models and piecewise affine Wiener models is formulated as a mixed-integer linear or quadratic programming problem, and then solved through a branch-and-bound algorithm. As the number of integer variables is proportional to the number of modes and the number of training samples, the approach in [22] is limited to medium-scale problems. The contributions [2], [5], [9]–[11], [14], [17] fall in the class of cluster-based two-stage methods. At the first stage, the training samples are clustered by assigning each datapoint to a submodel according to a certain criterion and, at the same time, the parameters of the affine submodels are estimated. In the second stage, the polyhedral partition of the regressor space is computed by linear separation techniques. Unlike the mixed-integer programming approach [22], which can be solved for the global optimum, sub-optimal solutions are obtained by the aforementioned two-stage methods.

In this paper, we propose a regularized moving-horizon PWA regression algorithm, which can be seen as a mix between the mixed-integer programming approach [22] and the cluster-based algorithm [10]. At the first stage, a *Mixed-Integer Quadratic-Programming* (MIQP) problem is formulated to compute both the optimal sequence of active modes within the considered horizon and the parameters of the affine submodels. Moreover, a regularization term is included in the cost of the formulated MIQP problem, to exploit the information from the past training samples outside the considered time window. Thus, the length N_p of the horizon acts as a knob to combine the advantages of the two-stage algorithm [10] (namely, computational efficiency and iterative processing of the training samples) and the advantages of the mixed-integer programming approach [22] (namely, non-decoupled optimization over the active modes and the sub-model parameters). According to a moving-horizon strategy, only the active mode at current sampling time is extracted from the computed optimal sequence of active modes, and the next training sample is processed by shifting forward the estimation horizon. At the second stage, the regressor space is partitioned using computationally efficient multi-class linear separation methods proposed in [10].

II. PROBLEM FORMULATION

Let us consider a data-generating system in the form

$$y(k) = f_{\rm o}(x(k)) + e_{\rm o}(k), \tag{1}$$

where $k \in \mathbb{N}$ is the time index, $y(k) \in \mathbb{R}^{n_y}$ is the measured output at time k, $e_{\rm o}(k) \in \mathbb{R}^{n_y}$ is an additive random noise, $x(k) \in \mathbb{R}^{n_x}$ is the regressor vector which is assumed to take values in a set $\mathcal{X} \subset \mathbb{R}^{n_x}$, and $f_{\rm o} : \mathcal{X} \to \mathbb{R}^{n_y}$ is an unknown and possible discontinuous function.

In this paper, we address a PWA regression problem, which amounts at computing a PWA function $f : \mathcal{X} \to \mathbb{R}^{n_y}$ approximating the regression function $f_{\rm o}$ based on a set of N

*This work was partially supported by the H2020 European project DISIRE, Grant Agreement No. 636834 and by the H2020-723248 project "DAEDALUS - Distributed control and simulation platform to support an ecosystem of digital automation developers".

Vihangkumar V. Naik, Manas Mejari and Alberto Bemporad are with the IMT School for Advanced Studies Lucca, Piazza San Francesco 19, 55100 Lucca, Italy. E-mail: {vihangkumar.naik, manas.mejari, alberto.bemporad}@imtlucca.it.

Dario Piga is with the IDSIA Dalle Molle Institute for Artificial Intelligence SUPSI-USI, 6928 Manno, Switzerland. E-mail: dario.piga@supsi.ch.

observations of the regressor/output pairs $\{x(k), y(k)\}_{k=1}^{N}$. The PWA vector-valued function f is described as:

$$f(x) = \begin{cases} \Theta_1 \left[\begin{smallmatrix} 1 \\ x \end{smallmatrix}\right] & \text{if } x \in \mathcal{X}_1, \\ \vdots & \\ \Theta_s \left[\begin{smallmatrix} 1 \\ x \end{smallmatrix}\right] & \text{if } x \in \mathcal{X}_s, \end{cases} \tag{2}$$

where $s \in \mathbb{N}$ denotes the number of modes, $\Theta_i \in \mathbb{R}^{n_y \times (n_x+1)}$ are parameter matrices, and $\mathcal{X}_i$, with $i = 1, \ldots, s$, are polyhedra ($\mathcal{H}_i x \leq \mathcal{D}_i$) that form a complete polyhedral partition[1] of the regressor space $\mathcal{X}$.

Estimation of the PWA function f in (2) thus requires: (i) selecting the number of modes s; (ii) estimating the parameter matrices Θ_i; and (iii) finding the polyhedra $\mathcal{X}_i$ (i.e., the matrices $\mathcal{H}_i$ and $\mathcal{D}_i$) defining the partition of the regressor space $\mathcal{X}$. Tradeoff between data fitting and model complexity should be taken into account while choosing the number of modes s. If the number of modes s is small, then the PWA map f may not be flexible enough to capture the shape of the underlying nonlinear data-generating function f_{o} (1). On the contrary, considering the high number of modes results in a more accurate description of the PWA map f with more degrees of freedom. However, this may cause overfitting and poor generalization to unseen data (not used in the training phase) as the final estimate is sensitive to the noise corrupting the observations, besides increasing the complexity of the estimation procedure and of the resulting PWA model. In the rest of the paper, we assume that s is fixed by the user, and chosen via cross-calibration. This is done by evaluating the performance of the estimated model for different values of s, on a fresh data set which is different from the training data set.

