MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Building Temperature Control by Simple MPC-like Feedback Laws Learned from Closed-Loop Data

Klauco, M.; Drgona, J.; Kvasnica, M.; Di Cairano, S.

TR2014-076 August 2014

Abstract

We show how to synthesize simple, yet well-performing feedback strategies that mimic the behavior of optimization-based controllers, such as those based on model predictive control (MPC). The approach is based on employing regression trees to derive dependence of real-valued control inputs on measurements. Quality of classical regression policies is improved by finding, simultaneously, optimal affine splits and optimal local affine regressors. We furthermore illustrate how to refine the local regressors such that the overall feedback strategy guarantees satisfaction of input constraints. The main advantage of the proposed regression- based control strategy stems from its fast implementation even on very simple hardware. The approach is demonstrated on a case study that assumes control of temperature in a one-zone building. Here, the data used in the learning process are generated by MPC. We show that the simple feedback law attains almost the same level of performance as the complex MPC controller.

IFAC 2014

Building Temperature Control by Simple MPC-like Feedback Laws Learned from Closed-Loop Data

Martin Klaučo [*], **Ján Drgoňa** [*], **Michal Kvasnica** [*], **Stefano Di Cairano** [**]

[*] *Slovak University of Technology in Bratislava, Slovakia (e-mail: {martin.klauco, jan.drgona, michal.kvasnica}@stuba.sk).*
[**] *Mitsubishi Electric Research Laboratories, Cambridge, MA, USA (e-mail:dicairano@ieee.org).*

Abstract: We show how to synthesize simple, yet well-performing feedback strategies that mimic the behavior of optimization-based controllers, such as those based on model predictive control (MPC). The approach is based on employing regression trees to derive dependence of real-valued control inputs on measurements. Quality of classical regression policies is improved by finding, simultaneously, optimal affine splits and optimal local affine regressors. We furthermore illustrate how to refine the local regressors such that the overal feedback strategy guarantees satisfaction of input constraints. The main advantage of the proposed regression-based control strategy stems from its fast implementation even on very simple hardware. The approach is demonstrated on a case study that assumes control of temperature in a one-zone building. Here, the data used in the learning process are generated by MPC. We show that the simple feedback law attains almost the same level of performance as the complex MPC controller.

Keywords: Predictive control for linear systems, Energy systems, Regression.

1. INTRODUCTION

According to the survey conducted by (Parry et al., 2007), the total energy consumed in heating, cooling, ventilation and air-conditions (HVAC) in commercial and residential buildings nowadays accounts for 40 % of global energy consumption. Thus energy-efficient approaches to HVAC control can significantly reduce overall level of pollution and mitigate emissions of greenhouse gases.

One of the control methodologies which can explicitly account for minimization of consumed energy is model predictive control, MPC. In MPC, control inputs that minimize a certain objective function (which accounts for consumption of energy and maximization of thermal comfort) are computed by solving a suitable optimization problem at each sampling instant. MPC approaches have been successfully applied to control of buildings with significant energy savings being reported, see, e.g., Oldewurtel et al. (2012); Cigler et al. (2013). Although several approches for a fast implementation of MPC have been suggested previously (Ma et al., 2012), the task is very challenging especially when the control algorithm has to be implemented on existing control hardware, such as on programmable logic controllers (PLC), which are predominant in building automation systems.

Therefore we aim at constructing a control policy that performs almost as good as one based on MPC, but offers a much simpler implementation. One way of achieving this goal is to calculate the explicit representation of the MPC feedback law (Bemporad et al., 2002; Borrelli, 2003). For a rich class of MPC problems the explicit MPC feedback takes a form of a piecewise affine (PWA) function defined over a polyhedral domain of the state space. Obtaining the optimal control input then reduces to a mere evaluation of the PWA function. Such a task can be easily performed even on very simple hardware. The crucial limitation of explicit MPC, however, is that the complexity of the optimal PWA feedback grows exponentially with the prediction horizon. Therefore it can only be applied if the control hardware offers enough storage to accommodate the function. This, however, is not always a realistic assumption, since the size of explicit MPC solutions can easily exceed several megabytes. To reduce the complexity, various approximation techniques have been proposed in the literature, see, e.g., Domahidi et al. (2011, 2012).

In this paper we propose to construct a simple MPC-like feedback strategy by approximating a finite set of training data by a PWA function, encoded as a binary tree. The training data are generated by an implicit MPC policy. The advantage over explicit MPC is that we can control the complexity of the function, hence meeting the limitations of the control hardware. We suggest to construct the PWA feedback by applying regression trees (Breiman, 1993). Here, the regressor is constructed by splitting the training data into cells organized as a binary tree. However, standard regression trees are limited to splitting functions that are orthogonal hyperplanes, and the local regressors inside each cell are assumed to be constant. In this paper we show how to split the nodes optimally by general hyperplanes. Simultaneously, we construct affine

local regressors that provide better approximation of the training data. As a consequence, our regression-based feedback policy can be described as a PWA function defined over a polyhedral domain, akin to explicit MPC solutions. The added benefit is that the regression function is directly encoded as a binary tree, which offers a fast implementation even on simple hardware.

