ISA Transactions 55 (2015) 49–62

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

# Nonlinear predictive control for Hammerstein–Wiener systems

Maciej Ławryńczuk *

*Institute of Control and Computation Engineering, Faculty of Electronics and Information Technology, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland*

ARTICLE INFO

*Article history:*
Received 20 April 2014
Received in revised form
12 September 2014
Accepted 15 September 2014
Available online 23 October 2014
This paper was recommended for publication by Dr. Rickey Dubay

*Keywords:*
Process control
Model Predictive Control
Hammerstein–Wiener systems
Optimisation
Linearisation

ABSTRACT

This paper discusses a nonlinear Model Predictive Control (MPC) algorithm for multiple-input multiple-output dynamic systems represented by cascade Hammerstein–Wiener models. The block-oriented Hammerstein–Wiener model, which consists of a linear dynamic block embedded between two nonlinear steady-state blocks, may be successfully used to describe numerous processes. A direct application of such a model for prediction in MPC results in a nonlinear optimisation problem which must be solved at each sampling instant on-line. To reduce the computational burden, a linear approximation of the predicted system trajectory linearised along the future control scenario is successively found on-line and used for prediction. Thanks to linearisation, the presented algorithm needs only quadratic optimisation, time-consuming and difficult on-line nonlinear optimisation is not necessary. In contrast to some control approaches for cascade models, the presented algorithm does not need inverse of the steady-state blocks of the model. For two benchmark systems, it is demonstrated that the algorithm gives control accuracy very similar to that obtained in the MPC approach with nonlinear optimisation while performance of linear MPC and MPC with simplified linearisation is much worse.

© 2014 ISA. Published by Elsevier Ltd. All rights reserved.

## 1. Introduction

The principal idea of Model Predictive Control (MPC) is to use on-line a dynamic model of the controlled process to predict its future behaviour and to repeatedly calculate on-line the control scenario from an optimisation problem in which the discrepancy between the predicted system trajectory and the desired set-point trajectory over some time horizon is minimised [9,29,40,46]. Many variants of the MPC technique have been successfully used for years in numerous advanced industrial applications [39]. There are a few reasons for that. Firstly, the MPC algorithms make it possible to impose constraints on process inputs (manipulated variables) and outputs (controlled variables) or state variables in a natural and efficient manner. Secondly, the general idea of MPC is very universal which means that the MPC algorithms can be successfully used for multiple-input multiple-output processes and for processes with difficult dynamic properties. Although MPC algorithms are particularly frequently used to provide good control of distillation columns, e.g. [16], and chemical reactors, e.g. [41,52], they may be also used in numerous other example applications to control: highway traffic [4], active queue management in TCP/IP networks [5], a high energy accelerator [6], a combustion engine [12], a flexible manipulator [14], a plastic injection moulding process [15], a greenhouse [20], a fuel cell [21], a robot [34], cruise control in cars [42], an air conditioning system [43], a cement kiln [44], an electric arc furnace [49], a boiler drum [51], and a baker's yeast drying process [53].

In the simplest case linear models are used for prediction in MPC. Such an approach frequently gives quite good results. In fact, the classical linear MPC algorithms turn out in practice to make it possible to significantly increase control accuracy when compared with the PID controllers, in particular in the case of multiple-input multiple-output processes. The majority of technological processes, however, are inherently nonlinear. In such cases the classical linear MPC algorithms may not work properly, i.e. for different changes of the set-point the process behaviour may be too slow, too fast, it may have unwanted overshoot or the controlled system may be unstable. That is why in such cases nonlinear MPC algorithms which use for prediction nonlinear models must be used [22,30,45,46]. A viable alternative to black-box models [33,38], e.g. neural networks of different kinds, is to use block-oriented cascade models which are composed of linear dynamic parts and nonlinear steady-state parts [24,25]. Such cascade models have the following advantages:

(a) good approximation accuracy, they are straightforward for many technological processes,
(b) ease of model development,

* Tel.: +48 22 234 76 73; fax: +48 22 825 37 19.
*E-mail address:* M.Lawrynczuk@ia.pw.edu.pl

http://dx.doi.org/10.1016/j.isatra.2014.09.018
0019-0578/© 2014 ISA. Published by Elsevier Ltd. All rights reserved.

(c) possibility to incorporate a priori process knowledge,
(d) possibility of being used for control.

Hammerstein and Wiener models are the most known and the most widely used block-oriented structures: in the first case the nonlinear steady-state part precedes the linear dynamic one, in the second case the connection order is reversed. The more general Hammerstein–Wiener structure consists of a linear dynamic block embedded between two nonlinear steady-state blocks.

The Hammerstein–Wiener structure may be successfully used to describe numerous processes, e.g. the human's muscle [1], a continuous stirred tank reactor [23], a micro-scale polymerase chain reaction reactor [26], temperature variations in a silage bale [31], a DC motor [32], a neutralisation reactor [35], a photovoltaic system [36] and even runtime management of quality of service performance and resource provisioning in shared resource software environments [37].

For identification of Hammerstein–Wiener systems a number of algorithms have been developed, e.g. the blind approach which allows a very general structure of the nonlinear steady-state blocks [2], an optimal two-stage identification algorithm [3], a two-stage recursive identification algorithm [8], an output error method for the identification of nonparametric models [11], an iterative scheme which finds nonparametric models [18], an identification algorithm in which parameters of the linear and nonlinear parts are found from a regression problem [19], an algorithm which makes it possible to easily use a priori process knowledge [28], an iterative estimation method based on the estimates of internal variables [47], an extended stochastic gradient algorithm [48], a subspace identification method for closed-loop systems [50], and a relaxation iteration scheme [54].

The Hammerstein–Wiener systems may be used for control, in particular in MPC. Because a direct application of a nonlinear model for prediction in MPC results in a nonlinear optimisation problem which must be solved at each sampling instant on-line, the predominant solution is to use in control algorithms inverse steady-state blocks which compensate for the process nonlinearity. Such an approach is only possible when the inverse blocks exist. Furthermore, as pointed out in [7], in such a case the influence of nonlinearity on the input–output behaviour of the process may be not taken into account by the controller. An extended Internal Model Control (IMC) control scheme for the Hammerstein–Wiener systems is presented in [10]. The IMC algorithm uses the inverse blocks. Similarly, the inverse blocks are used in the MPC algorithms for the Hammerstein–Wiener systems described in [13,23,37]. An alternative approach is presented in [7], where the nonlinearities are transformed into polytopic descriptions which means that the Hammerstein–Wiener model is represented as a linear model with uncertainty and it leads to a convex optimisation problem subject to linear matrix inequalities.

This paper describes a nonlinear MPC algorithm for multiple-input multiple-output dynamic systems represented by cascade Hammerstein–Wiener models. In contrast to the aforementioned MPC approaches, the algorithm does not need inverse of the steady-state blocks of the model. In order to obtain a computationally efficient solution, a linear approximation of the predicted system trajectory linearised along the future control scenario is successively found on-line and used for prediction. Linearisation of the trajectory makes it possible to formulate a quadratic optimisation MPC problem which is solved on-line to find the future control scenario, time-consuming and difficult on-line nonlinear optimisation is not necessary. This paper extends the work of [27], where the MPC algorithm with trajectory linearisation based on the single-input single-output Wiener system with a specific nonlinear part is described. Two benchmark systems are taken into account to show advantages of the presented algorithm. In particular, the algorithm is compared with the linear MPC strategy, with the MPC algorithm for cascade systems in which model linearisation is performed in a simplified manner for the current operating point of the process and with the MPC approach with nonlinear optimisation. It is demonstrated that the algorithm gives control accuracy very similar to that obtained in the MPC scheme with nonlinear optimisation.

This paper is organised as follows. Section 2 reminds the idea of MPC and Section 3 describes the structure of the Hammerstein–Wiener model. The main part of the paper, given in Section 4, discusses the MPC algorithm with on-line trajectory linearisation for Hammerstein–Wiener systems. Section 5 presents simulation results for two benchmark systems. Finally, Section 6 concludes the paper.

## 2. Model Predictive Control problem formulation

The process under consideration has $n_u$ inputs and $n_y$ outputs. The input vector is $u=\left[u_1 \ldots u_{n_u}\right]^T$, the output vector is $y=\left[y_1 \ldots y_{n_y}\right]^T$.

In MPC algorithms [9,29,40,46] at each consecutive sampling instant $k$ a set of future control increments is calculated

$$\triangle \boldsymbol{u}(k)=\begin{bmatrix}\triangle u(k|k)\\ \vdots \\ \triangle u(k+N_u-1|k)\end{bmatrix}\in \mathbb{R}^{n_u N_u} \tag{1}$$

where $N_u$ is the control horizon and the increments are defined by

$$\triangle u(k+p|k)=\begin{cases} u(k|k)-u(k-1) & \text{if } p=0\\ u(k+p|k)-u(k+p-1|k) & \text{if } p\geq 1\end{cases}$$

The control signals for the sampling instant $k+p$ found at the current instant $k$ are denoted by $u(k+p|k)$. It is assumed that $\triangle u(k+p|k)=0$ for $p\geq N_u$. The objective of the algorithm is to minimise differences between the set-point trajectory $y^{sp}(k+p|k)$ and predicted output values $\hat{y}(k+p|k)$ over the prediction horizon $N\geq N_u$ and to penalise excessive control increments. Hence, the following quadratic cost-function is usually used:

$$\begin{aligned} J(k)&=\sum_{p=1}^{N}\sum_{m=1}^{n_y}\mu_{p,m}\left(y_m^{sp}(k+p|k)-\hat{y}_m(k+p|k)\right)^2\\ &\quad+\sum_{p=0}^{N_u-1}\sum_{n=1}^{n_u}\lambda_{p,n}\left(\triangle u_n(k+p|k)\right)^2\\ &=\sum_{p=1}^{N}\|y^{sp}(k+p|k)-\hat{y}(k+p|k)\|^2_{\boldsymbol{M}_p}+\sum_{p=0}^{N_u-1}\|\triangle u(k+p|k)\|^2_{\boldsymbol{\Lambda}_p}\end{aligned} \tag{2}$$

where the weighting coefficients are $\mu_{p,m}\geq 0$ and $\lambda_{p,n}>0$, which comprise the matrices $\boldsymbol{M}_p\geq 0$ and $\boldsymbol{\Lambda}_p>0$ of dimensionality $n_y\times n_y$ and $n_u\times n_u$, respectively. The cost-function (2) is minimised on-line. As a result, future control increments (1) are calculated. Only the first $n_u$ elements of the determined sequence are applied to the process, i.e. $u(k)=\triangle u(k|k)+u(k-1)$. At the next sampling instant, $k+1$, output measurements are updated, the prediction is shifted one step forward and the whole procedure is repeated. For prediction, i.e. to calculate the quantities $\hat{y}(k+1|k),\ldots,\hat{y}(k+N|k)$, a dynamic model of the process is used. The problem of tuning MPC algorithms, i.e. adjusting parameters $\mu_{p,m}$, $\lambda_{p,n}$, $N$, $N_u$, is discussed elsewhere [9,29,46].

In real control systems there always exist constraints imposed on the input variables, which result from the physical limits of actuators. Additionally, it is sometimes necessary to take into account the constraints of the predicted output variables, which may be motivated by quality requirements or enforced by set-

point optimisation. In such cases the future control increments $\triangle \boldsymbol{u}(k)$ are found on-line from the following optimisation problem:

$$\min_{\triangle \boldsymbol{u}(k)} \left\{ J(k) = \sum_{p=1}^{N} \| y^{sp}(k+p|k) - \hat{y}(k+p|k) \|^2_{\boldsymbol{M}_p} + \sum_{p=0}^{N_u-1} \| \triangle u(k+p|k) \|^2_{\boldsymbol{\Lambda}_p} \right\}$$

subject to

$$\begin{aligned} & u^{min} \le u(k+p|k) \le u^{max}, \quad p=0,\ldots,N_u-1 \\ & -\triangle u^{max} \le \triangle u(k+p|k) \le \triangle u^{max}, \quad p=0,\ldots,N_u-1 \\ & y^{min} \le \hat{y}(k+p|k) \le y^{max}, \quad p=1,\ldots,N \end{aligned} \tag{3}$$

where $u^{min} \in \mathbb{R}^{n_u}$, $u^{max} \in \mathbb{R}^{n_u}$, $\triangle u^{max} \in \mathbb{R}^{n_u}$, $y^{min} \in \mathbb{R}^{n_y}$, $y^{max} \in \mathbb{R}^{n_y}$ define constraints imposed on the magnitude of the input variables, on the increment of the input variables and on the magnitude of the output variables, respectively. If the constraints of the predicted output variables are taken into account, the feasible set of the MPC optimisation problem (3) may be empty. In order to cope with such a situation output constraints have to be softened by means of slack variables [29,46]. It is assumed that the predicted values of the output variables may temporarily violate the original hard constraints $y^{min} \le \hat{y}(k+p|k) \le y^{max}$, which enforces the existence of the feasible set. The MPC optimisation problem with soft output constraints is

$$\min_{\substack{\triangle \boldsymbol{u}(k) \\ \varepsilon^{min}(k+p) \\ \varepsilon^{max}(k+p)}} \left\{ J(k) = \sum_{p=1}^{N} \| y^{sp}(k+p|k) - \hat{y}(k+p|k) \|^2_{\boldsymbol{M}_p} + \sum_{p=0}^{N_u-1} \| \triangle u(k+p|k) \|^2_{\boldsymbol{\Lambda}_p} + \rho^{min} \sum_{p=1}^{N} \| \varepsilon^{min}(k+p) \|^2 + \rho^{max} \sum_{p=1}^{N} \| \varepsilon^{max}(k+p) \|^2 \right\}$$

subject to

$$\begin{aligned} & u^{min} \le u(k+p|k) \le u^{max}, \quad p=0,\ldots,N_u-1 \\ & -\triangle u^{max} \le \triangle u(k+p|k) \le \triangle u^{max}, \quad p=0,\ldots,N_u-1 \\ & y^{min} - \varepsilon^{min}(k+p) \le \hat{y}(k+p|k) \le y^{max} + \varepsilon^{max}(k+p), \quad p=1,\ldots,N \\ & \varepsilon^{min}(k+p) \ge 0, \quad \varepsilon^{max}(k+p) \ge 0, \ p=1,\ldots,N \end{aligned} \tag{4}$$

where the additional decision variables of the MPC optimisation problem which determine the degree of constraint violation are denoted by the vectors $\varepsilon^{min}(k+p) \in \mathbb{R}^{n_y}$ and $\varepsilon^{max}(k+p) \in \mathbb{R}^{n_y}$, $\rho^{min}$ and $\rho^{max} > 0$ are penalty coefficients.

