



# Computational Efficiency of Suboptimal Nonlinear Predictive Control with Neural Models

Maciej Lawryńczuk

Institute of Control and Computation Engineering  
Warsaw University of Technology  
ul. Nowowiejska 15/19, 00-665 Warszawa, Poland  
tel. +48 22 234-73-97  
M.Lawrynczuk@ia.pw.edu.pl

**Abstract.** This paper studies computational efficiency of suboptimal Model Predictive Control (MPC) with neural models. The algorithm requires solving on-line only a quadratic optimisation problem. Considering a nonlinear polymerisation process, for which the linear MPC algorithm is inadequate, it is shown that the suboptimal algorithm results in closed-loop control performance similar to that obtained in the fully-fledged nonlinear MPC technique, which hinges on non-convex optimisation.

## 1 Introduction

Model Predictive Control (MPC) is recognised as the only advanced control technique which has been very successful in practical applications [2, 8, 13–15]. It is largely because MPC algorithms can take into account constraints imposed on both process inputs (manipulated variables) and outputs (controlled variables), which usually decide on quality, economic efficiency and safety. Moreover, MPC techniques are very efficient in multivariable process control.

Because properties of many technological processes are nonlinear, different nonlinear MPC techniques become more and more popular [2, 13, 14]. In view of practical importance, both accuracy and computational efficiency determine reliability of any nonlinear MPC algorithm. These factors are influenced by the structure of the model and the way it is used in the control algorithm. Fundamental models, although potentially very precise, are usually not suitable for on-line control because they are very complicated and may lead to numerical problems. Since neural network models [1] are able to approximate precisely nonlinear behaviour of technological processes [3, 10], have relatively small number of parameters and simple structure, they can be effectively used in MPC algorithms as process models [4–7, 10–12, 14, 15].

This paper investigates computational efficiency of a suboptimal MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL) [5–7, 14, 15] with neural models of processes. Unlike the MPC technique with Nonlinear Optimisation (MPC-NO), which hinges on on-line non-convex optimisation, the algorithm

needs solving on-line only a quadratic optimisation problem. At the same time, in practice the suboptimal algorithm gives control performance similar to that obtained in the MPC-NO one.

## 2 Model Predictive Control Algorithms

In the MPC algorithms [8, 14] at each consecutive sampling instant  $k$  a set of future controls

$$\mathbf{u}(k) = [u(k|k) \ u(k+1|k) \ \dots \ u(k+N_u-1|k)]^T \quad (1)$$

or corresponding increments

$$\Delta \mathbf{u}(k) = [\Delta u(k|k) \ \Delta u(k+1|k) \ \dots \ \Delta u(k+N_u-1|k)]^T \quad (2)$$

is calculated. It is assumed that  $\Delta u(k+p|k) = 0$  ( $u(k+p|k) = u(k+N_u-1|k)$ ) for  $p \geq N_u$ , where  $N_u$  is the control horizon. The objective is to minimise the differences between the reference trajectory  $y^{ref}(k+p|k)$  and the predicted values of the output  $\hat{y}(k+p|k)$  over the prediction horizon  $N$ . The following quadratic cost function is usually used

$$J(k) = \sum_{p=1}^N (y^{ref}(k+p|k) - \hat{y}(k+p|k))^2 + \sum_{p=0}^{N_u-1} \lambda_p (\Delta u(k+p|k))^2 \quad (3)$$

where  $\lambda_p > 0$  are weighting factors. Typically,  $N_u < N$ , which decreases the dimensionality of the optimisation problem and leads to smaller computational load. Only the first element of the determined sequence (1) or (2) is applied to the process, the control law is then

$$u(k) = u(k|k) \quad \text{or} \quad u(k) = \Delta u(k|k) + u(k-1) \quad (4)$$

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

Since the constraints have to be usually taken into account, future control increments are determined as the solution to the following optimisation problem (assuming hard output constraints [8], [14] for simplicity of presentation)

$$\begin{aligned} & \min_{u(k|k) \dots u(k+N_u-1|k)} \{J(k)\} \\ & \text{subject to} \\ & u^{\min} \leq u(k+p|k) \leq u^{\max}, \quad p = 0, \dots, N_u-1 \\ & -\Delta u^{\max} \leq \Delta u(k+p|k) \leq \Delta u^{\max}, \quad p = 0, \dots, N_u-1 \\ & y^{\min} \leq \hat{y}(k+p|k) \leq y^{\max}, \quad p = 1, \dots, N \end{aligned} \quad (5)$$

Predicted values of the output,  $\hat{y}(k+p|k)$ , over the prediction horizon are calculated using a dynamic model of the process. All MPC algorithms discussed in this paper use feedforward neural network models with one hidden layer and a linear output [1]. As an alternative, RBF type neural networks can be used in MPC algorithms [6].