III. PWA REGRESSION ALGORITHM

The developed algorithm for PWA regression consists of the following two stages:

S1. Recursive *estimation* of the model parameters Θ_i and simultaneous *clustering* of the regressors $\{x(k)\}_{k=1}^{N}$.

S2. *Computation of a polyhedral partition* of the regressor space $\mathcal{X}$ using computationally efficient multi-category linear separation methods already available in the literature. This can be computed either offline or online (recursively) and is executed after S1.

A. Recursive clustering and parameter estimation

Stage **S1** is carried out through a regularized moving-horizon identification algorithm. The training regressor/output pairs $\{x(k), y(k)\}$ are processed iteratively. At each time sample k, a moving-horizon window of length N_p containing regressor/output pairs from time $k - N_p + 1$ to time k is considered. The model parameters Θ_i and the active mode $\sigma(k)$ at time k are estimated simultaneously by solving the mixed-integer programming problem:

$$\min_{\substack{\{\Theta_i\}_{i=1}^{s} \\ \{\delta_i(k-t)\}_{i=1,t=0}^{s,N_p-1}}} \sum_{i=1}^{s} \sum_{t=0}^{N_p-1} \left\| \left(y(k-t) - \Theta_i \left[\begin{smallmatrix} 1 \\ x(k-t) \end{smallmatrix}\right]\right) \delta_i(k-t) \right\|^2 \tag{3a}$$

$$+ \sum_{t=1}^{k-N_p} \left\| y(t) - \Theta_{\sigma(t)} \left[\begin{smallmatrix} 1 \\ x(t) \end{smallmatrix}\right] \right\|^2 \tag{3b}$$

$$\text{s.t. } \delta_i(k-t) \in \{0,1\}, \ \sum_{i=1}^{s} \delta_i(k-t) = 1, \ t = 0, \ldots, N_p-1. \tag{3c}$$

The active mode $\sigma(k) \in \{1, \ldots, s\}$ represents the cluster that the regressor $x(k)$ is assigned to, and it is extracted from the optimizer of problem (3), i.e.,

$$\sigma(k) = i^*, \quad \text{with } \ i^* : \delta_{i^*}(k) = 1. \tag{4}$$

According to a moving-horizon estimation strategy, only the active mode $\sigma(k)$ at time k is kept, and the N_p-length time window is shifted forward to process the next pair $\{x(k+1), y(k+1)\}$.

Problem (3) aims at searching for the optimal sequence of active modes within the considered time window and the model parameters Θ_i which best match the available observations up to time k. Note that the term (3a) aims at finding both the model parameters and the sequence of active modes $\{\sigma(t)\}_{t=k-N_p+1}^{k}$ which best match the observations within the N_p-step time horizon. The term (3b) acts as a regularization term on the parameters Θ_i and it takes into account the time history of the observations outside the considered time window. More specifically, in (3b), the sequence of active modes is not optimized from time 1 to time $k - N_p$, but it is fixed to the estimates $\{\sigma(t)\}_{t=1}^{k-N_p}$ obtained from the previous iterations of the moving-horizon estimation algorithm. In-turn, the sequence of active modes is optimized only within the considered time horizon in (3a).

Increasing N_p increases the information used to cluster the regressor $x(k)$ and to estimate the model parameters Θ_i. On the one hand, increasing N_p increases the number of binary decision variables δ_i in (3). Thus, the length N_p of the horizon provides a trade off between complexity of the optimization problem (3), and accuracy in estimating the model parameters Θ_i and in clustering the regressor $x(k)$.

Recursive update of the objective function

Note that, at a first glance, the regularization cost (3b) requires to use, and thus to store, the whole time-history of observations up to time $k - N_p$ (i.e., the sequence of regressor/output pairs $\{x(k), y(k)\}_{k=1}^{k-N_p}$). Nevertheless, once a new observation is available at time k, the term (3b) can be recursively updated, as described in the following.

Let us rewrite the regularization term (3b) as

$$\sum_{t=1}^{k-N_p} \operatorname{tr}\left(\left(y(t) - \Theta_{\sigma(t)} \left[\begin{smallmatrix} 1 \\ x(t) \end{smallmatrix}\right]\right) \left(y(t) - \Theta_{\sigma(t)} \left[\begin{smallmatrix} 1 \\ x(t) \end{smallmatrix}\right]\right)' \right) =$$

$$\operatorname{tr}\left(\sum_{t=1}^{k-N_p} \Theta_{\sigma(t)} \left[\begin{smallmatrix} 1 \\ x(t) \end{smallmatrix}\right] \left[\begin{smallmatrix} 1 \\ x(t) \end{smallmatrix}\right]' \Theta_{\sigma(t)}' \right) -$$