2. MODEL PREDICTIVE CONTROL

We consider control of a building whose dynamical behavior is captured by the state-update equation

$$\xi(t+\Delta) = g(\xi(t), u(t), d(t)), \tag{1}$$

where $\xi(t) \in \mathbb{R}^{n_\xi}$, $u(t) \in \mathbb{R}^{n_u}$, $d(t) \in \mathbb{R}^{n_d}$ denote the vector of states, the vector of control inputs and the vector of disturbances at time t, respectively. In MPC the objective is to obtain an open-loop optimal sequence of control inputs $U_N = \{u_0, \dots, u_{N-1}\}$ over some prediction horizon N by solving an optimization problem, initialized by the current measurements of the state $\xi(t)$ and current disturbances $d(t)$:

$$\min_{u_0,\dots,u_{N-1}} \sum_{k=0}^{N-1} \ell(\xi_k, u_k, d_0) \tag{2a}$$

$$\text{s.t. } \xi_{k+1} = g(\xi_k, u_k, d_k),\ k = 0, \dots, N-1, \tag{2b}$$

$$\xi_{k+1} \in \Xi,\ u_k \in \mathcal{U},\ k = 0, \dots, N-1, \tag{2c}$$

$$\xi_0 = \xi(t),\ d_0 = d(t), \tag{2d}$$

where ξ_k, u_k, d_k denote predictions of the respective variables at the k-th step of the prediction horizon. The fitness of particular choice of control inputs is determined by the stage cost function $\ell : \mathbb{R}^{n_\xi} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_d} \to \mathbb{R}$. Typically, $\ell(\cdot,\cdot,\cdot)$ penalizes the control effort and the tracking error Alternatively, the stage cost can account for temperature range control, or for optimization of the Predicted Mean Vote (PMV) index for maintaining optimal thermal comfort (Fanger, 1970). Finally, the optimal solution to (2) must respect state and input constraints in (2c), where $\Xi \subseteq \mathbb{R}^{n_\xi}$ and $\mathcal{U} \subseteq \mathbb{R}^{n_u}$ are non-empty sets.

Denote by x the vector which encapsulates all time-varying parameters of (2), i.e. the current building state $\xi(t)$, current and/or future disturbances $d(t), \dots, d(t + N\Delta)$, as well as any reference signals.Then the receding horizon feedback law is then given by

$$u(x(t)) = [\mathbf{I}\ \mathbf{0}\ \cdots\ \mathbf{0}]\, U_N, \tag{3}$$

where $\mathbf{I}$ and $\mathbf{0}$ represent, respectively, identity and zero matrices of appropriate dimensions. Note that the optimal open-loop sequence U_N in (3) is defined as the optimal solution of (2), formulated for a particular value of the initialization parameters $x(t)$. Therefore to obtain the optimal control action for a particular value of $x(t)$ from (3), one needs to solve (2) at each sampling instant. Such a procedure, however, requires significant computational effort and must be implemented on a hardware platform that allows to run optimization algorithms. Such a requirement is in contrast to our objective of implementing the control strategy on very simple hardware with severely limited computational and storage resources.

3. REGRESSION-BASED MPC-LIKE POLICY

In a regression a set of p training data[1] $\{(x_1, u_1), \dots, (x_p, u_p)\}$ is given with $x_i \in \mathbb{R}^{n_x}$ and $u_i \in \mathbb{R}^{n_u}$. The objective is to devise a regression function $f_{\text{reg}} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ which predicts the values of u (often called the *response variable*) that correspond to future measurements x as accurately as possible.

The central idea of the paper is to replace the implicitly defined feedback policy (3) by an explicit representation of the feedback law $u = f_{\text{reg}}(x)$, constructed by regression on a set of training data. The main advantage over the explicit MPC approach is that in regression we can control the complexity of $f_{\text{reg}}(\cdot)$ directly. In other words, we can devise the regression-based feedback strategy while considering limitations of the control hardware. The implied limitation of the approach is that the regression-based control policy is suboptimal with respect to the MPC cost function (2a). However, our objective in Section 5 will be to devise the regressor $f_{\text{reg}}(\cdot)$ that minimizes the deterioration of performance.

We propose to construct the regression-based feedback policy as follows:

1. Fix the number of training data p and select the set $\{x_1, \dots, x_p\}$ of initial values of parameters for the MPC problem (2).
2. For each x_i, obtain the corresponding optimal control move u_i from (3) by solving (2).
3. Collect x_i and u_i into the training data set $\{(x_1, u_1), \dots, (x_p, u_p)\}$ and devise a regressor $f_{\text{reg}}(\cdot)$ that predicts the value of the control move for an arbitrary vector of parameters x by $u = f_{\text{reg}}(x)$.

The selection of training parameters x_i in the first step of the proposed procedure can be performed in two ways. The first option is to grid the region of parameters of interest $\mathcal{P} \subseteq \mathbb{R}^{n_x}$ into $x_1, \dots, x_p$. While doing so provides representative samples x_i for the whole range of parameters, such an approach is only applicable if the dimension of the parametric space, i.e., n, is low. If n is large, an alternative way is to extract the pairs (x_i, u_i) from closed-loop profiles. Here, we propose to control the building by an MPC strategy of Section 2 for a limited amount of time. While doing so, we record values of $x(t)$ and the corresponding optimal control moves $u(t)$ at a fixed sampling rate. The training dataset is then composed of the tuples $(x(i\Delta), u(i\Delta))$ for $i = 0, \dots, i_{\max}$, where Δ is the collection period and $i_{\max}$ is the number of samples to be collected. However, computing $u(t)$ on-line via (2)−(3) is costly.

