Chemical Engineering Journal 166 (2011) 269–287

Contents lists available at ScienceDirect

Chemical Engineering Journal

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

On-line set-point optimisation and predictive control using neural Hammerstein models

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 29 April 2010
Received in revised form 26 July 2010
Accepted 28 July 2010

Keywords:
Set-point optimisation
Model predictive control
Neutralisation reactors
Neural networks
Hammerstein models
Linearisation

ABSTRACT

This paper discusses a computationally efficient approach to set-point optimisation which cooperates with predictive control and its application to a multivariable neutralisation reactor. In the presented system structure a neural Hammerstein model of the process is used. For set-point optimisation, a linearisation of the steady-state model derived from the neural Hammerstein model is calculated on-line. As a result, the set-point is determined from a linear programming problem. For predictive control, a linear approximation of the neural Hammerstein model is calculated on-line and the control policy is determined from a quadratic programming problem. Thanks to linearisation, the necessity of on-line nonlinear optimisation is eliminated. This article emphasises advantages of neural Hammerstein models: accuracy, a limited number of parameters and a simple structure. Thanks to using such models, model transformations can be carried out very efficiently on-line. It is demonstrated that results obtained in the presented structure are very close to those achieved in a computationally demanding structure with on-line nonlinear optimisation. It is also shown that for the considered neutralisation reactor the classical system structure in which for control and set-point optimisation linear models are used gives numerically wrong results.

1. Introduction

In advanced process control the multilayer (hierarchical) control system structure is typically used [11,49,50]. It is comprised of three layers: the basic (direct) control layer which is responsible for safe operation of the process, the supervisory control layer (the advanced control layer) and the set-point optimisation layer which calculates on-line economically optimal set-point. Classical single-loop PID controllers are usually used for basic control whereas model predictive control (MPC) algorithms [35,46,50] are usually used for supervisory control. In MPC an explicit dynamic model of the process is used to predict on-line its future behaviour over some time horizon and to optimise the future control policy. A unique advantage of MPC algorithms is the fact that they can take into account constraints imposed on both process inputs (manipulated variables) and outputs (controlled variables). Constraints satisfaction is very important in practice because they usually determine quality, economic efficiency and safety. Moreover, MPC techniques are very efficient when applied to multivariable processes with many manipulated and many controlled variables (multiple-input multiple-output (MIMO) processes). The majority of chemical processes (reactors, distillation columns, etc.) are multivariable by nature, existing cross-couplings are strong and cannot be neglected in control.

Cooperation of set-point optimisation and MPC algorithms is an important field of research and it has great practical significance [4,5,9,44,49,50]. Although it is possible to integrate set-point optimisation and MPC into one optimisation problem [14,50,53], the multilayer control system structure is usually implemented. It is assumed that disturbances are slowly-varying (when compared to the dynamics of the process). Thanks to such an assumption, the nonlinear set-point optimisation problem can be solved reasonably less frequently than the MPC controller executes. Provided that the dynamics of disturbances is much slower than the dynamics of the plant, this approach gives satisfactory results. Very often disturbances (flow rates, properties of feed and energy streams, etc.) vary significantly and not much slower than the dynamics of the process. In such cases operation in the classical structure with frequency of set-point optimisation much lower than that of MPC may result in a significant loss of economic effectiveness [49,50]. Ideally, nonlinear set-point optimisation should be repeated on-line as often as MPC

Abbreviations: ASSTO, adaptive steady-state target optimisation; LSSO, local steady-state optimisation; MIMO, multiple-input multiple-output; MPC, model predictive control; MPC-NO, model predictive control with nonlinear optimisation; MPC-NPL, model predictive control with nonlinear prediction and linearisation; MSE, mean squared error; ss, steady-state; SSTO, steady-state target optimisation.

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

1385-8947/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.cej.2010.07.065

is activated. Because of high computational complexity, it is usually not possible. Moreover, nonlinear optimisation may terminate in local minima. Hence, nonlinear set-point optimisation is rarely used on-line.

To reduce the computational complexity of nonlinear set-point optimisation in the simplest case a constant linear steady-state model derived from the dynamic model used in MPC can be used [21,30,44,49,50]. It leads to the steady-state target optimisation (SSTO) structure in which set-points are calculated from an easy to solve linear programming problem. It can be solved on-line as frequently as MPC is activated. Unfortunately, in case of nonlinear processes such an approach can give economically and numerically wrong operating points. To solve this problem it is possible to estimate and take into account uncertainty in the steady-state gain in the framework of a robust steady-state target calculation [21]. Alternatively, in the adaptive steady-state target optimisation (ASSTO) structure the comprehensive nonlinear steady-state model is linearised on-line and set-point optimisation needs solving a linear programming task [30,44,49,50]. Piecewise-linear steady-state target optimisation [32] or a trained off-line neural network which approximates the solution to the set-point optimisation problem [27] can be also used.

In the multilayer control system structure for set-point optimisation a steady-state model of the process is used, for MPC a dynamic model is used. Both models are used on-line. In general, main measures of model utility are: approximation accuracy, suitability for on-line application, easiness of development and physical interpretation [43]. Fundamental (first-principle) models [26,37], although potentially very precise, are usually not suitable for on-line set-point optimisation and control. Fundamental dynamic models are comprised of systems of nonlinear differential and algebraic equations which have to be solved on-line in MPC. Fundamental steady-state models consist of systems of nonlinear algebraic equations which also have to be solved on-line. On-line solution of nonlinear equations, although possible, may be computationally demanding as some fundamental models are very complex and may lead to numerical problems (e.g. stiffness, ill-conditioning) [31]. Because neural networks [17,45] can be efficiently used for modelling of numerous technological processes, they can be also used for set-point optimisation and in MPC algorithms. In contrast to solutions based on fundamental models, the necessity of solving on-line differential and algebraic equations is eliminated. MPC algorithms based on neural models have been extensively researched recently, e.g. [1,2,13,33,38,39,50,52].

Classical neural models are entirely black-box models which means that the model structure has nothing to do with the physical nature of the process and model parameters (weights) have no physical interpretation. A sound alternative is to use a neural Hammerstein model which is composed of a nonlinear neural steady-state (static) part in series with a linear dynamic part [20]. Although Hammerstein models are usually obtained without understanding the technological nature of the process, unlike typical black-box models (e.g. neural ones), such models have a clear physical interpretation because their steady-state characteristics can be determined relatively easily. Hammerstein models can be efficiently used for modelling of various chemical processes: e.g. distillation columns [10,12,23], heat exchangers [10], neutralisation reactors [12,22,28,42]. An excellent review of identification algorithms and applications of Hammerstein models in process modelling, control and fault diagnosis is given in [20]. In particular, Hammerstein models can be used for prediction in MPC algorithms, e.g. [6,12,16,28,42].

This paper details a computationally efficient ASSTO approach to set-point optimisation which cooperates with predictive control and its application to a multivariable neutralisation reactor. Unlike solutions discussed in the literature (in which two models are used, i.e. a steady-state one and a dynamic one), in the presented system structure only one neural Hammerstein model of the process is used. For set-point optimisation, a linearisation of the steady-state model derived from the neural Hammerstein model is calculated on-line. As a result, the set-point is determined from a linear programming problem. For predictive control, a linear approximation of the neural Hammerstein model is calculated on-line and the control policy is determined from a quadratic programming problem. Thanks to linearisation, the necessity of on-line nonlinear optimisation is eliminated. It is demonstrated that results obtained in the presented structure are very close to those achieved in a computationally demanding structure with on-line nonlinear optimisation. It is also shown that for the considered neutralisation reactor the classical system structure in which for control and set-point optimisation linear models are used gives numerically wrong results. The general idea of using Hammerstein models for set-point optimisation which cooperates with MPC was first described in a conference paper [29], but only polynomial structures were considered and details not given.

This paper is organised as follows. Section 2 describes the classical multilayer system structure, set-point and MPC optimisation problems are given. Section 3 presents the computationally efficient structure with the ASSTO layer and an efficient MPC algorithm in which the neural Hammerstein model is used. The model structure is described. On-line derivation of a steady-state model from the dynamic model is given, the ASSTO optimisation problem based on a linearised steady-state model is described. On-line calculations necessary in MPC (linearisation of the dynamic neural Hammerstein model, etc.) are discussed, MPC quadratic optimisation problem is given. Section 4 presents development of models of the neutralisation reactor. Linear, polynomial Hammerstein and neural Hammerstein models are compared in terms of accuracy and complexity. Efficiency of the discussed approach to on-line set-point optimisation cooperating with MPC is shown. Section 5 concludes the paper.

2. The classical multilayer control structure

The considered process has n_{u} inputs, n_{e} measured or estimated disturbances (uncontrolled inputs) and n_{y} outputs. In this paper two notation methods are used: vectors and scalars. For compactness of presentation vectors $u = [\, u_1 \ \ldots \ u_{n_{\mathrm{u}}} \,]^{\mathrm{T}}$, $e = [\, e_1 \ \ldots \ e_{n_{\mathrm{e}}} \,]^{\mathrm{T}}$, $y = [\, y_1 \ \ldots \ y_{n_{\mathrm{y}}} \,]^{\mathrm{T}}$ are used. When it is necessary or convenient, elements of these vectors are used, i.e. scalars u_j, where $j = 1, \ldots, n_{\mathrm{u}}$, e_j, where $j = 1, \ldots, n_{\mathrm{e}}$ and y_j, where $j = 1, \ldots, n_{\mathrm{y}}$.

2.1. Set-point optimisation

The standard multilayer system structure is depicted in Fig. 1[11,49,50]. The basic control layer is responsible for safe operation of the process. Unlike other layers, this layer has direct access to input variables of the process. PID controllers or simplified MPC algorithms (i.e. based on linear models) are used in this layer. The second layer—the supervisory control layer (or the advanced control layer) calculates on-line set-points for the basic control layer. Frequently, determined set-points must satisfy predefined constraints. Hence, this layer can be also named the constraint control layer [49]. Because of the necessity of constraint satisfaction and considering the fact that the majority of processes are multivariable, with strong cross-coupling, the MPC technique is a natural choice for the supervisory control layer. The third layer—the local steady-state optimisation (LSSO) layer, calculates on-line economically optimal set-points for the supervisory control layer in such a way that the production profit is maximised and constraints are satisfied. In the case of complex plants, plant-wide optimisation can be

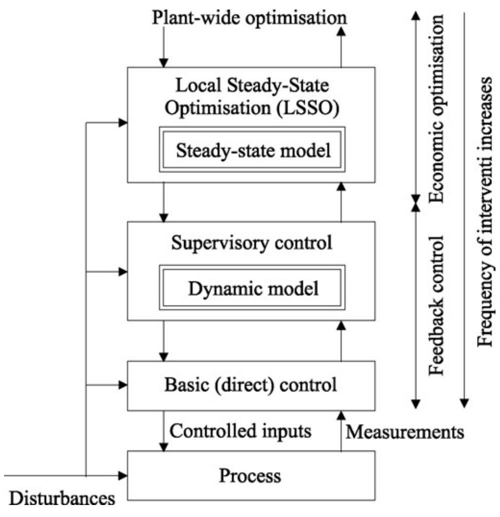

Fig. 1. The classical multilayer control system structure.

also used to determine optimal operating conditions for each unit of the plant [47]. Each layer has a different frequency of intervention, the basic feedback control layer is the fastest.

In this paper cooperation of economic set-point optimisation (the LSSO layer) and MPC algorithms (the supervisory control layer) is considered. Because the objective of set-point optimisation is to maximise the production profit and to satisfy constraints, which determine safety and quality of production, the LSSO layer usually solves on-line the following optimisation problem [49,50]:

$$\begin{aligned}&\min_{u^{ss}}\left\{J_E=c_u^T u^{ss}-c_y^T y^{ss}\right\}\\&\text{subject to}\\&u^{\min}\leq u^{ss}\leq u^{\max}\\&y^{\min}\leq y^{ss}\leq y^{\max}\\&y^{ss}=f^{ss}(u^{ss},e^{ss})\end{aligned}\tag{1}$$

where vectors $c_u=[\,c_{u,1}\ \ldots\ c_{u,n_u}\,]^T$, $c_y=[\,c_{y,1}\ \ldots\ c_{y,n_y}\,]^T$ represent economic prices, constraints imposed on input and output variables are defined by vectors:

$$u^{\min}=\left[u_1^{\min}\quad\ldots\quad u_{n_u}^{\min}\right]^T,\qquad y^{\min}=\left[y_1^{\min}\quad\ldots\quad y_{n_y}^{\min}\right]^T\tag{2}$$

$$u^{\max}=\left[u_1^{\max}\quad\ldots\quad u_{n_u}^{\max}\right]^T,\qquad y^{\max}=\left[y_1^{\max}\quad\ldots\quad y_{n_y}^{\max}\right]^T\tag{3}$$

The superscript ‘ss’ refers to the steady-state.

For set-point optimisation a steady-state model $f^{ss}:\mathbb{R}^{n_u+n_e}\to\mathbb{R}^{n_y}$ of the process is used. Usually, the steady-state model is nonlinear, which means that the LSSO task (1) is a nonlinear optimisation problem. Having obtained the solution to the LSSO problem, u_{lsso}^{ss}, the steady-state model $y^{ss}=f^{ss}(u^{ss},e^{ss})$ is next used to calculate the optimal set-point y_{lsso}^{ss} for the supervisory control layer.

2.2. Model predictive control optimisation

In MPC algorithms [35,46,50] at each sampling instant k (algorithm iteration), $k=1, 2, \ldots$, future control increments are calculated:

$$\triangle\boldsymbol{u}(k)=\begin{bmatrix}\triangle u(k|k)\\\vdots\\\triangle u(k+N_u-1|k)\end{bmatrix}\in\mathbb{R}^{n_uN_u}\tag{4}$$

It is assumed that $\triangle u(k+p\,|\,k)=0$ for $p\geq N_u$, where N_u is the control horizon. The objective of MPC is to minimise differences between the economically optimal set-point $y_{lsso}^{ss}(k)$ calculated by the LSSO layer and predicted output values $\hat{y}(k+p|k)\in\mathbb{R}^{n_y}$ over the prediction horizon $N\geq N_u$, i.e. for $p=1, \ldots, N$. The MPC optimisation problem is

$$\begin{aligned}&\min_{\triangle\boldsymbol{u}(k)}\left\{J_{MPC}(k)=\sum_{p=1}^{N}\left|y_{lsso}^{ss}(k)-\hat{y}(k+p|k)\right|_{\boldsymbol{M}_p}^2\right.\\&\qquad\left.+\sum_{p=0}^{N_u-1}\left|\triangle u(k+p|k)\right|_{\boldsymbol{\Lambda}_p}^2\right\}\\&\text{subject to}\\&u^{\min}\leq u(k+p|k)\leq u^{\max},\quad p=0,\ldots,N_u-1\\&-\triangle u^{\max}\leq\triangle u(k+p|k)\leq\triangle u^{\max},\quad p=0,\ldots,N_u-1\\&y^{\min}\leq\hat{y}(k+p|k)\leq y^{\max},\quad p=1,\ldots,N\end{aligned}\tag{5}$$

where $\boldsymbol{M}_p\geq 0$ and $\boldsymbol{\Lambda}_p>0$ are matrices of dimensionality $n_y\times n_y$ and $n_u\times n_u$, respectively. The second part of the cost function penalises excessive control increments. Only the first n_u elements of the determined sequence (4) are applied to the process (i.e. control moves for the current sampling instant k), $u(k)=\triangle u(k\,|\,k)+u(k-1)$. 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 $\hat{y}(k+1\,|\,k)$, $\hat{y}(k+2\,|\,k)$, $\ldots$, $\hat{y}(k+N\,|\,k)$, a dynamic model of the process is used.

In the MPC optimisation problem (5) magnitude constraints (determined by vectors $u^{\min}$, $u^{\max}\in\mathbb{R}^{n_u}$ and $y^{\min}$, $y^{\max}\in\mathbb{R}^{n_y}$) are usually the same as in the LSSO task (1). Additionally, constraints can be imposed on increments of input variables, $\triangle u^{\max}\in\mathbb{R}^{n_u}$.