[1]A collection $\{\mathcal{X}_i\}_{i=1}^{s}$ is a complete partition of the regressor domain $\mathcal{X}$ if $\bigcup_{i=1}^{s} \mathcal{X}_i = \mathcal{X}$ and $\mathring{\mathcal{X}}_i \cap \mathring{\mathcal{X}}_j = \emptyset$, $\forall i \neq j$, with $\mathring{\mathcal{X}}_i$ denoting the interior of $\mathcal{X}_i$.

$$2\mathrm{tr}\left(\sum_{t=1}^{k-N_p} \Theta_{\sigma(t)} \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right] y(t)'\right) + \mathrm{tr}\left(\sum_{t=1}^{k-N_p} y(t)y(t)'\right), \quad (5)$$

with $\mathrm{tr}(\cdot)$ denoting the matrix trace. Let us now define the matrices

$$H_i(k-N_p) = \sum_{t=1}^{k-N_p} \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right] \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right]' h_i(t), \quad (6a)$$

$$F_i(k-N_p) = \sum_{t=1}^{k-N_p} \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right] y(t)' h_i(t), \quad (6b)$$

with $h_i(t) = 1$, if $\sigma(t) = i$ or $h_i(t) = 0$, otherwise. Substituting (6) into the cost (5), we can represent (3b) as

$$\sum_{t=1}^{k-N_p} \left\|y(t) - \Theta_{\sigma(t)} \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right]\right\|^2 = \mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i H_i(k-N_p)\Theta_i'\right)$$
$$- 2\mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i F_i(k-N_p)\right) + \mathrm{tr}\left(\sum_{t=1}^{k-N_p} y(t)y(t)'\right). \quad (7)$$

Note that the matrices $H_i(k-N_p)$ and $F_i(k-N_p)$ can be recursively computed as

$$H_i(k-N_p) = H_i(k-N_p-1) + \left[\begin{smallmatrix}1\\x(k-N_p)\end{smallmatrix}\right] \left[\begin{smallmatrix}1\\x(k-N_p)\end{smallmatrix}\right]' h_i(k-N_p), \quad (8a)$$

$$F_i(k-N_p) = F_i(k-N_p-1) + \left[\begin{smallmatrix}1\\x(k-N_p)\end{smallmatrix}\right] y(k-N_p)' h_i(k-N_p). \quad (8b)$$

Thus, processing the observation $\{x(k), y(k)\}$ just requires to update the matrices $H_i(k-N_p-1)$ and $F_i(k-N_p-1)$ through (8), with no need to store the time history of observations $\{x(k), y(k)\}_{t=1}^{k-N_p-1}$.

MIQP formulation

To reformulate (3) as an MIQP problem, let us define the vector $z_i(k) \in \mathbb{R}^{n_y}$ with $\delta_i(k) \in \{0,1\}$ as

$$z_i(k) = \left(y(k) - \Theta_i \left[\begin{smallmatrix}1\\x(k)\end{smallmatrix}\right]\right) \delta_i(k). \quad (9a)$$

Note that:

$$z_i(k) = \begin{cases} y(k) - \Theta_i \left[\begin{smallmatrix}1\\x(k)\end{smallmatrix}\right] & \text{if } \delta_i(k) = 1 \\ 0 & \text{if } \delta_i(k) = 0 \end{cases} \quad (9b)$$

Let M and m be an arbitrary large (resp. small) upper (resp. lower) bound of the elements of the vector $y(k) - \Theta_i \left[\begin{smallmatrix}1\\x(k)\end{smallmatrix}\right]$,

$$m \leq y(k) - \Theta_i \left[\begin{smallmatrix}1\\x(k)\end{smallmatrix}\right] \leq M. \quad (9c)$$

Based on conditions (7) and (9), problem (3) can be equivalently written as the MIQP problem

$$\min_{\substack{\{\Theta_i\}_{i=1}^{s} \\ \{\delta_i(k-t)\}_{i=1,t=0}^{s,N_p-1} \\ \{z_i(k-t)\}_{i=1,t=0}^{s,N_p-1}}} \sum_{t=0}^{N_p-1} \sum_{i=1}^{s} z_i^2(k-t) + \quad (10a)$$

$$\mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i H_i(k-N_p)\Theta_i'\right) - 2\mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i F_i(k-N_p)\right), \quad (10b)$$

$$\text{s.t. } z_i(k-t) \leq M\delta_i(k-t), \quad (10c)$$
$$z_i(k-t) \geq m\delta_i(k-t), \quad (10d)$$
$$z_i(k-t) \leq y(k-t) - \Theta_i \left[\begin{smallmatrix}1\\x(k-t)\end{smallmatrix}\right] - m(1-\delta_i(k-t)), \quad (10e)$$
$$z_i(k-t) \geq y(k-t) - \Theta_i \left[\begin{smallmatrix}1\\x(k-t)\end{smallmatrix}\right] - M(1-\delta_i(k-t)), \quad (10f)$$
$$\sum_{i=1}^{s} \delta_i(k-t) = 1, \; t = 0, \ldots, N_p - 1, \quad (10g)$$
$$\delta_i(k-t) \in \{0,1\}, \; i = 1, \ldots, s, \quad (10h)$$

Based on the constructed problem (10), different MIQP solvers can be chosen to solve (10). For small-scale problems, the accelerated dual gradient projection (GPAD for short) [21] coupled with *Branch and Bound* (B&B) method (GPAD-B&B) can be used. The GPAD algorithm is very simple as it only needs basic arithmetic computations, which is used to solve the Quadratic Programming (QP) relaxations arising in B&B [16]. GUROBI solver can be used to solve medium and large-scale problems.