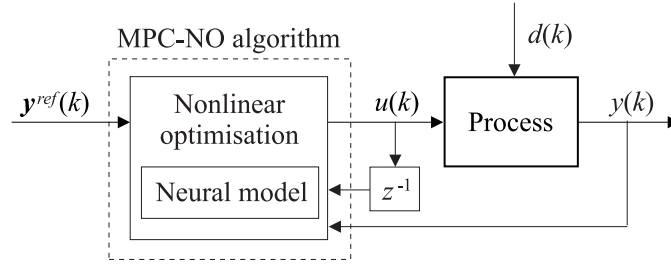


Fig. 1. Structure of the MPC algorithm with Nonlinear Optimisation (MPC-NO)

### 3 MPC Algorithm with Nonlinear Optimisation (MPC-NO) and Neural Models

In the MPC-NO algorithm for prediction purposes the nonlinear neural model is used without any simplifications [5, 14, 15]. At each sampling instant future values of control signals,  $\mathbf{u}(k)$ , are determined as the solution to a nonlinear optimisation problem. Structure of the MPC-NO algorithm is depicted in Fig. 1. From (3) and (5) the MPC-NO optimisation problem is

$$\begin{aligned} \min_{\mathbf{u}(k)} \left\{ J(k) = \|\mathbf{y}^{ref}(k) - \hat{\mathbf{y}}(k)\|^2 + \|\Delta\mathbf{u}(k)\|_{\mathbf{A}}^2 \right\} \\ \text{subject to} \\ \mathbf{u}^{\min} \leq \mathbf{u}(k) \leq \mathbf{u}^{\max} \\ -\Delta\mathbf{u}^{\max} \leq \Delta\mathbf{u}(k) \leq \Delta\mathbf{u}^{\max} \\ \mathbf{y}^{\min} \leq \hat{\mathbf{y}}(k) \leq \mathbf{y}^{\max} \end{aligned} \quad (6)$$

where

$$\mathbf{y}^{ref}(k) = [y^{ref}(k+1|k) \dots y^{ref}(k+N|k)]^T \quad (7)$$

$$\hat{\mathbf{y}}(k) = [\hat{y}(k+1|k) \dots \hat{y}(k+N|k)]^T \quad (8)$$

$$\mathbf{y}^{\min}(k) = [y^{\min} \dots y^{\min}]^T \quad (9)$$

$$\mathbf{y}^{\max}(k) = [y^{\max} \dots y^{\max}]^T \quad (10)$$

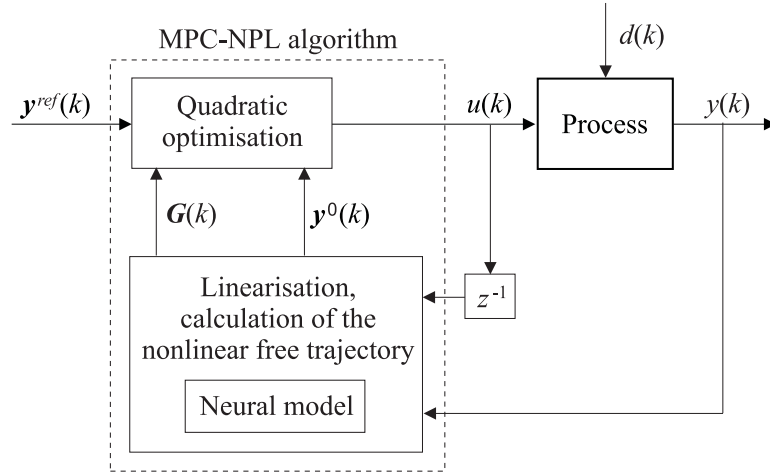
are vectors of length  $N$ ,

$$\mathbf{u}^{\min}(k) = [u^{\min} \dots u^{\min}]^T \quad (11)$$

$$\mathbf{u}^{\max}(k) = [u^{\max} \dots u^{\max}]^T \quad (12)$$

$$\Delta\mathbf{u}^{\max}(k) = [\Delta u^{\max} \dots \Delta u^{\max}]^T \quad (13)$$

