Fuzzy Model-Based Control: A Practical Approach

M. SETNES[1*] J.M. SOUSA[2]

[1]Delft University of Technology, Faculty of Information, Technology and Systems, Control Laboratory, P.O. Box 5031, 2600 GA Delft, The Netherlands
phone: +31 15 2783371, fax: +31 15 2786679
[2] Technical University of Lisbon, IST, Dep. of Mechanical Eng.
GCAR/IDMEC, Av. Rovisco Pais, 1, 1049-001 Lisbon, Portugal
phone: +351 1 8417471, fax: +351 1 8498097

Abstract: This paper addresses the main steps in model-based control: the identification of the model and the design of the controller. A series of recent research results is aggregated to present a complete approach. A fuzzy model of the system is identified from sampled data using supervised fuzzy clustering for rule extraction. This model is applied in model predictive control (MPC) of the process. The non-convex optimization problem introduced by a nonlinear plant model is solved by discretizing the control space and apply discrete search techniques. The trade-off between computational time and performance that follows from the discretization is addressed by using fuzzy rule-based optimization to adapt the discrete control actions. The global fuzzy model-based control approach is applied to pressure control of a fermentation tank.

Key-Words: fuzzy model-based control, fuzzy modeling, discrete model predictive control, discrete optimization.

1 Introduction

The classical control design of a model-based control system has several steps, starting with the modeling of the process under control, followed by the choice of the design specifications and their combination in performance criteria, and finally designing the controller. The choice of performance criteria and the design of the controller can be joined in several control schemes such as model predictive control (MPC). Thus, two main steps can be considered: modeling and design of the controller. For complex, nonlinear processes, the interest focus on data-driven modeling. In MPC, the non-linearity of the model and the presence of constraints lead to a non-convex optimization problem. There is thus a need for both accurate and compact (computationally fast) models as well as efficient optimization routines in the real-time application of MPC to general nonlinear processes.

This paper addresses these issues and presents a complete, data-driven approach based on fuzzy techniques, applicable to (small) real-world processes. Recent research by the authors is aggregated and presented in this paper to solve the various problems encountered in model-based control:

1. *Good performance in control of real-world processes* $\rightarrow$ apply model predictive control [1].
2. *Modeling of a nonlinear process* $\rightarrow$ use the Takagi-Sugeno (TS) type fuzzy model [2].
3. *Obtaining a compact TS model* $\rightarrow$ apply supervised fuzzy clustering to sampled data [3].
4. *Solve the non-convex optimization in MPC* $\rightarrow$ apply discrete MPC with branch-and-bound [4].
5. *Trade-off computational effort / performance in discrete MPC*$\rightarrow$ use adaptive control alternatives[5].

In the following, Section 2 describes fuzzy modeling based on supervised fuzzy clustering. The approach presented in [3] is extended to an adaptive distance clustering algorithm. A brief description of discrete MPC applied to nonlinear processes is presented in Section 3 together with the branch-and-bound (B&B) optimization. The computational cost / performance trade-off in discrete MPC is dealt with by using of a low number of *adaptive control alternatives*, and Section 4 describes adaptive control actions based on fuzzy rules. The presented model and control tools are applied to the pressure control of a fermentation tank in Section 5. Finally, Section 6 concludes the paper.

2 Fuzzy Modeling

A data-driven fuzzy modeling approach is described that provides the user with compact and accurate rule-based models. The approach can favorably be combined with the complexity reduction methods presented in [6].

*This work was supported by the the Research Council of Norway.

2.1 The Takagi-Sugeno fuzzy model

Takagi-Sugeno (TS) [2] fuzzy models have rules where the consequents are linear functions of the inputs:

$$R_i: \text{ IF } \boldsymbol{x} \text{ is } A_i \text{ THEN } c_i = \boldsymbol{a}_i^T \boldsymbol{x} + b_i\,,\ i = 1, \ldots, M. \quad (1)$$

Here $\boldsymbol{x} \in \mathbb{R}^n$ is the input vector and $c_i \in \mathbb{R}$ is the output (consequent). R_i denotes the ith rule whos antecedent fuzzy set A_i is defined by a multivariable membership function $\mu_{A_i}(\boldsymbol{x}) : \mathbb{R}^n \to [0, 1]$.