The described MPC optimisation problem (5) can be used if the number of inputs is the same as the number of outputs (i.e. $n_u=n_y$). When $n_u>n_y$ the solution to this problem may be not unique. That is why it is necessary to supplement the cost function $J_{MPC}(k)$ with a penalty term $\sum_{p=0}^{N_u-1}\rho_p\left|u(k+p|k)-u_{lsso}^{ss}(k)\right|^2$ or impose an additional constraint $u(k+N_u-1|k)=u_{lsso}^{ss}(k)$ [49].

All things considered, in the standard multilayer control structure depicted in Fig. 1 two optimisation problems have to be solved on-line: the nonlinear economic set-point optimisation problem (1) and the MPC problem (5). When nonlinearity of the process is not significant or when the region of operation is small, MPC algorithms can use linear models of the process. In such a way the MPC optimisation task is in fact an easy to solve quadratic programming problem. Ideally, for set-point optimisation a comprehensive nonlinear steady-state model should be used. As a result, a nonlinear set-point optimisation problem must be solved. Because of high computational complexity it may be impossible to repeat set-point optimisation frequently. The classical multilayer structure with low frequency of economic optimisation can be economically inefficient when disturbances (e.g. flow rates, properties of feed and energy streams) vary significantly and fast [49]. Set-point optimisation based on a steady-state linear model derived from the linear dynamic model used for MPC is possible. In such a case one obtains a linear programming problem (the steady-state target optimisation (SSTO) structure [21,44,49]). Unfortunately, as it is demonstrated in this paper, when set-point optimisation and MPC layers use linear models, for really nonlinear processes (e.g. the neutralisation reactor) such an approach is insufficient.

3. The multilayer structure with adaptive steady-state target optimisation and efficient MPC based on neural Hammerstein models

Because a constant linear steady-state model used for set-point optimisation is unlikely to describe properties of the nonlinear process well enough, a straightforward idea is to update the model on-line. Such an approach leads to the multilayer structure

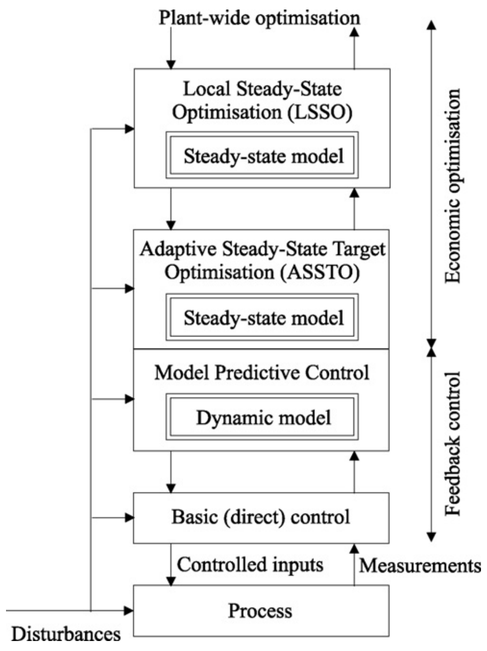

Fig. 2. The multilayer control system structure with adaptive steady-state target optimisation (ASSTO) and MPC.

with adaptive steady-state target optimisation (ASSTO) cooperating with MPC [30,44,49,50] shown in Fig. 2. It consists of the LSSO layer, the ASSTO layer and the MPC layer. Nonlinear LSSO set-point optimisation, in which the nonlinear optimisation problem (1) is solved, is activated infrequently (for verification). At each sampling instant ASSTO and MPC optimisation problems are solved. The ASSTO layer uses a calculated on-line linear approximation of the nonlinear steady-state model used in the LSSO layer. Thanks to linearisation, the ASSTO set-point optimisation problem is a linear programming task. Usually, a constant linear model is used for prediction in MPC. As a result, MPC optimisation (5) is a quadratic programming task.

All things considered, the rudimentary ASSTO structure cooperating with MPC uses a nonlinear steady-state model in the LSSO layer (activated infrequently), its linear approximation (updated on-line) in the ASSTO layer and a constant linear dynamic model in MPC. It means that two separate models are used (i.e. a steady-state one and a dynamic one). Naturally, steady-state properties of the linear dynamic model usually do not correspond with those of the nonlinear steady-state model. In this paper only one neural Hammerstein model of the process is used. The same model is used for both set-point optimisation and MPC. From the full dynamic neural Hammerstein model the nonlinear steady-state model is determined. It is used in the LSSO layer activated infrequently. This steady-state model is linearised on-line for the current operating point and the obtained linear approximation is used in the ASSTO layer. The dynamic neural Hammerstein model is also linearised on-line, this linearisation is next used in MPC. Analogously as in the classical ASSTO+MPC structure, the MPC layer solves a quadratic programming problem, but the model used for prediction is not constant but time-varying. Thanks to the specific structure of the neural Hammerstein model, linear approximations of steady-state and dynamic models can be obtained on-line very efficiently.

3.1. Neural Hammerstein models

The general structure of the considered neural Hammerstein model is depicted in Fig. 3. It is composed of a nonlinear steady-state part connected in series with a linear dynamic part. For the process with n_u inputs, n_e disturbances and n_y outputs, the steady-state nonlinear part can be described in a compact way as

$$x(k) = g(u(k), e(k)) \tag{6}$$

where $g : \mathbb{R}^{n_\mathrm{u}+n_\mathrm{e}} \to \mathbb{R}^{n_\mathrm{x}}$ is a nonlinear function, the linear dynamic part is described by a difference equation:

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

Auxiliary signals (i.e. outputs of the steady-state part) are denoted by $x(k) \in \mathbb{R}^{n_\mathrm{x}}$, q^{-1} is a backward shift operator. From (6) and (7):

$$\boldsymbol{A}(q^{-1})y(k) = \boldsymbol{B}(q^{-1})g(u(k), e(k)) \tag{8}$$

Two types of neural Hammerstein models of multivariable processes are thoroughly discussed in [28]. In the first case the nonlinear steady-state part of the model is realised by only one neural network, in the second case as many as n_x neural networks are used. It is shown that for the neutralisation process the second structure makes it possible to obtain models of good accuracy and significantly smaller number of parameters than the first structure. Hence, in this paper only such models are considered. Multivariable Hammerstein models discussed elsewhere, e.g. in [20,28], do not take into account disturbance signals explicitly, which are essential for set-point optimisation.

The nonlinear steady-state part of the model is realised by n_x multilayer perceptron (MLP) feedforward neural networks [17,45]. Each network has $n_\mathrm{u} + n_\mathrm{e}$ inputs, one hidden layer and a linear output. The structure of neural networks is depicted in Fig. 4. Con-

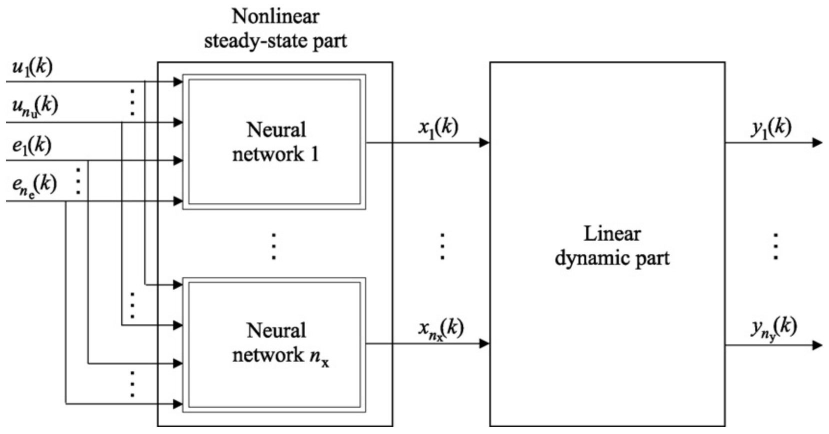

Fig. 3. The MIMO neural Hammerstein model.

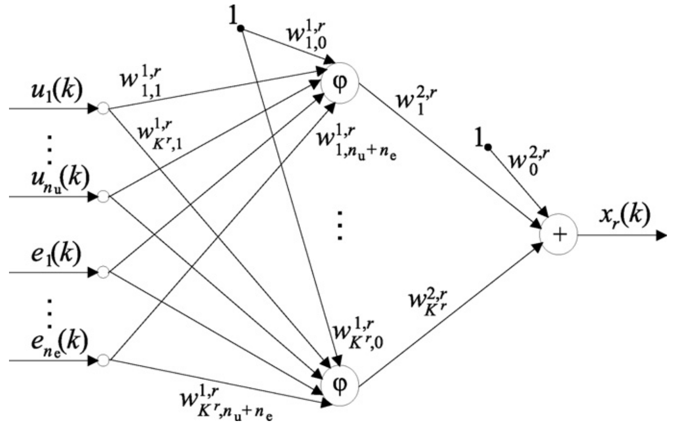

Fig. 4. Neural networks used in the steady-state part of the neural Hammerstein model ($r = 1, \ldots, n_x$).

secutive outputs of neural networks are

$$x_r(k) = w_0^{2,r} + \sum_{i=1}^{K^r} w_i^{2,r} \varphi(z_i^r(k)) \tag{9}$$

where sums of inputs of the ith hidden node are

$$z_i^r(k) = w_{i,0}^{1,r} + \sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k) + \sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k) \tag{10}$$

and K^r is the number of hidden nodes of the rth network ($r = 1, \ldots, n_x$). Weights of networks are denoted by $w_{i,j}^{1,r}$, $i = 1, \ldots, K^r$, $j = 0, \ldots, n_u + n_e$, $r = 1, \ldots, n_x$ and $w_i^{2,r}$, $i = 0, \ldots, K^r$, $r = 1, \ldots, n_x$, for the first and the second layer, respectively. From (9) and (10) outputs of the steady-state part of the model are

$$x_r(k) = w_0^{2,r} + \sum_{i=1}^{K^r} w_i^{2,r} \varphi\left(w_{i,0}^{1,r} + \sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k) + \sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k) \right) \tag{11}$$

The dynamic linear part (7) of the model is defined by the following polynomial matrices:

$$\boldsymbol{A}(q^{-1}) = \begin{bmatrix} A_{1,1}(q^{-1}) & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & A_{n_y,n_y}(q^{-1}) \end{bmatrix} \tag{12}$$

and

$$\boldsymbol{B}(q^{-1}) = \begin{bmatrix} B_{1,1}(q^{-1}) & \ldots & B_{1,n_x}(q^{-1}) \\ \vdots & \ddots & \vdots \\ B_{n_y,1}(q^{-1}) & \ldots & B_{n_y,n_x}(q^{-1}) \end{bmatrix} \tag{13}$$

Entries of matrices $\boldsymbol{A}(q^{-1})$, $\boldsymbol{B}(q^{-1})$ are polynomials in the backward shift operator q^{-1}:

$$A_{i,i}(q^{-1}) = 1 + a_1^i q^{-1} + \ldots + a_{n_A}^i q^{-n_A} \tag{14}$$

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

$$B_{i,j}(q^{-1}) = b_1^{i,j} q^{-1} + \ldots + b_{n_B}^{i,j} q^{-n_B} \tag{15}$$

for $i = 1, \ldots, n_y$, $j = 1, \ldots, n_x$.

From (7), (12)–(15) the mth output of the Hammerstein model ($m = 1, \ldots, n_y$) is

$$y_m(k) = \sum_{r=1}^{n_x} \sum_{l=1}^{n_B} b_l^{m,r} x_r(k-l) - \sum_{l=1}^{n_A} a_l^m y_m(k-l) \tag{16}$$

Because the dynamic Hammerstein model is used for prediction in MPC, it is necessary to express outputs of the model at the current sampling instant k as functions of inputs, disturbances and outputs at previous sampling instants:

$$\begin{aligned} y_m(k) = f_m(&u_1(k-1), \ldots, u_1(k-n_B), \ldots, u_{n_u}(k-1), \ldots, \\ &u_{n_u}(k-n_B), e_1(k-1), \ldots, e_1(k-n_B), \ldots, e_{n_e}(k-1), \ldots, \\ &e_{n_e}(k-n_B), y_m(k-1), \ldots, y_m(k-n_A)) \end{aligned} \tag{17}$$

where functions $f_m : \mathbb{R}^{n_A+(n_u+n_e)n_B} \rightarrow \mathbb{R}$, $m = 1, \ldots, n_y$ describe behaviour of the process. Using (11) and (16), one obtains:

$$\begin{aligned} y_m(k) = \sum_{r=1}^{n_x} \sum_{l=1}^{n_B} b_l^{m,r} \cdot \left[w_0^{2,r} + \sum_{i=1}^{K^r} w_i^{2,r} \varphi \left(w_{i,0}^{1,r} + \sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-l) \right.\right. \\ \left.\left. + \sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k-l) \right) \right] - \sum_{l=1}^{n_A} a_l^m y_m(k-l) \end{aligned} \tag{18}$$

3.2. Set-point optimisation based on neural Hammerstein models

The steady-state nonlinear model can be derived from the dynamic neural Hammerstein model. Using (6) and (7) and setting $q^{-1} = 1$ one has

$$\boldsymbol{A}(1) y^{ss} = \boldsymbol{B}(1) x^{ss} = \boldsymbol{B}(1) g(u^{ss}, e^{ss}) \tag{19}$$

The steady-state model can be expressed as

$$y^{ss} = f^{ss}(u^{ss}, e^{ss}) = \boldsymbol{C} g(u^{ss}, e^{ss}) \tag{20}$$

where $\boldsymbol{C} = \boldsymbol{A}^{-1}(1)\boldsymbol{B}(1)$ is a matrix of dimensionality $n_y \times n_x$ which is independent of the current operating point, it is calculated off-line. One has

$$y^{ss} = \begin{bmatrix} y_1^{ss} \\ \vdots \\ y_{n_y}^{ss} \end{bmatrix} = \begin{bmatrix} c_{1,1} & \ldots & c_{1,n_x} \\ \vdots & \ddots & \vdots \\ c_{n_y,1} & \ldots & c_{n_y,n_x} \end{bmatrix} \begin{bmatrix} x_1(u^{ss}, e^{ss}) \\ \vdots \\ x_{n_x}(u^{ss}, e^{ss}) \end{bmatrix} \tag{21}$$

Taking into account the structure of the steady-state neural part of the Hammerstein model given by (11), consecutive outputs y_m^{ss} ($m = 1, \ldots, n_y$) can be expressed as functions of inputs and disturbances:

$$\begin{aligned} y_m^{ss} &= f_m^{ss}(u^{ss}, e^{ss}) = \sum_{r=1}^{n_x} c_{m,r} x_r(u^{ss}, e^{ss}) \\ &= \sum_{r=1}^{n_x} c_{m,r} \left[w_0^{2,r} + \sum_{i=1}^{K^r} w_i^{2,r} \varphi \left(w_{i,0}^{1,r} + \sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j^{ss} \right.\right. \\ &\quad \left.\left. + \sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j^{ss} \right) \right] \end{aligned} \tag{22}$$

In order to emphasise the fact that the steady-state model can be derived from the neural Hammerstein model in a straightforward way, a classical dynamic black-box model—a neural model is considered. For simplicity of presentation first order dynamics is assumed, the process has one input, one disturbance and one output. The output signal for the current sampling instant k is

$$y(k) = g(u(k-1), e(k-1), y(k-1)) \tag{23}$$

The function $g : \mathbb{R}^3 \to \mathbb{R}^1$ is a nonlinear mapping realised by the neural network. In the steady-state one has

$$y^{ss} = g(u^{ss}, e^{ss}, y^{ss}) \tag{24}$$

The steady-state output y^{ss} is an argument of the function g. Hence, in order to find the steady-state model $y^{ss} = f^{ss}(u^{ss}, e^{ss})$ it would be necessary to solve on-line the nonlinear equation $y^{ss} = g(u^{ss}, e^{ss}, y^{ss})$. On the contrary, the steady-state model $y^{ss} = f^{ss}(u^{ss}, e^{ss})$ is derived directly from the neural Hammerstein model without the necessity of solving any equations.