## 3. Hammerstein–Wiener model of the process

The structure of the multiple-input multiple-output Hammerstein–Wiener model with $n_u$ inputs and $n_y$ outputs is depicted in Fig. 1. The auxiliary signals between the input steady-state nonlinear part and the linear dynamic one are denoted by $v_1(k),\ldots,v_{n_v}(k)$, the auxiliary signals between the linear dynamic part and the output steady-state nonlinear one are denoted by $x_1(k),\ldots,x_{n_x}(k)$. As many as $n_v$ independent input steady-state parts are used, each of them has $n_u$ inputs and one output. Similarly, as many as $n_y$ independent output steady-state parts are used, each of them has $n_x$ inputs and one output. The input nonlinear steady-state parts are described by

$$v_n(k) = g_n(u(k)) = g_n(u_1(k),\ldots,u_{n_u}(k)) \tag{5}$$

for $n=1,\ldots,n_v$ whereas the output steady-state parts are characterised by

$$y_m(k) = h_m(x(k)) = h_m(x_1(k),\ldots,x_{n_x}(k)) \tag{6}$$

for $m=1,\ldots,n_y$. It is assumed that the functions $g_n : \mathbb{R}^{n_u} \to \mathbb{R}$ and $h_m : \mathbb{R}^{n_x} \to \mathbb{R}$ are differentiable. The linear part of the model is described by

$$\boldsymbol{A}(q^{-1})x(k) = \boldsymbol{B}(q^{-1})v(k) \tag{7}$$

where $x(k) = [x_1(k)\ldots x_{n_x}(k)]^T$, $v(k) = [v_1(k)\ldots v_{n_v}(k)]^T$, the entries of the matrices

$$\boldsymbol{A}(q^{-1}) = \begin{bmatrix} A_{1,1}(q^{-1}) & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & A_{n_x,n_x}(q^{-1}) \end{bmatrix} \tag{8}$$

and

$$\boldsymbol{B}(q^{-1}) = \begin{bmatrix} B_{1,1}(q^{-1}) & \ldots & B_{1,n_v}(q^{-1}) \\ \vdots & \ddots & \vdots \\ B_{n_x,1}(q^{-1}) & \ldots & B_{n_x,n_v}(q^{-1}) \end{bmatrix} \tag{9}$$

are the following polynomials in the backward shift operator $q^{-1}$

$$A_{i,i}(q^{-1}) = 1 + a^i_1 q^{-1} + \cdots + a^i_{n_A} q^{-n_A} \tag{10}$$

for $i=1,\ldots,n_x$ and

$$B_{i,j}(q^{-1}) = b^{i,j}_1 q^{-1} + \cdots + b^{i,j}_{n_B} q^{-n_B} \tag{11}$$

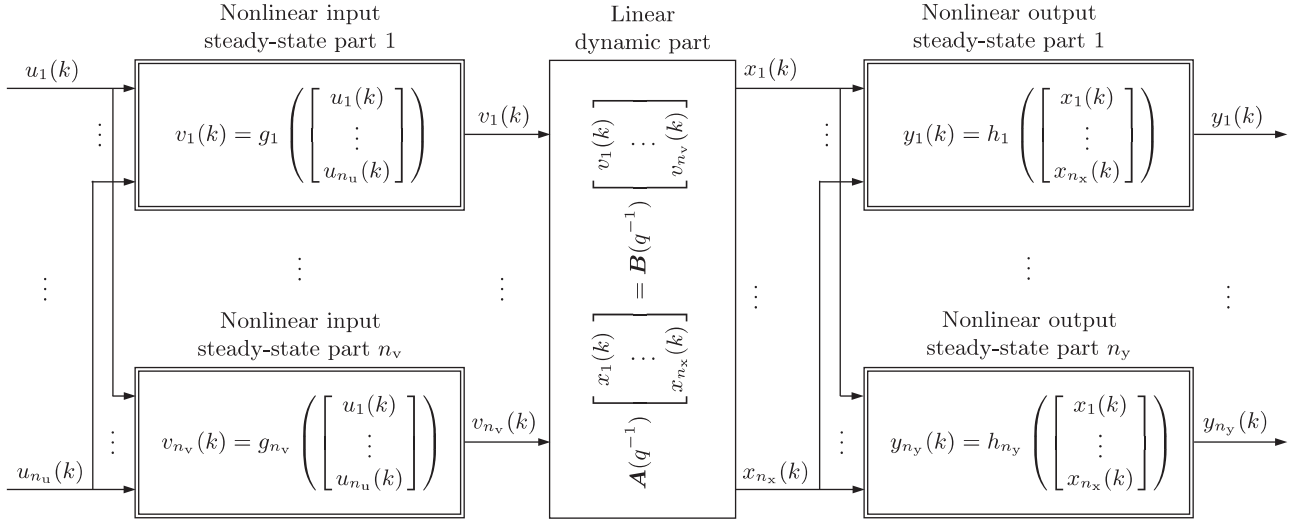


**Fig. 1.** The structure of the multiple-input multiple-output Hammerstein–Wiener model.

for $i=1,\ldots,n_x$, $j=1,\ldots,n_v$. From Eqs. (7) to (11), the $n$th output of the dynamic part of the model is

$$x_n(k)=\sum_{s=1}^{n_v}\sum_{l=1}^{n_B} b_l^{n,s}v_s(k-l)-\sum_{l=1}^{n_A} a_l^n x_n(k-l) \tag{12}$$

Using the input nonlinear steady-state relations (5), one has

$$x_n(k)=\sum_{s=1}^{n_v}\sum_{l=1}^{n_B} b_l^{n,s}g_s(u_1(k-l),\ldots,u_{n_u}(k-l))-\sum_{l=1}^{n_A} a_l^n x_n(k-l) \tag{13}$$

The consecutive outputs of the Hammerstein–Wiener model may be calculated from Eqs. (6) and (13).

## 4. MPC algorithm with trajectory linearisation for Hammerstein–Wiener systems

If the Hammerstein–Wiener model is directly used in MPC, the predictions of the output variables, i.e. the signals $\hat{y}_m(k+p|k)$ for all outputs (i.e. for $m=1,\ldots,n_y$) and over the whole prediction horizon (i.e. for $p=1,\ldots,N$), are nonlinear functions of the decision variables of the algorithm, i.e. the calculated on-line future control increments $\triangle\boldsymbol{u}(k)$. As a result, the MPC optimisation problem (3) or (4) is a nonlinear task which must be solved on-line in real time. Such an approach is referred to in this paper as the MPC algorithm with Nonlinear Optimisation (MPC-NO). The MPC-NO algorithm is usually computationally demanding and it may terminate at a shallow local minimum, which has a very negative effect on control quality.

In order to reduce computational complexity of MPC, the inverse nonlinear steady-state blocks may be used to compensate for the process nonlinearity. It is only possible when the inverse blocks exist. In practice, however, nonlinear steady-state process characteristics are frequently characterised by saturations, which are not invertible. Furthermore, from the perspective of input–output dynamic process behaviour, nonlinearity cancellation may be not fully possible as pointed out in [7]. An alternative straightforward and quite simple method to relax computational burden of MPC is to estimate the gains of the steady-state parts of the model for the current operating point of the process. Next, the linear part of the model with modified gains is used for prediction. In such a case simplified on-line model linearisation is carried out, because not the full nonlinear dynamic model is linearised but only the linear approximations of the steady-state blocks are found. The second disadvantage of such an approach is the fact that the same linearised model is used for prediction over the whole prediction horizon, which may result in insufficient control quality.

In this paper the MPC algorithm with Nonlinear Prediction and Linearisation along the Predicted Trajectory (MPC-NPLPT) for the Hammerstein–Wiener systems is discussed. Trajectory linearisation is conceptually a better method when compared with simplified on-line model linearisation and much more advanced than a fairly naive attempt to remove process nonlinearity by means of inverse steady-state blocks. The MPC-NPLPT algorithm for single-input single-output Wiener systems with a specific nonlinear part is discussed in [27] and the simulation results presented therein are characterised by very good control quality. This paper extends the work of [27] for the general case of the multiple-input multiple-output Hammerstein–Wiener systems.

At each sampling instant (main iteration) of the MPC-NPLPT algorithm a linear approximation of the predicted output trajectory is calculated on-line and used for optimisation of the future control policy $\triangle\boldsymbol{u}(k)$, which, thanks to linearisation, is simple to solve quadratic optimisation problem. Linearisation is carried out not for the current operating point of the process (which is usually defined by some present and past values of the process input and output variables), but along some future control scenario. In the MPC-NPLPT algorithm trajectory linearisation and optimisation of the control scenario are repeated a few times at each sampling instant in the internal iterations. In the internal iteration $t$ the predicted nonlinear output trajectory

$$\hat{\boldsymbol{y}}^t(k)=\begin{bmatrix}\hat{y}^t(k+1|k)\\ \vdots\\ \hat{y}^t(k+N|k)\end{bmatrix}\in\mathbb{R}^{n_yN}$$

is linearised along the input trajectory

$$\boldsymbol{u}^{t-1}(k)=\begin{bmatrix}u^{t-1}(k|k)\\ \vdots\\ u^{t-1}(k+N_u-1|k)\end{bmatrix}\in\mathbb{R}^{n_uN_u}$$

found in the previous internal iteration $(t-1)$. Using Taylor's series expansion method, a linear approximation of the nonlinear output trajectory $\hat{\boldsymbol{y}}^t(k)$ along the input trajectory $\boldsymbol{u}^{t-1}(k)$, i.e. linearisation of the function $\hat{\boldsymbol{y}}^t(\boldsymbol{u}^t(k)):\mathbb{R}^{n_uN_u}\to\mathbb{R}^{n_yN}$ is

$$\hat{\boldsymbol{y}}^t(k)=\hat{\boldsymbol{y}}^{t-1}(k)+\boldsymbol{H}^t(k)(\boldsymbol{u}^t(k)-\boldsymbol{u}^{t-1}(k)) \tag{14}$$

The control values $\boldsymbol{u}^t(k)$ calculated at the current internal iteration $t$ are the arguments of the function $\hat{\boldsymbol{y}}^t(k)$. The predicted output trajectory corresponding to the input trajectory $\boldsymbol{u}^{t-1}(k)$ is denoted by $\hat{\boldsymbol{y}}^{t-1}(k)$. The matrix of the derivatives of the predicted output trajectory with respect of the future control signals is of dimensionality $n_yN\times n_uN_u$ and it has the general structure

$$\begin{aligned}\boldsymbol{H}^t(k)&=\left.\frac{\mathrm{d}\hat{\boldsymbol{y}}(k)}{\mathrm{d}\boldsymbol{u}(k)}\right|_{\substack{\hat{\boldsymbol{y}}(k)=\hat{\boldsymbol{y}}^{t-1}(k)\\ \boldsymbol{u}(k)=\boldsymbol{u}^{t-1}(k)}}=\frac{\mathrm{d}\hat{\boldsymbol{y}}^{t-1}(k)}{\mathrm{d}\boldsymbol{u}^{t-1}(k)}\\ &=\begin{bmatrix}\frac{\partial\hat{y}^{t-1}(k+1|k)}{\partial u^{t-1}(k|k)} & \cdots & \frac{\partial\hat{y}^{t-1}(k+1|k)}{\partial u^{t-1}(k+N_u-1|k)}\\ \vdots & \ddots & \vdots\\ \frac{\partial\hat{y}^{t-1}(k+N|k)}{\partial u^{t-1}(k|k)} & \cdots & \frac{\partial\hat{y}^{t-1}(k+N|k)}{\partial u^{t-1}(k+N_u-1|k)}\end{bmatrix}\end{aligned} \tag{15}$$

where the matrices

$$\frac{\partial\hat{\boldsymbol{y}}^{t-1}(k+p|k)}{\partial\boldsymbol{u}^{t-1}(k+r|k)}=\begin{bmatrix}\frac{\partial\hat{y}_1^{t-1}(k+p|k)}{\partial u_1^{t-1}(k+r|k)} & \cdots & \frac{\partial\hat{y}_1^{t-1}(k+p|k)}{\partial u_{n_u}^{t-1}(k+r|k)}\\ \vdots & \ddots & \vdots\\ \frac{\partial\hat{y}_{n_y}^{t-1}(k+p|k)}{\partial u_1^{t-1}(k+r|k)} & \cdots & \frac{\partial\hat{y}_{n_y}^{t-1}(k+p|k)}{\partial u_{n_u}^{t-1}(k+r|k)}\end{bmatrix} \tag{16}$$

of dimensionality $n_y\times n_u$ are calculated for all $p=1,\ldots,N$ and $r=0,\ldots,N_u-1$.

