Int. J. Appl. Math. Comput. Sci., 2009, Vol. 19, No. 2, 233–246
DOI: 10.2478/v10006-009-0019-1

EFFICIENT NONLINEAR PREDICTIVE CONTROL BASED ON STRUCTURED NEURAL 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
e-mail: M.Lawrynczuk@ia.pw.edu.pl

This paper describes structured neural models and a computationally efficient (suboptimal) nonlinear Model Predictive Control (MPC) algorithm based on such models. The structured neural model has the ability to make future predictions of the process without being used recursively. Thanks to the nature of the model, the prediction error is not propagated. This is particularly important in the case of noise and underparameterisation. Structured models have much better long-range prediction accuracy than the corresponding classical Nonlinear Auto Regressive with eXternal input (NARX) models. The described suboptimal MPC algorithm needs solving on-line only a quadratic programming problem. Nevertheless, it gives closed-loop control performance similar to that obtained in fully-fledged nonlinear MPC, which hinges on on-line nonconvex optimisation. In order to demonstrate the advantages of structured models as well as the accuracy of the suboptimal MPC algorithm, a polymerisation reactor is studied.

Keywords: process control, model predictive control, neural networks, optimisation, linearisation.

1. Introduction

Model Predictive Control (MPC) refers to a class of computer control algorithms that directly use an explicit dynamic model in order to predict future behaviour of the process (Maciejowski, 2002; Rossiter, 2003; Tatjewski, 2007). At each sampling instant, the model is used to optimise a future control sequence, the first element of which is actually applied to the process.

MPC is recognised as the only one among advanced control techniques (defined as techniques more advanced than the PID approach) which has been exceptionally successful in numerous practical applications (Qin and Badgwell, 2003). Because the model is used to predict future behaviour of the process, MPC algorithms have the unique ability to take into account constraints imposed on both process inputs (manipulated variables) and outputs (controlled variables) or states. Constraints are very important in practice; they usually determine quality, economic efficiency and safety. Moreover, MPC techniques are very efficient in multivariable process control.

Since properties of many technological processes are nonlinear, different nonlinear MPC techniques have been developed (Henson, 1998; Morari and Lee, 1999; Qin and Badgwell, 2003; Tatjewski, 2007). In many cases, in comparison with MPC algorithms based on linear models, they make it possible to significantly improve the control quality.

The structure of the nonlinear model and the way it is used on-line affect the accuracy, computational burden and reliability of MPC. Fundamental (first-principle) models (Luyben, 1990; Marlin, 1995), although potentially very precise, are usually not suitable for on-line control. Such models are comprised of systems of differential and algebraic equations which have to be solved on-line in MPC. This may lead to numerical problems (e.g., ill-conditioning, stiffness). Moreover, in many cases, the development of fundamental models is difficult and the resulting models are very complex.

Among many structures of empirical models, neural networks (Haykin, 1999) can be effectively used on-line in different versions of MPC algorithms (Åkesson and Toivonen, 2006; Liu *et al.*, 1998; Ławryńczuk, 2007a; 2007b; Ławryńczuk and Tadej, 2008; Nørgaard *et al.*, 2000; Parisini *et al.*, 1998; Piche *et al.*, 2000; Pottmann and Seborg, 1997; Tatjewski, 2007; Tatjewski and Ławryńczuk, 2006; Trajanoski and Wach, 1998; Yu and Gomm, 2003). This is because neural networks are universal approximators (Hornik *et al.*,, 1989), and hence they may be used to approximate nonlinear behaviour of

technological dynamic processes (Hussain, 1999; Nørgaard *et al.*, 2000). In contrast to fundamental models, neural models have a simple structure and a relatively small number of parameters. Moreover, numerical problems typical of MPC algorithms based on comprehensive fundamental models are not encountered because neural models directly describe input-output relations of process variables; complicated systems of differential and algebraic equations do not have to be solved on-line.

Neural models are usually trained using the rudimentary backpropagation algorithm, which yields one-step ahead predictors. Recurrent neural network training is much more complicated. The prediction error propagation problem is a challenging theoretical and practical issue in nonlinear MPC. Models used in MPC have to be able to make good predictions not only one step ahead, but over the whole prediction horizon. If a one-step ahead predictor is used, consecutive predictions depend recursively on predictions calculated for previous sampling instants within the prediction horizon. Inevitably, the prediction error is propagated. In many cases, one-step ahead predictors are not suited to be used recursively in MPC for long-range prediction. Especially in the case of noise, model inaccuracies and underparameterisation, the order of the model used in MPC is usually significantly lower than the order of the real process, or even the proper model order is unknown. To solve the problem resulting from the inaccuracy of one-step ahead predictors in MPC, a linear multi-model approach was proposed in Liu *et al.*, 1999; Rossiter and Kouvaritakis, 2001) for processes whose properties can be precisely enough approximated by linear models. For each sampling instant within the prediction horizon, one independent linear model is used, and the prediction error is not propagated. In this work, a different approach to modelling is studied in which the structure of the model does not ignore its specific role in MPC.

The contribution of this paper is twofold. It describes structured neural models and a computationally efficient (suboptimal) nonlinear MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL) based on such models originally described in (Ławryńczuk, 2007b). Unlike classical one-step ahead predictors, the structured neural model has the ability to make future predictions of the process without being used recursively. Thanks to the nature of the model, the prediction error is not propagated. The model is easily trained by means of the classical backpropagation algorithm. In the suboptimal MPC-NPL algorithm, the model is used on-line to determine a local linearisation and a nonlinear free trajectory. Although the algorithm needs solving on-line only a quadratic programming problem, in practice it gives closed-loop control performance similar to that obtained in nonlinear MPC with on-line nonconvex optimisation repeated at each sampling instant.

This paper is organised as follows: Section 2 shortly presents the general idea of MPC and discusses prediction using classical models of the Nonlinear Auto Regressive with eXternal input (NARX) type. Next, in Section 3, the structured neural model is presented and its long-range prediction is discussed. Section 4 details the suboptimal MPC-NPL algorithm based on structured models. Section 5 demonstrates the advantages of structured neural models in long-range prediction as well as the accuracy of the suboptimal MPC-NPL algorithm in the context of a polymerisation reactor. Finally, Section 6 concludes the paper.

2. Model predictive control problem formulation

Although a number of different MPC techniques have been developed, the main idea (i.e., the explicit application of a process model, the optimisation of a cost function and the receding horizon) is always the same (Maciejowski, 2002; Rossiter, 2003; Tatjewski, 2007). At each consecutive sampling instant, k, a set of future control increments is calculated,

$$\Delta \boldsymbol{u}(k) = [\Delta u(k|k) \ldots \Delta u(k+N_u-1|k)]^T . \quad (1)$$

It is assumed that $\Delta u(k+p|k) = 0$ for $p \geq N_u$, where N_u is the control horizon. Usually, the objective is to minimise the differences between predicted values of the output $\hat{y}(k+p|k)$ and the reference trajectory $y^{ref}(k+p|k)$ over the prediction horizon N and to penalise excessive control increments. The following cost function is usually used:

$$J(k) = \sum_{p=1}^{N} \mu_p (y^{\mathrm{ref}}(k+p|k) - \hat{y}(k+p|k))^2 + \sum_{p=0}^{N_u-1} \lambda_p (\Delta u(k+p|k))^2, \quad (2)$$

where $\mu_p \geq 0$, $\lambda_p > 0$ are weighting factors. Typically, $N_u < N$. Only the first element of the determined sequence (1) is applied to the process, and the control law is

$$u(k) = \Delta u(k|k) + u(k-1). \quad (3)$$

At the next sampling instant, $k+1$, the prediction is shifted one step forward and the whole procedure is repeated.

Since constraints usually have to be taken into account, future control increments are determined from the following optimisation problem:

$$\min_{\Delta u(k|k) \ldots \Delta u(k+N_u-1|k)} \{J(k)\},$$

subject to

$$\begin{aligned} &u^{\min} \le u(k+p|k) \le u^{\max}, \quad p=0,\dots,N_u-1,\\ &-\Delta u^{\max} \le \Delta u(k+p|k) \le \Delta u^{\max},\\ &\qquad\qquad p=0,\dots,N_u-1,\\ &y^{\min} \le \hat{y}(k+p|k) \le y^{\max}, \quad p=1,\dots,N. \end{aligned} \tag{4}$$

The general prediction equation for $p = 1, \dots, N$ is

$$\hat{y}(k+p|k) = y(k+p|k) + d(k), \tag{5}$$

where the quantities $y(k+p|k)$ are calculated from a model of the process. The "DMC type" disturbance model is used in which the unmeasured disturbance $d(k)$ is assumed to be constant over the prediction horizon (Tatjewski, 2007). It is estimated from

$$d(k) = y(k) - y(k|k-1), \tag{6}$$

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

2.1. Prediction. Let the Single-Input Single-Output (SISO) process under consideration be described by the following nonlinear discrete-time Nonlinear Auto Regressive with eXternal input (NARX) model:

$$\begin{aligned} y(k) = f(\boldsymbol{x}(k)) = &f(u(k-\tau),\dots,u(k-n_B),\\ &y(k-1),\dots,y(k-n_A)), \end{aligned} \tag{7}$$

where $f : \mathbb{R}^{n_A+n_B-\tau+1} \longrightarrow \mathbb{R}$ is a nonlinear function (the integers τ, n_A, n_B define the order of the model, $\tau \le n_B$). Using the prediction equation (5) and the model (7), output predictions over the prediction horizon are calculated from

$$\begin{aligned} \hat{y}(k+p|k) = f(&\underbrace{u(k-\tau+p|k),\dots,u(k|k)}_{I_{uf}(p)},\\ &\underbrace{u(k-1),\dots,u(k-n_B+p)}_{I_u - I_{uf}(p)},\\ &\underbrace{\hat{y}(k-1+p|k),\dots,\hat{y}(k+1|k)}_{I_{yp}(p)},\\ &\underbrace{y(k),\dots,y(k-n_A+p)}_{n_A - I_{yp}(p)}) + d(k). \end{aligned} \tag{8}$$