The nonlinear model (22) is used for nonlinear set-point optimisation in the LSSO layer. Moreover, this model is linearised on-line around the current operating point of the process and next used for set-point optimisation in the ASSTO layer. The operating point is determined by most recent measurements of manipulated and disturbance variables, i.e. by $u(k-1)$ and $h(k)$, respectively. The function $f^{ss}(u^{ss}, e^{ss})$ depends on both inputs u^{ss} and disturbances e^{ss} but it is necessary to recall the fact that from the set-point optimisation problem (1) input values are obtained. Hence, it is obvious that the steady-state model must be linearised with respect to inputs, disturbances are parameters of the model (fixed at the current sampling instant k), they are not arguments of the linearised model. The linear approximation of the nonlinear steady-state model is

$$y^{ss} = f^{ss}(u^{ss}, e^{ss})\big|_{u^{ss}=u(k-1), e^{ss}=e(k)} + \boldsymbol{H}(k)(u^{ss} - u(k-1)) \tag{25}$$

where the matrix:

$$\boldsymbol{H}(k) = \left.\frac{\mathrm{d}f^{ss}(u^{ss}, e^{ss})}{\mathrm{d}u^{ss}}\right|_{u^{ss}=u(k-1), e^{ss}=e(k)} \tag{26}$$

of dimensionality $n_y \times n_u$ consists of partial derivatives of the nonlinear function $f^{ss}(u^{ss}, h^{ss})$:

$$\boldsymbol{H}(k) = \begin{bmatrix} \dfrac{\partial f_1^{ss}(u(k-1), e(k))}{\partial u_1(k-1)} & \cdots & \dfrac{\partial f_1^{ss}(u(k-1), e(k))}{\partial u_{n_u}(k-1)} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_{n_y}^{ss}(u(k-1), e(k))}{\partial u_1(k-1)} & \cdots & \dfrac{\partial f_{n_y}^{ss}(u(k-1), e(k))}{\partial u_{n_u}(k-1)} \end{bmatrix} \tag{27}$$

Taking into account the nonlinear steady-state model (22) derived from the neural Hammerstein model, entries of the matrix $\boldsymbol{H}(k)$ are

$$\frac{\partial f_m^{ss}(u(k-1), e(k))}{\partial u_n(k-1)} = \sum_{r=1}^{n_x} c_{m,r} \left[w_0^{2,r} + \sum_{i=1}^{K^r} w_i^{2,r} \frac{\mathrm{d}\varphi(z_i^r(k))}{\mathrm{d}z_i^r(k)} w_{i,n}^{1,r} \right] \tag{28}$$

where $m = 1, \ldots, n_y$, $n = 1, \ldots, n_u$ and

$$z_i^r(k) = w_{i,0}^{1,r} + \sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-1) + \sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k) \tag{29}$$

If hyperbolic tangent is used as the nonlinear transfer function φ in the hidden layer of steady-state part of the model:

$$\frac{\mathrm{d}\varphi(z_i^r(k))}{\mathrm{d}z_i^r(k)} = 1 - \tanh^2(z_i^r(k)) \tag{30}$$

Thanks to linearisation, from the nonlinear set-point optimisation (LSSO) problem (1), one obtains the equivalent linear programming ASSTO problem:

$$\begin{aligned} &\min_{u^{ss}} \left\{ J_E = c_u^T u^{ss} - c_y^T y^{ss} \right\} \\ &\text{subject to} \\ &u^{min} \le u^{ss} \le u^{max} \\ &y^{min} \le y^{ss} \le y^{max} \\ &y^{ss} = f^{ss}(u^{ss}, e^{ss})\big|_{u^{ss}=u(k-1), e^{ss}=e(k)} + \boldsymbol{H}(k)(u^{ss} - u(k-1)) \end{aligned} \tag{31}$$

Let the solution to the ASSTO problem for the current sampling instant be $u_{assto}^{ss}(k)$. The linearised steady-state model (25) is next used to calculate the optimal set-point $y_{assto}^{ss}(k)$.

Because the nonlinear steady-state model is linearised on-line, entries of the matrix $\boldsymbol{H}(k)$ depend on the current operating point. Hence, the ASSTO layer is adaptive in contrast to the classical SSTO approach which is based on a constant linear model derived from the dynamic one used in MPC. The linearised steady-state model used in ASSTO is consistent with the nonlinear model used in LSSO.

3.3. *Efficient nonlinear MPC-NPL algorithm based on neural Hammerstein models*

Predictions $\hat{y}(k+p|k) \in \mathbb{R}^{n_y}$, for $p = 1, \ldots, N$, which are used in the MPC optimisation problem (5), are calculated using a dynamic model of the process. The general prediction equation is

$$\hat{y}(k+p|k) = y(k+p|k) + d(k) \tag{32}$$

where quantities $y(k+p|k) \in \mathbb{R}^{n_y}$ are calculated from the model. Unmeasured disturbances $d(k) \in \mathbb{R}^{n_y}$ are estimated from

$$d(k) = y(k) - y(k|k-1) \tag{33}$$

where $y(k)$ are measured while $y(k\,|\,k-1)$ are calculated from the model.

If for prediction a nonlinear neural Hammerstein model is used directly, predictions are nonlinear functions of the optimised future control sequence (4). As a result, the MPC optimisation problem is a nonlinear task which has to be solved on-line at each sampling instant. Unfortunately, the computational burden of nonlinear MPC is usually enormous and nonlinear optimisation may terminate in local minima. As an alternative, the MPC scheme with nonlinear prediction and linearisation (MPC-NPL) [33] is used here. The neural Hammerstein model is linearised on-line around the current operating point. The obtained linearisation is used for prediction. Thanks to it, the MPC optimisation task becomes a quadratic programming problem, the necessity of on-line nonlinear optimisation is eliminated.

The MPC algorithm considered here is an extension of the algorithm described in [28] because disturbances e are taken into account.

3.3.1. *MPC quadratic programming problem*

The linear approximation of the nonlinear dynamic neural Hammerstein model, obtained at the sampling instant k, is

$$\boldsymbol{A}(q^{-1})y(k) = \tilde{\boldsymbol{B}}(k, q^{-1})u(k) \tag{34}$$

where

$$\tilde{\boldsymbol{B}}(k, q^{-1}) = \begin{bmatrix} \tilde{B}_{1,1}(k, q^{-1}) & \cdots & \tilde{B}_{1,n_u}(k, q^{-1}) \\ \vdots & \ddots & \vdots \\ \tilde{B}_{n_y,1}(k, q^{-1}) & \cdots & \tilde{B}_{n_y,n_u}(k, q^{-1}) \end{bmatrix} \tag{35}$$

and

$$\tilde{B}_{i,j}(k, q^{-1}) = \tilde{b}_1^{i,j}(k)q^{-1} + \ldots + \tilde{b}_{n_B}^{i,j}(k)q^{-n_B} \tag{36}$$

for $i = 1, \ldots, n_y$, $j = 1, \ldots, n_u$. The matrix $\boldsymbol{A}(q^{-1})$ of the linearised model is the same as that of the rudimentary nonlinear model (8). It is because for the currents sampling instant outputs of the model (18) depend in a linear way on output signals at previous instants. The mth output of the linearised Hammerstein model ($m = 1, \ldots, n_y$) is then

$$y_m(k) = \sum_{n=1}^{n_u} \sum_{l=1}^{n_B} \tilde{b}_l^{m,n}(k) u_n(k-l) - \sum_{l=1}^{n_A} a_l^m y_m(k-l) \tag{37}$$

Coefficients $\tilde{b}_l^{m,n}(k)$ of the linearised model (37) depend on the current operating point of the process.

Using the linearised model (37) recurrently, from the general prediction Eq. (32) output predictions can be expressed as functions of future control increments as proved in [28]:

$$\hat{\boldsymbol{y}}(k) = \underbrace{\boldsymbol{G}(k)\triangle\boldsymbol{u}(k)}_{\text{future}} + \underbrace{\boldsymbol{y}^0(k)}_{\text{past}} \tag{38}$$

where

$$\hat{\boldsymbol{y}}(k) = \begin{bmatrix} \hat{y}(k+1|k) \\ \vdots \\ \hat{y}(k+N|k) \end{bmatrix}, \quad \boldsymbol{y}^0(k) = \begin{bmatrix} y^0(k+1|k) \\ \vdots \\ y^0(k+N|k) \end{bmatrix} \tag{39}$$

are vectors of length $n_{\mathrm{y}}N$. The free trajectory $\boldsymbol{y}^0(k)$ depends only on the past. It is calculated on-line from the neural Hammerstein model. (Linearisation and the free trajectory calculation is discussed in Section 3.3.2.) The dynamic matrix $\boldsymbol{G}(k)$ of dimensionality $n_{\mathrm{y}}N \times n_{\mathrm{u}}N_{\mathrm{u}}$ is calculated on-line taking into account the current state of the plant:

$$\boldsymbol{G}(k) = \begin{bmatrix} \boldsymbol{S}_1(k) & 0 & \ldots & 0 \\ \boldsymbol{S}_2(k) & \boldsymbol{S}_1(k) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{S}_N(k) & \boldsymbol{S}_{N-1}(k) & \ldots & \boldsymbol{S}_{N-N_{\mathrm{u}}+1}(k) \end{bmatrix} \tag{40}$$

It contains step-response coefficients of the local linear approximation of the nonlinear Hammerstein model. Step-response submatrices are

$$\boldsymbol{S}_j(k) = \begin{bmatrix} s_j^{1,1}(k) & \ldots & s_j^{1,n_{\mathrm{u}}}(k) \\ \vdots & \ddots & \vdots \\ s_j^{n_{\mathrm{y}},1}(k) & \ldots & s_j^{n_{\mathrm{y}},n_{\mathrm{u}}}(k) \end{bmatrix} \tag{41}$$

for $j = 1, \ldots, N - N_{\mathrm{u}} + 1$. Step-response coefficients of the linearised model are determined recurrently for $m = 1, \ldots, n_{\mathrm{y}}$, $n = 1, \ldots, n_{\mathrm{u}}$, $j = 1, \ldots, N$ from

$$s_j^{m,n}(k) = \sum_{i=1}^{\min(j,n_{\mathrm{B}})} \tilde{b}_i^{m,n}(k) - \sum_{i=1}^{\min(j-1,n_{\mathrm{A}})} a_i^m s_{j-i}^{m,n}(k) \tag{42}$$

For prediction in (38) a linearised Hammerstein model is used. Thanks to it, the general MPC optimisation problem (5) becomes the following quadratic programming task:

$$\begin{aligned} &\min_{\triangle\boldsymbol{u}(k),\boldsymbol{\varepsilon}^{\min},\boldsymbol{\varepsilon}^{\max}} \left\{ \left|\boldsymbol{y}^{\mathrm{ref}}(k) - \boldsymbol{y}^0(k) - \boldsymbol{G}(k)\triangle\boldsymbol{u}(k)\right|^2 \right. \\ &\quad \left. + \left|\triangle\boldsymbol{u}(k)\right|_{\boldsymbol{\Lambda}}^2 + \rho^{\min}\left|\boldsymbol{\varepsilon}^{\min}\right|^2 + \rho^{\max}\left|\boldsymbol{\varepsilon}^{\max}\right|^2 \right\} \\ &\text{subject to} \\ &\boldsymbol{u}^{\min} \le \boldsymbol{J}\triangle\boldsymbol{u}(k) + \boldsymbol{u}(k-1) \le \boldsymbol{u}^{\max} \\ &-\triangle\boldsymbol{u}^{\max} \le \triangle\boldsymbol{u}(k) \le \triangle\boldsymbol{u}^{\max} \\ &\boldsymbol{y}^{\min} - \boldsymbol{\varepsilon}^{\min} \le \boldsymbol{y}^0(k) + \boldsymbol{G}(k)\triangle\boldsymbol{u}(k) \le \boldsymbol{y}^{\max} + \boldsymbol{\varepsilon}^{\max} \\ &\boldsymbol{\varepsilon}^{\min} \ge 0, \quad \boldsymbol{\varepsilon}^{\max} \ge 0 \end{aligned} \tag{43}$$

where

$$\boldsymbol{y}^{\mathrm{ref}}(k) = \begin{bmatrix} y_{\mathrm{assto}}^{\mathrm{ss}}(k) \\ \vdots \\ y_{\mathrm{assto}}^{\mathrm{ss}}(k) \end{bmatrix} \quad \text{or} \quad \boldsymbol{y}^{\mathrm{ref}}(k) = \begin{bmatrix} y_{\mathrm{lsso}}^{\mathrm{ss}}(k) \\ \vdots \\ y_{\mathrm{lsso}}^{\mathrm{ss}}(k) \end{bmatrix} \tag{44}$$

$$\boldsymbol{y}^{\min} = \begin{bmatrix} y^{\min} \\ \vdots \\ y^{\min} \end{bmatrix}, \quad \boldsymbol{y}^{\max} = \begin{bmatrix} y^{\max} \\ \vdots \\ y^{\max} \end{bmatrix} \tag{45}$$

are vectors of length $n_{\mathrm{y}}N$:

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

$$\triangle\boldsymbol{u}^{\max} = \begin{bmatrix} \triangle u^{\max} \\ \vdots \\ \triangle u^{\max} \end{bmatrix}, \quad \boldsymbol{u}(k-1) = \begin{bmatrix} u(k-1) \\ \vdots \\ u(k-1) \end{bmatrix} \tag{46}$$

are vectors of length $n_{\mathrm{u}}N_{\mathrm{u}}$, $\boldsymbol{M} = \mathrm{diag}(\boldsymbol{M}_1, \ldots, \boldsymbol{M}_N)$, $\boldsymbol{\Lambda} = \mathrm{diag}(\boldsymbol{\Lambda}_0, \ldots, \boldsymbol{\Lambda}_{N_{\mathrm{u}}-1})$ are matrices of dimensionality $n_{\mathrm{y}}N \times n_{\mathrm{y}}N$ and $n_{\mathrm{u}}N_{\mathrm{u}} \times n_{\mathrm{u}}N_{\mathrm{u}}$, respectively, and

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \ldots & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} \\ \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \ldots & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \ldots & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} \end{bmatrix} \tag{47}$$

is the matrix of dimensionality $n_{\mathrm{u}}N_{\mathrm{u}} \times n_{\mathrm{u}}N_{\mathrm{u}}$ comprised of identity and zeros submatrices of dimensionality $n_{\mathrm{u}} \times n_{\mathrm{u}}$.

If hard output constraints are taken into account, the MPC optimisation task may be affected by the infeasibility problem (the admissible set is be empty). To cope with such a situation, output constraints have to be softened by slack variables [35,50]. A quadratic penalty for constraint violations is used in the MPC-NPL optimisation problem (43), $\boldsymbol{\varepsilon}^{\min}$ and $\boldsymbol{\varepsilon}^{\max}$ are vectors of length $n_{\mathrm{y}}N$ comprising slack variables and $\rho^{\min}$, $\rho^{\max} > 0$ are weights.

3.3.2. Linearisation, the free trajectory calculation

Coefficients of the linearised model (34) and (37) are calculated from the neural Hammerstein model (18) as

$$\tilde{b}_l^{m,n}(k) = \frac{\partial y_m(k)}{\partial u_n(k-l)} \tag{48}$$

for all $m = 1, \ldots, n_{\mathrm{y}}$, $n = 1, \ldots, n_{\mathrm{u}}$, $l = 1, \ldots, n_{\mathrm{B}}$. Derivatives (48) are calculated analytically, the structure of the model is exploited.