The total output of the model is computed by aggregating the individual rules contributions

$$y(k) = \sum_{i=1}^{M} u_{ki} c_i \quad (2)$$

where u_{ki} is the normalized degree of fulfillment of the antecedent clause of rule R_i for an input $\boldsymbol{x}_k$

$$u_{ki} = \frac{\mu_{A_i}(\boldsymbol{x}_k)}{\sum_{j=1}^{M} \mu_{A_j}(\boldsymbol{x}_k)} \quad (3)$$

2.2 Data driven identification

Given N input-output data pairs $\{\boldsymbol{x}_k, y_k\}$ where $\boldsymbol{x}_k$ is the n dimensional input vector $[x_{1k}, x_{2k}, \ldots, x_{nk}]^T$ and y_k is to be approximated by the model given $\boldsymbol{x}_k$. We can write (2) as a linear regression model

$$\boldsymbol{y} = \mathrm{U}\boldsymbol{C} + \boldsymbol{e} \quad (4)$$

where $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^T$ are the measured outputs, $\boldsymbol{C} = [c_1, c_2, \ldots, c_M]^T$ are the consequents of the M rules, and $\boldsymbol{e}$ contains the approximation errors. The matrix $\mathrm{U} = [\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_M}] \in \mathbb{R}^{\mathbf{N}\times\mathbf{M}}$, contains the degrees of fulfillment $\boldsymbol{u}_i = [u_{1i}, u_{2i}, \ldots, u_{Ni}]^T$ of all the M rules for the N inputs $\boldsymbol{x}_k$.

The identification of (1) is a two step approach. First the fuzzy antecedents A_i are determined by means of fuzzy clustering. Then, when the matrix U in (4) is known, the rule consequents $\boldsymbol{C}$ are determined by least-squares parameter estimation.

2.2.1 Identification by fuzzy clustering

To identify the model (1), the regression matrix $\mathrm{X}^{\mathbf{T}} = [\boldsymbol{x_1}, \ldots, \boldsymbol{x_N}]$ and an output vector $\boldsymbol{y}^T = [y_1, \ldots, y_N]$ are constructed from the available data. Fuzzy clustering is applied to the product-space of X and $\boldsymbol{y}$ to identify regions where the systems behaviour is approximated by local linear models of the form (1). Given the data $\mathrm{Z}^{\mathbf{T}} = [\mathrm{X},\ \boldsymbol{y}]$ and an estimated number of clusters M, we apply the Gustafson-Kessel (GK) [7] algorithm to compute the fuzzy partition matrix U whose ikth element $u_{ik} \in [0, 1]$ is the membership degree of the data object $\boldsymbol{z}_k \in \mathrm{Z}$, in cluster i. Thus, the rows of U contain point-wise represented multi-dimensional fuzzy sets A_i, that are defined by the cluster prototype $\boldsymbol{v}_i$ in the input space, and a function assigning a degree of membership of the input to the set depending on its distance to the prototype (see the appendix):

$$A_i(\boldsymbol{x}) = \mu(\boldsymbol{v}_i, \boldsymbol{x})\,. \quad (5)$$

The most important issue when applying clustering is the determination of the relevant number of clusters (rules). In [3] it was proposed to use an orthogonal-least squares rule reduction algorithm [8] to supervise the clustering process. Using the Gram-Schmidt orthogonalization, the partition matrix U, corresponding also to U in (4), is decomposed into $\mathrm{U} = \mathrm{QR}$, where $\mathrm{Q} \in \mathbb{R}^{\mathbf{N}\times\mathbf{N}}$ is a matrix with orthogonal columns $\boldsymbol{q}_j$, and $\mathrm{R} \in \mathbb{R}^{\mathbf{M}\times\mathbf{M}}$ is upper triangular with unity diagonal elements. The error reduction ratio due to $\boldsymbol{q}_j$ can be defined as [8]