The predictions $\hat{y}(k+p|k)$ depend on $I_{uf}(p) = \max(\min(p-\tau+1, I_u), 0)$ future values of the control signal (i.e., decision variables of the MPC algorithm), where $I_u = n_B - \tau + 1$, $I_u - I_{uf}(p)$ are the values of the control signal applied to the plant at previous sampling instants, $I_{yp}(p) = \min(p-1, n_A)$, future output predictions and $n_A - I_{yp}(p)$ are plant output signal values measured at previous sampling instants. It is evident that for prediction in the MPC algorithm the NARX model has to be used recursively, because predictions depend on predictions calculated for previous sampling instants within the prediction horizon.

Typically, during neural network training, the following Sum Squared Error (SSE) performance function is minimised:

$$SSE = \sum_{k \in \text{data set}} (y(k|k-1) - y(k))^2, \tag{9}$$

where $y(k|k-1)$ denotes the output of the model for the sampling instant k calculated from the neural model using signals up to the sampling instant $k-1$ as in (7), and $y(k)$ is the real value of the process output variable collected during the identification experiment. The obtained models are of good quality when one-step ahead prediction is necessary. Conceptually, one-step ahead predictors are not suited to be used recursively in MPC for long-range prediction because the prediction error is propagated, especially in the case of noise, model inaccuracies and underparameterisation. For many real processes the order of models used in MPC is significantly lower than the order of the real process. Very frequently, the proper model order is unknown. In spite of the fact that a one-step ahead predictor is given as the result of backpropagation training, it is used for N-step ahead prediction (8). Recurrent neural network training, although possible and used in practice, is much more complicated.

3. Structured neural models

The prediction error propagation problem in the context of MPC was thoroughly studied in (Rossiter and Kouvaritakis, 2001). A multi-model approach was proposed in (Liu *et al.*, 1999; Rossiter and Kouvaritakis, 2001), but only for linear models. For each sampling instant within the prediction horizon, one independent linear model is used, and hence the prediction error is not propagated. The idea of the structured neural model presented in this paper is to use only one model which is also able to calculate predictions over the whole prediction horizon without being used recursively. The structured model is trained by means of the classical backpropagation algorithm, in which the SSE performance function (9) is minimised.

Rewriting the model (7) for sampling instants $k-1, \dots, k-N+1$, one has

$$\begin{aligned} y(k-1) = f(&u(k-\tau-1),\dots,u(k-n_B-1),\\ &y(k-2),\dots,y(k-n_A-1)), \end{aligned} \tag{10}$$

$$\begin{aligned} y(k-2) = f(&u(k-\tau-2),\dots,u(k-n_B-2)\\ &y(k-3),\dots,y(k-n_A-2)), \end{aligned} \tag{11}$$

$$\vdots$$

$$
\begin{aligned}
&y(k-N+2)\\
&\quad = f(u(k-\tau-N+2),\ldots,u(k-n_B-N+2),\\
&\qquad y(k-N+1),\ldots,y(k-n_A-N+2)),
\end{aligned}
\tag{12}
$$

$$
\begin{aligned}
&y(k-N+1)\\
&\quad = f(u(k-\tau-N+1),\ldots,u(k-n_B-N+1),\\
&\qquad y(k-N),\ldots,y(k-n_A-N+1)).
\end{aligned}
\tag{13}
$$

Using (13), the quantity $y(k-N+2)$ given by (11) can be expressed as

$$
\begin{aligned}
&y(k-N+2)\\
&\quad = f(u(k-\tau-N+2),\ldots,u(k-n_B-N+2),\\
&\qquad f(u(k-\tau-N+1),\ldots,u(k-n_B-N+1),\\
&\qquad y(k-N),\ldots,y(k-n_A-N+1)),\\
&\qquad y(k-N),\ldots,y(k-n_A-N+2)),
\end{aligned}
\tag{14}
$$

which can be rewritten as the function

$$
\begin{aligned}
&y(k-N+2)\\
&\quad = f_{N-2}(u(k-\tau-N+2),\ldots,u(k-n_B-N+1),\\
&\qquad y(k-N),\ldots,y(k-n_A-N+1)).
\end{aligned}
\tag{15}
$$

Model arguments rearrangement can be repeated for all quantities $y(k-N+2),\ldots,y(k)$, giving functions $f_{N-2},\ldots,f_0$. Finally, one has

$$
\begin{aligned}
y(k) = f(&u(k-\tau),\ldots,u(k-n_B),\\
&f_1(u(k-\tau-1),\ldots,u(k-n_B-N+1),\ldots,\\
&y(k-N),\ldots,y(k-n_A-N+1)),\ldots,\\
&f_{n_A}(u(k-\tau-n_A),\ldots,u(k-n_B-N+1),\\
&y(k-N),\ldots,y(k-n_A-N+1))),
\end{aligned}
\tag{16}
$$

which can be rewritten as the function

$$
\begin{aligned}
y(k) = f_0(&u(k-\tau),\ldots,u(k-n_B-N+1),\\
&y(k-N),\ldots,y(k-n_A-N+1)).
\end{aligned}
\tag{17}
$$

The obtained equation (17) represents the structured model, and $f_0 : \mathbb{R}^{n_A+n_B-\tau+N} \longrightarrow \mathbb{R}$ is a nonlinear function, $\tau \leq n_B + N - 1$.

3.1. Prediction. Using the prediction equation (5), output predictions calculated from the structured model (17) are

$$
\begin{aligned}
&\hat{y}(k+p|k)\\
&\quad = f_0(\underbrace{u(k-\tau+p|k),\ldots,u(k|k)}_{I_{uf}(p)},\\
&\qquad \underbrace{u(k-1),\ldots,u(k-n_B-N+1+p)}_{I_u-I_{uf}(p)},\\
&\qquad \underbrace{y(k-N+p),\ldots,y(k-n_A-N+1+p)}_{n_A})\\
&\quad + d(k).
\end{aligned}
\tag{18}
$$

For the structured model, $I_u = n_B + N - \tau$. As in the case of the classical NARX model (7) used for prediction (8), predictions $\hat{y}(k+p|k)$ calculated by means of the structured model depend on $I_{uf}(p)$ future values of the control signal and $I_u - I_{uf}(p)$ values of the control signal applied to the plant at previous sampling instants. Unlike the classical NARX predictions, they do not depend on predictions calculated for previous sampling instants within the prediction horizon, but only on n_A values of the plant output signal measured at previous sampling instants. As a result, the structured model is not used recursively and the prediction error is not propagated.

There is a clear link between the discussed structured model (17) and other types of models used in MPC. More specifically, the Dynamic Matrix Control (DMC) algorithm (Cutler and Ramaker, 1979; Tatjewski, 2007) uses step-response linear models in which the output signal depends only on the control signal. Consequently, the prediction error is not propagated. Because of their nature, step-response models need many coefficients (usually dozens or even hundreds), much more than ARX models of a similar accuracy, which depend on both input and output signals. As a result, in the DMC algorithm relatively long prediction horizons (and horizons of dynamics) should be used. Of course, it is possible to develop a nonlinear DMC algorithm which uses nonlinear step-response models, for example, of a neural type. Unfortunately, such models are likely to have many parameters, which unnecessarily complicates the whole control algorithm.

On the other hand, in the structured model the output signal depends not only on the control signal but also on previous values of the output signal in such a way that the prediction error is not propagated. Such an approach has two advantages. Intuitively, due to its dependence on previous values of the output, the structured model depends on a smaller number of past values of the input signal than the step-response model does. Secondly, in the MPC algorithm described in the following part of the article, short prediction horizons can be used, similarly as is possible in the GPC algorithm (Clarke and Mohtadi, 1989). Yet another model type whose output signal depends only on the control signal is the nonlinear Volterra system (Doyle and Ogunnaike, 2001; Doyle *et al.*, 1995). Unfortunately, Volterra models, similarly as step-response ones, are usually very complicated.