Let vectors $\bar{\boldsymbol{x}}_m(k) \in \mathbb{R}^{n_{\mathrm{A}}+(n_{\mathrm{u}}+n_{\mathrm{e}})n_{\mathrm{B}}}$ ($m = 1, \ldots, n_{\mathrm{y}}$) denote linearisation points. They are comprised of past input, disturbance and output signal values which are arguments of the Hammerstein model of the mth output (17):

$$\begin{aligned} \bar{\boldsymbol{x}}_m(k) = [&\bar{u}_1(k-1)\ldots\bar{u}_1(k-n_{\mathrm{B}})\ldots\bar{u}_{n_{\mathrm{u}}}(k-1)\ldots\bar{u}_{n_{\mathrm{u}}}(k-n_{\mathrm{B}}) \\ &\bar{e}_1(k-1),\ldots,\bar{e}_1(k-n_{\mathrm{B}}),\ldots,\bar{e}_{n_{\mathrm{u}}}(k-1),\ldots,\bar{e}_{n_{\mathrm{e}}}(k-n_{\mathrm{B}}), \\ &\bar{y}_m(k-1)\ldots\bar{y}_m(k-n_{\mathrm{A}})]^{\mathrm{T}} \end{aligned} \tag{49}$$

Using (18) and (48), one has

$$\tilde{b}_l^{m,n}(k) = \sum_{r=1}^{n_{\mathrm{x}}} b_l^{m,r} \sum_{i=1}^{K^r} w_i^{2,r} \frac{\mathrm{d}\varphi(z_i^r(\bar{\boldsymbol{x}}_m(k)))}{\mathrm{d}z_i^r(\bar{\boldsymbol{x}}_m(k))} w_{i,n}^{1,r} \tag{50}$$

If hyperbolic tangent is used as the nonlinear transfer function φ in the hidden layer of steady-state part of the model:

$$\frac{\mathrm{d}\varphi(z_i^r(\bar{\boldsymbol{x}}_m(k)))}{\mathrm{d}z_i^r(\bar{\boldsymbol{x}}_m(k))} = 1 - \tanh^2(z_i^r(\bar{\boldsymbol{x}}_m(k))) \tag{51}$$

The nonlinear free trajectory $y_m^0(k+p|k)$ over the prediction horizon, $p = 1, \ldots, N$, $m = 1, \ldots, n_{\mathrm{y}}$ is calculated on-line recursively from the general prediction Eq. (32) using the neural Hammerstein model (18). Predictions are

$$\hat{y}_m(k+p|k) = \sum_{r=1}^{n_{\mathrm{x}}} \sum_{l=1}^{I_{\mathrm{uf}}(p)} b_l^{m,r} \cdot \left[w_0^{2,r} + \sum_{i=1}^{K^r} w_i^{2,r} \varphi \left(w_{i,0}^{1,r} \right.\right.$$

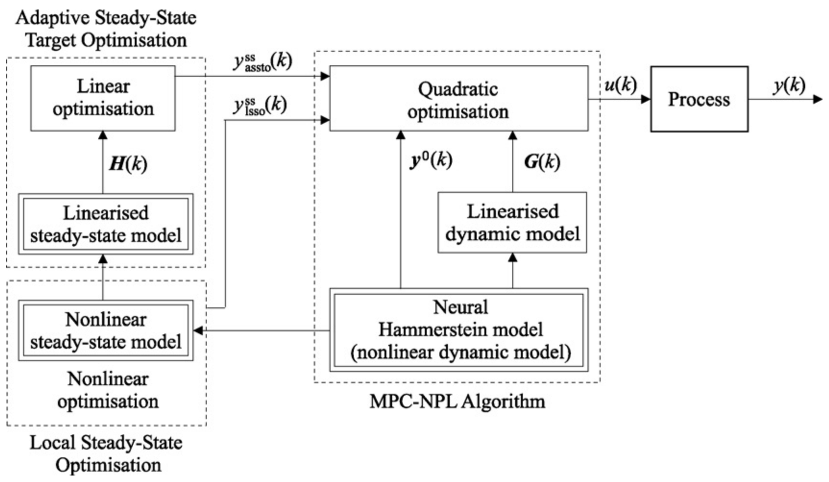

Fig. 5. Detailed configuration of the system structure with adaptive steady-state target optimisation and the MPC-NPL algorithm based on the neural Hammerstein model.

$$
\begin{aligned}
&+\sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-l+p|k)+\sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k-l+p|k)\Bigg)\Bigg]\\
&+\sum_{r=1}^{n_x}\sum_{l=I_{uf}(p)+1}^{n_B} b_l^{m,r}\cdot\Bigg[w_0^{2,r}+\sum_{i=1}^{K^r} w_i^{2,r}\varphi\Bigg(w_{i,0}^{1,r}\\
&+\sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-l+p)+\sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k-l+p)\Bigg)\Bigg]\\
&-\sum_{l=1}^{I_{yp}(p)} a_l^m \hat{y}_m(k-l+p|k)-\sum_{l=I_{yp}(p)+1}^{n_A} a_l^m y_m(k-l+p)\\
&+d_m(k)
\end{aligned}
\tag{52}
$$

where $I_{uf}(p)=\min(p,n_B)$, $I_{yp}(p)=\min(p-1,n_A)$. The free trajectory is derived from (52) taking into account the fact that it depends only on the past, no changes in the control signal from the sampling instant k onwards are assumed, i.e. $u_j(k-l+p\,|\,k)$ should be replaced by $u_j(k-1)$ and future predictions $\hat{y}_m(k+p\,|\,k)$ should be replaced by values of the free trajectory $y_m^0(k+p|k)$. Because future values of disturbances are usually not known in advance, $e_j(k-l+p\,|\,k)$ should be replaced by $e_j(k)$. Hence, the free trajectory is

$$
\begin{aligned}
y_m^0(k+p|k)=&\sum_{r=1}^{n_x}\sum_{l=1}^{I_{uf}(p)} b_l^{m,r}\cdot\Bigg[w_0^{2,r}+\sum_{i=1}^{K^r} w_i^{2,r}\varphi\Bigg(w_{i,0}^{1,r}\\
&+\sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-1)+\sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k)\Bigg)\Bigg]\\
&+\sum_{r=1}^{n_x}\sum_{l=I_{uf}(p)+1}^{n_B} b_l^{m,r}\cdot\Bigg[w_0^{2,r}+\sum_{i=1}^{K^r} w_i^{2,r}\varphi\Bigg(w_{i,0}^{1,r}\\
&+\sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-l+p)+\sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k-l+p)\Bigg)\Bigg]\\
&-\sum_{l=1}^{I_{yp}(p)} a_l^m y_m^0(k-l+p|k)-\sum_{l=I_{yp}(p)+1}^{n_A} a_l^m y_m(k-l+p)\\
&+d_m(k)
\end{aligned}
\tag{53}
$$

Using (18) and (33), the unmeasured disturbance affecting the mth output of the process is estimated from

$$
\begin{aligned}
d_m(k)=y_m(k)-&\sum_{r=1}^{n_x}\sum_{l=1}^{n_B} b_l^{m,r}\cdot\Bigg[w_0^{2,r}+\sum_{i=1}^{K^r} w_i^{2,r}\varphi\Bigg(w_{i,0}^{1,r}\\
&+\sum_{j=1}^{n_u} w_{i,j}^{1,r} u_j(k-l)+\sum_{j=1}^{n_e} w_{i,n_u+j}^{1,r} e_j(k-l)\Bigg)\Bigg]\\
&+\sum_{l=1}^{n_A} a_l^m y_m(k-l)
\end{aligned}
\tag{54}
$$

3.4. Summary of calculations

Detailed configuration of the discussed structure is shown in Fig. 5. Only one neural Hammerstein model is used. From this nonlinear dynamic model the corresponding nonlinear steady-state model is derived on-line. This model is used in the LSSO layer. A linear approximation of the nonlinear steady-state model is also calculated on-line and this linearisation is used in the ASSTO layer. Next, from the dynamic neural Hammerstein model its linearisation is calculated on-line and it is next used in the MPC-NPL algorithm.

The presented structure is computationally efficient because quadratic programming is used in the MPC-NPL algorithm and linear programming is used in the ASSTO layer. The ASSTO layer calculates the set-point as frequently as the MPC-NPL algorithm is activated (at each sampling instant). The LSSO layer is activated infrequently to verify the set-point calculated by the ASSTO layer.

All things considered, at each sampling instant k the following steps are repeated:

1. Linearisation of the steady-state neural model: the matrix $\boldsymbol{H}(k)$ (27) is calculated from (28)–(30).
2. The optimal set-point $u^{ss}_{assto}(k)$ is calculated from the ASSTO optimisation problem (31). Next, the optimal set-point $y^{ss}_{assto}(k)$ is determined using the linearised model (25).
3. If the calculated set-point is verified, the set-point $u^{ss}_{lsso}(k)$ is also calculated from the LSSO nonlinear optimisation problem (1). Next, the optimal set-point $y^{ss}_{lsso}(k)$ is determined using the nonlinear model (22).
4. Linearisation of the dynamic neural Hammerstein model: coefficients $\tilde{b}_l^{m,n}$, $m=1,\ldots,n_y$, $n=1,\ldots,n_u$, $l=1,\ldots,n_B$ are determined

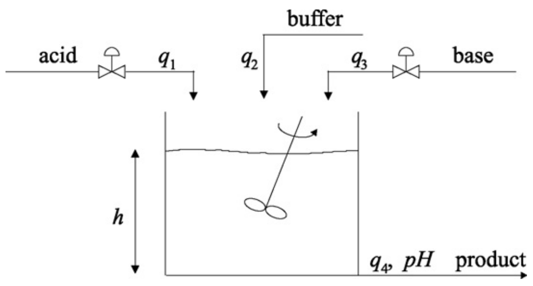

Fig. 6. The neutralisation reactor.

from (50) and (51) which comprise the step-response submatrices $\boldsymbol{S}_j(k)$ (41) and the dynamic matrix $\boldsymbol{G}(k)$ (40).

5. The nonlinear free trajectory $\boldsymbol{y}^0(k)$ is calculated from (53) using the neural Hammerstein model.
6. The MPC-NPL quadratic programming problem (43) is solved. As a result the optimal control policy $\triangle\boldsymbol{u}(k)$ is determined. As the set-point vectors $y^{ss}_{\text{lsso}}(k)$ or $y^{ss}_{\text{assto}}(k)$ calculated by LSSO or ASSTO layers are used.
7. The first n_u elements of the calculated vector $\triangle\boldsymbol{u}(k)$ are applied to the process, i.e. $u(k) = \triangle u(k|k) + u(k-1)$.
8. The iteration number is increased, i.e. $k := k+1$, go to step 1.

4. Simulation results

4.1. Neutralisation reactor

The process under consideration is a multivariable pH neutralisation process. The schematic diagram of the process is depicted in Fig. 6. In the tank acid (HNO_3), base ($NaOH$) and buffer ($NaHCO_3$) are continuously mixed. Chemical reactions occurring in the system are

$$H_2CO_3 \leftrightarrow HCO_3^- + H^+ \tag{55}$$

$$HCO_3^- \leftrightarrow CO_3^{2-} + H^+ \tag{56}$$

$$H_2O \leftrightarrow OH^- + H^+ \tag{57}$$

The liquid level h in the tank and the value of pH of the outlet stream q_4 are controlled by manipulating acid and base flow rates q_1 and q_3, respectively. It means that the process has two inputs (q_1, q_3) and two outputs (h, pH). The buffer flow rate q_2 is the measured disturbance.

The fundamental dynamic model of the process is obtained from component balance and equilibrium relationship under the assumption of perfect mixing. The model consists of three nonlinear ordinary differential equations [28,34,36]:

$$\frac{dW_{a_4}(t)}{dt} = \frac{q_1(t)(W_{a_1} - W_{a_4}(t))}{Ah(t)} + \frac{q_2(t)(W_{a_2} - W_{a_4}(t))}{Ah(t)} + \frac{q_3(t)(W_{a_3} - W_{a_4}(t))}{Ah(t)} \tag{58}$$

$$\frac{dW_{b_4}(t)}{dt} = \frac{q_1(t)(W_{b_1} - W_{b_4}(t))}{Ah(t)} + \frac{q_2(t)(W_{b_2} - W_{b_4}(t))}{Ah(t)} + \frac{q_3(t)(W_{b_3} - W_{b_4}(t))}{Ah(t)} \tag{59}$$

$$\frac{dh(t)}{dt} = \frac{q_1(t) + q_2(t) + q_3(t) - C_V\sqrt{h(t)}}{A} \tag{60}$$

and an algebraic equation for the pH

$$W_{a_4} + 10^{\text{pH}(t)-14} - 10^{-\text{pH}(t)} + W_{b_4}(t) \times \frac{1 + 2\times 10^{\text{pH}(t)-pK_2}}{1 + 10^{pK_1-\text{pH}(t)} + 10^{\text{pH}(t)-pK_2}} = 0 \tag{61}$$

where

$$pK_1 = -\log_{10}K_{a_1}, \qquad pK_2 = -\log_{10}K_{a_2} \tag{62}$$

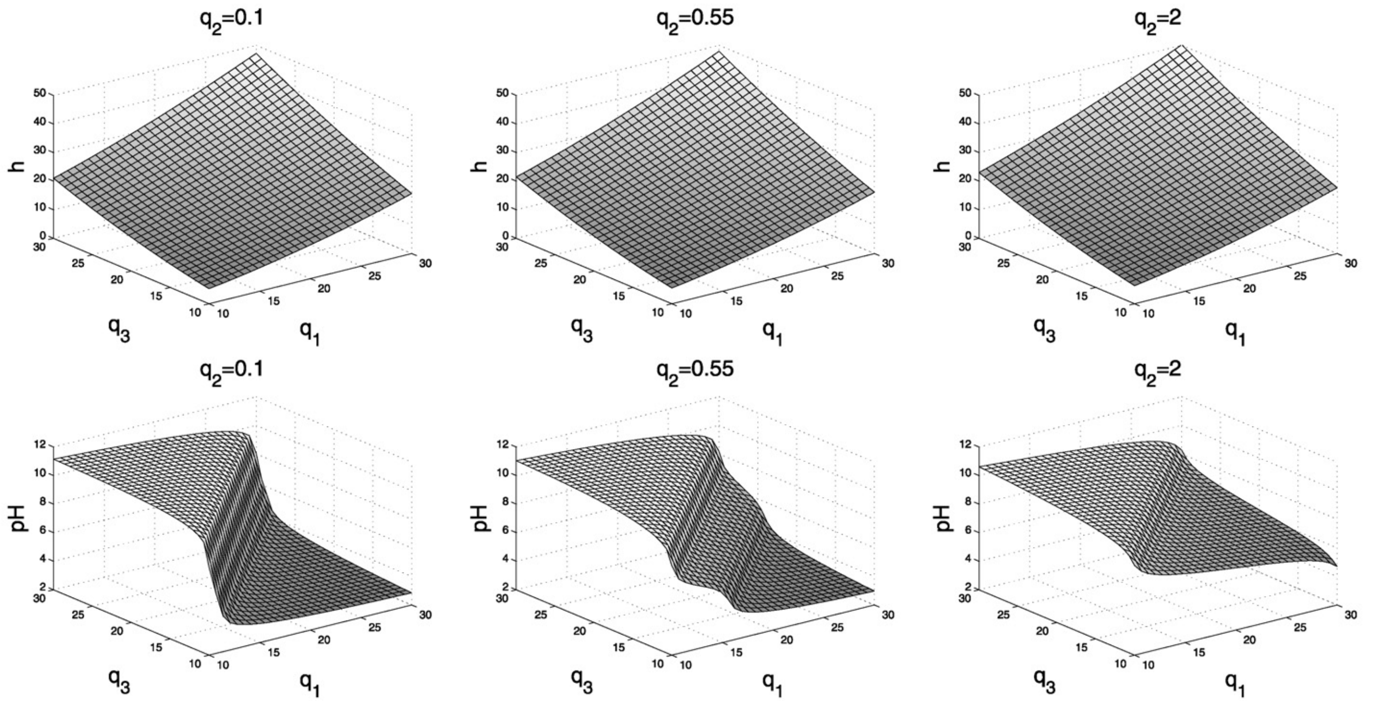

Fig. 7. Steady-state characteristics $h(q_1, q_3)$ and $\text{pH}(q_1, q_3)$ of the neutralisation process for different values of the disturbance: $q_2 = 0.1$ (*left panels*), $q_2 = 0.55$ (*middle panels*), $q_2 = 2$ (*right panels*).