Summary and iterative refinement

The steps described so far for recursive clustering of the regressors $\{x(k)\}_{k=1}^{N}$ and for model parameters Θ_i estimation are summarized in Algorithm 1. At the beginning of Algorithm 1, a mini-batch identification problem is solved to estimate the sequence of active modes $\sigma(t)$ from time 1 up to time N_p and to assign the regressor $\{x(t)\}_{t=1}^{N_p}$ to the cluster $\{\mathcal{C}_{\sigma(t)}\}_{t=1}^{N_p}$ (stages 2-6). Then, the observations $\{x(k), y(k)\}$ are processed iteratively. Besides updating the model parameters Θ_i at each time k (stage 7.3), the active mode $\sigma(k)$ is estimated (stages 7.4-7.6) and the regressor $x(k)$ is consequently assigned to cluster $\mathcal{C}_{\sigma(k)}$ (stage 7.7).

It is worth pointing out that, at the first iterations of Algorithm 1 (i.e., for $\bar{k} \ll N$), the observations $\{x(t), y(t)\}_{t=1}^{\bar{k}}$ may be wrongly classified, as the learning phase is based on a "small" set of observations. An error in the classification of the pairs $\{x(t), y(t)\}_{t=1}^{\bar{k}}$ (i.e., an incorrect estimate of the mode sequence $\{\sigma(t)\}_{t=1}^{\bar{k}}$) also influences the estimate of the active modes and of the model parameters Θ_i at the next time samples $k > \bar{k}$, as the regularization cost (3b) depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{\bar{k}}$. When working in a batch mode, the effect of the initial classification error may be reduced by running Algorithm 1 multiple times, including in the regularization term (3b) also the sequences of active modes estimated at the previous runs. More specifically, at the n_q-th run of Algorithm 1, the following cost is considered instead of (3a)-(3b):

$$\sum_{i=1}^{s} \sum_{t=0}^{N_p-1} \gamma_1 \left\|\left(y(k-t) - \Theta_i \left[\begin{smallmatrix}1\\x(k-t)\end{smallmatrix}\right]\right) \delta_i(k-t)\right\|^2 + \quad (11a)$$

$$\gamma_2 \sum_{t=1}^{k-N_p} \left\|y(t) - \Theta_{\sigma(t,n_q)} \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right]\right\|^2 + \quad (11b)$$

$$\sum_{q=1}^{n_q-1} \lambda^{n_q-q-1} \sum_{t=1}^{N-N_p} \left\|y(t) - \Theta_{\sigma(t,q)} \left[\begin{smallmatrix}1\\x(t)\end{smallmatrix}\right]\right\|^2, \quad (11c)$$

with $\sigma(t,q)$ $(q = 1, \ldots, n_q)$ being the estimate of the active

Algorithm 1 Recursive clustering of the regressors and model parameters estimation

Input: Observations sequence $\{x(k), y(k)\}_{k=1}^{N}$; number of modes s; horizon N_p.

```
1. let $H_i(0) \leftarrow 0$, $F_i(0) \leftarrow 0$, $\mathcal{C}_i \leftarrow \emptyset$, $i = 1, \dots, s$;
2. let $k \leftarrow N_p$;
3. solve the MIQP problem (10);
4. let $\{\delta_i^*(t)\}_{i=1,t=1}^{s,N_p}$ be the optimal parameters minimizing (10);
5. for $t = 1, \dots, N_p$ do
   5.1. let $i^*(t)$ be the index such that $\delta_{i^*}^*(t) = 1$;
   5.2. let $\sigma(t) \leftarrow i^*(t)$;
   5.3. let $\mathcal{C}_{\sigma(t)} \leftarrow \mathcal{C}_{\sigma(t)} \cup \{x(t)\}$;
6. end for
7. for $k = N_p + 1, \dots, N$ do
   7.1. update the matrices $H_i(k - N_p)$ and $F_i(k - N_p)$ through (8);
   7.2. solve the MIQP problem (10);
   7.3. let $\Theta_i^*(k)$ be the optimal parameters minimizing (10), $i = 1, \dots, s$;
   7.4. let $\{\delta_i^*(k-t)\}_{t=0}^{N_p-1}$ be the optimal parameters minimizing (10), $i = 1, \dots, s$;
   7.5. let $i^*$ be the index such that $\delta_{i^*}^*(k) = 1$;
   7.6. let $\sigma(k) \leftarrow i^*$;
   7.7. let $\mathcal{C}_{\sigma(k)} \leftarrow \mathcal{C}_{\sigma(k)} \cup x(k)$;
8. end for;
```