Using the relation

$$\boldsymbol{u}^t(k)=\boldsymbol{J}\triangle\boldsymbol{u}^t(k)+\boldsymbol{u}(k-1)$$

where the vector

$$\boldsymbol{u}(k-1)=\begin{bmatrix}u(k-1)\\ \vdots\\ u(k-1)\end{bmatrix}$$

is of length $n_uN_u$, the matrix

$$\boldsymbol{J}=\begin{bmatrix}\boldsymbol{I}_{n_u\times n_u} & \boldsymbol{0}_{n_u\times n_u} & \boldsymbol{0}_{n_u\times n_u} & \cdots & \boldsymbol{0}_{n_u\times n_u}\\ \boldsymbol{I}_{n_u\times n_u} & \boldsymbol{I}_{n_u\times n_u} & \boldsymbol{0}_{n_u\times n_u} & \cdots & \boldsymbol{0}_{n_u\times n_u}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{I}_{n_u\times n_u} & \boldsymbol{I}_{n_u\times n_u} & \boldsymbol{I}_{n_u\times n_u} & \cdots & \boldsymbol{I}_{n_u\times n_u}\end{bmatrix}$$

is of dimensionality $n_uN_u\times n_uN_u$ ($\boldsymbol{I}_{n_u\times n_u}$ and $\boldsymbol{0}_{n_u\times n_u}$ stand for unitary and zeros matrices, respectively, of dimensionality $n_u\times n_u$), the linear approximation of the nonlinear predicted output trajectory (14) becomes

$$\hat{\boldsymbol{y}}^t(k)=\boldsymbol{H}^t(k)\boldsymbol{J}\triangle\boldsymbol{u}^t(k)+\hat{\boldsymbol{y}}^{t-1}(k)+\boldsymbol{H}^t(k)(\boldsymbol{u}(k-1)-\boldsymbol{u}^{t-1}(k)) \tag{17}$$

Thanks to linearisation, the predicted output trajectory $\hat{\boldsymbol{y}}^t(k)$ is a linear function of the future control increments $\triangle\boldsymbol{u}^t(k)$ calculated for the current sampling instant $k$ and at the internal iteration $t$ of the MPC-NPLPT algorithm. Using the prediction equation (17), the optimisation problem (4) becomes the following quadratic programming task:

$$\begin{aligned}\min_{\substack{\triangle\boldsymbol{u}^t(k)\\ \boldsymbol{\varepsilon}^{\min}(k)\\ \boldsymbol{\varepsilon}^{\max}(k)}} \quad & \{J(k)=\|\boldsymbol{y}^{sp}(k)-\boldsymbol{H}^t(k)\boldsymbol{J}\triangle\boldsymbol{u}^t(k)\\ & -\hat{\boldsymbol{y}}^{t-1}(k)-\boldsymbol{H}^t(k)(\boldsymbol{u}(k-1)-\boldsymbol{u}^{t-1}(k))\|^2\\ & +\|\triangle\boldsymbol{u}^t(k)\|^2_{\boldsymbol{\Lambda}}+\rho^{\min}\|\boldsymbol{\varepsilon}^{\min}(k)\|^2+\rho^{\max}\|\boldsymbol{\varepsilon}^{\max}(k)\|^2\}\\ & \text{subject to}\\ & \boldsymbol{u}^{\min}\le\boldsymbol{J}\triangle\boldsymbol{u}^t(k)+\boldsymbol{u}(k-1)\le\boldsymbol{u}^{\max}\\ & -\triangle\boldsymbol{u}^{\max}\le\triangle\boldsymbol{u}^t(k)\le\triangle\boldsymbol{u}^{\max}\\ & \boldsymbol{y}^{\min}-\boldsymbol{\varepsilon}^{\min}(k)\le\boldsymbol{H}^t(k)\boldsymbol{J}\triangle\boldsymbol{u}^t(k)+\hat{\boldsymbol{y}}^{t-1}(k)\\ & +\boldsymbol{H}^t(k)(\boldsymbol{u}(k-1)-\boldsymbol{u}^{t-1}(k))\le\boldsymbol{y}^{\max}+\boldsymbol{\varepsilon}^{\max}(k)\\ & \boldsymbol{\varepsilon}^{\min}(k)\ge 0,\quad \boldsymbol{\varepsilon}^{\max}(k)\ge 0\end{aligned} \tag{18}$$

where the set-point trajectory

$$\boldsymbol{y}^{sp}(k)=\begin{bmatrix}y^{sp}(k+1|k)\\ \vdots\\ y^{sp}(k+N|k)\end{bmatrix}$$

is the vector of length $n_yN$, the additional variable vectors for the output constraints softening

$$\boldsymbol{\varepsilon}^{\min}(k)=\begin{bmatrix}\varepsilon^{\min}(k+1)\\ \vdots\\ \varepsilon^{\min}(k+N)\end{bmatrix},\quad \boldsymbol{\varepsilon}^{\max}(k)=\begin{bmatrix}\varepsilon^{\max}(k+1)\\ \vdots\\ \varepsilon^{\max}(k+N)\end{bmatrix}$$

are also of length $n_yN$ (where the consecutive subvectors $\varepsilon^{\max}(k+p)$ and $\varepsilon^{\max}(k+p)$ are of length $n_y$), the input constraint vectors

$$\boldsymbol{u}^{\min}=\begin{bmatrix}u^{\min}\\ \vdots\\ u^{\min}\end{bmatrix},\quad \boldsymbol{u}^{\max}=\begin{bmatrix}u^{\max}\\ \vdots\\ u^{\max}\end{bmatrix},\quad \triangle\boldsymbol{u}^{\max}=\begin{bmatrix}\triangle u^{\max}\\ \vdots\\ \triangle u^{\max}\end{bmatrix}$$

are of length $n_uN_u$, the output constraint vectors

$$\boldsymbol{y}^{\min}=\begin{bmatrix}y^{\min}\\ \vdots\\ y^{\min}\end{bmatrix},\quad \boldsymbol{y}^{\max}=\begin{bmatrix}y^{\max}\\ \vdots\\ y^{\max}\end{bmatrix}$$

are of length $n_yN$, the weighting matrices $\boldsymbol{M}=\mathrm{diag}(\boldsymbol{M}_1,\ldots,\boldsymbol{M}_N)\ge 0$ and $\boldsymbol{\Lambda}=\mathrm{diag}(\boldsymbol{\Lambda}_0,\ldots,\boldsymbol{\Lambda}_{N_u-1})>0$ are of dimensionality $n_yN\times n_yN$ and $n_uN_u\times n_uN_u$, respectively.

The Hammerstein–Wiener dynamic model is used to predict future behaviour of the process, i.e. to calculate the predicted output trajectory $\hat{\boldsymbol{y}}^t(k)$, which is composed of the predicted output values $\hat{y}_m(k+p|k)$ for all outputs, i.e. for $m=1,\ldots,n_y$, and over the whole prediction horizon, i.e. for $p=1,\ldots,N$. The general prediction equation is

$$\hat{y}^t_m(k+p|k)=y^t_m(k+p|k)+d_m(k) \tag{19}$$

where the quantities $y^t_m(k+p|k)$ are the model outputs for the sampling instant $k+p$ calculated at the current instant $k$ and $d_m(k)$ are estimations of the unmeasured disturbances. The prediction equation (19) makes it possible to compensate for the actual disturbances which affects the process and for the unavoidable model inaccuracy [46]. The disturbances are estimated from

$$d_m(k)=y_m(k)-y_m(k|k-1)$$

where the signals $y_m(k)$ are measured process outputs and the quantities $y_m(k|k-1)$ are calculated from the model using measurements up to the previous sampling instant $(k-1)$. From Eq. (6), the disturbance estimations are

$$d_m(k)=y_m(k)-h_m(x_1(k),\ldots,x_{n_x}(k)) \tag{20}$$

From Eqs. (6) and (19), the predicted value of the $m$th output for the sampling instant $k+p$ calculated at the current instant $k$ and at the current internal iteration $t$ is

$$\hat{y}^t_m(k+p|k)=h_m(x^t_1(k+p|k),\ldots,x^t_{n_x}(k+p|k))+d_m(k) \tag{21}$$

From Eq. (12), the predicted signals between the linear dynamic part and the output nonlinear steady-state one are

$$\begin{aligned}x^t_n(k+p|k)=\sum_{s=1}^{n_v}\Bigg[&\sum_{l=1}^{I_{uf}(p)}b^{n,s}_l v^t_s(k-l+p|k)+\sum_{l=I_{uf}(p)+1}^{n_B}b^{n,s}_l v_s(k-l+p)\\ &-\sum_{l=1}^{I_{yp}(p)}a^n_l x^t_n(k-l+p|k)-\sum_{l=I_{yp}(p)+1}^{n_A}a^n_l x_n(k-l+p)\Bigg]\end{aligned} \tag{22}$$

where $I_{uf}(p)=\min(p,n_B)$, $I_{yf}(p)=\min(p-1,n_A)$. From Eq. (5), the predicted signals between the input nonlinear steady-state part and the linear dynamic one are

$$v^t_s(k-l+p|k)=g_s(u^t_1(k-l+p|k),\ldots,u^t_{n_u}(k-l+p|k)) \tag{23}$$

Similarly, for the past one has

$$v_s(k-l+p)=g_s(u_1(k-l+p),\ldots,u_{n_u}(k-l+p)) \tag{24}$$

From Eq. (13), it is also possible to have

$$\begin{aligned}x^t_n(k+p|k)=&\sum_{s=1}^{n_v}\sum_{l=1}^{n_B}b^{n,s}_l g_s(u^t_1(k-l+p|k),\ldots,u^t_{n_u}(k-l+p|k))\\ &-\sum_{l=1}^{n_A}a^n_l x^t_n(k-l+p|k)\end{aligned} \tag{25}$$

The entries of the matrix (16), i.e. the derivatives of the predicted trajectory for the previous internal iteration with respect to the future control scenario are calculated from Eq. (21), which gives

$$\frac{\partial y^{t-1}_m(k+p|k)}{\partial u^{t-1}_i(k+r|k)}=\frac{\partial h_m(x^{t-1}_1(k+p|k),\ldots,x^{t-1}_{n_x}(k+p|k))}{\partial u^{t-1}_i(k+r|k)} \tag{26}$$

During derivatives' calculations in Eq. (26) the specific nature of the nonlinear output steady-state functions is taken into account. From Eq. (22), the derivatives of the predicted signals between the linear dynamic part and the steady-state output nonlinear one depend only on the future, i.e.

$$\frac{\partial x^{t-1}_n(k+p|k)}{\partial u^{t-1}_i(k+r|k)}=\sum_{s=1}^{n_v}\left[\sum_{l=1}^{I_{uf}(p)}b^{n,s}_l\frac{\partial v^{t-1}_s(k-l+p|k)}{\partial u^{t-1}_i(k+r|k)}-\sum_{l=1}^{I_{yp}(p)}a^n_l\frac{\partial x^{t-1}_n(k-l+p|k)}{\partial u^{t-1}_i(k+r|k)}\right] \tag{27}$$

where from Eq. (23), one has

$$\frac{\partial v^{t-1}_s(k+p|k)}{\partial u^{t-1}_i(k+r|k)}=\frac{\partial g_s(u^{t-1}_1(k+p|k),\ldots,u^{t-1}_{n_u}(k+p|k))}{\partial u^{t-1}_i(k+r|k)} \tag{28}$$

During derivatives' calculations in Eq. (28) the specific nature of the nonlinear input steady-state functions is taken into account.