Table 1
Parameters of the model and nominal operating conditions.

$W_{a_1} = 0.003$ M	$W_{b_3} = 0.00005$ M	$q_1 = 16.6$ ml/s
$W_{a_2} = -0.03$ M	$K_{a_1} = 4.47 \times 10^{-7}$	$q_2 = 0.55$ ml/s
$W_{a_3} = -0.00305$ M	$K_{a_2} = 5.62 \times 10^{-11}$	$q_3 = 15.6$ ml/s
$W_{b_1} = 0$ M	$A = 207$ cm^2	$h = 14.0090$ cm
$W_{b_2} = 0.03$ M	$C_V = 8.75$ ml/cm s	pH = 7.0255

and

$$K_{a_1} = \frac{[\mathrm{HCO_3^-}][\mathrm{H^+}]}{[\mathrm{H_2CO_3}]}, \qquad K_{a_2} = \frac{[\mathrm{CO_3^{2-}}][\mathrm{H^+}]}{[\mathrm{HCO_3^-}]} \tag{63}$$

are equilibrium constants of reactions. The pH concentration is the maximum real solution of the nonlinear Eq. (61). Quantities W_{a_i} and W_{b_i} ($i = 1, 2, 3, 4$) denote reaction invariants. Values of model parameters and nominal operating conditions (q_1, q_2, q_3, h, pH) are given in Table 1.

A simplified version of the neutralisation process (with one input and one output) is usually considered in the literature. The multivariable process is researched less frequently [22,24,28,34,36,42]. A noticeable exception is the reactor discussed in [52] which has 3 inputs and 3 outputs.

The pH neutralisation process exhibits severe nonlinear properties. Steady-state characteristics of the reactor, $h(q_1, q_3)$ and pH(q_1, q_3), are depicted in Fig. 7 for different values of the disturbance q_2. In particular, the relation pH(q_1, q_3) is nonlinear. Moreover, dynamic properties of the process are also nonlinear. As demonstrated elsewhere, e.g. in [28], for positive and negative changes in manipulated variables (q_1, q_3) time-constants of obtained step-responses are different. Moreover, these time-constants also depend on the operating point.

Control of pH is of crucial importance in many chemical and biochemical processes. Since the process is significantly nonlinear, it cannot be adequately controlled by means of the classical PI controller or simple MPC algorithms based on constant linear models [36]. Hence, the neutralisation reactor is a standard benchmark used for comparing different model structures and nonlinear control strategies. In the literature applications of different approaches are reported. In the simplest cases the PID controller whose parameters are updated on-line [7] or nonlinear internal model control (IMC) algorithm [19,51] can be used. A gain-scheduling controller is discussed in [40]. Alternatively, adaptive nonlinear control schemes can be used [15,18]. A model reference adaptive control approach is given in [24]. A multimodel robust controller is described in [41].

More recently, different MPC algorithms have been used for the neutralisation reactor. A fuzzy dynamic matrix control (DMC) algorithm is discussed in [36]. The algorithm uses a few local step responses, they are switched using a fuzzy approach. Local step responses are obtained easily from the real process, no complicated identification algorithms are necessary. Multiple model adaptive DMC control strategy is detailed in [8]. The MPC algorithm described in [12] compensates for the nonlinearity of the process but it needs the inverse steady-state model. The MPC algorithm discussed in [28] calculates on-line the local linear approximation of the nonlinear model, it is next used in a quadratic programming problem. In [25] a scheduling quasi-min-max MPC with an infinite horizon (to guarantee stability) is used. In [42] an approach to MPC which uses partial-least-squares (PLS) based Hammerstein models is presented. Input constraints are transformed into a nonlinear region in terms of the so called latent variables. Unfortunately, the MPC algorithm needs nonlinear optimisation. The MPC algorithm discussed in [52] also uses on-line nonlinear optimisation. An interesting alternative to nonlinear MPC of the neutralisation reactor is discussed in [2]. In place of the classical MPC algorithm based on on-line nonlinear optimisation, a neural network approximator is used which calculates on-line the MPC control policy.

Application of the classical multilayer control structure (with the LSSO layer) to the considered multivariable neutralisation reactor is described in [34]. Application of the integrated predictive optimiser and constraint supervisor which provides the regulatory layer with set-points calculated for both optimality and constraint handling is reported in [48]. In both these approaches a linear model is used in MPC which is sufficient because the region of operation is very small. In this article a significantly bigger region is assumed. As a result, the SSTO layer which cooperates with the MPC algorithm based on a linear model gives numerically wrong results.

4.2. Neutralisation reactor modelling

Modelling reported in this article extends previous research during which linear and Hammerstein models of the neutralisation reactor were obtained [28], but assuming a constant disturbance q_2. For set-point optimisation such models are useless because the optimal set-point depends on the current value of the disturbance. Moreover, bearing in mind that from the dynamic Hammerstein model its steady-state description is derived and next used for set-point optimisation, not only dynamic but also steady-state model accuracy is taken into account during model identification described in this paper.

Three types of dynamic models of the process are considered:

(a) the linear model with constant parameters:

$$\begin{aligned} y_1(k) = {} & b_1^{1,1}u_1(k-1) + b_2^{1,1}u_1(k-2) + b_1^{1,2}u_2(k-1) \\ & + b_2^{1,2}u_2(k-2) + c_1^1 e(k-1) \\ & + c_2^1 e(k-2) - a_1^1 y_1(k-1) - a_2^1 y_1(k-2) \end{aligned} \tag{64}$$

$$\begin{aligned} y_2(k) = {} & b_1^{2,1}u_1(k-1) + b_2^{2,1}u_1(k-2) + b_1^{2,2}u_2(k-1) \\ & + b_2^{2,2}u_2(k-2) + c_1^2 e(k-1) + c_2^2 e(k-2) \\ & - a_1^2 y_2(k-1) - a_2^2 y_2(k-2) \end{aligned} \tag{65}$$

where coefficients of the model are $a_i^m, b_i^{m,n}, c_i^m$ for all $m, n, i = 1, 2$,

(b) the neural Hammerstein model which can be expressed as general functions (17):

$$\begin{aligned} y_1(k) = f_1(& u_1(k-1), u_1(k-2), u_2(k-1), u_2(k-2), e(k-1), \\ & e(k-2), y_1(k-1), y_1(k-2)) \end{aligned} \tag{66}$$

$$\begin{aligned} y_2(k) = f_2(& u_1(k-1), u_1(k-2), u_2(k-1), u_2(k-2), e(k-1), \\ & e(k-2), y_2(k-1), y_2(k-2)) \end{aligned} \tag{67}$$

(c) the polynomial Hammerstein model with second-order dynamic part (it can be also expressed as functions (66) and (67)), as the steady-state part of the model two polynomials are used:

$$x_1(k) = \sum_{n_1=0}^{o_1}\sum_{n_2=0}^{o_2}\sum_{n_3=0}^{o_3} \alpha_{1,n_1,n_2,n_3} u_1^{n_1}(k) u_2^{n_2}(k) e^{n_3}(k) \tag{68}$$

$$x_2(k) = \sum_{n_1=0}^{o_1}\sum_{n_2=0}^{o_2}\sum_{n_3=0}^{o_3} \alpha_{2,n_1,n_2,n_3} u_1^{n_1}(k) u_2^{n_2}(k) e^{n_3}(k) \tag{69}$$

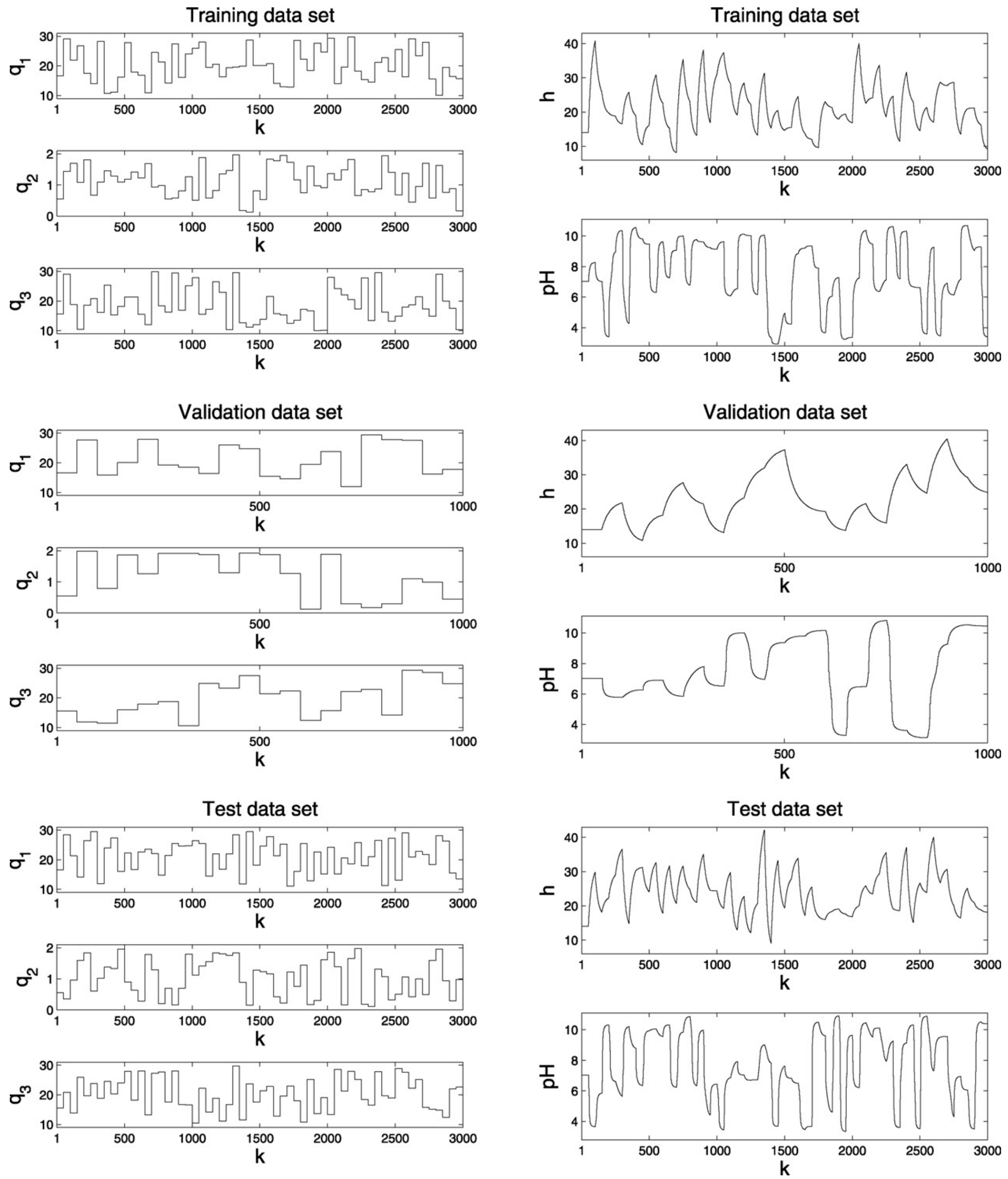

Fig. 8. The dynamic training data set (*top panels*), the dynamic validation data set (*middle panels*) and the dynamic test data set (*bottom panels*).

where integers o_1, o_2 and o_2 determine the order of polynomials, α_{1,n_1,n_2,n_3} and α_{2,n_1,n_2,n_3} are coefficients.

Both classes of Hammerstein models have the same structure depicted in Fig. 3. The only difference is the steady-state part, i.e. it can be realised by neural networks or polynomials.

All three empirical models have the same input arguments determined by $n_A^m = n_B^{m,n} = 2$, $m = 1, 2$, $n = 1, 2$, in case of neural and polynomial Hammerstein models $n_x = 2$. Because input and output process variables have a different order of magnitude, they are scaled:

$$u_1 = \frac{1}{15}(q_1 - q_{1,\mathrm{nom}}), \quad u_2 = \frac{1}{15}(q_3 - q_{3,\mathrm{nom}}), \quad e = \frac{1}{2}(q_2 - q_{2,\mathrm{nom}}),$$
$$y_1 = \frac{1}{35}(h - h_{\mathrm{nom}}), \quad y_2 = \frac{1}{4}(\mathrm{pH} - \mathrm{pH}_{\mathrm{nom}}) \tag{70}$$

where $q_{1,\mathrm{nom}}, q_{2,\mathrm{nom}}, q_{3,\mathrm{nom}}, h_{\mathrm{nom}}, \mathrm{pH}_{\mathrm{nom}}$ correspond to the nominal operating point given in Table 1.

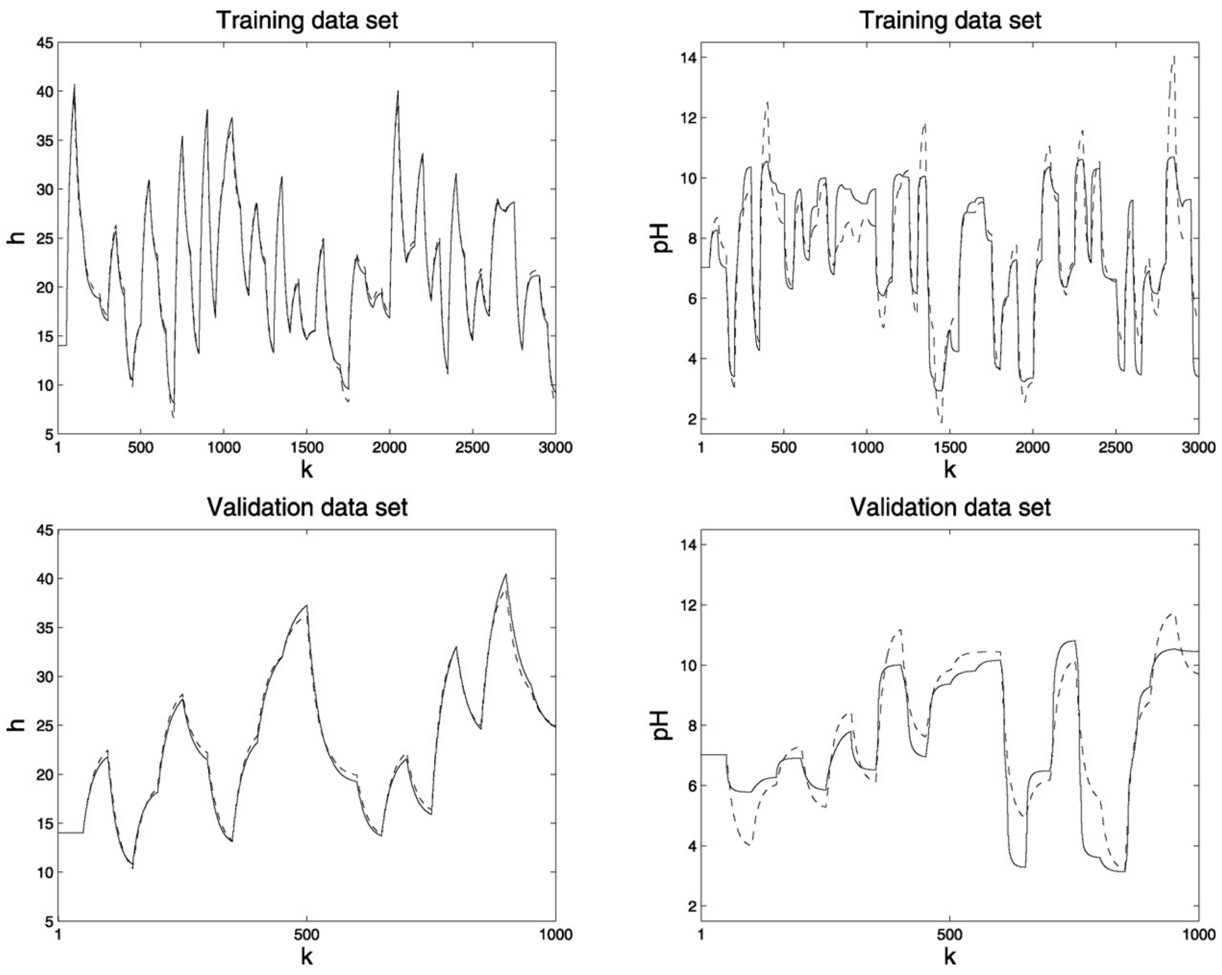

Fig. 9. Outputs of the process (*solid line*) vs. outputs of the linear model (*dashed line*) for the dynamic training data set (*top panels*) and for the dynamic validation data set (*bottom panels*).

