Effective Optimization for Fuzzy Model Predictive Control

Stanimir Mollov, Robert Babuška, Janos Abonyi, and Henk B. Verbruggen

***Abstract*—This paper addresses the optimization in fuzzy model predictive control. When the prediction model is a nonlinear fuzzy model, nonconvex, time-consuming optimization is necessary, with no guarantee of finding an optimal solution. A possible way around this problem is to linearize the fuzzy model at the current operating point and use linear predictive control (i.e., quadratic programming). For long-range predictive control, however, the influence of the linearization error may significantly deteriorate the performance. In our approach, this is remedied by linearizing the fuzzy model along the predicted input and output trajectories. One can further improve the model prediction by iteratively applying the optimized control sequence to the fuzzy model and linearizing along the so obtained simulated trajectories. Four different methods for the construction of the optimization problem are proposed, making difference between the cases when a single linear model or a set of linear models are used. By choosing an appropriate method, the user can achieve a desired tradeoff between the control performance and the computational load. The proposed techniques have been tested and evaluated using two simulated industrial benchmarks: pH control in a continuous stirred tank reactor and a high-purity distillation column.**

***Index Terms*—Linearization, model predictive control, multiple-input–multiple-output systems (MIMO), nonlinear control, quadratic programming, Takagi–Sugeno fuzzy models.**

I. Introduction

In MODEL predictive control (MPC), the control action is obtained by solving at each sampling instant an optimization problem in order to minimize the tracking error (and possibly also the control effort). Nonlinear MPC must be applied in situations where the controlled process is inherently nonlinear, or where large changes in the operating conditions can be anticipated during routine operation, such as in batch processes, or during the start-up and shutdown of continuous processes. The use of nonlinear models in MPC is motivated by the possibility to improve the control performance by improving the prediction accuracy. Fuzzy models of the Takagi–Sugeno (TS) type [1] proved to be suitable for the use in nonlinear MPC because of their ability to accurately approximate complex nonlinear systems by using data combined with prior knowledge [2]–[4]. Many successful applications of MPC using fuzzy models have been reported [5]–[10] and recently several papers appeared in which different fuzzy MPC algorithms are analyzed and compared; see [11]–[13]. The methods discussed in these papers can generally be classified into two groups: 1) methods using directly the fuzzy model in the optimization procedure [7]–[9], [12], [13], and 2) methods using a linearized model instead of the fuzzy one [5], [6], [10], [12]. For example, in [6], a linear step response model is extracted from the fuzzy model. The authors of [10] applied Jacobian linearization. Other possibilities are to compute the control signals for the different fuzzy rules separately and to weigh them, or to use only the rule with the largest membership degree [5], [12].

The main problem in the real-time application of fuzzy model predictive control is that a nonconvex optimization problem must be solved at each sampling period. This hampers the application to fast processes where iterative optimization techniques cannot be properly used due to short sampling times. In this paper, methods are presented that avoid nonconvex optimization by employing a single local linear model or a set of local linear models to approximate the fuzzy model along the predicted trajectory. The control signal is obtained by solving a constrained quadratic program (QP). To account for errors introduced by the linearization, an iterative optimization scheme is proposed. In such a setting, the QP solution provides a search direction toward the minimum of the optimization problem. Convergence is guaranteed through a line search mechanism that considers reduction both in the cost function and in the constraints. The proposed method belongs to the general class of sequential quadratic programming methods [14], [15]. However, a specific feature of our approach is the way we define (and update) the Hessian and the gradient, taking advantage of the linear time-varying (LTV) interpretation of the TS fuzzy model.

Recently, it was proven that even a suboptimal MPC solution is sufficient to guarantee stability [16]. Although we briefly discuss stability in this paper, it has not been the aim of this research (for details on stability of fuzzy predictive control, see [17]). The main topic of this article is to study the structure of the underlying optimization problem and to provide effective ways to obtain suboptimal solutions, as close as possible to the optimum in a limited amount of time. This makes nonlinear MPC suitable also for fast processes. Another contribution is a generalized formulation of the cost function used in the optimization problem. It penalizes not only the deviation of the model output from the output reference and the change in the input signal, but

Manuscript received June 28, 2001; revised August 2, 2002 and November 20, 2003. This work was supported in part by the FAMIMO Esprit project LTR 219 11 (for more information see http://iridia.ulb.ac.be/~famino). The work of J. Abonyi was supported by the Hungarian Ministry of Education under Grants FKFP-0073/2001 and 0063/2000, and by the Hungarian Research Fund under Grant OTKA T037600.

S. Mollov is with FCS Control Systems B.V., 1117 ZJ Schiphol, The Netherlands (e-mail: stanimir.mollov@fcs-cs.com).

R. Babuška and H. B. Verbruggen are with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: r.babuska@dcsc.tudelft.nl).

J. Abonyi is with the Department of Process Engineering, the University of Veszprém, Veszprém H-8201, Hungary (e-mail: abonyij@fmt.vein.hu).

Digital Object Identifier 10.1109/TFUZZ.2004.834812

also the variation in the predicted output and the deviation of the control input from an input reference.

The paper is organized as follows. Section II introduces the necessary background material, namely: 1) the multivariable Takagi–Sugeno (TS) fuzzy model, 2) the extraction of a linear state-space model in the controllable canonical form from the TS model, 3) the formulation of the optimization problem in the MPC based on LTV state-space models, and 4) the internal model control scheme. The methods for obtaining local linear models from the fuzzy model are introduced in Section III. To ensure convergence of the iterative optimization methods, a line search mechanism is given in Section IV. Sections V and VI present two simulation benchmarks: pH control in a continuous stirred tank reactor and purity control in a binary distillation column. Section VII concludes this paper. Detailed derivations of the Hessian and gradient of the Lagrange function based on linear time-varying models are given in the Appendix.

II. Fuzzy Model Predictive Control

A. Multivariable TS Fuzzy Model

Consider a MIMO system with m inputs: $\mathbf{u} \in U \subset \mathbb{R}^m$ and p outputs: $\mathbf{y} \in Y \subset \mathbb{R}^p$. This system is approximated by a collection of coupled multiple-input–single-output discrete-time fuzzy models of the input–output NARX type

$$\mathbf{y}_l(k+1) = \mathcal{R}_l\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right), \qquad l = 1, 2, \ldots, p.$$

The input vector $\mathbf{u}(k) \in \mathbb{R}^m$ contains the current inputs, and the regression vector $\boldsymbol{\xi}_l(k) \in \mathbb{R}^\varrho$ includes current and lagged outputs and inputs:

$$\boldsymbol{\xi}_l(k) = [\mathbf{y}_1(k), \ldots, \mathbf{y}_p(k), \mathbf{u}_1(k-1), \ldots, \mathbf{u}_m(k-1)]^T \quad (1)$$

with

$$\begin{aligned} \mathbf{y}_i(k) &= [y_i(k), y_i(k-1), \ldots, y_i(k-\mathbf{n}_{y,i})] \\ \mathbf{u}_j(k-1) &= [u_j(k-1), u_j(k-2), \ldots, u_j(k-\mathbf{n}_{u,j})], \\ i &= 1, \ldots, p, \qquad j = 1, \ldots, m \end{aligned}$$

where $n_{y,i}$ and $n_{u,j}$ specify the number of lagged values for the ith output and the jth input, respectively. $\mathcal{R}_l$ are rule-based fuzzy models of the TS type [1]

$$\begin{aligned} \mathcal{R}_{li}: &\text{If } \xi_{l1}(k) \text{ is } \Omega_{li,1} \text{ and } \ldots \text{ and } \xi_{l\varrho}(k) \text{ is } \Omega_{li,\varrho} \text{ and} \\ &u_1(k) \text{ is } \Omega_{li,\varrho+1} \text{ and } \ldots \text{ and } u_m(k) \text{ is } \Omega_{li,\varrho+m} \\ &\text{then } y_{li}(k+1) = \boldsymbol{\zeta}_{li}\boldsymbol{\xi}_l(k) + \boldsymbol{\eta}_{li}\mathbf{u}(k) + \theta_{li}, \\ &i = 1, 2, \ldots, K_l. \end{aligned} \quad (2)$$

Here, Ω_{li} are the antecedent fuzzy sets of the ith rule, $\boldsymbol{\zeta}_{li}$ and $\boldsymbol{\eta}_{li}$ are vectors containing the consequent parameters, and θ_{li} is the offset. K_l is the number of rules for the lth output. The model output is computed as the weighted average of the linear consequents in the individual rules

$$\mathbf{y}_l(k+1) = \frac{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\zeta}_{li}\boldsymbol{\xi}(k) + \boldsymbol{\eta}_{li}\mathbf{u}(k) + \theta_{li}\right)}{\sum_{i=1}^{K_l} \beta_{li}} \quad (3)$$

where the degree of fulfillment for the ith rule β_{li} is the product of the membership degrees of the antecedent variables (states and inputs) in that rule

$$\beta_{li}(\boldsymbol{\xi}_l, \mathbf{u}) = \prod_{h=1}^{\varrho} \mu_{\Omega_{li,h}}(\xi_{lh}) \prod_{j=1}^{m} \mu_{\Omega_{li,j}}(u_{lj}). \quad (4)$$

B. Fuzzy Model Linearization for Predictive Control

The TS model (2) can be locally represented by a state-space LTV model in the controllable canonical form

$$\begin{aligned} \mathbf{x}_{\text{lin}}(k+1) &= \mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) + \mathbf{B}(k)\mathbf{u}(k) \\ \mathbf{y}_{\text{lin}}(k) &= \mathbf{C}(k)\mathbf{x}_{\text{lin}}(k). \end{aligned} \quad (5)$$

To arrive at this representation, we express the model output (3) in the from

$$\mathbf{y}_l(k+1) = \boldsymbol{\zeta}_l^*\boldsymbol{\xi}_l(k) + \boldsymbol{\eta}_l^*\mathbf{u}(k) + \theta_l^*, \qquad i = 1, \ldots, p. \quad (6)$$

Comparing this expression with the LTV description (5), one can see that the state $\mathbf{x}_{\text{lin}}(k)$ contains the regression vectors ξ_l for the individual outputs. The elements of $\mathbf{x}_{\text{lin}}(k)$ are ordered such that the lagged outputs from all ξ_l, $i = 1, \ldots, p$ come first and then come the lagged inputs. We construct the matrices $\mathbf{A}(k)$, $\mathbf{B}(k)$ and $\mathbf{C}(k)$ by freezing the parameters of the fuzzy model at a certain operating point $\mathbf{y}(k)$ and $\mathbf{u}(k)$ as follows. First, calculate the degrees of fulfillment $\beta_{li}(\xi_l(k), \mathbf{u}(k))$ according to (4) and compute the aggregated parameters $\boldsymbol{\zeta}_l^*$, $\boldsymbol{\eta}_l^*$, and $\boldsymbol{\theta}_l^*$

$$\begin{aligned} \boldsymbol{\zeta}_l^* &= \frac{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right) \cdot \boldsymbol{\zeta}_{li}}{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right)} \\ \boldsymbol{\eta}_l^* &= \frac{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right) \cdot \boldsymbol{\eta}_{li}}{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right)} \\ \boldsymbol{\theta}_l^* &= \frac{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right) \cdot \theta_{li}}{\sum_{i=1}^{K_l} \beta_{li}\left(\boldsymbol{\xi}_l(k), \mathbf{u}(k)\right)}. \end{aligned} \quad (7)$$