Output: Estimated parameters $\Theta_1^*(N), \dots, \Theta_s^*(N)$; clusters $\mathcal{C}_1, \dots, \mathcal{C}_s$; sequence of active modes $\{\sigma(k)\}_{k=1}^{N}$.

mode at time t obtained at the q-th run of Algorithm 1. Note that (11c) is a regularization term based on the past runs of Algorithm 1, while (11b) plays the same role of (3b), as it regularizes the parameters Θ_i based on the estimate $\{\sigma(t)\}_{t=1}^{k-N_p}$ obtained at the current run of Algorithm 1. A forgetting factor $\lambda \in \mathbb{R} : 0 < \lambda \leq 1$ is also included in (11c) to exponentially downweight the estimates $\{\sigma(t,q)\}_{t=1}^{N}$ obtained at past runs of Algorithm 1. The regularization parameters $\gamma_1, \gamma_2 \in \mathbb{R}$ are introduced in (11a) and (11b), respectively.

Similar to the original regularization term (3b), the cost (11) can be also recursively updated as a new sample $\{x(k), y(k)\}$ is processed, without the need to store the whole time history of estimates $\{\sigma(k,q)\}_{k=1,q=1}^{N,n_q-1}$ obtained at the previous runs of Algorithm 1. As a matter of fact, the cost (11) can be written as

$$\sum_{i=1}^{s}\sum_{t=0}^{N_p-1} \gamma_1 \left\| \left(y(k-t) - \Theta_i \left[\begin{smallmatrix}1\\ x(k-t)\end{smallmatrix}\right]\right) \delta_i(k-t) \right\|^2 + \quad (12a)$$

$$\gamma_2 \left\{ \mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i H_i(k-N_p, n_q) \Theta_i' \right) - \right. \quad (12b)$$

$$2\mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i F_i(k-N_p, n_q) \right) + \quad (12c)$$

$$\left. \mathrm{tr}\left(\sum_{t=1}^{k-N_p} y(t)y(t)' \right) \right\} + \quad (12d)$$

$$\sum_{q=1}^{n_q-1} \mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i \lambda^{n_q-q-1} H_i(N-N_p, q) \Theta_i' \right) - \quad (12e)$$

$$2\sum_{q=1}^{n_q-1} \mathrm{tr}\left(\sum_{i=1}^{s} \Theta_i \lambda^{n_q-q-1} F_i(N-N_p, q) \right) + \quad (12f)$$

$$\sum_{q=1}^{n_q-1} \mathrm{tr}\left(\lambda^{n_q-q-1} \sum_{t=1}^{N-N_p} y(t)y(t)' \right), \quad (12g)$$

where $H_i(N-N_p, q)$ and $F_i(N-N_p, q)$ ($q = 1, \dots, n_q$) are defined similarly to (6) and computed based on the estimates $\sigma(t,q)$ computed at the q-th run of Algorithm 1. Specifically:

$$H_i(N-N_p, q) = \sum_{t=1}^{N-N_p} \left[\begin{smallmatrix}1\\ x(t)\end{smallmatrix}\right] \left[\begin{smallmatrix}1\\ x(t)\end{smallmatrix}\right]' h_i(t,q),$$

$$F_i(N-N_p, q) = \sum_{t=1}^{N-N_p} \left[\begin{smallmatrix}1\\ x(t)\end{smallmatrix}\right] y(t)' h_i(t,q),$$

with $h_i(t,q) = 0$, if $\sigma(t,q) = i$ or $h_i(t,q) = 1$, otherwise.

When the pair $\{x(k), y(k)\}$ is processed, the matrices $H_i(k-N_p, n_q)$ and $F_i(k-N_p, n_q)$ in (12b)-(12c) can be recursively updated through (8), while only the matrices $H_i(N-N_p, q)$ and $F_i(N-N_p, q)$ (with $q = 1, \dots, n_q - 1$) are needed to construct the terms (12e)-(12f).

B. Construction of the state partition

The partition $\{\mathcal{X}_i\}_{i=1}^{s}$ of the regressor space $\mathcal{X}$ can be found along with the estimation of the model parameters $\{\Theta_i\}_{i=1}^{s}$ and the sequence of active modes $\{\sigma(k)\}_{k=1}^{N}$. This is done by separating the computed clusters $\{\mathcal{C}_i\}_{i=1}^{s}$ using linear multicategory discrimination.

In the following subsection, we briefly describe the algorithms recently presented in [10], which are suited both for offline and online (i.e., recursive) computation of the state partition.

Linear multicategory discrimination: problem formulation

According to the formulation introduced in [7], the linear multicategory discrimination problem is tackled by searching for a convex piecewise affine separator function $\phi : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ discriminating between the clusters $\mathcal{C}_1, \dots, \mathcal{C}_s$. The separator function ϕ is defined as

$$\phi(x) = \max_{i=1,\dots,s} \left(\left[\begin{smallmatrix} x' & -1 \end{smallmatrix}\right] \left[\begin{smallmatrix}\omega^i\\ \gamma^i\end{smallmatrix}\right] \right), \quad (14)$$

where $\omega^i \in \mathbb{R}^{n_x}$ and $\gamma^i \in \mathbb{R}$ are the parameters to be computed. Let m_i denote the cardinality of the cluster $\mathcal{C}_i$ and let $M_i \in \mathbb{R}^{m_i \times n_x}$, for $i = 1, \dots, s$, which is obtained by stacking the regressors $x(k)'$ belonging to $\mathcal{C}_i$ in its rows.