To address the issue of large computational load induced by using the implicit MPC strategy, represented by (2), we propose to run the optimization on a remote machine. Here, a distant server takes over all computation for a limited amount of time, say for a week. During this time the remote machine communicates with the building over the Internet. After collecting enough closed-loop data, the regressor $f_{\text{reg}}(\cdot)$ is constructed on the remote machine, and is subsequently uploaded to the local (simple) control hardware that resides directly in the building. From this

[1] Here, $x_i \in \mathbb{R}^{n_x}$ denotes the i-the realization of a vector x.

moment, the control commands are generated by the regressor locally and the remote server is no longer needed.

We are interested in finding an optimal regressor $f_{\text{reg}}(\cdot)$ which partitions the training data into M cells, denoted by $\mathcal{P}_1, \dots, \mathcal{P}_M$, such that $f_{\text{reg}}(\cdot)$ is well posed (Bemporad et al., 2002). Once the optimal cells $\mathcal{P}_i$ and the associated local regression functions $f_{\text{reg},i}(\cdot)$ are constructed, the overall regression-based control policy is given by

$$u = f_{\text{reg}}(x) := f_{\text{reg},i}(x) \text{ if } x \in \mathcal{P}_i. \quad (4)$$

In the following two sections we show how to construct $f_{\text{reg}}(\cdot)$ of (4) using binary regression trees where the individual cells $\mathcal{P}_i$ are polyhedra, as opposed to standardly used hyperboxes.

4. CONSTRUCTION OF REGRESSION TREES

A standard approach to generate regressors $f_{\text{reg}}(\cdot)$ that are well posed is to employ binary regression trees (Breiman, 1993). Such trees consist of a finite number of nodes, each of which may contain pointers to two child nodes. Nodes without any children are called *leaf nodes*. Each leaf node contains a local expression of the regressor, i.e., $f_{\text{reg},i}(\cdot)$. All non-leaf nodes, on the other hand, contain an expression of a splitting function $\sigma : \mathbb{R}^{n_{\text{x}}} \to \mathbb{R}$ and pointers to a maximum of two child nodes. The *left* child node is visited if $\sigma(x) \le 0$, while the *right* node is explored if $\sigma(x) > 0$.

The regression tree can be constructed by a recursive procedure, summarized as Algorithm 1. In particular, executing $\mathcal{T} = \texttt{treeNode}(\{\texttt{x}_1, \dots, \texttt{x}_\texttt{p}\}, \{\texttt{u}_1, \dots, \texttt{u}_\texttt{p}\})$ generates the root node of the tree and starts its recursive exploration. In particular, in Step 1 of Alg. 1 we need to determine the optimal splitting function $\sigma : \mathbb{R}^{n_{\text{x}}} \to \mathbb{R}$, along with optimal local regressors $f_{\text{L}} : \mathbb{R}^{n_{\text{x}}} \to \mathbb{R}^{n_{\text{u}}}$ and $f_{\text{R}} : \mathbb{R}^{n_{\text{x}}} \to \mathbb{R}^{n_{\text{u}}}$ that solve the following optimization problem:

$$\min_{\sigma, f_{\text{L}}, f_{\text{R}}} \left(\sum_{x_i \in \mathcal{P}_{\text{L}}} \|u_i - f_{\text{L}}(x_i)\| + \sum_{x_j \in \mathcal{P}_{\text{R}}} \|u_j - f_{\text{R}}(x_j)\| \right), \quad (5)$$

where

$$\mathcal{P}_{\text{L}} = \{x \mid \sigma(x) \le 0\},\ \mathcal{P}_{\text{R}} = \{x \mid \sigma(x) > 0\}, \quad (6)$$

are the cells generated by the split $\sigma(\cdot)$. Note that, since $\mathcal{P}_{\text{L}}$ and $\mathcal{P}_{\text{R}}$ depend on σ, problem (5) is nonlinear in the decision variables.

Once the optimal split and the optimal local regressors are computed, we need to determine whether the currently explored node needs to be subdivided. This decision is based on two criteria: the number of points in each of the split cells, and the regression error in each cell. The former is computed in Step 2 and the latter is evaluated in Step 3. If the number of points in the left cell (cf. (6)) drops below a pre-defined threshold $p_{\min}$, or when the local regression error is smaller than $e_{\min}$, exploration of the left cell is terminated and a leaf node containing the corresponding local regressor $f_{\text{L}}(\cdot)$ is returned in Step 5. Otherwise the left cell is explored recursively in Step 7. The right cell is treated similarly in Steps 9–13. Finally, the node, which consists of the split $\sigma(\cdot)$ and the pointers to child nodes $\mathcal{N}_{\text{L}}$, $\mathcal{N}_{\text{R}}$, is returned in Step 14.

The difficulty of constructing an optimal regression tree stems from the nonlinearity of the optimization problem

Algorithm 1 Procedure `treeNode`

INPUT: Training data $\{x_1, \dots, x_p\}$, response data $\{u_1, \dots, u_p\}$.
OUTPUT: Node of the binary tree.