The state matrix $\mathbf{A}$ contains the parameters $\boldsymbol{\zeta}_l^*$ and $\boldsymbol{\theta}_l^*$ and the input matrix $\mathbf{B}$ the parameters $\boldsymbol{\eta}_l^*$ in the order reflecting the structure of the state vector $\mathbf{x}_{\text{lin}}$. Additional columns are inserted in $\mathbf{A}$ in order to account for the lagged outputs and inputs. The last column of $\mathbf{A}$ contains the offset for the corresponding output θ_l^*

$$\mathbf{A} = \begin{bmatrix} \zeta_{1,1}^* & \zeta_{1,2}^* & \cdots & \cdots & \cdots & \zeta_{1,\varrho}^* & \theta_1^* \\ 1 & 0 & \cdots & & & 0 & 0 \\ 0 & 1 & \vdots & & & 0 & 0 \\ \vdots & \vdots & \ddots & & & \vdots & \vdots \\ \zeta_{2,1}^* & \zeta_{2,2}^* & \cdots & \cdots & \cdots & \zeta_{2,\varrho}^* & \theta_2^* \\ 0 & \vdots & \ddots & & & \vdots & \vdots \\ \zeta_{p,1}^* & \zeta_{p,2}^* & \cdots & \cdots & \cdots & \zeta_{p,\varrho}^* & \theta_p^* \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \quad (8a)$$

$$\mathbf{B}=\begin{bmatrix}\eta^*_{1,1} & \eta^*_{1,2} & \cdots & \eta^*_{1,m}\\ 0 & \cdots & \cdots & 0\\ \vdots & & & \vdots\\ \eta^*_{2,1} & \eta^*_{2,2} & \cdots & \eta^*_{2,m}\\ 0 & \cdots & \cdots & 0\\ \vdots & \ddots & \ddots & \vdots\\ \eta^*_{p,1} & \eta^*_{p,2} & \cdots & \eta^*_{p,m}\\ 0 & \cdots & \cdots & 0\\ 1 & \cdots & \cdots & 0\\ 0 & \cdots & \cdots & 0\end{bmatrix} \tag{8b}$$

$$\mathbf{C}=\begin{bmatrix}1 & 0 & \cdots & \cdots & \cdots & \cdots & 0\\ \vdots & & \ddots & & \ddots & & \vdots\\ 0 & \cdots & \cdots & 1 & 0 & \cdots & 0\end{bmatrix}. \tag{8c}$$

Example: A 2×2 fuzzy model is given as an example to illustrate this procedure. The considered rules for the two outputs are

$$\begin{aligned}R_{1,i}:&\text{If } y_1(k) \text{ is } \Omega_{1i,1} \text{ and } y_1(k-2) \text{ is } \Omega_{1i,2} \text{ and } \ldots\\ &\quad u_1(k) \text{ is } \Omega_{1i,3} \text{ and } u_1(k-1) \text{ is } \Omega_{1i,4} \text{ and } \ldots\\ &\quad u_2(k) \text{ is } \Omega_{1i,5} \text{ and } u_2(k-1) \text{ is } \Omega_{1i,6} \text{ then}\\ &y_1(k+1)=\zeta_{1i,1}y_1(k)+\zeta_{1i,2}y_1(k-2)\\ &\quad+\zeta_{1i,3}u_1(k-1)+\zeta_{1i,4}u_2(k-1)+\eta_{1i,1}u_1(k)\\ &\quad+\eta_{1i,2}u_2(k)+\theta_{1i}\\ R_{2,i}:&\text{If } y_2(k) \text{ is } \Omega_{2i,1} \text{ and } y_2(k-2) \text{ is } \Omega_{2i,2} \text{ and } \ldots\\ &\quad u_1(k) \text{ is } \Omega_{2i,3} \text{ and } u_1(k-1) \text{ is } \Omega_{2i,4} \text{ and } \ldots\\ &\quad u_2(k) \text{ is } \Omega_{2i,5} \text{ and } u_2(k-1) \text{ is } \Omega_{2i,6} \text{ then}\\ &y_2(k+1)=\zeta_{2i,1}y_2(k)+\zeta_{2i,2}y_2(k-2)\\ &\quad+\zeta_{2i,3}u_1(k-1)+\zeta_{2i,4}u_2(k-1)+\eta_{2i,1}u_1(k)\\ &\quad+\eta_{2i,1}u_2(k)+\theta_{2i},\\ &i=1,\ldots,K_l,\quad l=1,2.\end{aligned}$$

For a given operating point $[y_{1,0}(k), y_{2,0}(k), u_{1,0}(k), u_{2,0}(k)]$, the degrees of fulfillment β_{li} are first computed by using (4) and the parameters $\boldsymbol{\zeta}^*_l$, $\boldsymbol{\eta}^*_l$ and $\boldsymbol{\theta}^*_l$ by using (7). The model outputs are then

$$\begin{aligned}\mathbf{y}_1(k+1)&=\boldsymbol{\zeta}^*_1\boldsymbol{\xi}_1(k)+\boldsymbol{\eta}^*_1\mathbf{u}(k)+\theta^*_1\\ \mathbf{y}_2(k+1)&=\boldsymbol{\zeta}^*_2\boldsymbol{\xi}_2(k)+\boldsymbol{\eta}^*_2\mathbf{u}(k)+\theta^*_2.\end{aligned}$$

The state, input, and output vectors for the state-space description are

$$\begin{aligned}\mathbf{x}_{\text{lin}}(k)&=[y_1(k),y_1(k-1),y_1(k-2),y_2(k),y_2(k-1),\ldots\\ &\qquad y_2(k-2),u_1(k-1),u_2(k-1),1]^T\\ \mathbf{u}(k)&=[u_1(k),u_2(k)]^T\\ \mathbf{y}_{\text{lin}}(k)&=[x_{\text{lin},1}(k),x_{\text{lin},4}(k)]^T\end{aligned}$$

and the $\mathbf{C}$, $\mathbf{B}$, and $\mathbf{A}$ matrices are

$$\mathbf{C}=\begin{bmatrix}1&0&0&0&0&0&0&0&0\\0&0&0&1&0&0&0&0&0\end{bmatrix}$$

$$\mathbf{B}=\begin{bmatrix}\eta^*_{1,1}&\eta^*_{1,2}\\0&0\\0&0\\\eta^*_{2,1}&\eta^*_{2,2}\\0&0\\0&0\\1&0\\0&1\\0&0\end{bmatrix}$$

$$\mathbf{A}=\begin{bmatrix}\zeta^*_{1,1}&0&\zeta^*_{1,2}&0&0&0&\zeta^*_{1,3}&\zeta^*_{1,4}&\theta^*_1\\1&0&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0\\0&0&0&\zeta^*_{2,1}&0&\zeta^*_{2,2}&\zeta^*_{2,3}&\zeta^*_{2,4}&\theta^*_2\\0&0&0&1&0&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&1\end{bmatrix}.$$

C. Relation to Jacobian Linearization

Note that the aforementioned linearization method differs from the standard Jacobian linearization.[1] Recall the input–output representation of the TS fuzzy model (3)

$$\begin{aligned}\mathbf{y}_l(k+1)&=\frac{\sum_{i=1}^{K_l}\beta_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)\cdot\left(\boldsymbol{\zeta}_{li}\boldsymbol{\xi}(k)+\boldsymbol{\eta}_{li}\mathbf{u}(k)+\theta_{li}\right)}{\sum_{i=1}^{K_l}\beta_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)}\\ &=\sum_{i=1}^{K_l}\omega_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)\cdot f_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)\end{aligned} \tag{9}$$

where

$$\omega_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)=\frac{\beta_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)}{\sum_{i=1}^{K_l}\beta_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)}$$

is the normalized degree of fulfillment and

$$f_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)=\boldsymbol{\zeta}_{li}\boldsymbol{\xi}(k)+\boldsymbol{\eta}_{li}\mathbf{u}(k)+\theta_{li}$$

is the linear model of the ith fuzzy rule.

Applying the Jacobian linearization to (9), we have

$$\begin{aligned}\frac{\partial\mathbf{y}_l(k+1)}{\partial\boldsymbol{\xi}(k)}&=\sum_{i=1}^{K_l}\left(\frac{\partial\omega_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)}{\partial\boldsymbol{\xi}(k)}\cdot f_{li}\left(\boldsymbol{\xi}(k)\right)\right.\\ &\qquad\left.+\,\omega_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)\cdot\frac{\partial f_{li}\left(\boldsymbol{\xi}(k)\right)}{\partial\boldsymbol{\xi}(k)}\right)\\ \frac{\partial\mathbf{y}_l(k+1)}{\partial\mathbf{u}(k)}&=\sum_{i=1}^{K_l}\left(\frac{\partial\omega_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)}{\partial\mathbf{u}(k)}\cdot f_{li}\left(\mathbf{u}(k)\right)\right.\\ &\qquad\left.+\,\omega_{li}\left(\boldsymbol{\xi}(k),\mathbf{u}(k)\right)\cdot\frac{\partial f_{li}\left(\mathbf{u}(k)\right)}{\partial\mathbf{u}(k)}\right).\end{aligned}$$

[1]The application of Jacobian linearization to TS fuzzy models is discussed, among others, in [18].

Comparing this result with (6) and (7), one can see that the terms containing the derivatives of the membership functions (degrees of fulfillment) are missing in these expressions. Thus, we can conclude that rather than a variant of the standard linearization, our approach is parameter scheduling that relies on the assumption that the individual rules of the TS model are good local linear models of the system. The linear parameter-varying model obtained at each operating condition then "tracks" the nonlinear process dynamics in a way similar to using a linear adaptive model to track nonlinear dynamics. This is in contrast with the "global approach," where the fuzzy rules typically do not have any local interpretation and the emphasis is on the global approximation accuracy. For the global approach, the standard Jacobian linearization would be more appropriate. For more details, see [18] and [19]. A clear advantage of our approach is that the (possibly time-consuming) computation of the membership function derivatives is avoided.

D. Optimization in Predictive Control

The considered predictive controller computes an optimal control sequence with respect to the following cost function:

$$\begin{aligned} J(\mathbf{u}) = &\sum_{i=H_{\min}}^{H_p} \|\mathbf{r}_y(k+i) - \hat{\mathbf{y}}(k+i)\|^2_{\mathbf{P}_i} \\ &+ \sum_{i=H_{\min}}^{H_p} \|\Delta\hat{\mathbf{y}}(k+i-1)\|^2_{\Delta\mathbf{P}_i} \\ &+ \sum_{j=1}^{H_c} \|\mathbf{r}_u(k+j-1) - \mathbf{u}(k+j-1)\|^2_{\mathbf{Q}_i} \\ &+ \sum_{j=1}^{H_c} \|\Delta\mathbf{u}(k+j-1)\|^2_{\Delta\mathbf{Q}_j}. \end{aligned} \tag{10}$$

Here, $\hat{\mathbf{y}}$ is the output predicted by the fuzzy model, $\Delta\hat{\mathbf{y}}$ is the output increment, $\mathbf{r}_y$ and $\mathbf{r}_u$ are the output and input references, $\mathbf{u}$ and $\Delta\mathbf{u}$ are the control signal and its increment, respectively, and $\|\cdot\|$ represents the inner-product norm. The parameters $H_{\rm p}$, $H_{\rm c}$ and $H_{\min}$ are the *prediction*, *control* and *minimum cost* horizon, respectively, over which the optimization is carried out. The weights $\mathbf{P}_i$, $\Delta\mathbf{P}_i$, $\mathbf{Q}_j$, and $\Delta\mathbf{Q}_j$ determine the relative importance of the different terms in the cost function. Note that in addition to the standard terms, this cost function also includes the variation in the predicted output and the deviation of the control input from an input reference (the second and third terms). By limiting the variation in the predicted output by penalizing it in the cost function rather than by hard constraints, one can avoid infeasibility of the optimization problem. Similarly, the weighting of the deviation of the control input from an input reference provides a certain flexibility of the control signal as opposed to the use of hard constraints. The inputs and outputs are subject to (time-varying) level and rate constraints

$$\begin{aligned} \mathbf{u}_{\min}(j) \le \mathbf{u}(j) \le \mathbf{u}_{\max}(j) \\ \Delta\mathbf{u}_{\min}(j) \le \Delta\mathbf{u}(j) \le \Delta\mathbf{u}_{\max}(j) \\ \mathbf{y}_{\min}(i) \le \mathbf{y}(i) \le \mathbf{y}_{\max}(i) \\ \Delta\mathbf{y}_{\min}(i) \le \Delta\mathbf{y}(i) \le \Delta\mathbf{y}_{\max}(i) \end{aligned}$$

where $i = k + H_{\min}, \ldots, k + H_p$ and $j = k+1, \ldots, k + H_c$. The output constraints can be transformed into constraints on the control signal [20], hence, the aforementioned constraints can be written as

$$G_j(\mathbf{u}(k)) \le 0, \qquad j = 1, \ldots, c \tag{11}$$

where c is the total number of constraints. The exact form of G_j will be given in the sequel.