$$[err]_j = (g_j^2 \boldsymbol{u}_j^T \boldsymbol{u}_j)/(\boldsymbol{y}^T \boldsymbol{y})\,,\ 1 \le j \le M\,, \quad (6)$$

where g_j is the $j$the entry of $\boldsymbol{g} = \mathrm{RC} \in \mathbb{R}^{\mathbf{M}}$. In [8] this ratio was used to seek a subset of important regressors in a forward-regression manner. For fuzzy modeling, this corresponds to a subset of important fuzzy rules, and in [3] a rule contribution factor was derived based on (6) and applied to clustering. When clustering is close to convergence, given the partition matrix U, the $j = 1, \ldots, M_S$ most important columns (clusters) of U are selected until the least important of the selected clusters has a relative contribution less than a factor ρ of the previously selected clusters:

$$\frac{\sum_{j=1}^{M_S}[err]_j - \sum_{j=1}^{M_S-1}[err]_j}{\sum_{j=1}^{M_S-1}[err]_j} < \rho\,,\ \rho \in [0, 1] \quad (7)$$

Only the M_S selected clusters are retained for further optimization by the cluster algorithm.

We combine this algorithm with the GK algorithm (see the appendix), that unlike the approach in [3], applies an adaptive distance measure to detect clusters of different shapes and orientation in the same data set.

2.2.2 Estimating the rule consequents

The parameters of the consequents $c_i = \boldsymbol{a}_i^T \boldsymbol{x} + b_i$ are obtained as a least-square estimate. Let $\mathrm{X_e}$ denote the matrix $[\mathrm{X};\ \mathbf{1}]$, $\Gamma_\mathbf{i}$ is a diagonal matrix in $\mathbb{R}^{N\times N}$ having the normalized degree of fulfillment u_{ki} as its kth diagonal element. Further, denote X' the matrix in $\mathbb{R}^{N\times MN}$ composed of matrices $\Gamma_\mathbf{i}$ and $\mathrm{X_e}$

$$\mathrm{X}' = [(\Gamma_\mathbf{1}\mathrm{X_e}); (\Gamma_\mathbf{2}\mathrm{X_e}); \ldots; (\Gamma_\mathbf{M}\mathrm{X_e})]\,. \quad (8)$$

Denote $\boldsymbol{\theta}'$ the vector in $\mathbb{R}^{M(n+1)}$ given by

$$\boldsymbol{\theta}' = \left[\boldsymbol{\theta}_1^T; \boldsymbol{\theta}_2^T; \ldots; \boldsymbol{\theta}_M^T\right]^T \quad (9)$$

where $\boldsymbol{\theta}_i^T = [\boldsymbol{a}_i^T; b_i]$ for $1 \le i \le M$. The resulting least-squares problem $\boldsymbol{y} = \mathrm{X}'\boldsymbol{\theta}' + \boldsymbol{e}$, corresponding to (4), has the solution

$$\boldsymbol{\theta}' = \left[(\mathrm{X}')^{\mathbf{T}}\mathrm{X}'\right]^{-1}(\mathrm{X}')^{\mathbf{T}}\boldsymbol{y}\,. \quad (10)$$

From eq. (9), the parameters $\boldsymbol{a}_i$ and b_i are obtained by:

$$\boldsymbol{a}_i = [\theta'_{q+1}, \theta'_{q+2}, \ldots, \theta'_{q+n}]^T, \quad b_i = [\theta_{q+n+1}] \,, \qquad (11)$$

where $q = (i-1)(n+1)$.

3 Model Predictive Control

Predictive control is a general methodology for solving control problems in the time domain having one common feature; the controller is based on the prediction of the future system behavior by using a process model. MPC is based on the following basic concepts: 1) Use of an available (nonlinear) model to predict the process outputs at future discrete time instants over a prediction horizon. 2) Computation of a sequence of future control actions using the model of the system by minimizing a certain objective function. 3) Receding horizon principle; at each sampling instant the optimization process is repeated with new measurements, and the first control action obtained is applied to the process. Because of the explicit use of a process model and the optimization approach, MPC can handle multivariable processes with nonlinearities, non-minimum phase behavior or long time delays, and can efficiently deal with constraints.