If the process is close to the desired set-point or the changes of the set-point are not significant, only one internal iteration is carried out ($t^{\max}=1$). In different words, the predicted output trajectory is linearised only once along some future input scenario and only one quadratic programming problem is solved at each sampling instant (the main iteration of the algorithm). For linearisation $n_u(N_u-1)$ elements of the future control sequence calculated at the previous sampling instant are used

$$\boldsymbol{u}^0(k)=\begin{bmatrix}u^0(k|k)\\ \vdots\\ u^0(k+N_u-3|k)\\ u^0(k+N_u-2|k)\\ u^0(k+N_u-1|k)\end{bmatrix}=\begin{bmatrix}u(k|k-1)\\ \vdots\\ u(k+N_u-3|k-1)\\ u(k+N_u-2|k-1)\\ u(k+N_u-2|k-1)\end{bmatrix} \tag{29}$$

The quantities $u(k+N_u-2|k-1)$ are repeated twice because the first $n_u$ elements of the optimal control sequence calculated at the previous sampling instant, i.e. $u(k-1|k-1)$, cannot be used at the current sampling instant $k$ because they are actually applied to the process at the previous instant $k-1$. Internal iterations are continued if

$$\sum_{p=0}^{N_0} \| y^{sp}(k-p)-y(k-p) \|^2 \geq \triangle_y \tag{30}$$

where $N_0$ is a time horizon and the value of $\triangle_y$ is chosen experimentally (the smaller the $\triangle_y$, the bigger the chance of starting internal iterations). If the difference between future control increments calculated in consecutive internal iterations is small, it means when

$$\| \triangle \boldsymbol{u}^t(k)-\triangle \boldsymbol{u}^{t-1}(k) \|^2 < \triangle_u \tag{31}$$

internal iterations are terminated, the first $n_u$ elements of the obtained control policy $\triangle \boldsymbol{u}^t(k)$ are applied to the process. The quantity $\triangle_u > 0$ is chosen experimentally (the smaller the $\triangle_u$, the more the internal iterations).

At each sampling instant $k$ of the MPC-NPLPT algorithm the following steps are repeated:

1. The unmeasured disturbances are estimated from Eq. (20).
2. The first internal iteration ($t=1$): the Hammerstein–Wiener model is used to find from Eqs. (21) to (25) the future output trajectory $\hat{\boldsymbol{y}}^0(k)$ which corresponds to the initial input trajectory $\boldsymbol{u}^0(k)$ defined by Eq. (29).
3. A linear approximation (17) of the output predicted trajectory $\hat{\boldsymbol{y}}^1(k)$ along the input trajectory $\boldsymbol{u}^0(k)$ is found using the Hammerstein–Wiener model, i.e. the entries of the matrix $\boldsymbol{H}^1(k)$ defined by Eq. (15) are obtained from Eqs. (26) to (28).
4. The MPC-NPLPT quadratic optimisation problem (18) is solved to find the future control increments $\triangle \boldsymbol{u}^1(k)$ in the first internal iteration.
5. If the condition (30) is satisfied, internal iterations are continued for $t=2,\ldots,t_{max}$.
   5.1. The Hammerstein–Wiener model is used to calculate the predicted output trajectory $\hat{\boldsymbol{y}}^{t-1}(k)$ corresponding to the future input trajectory $\boldsymbol{u}^{t-1}(k)=\boldsymbol{J}\triangle \boldsymbol{u}^{t-1}(k)+\boldsymbol{u}(k-1)$ from Eqs. (21) to (25).
   5.2. A linear approximation (17) of the predicted output trajectory $\hat{\boldsymbol{y}}^t(k)$ along the input trajectory $\boldsymbol{u}^{t-1}(k)$ is found using the Hammerstein–Wiener model, i.e. the entries of the matrix $\boldsymbol{H}^t(k)$ defined by Eq. (15) are obtained from Eqs. (26) to (28).
   5.3. The MPC-NPLPT quadratic optimisation problem (18) is solved to find the future control increments $\triangle \boldsymbol{u}^t(k)$ in the current internal iteration $t$.
   5.4. If the condition (31) is satisfied or $t > t_{max}$, internal iterations are terminated. Otherwise, the internal iteration index is increased ($t:=t+1$), the algorithm goes to step 5.1.
6. The first $n_u$ elements of the determined sequence $\triangle \boldsymbol{u}^t(k)$ are applied to the process, i.e. $u(k)=\triangle u^t(k|k)+u(k-1)$.
7. The main iteration index of the algorithm is increased, i.e. $k:=k+1$, at the next sampling instant the algorithm goes to step 1.

## 5. Simulation results

In this section predictive control of two benchmark Hammerstein–Wiener systems is discussed. Both benchmarks are frequently used for comparing efficiency of identification algorithms and control schemes. The following MPC algorithms are compared:

(a) The classical linear MPC algorithm (the Generalised Predictive Control algorithm [46]) based on a linear model corresponding to the nominal operating point of the process. The algorithm requires on-line quadratic optimisation.
(b) The MPC algorithm with Nonlinear Prediction and Simplified Linearisation (MPC-NPSL) in which the gains of the steady-state parts of the model for the current operating point of the process are estimated and they are used to modify the gains of the linear dynamic part whereas the nonlinear Hammerstein–Wiener model is used to determine the reaction to the past (the nonlinear free trajectory). The rudiments of the MPC-NPSL algorithm for the Wiener system is discussed in [27]. The algorithm requires on-line quadratic optimisation.
(c) The MPC algorithm with Nonlinear Prediction along the Trajectory (MPC-NPLT) in which at each sampling instant the predicted output trajectory linearisation and quadratic optimisation are carried out only once (only one internal iteration is used). For linearisation $n_u(N_u-1)$ elements of the future control sequence calculated at the previous sampling instant are used, the trajectory $\boldsymbol{u}^0(k)$ is defined by Eq. (29). The algorithm requires on-line quadratic optimisation.
(d) The discussed MPC-NPLPT algorithm with trajectory linearisation repeated at each sampling instant (in the internal iterations). At each sampling instant the algorithm requires on-line solution of one or maximally $t^{max}$ quadratic optimisation problems.
(e) The "ideal" MPC algorithm with Nonlinear Optimisation (MPC-NO). For prediction the full nonlinear model is used. At each sampling instant the algorithm requires on-line nonlinear optimisation.

It is necessary to emphasise the fact that all four nonlinear MPC algorithms use on-line the same nonlinear Hammerstein–Wiener model. For prediction a linear approximation of the nonlinear model is used in the MPC-NPSL algorithm, a linear approximation of the nonlinear trajectory is used in the MPC-NPLT and MPC-NPLPT algorithms whereas the full model without any simplifications is used in the MPC-NO strategy. Tuning parameters of all algorithms are the same and the same set-point trajectories are considered (separate for two benchmark systems). All Simulations are carried out in Matlab.

### 5.1. Example 1

The first process under consideration is a single-input single-output Hammerstein–Wiener system ($n_u=n_y=1$), all three parts of the model have one input and one output ($n_v=n_x=1$) [54]. The first (input) nonlinear steady-state part is described by

$$v(k)=g(u(k))=\frac{u(k)}{\sqrt{0.1+0.9u^2(k)}} \tag{32}$$

The second (output) nonlinear steady-state part is

$$y(k)=h(x(k))=x(k)+0.2x^3(k) \tag{33}$$

Fig. 2 depicts input and output nonlinear steady-state characteristics for the operation range $u^{min}=-2.5$, $u^{max}=2.5$. The linear part is described by

$$(1-1.5q^{-1}+0.7q^{-2})x(k)=(0.5q^{-1}+0.25q^{-2})v(k) \tag{34}$$

i.e. $n_A=n_B=2$, $a_1=-1.5$, $a_2=0.7$, $b_1=0.5$, $b_2=0.25$. From Eq. (34), the output of the linear dynamic part is

$$x(k)=b_1v(k-1)+b_2v(k-2)-a_1x(k-1)-a_2x(k-2)$$

Implementation details of the discussed MPC-NPLPT algorithm depends on the specific nature of the nonlinear input and output

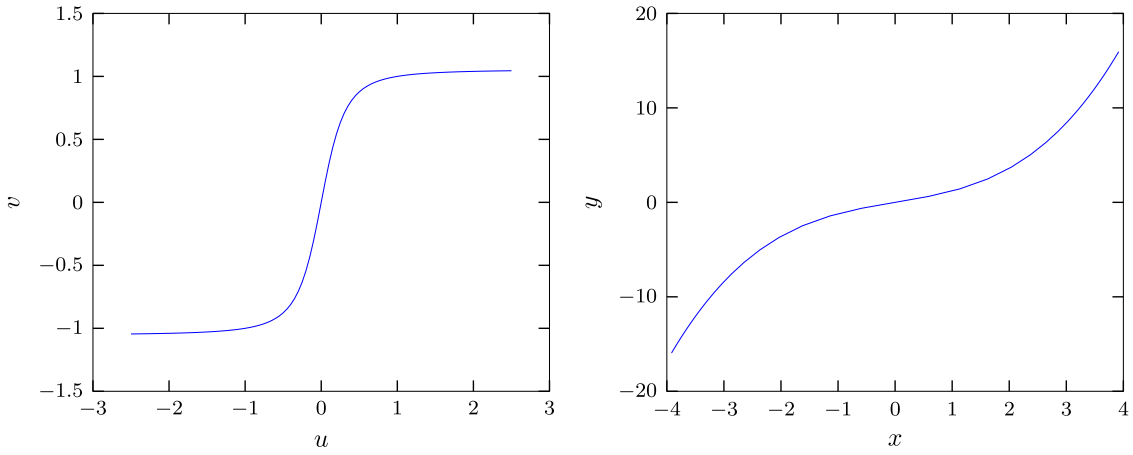


**Fig. 2.** Example 1: characteristics $v=g(u)$ and $y=h(x)$ of the nonlinear steady-state blocks.

steady-state functions. From the general prediction equation (21) and for the specific output nonlinearity defined by Eq. (33), the predicted value of the output for the sampling instant $k+p$ calculated at the current instant $k$ is

$$\hat{y}^t(k+p|k)=h(x^t(k+p|k))+d(k)$$
$$=x^t(k+p|k)+0.2(x^t(k+p|k))^3+d(k)$$

From Eqs. (20) and (33), the unmeasured disturbance is estimated from

$$d(k)=y(k)-h(x(k))=y(k)-x(k)-0.2x^3(k)$$

where $y(k)$ denotes the current measurement of the output signal. From Eq. (22), the predicted signal between the linear dynamic part and the output nonlinear steady-state one is

$$x^t(k+p|k)=\sum_{l=1}^{I_{uf}(p)}b_l v^t(k-l+p|k)+\sum_{l=I_{uf}(p)+1}^{2}b_l v(k-l+p)$$
$$-\sum_{l=1}^{I_{yp}(p)}a_l x^t(k-l+p|k)-\sum_{l=I_{yp}(p)+1}^{2}a_l x(k-l+p)$$

where $I_{uf}(p)=\min(p,2)$, $I_{yf}(p)=\min(p-1,2)$. From Eqs. (23) and (24), the predicted and past signals between the input nonlinear steady-state part and the linear dynamic one are

$$v^t(k-l+p|k)=g(u^t(k-l+p|k))$$
$$v(k-l+p)=g(u(k-l+p))$$

The entries of the matrix (16), i.e. the derivatives of the predicted output trajectory for the previous internal iteration with respect to the future control scenario, are calculated from the general equation (26), which gives

$$\frac{\partial y^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}=\frac{\partial h(x^{t-1}(k+p|k))}{\partial u^{t-1}(k+r|k)}$$
$$=\frac{\mathrm{d}h(x^{t-1}(k+p|k))}{\mathrm{d}x^{t-1}(k+p|k)}\frac{\partial x^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}$$

For the given output steady-state nonlinear part defined by Eq. (33), one obtains

$$\frac{\mathrm{d}h(x^{t-1}(k+p|k))}{\mathrm{d}x^{t-1}(k+p|k)}=1+0.6(x^{t-1}(k+p|k))^2$$

The derivatives of the predicted signals between the linear dynamic part and the steady-state output nonlinear one are calculated from the general equation (27). For the single-input single-output process with $n_v=1$ one has

$$\frac{\partial x^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}=\sum_{l=1}^{I_{uf}(p)}b_l\frac{\partial v^{t-1}(k-l+p|k)}{\partial u^{t-1}(k+r|k)}-\sum_{l=1}^{I_{yp}(p)}a_l\frac{\partial x^{t-1}(k-l+p|k)}{\partial u^{t-1}(k+r|k)}$$

and the general equation (28) reduces to

$$\frac{\partial v^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}=\frac{\mathrm{d}g(u^{t-1}(k+p|k))}{\mathrm{d}u^{t-1}(k+r|k)}\frac{\partial u^{t-1}(k-l+p|k)}{\partial u^{t-1}(k+r|k)}$$

For the given input steady-state nonlinear part defined by Eq. (32), one obtains

$$\frac{\mathrm{d}g(u^{t-1}(k+p|k))}{\mathrm{d}u^{t-1}(k+r|k)}=(0.1+0.9(u^t(k+p|k))^2)^{-0.5}$$
$$+0.9(u^{t-1}(k+p|k))^2(0.1+0.9(u^{t-1}(k+p|k))^2)^{-1.5}$$

Because $u(k+p|k)=u(k+N_u-1|k)$ for $p\geq N_u$, one has

$$\frac{\partial u^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}=\begin{cases}1 & \text{if } p=r, p>rir=N_u-1\\ 0 & \text{otherwise}\end{cases}$$

Parameters of all compared MPC algorithms are the same: $N=10$, $N_u=3$, $\mu_p=1$ for $p=1,\ldots,N$, $\lambda_p=150$ for $p=1,\ldots,N_u$, constraints imposed on the manipulated variable are $u^{\min}=-2.5$, $u^{\max}=2.5$.