are vectors of length  $N_u$  and  $\mathbf{A} = \text{diag}(\lambda_0, \dots, \lambda_{N_u-1})$ . In the simplest and the most common case the reference trajectory is not known in advance, hence  $\mathbf{y}^{ref}(k) = [y^{ref}(k) \dots y^{ref}(k)]^T$  where  $y^{ref}(k)$  is the current set-point.



**Fig. 2.** Structure of the MPC algorithm with Nonlinear Prediction and Linearisation (MPC-NPL)

Sequential Quadratic Programming (SQP) is used for solving the nonlinear MPC-NO optimisation problem (6). Structure of the neural model is exploited, gradients of the cost function  $J(k)$  with respect to future controls,  $\mathbf{u}(k)$ , are calculated analytically. It is necessary to emphasise the fact that the difficulty of the MPC-NO optimisation problem (6) is twofold. First of all, it is nonlinear, computationally demanding, the computational burden is big. Secondly, it may be non-convex and even multi-modal. Unfortunately, for such problems there are no sufficiently fast and reliable optimisation algorithms, i.e. those which would be able to determine the global optimal solution at each sampling instant and within predefined time limit as it is required in on-line control.

#### 4 MPC Algorithm with Nonlinear Prediction and Linearisation (MPC-NPL) and Neural Models

In the MPC-NPL algorithm at each sampling instant  $k$  the neural model is used on-line twice: to determine a local linearisation and a nonlinear free trajectory [5], [6], [7], [14], [15]. 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  $\Delta\mathbf{u}(k)$ ) and the free trajectory  $\mathbf{y}^0(k)$ , which depends only on the past

$$\hat{\mathbf{y}}(k) = \mathbf{y}^0(k) + \mathbf{G}(k)\Delta\mathbf{u}(k) \quad (14)$$

where  $\mathbf{G}(k)$  is a dynamic matrix of the linearised model and

$$\mathbf{y}^0(k) = [y^0(k+1|k) \dots y^0(k+N|k)]^T \quad (15)$$

Both the free trajectory and the dynamic matrix are calculated on-line from the nonlinear model taking into account the current state of the plant.

On the one hand, the suboptimal prediction calculated from (14) is different from the optimal one determined from the nonlinear neural model as it is done in the MPC-NO algorithm. On the other hand, thanks to using the superposition principle, the optimisation problem (6) becomes the following quadratic programming task

$$\begin{aligned} & \min_{\Delta \mathbf{u}(k)} \left\{ J(k) = \|\mathbf{y}^{ref}(k) - \mathbf{y}^0(k) - \mathbf{G}(k)\Delta \mathbf{u}(k)\|^2 + \|\Delta \mathbf{u}(k)\|_{\Lambda}^2 \right\} \\ & \text{subject to} \\ & \quad \mathbf{u}^{\min} \leq \mathbf{u}(k) \leq \mathbf{u}^{\max} \\ & \quad -\Delta \mathbf{u}^{\max} \leq \Delta \mathbf{u}(k) \leq \Delta \mathbf{u}^{\max} \\ & \quad \mathbf{y}^{\min} \leq \hat{\mathbf{y}}(k) \leq \mathbf{y}^{\max} \end{aligned} \quad (16)$$

In the MPC-NPL algorithm it is convenient to calculate future control increments  $\Delta \mathbf{u}(k)$  rather than absolute values  $\mathbf{u}(k)$  of future control signal as it is done in the MPC-NO algorithm.