The future plant outputs for a determined *prediction horizon* H_p are predicted at each time instant k using a model of the process. The predicted output values $\hat{y}(k+i)$, $i = 1, \ldots, H_p$ depend on the states of process at the current time k (given, for instance, by the past input and outputs) and on the future control signals $u(k+j)$, $j = 1, \ldots, H_c$, where H_c is the *control horizon*. The control signals change only inside the control horizon, remaining constant afterwards, i.e., $u(k+j) = u(k+H_c-1)$, for $j = H_c, \ldots, H_p - 1$. The sequence of future control signals is obtained by optimizing an objective (cost) function which describes the control goals. The objective function is usually of the following quadratic form:

$$J(u) = \sum_{i=1}^{H_p} (r(k+i) - \hat{y}(k+i))^2 + \beta(\Delta u(k+i-1))^2, \quad (12)$$

or some small modifications of it, where $\hat{y}$ are the predicted process outputs, r is the reference trajectory, and Δu is the change in the control signal weighted by the parameter β. The first term of (12) accounts for the minimization of the output errors, the second term represents the minimization of the control effort, and β determine the weighting between the two in the global criterion [9]. The process inputs and outputs, as well as state variables, can be subjected to constraints, which are then incorporated in the optimization problem.

The performance of MPC depends largely on the used process model. The model must be able to accurately predict the future process outputs, and at the same time be computationally attractive to meet real-time demands. Using nonlinear fuzzy models, as described in Section 2, results in a non-convex optimization problem. In this case, both the *Sequential Quadratic Programming* (SQP) method [10] and the *simplex method* [11], which are both iterative optimization techniques, can be considered. However, these methods have generally high computational costs and often converge to local minima. By discretizing the control actions, the efficient branch-and-bound algorithm can be used to search the discrete space for the best solution. This has proven to give better results than iterative optimization techniques [4].

Branch-and-Bound Optimization

In discrete MPC, B&B can be used for the optimization problem that must be solved at each time instant k. A B&B algorithm can be characterized by the following two rules: i) *Branching rule* - defines how to divide a problem into sub-problems; ii) *Bounding rule* - establishes lower and upper bounds in the optimal solution of a sub-problem, where these bounds allow for the elimination of sub-problems that do not constitute an optimal solution. Usually, these two basic rules are applied recursively in B&B methods.

The B&B method can only be applied to predictive control when the control actions are discretized. The model of the system under control predict the future outputs of the system $\hat{y}(k+1), \ldots, \hat{y}(k+H_p)$, and is given by:

$$\hat{y}(k+i) = f(\boldsymbol{x}(k+i-1), u(k+i-1)), \quad i = 1, \ldots, H_p \,. \qquad (13)$$

Let the possible inputs of the system be discretized in K possible control actions. Let also the discretized control actions be denoted ω_j. Thus, at each step the control actions $u(k+i-1) \in \Omega$, are given by

$$\Omega = \{\omega_j | j = 1, 2, \ldots, K\}. \qquad (14)$$

In predictive control, the problem to be solved is normally represented by an objective function minimizing the predicted error and the control effort. This optimization problem is successively decomposed by the branching rule into smaller sub-problems. At time instant $k+i$ the cumulative cost of a certain path followed so far, and leading to the state $\boldsymbol{x}(k+i)$ and output $\hat{y}(k+i)$ is given by

$$J^{(i)} = \sum_{\ell=1}^{i} \left[(r(k+\ell) - \hat{y}(k+\ell))^2 + \beta(\Delta u(k+\ell-1))^2 \right], \qquad (15)$$

where $i = 1, \ldots, H_p$, denotes the level corresponding to the time step $k+i$. A particular branch j at level i is created when the cumulative cost $J^{(i)}$ plus a *lower bound* on the cost from the level i to the terminal level H_p for the branch j, denoted J_{L_j}, is lower than an *upper bound* of the total cost, denoted J_U:

$$J^{(i)} + J_{L_j} < J_U. \qquad (16)$$

Let the total number of branches verifying this rule at level i be given by N. In order to increase the efficiency of the

B&B method, is required that this number should be as low as possible, i.e. $N \ll K$. Note that no branching takes place for $i > H_c - 1$ (beyond the control horizon), i.e., the last control action $u(k + H_c - 1)$ is applied successively to the model, until H_p is reached. In order to achieve $N \ll K$, the upper bound should be as low as possible (close to the optimal solution) and the lower bound as large as possible, in order to decrease the number of new branches N.