The linear model is obtained in such a way that the following mean squared error (MSE) performance function is minimised:

$$\mathrm{MSE} = \frac{1}{S}\sum_{k=1}^{S}\sum_{m=1}^{2}(y_m(k|k-1) - y_m(k))^2 \tag{71}$$

where $y_m(k\,|\,k-1)\,(m=1,\ldots,n_{\mathrm{y}}=1,2)$ denote outputs of the model for the sampling instant k calculated using signals up to the sampling instant $k-1$, $y_m(k)$ are targets, i.e. recorded real values of process output variables, S is the number of samples. Because the MSE function (71) is quadratic, the linear model can be easily calculated using the least-squares method.

Table 2
Properties of neural and polynomial Hammerstein models in terms of the number of parameters and the MSE performance index.

Model	Parameters	$\mathrm{MSE}_{\mathrm{training}}$	$\mathrm{MSE}_{\mathrm{validation}}$	$\mathrm{MSE}_{\mathrm{test}}$
Neural Hammerstein, $K^1=K^2=1$	24	9.4391×10^{-2}	1.1603×10^{-1}	–
Neural Hammerstein, $K^1=K^2=2$	34	3.0922×10^{-2}	3.1471×10^{-2}	–
Neural Hammerstein, $K^1=K^2=3$	44	2.6041×10^{-2}	2.5673×10^{-2}	–
Neural Hammerstein, $K^1=K^2=4$	54	2.3771×10^{-2}	2.3204×10^{-2}	–
Neural Hammerstein, $K^1=K^2=5$	64	2.3288×10^{-2}	2.1752×10^{-2}	2.5830×10^{-2}
Neural Hammerstein, $K^1=K^2=6$	74	2.3170×10^{-2}	2.1514×10^{-2}	–
Neural Hammerstein, $K^1=K^2=7$	84	2.2671×10^{-2}	2.1385×10^{-2}	–
Neural Hammerstein, $K^1=K^2=8$	94	2.2534×10^{-2}	2.1360×10^{-2}	–
Neural Hammerstein, $K^1=K^2=9$	104	2.2463×10^{-2}	2.1275×10^{-2}	–
Neural Hammerstein, $K^1=K^2=10$	114	2.2187×10^{-2}	2.0502×10^{-2}	–
Polynomial Hammerstein, 2th order	39	5.2803×10^{-2}	5.3057×10^{-2}	–
Polynomial Hammerstein, 3th order	76	3.8156×10^{-2}	3.8061×10^{-2}	–
Polynomial Hammerstein, 4th order	137	3.2508×10^{-2}	3.1623×10^{-2}	–
Polynomial Hammerstein, 5th order	228	2.9626×10^{-2}	2.8917×10^{-2}	–
Polynomial Hammerstein, 6th order	355	2.7415×10^{-2}	2.6584×10^{-2}	–
Polynomial Hammerstein, 7th order	524	2.5575×10^{-2}	2.5296×10^{-2}	–
Polynomial Hammerstein, 8th order	741	2.4695×10^{-2}	2.4077×10^{-2}	–
Polynomial Hammerstein, 9th order	1012	2.4381×10^{-2}	2.3585×10^{-2}	–
Polynomial Hammerstein, 10th order	1343	2.4203×10^{-2}	2.3418×10^{-2}	–

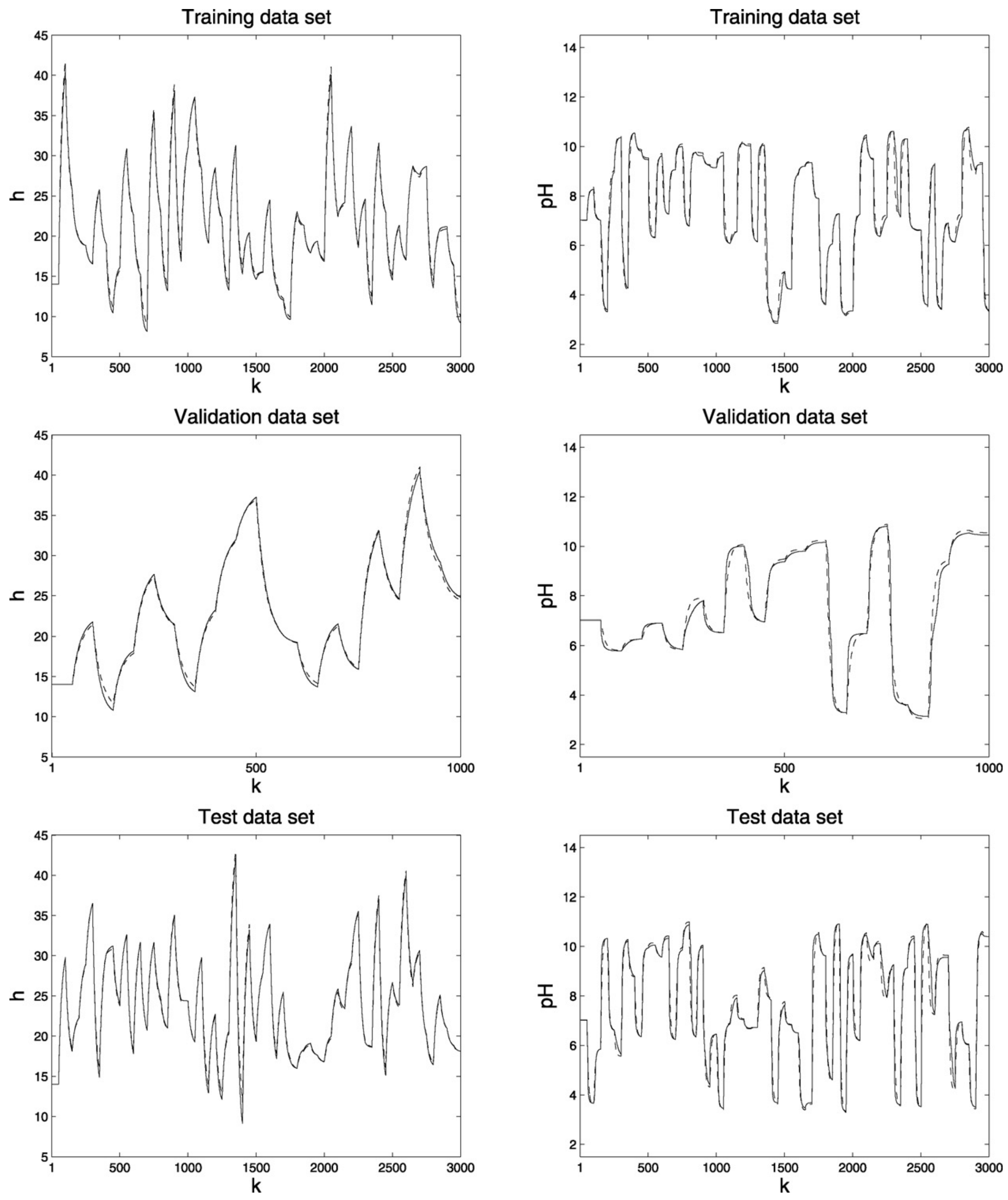

Fig. 10. Outputs of the process (*solid line*) vs. outputs of the chosen neural Hammerstein model (*dashed line*) for the dynamic training data set (*top panels*), for the dynamic validation data set (*middle panels*) and for the dynamic test data set (*bottom panels*).

The dynamic Hammerstein model is used in MPC. In addition to that, from this model its steady-state description is derived and used for set-point optimisation in LSSO and ASSTO layers. It means that the model must be precise in two respects: it must mimic the dynamic behaviour of the process and reflect its steady-state properties. Hence, in contrast to classical identification algorithms used for Hammerstein models [20], in this study nonlinear neural and polynomial Hammerstein models are obtained in such a way that the following MSE performance function is minimised:

$$\mathrm{MSE}=\frac{1}{S}\sum_{k=1}^{S}\sum_{m=1}^{2}(y_m(k|k-1)-y_m(k))^2 +\eta\frac{1}{S^{\mathrm{ss}}}\sum_{s=1}^{S^{\mathrm{ss}}}\sum_{m=1}^{2}(y_{\mathrm{mod},m}^{\mathrm{ss}}(s)-y_m^{\mathrm{ss}}(s))^2 \quad (72)$$

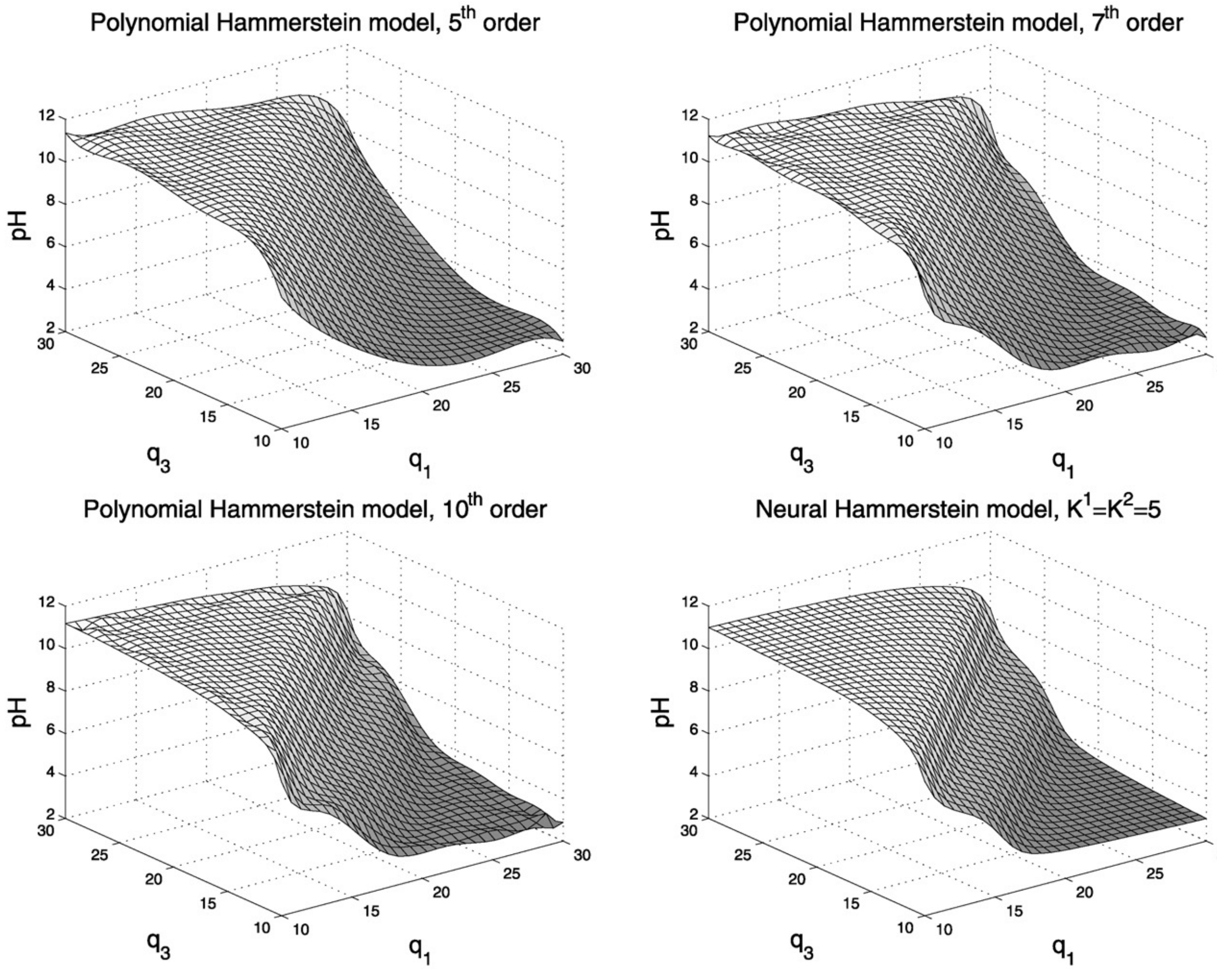

Fig. 11. The steady-state characteristics pH(q_1, q_3) for q_2 = 0.55 obtained from polynomial Hammerstein models of order 5, 7, 10 and from the neural Hammerstein model.

The first part of the above performance function is exactly the same as in the case of the linear model (71), it describes accuracy of the dynamic behaviour of the model. The second part of the MSE function describes accuracy of the steady-state model derived from the dynamic one. Quantities $y^{ss}_{\text{mod},m}(s)$ denote outputs of the nonlinear steady-state model (22), $y^{ss}_m(s)$ are steady-state targets, S^{ss} is the number of steady-state samples. The coefficient η determines the relation of two parts of the MSE function. Its value is adjusted by trial and error to obtain models which have good dynamic and steady-state accuracy. During calculations $\eta = 0.1$.

The fundamental model of the neutralisation reactor (58)–(61) is used as the real process. The system of differential equations (58)–(60) is solved using the Runge–Kutta 45 method. The fundamental model is simulated open-loop in order to obtain three sets of dynamic data, namely training, validation and test data sets depicted in Fig. 8. The domain of interest is: q_1 = 10, ..., 30 ml/s, q_2 = 0.1, ..., 2 ml/s, q_3 = 10, ..., 30 ml/s. Training and test data sets contain 3000 samples, the validation data set contains 1000 samples. The sampling time is 10 s. Output signals contain small measurement noise.

Analogously, steady-state training, validation and test data sets are also obtained from the steady-state fundamental model derived from the dynamic fundamental model (58)–(61). These data sets are generated randomly in such a way that the whole domain of interest is covered. Training and test sets have 8000 samples, the validation set has 2500 samples.

Three dynamic and three steady-state data sets are used. Training data sets are used only for model training, i.e. the MSE performance function (72) is minimised on these sets. As a result, parameters of the model are calculated. Because models should have good generalisation properties, the value of the MSE index for validation sets is monitored during training. When the validation error increases, training is terminated to avoid overfitting. Model selection is performed taking into account the value of MSE only for validation data sets. Test data sets are used to assess generalisation abilities of the chosen model.

The MSE performance index (72) is minimised using the quasi-Newton Broyden–Fletcher–Goldfarb–Shanno algorithm (BFGS) [3] optimisation method. Since training of Hammerstein models is in fact an unconstrained nonlinear minimisation problem, training is repeated 10 times for each model configuration, model parameters are initialised randomly.

At first, the linear model (64), (65) is calculated. Because the process is nonlinear, accuracy of the linear model is very low. Outputs of the process and outputs of the linear model for training and validation data sets (dynamic data) are compared in Fig. 9. Because the steady-state relation pH(q_1, q_2, q_3) is significantly nonlinear as shown in Fig. 7, the linear dynamic model of pH is also very inaccurate. Obtained values of the performance function (71) are: $\text{MSE}_{\text{training}} = 4.3056 \times 10^{-2}$, $\text{MSE}_{\text{validation}} = 4.9998 \times 10^{-2}$.

The steady-state part of the neural Hammerstein model has two separate networks ($n_x = 2$) which have K^1 and K^2 hidden nodes, respectively. The hyperbolic tangent transfer function is used as the function φ in hidden layers. Models with $K^1 = K^2 = 1, \ldots, 10$ hidden nodes are considered. In the steady-state part of the polynomial

Hammerstein model polynomials of order $o_1 = o_2 = o_3 = 2, \ldots, 10$ are used.