Fig. 3 depicts simulation results for an arbitrarily chosen set-point trajectory. The algorithm uses for prediction the linear model corresponding to the nominal operating point of the process ($u=y=0$). Because of significant nonlinear nature of the process, the obtained control quality is very low, i.e. the output trajectory is characterised by big overshoot and oscillations.

The following experiments are concerned with nonlinear MPC algorithms which use for prediction, although is different ways, the nonlinear Hammerstein–Wiener model. Simulation results of the MPC-NPSL algorithm with simplified on-line model linearisation and quadratic optimisation are shown in Fig. 4. Although the oscillations are reduced faster than in the case of the linear MPC algorithm, the overshoot is even bigger. Fig. 5 presents simulation results of the MPC-NPLT algorithm with on-line trajectory linearisation performed once at each sampling instant along the input trajectory $\boldsymbol{u}^0(k)$ defined by Eq. (29) and quadratic optimisation. When compared with the MPC-NPSL strategy, the algorithm has very small overshoot, but for the second and the fourth set-point change it is somehow slower. Finally, Fig. 6 compares simulation results obtained in the described MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation and in the MPC-NO algorithm with on-line nonlinear optimisation. The parameters of the MPC-NPLPT algorithm are $N_0=2$, $\triangle_u=\triangle_y=1$,

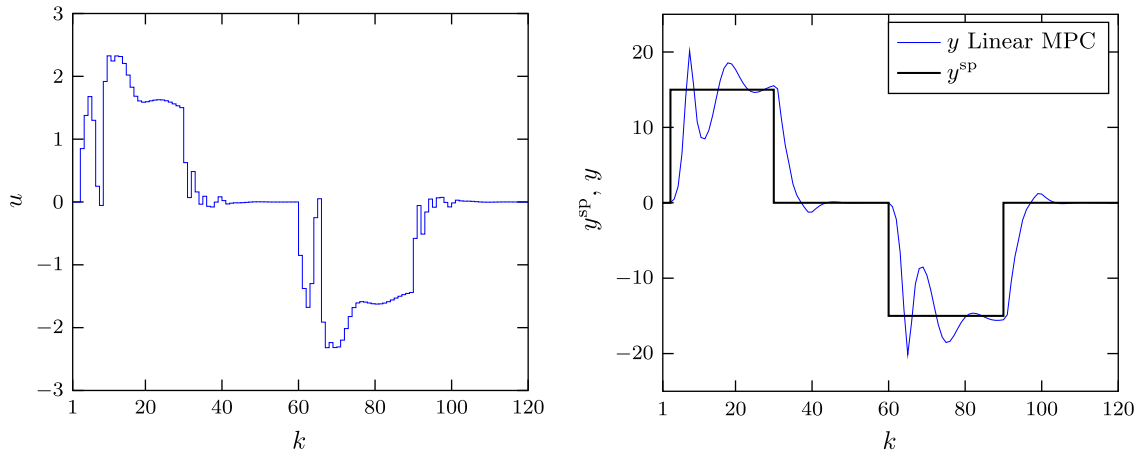


**Fig. 3.** Example 1: simulation results of the linear MPC algorithm.

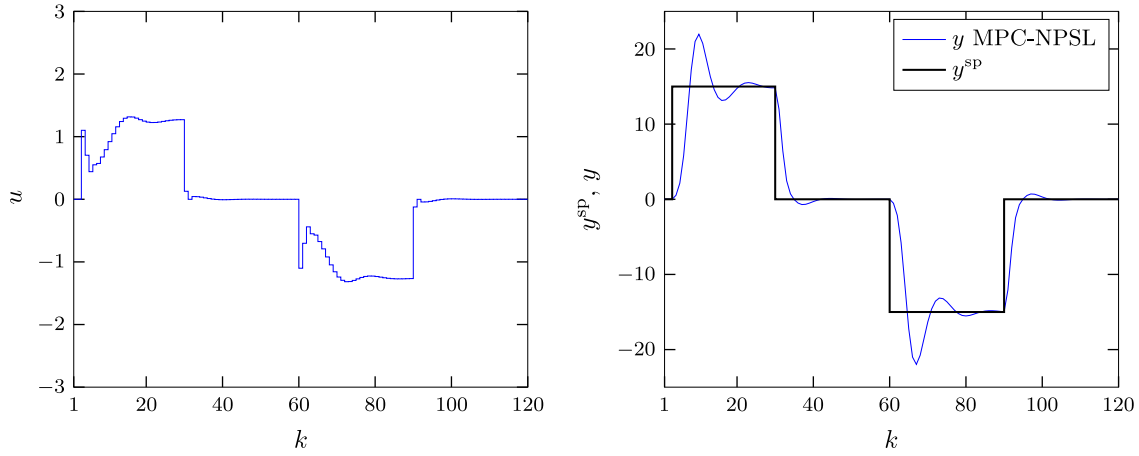


**Fig. 4.** Example 1: simulation results of the MPC-NPSL algorithm with simplified on-line model linearisation and quadratic optimisation.

$t^{\max}=5$. It is necessary to emphasise the fact that the MPC-NPLPT algorithm with on-line linearisation gives practically the same results as the computationally demanding MPC-NO strategy, the differences, even for big set-point changes, are very small. Table 1 compares control accuracy in terms of Sum of Squared Errors (SSE) defined by

$$\mathrm{SSE}=\sum_{k=1}^{120}\left(y^{\mathrm{sp}}(k)-y(k)\right)^2$$

and computational burden in terms of Millions of FLOating Point operationS (MFLOPS) of MPC algorithms, for the MPC-NPLPT strategy the sum of internal iterations is also given. Additionally, to demonstrate the role of tuning possibilities, the results for the MPC-NPLPT algorithm with parameters $\triangle_{\mathrm{u}}=\triangle_{\mathrm{y}}=0.5$ and $\triangle_{\mathrm{u}}=\triangle_{\mathrm{y}}=2$ are also given. Reduction of the tuning parameters $\triangle_{\mathrm{u}}$ and $\triangle_{\mathrm{y}}$ results in very small reduction of the SSE, but more internal iterations are necessary, which increases computational burden. Further reduction of the parameters $\triangle_{\mathrm{u}}$ and $\triangle_{\mathrm{y}}$ does not lead to any significant increase of control accuracy whereas their increase leads to worse control. Fig. 7 shows the number of internal iterations in consecutive sampling instants of three versions of the MPC-NPLPT algorithm. In the majority of internal iterations only one internal iteration is sufficient, more of them are necessary when the set-point changes. The smaller the parameters $\triangle_{\mathrm{u}}$ and $\triangle_{\mathrm{y}}$, the more the internal iterations.

### 5.2. Example 2

The second process under consideration is an input–output approximation of the state-space double-input double-output Hammerstein–Wiener system ($n_{\mathrm{u}}=n_{\mathrm{y}}=2$) with two signals $v_1, v_2$ and two signals $x_1, x_2$ ($n_{\mathrm{v}}=n_{\mathrm{x}}=2$) presented in [7]. The input nonlinear steady-state parts are described by

$$v_n(k)=g_n(u_n(k))=\frac{\exp(u_n(k))-1}{\exp(u_n(k))+1} \tag{35}$$

for $n=1,2$. The output nonlinear steady-state parts are

$$y_m(k)=h_m(x_m(k))=-\exp(-x_m(k))+1 \tag{36}$$

for $m=1,2$. Fig. 8 depicts input and output nonlinear steady-state characteristics for the operation range $u_i^{\min}=-5$, $u_i^{\max}=5$, $x_i^{\min}=-2.5$, $x_i^{\max}=2.5$ for $i=1,2$. The considered benchmark system can be associated with real applications since exponential functions in output steady-state parts are often used to approximate the non-linearity of high-purity distillation columns as emphasised in [17]. Moreover, the saturations used in the input steady-state parts are typical of actuators. The linear part of the model is described by

$$\begin{bmatrix} A_{1,1}(q^{-1}) & 0 \\ 0 & A_{2,2}(q^{-1}) \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}=\begin{bmatrix} B_{1,1}(q^{-1}) & B_{1,2}(q^{-1}) \\ B_{2,1}(q^{-1}) & B_{2,2}(q^{-1}) \end{bmatrix}\begin{bmatrix} v_1(k) \\ v_2(k) \end{bmatrix} \tag{37}$$

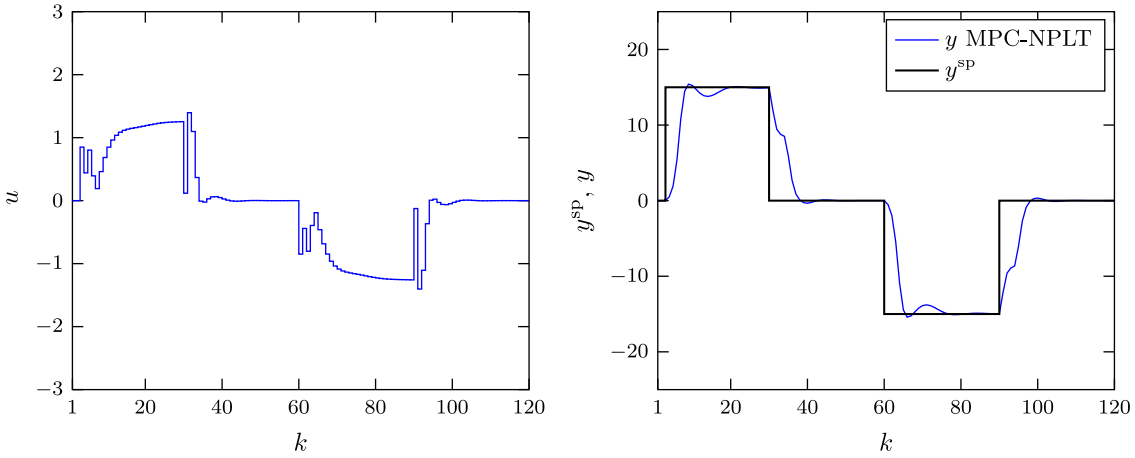


**Fig. 5.** Example 1: simulation results of the MPC-NPLT algorithm with on-line trajectory linearisation and quadratic optimisation.

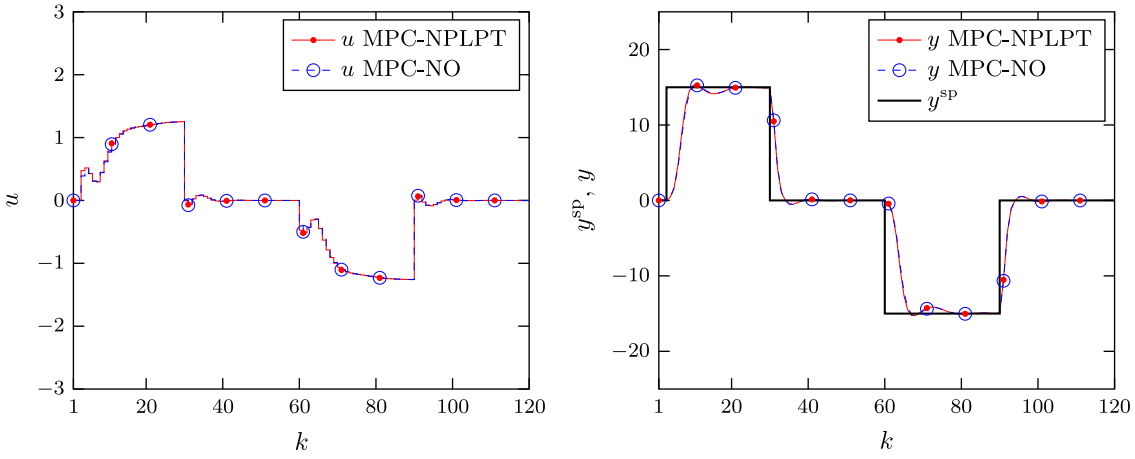


**Fig. 6.** Example 1: simulation results: the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation (*solid line with dots*), the MPC-NO algorithm with on-line nonlinear optimisation (*dashed line with circles*).