If the clusters $\{\mathcal{C}_i\}_{i=1}^{s}$ are linearly separable, then the separator function ϕ satisfies the following conditions:

$$\left[\begin{matrix} M_i & -\mathbf{1}_{m_i} \end{matrix}\right] \left[\begin{smallmatrix}\omega^i\\ \gamma^i\end{smallmatrix}\right] \geq \left[\begin{matrix} M_i & -\mathbf{1}_{m_i} \end{matrix}\right] \left[\begin{smallmatrix}\omega^j\\ \gamma^j\end{smallmatrix}\right] + \mathbf{1}_{m_i}, \quad (15)$$
$$i, j = 1, \dots, s, \; i \neq j,$$

where $\mathbf{1}_{m_i}$ is an m_i-dimensional vector of ones.

The piecewise-affine separator ϕ thus satisfies the conditions:

$$\begin{cases} \phi(x) = [x' \quad -1] \begin{bmatrix} \omega^i \\ \gamma^i \end{bmatrix}, \ \forall x \in \mathcal{C}_i, \quad i = 1, \dots, s \\ \phi(x) \geq [x' \quad -1] \begin{bmatrix} \omega^j \\ \gamma^j \end{bmatrix} + 1, \ \forall x \in \mathcal{C}_i, \ i \neq j \end{cases} \tag{16}$$

From (16), the polyhedra $\{\mathcal{X}_i\}_{i=1}^s$ are defined as

$$\mathcal{X}_i = \left\{ x \in \mathbb{R}^{n_x} : [x' \quad -1] \begin{bmatrix} \omega^i - \omega^j \\ \gamma^i - \gamma^j \end{bmatrix} \geq 1, \ j = 1, \dots, s, \ j \neq i \right\}.$$

Off-line multicategory discrimination

The parameters $\{\omega^i, \gamma^i\}_{i=1}^s$ are calculated by solving the optimization problem which is convex [10] (instead of solving a *robust linear programming* (RLP) problem as in [7]),

$$\min_{\xi} \frac{\kappa}{2} \sum_{i=1}^{s} \left(\|\omega^i\|_2^2 + (\gamma^i)^2 \right) + \sum_{i=1}^{s} \sum_{\substack{j=1 \\ j \neq i}}^{s} \frac{1}{m_i} \left\| \left(\left[M_i \ -\mathbf{1}_{m_i} \right] \begin{bmatrix} \omega^j - \omega^i \\ \gamma^j - \gamma^i \end{bmatrix} + \mathbf{1}_{m_i} \right)_+ \right\|_2^2, \tag{17}$$

with $\xi = \left[(\omega^1)' \dots (\omega^s)' \ \gamma^1 \dots \gamma^s \right]'$. Problem (17) minimizes the averaged squared 2-norm of the violation of the inequalities in (15). The regularization parameter $\kappa > 0$ guarantees that the objective function (17) is strongly convex. Problem (17) is then solved through the *Regularized Piecewise-Smooth Newton* (RPSN) method explained in [10] and originally proposed in [3].

Recursive multicategory discrimination

An online approach can be used, either in place of the off-line approach or to refine the partition ϕ online based on streaming data. A recursive approach using on-line convex programming can be used to solve (17).

Considering the data-points $x \in \mathbb{R}^{n_x}$ as random vectors and let us assume that there exists an oracle function $i : \mathbb{R}^{n_x} :\rightarrow \{1, \dots, s\}$ that assigns the corresponding mode $i(x) \in \{1, \dots, s\}$ to a given $x \in \mathbb{R}^{n_x}$. By definition, function i describes the clusters in the data-point space $\mathbb{R}^{n_x}$. Let us also assume that the following probabilities

$$\pi_i = \text{Prob}[i(x) = i] = \int_{\mathbb{R}^{n_x}} \delta(i, i(x)) p(x) dx,$$

are known for all $i = 1, \dots, s$, where $\delta(i, j) = 1$ if $i = j$, zero otherwise. Problem (17) can be the generalized as the following convex regularized stochastic optimization problem

$$\xi^* = \min_{\xi} E_{x \in \mathbb{R}^{n_x}} \left[\ell(x, \xi) \right] + \frac{\kappa}{2} \|\xi\|_2^2 \tag{18}$$

$$\ell(x, \xi) = \sum_{\substack{j=1 \\ j \neq i(x)}}^{s} \frac{1}{\pi_{i(x)}} \left(x'(\omega^j - \omega^{i(x)}) - \gamma^j + \gamma^{i(x)} + 1 \right)_+^2,$$

where $E_x[\cdot]$ denotes the expected value w.r.t. x. Problem (18) aims at violating the least, on average over x, the condition in (15) for $i = i(x)$. The coefficients π_i can be estimated offline from a data subset, specifically $\pi_i = \frac{m_i}{N}$, and can be updated iteratively. Nevertheless, numerical experiments have shown that uniform coefficients $\pi = \frac{1}{s}$ work equally well. Problem (18) can be solved online using convex optimization algorithm, *Averaged Stochastic Gradient Descent* (ASGD) method described in [10].