The B&B algorithm applied to MPC always finds the global discrete optimal solution, guaranteeing a good control performance when the number of control actions K is sufficiently large. Moreover, B&B does not need any initial guess, and hence its performance cannot be negatively influenced by a poor initialization, as in the case of iterative optimization methods, and the B&B method implicitly deals with constraints. In fact, the presence of constraints improve the efficiency of bounding, restricting the search space by eliminating non-feasible sub-problems.

Two serious drawbacks of B&B are the exponential increase of the computational time with the control horizon and the number of alternatives, and the discretization of the possible control actions. This discretization can cause chattering, overshoots and slow step-responses. A solution to this problem is to adapt the control alternatives to better suit the present situation in the system [5].

4 Adaptive control actions

In order to solve the drawbacks of the B&B optimization described in the previous section, adaptive set of discrete inputs can be used. Let $u(k) \in U$ represent the control action at time instance k, where $U = [U^-, U^+]$ is the domain of the manipulated variable. Let also the upper and lower bounds of the possible change in the control signal at time k, u_k^+ and u_k^- respectively, be given by

$$u_k^+ = U^+ - u(k-1)\,, u_k^- = U^- - u(k-1)\,.$$

The values u_k^+ and u_k^- are thus the maximum changes allowed for the control action, when this is being increased or decreased, respectively. The control action at time $k + i$, $i = 0, \ldots, H_c - 1$ is given by

$$u(k+i) = u(k+i-1) + \omega_j(k)\,, \qquad (17)$$

where $\omega_j(k) \in \Omega_k$ is the change in the control action selected by the B&B optimization algorithm from the set of alternatives Ω_k. An adaptive set Ω_k is created by using a scaling factor $\gamma(k) \in [0, 1]$, which depends on the activation of simple fuzzy rules, applied to a dynamic set of actions as opposed to the static set Ω in (14):

$$\Omega_k = \gamma(k) \cdot \{\lambda_l\, u_k^+,\, 0\, \lambda_l\, u_k^- \,|l = 1, 2, \ldots, N\}\,, \qquad (18)$$

where the values λ_l can be chosen such that the control alternatives are linear or logarithmically distributed. In the application in Section 5, we simply use $\lambda_l = 1/(4^l)$, with

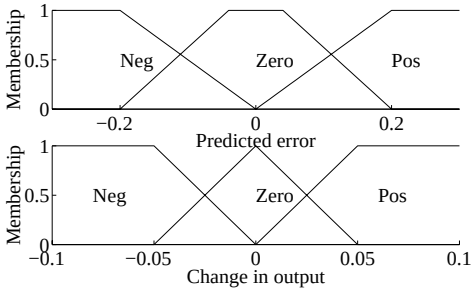

Figure 1: Fuzzy sets used in rule base.

$N = 1$, giving a minimum set of possible actions

$$\Omega_k = \gamma(k) \cdot \{u_k^+/4,\, 0,\, u_k^-/4\}\,. \qquad (19)$$

The scaling factor $\gamma(k)$ is determined at each instant k by simple fuzzy rules concerning the predicted error $\hat{e}(k + H_p)$ and the change in the output $\Delta y(k) = y(k) - y(k-1)$. The rules are derived from simple heuristics. For instance, when $\Delta y(k)$ is small, and $\hat{e}(k + H_p)$ is also small, the system is close to a steady-state situation. The set of control alternatives should then be scaled down to allow finer control actions, i.e. $\gamma(k) \to 0$, in order to approach zero steady-state error without introducing oscillations around the set-point. When the predicted error is high, bigger corrective steps should be taken, i.e. $\gamma(k) \to 1$.