Table 2 shows properties of compared neural and polynomial Hammerstein models in terms of the number of parameters and the MSE performance index (72). In general, neural models turn out to be more precise and have a significantly smaller number of parameters. Such an observation is even more evident than in the case of polynomial Hammerstein models which do not take into account the disturbance q_2 [28]. It is because now the steady-state part of the model, given by (68) and (69), has 3 inputs. For example, the neural model with $K^1 = K^2 = 3$ hidden nodes (44 parameters) gives the value of the $\text{MSE}_{\text{validation}}$ index comparable with that obtained by the polynomial model of the 7th order (524 parameters). More complex neural models (i.e. with $K^1 = K^2 > 3$) are characterised by even smaller $\text{MSE}_{\text{validation}}$ values. Hence, considering two compared classes of Hammerstein models, the neural Hammerstein structure is a straightforward choice.

The neural Hammerstein model next used for on-line set-point optimisation and MPC should be both precise and have a limited number of parameters. The model is chosen on the basis of the MSE index for validation data sets. As a reasonable compromise between accuracy and complexity the model with $K^1 = K^2 = 5$ hidden nodes is finally chosen. Although increasing the number of hidden nodes leads to reducing the $\text{MSE}_{\text{validation}}$ index, but this reduction is not significant. The chosen model has only 64 parameters (counting weights of the neural steady-state part and parameters of the linear dynamic part). It is necessary to point out that all polynomial Hammerstein models are less precise.

Finally, for the chosen neural Hammerstein model the MSE index for test data sets is calculated, ($\text{MSE}_{\text{test}} = 2.5830 \times 10^{-2}$). Since these sets are used neither for training nor for model selection, the error on test sets gives an unbiased estimate of the generalisation error. Outputs of the process and outputs of the chosen neural Hammerstein model for training, validation and test data sets (dynamic data) are depicted in Fig. 10. For better comparison, the same axes as in the case of the linear model are used (Fig. 9).

From the dynamic Hammerstein model its steady-state description is derived on-line and next used for set-point optimisation. Example steady-state characteristics pH(q_1, q_3) for $q_2 = 0.55$ obtained from polynomial and neural Hammerstein models are depicted in Fig. 11. Polynomials of order 5, 7 and 10 are used, which means that polynomial Hammerstein models have as many as 228, 524 and 1343 parameters, respectively, whereas the chosen neural model has only 64 parameters. The steady-state characteristics obtained from the Hammerstein model in which polynomials of the 5th order are used is not precise. It is not flat when necessary, in the central part it is not characteristically curved. Of course, by increasing the polynomial order it is possible to improve the shape of the steady-state characteristics of models, but some inaccuracies are still present and such models have a huge number of parameters.

4.3. On-line set-point optimisation and MPC of the neutralisation reactor

To demonstrate accuracy and computational efficiency of the discussed approach, three system structures are compared:

1. The ideal classical multilayer structure with nonlinear set-point optimisation (the LSSO layer) activated at each sampling instant and the MPC-NO algorithm with full nonlinear optimisation.
2. The discussed structure with the ASSTO layer and the MPC-NPL algorithm, the LSSO layer is activated for verification 100 times less frequently than the MPC-NPL algorithm.
3. The classical multilayer structure with the LSSO layer activated 100 times less frequently than the linear MPC algorithm. The steady-state target optimisation (SSTO) layer recalculates the set-point at as frequently as MPC is activated [21,44,49,50].

In the first case at each sampling instant two nonlinear optimisation problems are solved on-line. This structure is computationally demanding, it is treated as the reference. In the second case the LSSO layer is activated infrequently, but the set-point is calculated at each sampling instant by the ASSTO layer which needs linear programming. In the second structure the MPC-NPL algorithms is used, it leads to quadratic programming. In both structures the same neural Hammerstein model (66), (67) with $K^1 = K^2 = 5$ hidden nodes is used, i.e. the full model is used for prediction in MPC, a steady-state model derived from the dynamic one is used for set-point optimisation (LSSO and ASSTO layers).

In the third structure the constant linear model (64), (65) is used for prediction in MPC. The LSSO layer is activated infrequently, it uses a neural steady-state model. The set-point is calculated at each sampling instant by the SSTO layer. This layer uses the linear steady-state model derived from the dynamic one and for calculations linear programming is used. In contrast to the first two structures, in which all models correspond to the rudimentary neural Hammerstein one, in the third structure different models are used in LSSO and SSTO layers.

The fundamental model (58)–(61) is used as the real process. Differential equations (58)–(60) are solved using the Runge–Kutta 45 method. All algorithms are implemented in Matlab. For nonlinear optimisation the sequential quadratic programming (SQP) [3] algorithm is used.

In order to maximise production, for set-point optimisation the objective function is

$$J_{\text{E}} = c_1 q_1^{\text{ss}} + c_3 q_3^{\text{ss}} - c_4 q_4^{\text{ss}} \tag{73}$$

Because in steady-state $q_4^{\text{ss}} = q_1^{\text{ss}} + q_2^{\text{ss}} + q_3^{\text{ss}}$:

$$J_{\text{E}} = (c_1 - c_4) q_1^{\text{ss}} - c_4 q_2^{\text{ss}} + (c_3 - c_4) q_3^{\text{ss}} \tag{74}$$

Using scaling (70):

$$\begin{aligned} J_E \quad = \quad & 15(c_1 - c_4)u_1^{ss} + 15(c_3 - c_4)u_2^{ss} \\ & +(c_1 - c_4)q_{1,nom} + (c_3 - c_4)q_{3,nom} - c_4 q_2^{ss} \end{aligned} \tag{75}$$

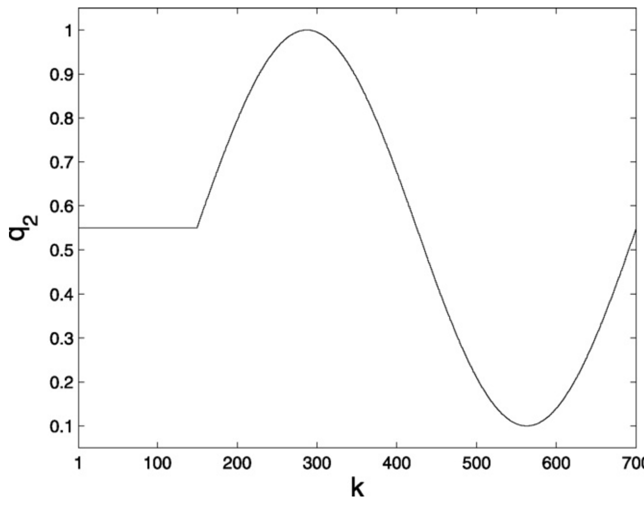

Fig. 12. The disturbance scenario.

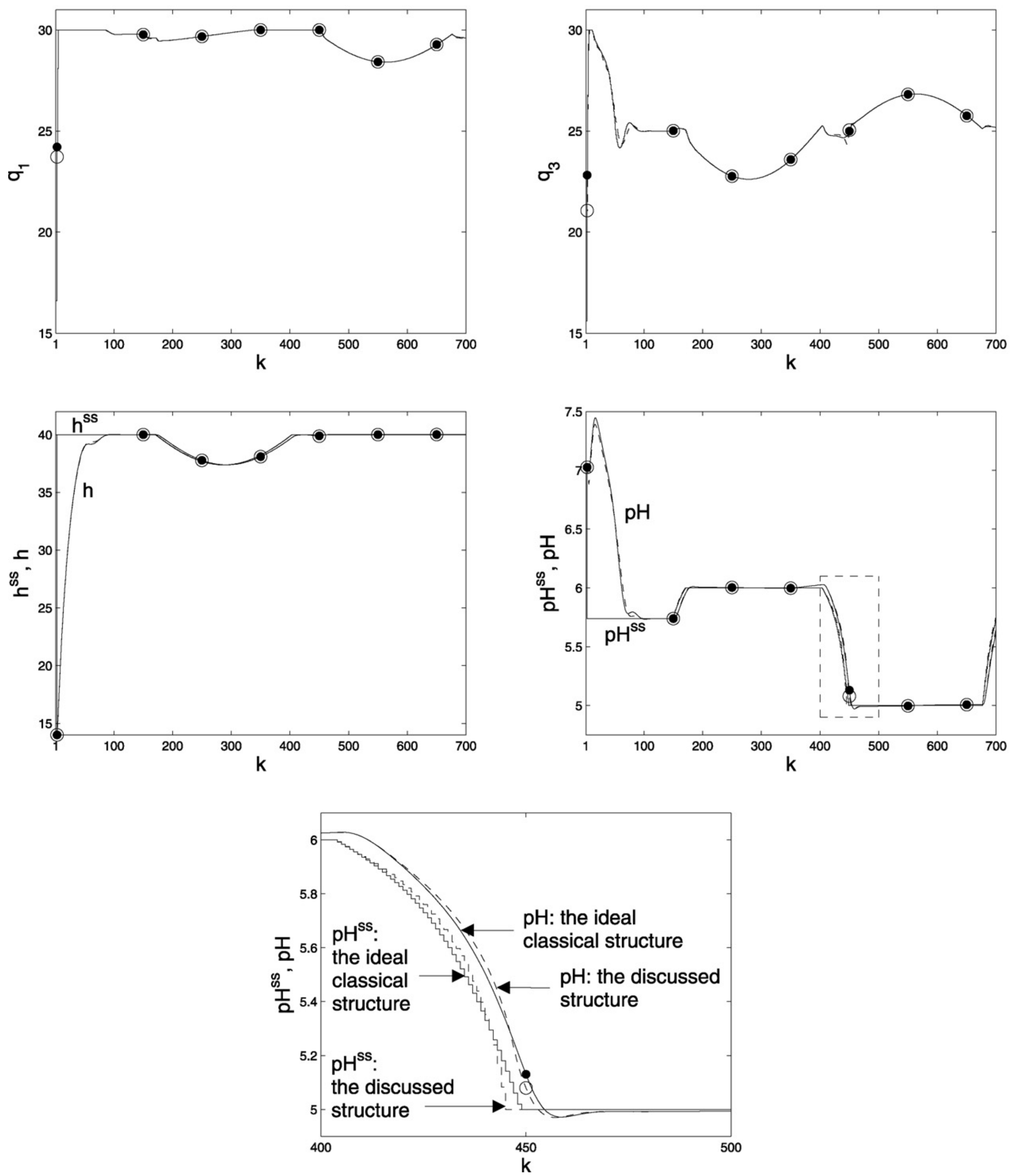

Fig. 13. Simulation results: the ideal classical structure with nonlinear set-point optimisation and the MPC-NO algorithm (*solid line with dots*) vs. the discussed structure with the ASSTO layer and the MPC-NPL algorithm (*dashed line with circles*). In both structures the same same neural Hammerstein model is used. The bottom panel shows an enlarged fragment of the output trajectory.

and taking into account only decision variables of the set-point optimisation, one obtains:

$$J_E = 15(c_1 - c_4)u_1^{ss} + 15(c_3 - c_4)u_2^{ss} \tag{76}$$

In comparison with (1), $u^{ss} = \left[u_1^{ss}\, u_2^{ss}\right]^T$, economic prices are $c_u = [15(c_1 - c_4)\, 15(c_3 - c_4)]^T$, $c_y = [0\,0]^T$. During simulations $c_1 = 1$, $c_3 = 2$, $c_4 = 5$ [34].

The same constraints imposed on manipulated and controlled variables are used in set-point optimisation and MPC. Constraints of manipulated variables are

$$10\,\text{ml/s} \le q_1, \quad q_1^{ss} \le 30\,\text{ml/s}, \quad 10\,\text{ml/s} \le q_3, \quad q_3^{ss} \le 30\,\text{ml/s} \tag{77}$$

whereas constraints of controlled variables are

$$35\,\text{cm} \le h, \quad h^{ss} \le 40\,\text{cm}, \quad 5 \le \text{pH}, \text{pH}^{ss} \le 6 \tag{78}$$

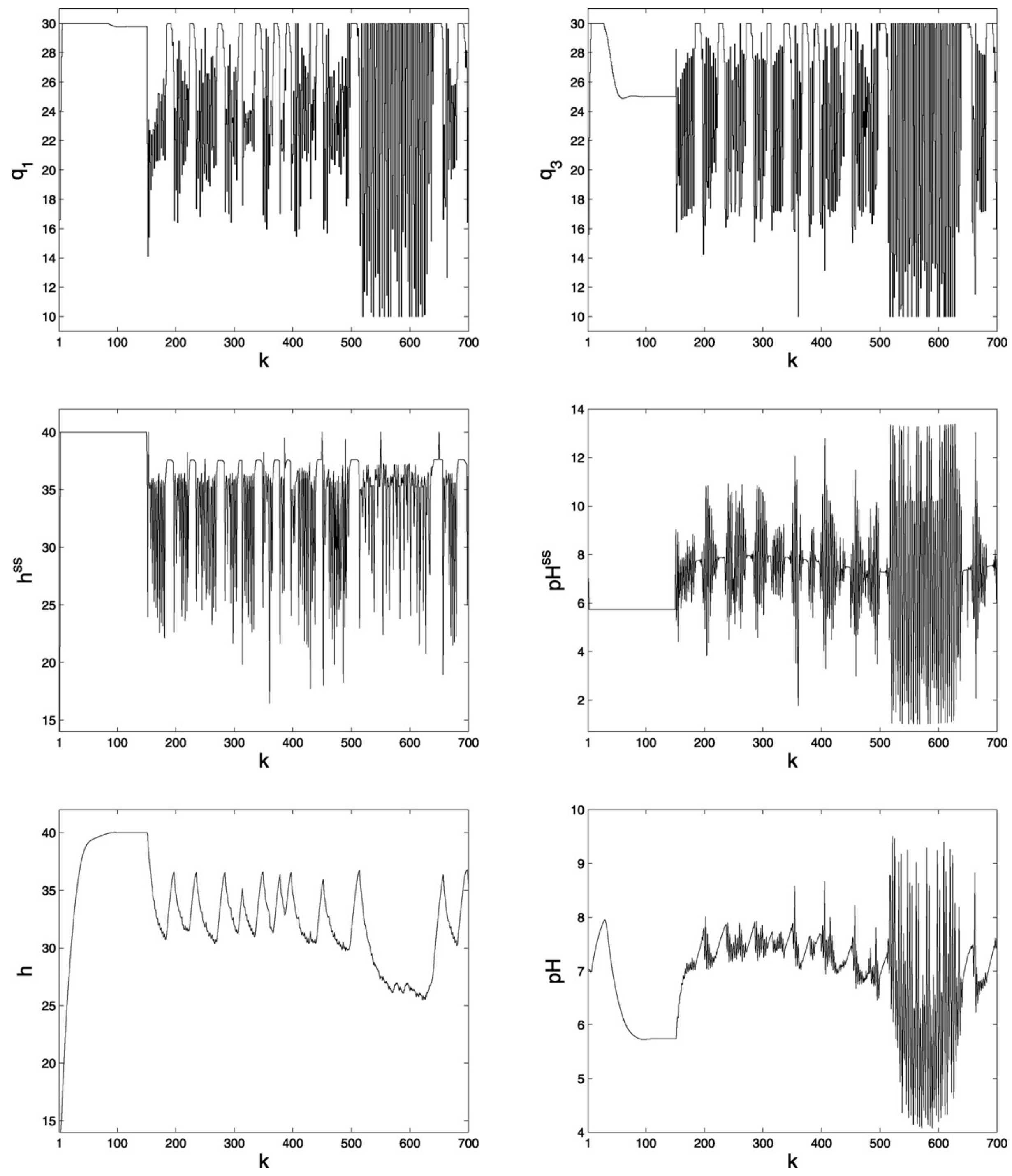

Fig. 14. Simulation results: the classical multilayer structure with the SSTO layer and MPC based on linear models.