**Table 1**
Example 1: control accuracy (SSE) and computational burden (MFLOPS) of compared MPC algorithms, for the MPC-NPLPT strategy the sum of internal iterations is also given.

| Algorithm | SSE | MFLOPS | Sum of internal iterations |
|---|---|---|---|
| Linear MPC | $3.1243 \times 10^3$ | 0.1317 | – |
| MPC-NPSL | $2.5254 \times 10^3$ | 0.2125 | – |
| MPC-NPLT | $2.7507 \times 10^3$ | 0.3900 | – |
| MPC-NPLPT, $\delta_u = \delta_y = 0.5$ | $2.2811 \times 10^3$ | 0.5664 | 170 |
| MPC-NPLPT, $\delta_u = \delta_y = 1$ | $2.2814 \times 10^3$ | 0.5333 | 160 |
| MPC-NPLPT, $\delta_u = \delta_y = 2$ | $2.7794 \times 10^3$ | 0.5003 | 150 |
| MPC-NO | $2.2347 \times 10^3$ | 4.0832 | – |

where the polynomials $A_{i,i}(q^{-1})$ and $A_{i,i}(q^{-1})$ are of order 5 (i.e. $n_A = n_B = 5$)

$$A_{1,1}(q^{-1}) = 1 - 3.0119q^{-1} + 3.1433q^{-2} - 1.1429q^{-3} - 8.1583 \times 10^{-2}q^{-4} + 9.3456 \times 10^{-2}q^{-5}$$

$$A_{2,2}(q^{-1}) = 1 - 3.0129q^{-1} + 3.1142q^{-2} - 1.0465q^{-3} - 1.8082 \times 10^{-1}q^{-4} + 1.2656 \times 10^{-1}q^{-5}$$

and

$$B_{1,1}(q^{-1}) = -7.4786 \times 10^{-1}q^{-1} + 7.8402 \times 10^{-1}q^{-2} + 6.1410 \times 10^{-2}q^{-3} - 1.150031 \times 10^{-1}q^{-4} - 1.490090 \times 10^{-2}q^{-5}$$

$$B_{1,2}(q^{-1}) = 6.3866 \times 10^{-1}q^{-1} - 6.5725 \times 10^{-1}q^{-2} - 1.1910 \times 10^{-1}q^{-3} + 1.2811 \times 10^{-1}q^{-4} + 3.0992 \times 10^{-2}q^{-5}$$

$$B_{2,1}(q^{-1}) = 6.1061 \times 10^{-1}q^{-1} - 6.6366 \times 10^{-1}q^{-2} - 6.1431 \times 10^{-2}q^{-3} + 1.2374 \times 10^{-1}q^{-4} + 1.9351 \times 10^{-2}q^{-5}$$

$$B_{2,2}(q^{-1}) = -8.2348 \times 10^{-1}q^{-1} + 9.3053 \times 10^{-1}q^{-2} + 1.5599 \times 10^{-1}q^{-3} - 2.3120 \times 10^{-1}q^{-4} - 5.3329 \times 10^{-2}q^{-5}$$

From Eq. (37), the outputs of the linear dynamic part are

$$x_n(k) = \sum_{s=1}^{2} \sum_{l=1}^{5} b_l^{n,s} v_s(k-l) - \sum_{l=1}^{5} a_l^n x_n(k-l)$$

where $n = 1,2$.

From the general prediction equation (21) and for the specific output nonlinearities defined by Eq. (36), the predicted values of the outputs for the sampling instant $k+p$ calculated at the current instant $k$ are

$$\hat{y}_m^t(k+p|k) = h_m(x_m^t(k+p|k)) + d_m(k) = -\exp(-x_m^t(k+p|k)) + 1 + d_m(k)$$

for $m=1,2$. From Eqs. (20) and (36), the unmeasured disturbances are estimated from

$$d_m(k)=y_m(k)-h_m(x_m(k))=y_m(k)+\exp(-x_m(k))-1$$

where $y_m(k)$ denote the current measurements of the output signals, $m=1,2$. From Eq. (22), the predicted signals between the linear dynamic part and the output nonlinear steady-state one are

$$x_n^t(k+p|k)=\sum_{s=1}^{2}\left[\sum_{l=1}^{I_{\mathrm{uf}}(p)}b_l^{n,s}v_s^t(k-l+p|k)+\sum_{l=I_{\mathrm{uf}}(p)+1}^{5}b_l^{n,s}v_s(k-l+p)-\sum_{l=1}^{I_{\mathrm{yp}}(p)}a_l^n x_n^t(k-l+p|k)-\sum_{l=I_{\mathrm{yp}}(p)+1}^{5}a_l^n x_n(k-l+p)\right]$$

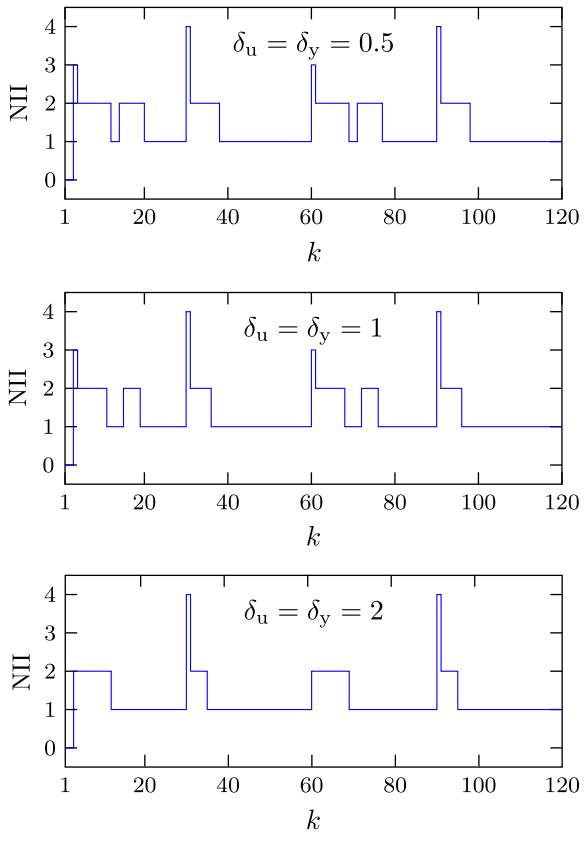


**Fig. 7.** Example 1: The number of internal iterations (NII) in consecutive sampling instants of three versions of the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation.

where $I_{\mathrm{uf}}(p)=\min(p,5)$, $I_{\mathrm{yf}}(p)=\min(p-1,5)$, $n=1,2$. From Eqs. (23) and (24), the predicted and past signals between the input nonlinear steady-state part and the linear dynamic one are

$$v_s^t(k-l+p|k)=g_s(u_s^t(k-l+p|k))$$
$$v_s(k-l+p)=g_s(u_s(k-l+p))$$

where $s=1,2$. The entries of the matrix (16) are calculated from the general equation (26)

$$\frac{\partial y_m^{t-1}(k+p|k)}{\partial u_i^{t-1}(k+r|k)}=\frac{\partial h_m(x_m^{t-1}(k+p|k))}{\partial u_i^{t-1}(k+r|k)}=\frac{\mathrm{d}h_m(x_m^{t-1}(k+p|k))}{\mathrm{d}x_m^{t-1}(k+p|k)}\frac{\partial x_m^{t-1}(k+p|k)}{\partial u_i^{t-1}(k+r|k)}$$

where $m=1,2$. For the given output steady-state nonlinear part defined by Eq. (36), one obtains

$$\frac{\mathrm{d}h_m(x_m^{t-1}(k+p|k))}{\mathrm{d}x_m^{t-1}(k+p|k)}=\exp(-x_m^{t-1}(k+p|k))$$

where $m=1,2$. The derivatives of the predicted signals between the linear dynamic part and the steady-state output nonlinear are calculated from the general equation (27). For the two-input two-output process with $n_v=2$ one has

$$\frac{\partial x_n^{t-1}(k+p|k)}{\partial u_i^{t-1}(k+r|k)}=\sum_{s=1}^{2}\left[\sum_{l=1}^{I_{\mathrm{uf}}(p)}b_l^{n,s}\frac{\partial v_s^{t-1}(k-l+p|k)}{\partial u_i^{t-1}(k+r|k)}-\sum_{l=1}^{I_{\mathrm{yp}}(p)}a_l^n\frac{\partial x_n^{t-1}(k-l+p|k)}{\partial u_i^{t-1}(k+r|k)}\right]$$

where $n=1,2$, $i=1,2$. From Eq. (28) one has

$$\frac{\partial v_n^{t-1}(k+p|k)}{\partial u_i^{t-1}(k+r|k)}=\frac{\mathrm{d}g_n(u_i^{t-1}(k+p|k))}{\mathrm{d}u_i^{t-1}(k+r|k)}\frac{\partial u_i^{t-1}(k-l+p|k)}{\partial u_i^{t-1}(k+r|k)}$$

where $n=1,2$, $i=1,2$. For the given input steady-state nonlinear part defined by Eq. (35), one obtains

$$\frac{\mathrm{d}g_n(u_i^{t-1}(k+p|k))}{\mathrm{d}u_i^{t-1}(k+r|k)}=\frac{2\exp(u_i^{t-1}(k+p|k))}{(\exp(u_i^{t-1}(k+p|k))+1)^2}$$

where $n=1,2$, $i=1,2$. Because $u_i(k+p|k)=u_i(k+N_u-1|k)$ for $p\geq N_u$, one has

$$\frac{\partial u_i^{t-1}(k+p|k)}{\partial u_i^{t-1}(k+r|k)}=\begin{cases}1 & \text{if } p=r, p>r \text{ i } r=N_u-1\\ 0 & \text{otherwise}\end{cases}$$

Parameters of all compared MPC algorithms are the same: $N=10$, $N_u=3$, $\mu_{p,m}=1$ for $m=1,2$, $p=1,\ldots,N$, $\lambda_{p,n}=0.01$ for $n=1,2$, $p=1,\ldots,N_u$. The same constraints as in [7] are imposed on the manipulated variable, i.e. $u_n^{\min}=-5$, $u_n^{\max}=5$ for $n=1,2$.

Fig. 9 depicts simulation results of the linear MPC algorithm for the set-point trajectory similar to the one used in [7].