1: Compute optimal splitting function $\sigma(\cdot)$ and optimal local regressors $f_{\text{L}}(\cdot)$ and $f_{\text{R}}(\cdot)$ from (5).
2: Split the set $\{1, \dots, p\}$ into subsets $\mathcal{L} = \{i \mid \sigma(x_i) \le 0\}$ and $\mathcal{R} = \{i \mid \sigma(x_i) > 0\}$. Denote $(x_{\text{L}}, u_{\text{L}}) = \{(x_i, u_i) \mid i \in \mathcal{L}\}$ and $(x_{\text{R}}, u_{\text{R}}) = \{(x_i, u_i) \mid i \in \mathcal{R}\}$.
3: Evaluate the regression errors

$$e_{\text{L}} = \sum_{i \in \mathcal{L}} \|u_i - f_{\text{L}}(x_i)\|,\ e_{\text{R}} = \sum_{j \in \mathcal{R}} \|u_j - f_{\text{R}}(x_j)\|. \quad (7)$$

4: **if** $\text{card}(x_{\text{L}}) < p_{\min}$ **or** $e_{\text{L}} < e_{\min}$ **then**
5: $\quad \mathcal{N}_{\text{L}} = \texttt{leafNode}(\texttt{f}_{\text{L}})$.
6: **else**
7: $\quad \mathcal{N}_{\text{L}} = \texttt{treeNode}(\texttt{x}_{\text{L}}, \texttt{u}_{\text{L}})$.
8: **end if**
9: **if** $\text{card}(x_{\text{R}}) < p_{\min}$ **or** $e_{\text{R}} < e_{\min}$ **then**
10: $\quad \mathcal{N}_{\text{R}} = \texttt{leafNode}(\texttt{f}_{\text{R}})$.
11: **else**
12: $\quad \mathcal{N}_{\text{R}} = \texttt{treeNode}(\texttt{x}_{\text{R}}, \texttt{u}_{\text{R}})$.
13: **end if**
14: **return** Tree node $\mathcal{N} = (\sigma, \mathcal{N}_{\text{L}}, \mathcal{N}_{\text{R}})$.

in (5)–(6) for general types of the split function $\sigma(\cdot)$ and of the local regressors $f_{\text{L}}(\cdot)$, $f_{\text{R}}(\cdot)$. To simplify the computation, standard regression tree approaches restrict splits to orthogonal hyperplanes of the form $\sigma = e^T x + \beta$, where the optimal vector e is selected from the finite set $\{[1, 0, \dots, 0]^T, [0, 1, 0, \dots, 0]^T, \dots, [0, \dots, 0, 1]^T\}$. Moreover, the local regressors are typically assumed to be constant functions, i.e., $f_{\text{L}}(x) = g_{\text{L}}$ and $f_{\text{R}}(x) = g_{\text{R}}$. Then Algorithm 1 generates a binary tree that encodes a piecewise constant regressor $f_{\text{reg}}(\cdot)$ where each cell $\mathcal{P}_i$ is an axis-aligned hyperbox. Clearly, such simplifications have adverse effect on quality of the regression. As a consequence, high number of nodes is typically required to achieve a desired regression error.

Therefore in the next section we show how to solve (5) when $\sigma(\cdot)$, $f_{\text{L}}(\cdot)$, $f_{\text{R}}(\cdot)$ are allowed to be linear (in fact, affine) functions. As a consequence, the cells $\mathcal{P}_i$ are allowed to be polyhedra, and the regression error in each node is decreased by employing linear regressors instead of constant functions.

5. OPTIMAL NODE SPLITTING WITH AFFINE SPLITS AND AFFINE REGRESSORS

First we show how to obtain optimal expressions of affine functions

$$\sigma(x) := \alpha^T x - \beta, \quad (8a)$$
$$f_{\text{L}}(x) := F_{\text{L}} x + g_{\text{L}}, \quad (8b)$$
$$f_{\text{R}}(x) := F_{\text{R}} x + g_{\text{R}} \quad (8c)$$

from (5)–(6) by solving a mixed-integer quadratic program (MIQP). Here, $\alpha \in \mathbb{R}^{n_{\text{x}}}$, $\beta \in \mathbb{R}$, $F_{\text{L}} \in \mathbb{R}^{n_{\text{u}} \times n_{\text{x}}}$, $g_{\text{L}} \in \mathbb{R}^{n_{\text{u}}}$, $F_{\text{R}} \in \mathbb{R}^{n_{\text{u}} \times n_{\text{x}}}$, $g_{\text{R}} \in \mathbb{R}^{n_{\text{u}}}$.