The disturbance scenario is

$$q_2(k) = \begin{cases} q_{2,\mathrm{nom}} & k < 150 \\ q_{2,\mathrm{nom}} + 0.45(\sin(\alpha(k-150)))\,\mathrm{ml/s} & 150 \le k \le 700 \end{cases} \tag{79}$$

where $\alpha = 2\pi/(700-150+1)$, i.e. the disturbance changes sinusoidally from the sampling instant $k = 150$, its amplitude is 0.45. The disturbance signal is shown in Fig. 12.

Parameters of MPC are: $N = 10$, $N_u = 2$, $\boldsymbol{M}_p = \mathrm{diag}(1,\ 1)$, $\boldsymbol{\Lambda}_p = \mathrm{diag}(1,1)$, analogously as in [28].

Fig. 13 depicts simulation results obtained in the ideal classical structure with nonlinear set-point optimisation and the MPC-NO algorithm and in the discussed structure with the ASSTO layer and the MPC-NPL algorithm. In both structures the same neural Hammerstein model is used. The following trajectories are depicted:

(a) optimal output trajectories h^{ss}, pH^{ss} calculated from the set-point optimisation problem and
(b) actual (dynamic) input trajectories q_1, q_3 and actual output trajectories h, pH.

At the beginning of simulations the LSSO layer calculates the optimal set-point ($h^{ss} = 40$ cm, $\text{pH}^{ss} = 5.7387$). Because the process is started using nominal conditions (Table 1), MPC algorithms need some time to steer the process to the desired set-point. In the discussed structure the LSSO layer is also activated for $k = 150$, 250, 350, 450, 550, 650, in all remaining iterations the set-point is calculated using the ASSTO layer.

It is evident that all trajectories (set-points h^{ss}, pH^{ss}, inputs q_1, q_3 and outputs h, pH) obtained in the described structure are very close to those obtained in the ideal but unrealistic case when nonlinear set-point optimisation and nonlinear MPC optimisation are repeated at each sampling instant on-line.

The bottom panel in Fig. 13 shows an enlarged fragment of the output pH trajectory. This particular fragment is chosen because differences between two compared structures are the biggest when the set-point is changed from pH = 5 to pH = 6. For the whole simulation horizon (700 sampling instants) the economic performance index:

$$J_{\text{E}} = \sum_{k=1}^{700} J_{\text{E}}(k) = \sum_{k=1}^{700} (c_1 q_1(k) + c_3 q_3(k) - c_4 q_4(k)) \tag{80}$$

is calculated after simulations. Obtained values are very similar, in the first structure $J_{\text{E}} = -13423.30$, in the second structure $J_{\text{E}} = -13422.58$ (the bigger the negative value the better).

Because trajectories and economic results obtained in both structures are similar, it is interesting to compare their computational complexity. The computational cost (in terms of floating point operations) of both structures is assessed. Next, the computational complexity reduction factor is calculated from

$$F = \frac{\text{computational cost of structure 1}}{\text{computational cost of structure 2}} \tag{81}$$

The factor F shows how many times the discussed structure is less computationally demanding in comparison with the first, ideal one. Because overall computational complexity is strongly influenced by the control horizon, for $N_{\text{u}} = 2$: $F = 5.40$, for $N_{\text{u}} = 5$: $F = 13.15$ and for $N_{\text{u}} = 10$: $F = 30.93$.

Finally, the classical multilayer structure with the LSSO layer activated 100 times less frequently than the linear MPC algorithm and the SSTO layer which recalculates the set-point at as frequently as MPC is activated is verified. In MPC a linear dynamic model is used, in the SSTO layer its steady-state version is used. Simulation results are depicted in Fig. 14. Because the process is significantly nonlinear and the linear model is very inaccurate as shown in Fig. 9, when it is used for MPC and set-point optimisation, one obtains numerically wrong results. System trajectories are completely different from those obtained in the ideal multilayer structure and in the discussed structure (Fig. 13).

5. Conclusions

Properties of the considered neutralisation reactor are significantly nonlinear. In consequence, the classical system structure in which for control and set-point optimisation linear models are used gives numerically wrong results. In the computationally efficient system structure described in this paper the neural Hammerstein model of the process is used for both set-point optimisation and MPC. For the ASSTO layer, a linearisation of the steady-state model derived from the Hammerstein one is calculated on-line. Thanks to it, the set-point is determined from an easy to solve linear programming problem. The MPC-NPL algorithm is used for control in which a linear approximation of the Hammerstein model is calculated on-line. Thanks to it, the control policy is determined from a quadratic programming problem. Linearisation makes it possible to eliminate the necessity of repeating on-line nonlinear optimisation at each sampling instant. It is demonstrated that results obtained in the discussed structure are very close to those obtained when for set-point optimisation and MPC nonlinear optimisation is used on-line.

In contrast to the rudimentary structure with the ASSTO layer [49,50] in which two separate models are used (steady-state and dynamic), in the described system structure only one neural Hammerstein model is used. Thanks to the fact that such a model has a specific structure, its steady-state description can be derived in a straightforward manner. It is not possible for a general nonlinear black-box model, e.g. a neural one. Moreover, the model has a regular structure and a limited number of parameters. As a result, linearisation, unmeasured disturbance estimation and free trajectory calculation can be carried out very efficiently on-line.

It is shown in this paper that for the considered neutralisation reactor the neural Hammerstein structure is significantly better than the polynomial Hammerstein one in terms of accuracy and the number of parameters. The reactor is a multivariable process, the polynomial steady-state part of the model needs a big number of parameters as shown in Table 2. Because neural networks are universal approximators, they can be much more efficiently used as the steady-state part of the model. Results reported in this paper confirm the general statement given in [20]. Moreover, because the steady-state model is derived from the dynamic Hammerstein one, it is necessary to point out that models obtained from neural Hammerstein systems are precise whereas models derived from polynomial Hammerstein systems have some significant inaccuracies as shown in Fig. 11.

On the one hand, all above observations have been made for the particular neutralisation reactor. On the other hand, they are quite general because there are numerous multivariable processes in chemical engineering whose properties are nonlinear. It is obvious that for such processes set-point optimisation and predictive control based on linear models are likely to give poor results. Hence, the efficient multilayer system structure based on neural Hammerstein models is a viable alternative.

Acknowledgement

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

References

[1] A. Alexandridis, H. Sarimveis, Nonlinear adaptive model predictive control based on self-correcting neural network models, AIChE J. 51 (2005) 2495–2506.
[2] B.M. Åkesson, H.T. Toivonen, J.B. Waller, R.H. Nyström, Neural network approximation of a nonlinear model predictive controller applied to a pH neutralization process, Comput. Chem. Eng. 29 (2005) 323–335.
[3] M.S. Bazaraa, J. Sherali, K. Shetty, Nonlinear Programming: Theory and Algorithms, John Wiley & Sons, Hoboken, NJ, 2006.
[4] T.L. Blevins, G.K. McMillan, M.W. Wojsznis, Advanced Control Unleashed, ISA, 2005.
[5] M.A. Brdys, P. Tatjewski, Iterative Algorithms for Multilayer Optimizing Control, Imperial College Press/World Scientific, London, 2005.
[6] K.H. Chan, J. Bao, Model predictive control of Hammerstein systems with multivariable nonlinearities, Ind. Eng. Chem. Res. 46 (2007) 168–180.
[7] J. Chen, T.C. Huang, Applying neural networks to on-line updated PID controllers for nonlinear process control, J. Proc. Control 14 (2004) 211–230.
[8] D. Dougherty, D. Cooper, A practical multiple model adaptive strategy for single-loop MPC, Control Eng. Pract. 11 (2003) 141–159.
[9] S. Engell, Feedback control for optimal process operation, J. Proc. Control 17 (2007) 203–219.
[10] E. Eskinat, S. Johnson, W.L. Luyben, Use of Hammerstein models in identification of nonlinear systems, AIChE J. 37 (1991) 255–268.
[11] W.M. Findeisen, F.N. Bailey, M. Brdyś, K. Malinowski, P. Tatjewski, A. Woźniak, Control and Coordination in Hierarchical Systems, J. Wiley & Sons, Chichester, 1980.
[12] K.P. Fruzzetti, A. Palazoğlu, K.A. McDonald, Nonlinear model predictive control using Hammerstein models, J. Proc. Control 7 (1997) 31–41.

[13] M.Y. El Ghoumari, H.J. Tantau, Non-linear constrained MPC: real-time implementation of greenhouse air temperature control, Comput. Electron. Agric. 49 (2005) 345–356.
[14] M. Tvrzska de Gouvea, D. Odloak, One-layer real time optimization of LPG production in the FCC unit: procedure, advantages and disadvantages, Comput. Chem. Eng. 22 (1998) S191–S198.
[15] T. Gustafsson, K. Waller, Nonlinear and adaptive control of pH, Ind. Eng. Chem. Res. 31 (1992) 2681–2693.
[16] G. Harnischmacher, W. Marquardt, Nonlinear model predictive control of multivariable processes using block-structured models, Control Eng. Pract. 15 (2007) 1238–1256.
[17] S. Haykin, Neural Networks—A Comprehensive Foundation, Prentice Hall, Englewood Cliffs, 1999.
[18] M.A. Henson, D.E. Seborg, Adaptive nonlinear control of a pH neutralization process, IEEE Trans. Control Syst. Technol. 2 (1994) 169–182.
[19] Q. Hu, P. Saha, G.P. Rangaiah, Experimental evaluation of an augmented IMC for nonlinear systems, Control Eng. Pract. 8 (2000) 1167–1176.
[20] A. Janczak, Identification of Nonlinear Systems Using Neural Networks and Polynomial Models: Block Oriented Approach, Springer, Berlin, 2004.
[21] D.E. Kassmann, T.A. Badgwell, R.B. Hawkins, Robust steady-state target calculation for model predictive control, AIChE J. 46 (2000) 1007–1024.
[22] S. Lakshminarayanan, S.L. Shah, K. Nandakumar, Identification of Hammerstein models using multivariate statistical tools, Chem. Eng. Sci. 50 (1995) 3599–3613.
[23] W.M. Ling, D. Rivera, Nonlinear black-box identification of distillation column models—design variable selection for model performance enhancement, Int. J. Appl. Math. Comput. Sci. 8 (1998) 793–813.
[24] A.P. Loh, D.S. De, P.R. Krishnaswamy, pH and level controller for a pH neutralization process, Ind. Eng. Chem. Res. 40 (2004) 3579–3584.
[25] Y. Lu, Y. Arkun, A. Palazoğlu, Real-time application of scheduling quasi-min-max model predictive control to a bench-scale neutralization reactor, Ind. Eng. Chem. Res. 43 (2004) 2730–2735.
[26] W.L. Luyben, Process Modelling, Simulation and Control for Chemical Engineers, McGraw Hill, New York, 1990.
[27] M. Ławryńczuk, P. Tatjewski, Approximate neural economic set-point optimisation for control systems, in: L., Rutkowski, R., Scherer, R., Tadeusiewicz, L.A., Zadeh, J., Zurada, (Eds.), Lecture Notes in Artificial Intelligence, vol. 6114, Part II: Proceedings of the 10th International Conference on Artificial Intelligence and Soft Computing, ICAISC 2010, Zakopane, Poland, Springer-Verlag, Berlin/Heidelberg, 2010, pp. 305–312.
[28] M. Ławryńczuk, Suboptimal nonlinear predictive control based on multivariable neural Hammerstein models, Appl. Intel. 32 (2010) 173–192.
[29] M. Ławryńczuk, P. Marusak, P. Tatjewski, On cooperation of set-point optimisation and predictive control based on Hammerstein models, Proceedings of the 7th Workshop on Advanced Control and Diagnosis, ACD, 2009, Zielona Góra, Poland, CD-ROM, paper no. 88.
[30] M. Ławryńczuk, P. Marusak, P. Tatjewski, Cooperation of model predictive control with steady-state economic optimisation, Control Cybernet 37 (2008) 133–158.
[31] M. Ławryńczuk, Modelling and nonlinear predictive control of a yeast fermentation biochemical reactor using neural networks, Chem. Eng. J. 145 (2008) 290–307.
[32] M. Ławryńczuk, P. Marusak, P. Tatjewski, Piecewise linear steady-state target optimization for control systems with MPC: a case study, in: Proceedings of the 17th IFAC World Congress, Seoul, South Korea, 2008, pp. 13169–13174.
[33] M. Ławryńczuk, A family of model predictive control algorithms with artificial neural networks, Int. J. Appl. Math. Comput. Sci. 17 (2007) 217–232.
[34] M. Ławryńczuk, P. Marusak, P. Tatjewski, Economic efficacy of multilayer constrained predictive control structures: an application to a MIMO neutralisation reactor, Proceedings of the 11th IFAC/IFORS/IMACS/IFIP Symposium on Large Scale Systems: Theory and Applications, LSS, 2007, Gdańsk, Poland, CD-ROM, paper no. 93.
[35] J.M. Maciejowski, Predictive Control with Constraints, Prentice Hall, Harlow, 2002.
[36] P. Marusak, Advantages of an easy to design fuzzy predictive algorithm in control systems of nonlinear chemical reactors, Appl. Soft Comput. 9 (2009) 1111–1125.
[37] T.E. Marlin, Process Control, McGraw-Hill, New York, 1995.
[38] L.A. da Cruz Meleiro, F. José, V. Zuben, R.M. Filho, Constructive learning neural network applied to identification and control of a fuel-ethanol fermentation process, Eng. Appl. Artif. Intel. 22 (2009) 201–215.
[39] M. Nørgaard, O. Ravn, N.K. Poulsen, L.K. Hansen, Neural Networks for Modelling and Control of Dynamic Systems, Springer, London, 2000.
[40] R.H. Nyström, B.M. Åkesson, H.T. Toivonen, Gain-scheduling controllers based on velocity-form linear parameter-varying models applied to an example process, Ind. Eng. Chem. Res. 41 (2002) 220–229.
[41] R.H. Nyström, K.V. Sandström, T.K. Gustafsson, H.T. Toivonen, Multimodel robust control of nonlinear plants: a case study, J. Proc. Control 9 (1999) 135–150.
[42] R.S. Patwardhan, S. Lakshminarayanan, S.L. Shah, Constrained nonlinear MPC usnig Hammerstein and Wiener models, AIChE J. 44 (1998) 1611–1622.
[43] R.K. Pearson, Selecting nonlinear model structures for computer control, J. Proc. Control 13 (2003) 1–26.
[44] S.J. Qin, T.A. Badgwell, A survey of industrial model predictive control technology, Control Eng. Pract. 11 (2003) 733–764.
[45] B.D. Ripley, Pattern Recognition and Neural Networks, Cambridge University Press, Cambridge, 1996.
[46] J.A. Rossiter, Model-based Predictive Control, CRC Press, Boca Raton, 2003.
[47] S. Skogestad, Plantwide control: the search for the self-optimizing control structure, J. Proc. Control 10 (2000) 487–507.
[48] P. Tatjewski, M. Ławryńczuk, P. Marusak, Integrated predictive optimiser and constraint supervisor for processes with basic feedback control algorithm, in: Proceedings of the European Control Conference, ECC 2009, Budapest, Hungary, 2009, pp. 3359–3364.
[49] P. Tatjewski, Advanced control and on-line process optimization in multilayer structures, Ann. Rev. Control 32 (2008) 71–85.
[50] P. Tatjewski, Advanced Control of Industrial Processes, Structures and Algorithms, Springer, London, 2007.
[51] H.T. Toivonen, K.V. Sandström, R.H. Nyström, Internal model control of nonlinear systems described by velocity-based linearizations, J. Proc. Control 13 (2003) 215–224.
[52] D.L. Yu, J.B. Gomm, Implementation of neural network predictive control to a multivariable chemical reactor, Control Eng. Pract. 11 (2003) 1315–1323.
[53] A. Zanin, M. Tvrzska de Gouvea, D. Odloak, Integrating real-time optimization into model predictive controller of the FCC system, Comput. Chem. Eng. 26 (2002) 819–831.