Given the optimization problem (10) and (11), and seeking the optimal control, $\mathbf{u}_{\rm opt}$, the idea is to formulate a QP sub-problem based on a second-order approximation of the Lagrangian function

$$L(\mathbf{u}, \lambda) = J(\mathbf{u}) + \sum_{j=1}^{c} \lambda_j \cdot G_j(\mathbf{u}). \tag{12}$$

If the optimization problem is a convex programming problem, i.e., $J(\mathbf{u})$ and $G_j(\mathbf{u}(k))$, $j = 1, \ldots, c$ are convex functions, then the Kuhn–Tucker equations

$$\nabla J(\mathbf{u}_{\rm opt}) + \sum_{j=1}^{c} \lambda_j^{\rm opt} \cdot \nabla G_j(\mathbf{u}_{\rm opt}) = 0 \tag{13}$$

are the necessary and sufficient conditions for the attainment of a global optimum [21]. The previous equation describes the cancellation of the gradients between the cost function J and the active constraints G_j at the solution point $\mathbf{u}_{\rm opt}$. For the gradients to be canceled, the Lagrange multipliers $(\lambda_j, j = 1, \ldots, c)$ are necessary to balance the deviations of the gradients in magnitude. Since only active constraints are included in this canceling operation, the remaining Lagrange multipliers are zero.

If we use the nonlinear TS model, however, the cost function is nonconvex with respect to the control signal. For the resulting optimization problem, the KT equations are then only necessary but not sufficient to guarantee that a global minimum is attained. With the method we propose, the optimization problem is approached in a different manner. Instead of linearly approximating the gradient and the Hessian of the Lagrangian function, we approximate the original fuzzy model through the LTV model (8). The gradient and the Hessian of (12), are updated at each iteration through a quasi-Newton approximation, and are used in a QP sub-problem whose solution forms the search direction. Using the LTV model (8), we can apply the results from linear MPC [22] to get a direct (not approximated) expression for the gradient and the Hessian of (12). The convergence of the optimized control sequences to the optimal one is guaranteed through the line search routine considering reduction both in the cost function and in the constraints [23]. This line-search routine is presented in Section IV.

Let the fuzzy model be locally represented by the LTV model (5). To ensure offset-free reference tracking, the optimization problem is defined with respect to the increment in the control signal, $\Delta\mathbf{u}$, rather than the control signal $\mathbf{u}$. The state-space description is extended correspondingly

$$\begin{aligned} \begin{bmatrix} \mathbf{x}_{\rm lin}(k+1) \\ \mathbf{u}(k) \end{bmatrix} &= \begin{bmatrix} \mathbf{A}(k) & \mathbf{B}(k) \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\rm lin}(k) \\ \mathbf{u}(k-1) \end{bmatrix} \\ &\quad + \begin{bmatrix} \mathbf{B}(k) \\ \mathbf{I} \end{bmatrix} \Delta\mathbf{u}(k) \\ \mathbf{y}_{\rm lin}(k) &= [\mathbf{C}(k) \quad \mathbf{0}] \begin{bmatrix} \mathbf{x}_{\rm lin}(k) \\ \mathbf{u}(k-1) \end{bmatrix} \end{aligned} \tag{14}$$

$$\Updownarrow$$

$$\bar{\mathbf{x}}_{\text{lin}}(k+1) = \bar{\mathbf{A}}(k)\bar{\mathbf{x}}_{\text{lin}}(k) + \bar{\mathbf{B}}(k)\Delta\mathbf{u}(k)$$
$$\mathbf{y}_{\text{lin}}(k) = \bar{\mathbf{C}}(k)\bar{\mathbf{x}}_{\text{lin}}(k). \tag{15}$$

For the sake of simplicity, in the sequel we drop the bars from this extended state-space description. Assuming that at time instant k, the state vector and the future control sequence are known, the future process outputs can be predicted through successive substitution. The complete output sequence over the prediction horizon (for $H_{\min} = 1$) is given by

$$\begin{bmatrix} \hat{\mathbf{y}}(k+1) \\ \hat{\mathbf{y}}(k+2) \\ \vdots \\ \hat{\mathbf{y}}(k+H_p) \end{bmatrix} = \mathbf{R}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) + \mathbf{R}_u \begin{bmatrix} \Delta\mathbf{u}(k) \\ \Delta\mathbf{u}(k+1) \\ \vdots \\ \Delta\mathbf{u}(k+H_c-1) \end{bmatrix} \tag{16}$$

where

$$\mathbf{R}_x = \begin{bmatrix} \mathbf{C}(k) \\ \mathbf{C}(k+1)\mathbf{A}(k) \\ \vdots \\ \mathbf{C}(k+H_p)\prod_{i=H_p-1}^{0}\mathbf{A}(k+i) \end{bmatrix}$$

and $\mathbf{R}_u$ is given at the bottom of the page. Given the LTV model (15), the constrained linear MPC (10), (11) is solved as the following QP:

$$\min_{\Delta\mathbf{u}} \left\{ \frac{1}{2}\Delta\mathbf{u}^T \cdot \mathbf{H} \cdot \Delta\mathbf{u} + \mathbf{f}^T \cdot \Delta\mathbf{u} \right\} \tag{17}$$

subject to input and output constraints transformed into constraints on the input increment

$$\begin{bmatrix} -\mathbf{I}_{\Delta u} \\ \mathbf{I}_{\Delta u} \\ -\mathbf{I}_{H_p m} \\ -\mathbf{I}_{H_p m} \\ -\mathbf{R}_u \\ \mathbf{R}_u \\ -\mathbf{dR}_{u1} \\ \mathbf{dR}_{u1} \\ -\mathbf{dR}_u \\ \mathbf{dR}_u \end{bmatrix} \Delta\mathbf{u} \le \begin{bmatrix} \mathbf{I}_u\,(-\mathbf{u}_{\min}+\mathbf{u}(k-1)) \\ \mathbf{I}_u\,(\mathbf{u}_{\max}-\mathbf{u}(k-1)) \\ -\mathbf{I}_{H_p m}\Delta\mathbf{u}_{\min} \\ \mathbf{I}_{H_p m}\Delta\mathbf{u}_{\max} \\ -\mathbf{y}_{\min}+\mathbf{R}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \\ \mathbf{y}_{\max}-\mathbf{R}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \\ -\Delta\mathbf{y}_{\min 1}+\mathbf{dR}_{x1}\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k)-\mathbf{y}_{\text{lin}}(k) \\ \Delta\mathbf{y}_{\max 1}-\mathbf{dR}_{x1}\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k)+\mathbf{y}_{\text{lin}}(k) \\ -\Delta\mathbf{y}_{\min}+\mathbf{dR}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \\ \Delta\mathbf{y}_{\max}-\mathbf{dR}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \end{bmatrix}. \tag{18}$$

The Hessian $\mathbf{H}$ and the gradient $\mathbf{f}$ of the Lagrangian function (12), are expressed in terms of the LTV model parameters

$$\begin{aligned} \mathbf{H} &= 2\left\{\mathbf{R}_u^T \cdot \bar{\mathbf{P}} \cdot \mathbf{R}_u + (\mathbf{R}_u - \mathbf{R}_{u2})^T \cdot \Delta\bar{\mathbf{P}} \cdot (\mathbf{R}_u - \mathbf{R}_{u2}) \right. \\ &\quad \left. + \mathbf{I}_{\Delta u}^T \cdot \bar{\mathbf{Q}} \cdot \mathbf{I}_{\Delta u} + \Delta\bar{\mathbf{Q}}\right\} \\ \mathbf{f} &= 2\Big\{(\mathbf{R}_x \cdot \mathbf{A}(k) \cdot \mathbf{x}_{\text{lin}}(k) - \mathbf{r}_y)^T \cdot \bar{\mathbf{P}} \cdot \mathbf{R}_u \\ &\quad + (\mathbf{I}_u \cdot \mathbf{u}(k) - \mathbf{r}_u)^T\, \bar{\mathbf{Q}}\mathbf{I}_{\Delta u} + (\mathbf{R}_x \cdot \mathbf{A}(k) \cdot \mathbf{x}_{\text{lin}}(k) \\ &\quad - \mathbf{R}_x \cdot \mathbf{x}_{\text{lin}}(k))^T \cdot \Delta\bar{\mathbf{P}}(\mathbf{R}_u - \mathbf{R}_{u2})\Big\}. \end{aligned} \tag{19}$$

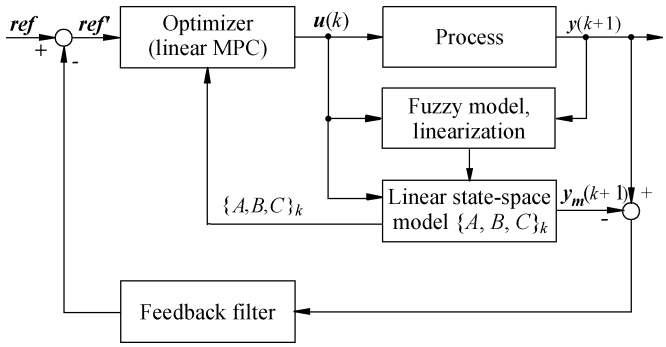


Fig. 1. Fuzzy model predictive control scheme.

The advantage of the such posed optimization problem is that now the $\mathbf{H}$ and $\mathbf{f}$ can be derived in a straightforward way, as it is shown in Appendix A, where all the remaining matrices in (19) are also defined.

E. Internal Model Control Scheme

To compensate for the disturbances acting on the process, measurement noise and model-plant mismatch, the internal model control (IMC) scheme [24] is applied (Fig. 1). This scheme guarantees nominal stability when the model-plant mismatch can be neglected. For a significant model-plant mismatch, stability can be guaranteed by imposing additional constraints on the control signal [17].

III. Methods for Obtaining Linear Models

The basic idea is to use the fuzzy model to predict the future process behavior. LTV models are then derived around the obtained input and output trajectories. Fig. 1 presents the flow diagram of the fuzzy model predictive controller. Denote by $\mathcal{U} = [\mathbf{u}(k+1), \ldots, \mathbf{u}(k+H_p)] \in \mathbb{R}^{m\times H_p}$ and $\mathcal{Y} = [\mathbf{y}(k+1), \ldots, \mathbf{y}(k+H_p)] \in \mathbb{R}^{p\times H_p}$ some general input and output model trajectories, and by $\mathcal{U}^* \in \mathbb{R}^{m\times H_p}$ and $\mathcal{Y}^* \in \mathbb{R}^{p\times H_p}$ the specific trajectories computed by the linear predictive controller.

Further denote the LTV model (5) obtained at the $(k+i)$th step by $M(k+i) = \{\mathbf{A}(k+i), \mathbf{B}(k+i), \mathbf{C}(k+i)\}$, and the set of models $M(k+i)$ extracted along the trajectories $\mathcal{U}$ and $\mathcal{Y}$ by $\mathcal{M} = \{M(k+1), \ldots, M(k+H_p)\}$. Depending on the way $\mathcal{M}$ is obtained, several approaches can be distinguished. Generally, they can be classified into two groups: noniterative and iterative methods.

A. Noniterative Methods

In the noniterative methods, a single local model or a set of local models, obtained at the current time instant k, is used in

$$\mathbf{R}_u = \begin{bmatrix} \mathbf{C}(k+1)\mathbf{B}(k) & \ldots & \mathbf{0} \\ \mathbf{C}(k+2)\mathbf{A}(k+1)\mathbf{B}(k) & \ldots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{C}(k+H_p)\prod_{i=H_p-1}^{1}\mathbf{A}(k+i)\mathbf{B}(k) & \ldots & \mathbf{C}(k+H_p)\prod_{i=H_p-H_c}^{H_c}\mathbf{A}(k+i)\mathbf{B}(k+H_c-1) \end{bmatrix}.$$

the quadratic program (15)–(18) and the obtained QP solution is directly applied to the controlled process.

1) Single-Model (SM) Method: The model set $\mathcal{M}$ contains a single linear model $M(k)$, extracted at the operating point $\{\mathbf{u}(k-1), \mathbf{y}(k)\}$. This model is used throughout the entire prediction horizon. Even if this model is very accurate at the linearization point, its accuracy decreases over the prediction horizon. As a consequence, there may be a significant prediction error at $k + H_p$. To reduce the error, a set of local linear models can be extracted from the fuzzy model along the trajectories found at the previous time instant $k-1$. This is the basis of the multimodel (MM) method.