Lemma 1. Given are p datapoints $\{(x_1, u_1), \dots, (x_p, u_p)\}$. The optimal split $\sigma(\cdot)$ and optimal local regressors $f_{\text{L}}(\cdot)$, $f_{\text{R}}(\cdot)$ as in (8) that solve (5)–(6) are given as the optimal solution to the mixed-integer quadratic program (MIQP)

$$\min \sum_{i=1}^{p}(u_i - z_i)^T(u_i - z_i) \tag{9a}$$

$$\text{s.t. } -M(1-\delta_i) \le z_i - (F_{\mathrm{L}}x_i + g_{\mathrm{L}}) \le M(1-\delta_i), \tag{9b}$$

$$-M\delta_i \le z_i - (F_{\mathrm{R}}x_i + g_{\mathrm{R}}) \le M\delta_i, \tag{9c}$$

$$\alpha^T x_i \le \beta + M(1-\delta_i), \tag{9d}$$

$$\alpha^T x \ge \beta + \epsilon - M\delta_i, \tag{9e}$$

$$\|\alpha\|_\infty = 1, \tag{9f}$$

where the minimization is performed over continuous decision variables α, β, F_{L}, g_{L}, F_{R}, g_{R}, $z_i \in \mathbb{R}^{n_{\mathrm{u}}}$, and the binary variables $\delta_i \in \{0,1\}$, $i = 1,\dots,p$. Note that constraints in (9b)−(9e) are enforced for $i = 1,\dots,p$ with a small positive numerical tolerance ϵ and a sufficiently large constant M. ■

Although MIQPs are still nonconvex optimization problems due to presence of binary variables δ_i, they can be solved efficiently with state-of-the-art solvers.

Remark 2. The MIQP (8) is related to construction of so-called *hinging hyperplane* (HH) regressors that are frequently used to identify PWARX models of dynamical systems. Compared to the HH formulation of Roll et al. (2004), problem (8) has fewer binary optimization variables and is thus easier to solve as the number of datapoints increases. An another advantage is that construction of discontinuous regressors does not require additional binary/continuous variables as in the case of HH formulations. □

Remark 3. The reason for constraint (9f) is to rule out the trivial solution $\alpha = \beta = 0$ when all points are allocated only to one side of the split. Note that (9f) normalizes the largest value in α to ± 1. □

Remark 4. If M is chosen sufficiently large, problem (9) is always feasible. Rules of selecting a suitable constant M are discussed e.g. in Bemporad and Morari (1999). Note that finding a suitable M requires the decision variables of (9) to be bounded. □

Employing (9) in Step 1 of Algorithm 1 generates a node $\mathcal{N}$ that consists of an affine split $\sigma(\cdot)$. Such a split divides the space of independent variables x into polyhedra $\mathcal{P}_{\mathrm{L}}$, $\mathcal{P}_{\mathrm{R}}$ per (6). These polyhedra are then subdivided further by recursively invoking Algorithm 1 in Steps 7 and 12. Upon termination, the algorithm returns local affine regressors $f_{\mathrm{reg},i}(x)$, along with the corresponding regions of validity $\mathcal{P}_i$.

The overall PWA regressor is given per (4). $f_{\mathrm{reg}}(x)$ can be evaluated by searching sequentially through $\mathcal{P}_1,\dots,\mathcal{P}_M$, stopping once $x \in \mathcal{P}_i$. However, a more efficient way to evaluate $f_{\mathrm{reg}}(\cdot)$ is to directly traverse the binary tree starting from its root node. Here, the associate splitting function is evaluated and, based on its value,either the left or the right branch is explored. Such a procedure is repeated at each subsequent child node until a leaf node is encountered, whereupon the evaluation is stopped and the predicted value $u = f_{\mathrm{reg},i}(x)$ is returned. It is well known that the computational effort needed to evaluate $f_{\mathrm{reg}}(x)$ via a binary tree is $\mathcal{O}(\log_2 M)$, where M is the number of leaf nodes of the tree. The total memory storage is $\mathcal{O}(M)$. The data that need to be stored are the splits associated to each non-leaf node, pointers to child nodes, and local regressors in each leaf node.

6. REFINEMENT OF LOCAL REGRESSORS

Although the regression-based control policy (4) synthesized per Sections 4 and 5 provides an optimal regression of the training data, it does not posses guarantees that $u = f_{\mathrm{reg}}(x)$ satisfies input constraints $u \in \mathcal{U}$. Therefore in this section we show how to replace $f_{\mathrm{reg}}(\cdot)$ in (4) by a different function

$$\tilde{f}_{\mathrm{reg}}(x) := \tilde{f}_{\mathrm{reg},i}(x) \text{ if } x \in \mathcal{P}_i, \tag{10}$$

defined over the same cells $\mathcal{P}_1,\dots,\mathcal{P}_M$, but with the local regressors $\tilde{f}_{\mathrm{reg},i}(\cdot)$ being such that $u = \tilde{f}_{\mathrm{reg}}(x) \in \mathcal{U}$ for all $x \in \mathcal{P}$.

The case in which satisfaction of input constraints is easily obtained is when $\mathcal{P}$ is a polytope and $\mathcal{U} \subseteq \mathbb{R}^{n_{\mathrm{u}}}$ is a polyhedron. Consider the i-th terminal node of the binary tree, which is composed of the local regressor $f_{\mathrm{reg},i}(x) = F_i x + g_i$ and the polytopic region of validity $\mathcal{P}_i$. Then, the optimal refinement $\tilde{f}_{\mathrm{reg},i}(x) = \tilde{F}_i x + \tilde{g}$ of $f_{\mathrm{reg},i}(\cdot)$ such that $\tilde{f}_{\mathrm{reg},i}(x) \in \mathcal{U}$ for all $x \in \mathcal{P}_i$ is computed by solving the following quadratic program:

$$\min_{\tilde{F}_i,\tilde{g}_i} \sum_{v\in\mathcal{V}_i} (f_{\mathrm{reg},i}(v) - \tilde{f}_{\mathrm{reg},i}(v))^T (f_{\mathrm{reg},i}(v) - \tilde{f}_{\mathrm{reg},i}(v)) \tag{11a}$$

$$\text{s.t. } \tilde{f}_{\mathrm{reg},i}(v) \in \mathcal{U},\ \forall v \in \mathcal{V}_i, \tag{11b}$$

where $\mathcal{V}_i = \{v_{i,1},\dots,v_{i,n_v}\}$ are the vertices of $\mathcal{P}_i$. Here, (11a) optimizes parameters of $\tilde{f}_{\mathrm{reg},i}(\cdot)$ such that they are as close as possible to $f_{\mathrm{reg},i}(\cdot)$. Since both functions are assumed to be affine, minimizing the point-wise mismatch at the vertices of $\mathcal{P}_i$ is equivalent to minimizing the integrated squared error $\int \|f_{\mathrm{reg},i}(x) - \tilde{f}_{\mathrm{reg},i}(x)\|\, \mathrm{d}x$, evaluated over $\mathcal{P}_i$. Moreover, it is trivial to prove that if $\mathcal{U}$ is a convex set, $\mathcal{P}_i$ is a polytope, and $\tilde{f}_{\mathrm{reg},i}(\cdot)$ is an affine function, then $\tilde{f}_{\mathrm{reg},i}(v) \in \mathcal{U}$ for each vertex of $\mathcal{P}_i$ is necessary and sufficient for $\tilde{f}_{\mathrm{reg},i}(x) \in \mathcal{U}$ for all points $x \in \mathcal{P}_i$. Finally, note that with $\mathcal{U}$ a polyhedron, constraints in (11b) are linear in the decision variables $\tilde{F}_i$, $\tilde{g}_i$ that constitute $\tilde{f}_{\mathrm{reg},i}(x) = \tilde{F}_i x + \tilde{g}_i$ since the vertices are known. Moreover, the objective function (11a) is quadratic in $\tilde{F}_i$ and $\tilde{g}_i$ since $f_{\mathrm{reg},i}(x) = F_i x + g_i$ is known in each terminal node. Notice that (11) is feasible in each terminal node for an arbitrary non-empty polyhedron $\mathcal{U}$.

Therefore the refined feedback policy $u = \tilde{f}_{\mathrm{reg}}(x)$ which provides $u \in \mathcal{U}$ for all $x \in \mathcal{P}$ can be easily obtained by solving (11) in each terminal node of $f_{\mathrm{reg}}(\cdot)$ in (4). Notice that only parameters of the local regressors are modified, while the splits stay the same. Clearly, there is no guarantee that the refined local regressor $\tilde{f}_{\mathrm{reg},i}(x) = \tilde{F}_i x + \tilde{g}$ is optimal w.r.t. (5). Unfortunately, since the refinement in (11) depends on vertices of $\mathcal{P}_i$, which in turn depend on the split $\sigma(\cdot)$, it is not possible to include (11b) directly into (9) and still be able to solve the regression problem as a mixed-integer quadratic program. Also note that enforcing $f_{\mathrm{reg}}(x_j) \in \mathcal{U}$ for the training datapoints $x_1,\dots,x_p$ is merely necessary, but not sufficient, to guarantee that $f_{\mathrm{reg}}(x) \in \mathcal{U}$ for an arbitrary $x \in \mathcal{P}$.

7. PERFORMANCE EVALUATION

In this section we demonstrate performance of the proposed regression-based control policy on simulation scenario that involves control of a one-zone building whose model was extracted from the ISE toolbox (van Schijndel, 2005). The particular building model is represented by a linear time-invariant discrete-time system

$$\dot{\xi} = A\xi + Bu + Ed, \tag{12}$$

with

$$A = 10^{-3} \cdot \begin{bmatrix} -0.020 & 0 & 0 & 0.020 \\ 0 & -0.020 & 0.001 & 0.020 \\ 0 & 0.001 & -0.056 & 0 \\ 1.234 & 2.987 & 0 & -4.548 \end{bmatrix}$$

$$B = 10^{-3} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.003 \end{bmatrix}, \; E = 10^{-3} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.055 & 0 & 0 \\ 0.327 & 0.003 & 0.001 \end{bmatrix}.$$

The model contains 4 state variables: ξ_1 represents temperature of the floor, ξ_2 is the internal facade temperature, ξ_3 denotes the external facade temperature, and ξ_4 is the temperature inside the building. All state variables are expressed in °C. The model considers a single real-valued control input u, which represents the amount of heating/cooling injected to the zone, expressed in Watts. Finally, the model accounts for 3 disturbances: d_1 is the external air temperature (in °C), d_2 is the heat generated inside in the zone due to occupancy (in W), and d_3 is the solar radiation (in W). Historical data over the period of 31 days is shown in Fig. 1.

The control objective is to manipulate the real-valued heat flow u such that the internal temperature (represented by ξ_4) stays within a $\pm\zeta$ tolerance of a prescribed reference temperature T_{ref}, i.e.,

$$T_{\text{ref}} - \zeta \leq \xi_4 \leq T_{\text{ref}} + \zeta, \tag{13}$$

while satisfying constraints on the control input

$$\underline{u} \leq u \leq \overline{u}, \tag{14}$$

and minimizing the consumption of energy, expressed by $|u|$. Here, a positive u represents heating, while a negative u represents cooling.