A Multi Layer Perceptron (MLP) feedforward neural network with one hidden layer and a linear output (Haykin, 1999) is used as the function f_0 in (17). The output of the model is

$$
y(k) = f(\boldsymbol{x}(k)) = w_0^2 + \sum_{i=1}^{K} w_i^2 \varphi(z_i(k)),
\tag{19}
$$

where $z_i(k)$ is the sum of inputs of the i-th hidden node,

$\varphi : \mathbb{R} \longrightarrow \mathbb{R}$ is a nonlinear transfer function (e.g., hyperbolic tangent), K is the number of hidden nodes. Taking into account arguments of the structured model (17),

$$z_i(k) = w^1_{i,0} + \sum_{j=1}^{I_u} w^1_{i,j} u(k-\tau+1-j) + \sum_{j=1}^{n_A} w^1_{i,I_u+j} y(k-j-N+1). \quad (20)$$

Weights of the network are denoted by $w^1_{i,j}$, $i = 1, \ldots, K$, $j = 0, \ldots, n_A + n_B - \tau + N$, and w^2_i, $i = 0, \ldots, K$, for the first and the second layer, respectively.

Using (5) and (20), predictions calculated from the structured model are

$$\hat{y}(k+p|k) = w^2_0 + \sum_{i=1}^{K} w^2_i \varphi(z_i(k+p|k)) + d(k), \quad (21)$$

where, using (18), one has

$$\begin{aligned} z_i(k+p|k) &= w^1_{i,0} + \sum_{j=1}^{I_{uf}(p)} w^1_{i,j} u(k-\tau+1-j+p|k) \\ &+ \sum_{j=I_{uf}(p)+1}^{I_u} w^1_{i,j} u(k-\tau+1-j+p) \\ &+ \sum_{j=1}^{n_A} w^1_{i,I_u+j} y(k-j-N+1+p). \end{aligned} \quad (22)$$

From (6) and (19), the unmeasured disturbance is

$$d(k) = y(k) - \left(w^2_0 + \sum_{i=1}^{K} w^2_i \varphi(z_i(k)) \right). \quad (23)$$

4. MPC algorithm with nonlinear prediction and linearisation based on structured neural models

In general, two approaches to nonlinear MPC can be distinguished: MPC with Nonlinear Optimisation (MPC-NO) and suboptimal MPC. If for prediction a nonlinear model (e.g., a neural one) is used without any simplifications, at each sampling instant a nonlinear optimisation problem (4) has to be solved on-line (Ławryńczuk, 2007a; Tatjewski, 2007; Tatjewski and Ławryńczuk, 2006). Although the MPC-NO algorithm potentially seems to be very accurate, the difficulty of the resulting optimisation problem is twofold. First of all, it is nonlinear and computationally demanding, and the computational burden is high. Secondly, it may be nonconvex and even multimodal.

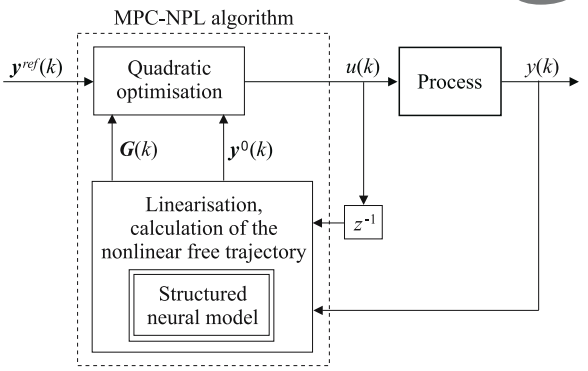

Fig. 1. Structure of the MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL).

4.1. MPC-NPL optimisation problem. The MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL) (Ławryńczuk, 2007a; Tatjewski, 2007; Tatjewski and Ławryńczuk, 2006) which needs solving on-line only a quadratic programming problem is adopted. The structure of the algorithm is shown in Fig. 1. At each sampling instant k the neural model is used on-line twice: to find a local linearisation and a nonlinear free trajectory. It is assumed that the output prediction can be expressed as the sum of the forced trajectory, which depends only on the future (on future input moves $\Delta \boldsymbol{u}(k)$) and the free trajectory $\boldsymbol{y}^0(k)$, which depends only on the past

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{G}(k)\Delta \boldsymbol{u}(k) + \boldsymbol{y}^0(k), \quad (24)$$

where

$$\hat{\boldsymbol{y}}(k) = [\hat{y}(k+1|k) \ldots \hat{y}(k+N|k)]^T, \quad (25)$$

$$\boldsymbol{y}^0(k) = \left[y^0(k+1|k) \ldots y^0(k+N|k)\right]^T. \quad (26)$$

The $N \times N_u$ dynamic matrix $\boldsymbol{G}(k)$ is comprised of step-response coefficients of the linearised model calculated on-line taking into account the current state of the process,

$$\boldsymbol{G}(k) = \begin{bmatrix} s_1(k) & 0 & \ldots & 0 \\ s_2(k) & s_1(k) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N(k) & s_{N-1}(k) & \ldots & s_{N-N_u+1}(k) \end{bmatrix}. \quad (27)$$

The calculation of the step-response and of the nonlinear free trajectory is detailed in the following subsection.

Because in (24) it is assumed that the future output prediction is a linear function of future input increments $\Delta \boldsymbol{u}(k)$, the general nonlinear MPC optimisation problem (4) becomes the following quadratic programming task:

$$\min_{\Delta \boldsymbol{u}(k),\, \boldsymbol{\varepsilon}^{\min},\, \boldsymbol{\varepsilon}^{\max}} \Big\{ \left\| \boldsymbol{y}^{\mathrm{ref}}(k) - \boldsymbol{G}(k)\Delta \boldsymbol{u}(k) - \boldsymbol{y}^0(k) \right\|^2_{\boldsymbol{M}} + \|\Delta \boldsymbol{u}(k)\|^2_{\boldsymbol{\Lambda}} + \rho^{\min} \left\| \boldsymbol{\varepsilon}^{\min} \right\|^2 + \rho^{\max} \left\| \boldsymbol{\varepsilon}^{\max} \right\|^2 \Big\},$$

Table 1. Parameters of the fundamental model.

Parameter	Value	Parameter	Value
$C_{I_{in}}$	8 kmol m^{-3}	R	8.314 kJ kmol^{-1} K^{-1}
$C_{m_{in}}$	6 kmol/m^{-3}	T	335 K
E_{T_c}	2.9442×10^3 kJ kmol^{-1}	Z_{T_c}	3.8223×10^{10} m^3 kmol^{-1} h^{-1}
E_{T_d}	2.9442×10^3 kJ kmol^{-1}	Z_{T_d}	3.1457×10^{11} m^3 kmol^{-1} h^{-1}
E_{f_m}	7.4478×10^4 kJ kmol^{-1}	Z_{f_m}	1.0067×10^{15} m^3 kmol^{-1} h^{-1}
E_I	1.2550×10^5 kJ kmol^{-1}	Z_I	3.7920×10^{18} h^{-1}
E_P	1.8283×10^4 kJ kmol^{-1}	Z_P	1.7700×10^9 m^3 kmol^{-1} h^{-1}
f^*	0.58	V	0.1 m^3
M_m	100.12 kg kmol^{-1}		

subject to

$$\begin{aligned} \boldsymbol{u}^{\min} \le \boldsymbol{J}\Delta\boldsymbol{u}(k) + \boldsymbol{u}^{k-1}(k) \le \boldsymbol{u}^{\max}, \\ -\Delta\boldsymbol{u}^{\max} \le \Delta\boldsymbol{u}(k) \le \Delta\boldsymbol{u}^{\max}, \\ \boldsymbol{y}^{\min} - \boldsymbol{\varepsilon}^{\min} \le \boldsymbol{G}(k)\Delta\boldsymbol{u}(k) + \boldsymbol{y}^0(k) \le \boldsymbol{y}^{\max} + \boldsymbol{\varepsilon}^{\max}, \\ \boldsymbol{\varepsilon}^{\min} \ge 0, \quad \boldsymbol{\varepsilon}^{\max} \ge 0, \end{aligned} \tag{28}$$

where

$$\boldsymbol{y}^{\mathrm{ref}}(k) = \left[y^{\mathrm{ref}}(k+1|k) \dots y^{\mathrm{ref}}(k+N|k)\right]^T, \tag{29}$$

$$\boldsymbol{y}^{\min} = \left[y^{\min} \dots y^{\min}\right]^T, \tag{30}$$

$$\boldsymbol{y}^{\max} = \left[y^{\max} \dots y^{\max}\right]^T \tag{31}$$

are N-element,

$$\boldsymbol{u}^{\min} = \left[u^{\min} \dots u^{\min}\right]^T, \tag{32}$$

$$\boldsymbol{u}^{\max} = \left[u^{\max} \dots u^{\max}\right]^T, \tag{33}$$

$$\Delta\boldsymbol{u}^{\max} = \left[\Delta u^{\max} \dots \Delta u^{\max}\right]^T, \tag{34}$$

$$\boldsymbol{u}^{k-1}(k) = \left[u(k-1) \dots u(k-1)\right]^T \tag{35}$$

are N_u-element vectors, $\boldsymbol{J}$ is the all-ones lower triangular $N_u \times N_u$ matrix, $\boldsymbol{M} = \mathrm{diag}(\mu_1, \dots, \mu_N)$ and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_0, \dots, \lambda_{N_u-1})$.

If output constraints have to be taken into account, the MPC optimisation task (4) may be affected by the infeasibility problem. That is why in (28) output constraints are softened by slack variables (Maciejowski, 2002; Tatjewski, 2007). A quadratic penalty for constraint violations is used, $\boldsymbol{\varepsilon}^{\min}$ and $\boldsymbol{\varepsilon}^{\max}$ are vectors of length N comprising slack variables, and $\rho^{\min}$, $\rho^{\max} > 0$ are weights.