IV. SIMULATION EXAMPLE

The performance of the proposed PWA regression algorithm is shown via identification of *PieceWise Affine autoRegressive with eXogenous input* (PWARX) systems in this section. The output sequence used for training is corrupted by a zero mean white Gaussian noise process e_o. The *Signal-to-Noise Ratio* (SNR) index

$$\text{SNR} = 10 \log \frac{\sum_{t=1}^{N} (y(t) - e_o(t))^2}{\sum_{t=1}^{N} (e_o(t))^2}, \tag{19}$$

quantifies the effect of the measurement noise on the output. The quality of the identified models is assessed on a noiseless validation dataset (not used for training) through the *Best Fit Rate* (BFR) index

$$\text{BFR} = \max \left\{ 1 - \sqrt{\frac{\sum_{k=1}^{N_{\text{val}}} \left(y(k) - \hat{y}(k) \right)^2}{\sum_{k=1}^{N_{\text{val}}} \left(y(k) - \bar{y} \right)^2}}, 0 \right\}, \tag{20}$$

with N_{val} being the length of the validation set and $\hat{y}$ being the estimated model output and $\bar{y}$ the sample mean of the output signal. All the simulations are carried out using a desktop computer with MATLAB R2015a, Intel Core i7-4700MQ CPU with 2.40 GHz and 8 GB of RAM.

A. Identification of SISO PWARX system

As a first example, we consider a *single-input single-output* (SISO) PWARX system for the data generation, described by the difference equation

$$\begin{aligned} y(k) = & 0.8y(k-1) + 0.4u(k-1) - 0.1 + \max\{-0.3y(k-1) \\ & + 0.6u(k-1) + 0.3, 0\} + e_o(k), \end{aligned}$$

with $\bar{s} = 2$ modes, based on the possible combinations generated by the sign of the "max" operator. To gather the data, the system is excited by an input $u(k)$ which is chosen to be white noise with uniform distribution $\mathcal{U}(-1, 1)$ and length $N = 1000$, $e_o(k) \in \mathbb{R}$ is a zero-mean white Gaussian noise with variance $\sigma_e^2 = 6.25 \cdot 10^{-4}$. This corresponds to a SNR on the output channel equal to 20 dB.

For identification, Algorithm 1 is run for 3 iterations (i.e. $n_q = 3$) with $s = \bar{s} = 2$ and with prediction horizon $N_p = 5$, forgetting factor $\lambda = 0.01$, $\gamma_1 = 10$, $\gamma_2 = 1$. The resultant MIQP problem (10) consists of 26 variables out of which 10 are binary, 40 inequality and 5 equality constraints. The MIQP problem is solved with the recently proposed GPAD-B&B algorithm [16] and the performance is compared with the commercial solver GUROBI. In the second stage, off-line multicategory discrimination algorithm (section III-B) is executed for the partitioning of the regressor space, with parameter $\kappa = 10^{-5}$. The BFR on the noise-free validation data-set of length $N_{\text{val}} = 300$ are summarized in Table (I). The mean time taken to process a single training sample

TABLE I

BFR ON THE VALIDATION DATA SET FOR SISO PWARX SYSTEM.

runs(n_q)	GUROBI	GPAD-B&B
1	0.86	0.86
2	0.90	0.89
3	0.92	0.90

by GPAD-B&B is 0.13 sec with the feasibility tolerance ϵ_G=$1 \cdot 10^{-3}$, optimality tolerance ϵ_V=$1 \cdot 10^{-3}$, infeasibility detection tolerance $\epsilon_I = 1 \cdot 10^{-3}$, whereas GUROBI with default settings takes 0.09 sec. GPAD-B&B makes a trade off between the execution time and quality of solution, by selecting appropriate tolerance values. It is simple library-free solver, yet rendered comparable performance with respect to the commercial solver for the given problem.

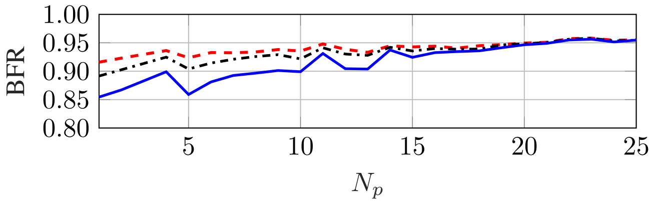

Fig. 1. BFR vs N_p : —— $n_q = 1$, - · - · $n_q = 2$, - - - $n_q = 3$.

The effect of increasing the prediction horizon N_p on the BFR is shown in fig. 1, for $n_q = 1, 2$ and 3 runs respectively.