2) Multiple Models Based on the Trajectory Computed at the Previous Sampling Instant: Let $\mathcal{U}^*(k-1)$ and $\mathcal{Y}^*(k-1)$ be the input and output sequences obtained by optimization at time step $k-1$ and by subsequent simulation of the nonlinear fuzzy model

$$\mathcal{U}^*(k-1) = [\mathbf{u}^*(k-1), \mathbf{u}^*(k), \ldots, \mathbf{u}^*(k+H_p-1)]$$
$$\mathcal{Y}^*(k-1) = [\mathbf{y}^*(k-1), \mathbf{y}^*(k), \ldots, \mathbf{y}^*(k+H_p-1)].$$

We use this information to approximate the yet unknown input and output sequences at time k by shifting the elements one step forward and replicating the last element

$$\mathcal{U}(k) = [\mathbf{u}^*(k), \ldots, \mathbf{u}^*(k+H_p-1), \mathbf{u}^*(k+H_p-1)]$$
$$\mathcal{Y}(k) = [\mathbf{y}^*(k), \ldots, \mathbf{y}^*(k+H_p-1), \mathbf{y}^*(k+H_p-1)].$$

The local linear models, $\mathcal{M} = \{M(k), M(k+1) \ldots, M(k+H_p)\}$ extracted from the fuzzy model along $\{\mathcal{U}(k), \mathcal{Y}(k)\}$ are then used to construct the quadratic programme and to compute $\mathcal{U}^*(k)$. In this way, the prediction error is reduced.

Note, however, that this method is superior to the SM method only if the fuzzy model is accurate enough.

B. Iterative Methods

With the noniterative methods, the performance still may be suboptimal due to the fact that the control sequence, along which the fuzzy model is linearized, is from the previous time step. To improve the performance, iterative methods can be introduced, in which the optimized control sequence is not directly applied to the process, but it is first used to simulate the fuzzy model. The resulting model outputs (and the control inputs) then provide more accurate trajectories along which a new set of local models are obtained and the whole procedure is iteratively repeated.

1) Iterative Version of the SM Method: In the first iteration, a linear model $M(k)$ is obtained at $\{\mathbf{u}^*(k-1), \mathbf{y}(k)\}$. This model is used to compute $\mathcal{U}^*(k)$. Thereafter, in the following iterations, $\{\mathbf{u}^*(k), \mathbf{y}(k)\}$ is used as the linearization point (see Fig. 2).

To further reduce the error, a set of local linear models can be extracted from the fuzzy model along the trajectories in an iterative manner. This is the basis of the following iterative MM method.

2) MM Method: Rather than using a single model over the whole prediction horizon, separate models are employed for each step within the prediction horizon. The algorithm is summarized as follows.

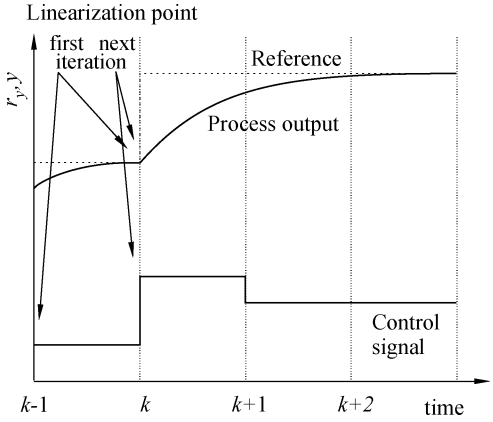


Fig. 2. Linearization points within the prediction horizon.

1) Use the already obtained linear model $M(k)$ and compute the control sequence $\mathcal{U}^*$ for the whole prediction horizon.
2) Simulate the fuzzy model over the prediction horizon.
3) Linearize the fuzzy model along the predicted trajectory $\{\mathcal{U}^*, \mathcal{Y}^*\}$ and obtain $\mathcal{M}$.
4) Use $\mathcal{M}$ to compute a new control sequence $\mathcal{U}^*$ for the whole prediction horizon.

Again, this method should be used only if the fuzzy model is accurate enough, otherwise the iterative version of the SM method generally gives better results.

Although, according to the receding horizon principle, only the first control action is applied to the process, the complete sequence is available. This can be used to initialize this iterative routine at the next sampling instant, starting from a point close to the optimum. Table I summarizes how the linear models are obtained from the fuzzy model by the different methods.

IV. Line Search

The iterative optimization methods account for errors introduced by the linearization. In such an iterative optimization, the QP solution provides a search direction toward the minimum of the optimization problem. To guarantee convergence, the following line search mechanism is used that considers reduction both in the cost function and in the constraints. In the following, index j denotes the jth element of a vector and index i the ith iteration.

Let $\mathbf{d}_i(k)$ be the QP solution (Section II-D) in ith iteration at time instant k. To provide a convergence mechanism, we use

$$\mathbf{u}_i(k) = \begin{cases} \mathbf{u}(k-1) + \alpha_i \mathbf{d}_i(k), & \text{for } i = 1 \\ \mathbf{u}_{i-1}(k) + \alpha_i \mathbf{d}_i(k), & \text{for } i > 1 \end{cases} \tag{20}$$

where $0 < \alpha_i \leq 1$ is the step length, chosen by a line search routine to give a reduction in the merit function proposed in [23]

$$\Psi(\mathbf{u}_i, \mathbf{r}_i) = J(\mathbf{u}_i) + \sum_{j=1}^{c} \mathbf{r}_{i,j} \cdot \max[0, G_j(\mathbf{u}_i)] \tag{21}$$

TABLE I
METHODS FOR OBTAINING LOCAL LINEAR MODELS. IN THE ITERATIVE METHODS THESE LOCAL LINEAR MODELS ARE OBTAINED FROM THE FUZZY MODEL IN EACH ITERATION ALONG THE CORRESPONDING SEQUENCES $\mathcal{Y}_{\text{Iter}}(k)$ AND $\mathcal{U}^*_{\text{Iter}}(k)$

Method, *Iteration*	Model used at time instant				
	k	$k+1$	$k+2$	$\dots$	$k+H_p$
Single model (III-A.1)	$M(k)$	$M(k)$	$M(k)$	$\dots$	$M(k)$
Multiple model based on trajectory computed at the previous instant (III-A.2)	$M(k)$	$M(k+1)$	$M(k+2)$	$\dots$	$M(k+H_p)$
Iterative version of the *Single model* (III-B.1)					
Iteration 1	$M_1(k)$	$M_1(k)$	$M_1(k)$	$\dots$	$M_1(k)$
Iteration 2	$M_2(k)$	$M_2(k)$	$M_2(k)$	$\dots$	$M_2(k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Iteration N	$M_N(k)$	$M_N(k)$	$M_N(k)$	$\dots$	$M_N(k)$
Multi-model (III-B.2)					
Iteration 1	$M_1(k)$	$M_1(k+1)$	$M_1(k+2)$	$\dots$	$M_1(k+H_p)$
Iteration 2	$M_2(k)$	$M_2(k+1)$	$M_2(k+2)$	$\dots$	$M_2(k+H_p)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Iteration N	$M_N(k)$	$M_N(k+1)$	$M_N(k+2)$	$\dots$	$M_N(k+H_p)$

where $\mathbf{r} \in \mathbb{R}^c$ is a vector of positive parameters, updated according to

$$\mathbf{r}_{i+1,j} = \max_j \left\{ \lambda_j, \frac{1}{2}(\mathbf{r}_{i,j} + \lambda_j) \right\}, \qquad j = 1, \dots, c \quad (22)$$

with λ_j being the Lagrangian multipliers for the current QP solution. This allows positive contributions from constraints that are inactive in the QP solution, but were recently active. The penalty parameter $\mathbf{r}$ is initially set to

$$\mathbf{r}_{1,j} = \frac{\|\nabla J(\mathbf{u})\|}{\|\nabla G_j(\mathbf{u})\|}, \qquad j = 1, \dots, c \quad (23)$$

where $\|\cdot\|$ represents the Euclidean norm. This results in larger contributions to the penalty parameter from constraints with smaller gradients, which would be the case for active constraints at the solution point.

Let α_1 be the first element in a monotonically decreasing sequence $\{\alpha_1, \alpha_2, \alpha_3, \dots\}$. Then, it can be proven [23] that the line-search method generates a sequence $\{\mathbf{u}_i(k), i = 1, 2, 3, \dots\}$ whose limit is the Kuhn–Tucker point of the optimization problem (15)–(18). The line search routine is summarized as follows.

1) In the first iteration, $\mathbf{r}$ is initialized according to (23), otherwise, it is modified according to (22).
2) The cost function $J(\mathbf{u}_i(k))$ and the constraints $G_j(\mathbf{u}_i(k))$, $j = 1, \dots, c$ are calculated. The merit function $\Psi(\mathbf{u}_i(k), \mathbf{r})$ is split into two parts, looking for improvements in the cost function, Ψ_J, and the constraints Ψ_G

$$\Psi_{J,\text{init}} = -\frac{1}{(J(\mathbf{u}_i(k)) + 1)}$$

$$\Psi_{G,\text{init}} = J(\mathbf{u}_i(k)) + \sum_{j=1}^{c} \mathbf{r}_{i,j} \cdot \max(0, G_j(\mathbf{u}_i(k))).$$

3) The step length α_i is set to $\alpha_i = 2$, $\Psi_J = \Psi_{J,\text{init}} + 1$ and $\Psi_G = \Psi_{G,\text{init}} + 1$.
4) Loop wherein the step length α_i is reduced until improvement in either Ψ_J or Ψ_G is achieved or the maximum number of iterations is reached.

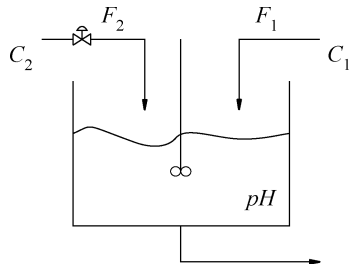


Fig. 3. Continuous stirred tank reactor.

```
while Ψ_J > Ψ_J,init and Ψ_G > Ψ_G,init and
  NOT maxiterations
    update the step length α_i := α_i/2
    update the solution u_i(k) := u_i−1(k) + α_i · d_i(k)
    update the cost function J := J(u_i(k))
    update the constraints and take max
    G_j := G_j(u_i(k)), j = 1,...,c
    G_max := max_j(G_j)
    update Ψ_J and Ψ_G
    if G_max > 0 then Ψ_J := G_max
    elseif J ≥ 0 then Ψ_J := −1/(J + 1)
    else Ψ_J := 0
    Ψ_G := J + Σ_{j=1}^{c} r_{i,j} · max(0, G_j(u_i(k)))
end loop
```

The introduction of the line search routine, however, as reported by [23] may decrease the convergence rate. The situation $\alpha = 1$ which gives superlinear convergence in (20) is not always allowed by the merit function (21).

V. APPLICATION TO A pH PROCESS

A. Process Description

The control of the pH (the concentration of hydrogen ions) in a continuous stirred tank reactor (CSTR) is a well-known benchmark problem that presents difficulties due to the nonlinearity of the process dynamics [25]. The CSTR, given in Fig. 3, has two input streams, one containing sodium hydroxide and the other one acetic acid. The acid neutralizes sodium hydroxide of concentration C_2 which flows into the reactor at a

TABLE II
PARAMETERS USED IN THE CSTR SIMULATION STUDY

Parameter	Description	Nominal Value
V	Volume of the tank	1000 [l]
F_1	Flow rate of acetic acid	81 [l/min]
F_2	Flow rate of NaOH	515 [l/min]
C_1	Inlet concentration of acetic acid	0.32 [mol/l]
C_2	Inlet concentration of NaOH	0.05 [mol/l]
$\chi = [Na^+]$	Initial concentration of sodium in the CSTR	0.0432 [mol/l]
$\rho = [HAC + AC^-]$	Initial concentration of acetate in the CSTR	0.0432 [mol/l]
K_a	Acid equilibrium constant	1.75310^{-5}
K_w	Water equilibrium constant	10^{-14}

rate F_2. The volume of the reactor is constant and equal to V. The variable of interest is the concentration of hydrogen ions, pH, ($\mathrm{pH} = -\log_{10}[H^+]$) of the outlet stream, which can be expressed by (24), as shown at the bottom of the page.

The parameters used in the simulation are given in Table II. For simplicity the acetic acid stream, F_1, is considered to be constant at its nominal value. Thus, we end up with a single-input–single-output process with the sodium hydroxide stream F_2 as input and the pH as output.