The exact definition of the rules is not critical, and the domains of the two variables $\hat{e}(k + H_p)/y(k)$ (relative predicted error) and $\Delta y(k)/y(k)$ (relative change in output) have both been partitioned by three fuzzy sets representing *Negative*, *Zero*, and *Positive*, as shown in Fig. 1. By combination, this gives a simple fuzzy rule base of nine rules, determining the scaling factor $\gamma(k)$, as given in Table 1. The

Table 1: Fuzzy scaling rules for $\gamma(k)$

$\hat{e}(k+H_p)/y(k)$	$\Delta y(k)/y(k)$	$\gamma(k)$
Neg	*Neg*	1.0
Neg	*Zero*	1.0
Neg	*Pos*	0.2
Zero	*Neg*	0.7
Zero	*Zero*	0
Zero	*Pos*	0.7
Pos	*Neg*	0.2
Pos	*Zero*	1.0
Pos	*Pos*	1.0

rule base is a zero order Takagi-Sugeno model [2], and the resulting control surface for the scaling factor $\gamma(k)$ is shown in Fig. 2.

To summarize, equation (18) represents a dynamic set of alternative changes of the control action that can be applied at time instance k. The control alternatives are determined by the available control space at this time, as defined in (17).

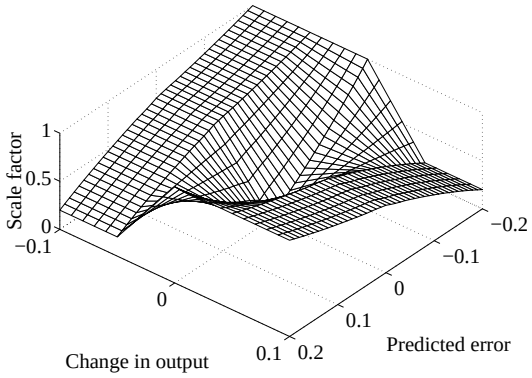

Figure 2: Value of $\gamma(k)$ as function of predicted error $\hat{e}(k + H_p)$ and change in output $\Delta y(k)$.

Further, the alternatives are scaled by the factor $\gamma(k)$, which is determined by the fuzzy rule base in Table 1 considering the state of the system at time k. Finally, constraints on the control signal concerning, e.g., the rate of change can be directly implemented when selecting the parameters λ_i.

5 Application example

The described modeling and control scheme is applied in simulation to pressure control in the fermenter shown in Fig. 3. At the bottom of the fermenter tank, air is fed into the water at a specified flow-rate, which is kept constant by the controller u_2. In the presented experiments, the air pressure y in the tank is controlled by the outlet valve u_1 at the top of the tank. Because of the underlying physical mechanisms, and the nonlinear characteristics of the control valve, the process has nonlinear steady-state characteristics as well as nonlinear dynamics.

By the fuzzy modeling approach described in Section 2, a MISO fuzzy model of the pressure dynamics is identified from systems measurements:

$$y(k+1) = f(y(k), u_1(k)) \ , \tag{20}$$

where $f(\cdot)$ is a fuzzy model of the TS type constituted by five rules with a two-dimensional premise:

$$R_i: \text{ IF } \boldsymbol{x}(k) \text{ is } A_i \text{ THEN } y(k+1) = \boldsymbol{a}_i^T \boldsymbol{x}(k) + b_i \ , \tag{21}$$

where $i = 1, \ldots, 5$ and $\boldsymbol{x}(k) = [y(k), u_1(k)]$. The model was obtained by using $m = 2$ and a rule contribution threshold of 10% ($\rho = .1$) in the cluster algorithm (see appendix). A validation of the model in a recursive simulation (free-run) is shown in Fig. 3. A description of the used simulation model can be found in [12, 13] where this system was also studied.