Structure of the MPC-NPL algorithm is depicted in Fig. 2. At each sampling instant  $k$  the following steps are repeated:

1. Linearisation of the neural model: obtain the matrix  $\mathbf{G}(k)$ .
2. Calculate the nonlinear free trajectory  $\mathbf{y}^0(k)$  using the neural model.
3. Solve the quadratic programming problem (16) to determine  $\Delta \mathbf{u}(k)$ .
4. Apply  $u(k) = \Delta u(k|k) + u(k-1)$ .
5. Set  $k := k + 1$ , go to step 1.

## 5 Simulation Results

### 5.1 Process, Models and Compared MPC Strategies

The process under consideration is a polymerisation reaction taking place in a jacketed continuous stirred tank reactor [9]. The reaction is the free-radical polymerisation of methyl methacrylate with azo-bis-isobutyronitrile as initiator and toluene as solvent. The output *NAMW* (Number Average Molecular Weight) is controlled by manipulating the inlet initiator flow rate  $F_I$ .

Three models of the process are used. The fundamental model [9] is used as the real process during simulations. An identification procedure is carried out, a linear model and a neural one are obtained. Empirical models used in MPC algorithms are:

- a) a linear model

$$y(k) = b_2 u(k-2) - a_1 y(k-1) - a_2 y(k-2) \quad (17)$$

- b) a neural model containing 6 neurons in the hidden layer

$$y(k) = g(u(k-2), y(k-1), y(k-2)) \quad (18)$$

where  $u = F_I$ ,  $y = NAMW$ . Compared MPC strategies are:

- a) the linear MPC algorithm with the linear model (17),
- b) the nonlinear MPC-NO algorithm with the neural model (18),
- c) the suboptimal nonlinear MPC-NPL algorithm with the neural model (18),

Because one may expect that the initial point for the MPC-NO optimisation problem,  $\mathbf{u}^0(k)$ , is likely to affect the computational burden, four subversions of the MPC-NO algorithm are used:

- a) MPC-NOa:  $\mathbf{u}^0(k)$  uses  $N_u - 1$  control values calculated at the previous sampling instant and not applied to the process

$$\mathbf{u}^0(k) = \begin{bmatrix} u^0(k|k) \\ \vdots \\ u^0(k + N_u - 3|k) \\ u^0(k + N_u - 2|k) \\ u^0(k + N_u - 1|k) \end{bmatrix} = \begin{bmatrix} u(k|k-1) \\ \vdots \\ u(k + N_u - 3|k-1) \\ u(k + N_u - 2|k-1) \\ u(k + N_u - 2|k-1) \end{bmatrix} \quad (19)$$

- b) MPC-NOb:  $\mathbf{u}^0(k)$  uses the value of the manipulated variable calculated and applied to the plant at the previous sampling instant, i.e  $\mathbf{u}^0(k) = [u(k-1) \dots u(k-1)]^T$ ,
- c) MPC-NOc:  $\mathbf{u}^0(k)$  is found from the steady-state characteristic of the process for the current set-point  $NAMW^{ref}(k)$ ,
- d) MPC-NOd:  $\mathbf{u}^0(k) = \text{const.}$

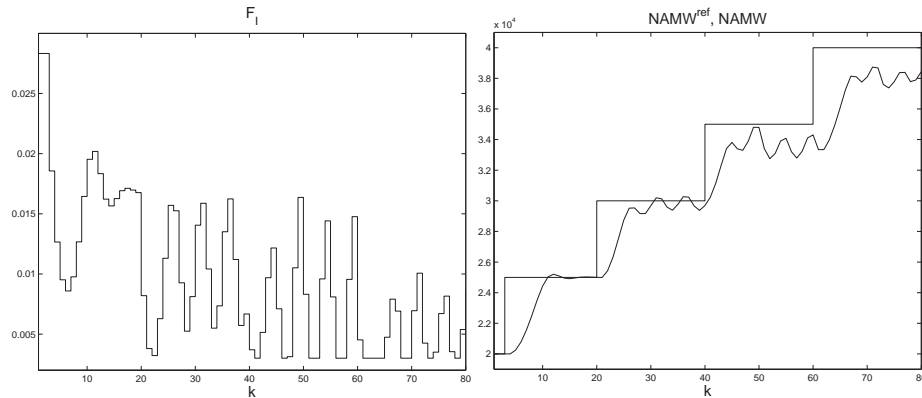
If  $N_u = 1$ , in the MPC-NOa algorithm it is assumed that  $\mathbf{u}^0(k) = u(k-1)$ , as it is done in the MPC-NOb one. The manipulated variable is constrained,  $F_I^{\min} = 0.003$ ,  $F_I^{\max} = 0.06$ ,  $\lambda_p = 0.2$ , the sampling time is 1.8 min.