B. Fuzzy Modeling

A TS fuzzy model is obtained from simulated input-output data through fuzzy clustering-based identification [26]. A sampling time $T_s = 12$ s is used. A first-order fuzzy model with one sample time delay in the input appeared to give satisfactorily results. Three fuzzy IF–THEN rules are used

$$\begin{aligned}
&1)\ \text{If } pH(k) \text{ is LOW and } F_2(k) \text{ is LOW}\\
&\quad \text{then } pH_1(k+1) = 0.868pH(k) + 0.046F_2(k) - 22.9\\
&2)\ \text{If } pH(k) \text{ is MEDIUM and } F_2(k) \text{ is MEDIUM}\\
&\quad \text{then } pH_2(k+1) = 0.909pH(k) + 0.187F_2(k) - 96.0\\
&3)\ \text{If } pH(k) \text{ is HIGH and } F_2(k) \text{ is HIGH}\\
&\quad \text{then } pH_3(k+1) = 0.817pH(k) + 0.122F_2(k) - 61.4
\end{aligned} \tag{25}$$

with piecewise exponential membership functions

$$\mu(x; c_l, c_r, w_l, w_r) = \begin{cases} \exp\left(-\left(\frac{x-c_l}{2w_l}\right)^2\right), & \text{if } x < c_l \\ \exp\left(-\left(\frac{x-c_r}{2w_r}\right)^2\right), & \text{if } x > c_r \\ 1, & \text{otherwise.} \end{cases}$$

The values of the left and right shoulders (c_l, c_r) and the left and right widths (w_l, w_r) can be seen in Fig. 4. The performance of the model for an validation data set is shown in Fig. 5. To give a quantitative measure of the model accuracy, two performance

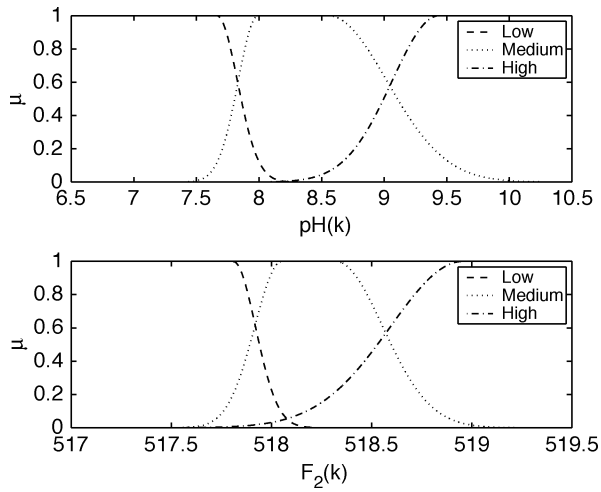


Fig. 4. CSTR: Membership functions.

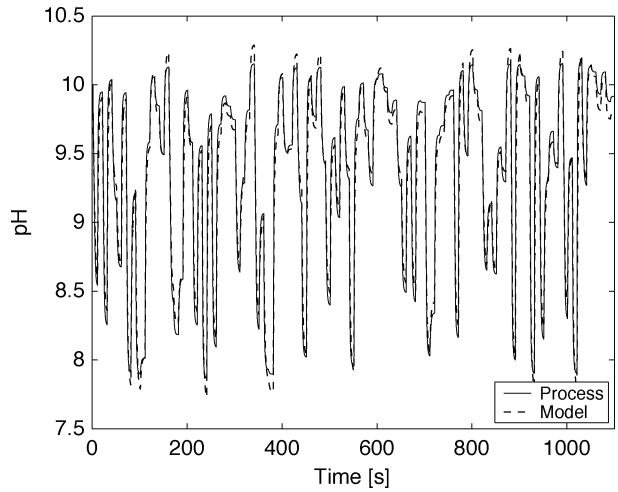


Fig. 5. CSTR: Validation of the fuzzy model (solid line: process output, dashed line: model prediction).

indices are used: The root mean square (rms) error and the variance accounted for (VAF), in %. For the validation data, the obtained measures are $\mathrm{rms} = 0.1024$ and $\mathrm{VAF} = 98.02$. For a comparison, a linear model has correspondingly $\mathrm{rms} = 0.4225$, $\mathrm{VAF} = 63.41$ on the same validation data set, thus the fuzzy model is significantly more accurate than the linear one.

C. Controller Configurations

Four different controller configurations are used to illustrate the performance of the optimization methods.

1) Linear MPC using a model obtained by Jacobian linearization of the nonlinear process model at ($pH = 8.3$ and $F_2 = 518.28$).
2) SM method (Section III-A.1).
3) MM method (Section III-B.2).

$$[\dot{H}^+] = \frac{1}{V} \cdot \frac{-[H^+]^2\{F_2C_2 - (F_1+F_2)\chi\} - [H+]K_a\{F_2C_2 - F_1C_1 - (F_1+F_2)(\chi-\rho)\}}{3[H^+]^2 + 2[H^+]\{K_a+\chi\} + \{K_a(\chi-\rho) - K_w\}}. \tag{24}$$

4) A nonlinear optimization method, referred to as *Nlinear*, that directly uses the fuzzy model. The MATLAB implementation of sequential quadratic programming (SQP) [27] is used.

All the methods use identical control settings: $H_{\mathrm{p}} = 5$, $H_{\min} = 1$ and $H_{\mathrm{c}} = 2$. The acetic acid flow rate is kept at its nominal value, $F_1 = 81\ \mathrm{l/min}$. The sodium hydroxide flow rate, F_2, is subject to input level and rate constraints $510 \leq F_2 \leq 525\ [\mathrm{l/min}]$ and $-2 \leq \Delta F_2 \leq 2\ [\mathrm{l/min^2}]$, respectively. The nonzero weights in the cost function (10) are $\mathbf{P} = 1.0$ and $\Delta\mathbf{Q} = 0.2$; the latter is used to prevent aggressive actions.

D. State-Space Model Extraction

To illustrate the extraction of a local linear state-space model from the fuzzy one, the first iteration in the MM controller (Section III-B.2) is outlined. The initial process input and output are $F_{2,0} = 517.5\ \mathrm{l/min}$ and $pH_0 = 7$, respectively. The fuzzy model state is $\xi = 7$. To obtain an initial linear model $M(k)$ for $k = 1$, the vector of the degrees of fulfillment $\boldsymbol{\beta}$ is calculated according to (4):

$$\boldsymbol{\beta}(517.5, 7) = [1.0,\ 0.0,\ 0.0].$$

These degrees of fulfillment are then combined then with the rules' consequents in (25), to get ζ^*, η^* and θ^* by using (7)

$$\boldsymbol{\zeta}^* = \frac{\sum_{i=1}^{3} \beta_i \cdot \boldsymbol{\zeta}_i}{\sum_{i=1}^{3} \beta_i} = [1.0\ 0.0\ 0.0] \begin{bmatrix} 0.8677 \\ 0.9095 \\ 0.8165 \end{bmatrix} = 0.8677$$

$$\boldsymbol{\eta}^* = \frac{\sum_{i=1}^{3} \beta_i \cdot \boldsymbol{\eta}_i}{\sum_{i=1}^{3} \beta_i} = [1.0\ 0.0\ 0.0] \begin{bmatrix} 0.0461 \\ 0.1867 \\ 0.1217 \end{bmatrix} = 0.0461$$

$$\boldsymbol{\theta}^* = \frac{\sum_{i=1}^{3} \beta_i \cdot \theta_i}{\sum_{i=1}^{3} \beta_i} = [1.0\ 0.0\ 0.0] \begin{bmatrix} -22.88 \\ -96.03 \\ -61.4 \end{bmatrix} = -22.88.$$

The corresponding matrices $\mathbf{A}(k)$ and $\mathbf{B}(k)$ are (14)

$$\mathbf{A}(k) = \begin{bmatrix} 0.8677 & -22.8847 \\ 0 & 1.0000 \end{bmatrix} \quad \mathbf{B}(k) = \begin{bmatrix} 0.0461 \\ 0 \end{bmatrix}$$

and $\mathbf{C}(k) = [1\ 0]$. The obtained matrices are used in the quadratic program (17) and (18). The optimized control sequence $\mathcal{U}^* = [519.8\,612,\ 518.6208]$ is then applied to simulate the fuzzy model, in order to provide new linearizations points. The degrees of fulfillment, β's, for the individual rules at these points are

$$\begin{aligned} \boldsymbol{\beta}(k+1) &= [0.0015,\ 0.2277,\ 0.7707] \\ \boldsymbol{\beta}(k+2) &= [0.0000,\ 0.9081,\ 0.0918] \\ \boldsymbol{\beta}(k+3) &= [0.0000,\ 0.9860,\ 0.0139] \\ \boldsymbol{\beta}(k+4) &= [0.0000,\ 0.9872,\ 0.0127] \\ \boldsymbol{\beta}(k+5) &= [0.0000,\ 0.9600,\ 0.0399] \end{aligned}$$

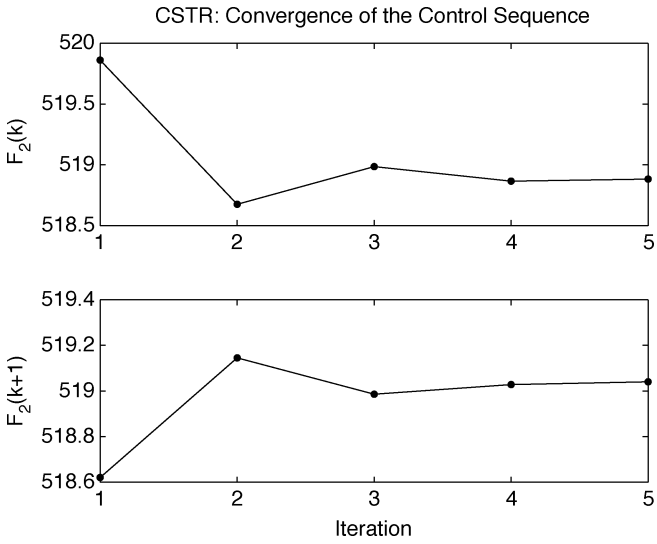


Fig. 6. CSTR: Convergence of the control sequence.

TABLE III
CSTR: DIFFERENCES AMONG ON THE CONTROL SEQUENCES THROUGH THE ITERATIONS

Iteration	$F_2(k)$			$F_2(k+1)$		
	Value	*Deviation*		*Value*	*Deviation*	
		Abs.	*Rel.,%*		*Abs.*	*Rel.,%*
1	519.8612	–	–	518.6208	–	–
2	518.6760	1.1852	39.50	519.1455	0.5247	17.49
3	518.9860	0.3100	10.33	518.9860	0.1595	5.32
4	518.8661	0.1199	3.99	519.0283	0.0423	1.41
5	518.8833	0.0172	0.57	519.0402	0.0119	0.40

for which the local state-space models $M(k+i)$ for $k = 1$ and $i = 1, \ldots, 5$ are

$$\mathbf{A}(k{+}1) = \begin{bmatrix} 0.8378 & -69.2250 \\ 0 & 1.0000 \end{bmatrix} \quad \mathbf{B}(k{+}1) = \begin{bmatrix} 0.1364 \\ 0 \end{bmatrix}$$

$$\mathbf{A}(k{+}2) = \begin{bmatrix} 0.9009 & -92.8540 \\ 0 & 1.0000 \end{bmatrix} \quad \mathbf{B}(k{+}2) = \begin{bmatrix} 0.1808 \\ 0 \end{bmatrix}$$

$$\mathbf{A}(k{+}3) = \begin{bmatrix} 0.9082 & -95.5506 \\ 0 & 1.0000 \end{bmatrix} \quad \mathbf{B}(k{+}3) = \begin{bmatrix} 0.1858 \\ 0 \end{bmatrix}$$

$$\mathbf{A}(k{+}4) = \begin{bmatrix} 0.9083 & -95.5907 \\ 0 & 1.0000 \end{bmatrix} \quad \mathbf{B}(k{+}4) = \begin{bmatrix} 0.1859 \\ 0 \end{bmatrix}$$

$$\mathbf{A}(k{+}5) = \begin{bmatrix} 0.9057 & -94.6492 \\ 0 & 1.0000 \end{bmatrix} \quad \mathbf{B}(k{+}5) = \begin{bmatrix} 0.1841 \\ 0 \end{bmatrix}.$$

This set of linear models $\mathcal{M} = \{M(k+1), \ldots, M(k+5)\}$ again forms the quadratic program (17) and (18). Now, the optimized control sequence is $\mathcal{U}^* = [518.6\,760\,519.1455]$. After three more iterations, the final control sequence is $\mathcal{U}^* = [518.8\,833,\ 519.0402]$. The convergence of the optimized control sequence is shown in Fig. 6. The absolute differences among the control sequences computed in the individual iterations are rather small, but note that the range of F_2 is very narrow: $F_2 \in [516.5, 519.5]$. Therefore, the relative differences amount up to 40% (see Table III).