In the MPC-NPL algorithm (Fig. 1), at each sampling instant k the following steps are repeated:

1. Linearisation of the structured neural model: obtain the matrix $\boldsymbol{G}(k)$.
2. Find the nonlinear free trajectory $\boldsymbol{y}^0(k)$ using the structured neural model.
3. Solve the quadratic programming problem (28) to determine $\Delta\boldsymbol{u}(k)$.
4. Apply $u(k) = \Delta u(k|k) + u(k-1)$.
5. Set $k := k+1$, go to Step 1.

Although in practice the stability of the MPC-NPL algorithm can be achieved by proper tuning of the prediction horizon and weighting coefficients μ_p, λ_p, it can be also combined with the stabilising dual-mode approach, similarly as in the case of the MPC-NPL algorithm based on neural NARX models (Ławryńczuk and Tadej, 2008). If the MPC-NPL algorithm is implemented without any constraints, the optimal future control moves can be easily calculated analytically, without any on-line optimisation. In such a case, the controller unconstrained optimisation problem is formulated as a least-squares one and solved by means of the numerically reliable Singular Value Decomposition (SVD), similarly as can be done in the unconstrained MPC algorithms based on linear models (Maciejowski, 2002; Tatjewski, 2007).

4.2. On-line linearisation of the structured neural model and calculation of the nonlinear free trajectory.

Defining the linearisation point as a vector composed of past input and output signal values corresponding to the arguments of the structured neural model (17),

$$\begin{aligned} \bar{\boldsymbol{x}}(k) = [&\bar{u}(k-\tau) \dots \bar{u}(k-n_B-N+1) \\ &\bar{y}(k-N) \dots \bar{y}(k-n_A-N+1)]^T, \end{aligned} \tag{36}$$

and using Taylor series expansion at this point, the linear approximation of the model, obtained at a sampling instant k, can be expressed as

$$\begin{aligned} y(k) = &f_0(\bar{\boldsymbol{x}}(k)) \\ &+ \sum_{l=\tau}^{n_B+N-1} b_l(\bar{\boldsymbol{x}}(k))(u(k-l) - \bar{u}(k-l)) \\ &- \sum_{l=N}^{n_A+N-1} a_l(\bar{\boldsymbol{x}}(k))(y(k-l) - \bar{y}(k-l)), \end{aligned} \tag{37}$$

where

$$a_l(\bar{\boldsymbol{x}}(k)) = -\frac{\partial f_0(\bar{\boldsymbol{x}}(k))}{\partial y(k-l)},$$
$$b_l(\bar{\boldsymbol{x}}(k)) = \frac{\partial f_0(\bar{\boldsymbol{x}}(k))}{\partial u(k-l)}$$

are coefficients of the linearised model.

Taking into account the structured neural model described by (19) and (20), these coefficients are calculated from

$$a_l(\bar{\boldsymbol{x}}(k)) = \begin{cases} 0 & \text{if} \quad l = 1, \ldots, N-1, \\ -\sum_{i=1}^{K} w_i^2 \dfrac{\mathrm{d}\varphi(z_i(\bar{\boldsymbol{x}}(k)))}{\mathrm{d}z_i(\bar{\boldsymbol{x}}(k))} w_{i,I_u+l-N+1}^1 & \\ & \text{if} \quad l = N, \ldots, n_A+N-1, \end{cases} \tag{38}$$

and

$$b_l(\bar{\boldsymbol{x}}(k)) = \begin{cases} 0 & \text{if} \quad l = 1, \ldots, \tau-1, \\ \sum_{i=1}^{K} w_i^2 \dfrac{\mathrm{d}\varphi(z_i(\bar{\boldsymbol{x}}(k)))}{\mathrm{d}z_i(\bar{\boldsymbol{x}}(k))} w_{i,l-\tau+1}^1 & \\ & \text{if} \quad l = \tau, \ldots, n_B+N-1. \end{cases} \tag{39}$$

If hyperbolic tangent is used as the nonlinear transfer function φ in the hidden layer of the model,

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

.

Step-response coefficients of the linearised model are

$$s_j(k) = \sum_{l=1}^{\min(j,n_B+N-1)} b_l(\bar{\boldsymbol{x}}(k)) - \sum_{l=1}^{\min(j-1,n_A+N-1)} a_l(\bar{\boldsymbol{x}}(k)) s_{j-l}(k). \tag{40}$$

The nonlinear free trajectory $y^0(k+p|k)$, $p = 1, \ldots, N$, is calculated on-line analogously as in the general prediction equation (21) but taking into account only the influence of the past,

$$y^0(k+p|k) = w_0^2 + \sum_{i=1}^{K} w_i^2 \varphi(z_i^0(k+p|k)) + d(k). \tag{41}$$

The quantities $z_i^0(k+p|k)$ are determined from (22) assuming no changes in the control signal from a sampling instant k onwards, i.e., $u(k+p|k) := u(k-1)$ for $p \geq 0$.

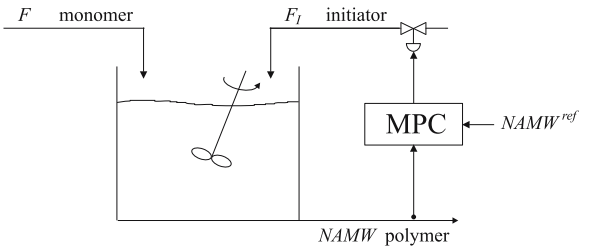

Fig. 2. The polymerisation reactor control system structure.

Table 2. Parameters of neural models before pruning, N_w—the number of weights.

Model	SSE_{training}	SSE_{test}	N_w
NARX	0.3671	0.5618	25
Structured for $N = 5$	0.3788	1.2164	41
Structured for $N = 10$	0.3204	0.8568	66
Structured for $N = 15$	0.1910	0.3312	91

Table 3. Parameters of neural models after pruning, N_w—the number of weights.

Model	SSE_{training}	SSE_{test}	N_w
NARX	0.3762	0.5582	21
Structured for $N = 5$	0.3824	1.1434	28
Structured for $N = 10$	0.3289	0.8501	34
Structured for $N = 15$	0.1993	0.3198	39

One has

$$\begin{aligned} z_i^0(k+p|k) = & w_{i,0}^1 + \sum_{j=1}^{I_{uf}(p)} w_{i,j}^1 u(k-1) \\ & + \sum_{j=I_{uf}(p)+1}^{I_u} w_{i,j}^1 u(k-\tau+1-j+p) \\ & + \sum_{j=1}^{n_A} w_{i,I_u+j}^1 y(k-j-N+1+p). \end{aligned} \tag{42}$$

5. Experiments

5.1. Polymerisation reactor control system. The process under consideration is a polymerisation reaction taking place in a jacketed continuous stirred tank reactor (Doyle *et al.*, 1995) depicted in Fig. 2. The reaction is t free-radical polymerisation of methyl methacrylate with azo-bis-isobutyronitrile as the initiator and toluene as the solvent. The output $NAMW$ (Number Average Molecular Weight) [kg kmol^{-1}] is controlled by manipulating the inlet initiator flow rate F_I [m^3 h^{-1}]. The flow rate F [m^3 h^{-1}] of the monomer is a disturbance. Because both steady-state and dynamic properties of the polymerisation reactor are nonlinear, it is frequently used as a benchmark

for comparing nonlinear control strategies (Doyle *et al.*, 1995; Ławryńczuk, 2007a; 2007b; Tatjewski, 2007).

Assuming isothermal operation, perfect mixing, constat heat capacity, no polymer in the inlet stream, no gel effect, constant reactor volume, negligible initiator flow rate (in comparison with monomer flow rate) and quasisteady state and long-chain hypothesis, the continuous-time fundamental model of the polymerisation reactor is comprised of four nonlinear ordinary differential equations,

$$\frac{\mathrm{d}C_m(t)}{\mathrm{d}t} = -\left[Z_P \exp\left(\frac{-E_P}{RT}\right) + Z_{f_m}\exp\left(\frac{-E_{f_m}}{RT}\right)\right] C_m(t)P_0(t) - \frac{F(t)C_m(t)}{V} + \frac{F(t)C_{m_{in}}}{V}, \tag{43}$$

$$\frac{\mathrm{d}C_I(t)}{\mathrm{d}t} = -Z_I \exp\left(\frac{-E_I}{RT}\right) C_I(t) - \frac{F(t)C_I}{V} + \frac{F_I(t)C_{I_{in}}}{V}, \tag{44}$$

$$\frac{\mathrm{d}D_0(t)}{\mathrm{d}t} = \left[0.5Z_{T_c}\exp\left(\frac{-E_{T_c}}{RT}\right) + Z_{T_d}\exp\left(\frac{-E_{T_d}}{RT}\right)\right] P_0^2(t) + Z_{f_m}\exp\left(\frac{-E_{f_m}}{RT}\right) C_m(t)P_0(t) - \frac{F(t)D_0(t)}{V}, \tag{45}$$

$$\frac{\mathrm{d}D_I(t)}{\mathrm{d}t} = M_m\left[Z_P \exp\left(\frac{-E_P}{RT}\right) + Z_{f_m}\exp\left(\frac{-E_{f_m}}{RT}\right)\right] C_m(t)P_0(t) - \frac{F(t)D_I(t)}{V}, \tag{46}$$

where

$$P_0(t) = \sqrt{\frac{2f^* C_I(t) Z_I \exp\left(\frac{-E_I}{RT}\right)}{Z_{T_d}\exp\left(\frac{-E_{T_d}}{RT}\right) + Z_{T_c}\exp\left(\frac{-E_{T_c}}{RT}\right)}}, \tag{47}$$

and the algebraic output equation

$$NAMW(t) = \frac{D_I(t)}{D_0(t)}. \tag{48}$$

State variables are C_m—the monomer concentration and C_I—the initiator concentration, and D_I/D_0 is the number average molecular weight. The initial operating conditions are $F_I = 0.028328$ m^3 h^{-1}, $F = 1$ m^3 h^{-1}, $NAMW = 20000$ kg kmol^{-1}, $C_m = 5.3745$ kmol m^{-3}, $C_I = 2.2433 \times 10^{-1}$ kmol m^{-3}, $D_0 = 3.1308 \times 10^{-3}$ kmol m^{-3}, $D_I = 6.2616 \times 10^{-1}$ kmol m^{-3}. Parameters of the fundamental model are given in Table 1.