The performance of the discrete MPC controller with B&B optimization as described in Section 3 is shown in Fig. 4. The set of three adaptive decision alternatives with fuzzy scaling as given in (19) was used. From the results it is seen that the controller acts very fast to changes in the

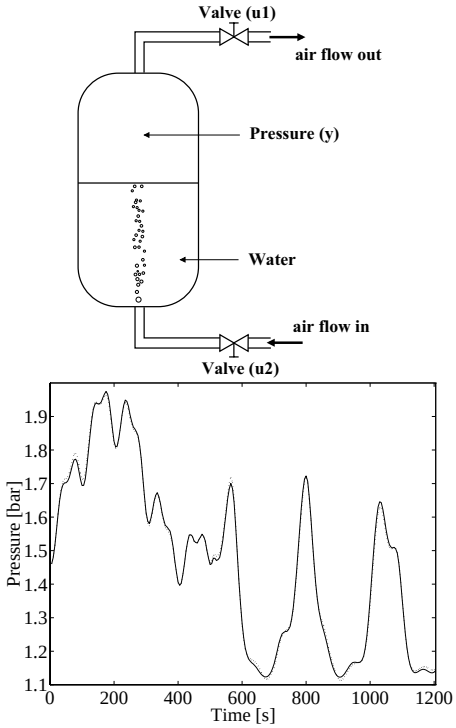

Figure 3: *Top:* Laboratory fermenter. *Bottom:* Evaluation of the model in a free-run.

reference, but at the same time also quickly settles at all the correct steady-state values, even though only three control alternatives are available at each sampling instance. This effect is due to the adaptive, scaled set of alternatives.

6 Conclusions

This paper addresses the main steps in model-based control: the identification of the model and the design of the controller. It presents a complete, data-driven approach based on fuzzy techniques, which are applicable to (small) real-world processes. Recent research by the authors was aggregated and presented in this paper to solve the main problems encountered in model-based control. First, the need for good control performance was solved by applying model predictive control, which needs an accurate and computationally fast model, especially for nonlinear processes. Fuzzy models of the Takagi-Sugeno type present both properties for a large number of processes. When the model is too complex for control purposes, it can be simplified by applying supervised fuzzy clustering to sampled data, as in this paper. The application of nonlinear models to MPC results usually in a non-convex optimization problem. This problem is solved by applying branch-and-bound; a discrete optimization algorithm. The application of B&B, however, introduces a trade-off between computational time and performance in discrete MPC. This problem was solved by using adaptive control alternatives. The global fuzzy model-based control approach is applied to pressure control of a

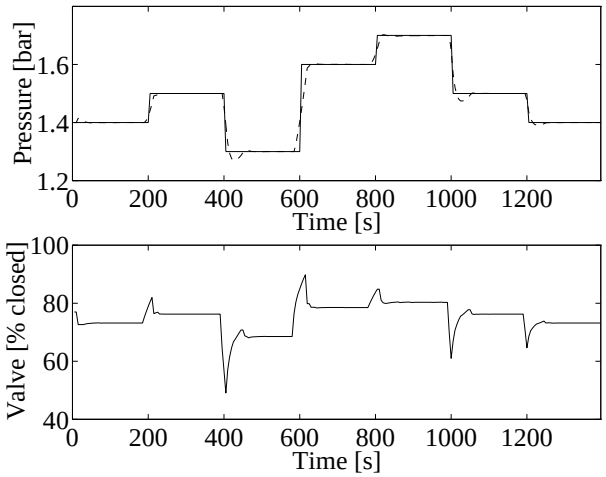

Figure 4: *Top:* Time responses for a given reference (solind line). *Bottom:* The corresponding control effort ($H_p = H_c = 4$).

fermentation tank. The obtained performance revealed to be considerably better than other model-based control techniques used previously.

Appendix: Cluster algorithm

Given the data Z, a too high number of clusters $1 < M < N$, the fuzziness $m > 1$, the rule contribution threshold ρ, and the termination tolerance $\epsilon > 0$. Initialize $\mathrm{U}^{(\mathbf{0})}$ randomly.