TABLE IV
RELATIVE PERFORMANCE OF THE DIFFERENT CONTROLLERS. THE LINEAR MPC IS TAKEN AS 100%

Controller type	V_{pH}	V_{F_2}	FLOPS
Linear	100.0%	100.0%	100.0%
SM	88.89%	99.98%	118%
SM+LS	85.05%	99.87%	123%
MM	79.11%	99.98%	149%
MM+LS	79.11%	99.97%	153%
Nlinear	162.44%	102.4%	374%
Nlinear & SM	68.55%	99.97%	381%
Nlinear & Linear	71.91%	99.96%	378%

E. Performance Comparison

For performance evaluation, two criteria are used to calculate the output and input variations. The output variation criterion gives a quantitative measure how close the process output, $y(k)$, is to the reference $y_{\mathrm{ref}}(k)$, according to

$$V_y = \sum_{k=1}^{N} \left(\frac{y(k) - y_{\mathrm{ref}}(k)}{y_{\mathrm{ref}}(k)} \right)^2 \tag{26}$$

where N is the length of the simulation experiment. The input variation criterion gives a measure of the energy in the process input $u(k)$

$$V_u = \sum_{k=1}^{N} u^2(k). \tag{27}$$

The values of these performance indexes for the considered controllers are given in Table IV, taking the linear MPC as 100%. Note that a lower percentage means better performance. To asses the influence of the line search, the SM and MM performance is compared with and without the line search. As expected, the initial guess for the nonlinear optimization (Nlinear) turned out to be of crucial importance for the performance. A good initial guess can be obtained with the SM method (III-A.1); see "Nlinear and SM." One can also use the linear MPC to provide such an initial guess; see "Nlinear and SM."

One can see that the SM and MM methods lead to a significant improvement in the performance criteria over the values achieved through the linear MPC. This improvement is, however, achieved at the expense of a higher computational load (FLOPS, floating point operations). The introduction of the line search slightly improves the controller performance. When the nonlinear fuzzy model is used in the optimization routine, the computational load is approximately four times higher than with the linear MPC and the performance is poor. However, if the control sequence, obtained through a single iteration with SM or linear MPC is used as an initial guess to the nonlinear optimization, a superior performance is achieved. Comparing this performance to the one obtained with MM, the improvement of about 10% is paid for with nearly 2.5 times as many FLOPS, which correspondingly increases the computing time.

Fig. 7 depicts the process behavior for a part of the considered trajectory. When linear MPC is used, the process behavior is

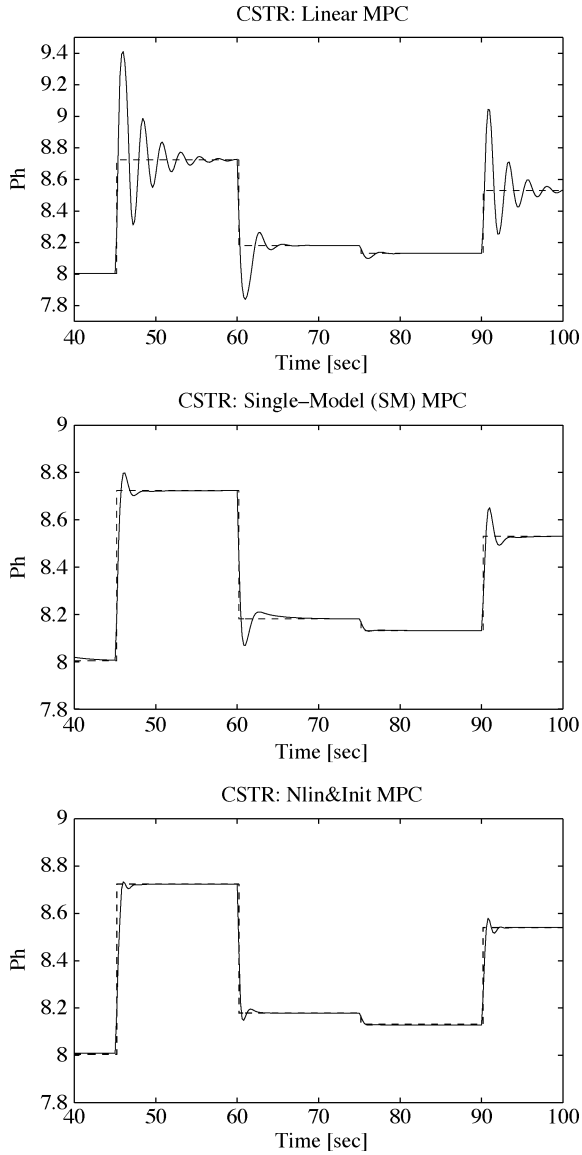


Fig. 7. CSTR: Performance of the different controllers. Top: Linear MPC. Middle: Single-model MPC. Bottom: Nlin and SM MPC.

fairly oscillatory, while both SM and ; see Nlinear and SM. provide satisfactory performance with only a marginal superiority of the latter (achieved at the expense of a computational load approximately three times higher than with the SM method).

VI. DISTILLATION COLUMN

A high-purity binary distillation column is used as a second benchmark example. The column has 39 trays, a reboiler and a condenser. The simulation model, developed by Skogestad [28] under the assumptions: 1) equilibrium on all trays, 2) total condenser, 3) no vapor holdup, and 4) linearized liquid dynamics, captures the most important dynamics of a real distillation column. The column operates in the LV configuration with the reflux and reboiler flows L and V being the manipulated variables and the top and bottom composition impurities y_D and x_B being the controlled variables.

TABLE V
CONSEQUENT PARAMETERS FOR THE FUZZY MODEL OUTPUT "TOP COMPOSITION IMPURITY" (y_D)

rule i	$\zeta_{1i,1}$	$\zeta_{1i,2}$	$\zeta_{1i,3}$	$\zeta_{1i,4}$	$\eta_{1i,1}$	$\eta_{1i,2}$	$\eta_{1i,3}$	$\eta_{1i,4}$	$\theta_{1,i}$
1	1.699	−0.698	0.2374	−0.2333	−0.001	−0.0006	0.0013	0.0075	−0.0070
2	1.804	−0.807	0.0564	−0.0560	−0.0008	−0.0003	0.0014	−0.0002	−0.0005
3	1.543	−0.545	0.1755	−0.1724	−0.0032	−0.0027	0.0038	0.0021	−0.0030
4	1.156	−0.149	0.1511	−0.1405	−0.0049	−0.007	0.0047	0.0075	−0.007

TABLE VI
CONSEQUENT PARAMETERS FOR THE FUZZY MODEL OUTPUT "BOTTOM COMPOSITION IMPURITY" (x_B)

rule i	$\zeta_{2i,1}$	$\zeta_{2i,2}$	$\zeta_{2i,3}$	$\zeta_{2i,4}$	$\eta_{2i,1}$	$\eta_{2i,2}$	$\eta_{2i,3}$	$\eta_{2i,4}$	$\theta_{2,i}$
1	0.2488	−0.2561	1.7228	−0.731	−0.0001	0.0017	−0.0026	−0.0056	0.0118
2	0.2859	−0.2803	1.4844	−0.491	0.0001	0.0089	−0.0104	0.0015	0.0044
3	0.2629	−0.2647	1.4200	−0.429	−0.0001	0.0106	−0.0114	0.0008	0.0057
4	0.3861	−0.4135	1.2472	−0.283	0.0001	0.0146	−0.0103	−0.0056	0.0118

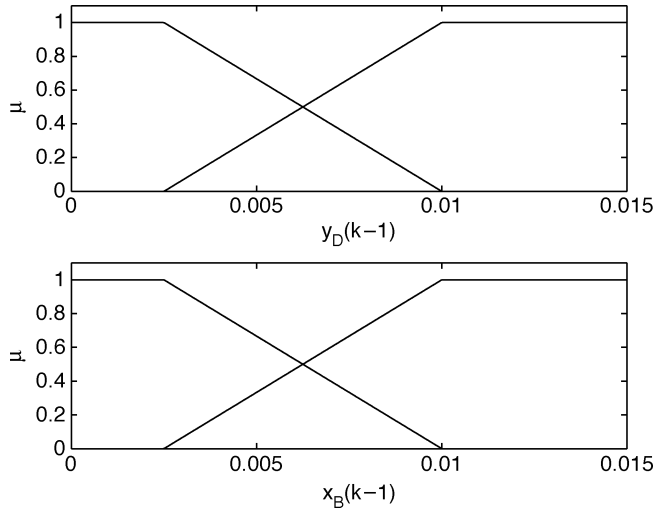


Fig. 8. Membership functions for y_D and x_B.

A. Fuzzy Modeling

A fuzzy model was constructed through fuzzy clustering [26] using 8000 input–output data pairs sampled with the sampling time of 120 s. Since the process gain is related to the concentrations, the rule antecedents are defined on the domain of the product impurities. The order of the local models is set to $n_y = n_u = 2$. The rules for both outputs are given in (28), with the trapezoidal membership functions shown in Fig. 8, and the consequent parameters given in Tables V and VI, respectively. Each output is described by four fuzzy rules $i = 1, \ldots, 4$ of the following form:

$$\begin{aligned} R_{1i}: &\text{If } y_D(k-1) \text{ is } \Omega_{1i,1} \text{ and } x_B(k-1) \text{ is } \Omega_{1i,2} \text{ then} \\ &y_D(k) = \zeta_{1i,1} \cdot y_D(k-1) + \zeta_{1i,2} \cdot y_D(k-2) \\ &\quad + \zeta_{1i,3} \cdot x_B(k-1) + \zeta_{1i,4} \cdot x_B(k-2) \\ &\quad + \eta_{1i,1} \cdot L(k-1) + \eta_{1i,2} \cdot L(k-2) \\ &\quad + \eta_{1i,3} \cdot V(k-1) + \eta_{1i,4} \cdot V(k-2) + \theta_{1i} \\ R_{2i}: &\text{If } y_D(k-1) \text{ is } \Omega_{2i,1} \text{ and } x_B(k-1) \text{ is } \Omega_{2i,2} \text{ then} \\ &x_B(k) = \zeta_{2i,1} \cdot y_D(k-1) + \zeta_{2i,2} \cdot y_D(k-2) \\ &\quad + \zeta_{2i,3} \cdot x_B(k-1) + \zeta_{2i,4} \cdot x_B(k-2) \\ &\quad + \eta_{2i,1} \cdot L(k-1) + \eta_{2i,2} \cdot L(k-2) \\ &\quad + \eta_{2i,3} \cdot V(k-1) + \eta_{2i,4} \cdot V(k-2) + \theta_{2i}. \end{aligned} \tag{28}$$

TABLE VII
PI CONTROLLERS PARAMETERS

Controller type	$K_{L,P}$	$K_{L,I}$	$K_{V,P}$	$K_{V,I}$
PI1 (Skogestad 1996)	26.1	6.94	−37.5	−11.329
PI2 (Chou et. al. 1999)	8.1	0.69	−9.5	−0.772

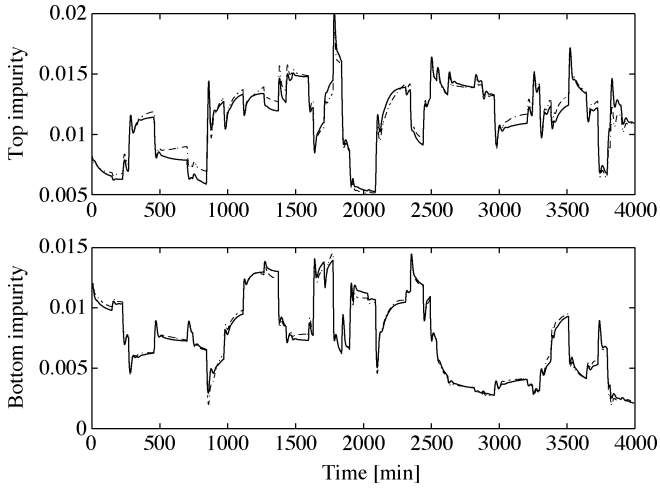


Fig. 9. Fuzzy model validation. Solid line: validation data. Dashed line: model prediction.

The model performance on a validation data set is illustrated in Fig. 9. To give a quantitative measure of the model accuracy, the rms error and the variance accounted for (VAF) are computed: $\mathrm{rms} = 0.0561$, $\mathrm{VAF} = 96\%$ for the top composition impurity (y_D) and $\mathrm{rms} = 0.0296$, $\mathrm{VAF} = 99\%$ for the bottom composition impurity (x_D).