5.2. Neural modelling of the polymerisation reactor. During the identification experiment, the fundamental model (43)–(48) is used as the real process, and it is simulated in open loop in order to obtain two sets of data, namely, training and test data sets, depicted in Fig. 3. Both sets contain 2000 samples, and the sampling time is 1.8 min. The output signal contains noise. During calculations the system of differential equations comprising the fundamental model is solved using the Runge-Kutta RK45 method.

Four model structures are trained: an NARX (one-step ahead) model (7) and structured models (17) for $N = 5$, $N = 10$, $N = 15$. A delayed first order NARX model is used,

$$y(k) = f(u(k-2), y(k-1)), \tag{49}$$

i.e., $\tau = n_B = 2$, $n_A = 1$. In structured models, the same parameters τ, n_A, n_B are used. To show the advantages of structured models all four models are underparametrised, because in fact the fundamental model (43)–(48) consists of four differential equations. As previous research shows (Ławryńczuk, 2007a; Tatjewski, 2007), in order to precisely capture the nature of the process, the NARX model should be of at least the second order,

$$y(k) = f(u(k-2), y(k-1), y(k-2)), \tag{50}$$

i.e., $n_A = n_B = \tau = 2$. Since input and output process variables have different orders of magnitude, they are scaled as $u = 100(F_I - F_{I0})$, $y = 0.0001(NAMW - NAMW_0)$, where $F_{I0} = 0.028328$, $NAMW_0 = 20000$ correspond to the initial operating point.

The NARX model has $K = 6$ hidden nodes while structured models have $K = 5$ nodes. All neural models are trained as one-step ahead predictors using the BFGS optimisation algorithm (Bazaraa *et al.*, 1993) which minimises the Sum Squared Error (SSE) performance index (9). For each neural model the identification experiment is repeated 10 times and weights of neural networks are initialised randomly. The results presented are the best ones obtained. Table 2 compares the accuracy of models for training and data sets in terms of SSE and the number of weights. Naturally, the longer the prediction horizon N, the bigger the number of models' parameters (weights). In order to reduce the complexity of models, the Optimal Brain Damage (OBD) pruning algorithm is used (LeCun *et al.*,, 1989). Parameters of pruned models are compared in Table 3. The complexity of the NARX model is reduced by only 16.0% whereas in the case of structured models by 31.7%, 48.5% and 57.14% for models with $N = 5$, $N = 10$, $N = 15$, respectively.

Because when trained and tested as one-step ahead predictors both NARX and structured model classes give comparable SSE (i.e., of the same order), it is interesting to compare their step-responses. Figure 4 shows responses

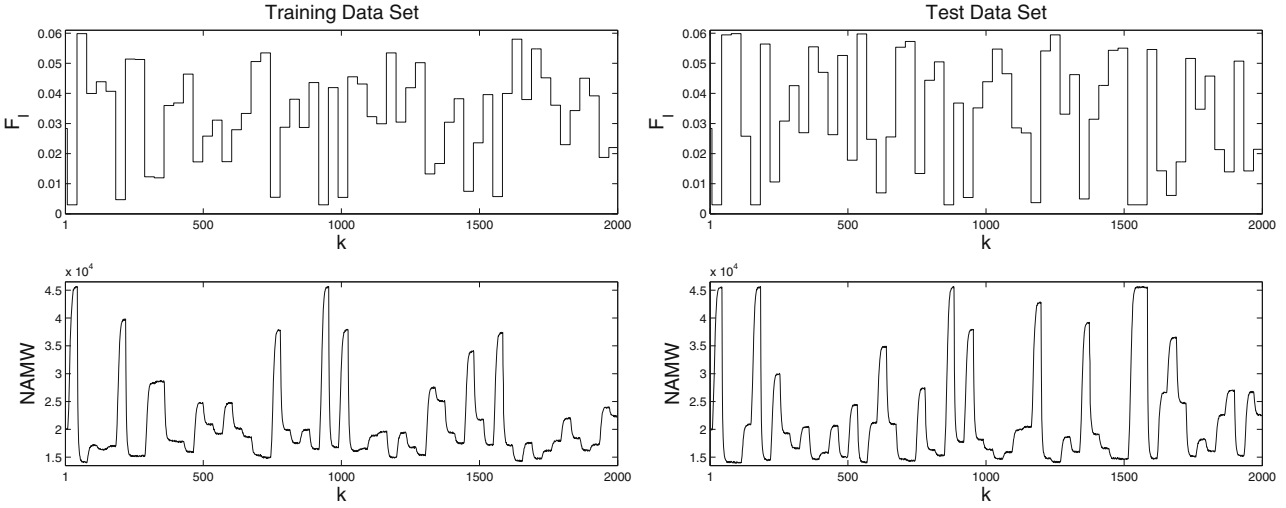

Fig. 3. Training and test data sets.

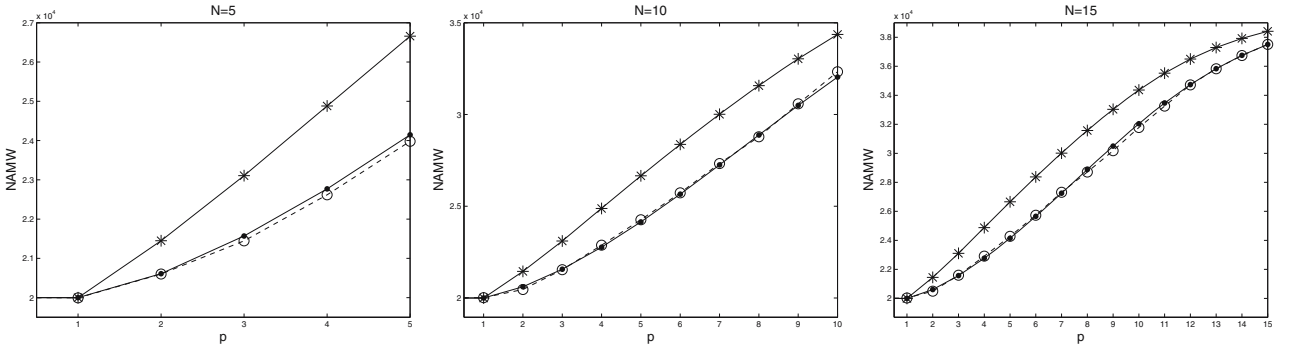

Fig. 4. Step-responses (long-range predictions) calculated by the classical NARX neural model (*solid line with asterisks*) and by the structured neural model (*dashed line with circles*) vs. the real process (*solid line with points*).

Table 4. Average accuracy ratios of structured models in comparison with the NARX model: the training data set.

Model	R_5	R_{10}	R_{15}
Structured for $N = 5$	0.5737	–	–
Structured for $N = 10$	0.4320	0.2950	–
Structured for $N = 15$	0.2541	0.1717	0.1367

Table 5. Average accuracy ratios of structured models in comparison with the NARX model: the test data set.

Model	R_5	R_{10}	R_{15}
Structured for $N = 5$	0.9100	–	–
Structured for $N = 10$	0.5658	0.3554	–
Structured for $N = 15$	0.2398	0.1499	0.1150

of the process and predictions. The manipulated variable F_I changes at the sampling instant $k = 0$ from 0.028328 to 0.004602, which corresponds to changing the operating point from $NAMW = 20000$ to $NAMW = 40000$. The one-step ahead NARX neural model is used recurrently, and it correctly calculates only the first prediction (for $k = 1$). As a result of underparameterisation, for the next sampling instants the prediction error is propagated and consecutive predictions significantly differ from the real process. Conversely, structured neural models have fundamentally better prediction abilities, and differences between the process and predictions are very small. The prediction error is not propagated, and the models predict correctlythe behaviour of the process over the whole prediction horizon.