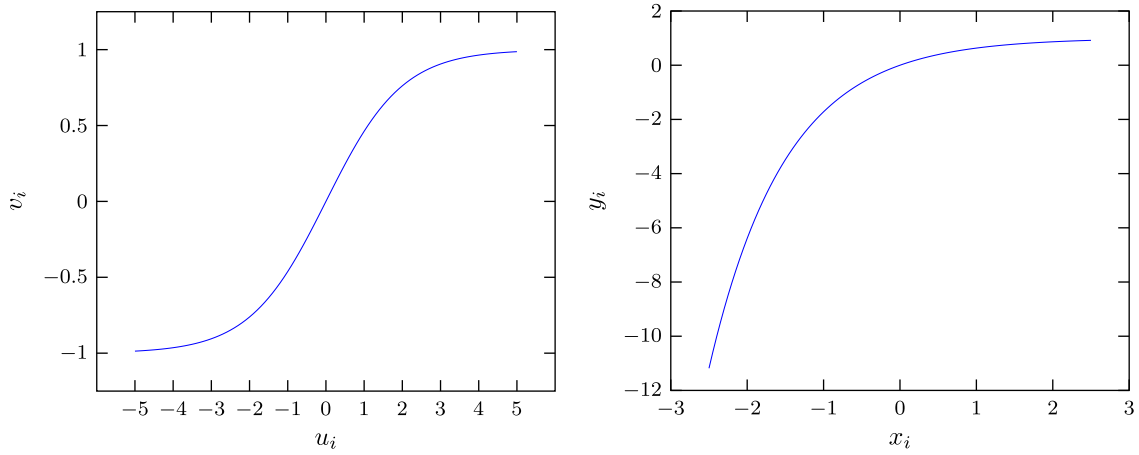


**Fig. 8.** Example 2: characteristics $v_i=g_i(u_i)$ and $y_i=h_i(x_i)$ of the nonlinear steady-state blocks, $i=1,2$.

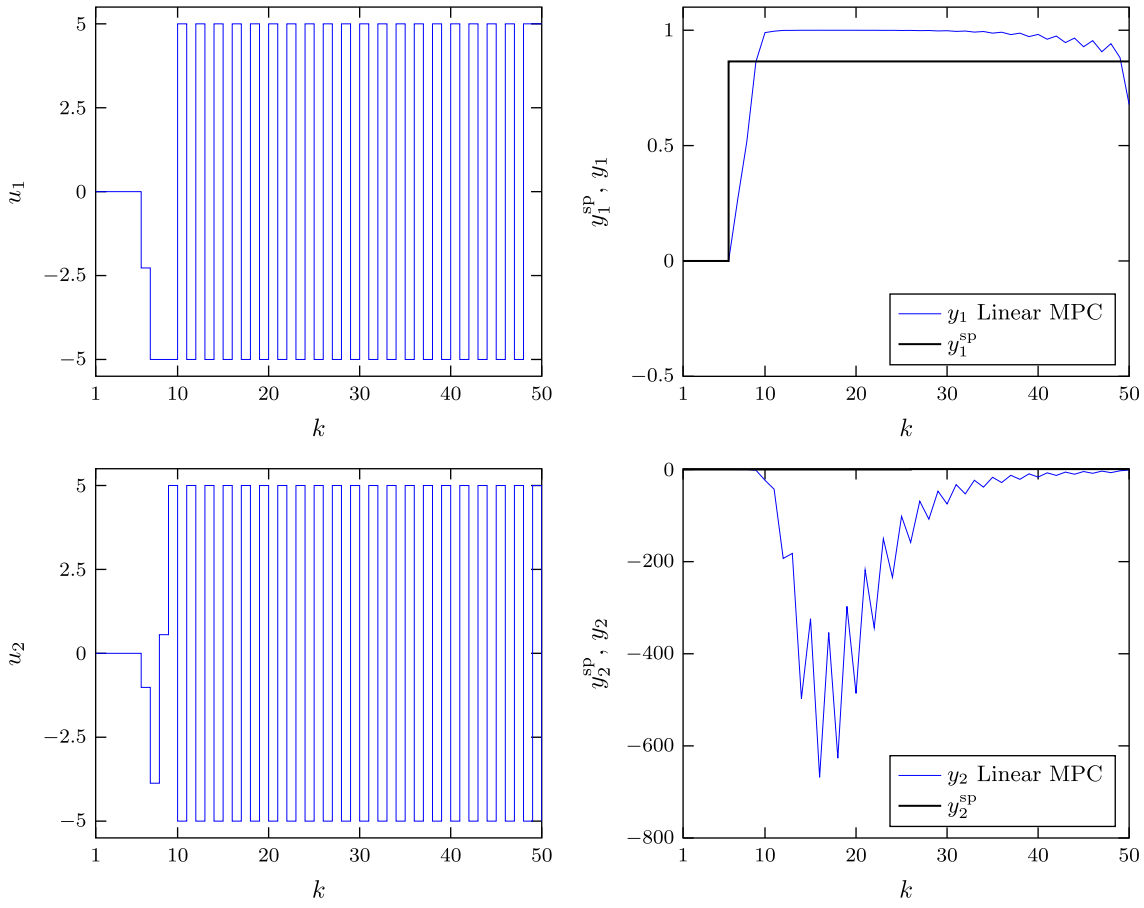


**Fig. 9.** Example 2: simulation results of the linear MPC algorithm.

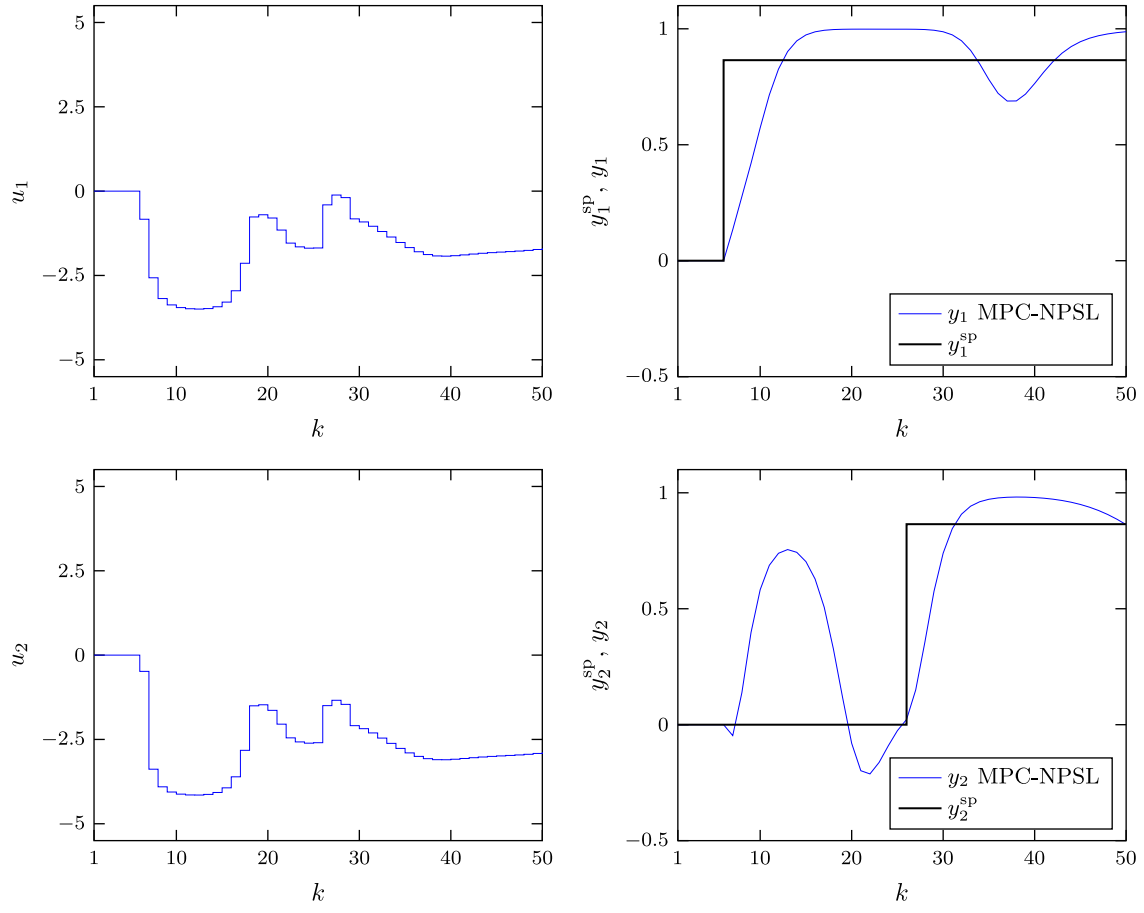


**Fig. 10.** Example 2: simulation results of the MPC-NPSL algorithm with simplified on-line model linearisation and quadratic optimisation.

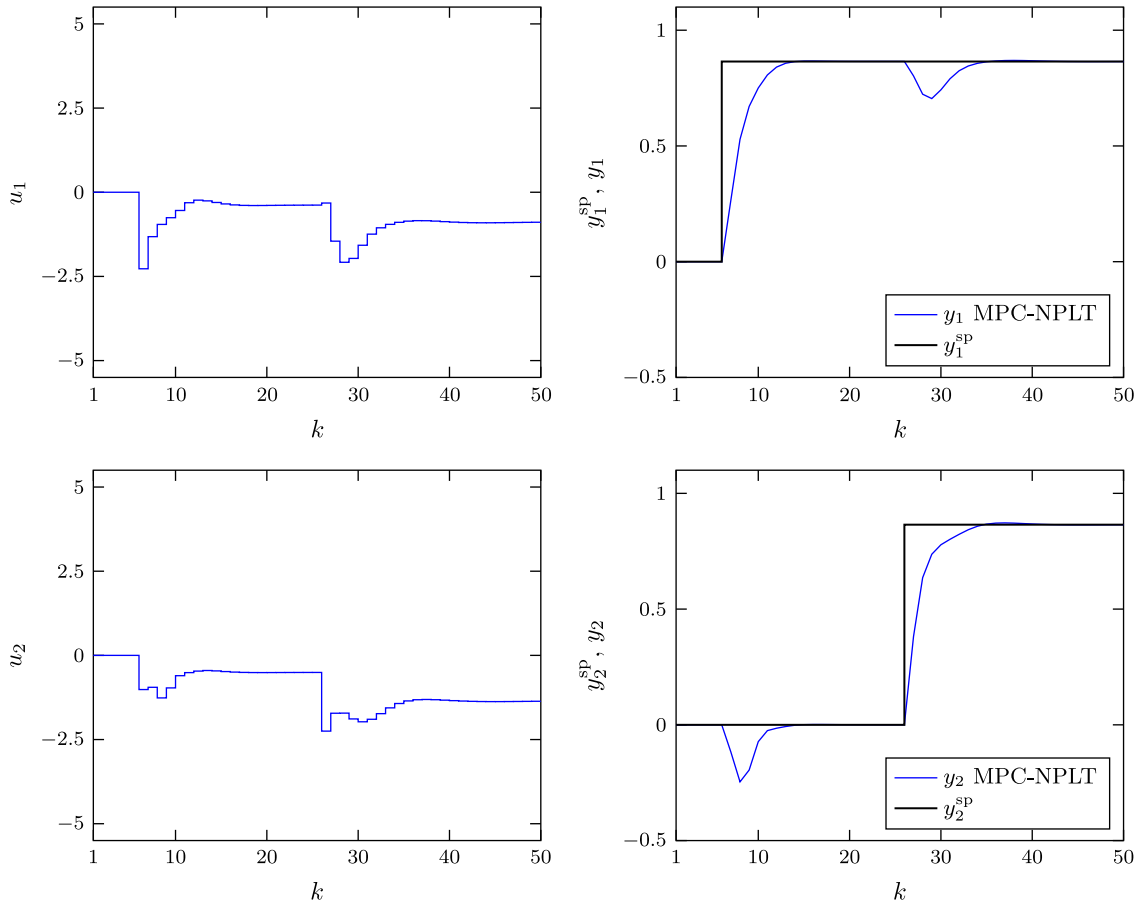


**Fig. 11.** Example 2: simulation results of the MPC-NPLT algorithm with on-line trajectory linearisation and quadratic optimisation.

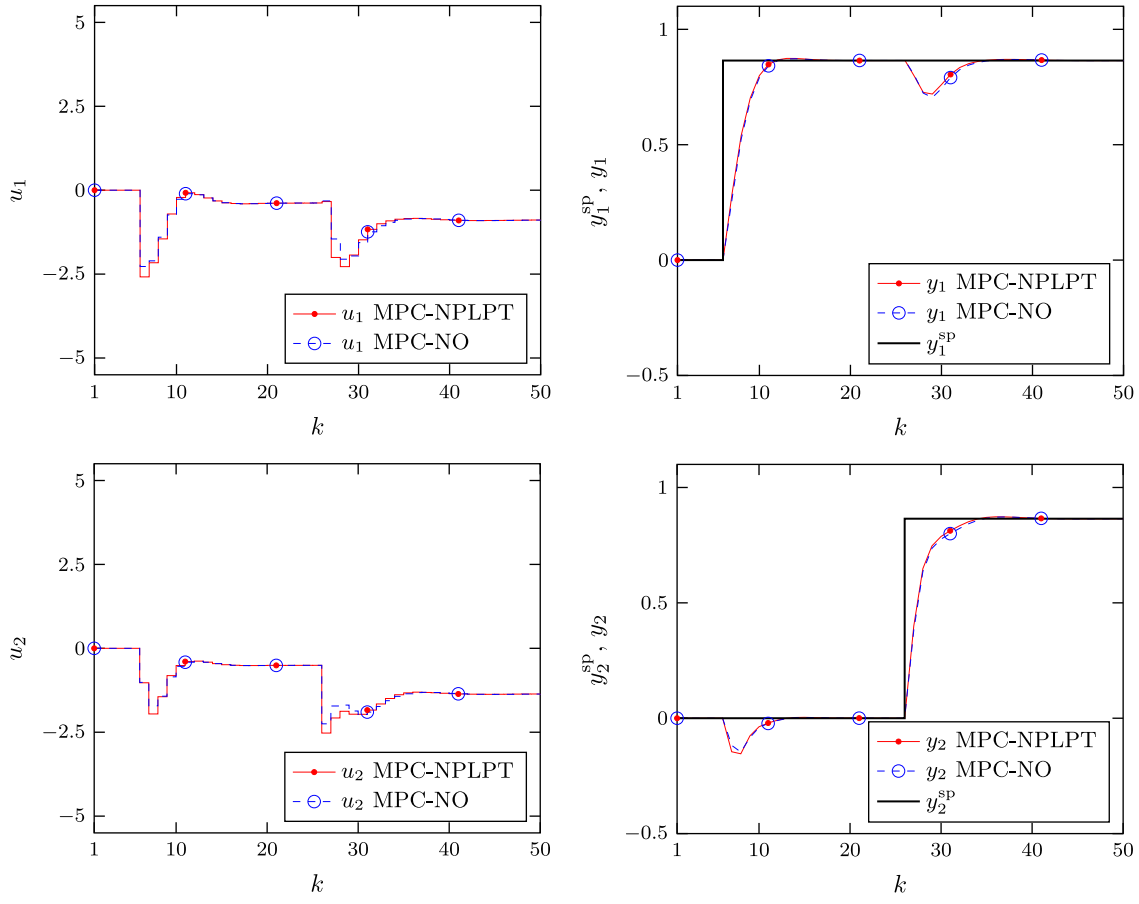


**Fig. 12.** Example 2: simulation results: the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation (*solid line with dots*), the MPC-NO algorithm with on-line nonlinear optimisation (*dashed line with circles*).