We assume that the state vector $\xi(t)$ and the vector of disturbances $d(t)$ can be measured at each time instant t, but future evolution of disturbances is unknown. Therefore we assume a constant dynamics for disturbances in the prediction model. The MPC problem is posed as

$$\min_{u_0,\ldots,u_{N-1}} \sum_{k=0}^{N-1} (q_s s_k + |u_k|) \tag{15a}$$

$$\text{s.t. } \xi_{k+1} = \tilde{A}\xi_k + \tilde{B}u_k + \tilde{E}d_0, \tag{15b}$$

$$\underline{u} \leq u_k \leq \overline{u}, \tag{15c}$$

$$T_{\text{ref}} - \zeta - s_k \leq C\xi_k \leq T_{\text{ref}} + \zeta + s_k, \tag{15d}$$

$$s_k \geq 0, \tag{15e}$$

$$\xi_0 = \xi(t), \; d_0 = d(t), \tag{15f}$$

where s_k are slack variables that soften the thermal comfort restrictions in (13). To limit the magnitude of violations of the optimal thermal zone, the non-negative slacks are penalized by a large penalty q_s in (15a). Moreover, $\tilde{A}$, $\tilde{B}$, $\tilde{E}$ in (15b) are the state-update matrices of (12) discretized with sampling $\Delta = 900$ seconds, along with $C = [\,0\;0\;0\;1\,]$. The input constraints are $\underline{u} = -1000$ W, $\overline{u} = 2000$ W, and the width of the thermal comfort zone range is $\zeta = 0.5$°C. Note that (15) is a linear program with parameters $x = [\xi(t)^T, d(t)^T, T_{\text{ref}}] \in \mathbb{R}^8$.

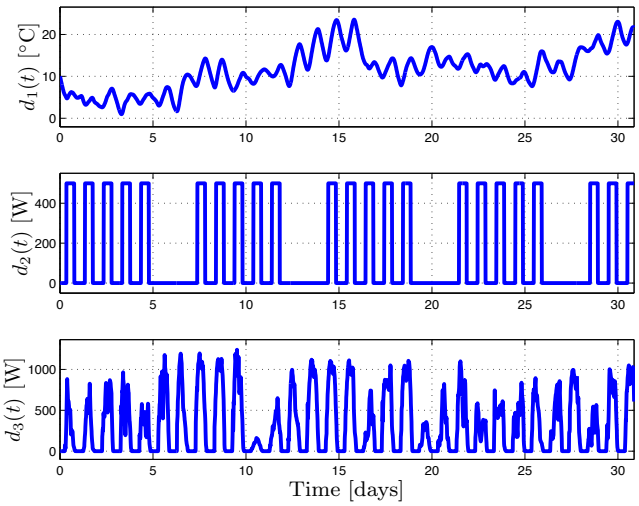

Fig. 1. Historical trends of the disturbance variables over 31 days: d_1 is the external temperature, d_2 stands for heat due to occupancy, and d_3 represents the solar radiation.

To construct the regression-based control policy $u = f_{\text{reg}}(x)$ as in (4), we have first performed a closed-loop simulation over the period of 5 days and collected the closed-loop training data $\{(x_1, u_1), \ldots, (x_p, u_p)\}$ with $p = 486$ (5 days sampled at 900 seconds). The loop was closed by the receding horizon feedback (3) where the optimal open-loop sequence was generated at each simulation step by solving the MPC problem (15) with prediction horizon $N = 48$ steps (12 hours sampled at 900 seconds), the reference temperature set constantly to $T_{\text{ref}} = 20$°C, and penalty on the slacks $q_s = 1 \cdot 10^6$. The same historical profiles of disturbances as in Fig. 1 were employed in the simulation. The initial state for the simulation was set to $\xi(0) = [20, 20, 20, 20]^T$. The training data generated by the MPC controller are shown in the top plot in Fig. 2. The same figure also depicts the performance of the MPC controller on the remaining 24 days of simulation. The accumulated heating/cooling cost under the optimal MPC feedback policy was 592 kWh. Note that the violations of the lower temperature range in the first 7 days are due to the control action being saturated at $\overline{u}$.

Then we have computed the regression-based feedback policy $u = f_{\text{reg}}(x)$ as in (4) by applying Algorithm 1 to the training data. In Step 1 of the algorithm, optimal splits and local regressors were calculated by solving the MIQP problem (9). The final tree consisted of 5 non-terminal, and 6 terminal nodes, along with 6 local affine regressors. Subsequently, the local regressors were refined per the procedure of Section 6 to guarantee that the regression-based feedback always provides satisfaction of input constraints.