In order to further compare the long-range prediction accuracy and show the potential of using structured neural models for long-range prediction in MPC, the ratio

$$R_N = \frac{1}{N}\sum_{p=1}^{N} \frac{\sum\limits_{k\in\text{data set}} (y(k+p|k) - y(k+p))^2}{\sum\limits_{k\in\text{data set}} (y_{NARX}(k+p|k) - y(k+p))^2} \tag{51}$$

is considered. The coefficient R_N compares the average long-range prediction accuracy of the structured neural model (the numerator) and the classical NARX model (the denominator). The output of the classical one-step ahead model used for long-range prediction is denoted by $y_{NARX}(k+p|k)$, the output of the structured model is denoted by $y(k+p|k)$, $y(k+p)$ is the real data.

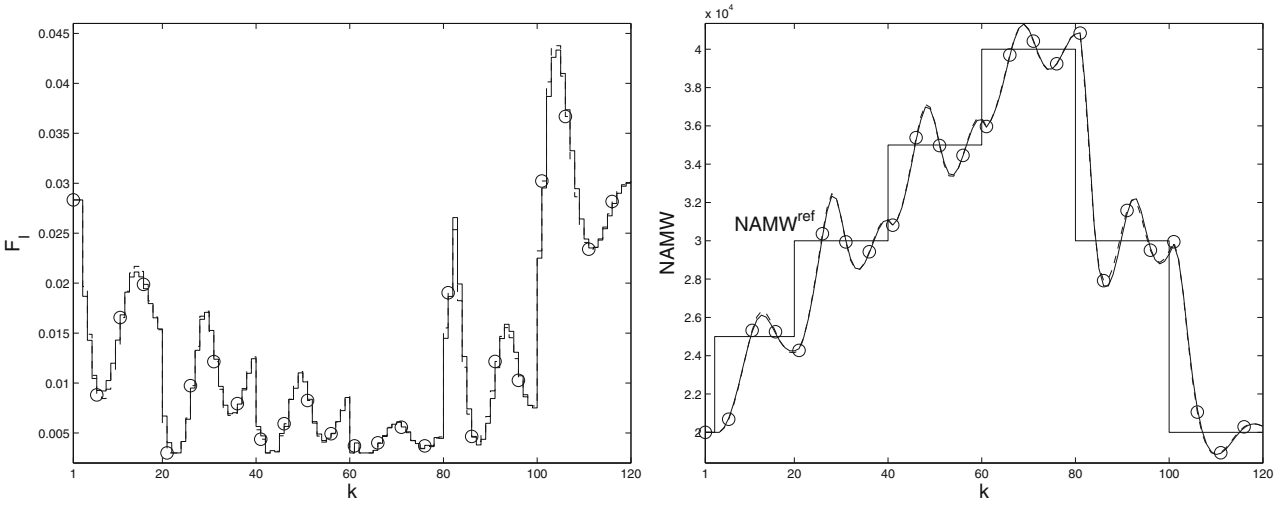

Fig. 5. Simulation results: the MPC-NPL algorithm (*dashed line with circles*) and the MPC-NO algorithm (*solid line*) based on the same NARX neural model.

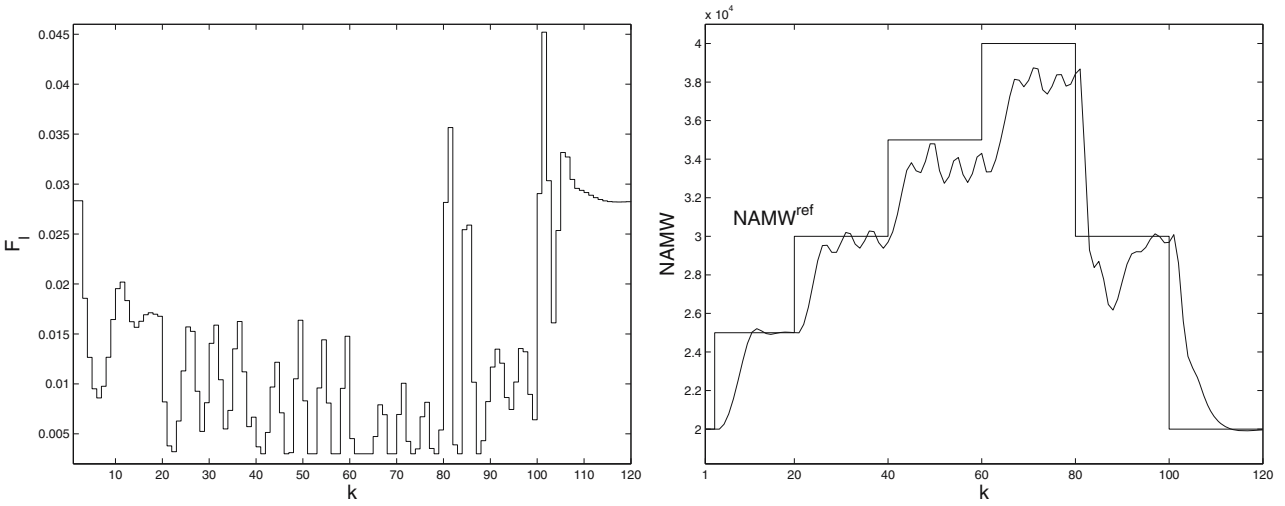

Fig. 6. Simulation results: the MPC algorithm based on the linear model.

If $R_N < 1$, it is clear that there is a potential for using in MPC structured models rather than classical NARX models because the former have better long-range prediction abilities. The smaller the value of R_N, the worse long-range prediction abilities of classical NARX models and it is more appropriate to use structured models in MPC. Tables 4 and 5 present values of the ratio R_N of structured models trained for different prediction horizons. In general, the longer the prediction horizon, the worse the prediction accuracy of the NARX model. At the same time, structured models are characterised by good long-range prediction accuracy. Of course, it is possible to calculate predictions from a structured model for a shorter horizon than used during training.

5.3. MPC of the polymerisation reactor. The fundamental model (43)–(48) is used as the real process during simulations of MPC algorithms. The model is solved using the Runge-Kutta RK45 method. The horizons of MPC are $N = 5$ or $N = 10$, $Nu = 3$, and the weighting coefficients are $\mu_p = 1$, $\lambda_p = 0.2$. (As far as choosing parameters of MPC, there are many tuning criteria in the literature (Clarke and Mohtadi, 1989; Scattolini and Bittanti, 1990); this issue is not considered here.) The manipulated variable is constrained, $F_I^{\min} = 0.003$, $F_I^{\max} = 0.06$.

Because of a highly nonlinear nature of the polymerisation reactor, the MPC algorithm based on a linear model is unable to control the process efficiently as shown in Fig. 6 and discussed in (Ławryńczuk, 2007a; Tatjewski, 2007). As the reference trajectory ($NAMW^{\mathrm{ref}}$), six set-point changes are considered. The algorithm works satisfactorily for the smallest set-point change, but for bigger ones the system becomes unstable. This is so because dynamic and steady-state properties of the process are

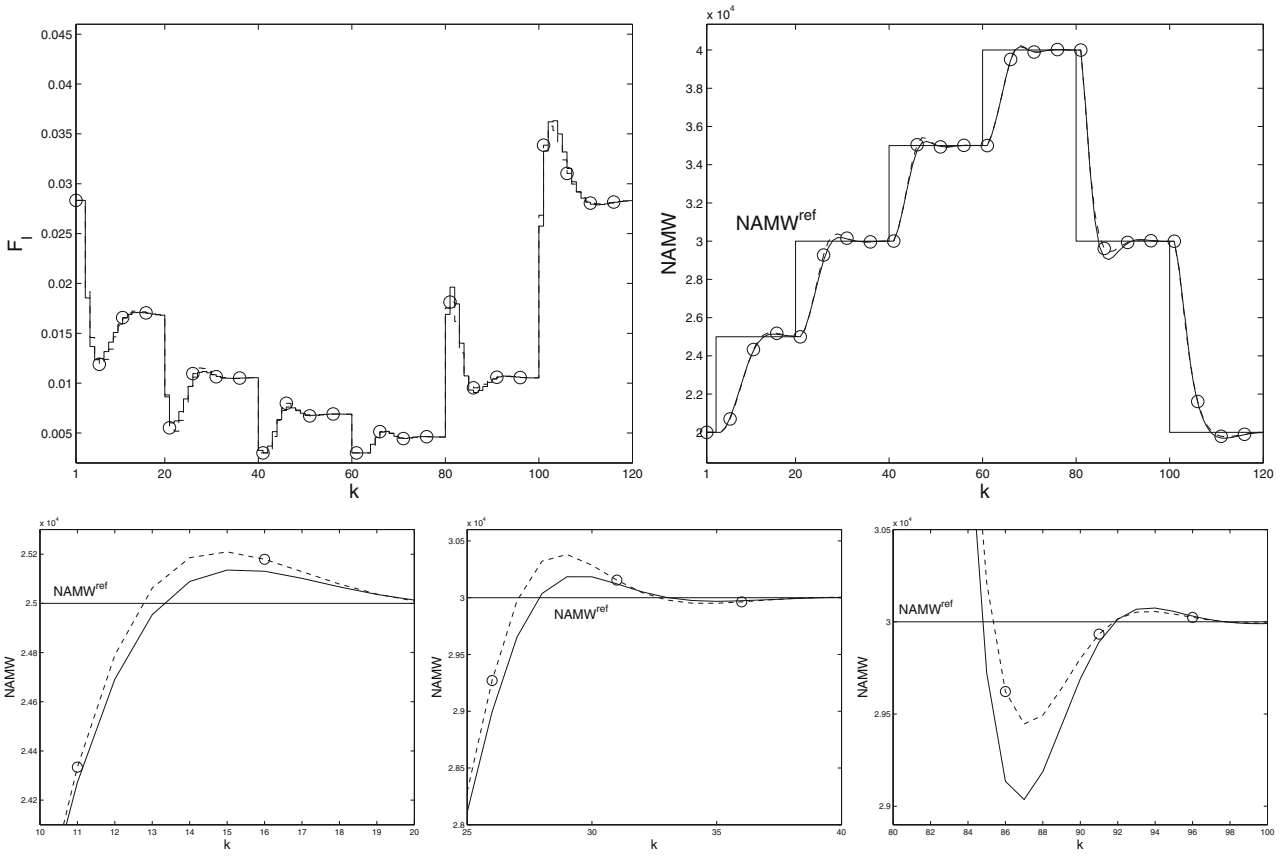

Fig. 7. Simulation results: the MPC-NPL algorithm (*dashed line with circles*) and the MPC-NO algorithm (*solid line*) based on the same structured neural model, *top*: the whole simulation, *bottom*: enlarged fragments.

nonlinear. Hence, it is justified to use nonlinear models in MPC. The second order linear model used in this simulation is not underparameterised (i.e., $\tau = n_A = n_B = 2$), and the role of this experiment is to justify the necessity of using nonlinear models of the discussed polymerisation process in MPC.