B. Identification of MIMO PWARX system

As a second example, the following *Multiple-Input Multiple-Output* (MIMO) PWARX data generating system, taken from [9], is considered

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} -0.83 & 0.20 \\ 0.30 & -0.52 \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} -0.34 & 0.45 \\ -0.30 & 0.24 \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix}$$
$$+ \begin{bmatrix} 0.20 \\ 0.15 \end{bmatrix} + \max \left\{ \begin{bmatrix} 0.20 & -0.90 \\ 0.10 & -0.42 \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} \right.$$
$$\left. + \begin{bmatrix} 0.42 & 0.20 \\ 0.50 & 0.64 \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + \begin{bmatrix} 0.40 \\ 0.30 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\} + e_o(k),$$

described by $\bar{s} = 4$ modes, based on the possible combinations generated by the sign of the vector-valued "max" operator. To gather the data, the input sequence $u(k)$ is chosen to be a white noise process of length $N = 3000$, having uniform distribution in the boxes $[-2\ 2] \times [-2\ 2]$. The noise signal $e_o \in \mathbb{R}^2$ is a white Gaussian noise with covariance matrix $\Lambda_e = \begin{bmatrix} 2.5 \cdot 10^{-3} & 0 \\ 0 & 2.5 \cdot 10^{-3} \end{bmatrix}$. This results in SNRs equal to 27 dB and 26 dB for the first and the second output channel, respectively.

Algorithm (1) is run for one iteration (i.e. $n_q = 1$) with $s = \bar{s} = 4$ and with a prediction horizon $N_p = 10$, $\gamma_1 = 1$, $\gamma_2 = 1$. GUROBI is used to solve the MIQP problem (10). In the second stage, off-line multicategory discrimination algorithm (reported in section III-B) is run with parameter $\kappa = 10^{-5}$, for the partitioning of the regressor space. The achieved results in terms of BFR are 0.95 and 0.94 for channel 1 and 2 respectively on a noise-free validation data set of length $N_{\text{val}} = 500$.

V. CONCLUSIONS

A novel moving-horizon algorithm for *PieceWise Affine* regression has been described in this paper. The proposed method combines the advantages of the mixed-integer programming method [22] (namely, simultaneous choice of the model parameters and of the optimal sequence of active modes within a relatively short time horizon) and the recursive algorithm [10] (namely, computational efficiency and iterative processing of the training samples).

ACKNOWLEDGMENT

The authors would like to thank Valentina Breschi for her help with the multicategory discrimination algorithm.

REFERENCES

[1] L. Bako. Identification of switched linear systems via sparse optimization. *Automatica*, 47(4):668–677, 2011.

[2] L. Bako, K. Boukharouba, E. Duviella, and S. Lecoeuche. A recursive identification algorithm for switched linear/affine models. *Nonlinear Analysis: Hybrid Systems*, 5(2):242–253, 2011.

[3] A. Bemporad, D. Bernardini, and P. Patrinos. A convex feasibility approach to anytime model predictive control. Technical report, IMT Institute for Advanced Studies, Lucca, February 2015. `http://arxiv.org/abs/1502.07974`.

[4] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE transactions on automatic control*, 45(10):1864–1876, 2000.

[5] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino. A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10):1567–1580, 2005.

[6] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[7] K.P. Bennett and O.L. Mangasarian. Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3:27–39, 1994.

[8] L. Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999–1013, 1993.

[9] V. Breschi, A. Bemporad, and D. Piga. Identification of hybrid and linear parameter varying models via recursive piecewise affine regression and discrimination. In *Proc. 15th European Control Conference*, pages 2632–2637, 2016.

[10] V. Breschi, D. Piga, and A. Bemporad. Piecewise affine regression via recursive multiple least squares and multicategory discrimination. *Automatica*, 73:155–162, 2016.

[11] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

[12] A. Garulli, S. Paoletti, and A. Vicino. A survey on switched and piecewise affine system identification. In *Proc. 16th IFAC Symposium on System Identification*, pages 344–355, Brussels, Belgium, 2012.

[13] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.

[14] A. L. Juloski, S. Weiland, and W.P.M.H. Heemels. A Bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50(10):1520–1533, 2005.

[15] F. Lauer. On the complexity of piecewise affine system identification. *Automatica*, 62:148–153, 2015.

[16] V. V. Naik and A. Bemporad. Embedded mixed-integer quadratic optimization using accelerated dual gradient projection. In *Proc. 20th IFAC World Congress*, Toulouse, France, 2017.

[17] H. Nakada, K. Takaba, and T. Katayama. Identification of piecewise affine systems based on statistical clustering technique. *Automatica*, 41(5):905–913, 2005.

[18] H. Ohlsson and L. Ljung. Identification of switched linear regression models using sum-of-norms regularization. *Automatica*, 49(4):1045–1050, 2013.

[19] N. Ozay, C. Lagoa, and M. Sznaier. Set membership identification of switched linear systems with known number of subsystems. *Automatica*, 51:180–191, 2015.

[20] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal. Identification of hybrid systems a tutorial. *European journal of control*, 13(2):242–260, 2007.

[21] P. Patrinos and A. Bemporad. An accelerated dual gradient-projection algorithm for embedded linear model predictive control. *IEEE Trans. Automatic Control*, 59(1):18–33, 2014.

[22] J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004.