To validate performance of the proposed regression-based feedback strategy and to evaluate decrease in performance with respect to MPC, we have performed a closed-loop simulation under the same conditions of the MPC scenario described above. The closed-loop profiles of the internal

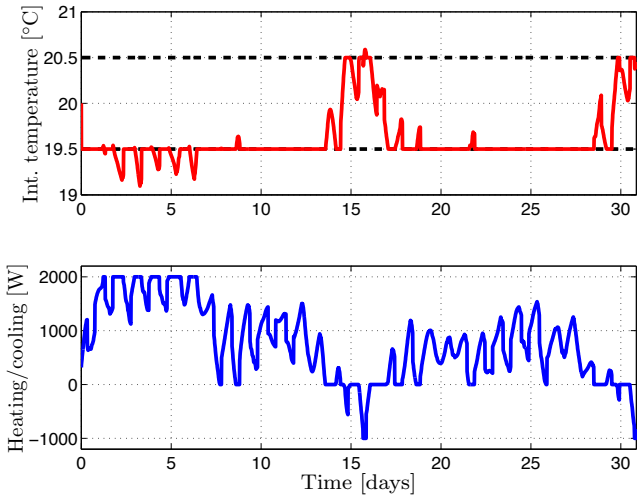

Fig. 2. Closed-loop profiles under the MPC feedback. The first 5 days were used as training data. The top figure shows the internal building temperature (solid red line), along with $\pm\zeta$ range of the reference temperature (black dashed lines). The bottom figure depicts the associated optimal control actions.

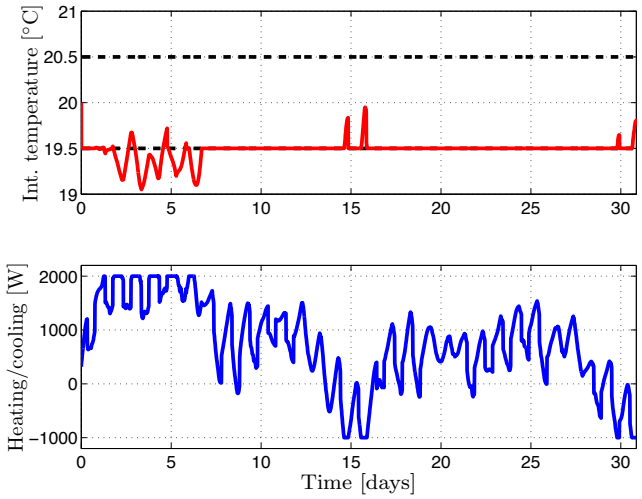

Fig. 3. Closed-loop profiles under the regression-based feedback policy in (4).

building temperature and of the control actions provided by $f_{\text{reg}}(\cdot)$ are shown in Fig. 3. As can be clearly seen, performance of $f_{\text{reg}}(\cdot)$ is close to the behavior of the MPC policy in Fig. 2. The only notable differences are between days 13 to 17, which correspond to hot days (cf. top part of Fig. 1). Here, the MPC policy is able to exploit the thermal zone to slightly reduce consumption of cooling energy by allowing the internal temperature to hit the upper limit of the thermal comfort zone. A similar phenomenon also occurs in the final 3 days of the simulation.

Besides these difference, the regression-based feedback matches the MPC controller appropriately. Although the regressor was only trained on the first 5 days of MPC profiles, the local affine regressors are able to reasonably extrapolate the control actions also in situations that were not present in the training data. The accumulated energy consumption under $f_{\text{reg}}(\cdot)$ was 611 kWh, an increase of mere 3% over the MPC strategy.

ACKNOWLEDGEMENTS

M. Klaučo, J. Drgoňa and M. Kvasnica gratefully acknowledge the contribution of the Scientific Grant Agency of the Slovak Republic under the grants 1/0095/11 and 1/0973/12. This research was supported by Mitsubishi Electric Research Laboratories, under a Collaborative Research Agreement.

REFERENCES

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

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E.N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1), 3–20.

Borrelli, F. (2003). *Constrained Optimal Control of Linear and Hybrid Systems*, volume 290 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag.

Breiman, L. (1993). *Classification and regression trees*. CRC press.

Cigler, J., Gyalistras, D., Širokỳ, J., Tiet, V., and Ferkl, L. (2013). Beyond Theory: the Challenge of Implementing Model Predictive Control in Buildings. In *Proceedings of 11th Rehva World Congress, Clima*.

Domahidi, A., Ullmann, F., Morari, M., and Jones, C. (2012). Learning near-optimal decision rules for energy efficient building control. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, 7571–7576.

Domahidi, A., Zeilinger, M., Morari, M., and Jones, C. (2011). Learning a feasible and stabilizing explicit model predictive control law by robust optimization. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, 513–519.

Fanger, P. (1970). Thermal comfort. analysis and applications in environmental engineering. *Thermal comfort. Analysis and applications in environmental engineering*.

Ma, Y., Vichik, S., and Borrelli, F. (2012). Fast stochastic MPC with optimal risk allocation applied to building control systems. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, 7559–7564.

Oldewurtel, F., Parisio, A., Jones, C., Gyalistras, D., Gwerder, M., Stauch, V., Lehmann, B., and Morari, M. (2012). Use of model predictive control and weather forecasts for energy efficient building climate control. *Energy and Buildings*, 45, 15–27.

Parry, M., Canziani, O., Palutikof, J., van der Linden, P., and Hanson, C. (2007). *Climate change 2007: impacts, adaptation and vulnerability*. Intergovernmental Panel on Climate Change.

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

van Schijndel, A. (2005). Integrated heat, air and moisture modeling and simulation in hamlab. In *IEA Annex 41 working meeting, Montreal, May*.