In the following part of the article, both classes of neural models (i.e., NARX and structured models) under consideration are underparameterised (i.e., $\tau = n_B = 2$, $n_A = 1$). To emphasise the accuracy and computational efficiency of the MPC-NPL algorithm based on the structured model, four MPC algorithms are compared:

(a) suboptimal MPC-NPL algorithm with on-line quadratic programming based on the NARX neural model,
(b) MPC algorithm with on-line Nonlinear Optimisation (MPC-NO) based on the NARX neural model,
(c) suboptimal MPC-NPL algorithm based on the structured neural model,
(d) MPC-NO algorithm based on the structured neural model.

In the MPC-NO algorithm, Sequential Quadratic Programming (SQP) (Bazaraa *et al.*, 1993) is used.

It is interesting to evaluate MPC based on the classical NARX neural model whose long-range prediction accuracy is poor in comparison with structured models (Fig. 4, Tables 4 and 5). Simulation results are shown in Fig. 5. Although the NARX model is trained as a one-step ahead predictor, it is used recurrently and the resulting MPC algorithms work unsatisfactorily. Both nonlinear MPC-NPL and MPC-NO algorithms give similar transient responses; very low control accuracy results from the inaccurate long-range prediction.

Simulation results of the MPC-NPL algorithm and the MPC-NO algorithm based on the same structured neural model (trained for $N = 10$) are depicted in Fig. 7. Both nonlinear algorithms are stable. Moreover, for all six set point changes considered the closed-loop performance obtained in the suboptimal MPC-NPL algorithm with quadratic programming is similar to that obtained in the computationally demanding MPC-NO approach, in which a nonlinear optimisation problem has to be solved on-line at each sampling instant.

Simulation results of both algorithms based on the structured neural model with the prediction horizon shortened to $N = 5$ are shown in Fig. 8. The shorter prediction horizon results in slightly bigger overshoot in comparison with the nominal case, in which $N = 10$. The shortening of the horizon can be considered when one wants to reduce the complexity of the model or the computational

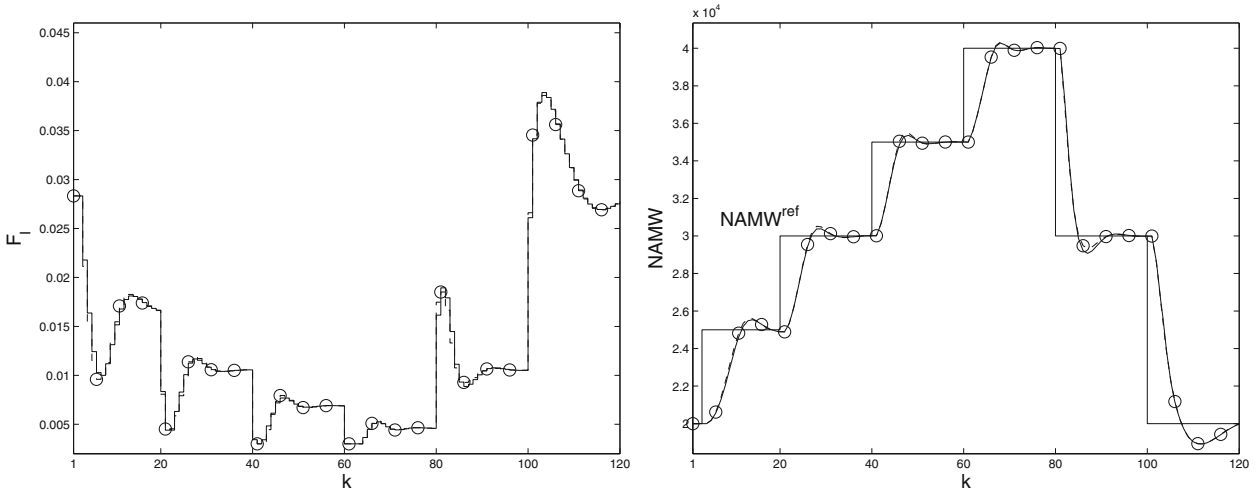

Fig. 8. Simulation results: the MPC-NPL algorithm (*dashed line with circles*) and the MPC-NO algorithm (*solid line*) based on the same structured neural model with the prediction horizon shortened to $N = 5$.

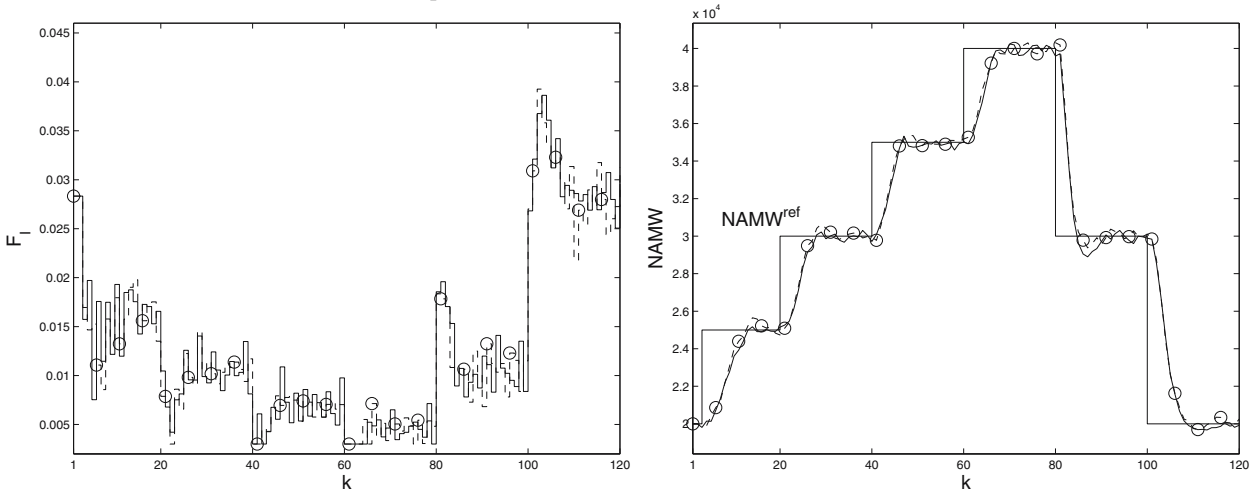

Fig. 9. Simulation results: the MPC-NPL algorithm (*dashed line with circles*) and the MPC-NO algorithm (*solid line*) based on the same structured neural model in presence of unmeasured disturbances.

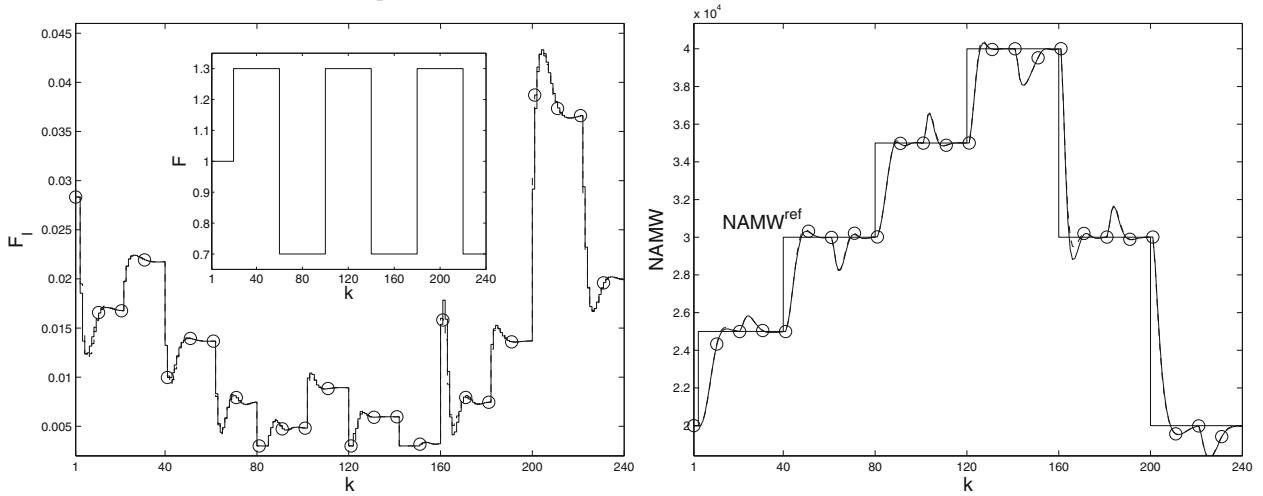

Fig. 10. Simulation results: the MPC-NPL algorithm (*dashed line with circles*) and the MPC-NO algorithm (*solid line*) based on the same structured neural model in presence of step changes in the flow rate F of the monomer.

burden. On the other hand, excessive shortening may result in instability (Maciejowski, 2002; Tatjewski, 2007).