B. Comparison of Controllers

Four different controllers are compared: two PI controllers (Table II), used in [29] and [30], respectively, and two predictive controllers using fuzzy models. Both predictive controllers use the SM method (Section III-B.1). The horizons are $H_p = 6$, $H_{\min} = 1$, $H_c = 2$ and the weighting matrices are $\mathbf{P} = \mathbf{I}$, $\Delta\mathbf{P} = \mathbf{0}$, $\mathbf{Q} = \mathbf{0}$, and $\Delta\mathbf{Q} = 0.025 \cdot \mathbf{I}$. The following constraints on the inputs are used:

$$L \in [1.7371\ 3.7371] \quad V \in [2.1536\ 4.1536]$$
$$\Delta L \in [-0.2\ 0.2] \qquad \Delta V \in [-0.2\ 0.2].$$

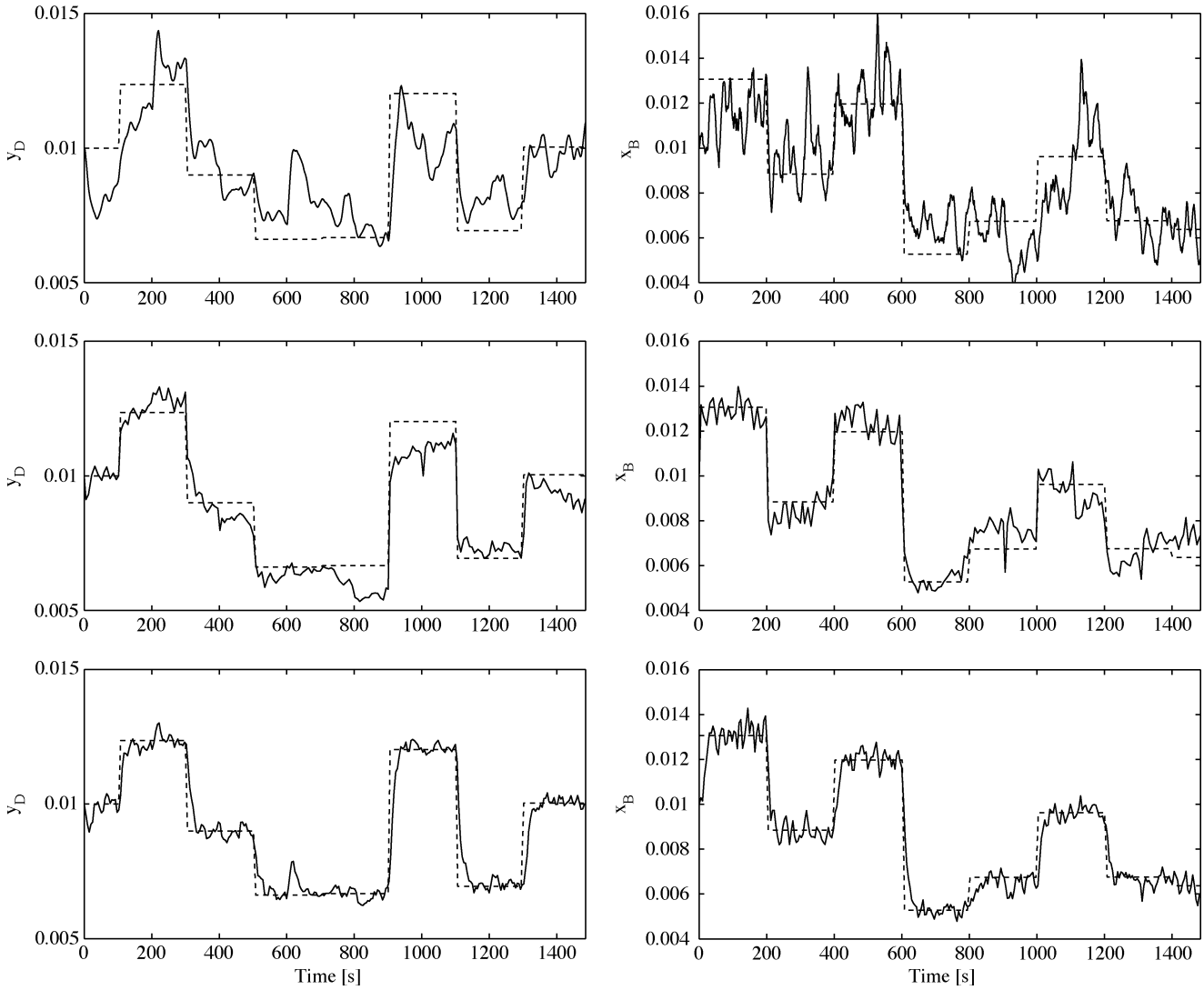


Fig. 10. Distillation column: control performance for varying feed rate and feed composition. Top: PI2 controller. Middle: Fuzzy predictive controller. Bottom: Fuzzy predictive controller with fuzzy model including the feed composition z_F. Left: Top composition impurity y_D. Right: Bottom composition impurity x_B.

TABLE VIII
RELATIVE PERFORMANCE OF THE DIFFERENT CONTROLLERS. SKOGESTAD'S PI CONTROLLER PERFORMANCE IS TAKEN AS 100%

Controller type	V_{y_D}	V_{x_B}	V_L	V_V
PI1 (Skogestad 1996)	100.0%	100.0%	100.0%	100.0%
PI2 (Chou et. al. 1999)	99.6%	98.7%	92.0%	92.5%
MPC	37.4%	33.2%	93.2%	93.7%
MPC+z_F	30.4%	32.0%	91.7%	92.5%

To make the control problem more realistic, both feed rate and feed composition are randomly varied as disturbances: $z_F \in [0.45, 0.55]$ and $F \in [0.9, 1.0]$. The process behavior for the different controllers is presented in Fig. 10. PI1 (not shown in the figure) and PI2 have a similar performance, compared to which both predictive controllers are superior (Table VIII). To show how the quality of the model prediction influences the control performance, the second MPC uses a fuzzy model which has as an input also the feed composition z_F (usually not measurable variable). Obviously, this leads to an improvement in the performance (Fig. 10).

VII. CONCLUDING REMARKS

In this paper, we have proposed and compared methods for effective optimization in fuzzy model predictive control. A linearization approach is followed to construct a quadratic optimization problem, as opposed to other approaches from the literature that use a nonlinear process model directly in the optimization and an approximation is made of the cost function gradient and Hessian. Four methods for extracting single or multiple models along the predicted trajectory based upon the fuzzy model trajectory have been introduced. Two simulation benchmark examples were presented to illustrate the advantages and the drawbacks of the methods. For the first benchmark, a detailed comparison of the proposed MPC methods has been done, including a comparison with linear and nonlinear predictive control. The performance achieved by the proposed methods is comparable to the performance attained through properly initialized nonlinear optimization, for which, however, the computational load is considerably higher. The second benchmark shows an application to a multivariable system and a comparison with alternative controllers.

APPENDIX
DERIVATION OF THE HESSIAN AND GRADIENT OF THE LAGRANGE FUNCTION BASED ON LTV MODELS

Assuming that at time instant k both the state vector and the future control sequence are known, the future process outputs can be predicted through successive substitution

$$\begin{aligned}\mathbf{x}_{\text{lin}}(k+2) &= \mathbf{A}(k+1)\mathbf{x}_{\text{lin}}(k+1)+\mathbf{B}(k+1)\Delta\mathbf{u}(k+1)\\ &= \mathbf{A}(k+1)\left[\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k)+\mathbf{B}(k)\Delta\mathbf{u}(k)\right]\\ &\quad +\mathbf{B}(k+1)\Delta\mathbf{u}(k+1)\\ &= \mathbf{A}(k+1)\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k)+\mathbf{A}(k+1)\mathbf{B}(k)\Delta\mathbf{u}(k)\\ &\quad +\mathbf{B}(k+1)\Delta\mathbf{u}(k+1)\\ &\quad \cdots\end{aligned}$$

$$\begin{aligned}\mathbf{x}_{\text{lin}}(k+H_p) = &\prod_{i=H_p-1}^{0}\mathbf{A}(k+i)\mathbf{x}_{\text{lin}}(k)\\ &+\sum_{i=0}^{H_p-1}\prod_{\substack{j=H_p-1,\\ j\geq i+1}}^{i+1}\mathbf{A}(k+j)\mathbf{B}(k+i)\Delta\mathbf{u}(k+i)\end{aligned}$$

where $\prod_{i=H_p-1}^{0}\mathbf{A}(k+i) = \mathbf{A}(k+H_p-1)\mathbf{A}(k+H_p-2)\cdots\mathbf{A}(k+1)\mathbf{A}(k)$. The predicted output follows directly:

$$\begin{aligned}\hat{\mathbf{y}}_{\text{lin}}(k+H_p) = &\mathbf{C}(k+H_p)\prod_{i=H_p-1}^{0}\mathbf{A}(k+i)\mathbf{x}_{\text{lin}}(k)\\ &+\mathbf{C}(k+H_p)\sum_{i=0}^{H_p-1}\prod_{j=H_p-1,j\geq i+1}^{i+1}\mathbf{A}(k+j)\\ &\times\mathbf{B}(k+i)\Delta\mathbf{u}(k+i).\end{aligned}\tag{29}$$

The input sequence for the input level constraints (11) can be defined as

$$\begin{aligned}\begin{bmatrix}\mathbf{u}(k)\\ \mathbf{u}(k+1)\\ \vdots\\ \mathbf{u}(k+H_c-1)\end{bmatrix} &= \underbrace{\begin{bmatrix}\mathbf{I}_m\\ \mathbf{I}_m\\ \vdots\\ \mathbf{I}_m\end{bmatrix}}_{\mathbf{I}_u}\mathbf{u}(k-1)\\ &\quad+\underbrace{\begin{bmatrix}\mathbf{I}_m & \mathbf{0} & \dots & \mathbf{0}\\ \mathbf{I}_m & \mathbf{I}_m & \dots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{I}_m & \mathbf{I}_m & \dots & \mathbf{I}_m\end{bmatrix}}_{\mathbf{I}_{\Delta u}}\cdot\begin{bmatrix}\Delta\mathbf{u}(k)\\ \Delta\mathbf{u}(k+1)\\ \vdots\\ \Delta\mathbf{u}(k+H_c-1)\end{bmatrix}\end{aligned}\tag{30}$$

where $\mathbf{I}_m$ is an $m\times m$ identity matrix. Inserting (30) in (11) gives

$$\begin{bmatrix}-\mathbf{I}_{\Delta u}\\ \mathbf{I}_{\Delta u}\end{bmatrix}\Delta\mathbf{u}\leq\begin{bmatrix}\mathbf{I}_u\left(-\mathbf{u}_{\min}(k)+\mathbf{u}(k-1)\right)\\ \mathbf{I}_u\left(\mathbf{u}_{\max}(k)-\mathbf{u}(k-1)\right).\end{bmatrix}$$

The input rate constraints (11) are, respectively

$$\begin{bmatrix}-\mathbf{I}_{H_pm}\\ \mathbf{I}_{H_pm}\end{bmatrix}\Delta\mathbf{u}\leq\begin{bmatrix}-\mathbf{I}_{H_pm}\cdot\Delta\mathbf{u}_{\min}(k)\\ \mathbf{I}_{H_pm}\cdot\Delta\mathbf{u}_{\min}(k)\end{bmatrix}$$

where $\mathbf{I}_{\mathbf{H_p}}\mathbf{m}$ is an $(H_p\cdot m\times H_p\cdot m)$ identity matrix. The output level constraints (11) can be derived through the prediction (16)

$$\begin{bmatrix}-\mathbf{R}_u\\ \mathbf{R}_u\end{bmatrix}\Delta\mathbf{u}\leq\begin{bmatrix}-\mathbf{y}_{\min}(k)+\mathbf{R}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k)\\ \mathbf{y}_{\max}(k)-\mathbf{R}_x\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k).\end{bmatrix}$$

$$\mathbf{dR}_x=\begin{bmatrix}\mathbf{C}(k+1)\mathbf{A}(k)-\mathbf{C}(k)\\ \mathbf{C}(k+2)\mathbf{A}(k+1)\mathbf{A}(k)-\mathbf{C}(k+1)\mathbf{A}(k)\\ \vdots\\ \mathbf{C}(k+H_p)\prod_{i=H_p-1}^{0}\mathbf{A}(k+i)-\mathbf{C}(k+H_p-1)\prod_{i=H_p-2}^{0}\mathbf{A}(k+i)\end{bmatrix}\tag{31}$$

$$\mathbf{dR}_u=\begin{bmatrix}\mathbf{C}(k+2)\mathbf{A}(k+1)\mathbf{B}(k)-\mathbf{C}(k+1)\mathbf{B}(k) & \dots\\ \mathbf{C}(k+3)\mathbf{A}(k+2)\mathbf{A}(k+1)\mathbf{B}(k)-\mathbf{C}(k+2)\mathbf{A}(k+1)\mathbf{B}(k) & \dots\\ \vdots & \ddots\\ \mathbf{C}(k+H_p)\prod_{i=H_p-1}^{1}\mathbf{A}(k+i)\mathbf{B}(k)-\mathbf{C}(k+H_p-1)\prod_{i=H_p-2}^{1}\mathbf{A}(k+i)\mathbf{B}(k) & \dots\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{C}(k+H_p)\prod_{i=H_p-1}^{H_c+1}\mathbf{A}(k+i)\mathbf{B}(k+H_c)-\mathbf{C}(k+H_p-1)\prod_{i=H_p-2}^{H_c+1}\mathbf{A}(k+i)\mathbf{B}(k+H_c)\end{bmatrix}\tag{32}$$

$$\mathbf{R}_{u2}=\begin{bmatrix}\mathbf{0} & \dots & \mathbf{0}\\ \mathbf{C}(k+1)\mathbf{B}(k) & \dots & \mathbf{0}\\ \mathbf{C}(k+2)\mathbf{A}(k+1)\mathbf{B}(k) & \dots & \mathbf{0}\\ \vdots & \ddots & \vdots\\ \mathbf{C}(k+H_p-1)\prod_{i=H_p-2}^{1}\mathbf{A}(k+i)\mathbf{B}(k) & \dots & \mathbf{C}(k+H_p-1)\prod_{i=H_p-H_c-1}^{1}\mathbf{A}(k+i)\mathbf{B}(k+H_c-1)\end{bmatrix}.\tag{33}$$