Repeat for $l = 1, 2, \ldots$
Step 1: **Compute cluster prototypes:**

$$\boldsymbol{v}_i^{(l)} = \frac{\sum_{k=1}^{N} (u_{ki}^{(l-1)})^m \boldsymbol{z}_k}{\sum_{k=1}^{N} (u_{ki}^{(l-1)})^m}\,,\ 1 \le i \le M\,.$$

Step 2: **Compute covariance matrices:**

$$\mathrm{F_i} = \frac{\sum_{\mathbf{k=1}}^{\mathbf{N}} (\mathbf{u}_{\mathbf{ki}}^{(\mathbf{l-1})})^{\mathbf{m}} (\boldsymbol{z}_{\mathbf{k}} - \boldsymbol{v}_{\mathbf{i}}^{(\mathbf{l})})(\boldsymbol{z}_{\mathbf{k}} - \boldsymbol{v}_{\mathbf{i}}^{(\mathbf{l})})^{\mathbf{T}}}{\sum_{\mathbf{k=1}}^{\mathbf{N}} (\mathbf{u}_{\mathbf{ki}}^{(\mathbf{l-1})})^{\mathbf{m}}}\,,$$

for $1 \le i \le M$.

Step 3: **Compute distances to cluster prototypes:**

$$d_{ki} = (\boldsymbol{z}_k - \boldsymbol{v}_i)^T \left[(\det(\mathrm{F_i})^{\mathbf{1/(n+1)}} \mathrm{F_i^{-1}} \right] (\boldsymbol{z}_k - \boldsymbol{v}_i)\,,$$

for $1 \le i \le M,\ 1 \le k \le N$.

Step 4: **Update the partition matrix:**
for $1 \le i \le M,\ 1 \le k \le N$,
if $d_{ki} > 0$

$$u_{ki}^{(l)} = \frac{1}{\sum_{j=1}^{M} (d_{ki}/d_{kj})^{2/(m-1)}},$$

else if $d_{ki} = 0$

$$u_{ki}^{(l)} = 1.$$

Step 5: **Run OLS reduction algorithm:**
if $\|\mathrm{U}^{(\mathbf{l})} - \mathrm{U}^{(\mathbf{l-1})}\| < \mathbf{2}\epsilon$ run OLS algorithm and keep only the the selected M_S clusters

$$\mathrm{U}^{(\mathbf{l})} := \mathrm{U}^{(\mathbf{M_s})}$$
$$M := M_s$$

until $\|\mathrm{U}^{(\mathbf{l})} - \mathrm{U}^{(\mathbf{l-1})}\| < \epsilon$.

References

[1] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29:1251–1274, 1993.

[2] T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, 15(1):116–132, 1985.

[3] M. Setnes. Supervised fuzzy clustering for rule extraction. In *Proceedings of FUZZ-IEEE'99*, Seoul, Korea, August 1999.

[4] J.M. Sousa, R. Babuška, and H.B. Verbruggen. Fuzzy predictive control applied to an air-conditioning system. *Control Engineering Practice*, 5(10):1395–1406, 1997.

[5] M. Setnes and J.M. Sousa. Fuzzy rule-based optimization in nonlinear predictive control. In *Proceedings of ECC'99*, Karlsruhe, Germany, Aug.-Sept. 1999.

[6] M. Setnes, R. Babuška, and H. B. Verbruggen. Complexity reduction in fuzzy modeling. *Mathematics And Computers In Simulation*, 46(5–6):507–516, 1998.

[7] D. E. Gustafson and W. C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Proceedings IEEE CDC*, pages 761–766, San Diego, USA, 1979.

[8] L.X. Wang and J.M. Mendel. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. *IEEE Transactions on Neural Networks*, 3(5):807–813, 1992.

[9] R. Soeterboek. *Predictive control: a unified approach*. Prentice Hall, New York, USA, 1992.

[10] P. E. Gill, W. Murray, and M.H. Wright. *Practical Optimization*. Academic Press, New York and London, 1981.

[11] J.A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[12] H.J.L van Can, H.A.B te Braake, C. Hellinga, K.Ch.A.M. Luyben, and J.J. Heijnen. A strategy for dynamic process modeling based on neural networks in macroscopic balances. *AIChE Journal*, 42:3403–3418, 1996.

[13] C. Onnen, R. Babuška, U.Kaymak, J.M. Sousa, H.B. Verbruggen, and R. Isermann. Genetic algorithms for optimization in predictive control. *Control Engineering Practice*, 5(10):1363–1372, 1997.