Finally, the efficiency of both MPC algorithms based on the structured model is evaluated in the presence of disturbances, which are unavoidable in practice. Simulation results in the presence of unmeasured disturbances are shown in Fig. 9. Simulation results in the presence of step changes in the flow rate F of the monomer are shown in Fig. 10.

6. Conclusions

The presented MPC-NPL algorithm with structured neural network models has the following advantages: reliability, computational efficiency and the closed-loop accuracy. The algorithm uses on-line only the numerically reliable quadratic programming procedure, so the necessity of repeating full nonlinear optimisation at each sampling instant is avoided. Although suboptimal, in practice the algorithm gives closed-loop control performance similar to that obtained in MPC with nonlinear optimisation.

The structure of the model and its identification should be chosen with its further application in MPC algorithms in mind. MPC algorithms are very model based and likely to offer good control performance provided that predictions calculated from the model are accurate enough. The very specific role of the model in MPC cannot be ignored, but it is sometimes overlooked, as was emphasised in (Rossiter, 2003).

The prediction error propagation problem is one of the most important issues in nonlinear MPC. Although models used in MPC have to be able to make good predictions of future behaviour of the process over the whole prediction horizon, neural models trained by means of the rudimentary backpropagation algorithm are in fact one-step ahead predictors. When such models are used in MPC, the prediction error is inevitably propagated. This is so because of noise, model inaccuracies and underparameterisation. In particular, underparameterisation is potentially a very frequent source of prediction inaccuracies as was demonstrated in the example polymerisation reactor studied in this paper. Usually, the order of models used in MPC is significantly lower than the order of the real process, or even the proper model order is unknown.

The structured neural model described in the paper predicts future values of the output without taking into account previous predictions calculated within the prediction horizon. It is not used recursively and the prediction error is not propagated. Structured models have much better long-range prediction accuracy in comparison with the corresponding classical NARX models. Conceptually, a modelling idea presented in this paper can be regarded as a modification of the linear multi-model approach (Liu *et al.*, 1999; Rossiter and Kouvaritakis, 2001) designed to effectively deal with nonlinear processes. Instead of having a set of separate models, i.e., one model for each sampling instant within the prediction horizon, only one structured neural model is used. The structured model is easily trained by means of the classical backpropagation algorithm, and it is not necessary to use recurrent neural network training, which is much more complicated. In comparison with the corresponding classical NARX neural models, structured models have more input nodes so it is very important to prune these models. In this study, the OBD pruning algorithm is used, which significantly reduces the number of weights and improves generalisation abilities.

Acknowledgement

The work presented in this paper was supported by the Polish national budget funds for science for the years 2007–2009 in the framework of a research project.

References

Åkesson, B. M. and Toivonen, H. T. (2006). A neural network model predictive controller, *Journal of Process Control* **16**(3): 937–946.

Bazaraa, M. S., Sherali, J. and Shetty, K. (1993). *Nonlinear Programming: Theory and Algorithms*, John Wiley & Sons, New York, NY.

Clarke, D. W. and Mohtadi, C. (1989). Properties of generalized predictive control, *Automatica* **25**(6): 859–875.

Cutler, R. and Ramaker, B. (1979). Dynamic matrix control—A computer control algorithm, *Proceedings of the AIChE National Meeting, Houston, TX, USA*.

Doyle, F. J., Ogunnaike, B. A. and Pearson, R. K. (1995). Nonlinear model-based control using second-order Volterra models, *Automatica* **31**(5): 697–714.

Doyle, F. J., R. K. P. and Ogunnaike, B. A. (2001). *Identification and Control of Process Systems Using Volterra Models*, Springer, New York, NY.

Haykin, S. (1999). *Neural Networks. A Comprehensive Foundation, 2nd Edition*, Prentice Hall, Englewood Cliffs, NJ.

Henson, M. A. (1998). Nonlinear model predictive control: Current status and future directions, *Computers and Chemical Engineering* **23**(2): 187–202.

Hornik, K., Stinchcombe, M. and White, H. (1989). Multilayer feedforward networks are universal approximators, *Neural Networks* **2**(5): 359–366.

Hussain, M. A. (1999). Review of the applications of neural networks in chemical process control—Simulation and online implementation, *Artificial Intelligence in Engineering* **13**(1): 55–68.

Ławryńczuk, M. (2007a). A family of model predictive control algorithms with artificial neural networks, *International Journal of Applied Mathematics and Computer Science* **17**(2): 217–232.

Ławryńczuk, M. (2007b). Suboptimal nonlinear predictive control with structured neural models, *in* J. M. de Sá, J. M. Alexandre, W. Duch and D. Mandic (Eds.), *The 17th International Conference on Artificial Neural Networks, ICANN 2007, Porto, Portugal*, Springer, Heidelberg, pp. 630–639.

Ławryńczuk, M. and Tadej, W. (2008). A computationally efficient stable dual-mode type nonlinear predictive control algorithm, *Control and Cybernetics* **37**(1): 99–132.

LeCun, Y., Denker, J. and Solla, S. (1989). Optimal brain damage, *in* D. Touretzky (Ed.), *Advances of NIPS2*, Morgan Kaufmann, San Mateo, CA, pp. 598–605.

Liu, D., Shah, S. L. and Fisher, D. G. (1999). Multiple prediction models for long range predictive control, *Proceedings of the IFAC World Congress, Beijing, China*, (on CD-ROM).

Liu, G. P., Kadirkamanathan, V. and Billings, S. A. (1998). Predictive control for non-linear systems using neural networks, *International Journal of Control* **71**(6): 1119–1132.

Luyben, W. L. (1990). *Process Modelling, Simulation and Control for Chemical Engineers*, McGraw Hill, New York, NY.

Maciejowski, J. M. (2002). *Predictive Control with Constraints*, Prentice Hall, Harlow.

Marlin, T. E. (1995). *Process Control*, McGraw-Hill, New York, NY.

Morari, M. and Lee, J. (1999). Model predictive control: Past, present and future, *Computers and Chemical Engineering* **23**(4): 667–682.

Nørgaard, M., Ravn, O., Poulsen, N. K. and Hansen, L. K. (2000). *Neural Networks for Modelling and Control of Dynamic Systems*, Springer, London.

Parisini, T., Sanguineti, M. and Zoppoli, R. (1998). Nonlinear stabilization by receding-horizon neural regulators, *International Journal of Control* **70**(3): 341–362.

Piche, S., Sayyar-Rodsari, B., Johnson, D. and Gerules, M. (2000). Nonlinear model predictive control using neural networks, *IEEE Control Systems Magazine* **20**(3): 56–62.

Pottmann, M. and Seborg, D. E. (1997). A nonlinear predictive control strategy based on radial basis function models, *Computers and Chemical Engineering* **21**(9): 965–980.

Qin, S. J. and Badgwell, T. (2003). A survey of industrial model predictive control technology, *Control Engineering Practice* **11**(7): 733–764.

Rossiter, J. A. (2003). *Model-Based Predictive Control*, CRC Press, Boca Raton, FL.

Rossiter, J. A. and Kouvaritakis, B. (2001). Modelling and implicit modelling for predictive control, *International Journal of Control* **74**(11): 1085–1095.

Scattolini, R. and Bittanti, S. (1990). On the choice of the horizon in long-range predictive control—Some simple criteria, *Automatica* **26**(5): 915–917.

Tatjewski, P. (2007). *Advanced Control of Industrial Processes, Structures and Algorithms*, Springer, London.

Tatjewski, P. and Ławryńczuk, M. (2006). Soft computing in model-based predictive control, *International Journal of Applied Mathematics and Computer Science* **16**(1): 101–120.

Trajanoski, Z. and Wach, P. (1998). Neural predictive control for insulin delivery using the subcutaneous route, *IEEE Transactions on Biomedical Engineering* **45**(9): 1122–1134.

Yu, D. L. and Gomm, J. B. (2003). Implementation of neural network predictive control to a multivariable chemical reactor, *Control Engineering Practice* **11**(11): 1315–1323.

Maciej Ławryńczuk obtained his M.Sc. and Ph.D. degrees in automatic control from the Warsaw University of Technology, Poland, in 1998 and 2003, respectively. Currently, he is an assistant professor at the Institute of Control and Computation Engineering of the Warsaw University of Technology. His research interests include the application of neural networks in process modelling, control (in particular model predictive control), and optimisation.

Received: 3 April 2008
Revised: 14 September 2008