The output rate constraints (11) are also derived through (16), although the expression is more involved. Since $\mathbf{y}(k)$ is not predicted but measured, for the prediction of $\Delta\hat{\mathbf{y}}(k+1)$ we have

$$\begin{bmatrix} -\mathbf{dR}_{u1} \\ \mathbf{dR}_{u1} \end{bmatrix} \Delta\mathbf{u} \leq \begin{bmatrix} -\mathbf{y}_{\min}(k) + \mathbf{dR}_{x1}\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \\ \mathbf{y}_{\max}(k) - \mathbf{dR}_{x1}\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \end{bmatrix}$$

where the matrices $\mathbf{dR}_{x1}$ and $\mathbf{dR}_{u1}$ are defined as $\mathbf{dR}_{x1} = \mathbf{C}(k)$ and $\mathbf{dR}_{u1} = \mathbf{C}(k)\mathbf{B}(k) - \mathbf{C}(k)$. The constraints for the predicted outputs $\Delta\hat{\mathbf{y}}(k+2), \ldots, \Delta\hat{\mathbf{y}}(k+H_p)$ are

$$\begin{bmatrix} -\mathbf{dR}_{u} \\ \mathbf{dR}_{u} \end{bmatrix} \Delta\mathbf{u} \leq \begin{bmatrix} -\mathbf{y}_{\min}(k) + \mathbf{dR}_{x}\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \\ \mathbf{y}_{\max}(k) - \mathbf{dR}_{x}\mathbf{A}(k)\mathbf{x}_{\text{lin}}(k) \end{bmatrix}$$

where $\mathbf{dR}_x$ and $\mathbf{dR}_u$ are given by (31) and (32), respectively. The matrix $\mathbf{R}_{u2}$ used in (19) is defined in (33) and $\bar{\mathbf{P}}$, $\Delta\bar{\mathbf{P}}$, $\bar{\mathbf{Q}}$ and $\Delta\bar{\mathbf{Q}}$ in (19) are block-diagonal matrices of suitable dimensions, where the diagonal blocks are respectively the weights $\mathbf{P}_i$, $\Delta\mathbf{P}_i$, $\mathbf{Q}_j$ and $\Delta\mathbf{Q}_j$ in the cost function (10); see (31)–(33), as shown at the bottom of the previous page.

REFERENCES

[1] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 116–132, Jan. 1985.

[2] R. R. Yager and D. P. Filev, *Essentials of Fuzzy Modeling and Control*. New York: Wiley, 1994.

[3] J. Zhao, V. Wertz, and R. Gorez, "A fuzzy clustering method for the identification of fuzzy models for dynamical systems," in *Proc. 1994 IEEE Int. Symp. Intelligent Control*, Columbus, OH, Aug. 1994, pp. 172–177.

[4] R. Babuška, J. A. Roubos, and H. B. Verbruggen, "Identification of mimo systems by input-output TS fuzzy models," in *Proc. FUZZ-IEEE'98*, Anchorage, AK, 1998, pp. 657–662.

[5] D. Saez and A. Cipriano, "Design of fuzzy model based predictive controllers and its application to an inverted pendulum," in *Proc. 6th IEEE Int. Conf. Fuzzy Systems*, vol. 2, Barcelona, Spain, 1997, pp. 915–919.

[6] B. K. Kavsek, I. Skrjanc, and D. Matko, "Fuzzy predictive control of a highly nonlinear pH process," *Comput. Chem. Eng.*, vol. 21, pp. S613–S618, 1997.

[7] M. Fischer, O. Nelles, and R. Isermann, "Adaptive predictive control of a heat exchanger based on a fuzzy model," *Control Eng. Practice*, vol. 6, no. 2, pp. 259–269, 1998.

[8] ——, "Predictive control based on local linear fuzzy models," *Int. J. Syst. Sci.*, vol. 29, no. 7, pp. 679–697, 1998.

[9] J. Q. Hu and T. Rose, "Generalized predictive control using a neuro-fuzzy model," *Int. J. Syst. Sci.*, vol. 30, no. 1, pp. 117–122, 1999.

[10] J. Abonyi, L. Nagy, and F. Szeifert, "Fuzzy model based predictive control by instantaneous linearization," *Fuzzy Sets Syst.*, vol. 120, no. 1, pp. 109–122, 2001.

[11] J. J. Espinosa, M. L. Hadjili, V. Wertz, and J. Vandewalle, "Predictive control using fuzzy models—comparative study," in *Proc. Eur. Control Conf.*, 1999, p. F0547.

[12] H. N. Nounou and K. M. Passino, "Fuzzy model predictive control: techniques, stability issues, and examples," in *Proc. 1999 IEEE Int. Symp. Intelligent Control Intelligent Systems and Semiotics*, 1999, pp. 423–428.

[13] V. J. de Oliveira and J. M. Lemos, "A comparison of some adaptive-predictive fuzzy-control strategies," *IEEE Trans. Syst., Man, Cybern. C*, vol. 30, pp. 138–145, Feb. 2000.

[14] M. J. D. Powell, "A fast algorithm for nonlinearly constrained optimization calculations," in *Numerical Analysis*, G. A. Watson, Ed. New York: Springer-Verlag, 1978, vol. 630, Lecture Notes in Mathematics, pp. 144–157.

[15] B. N. Pshenichnyj, *The Linearization Method for Constrained Optimization*. Berlin, Germany: Springer-Verlag, 1994.

[16] O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive controller (feasibility implies stability)," *IEEE Trans. Automat. Contr.*, vol. 44, pp. 648–654, Mar. 1999.

[17] S. Mollov, T. J. J. van den Boom, F. Cuesta, A. Ollero, and R. Babuška, "Robust stability constraints for fuzzy model predictive control," *IEEE Trans. Fuzzy Syst.*, vol. 10, pp. 50–64, Feb. 2002.

[18] T. A. Johansen, R. Shorten, and R. Murray-Smith, "On the interpretation and identification of dynamic Takagi-Sugeno fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 297–313, June 2000.

[19] J. Abonyi and R. Babuška, "Local and global identification and interpretation of parameters in Takagi-Sugeno fuzzy models," in *Proc. FUZZ-IEEE*, San Antonio, TX, 2000, pp. 835–840.

[20] S. P. Boyd, C. Crusius, and A. Hansson, "Control applications of nonlinear convex programming," *J. Process Control*, vol. 8, no. 5–6, pp. 313–324, 1998.

[21] H. Kuhn and A. W. Tucker, "Nonlinear programming," presented at the 2nd Berkeley Symp. Mathematical Statistics and Probability, Berkeley, CA, 1951.

[22] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[23] M. J. D. Powell, "Variable metric methods for constrained optimization," in *Mathematical Programming: The State of the Art*, A. Bachem, M. Grotschel, and B. Korte, Eds. New York: Springer-Verlag, 1983, pp. 288–311.

[24] C. G. Economou, M. Morari, and B. O. Palsson, "Internal model control: extension to nonlinear systems," *Ind. Eng. Chem. Process Des. Dev.*, vol. 25, pp. 403–411, 1986.

[25] T. J. McAvoy, E. Hsu, and S. Lowenthal, "Dynamics of pH in controlled stirred tank reactor," *Ind. Eng. Chem Process Des. Develop.*, vol. 11, no. 1, pp. 68–70, 1972.

[26] R. Babuška, *Fuzzy Modeling for Control*. Norwell, MA: Kluwer, 1998.

[27] T. Coleman, M. A. Branch, and A. Grace, *Optimization Toolbox for Use With MATLAB*. Natick, MA: The MathWorks, Inc., 1999.

[28] S. Skogestad, "Dynamics and control of distillation columns. A tutorial introduction," *Chem. Eng. Res. Des. (Trans. IChemE)*, vol. 75, pp. 539–562, 1997.

[29] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control*. Chichester, U.K.: Wiley, 1996.

[30] C. T. Chou and M. Verhaegen, "An indirect approach to closed-loop identification of Wiener models," in *Proc. Amer. Control Conf.*, San Diego, CA, 1999, pp. 3456–2461.

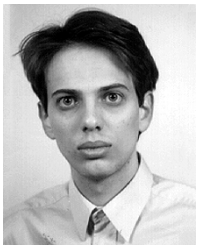
Stanimir Mollov received the M.Sc. degree in control engineering from the Technical University of Sofia, Sofia, Bulgaria, in 1990, and the Ph.D. degree from the Delft University of Technology, Delft, The Netherlands, in 2002.

Currently, he is with FCS Control Systems B.V., Schiphol, The Netherlands. His main research interests include multivariable (fuzzy) process identification and control.

Robert Babuška received the M.Sc. degree in control engineering from the Czech Technical University (CTU), Prague, Czech Republic, in 1990 and the Ph.D. degree from the Delft University of Technology, Delft, The Netherlands, in 1997.

Currently, he is a Professor with the Delft Center for Systems and Control, Faculty of Mechanical Engineering, Delft University of Technology. He has coauthored more than 50 journal papers and chapters in books, and has published a research monograph *Fuzzy Modeling for Control* (Norwell, MA: Kluwer, 1998). His research interests include the use of fuzzy set techniques and neural networks in nonlinear system identification and control.

Dr. Babuška is serving as an Associate Editor for the IEEE TRANSACTIONS ON FUZZY SYSTEMS and *Engineering Applications of Artificial Intelligence*, and as an Area Editor of *Fuzzy Sets and Systems*.



Janos Abonyi received the M.Eng. and Ph.D. degrees in chemical engineering from the University of Veszprem, Veszprem, Hungary, in 1997 and 2000, respectively.

Currently, he is an Assistant Professor with the Department of Process Engineering of the University of Veszprem. From 1999 to 2000, he was a Research Fellow at the Control Laboratory at Delft University of Technology, Delft, The Netherlands. His research interests include the applications of fuzzy systems, genetic algorithms, and neural networks in process engineering and data-mining. He has authored or coauthored the research monograph *Fuzzy Model Identification for Control* (Boston, MA: Birkhäuser, 2003), and more than 30 journal papers and chapters in books in these areas.

Henk B. Verbruggen received the M.Sc. degree in electrical engineering from the Delft University of Technology, Delft, The Netherlands, in 1963.

Since 1963, he has been a Staff Member with the Control Engineering Laboratory at the Electrical Engineering Department, Delft University of Technology. In 1980, he was appointed a Full Professor. His research interests include model-based predictive control, fuzzy logic and neural networks for modeling, control, fault detection, and controller reconfiguration. He is the author and coauthor of more than 200 publications. He has served as Chairman of the Coordinating Committee on Computer Control of IFAC, as an Associate Editor of the IFAC journal *Engineering Applications of AI*, and as Area Editor of *Fuzzy Sets and Systems*. He has been involved in a number of EC-sponsored research projects and working groups.