**Table 2**
Example 2: control accuracy (SSE) and computational burden (MFLOPS) of compared MPC algorithms.

| Algorithm | SSE | MFLOPS |
|---|---|---|
| Linear MPC | $2.0266 \times 10^6$ | 0.9127 |
| MPC-NPSL | 8.2195 | 0.5311 |
| MPC-NPLT | 2.5281 | 1.3605 |
| MPC-NPLPT | 2.4386 | 1.4820 |
| MPC-NO | 2.3629 | 23.8996 |

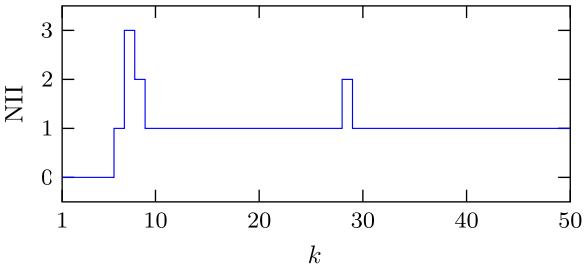


**Fig. 13.** Example 2: The number of internal iterations (NII) in consecutive sampling instants of the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation.

The algorithm uses for prediction the linear model corresponding to the nominal operating point of the process ($u_1 = u_2 = y_1 = y_2 = 0$). Performance of the linear MPC algorithm is much worse than in the case of the first benchmark, it practically does not work for the chosen set-point scenario and simulation horizon. Simulation results of the MPC-NPSL algorithm with simplified on-line model linearisation, shown in Fig. 10, are somehow better, but the algorithm does not converge to the desired set-point. Significantly better responses are obtained in the case of the MPC-NPLT algorithm the simulation results of which are presented in Fig. 11. The algorithm is fast, it successfully follows changes of the set-point trajectory and converges to desired set-points. Finally, Fig. 12 compares simulation results obtained in the described MPC-NPLPT algorithm and in the MPC-NO algorithm. The parameters of the MPC-NPLPT algorithm are $N_0 = 2$, $\triangle_u = \triangle_y = 1$, $t^{max} = 5$. Analogous to Fig. 6 obtained for the first benchmark system, the MPC-NPLPT algorithm gives very good results, practically the same as the computationally demanding MPC-NO strategy. The MPC-NPLPT algorithm has a smaller overshoot than the MPC-NPLT strategy. Table 2 compares control accuracy defined by

$$\mathrm{SSE} = \sum_{k=1}^{50} \sum_{m=1}^{2} \left(y_m^{sp}(k) - y_m(k)\right)^2$$

and computational burden of MPC algorithms. Fig. 13 shows the number of internal iterations in consecutive sampling instants of the MPC-NPLPT algorithm. In the majority of internal iterations only one internal iteration is sufficient, more of them are necessary when the set-point changes. The influence of the tuning parameters $\triangle_u$ and $\triangle_y$ is the same as in the case of the first benchmark system, that is why the results for only one set of parameters are presented which leads good compromise between control accuracy and computational burden.

## 6. Conclusions

This paper describes a nonlinear MPC algorithm for cascade Hammerstein–Wiener systems. A linear approximation of the predicted output trajectory is successively calculated on-line which makes it possible to find the future control scenario from an easy to solve quadratic optimisation problem (or a series of such problems) at each sampling instant. The general multiple-input multiple-output case is considered. Performance of the algorithm is demonstrated in the control system of two benchmark systems. In both cases the responses of obtained in the described algorithm are very close to the optimal ones, possible in the MPC strategy with nonlinear optimisation repeated at each sampling instant on-line. At the same time, as the discussed algorithm uses quadratic optimisation, its computational burden is lower. The considered benchmarks are significantly nonlinear, they are difficult to control by a linear MPC strategy and by a nonlinear MPC algorithm for cascade systems with simplified model linearisation.

Unlike many control schemes for cascade systems, the inverse models of the steady-state parts are not used to compensate for nonlinearity of the process. Instead, the full nonlinear model is taken into account to determine on-line the future predicted output trajectory and its linear approximation. That means that the nonlinear input–output process behaviour is considered. Thanks to such an approach, the process may be characterised by steady-state saturations (which are quite typical in practice), the only condition is that the steady-state nonlinear parts of the system must be described by differentiable functions.

## Acknowledgements

The work presented in this paper was supported by the Polish national budget funds for science.

## References

[1] Abbasi-Asl R, Khorsandi R, Farzampour S, Zahedi E. Estimation of muscle force with EMG signals using Hammerstein–Wiener model. In: Abu Osman NA, et al., editors. Proceedings of BIOMED 2011, IFMBE proceedings, vol. 35. Springer, Springer Berlin Heidelberg; 2011. p. 157–60.
[2] Bai EW. A blind approach to the Hammerstein–Wiener model identification. Automatica 2002;38:967–79.
[3] Bai EW. An optimal two-stage identification algorithm for Hammerstein–Wiener nonlinear systems. Automatica 1998;34:333–8.
[4] Bellemans T, De Schutter B, De Moor B. Model predictive control for ramp metering of motorway traffic: a case study. Control Eng Pract 2006;14:757–67.
[5] Bigdeli N, Haeri M. Predictive functional control for active queue management in congested TCP/IP networks. ISA Trans 2009;48:107–21.
[6] Blanco E, de Prada C, Cristea S, Casas J. Nonlinear predictive control in the LHC accelerator. Control Eng Pract 2009;17:1136–47.
[7] Bloemen HHJ, Boom TJJ, Verbruggen HB. Model-based predictive control for Hammerstein–Wiener systems. Int J Control 2001;74:482–95.
[8] Bolkvadze GR. The Hammerstein–Wiener model for identification of stochastic systems. Autom Remote Control 2003;64:1418–31.
[9] Camacho EF, Bordons C. Model predictive control. London: Springer; 1999.
[10] Chan KH, Bao J, Whiten WJ. A new approach to control of MIMO processes with static nonlinearities using an extended IMC framework. Comput Chem Eng 2005;30:329–42.
[11] Chernyshov K. Identification of stochastic N-L-N systems using monotone correlations. In: Proceedings of the 12th IFAC symposium on system identification, Santa Barbara, USA, 2000.
[12] Colin G, Chamaillard Y, Bloch G, Corde G. Neural control of fast nonlinear systems–application to a turbocharged SI engine with VCT. IEEE Trans Neural Netw 2007;18:1101–14.
[13] Ding B, Ping X. Dynamic output feedback model predictive control for nonlinear systems represented by Hammerstein–Wiener model. J Process Control 2012;22:1773–84.
[14] Dubay R, Hassan M, Li C, Charest M. Finite element based model predictive control for active vibration suppression of a one-link flexible manipulator. ISA Trans 2014;53:1609–19 http://dx.doi.org/10.1016/j.isatra.2014.05.023.
[15] Dubay R. Self-optimizing MPC of melt temperature in injection moulding. ISA Trans 2002;41:81–94.
[16] Dutta P, Rhinehart RR. Application of neural network control to distillation and an experimental comparison with other advanced controllers. ISA Trans 1999;38:251–78.
[17] Eskinat E, Johnson SH, Luyben WL. Use of Hammerstein models in identification of nonlinear systems. AIChE J 1991;37:255–68.
[18] Falkner AH. Iterative technique in the identification of a non-linear system. Int J Control 1988;48:385–96.

[19] Goethals I, Pelckmans K, Hoegaerts L, Suykens JAK, De Moor B. Subspace intersection identification of Hammerstein–Wiener systems. In: Proceedings of the 44th Conference on Decision Control (CDC) and European Control Conference (ECC), Seville, Spain, 2005.
[20] Gruber JK, Ramírez DR, Álamo T, Camacho EF. Min-max MPC based on an upper bound of the worst case cost with guaranteed stability. Application to a pilot plant. J Process Control 2011;21:194–204.
[21] Gruber JK, Doll M, Bordons C. Design and experimental validation of a constrained MPC for the air feed of a fuel cell. Control Eng Pract 2009;17:874–85.
[22] Henson MA. Nonlinear model predictive control: current status and future directions. Comput Chem Eng 1998;23:187–202.
[23] Hong M, Cheng S. Hammerstein–Wiener model predictive control of continuous stirred tank reactor. In: Hu W, editor. Electronics and signal processing. Lecture notes in electric engineering, vol. 97. Heidelberg: Springer; 2011. p. 235–42.
[24] Giri F, Bai EW, editors. Block-oriented nonlinear system identification. Lecture notes in control and information sciences, vol. 404. Berlin: Springer; 2010.
[25] Janczak A. Identification of nonlinear systems using neural networks and polynomial models. A block-oriented approach. Lecture notes in control and information sciences, vol. 310. Berlin: Springer; 2004.
[26] Lee YJ, Sung SW, Park S, Park S. Input test signal design and parameter estimation method for the Hammerstein–Wiener processes. Ind Eng Chem Res 2004;43:7521–30.
[27] Ławryńczuk M. Practical nonlinear predictive control algorithms for neural Wiener models. J Process Control 2013;23:696–714.
[28] MacArthur JW. A new approach for nonlinear process identification using orthonormal bases and ordinal splines. J Process Control 2013;22:375–89.
[29] Maciejowski JM. Predictive control with constraints. Englewood Cliffs: Prentice Hall; 2002.
[30] Morari M, Lee JH. Model predictive control: past, present and future. Comput Chem Eng 1999;23:667–82.
[31] Nadimi ES, Green O, Blanes-Vidal V, Larsen JJ, Christensen LP. Hammerstein–Wiener model for the prediction of temperature variations inside silage stack–bales using wireless sensor networks. Biosyst Eng 2012;112:236–47.
[32] Nemati A, Faieghi M. The performance comparison of ANFIS and Hammerstein–Wiener models for BLDC motors. In: Hu W, editor. Electronics and signal processing. Lecture notes in electric engineering, vol. 97. Heidelberg: Springer; 2011. p. 29–37.
[33] Nelles O. Nonlinear system identification. From classical approaches to neural networks and fuzzy models. Berlin: Springer; 2001.
[34] Ortega JG, Camacho EF. Mobile robot navigation in a partially structured static environment, using neural predictive control. Control Eng Pract 1996;4:1669–79.
[35] Park HCh, Sung SW, Lee J. Modeling of Hammerstein–Wiener processes with special input test signals. Ind Eng Chem Res 2006;45:1029–38.
[36] Patcharaprakiti N, Kirtikara K, Monyakul V, Chenvidhya D, Thongpron J, Sangswang A, et al. Modeling of single phase inverter of photovoltaic system using Hammerstein–Wiener nonlinear system identification. Curr. Appl Phys 2010;10:S532–6.
[37] Patikirikorala T, Wang L, Colman A, Han J. Hammerstein–Wiener nonlinear model based predictive control for relative QoS performance and resource management of software systems. Control Eng Pract 2012;20:49–61.
[38] Pearson RK. Nonlinear empirical modeling techniques. Comput Chem Eng 2006;30:1514–28.
[39] Qin SJ, Badgwell TA. A survey of industrial model predictive control technology. Control Eng Pract 2003;11:733–64.
[40] Rawlings JB, Mayne DQ. Model predictive control: theory and design. Madison: Nob Hill Publishing; 2009.
[41] Schnelle PD, Rollins DL. Industrial model predictive control technology as applied to continuous polymerization processes. ISA Trans 1998;36:281–92.
[42] Shakouri P, Ordys A, Askari MR. Adaptive cruise control with stop&go function using the state-dependent nonlinear model predictive control approach. ISA Trans 2012;51:622–31.
[43] Škrjanc I, Blažič S, Oblak S, Richalet J. An approach to predictive control of multivariable time-delayed plant: stability and design issues. ISA Trans 2004;43:585–95.
[44] Stadler KS, Poland J, Gallestey E. Model predictive control of a rotary cement kiln. Control Eng Pract 2011;19:1–9.
[45] Tatjewski P. Supervisory predictive control and on-line set-point optimization. Int J Appl Math Comput Sci 2010;20:483–95.
[46] Tatjewski P. Advanced control of industrial processes, structures and algorithms. London: Springer; 2007.
[47] Vörös J. An iterative method for Hammerstein–Wiener systems parameter identification. J Electr Eng 2004;55:328–31.
[48] Wang D, Ding F. Extended stochastic gradient identification algorithms for Hammerstein–Wiener ARMAX systems. Comput Math Appl 2008;56:3157–64.
[49] Wei D, Craig IK, Bauer M. Multivariate economic performance assessment of an MPC controlled electric arc furnace. ISA Trans 2007;46:429–36.
[50] van Wingerden JW, Verhaegen M. Subspace identification of Hammerstein–Wiener systems operating in closed-loop. In: Giri F, Bai EW, editors. Block-oriented nonlinear system identification. Lecture notes in control and information sciences, vol. 404. Berlin: Springer; 2010. p. 229–39.
[51] Xu M, Li S, Cai W. Cascade generalized predictive control strategy for boiler drum level. ISA Trans 2005;44:399–411.
[52] Yu DW, Yu DL. Modeling a multivariable reactor and on-line model predictive control. ISA Trans 2005;44:539–59.
[53] Yuzgec U, Becerikli Y, Turker M. Nonlinear predictive control of a drying process using genetic algorithms. ISA Trans 2006;45:589–602.
[54] Zhu Y. Estimation of an N-L-N Hammerstein–Wiener model. Automatica 2002;38:1607–14.