## 5.2 Control Accuracy

Simulation results of the MPC algorithm with the linear model are depicted in Fig. 3. As the reference trajectory, four set-point changes are considered, the horizons are:  $N = 10$ ,  $N_u = 3$ . The linear algorithm works well only for the smallest set-point change, whereas for bigger ones the system becomes unstable. Simulation results of the MPC-NPL and the MPC-NO (MPC-NOa) algorithms with the same neural network model are depicted in Fig. 4. Both nonlinear algorithms are stable. Moreover, the closed-loop performance obtained in the suboptimal MPC-NPL algorithm with quadratic programming is very close 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.

## 5.3 Computational Complexity

Table 1 shows the influence of control and prediction horizons on floating point operation (MFLOPS). Six control horizon ( $N_u = 1, 2, 3, 4, 5, 10$ ) and two prediction horizon ( $N = 5, 10$ ) lengths are considered. Fig. 5 depicts this influence for



**Fig. 3.** Simulation results of the MPC algorithm with the linear model

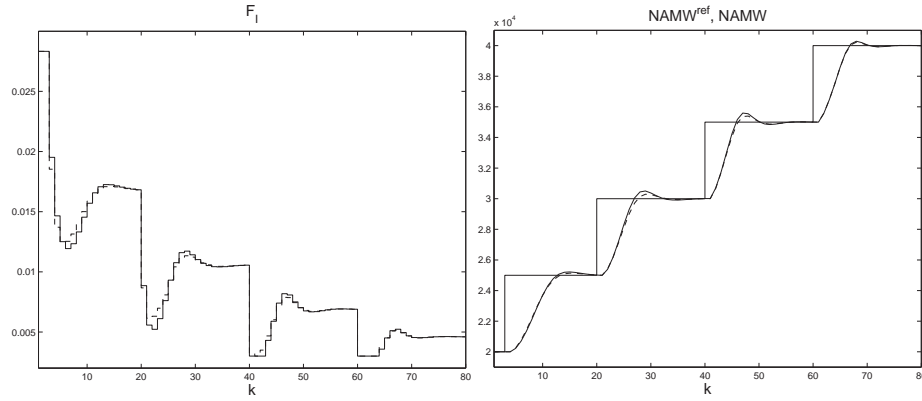
all possible combinations of horizons ( $N_u = 1, \dots, 10$ ,  $N = 2, \dots, 10$ ). In general, the control horizon has significantly bigger impact on computational burden than the prediction horizon since  $N_u$  is the number of the decision variables of the optimisation problem (6) or (16).

It is best when the MPC-NO algorithm uses as the initial point for the optimisation problem  $N_u - 1$  control values calculated at the previous sampling instant and not applied to the process (the MPC-NOa algorithm), which is clearly visible when the control horizon is long. Quite good results (for short control horizons comparable to those obtained in the MPC-NOa algorithm) are obtained when as the initial point the value of the manipulated variable calculated and applied to the plant at the previous sampling instant is used (the MPC-NOb algorithm). When the initial point is found from the steady-state characteristic of the process for the current set-point (the MPC-NOc algorithm) or a constant value is used (the MPC-NOd algorithm), the computational burden is very big.

Table 2 shows floating point operation reduction obtained in the MPC-NPL algorithm compared to the MPC-NO algorithm with different initial points for all possible combinations of horizons. In the worst case the MPC-NPL algorithm is more than 3 times less computationally demanding than the MPC-NO one, in the best case more than 25 times.

## 6 Conclusions

Accuracy and computational efficiency are the advantages of the suboptimal MPC-NPL algorithm with neural models. In practice [5–7, 14, 15], the algorithm gives closed-loop control performance comparable to that obtained in the MPC-NO schemes with nonlinear optimisation. Computational efficiency is twofold. First of all, the MPC-NPL algorithm uses on-line only the numerically reliable quadratic programming procedure, unlike the nonlinear optimisation, which may



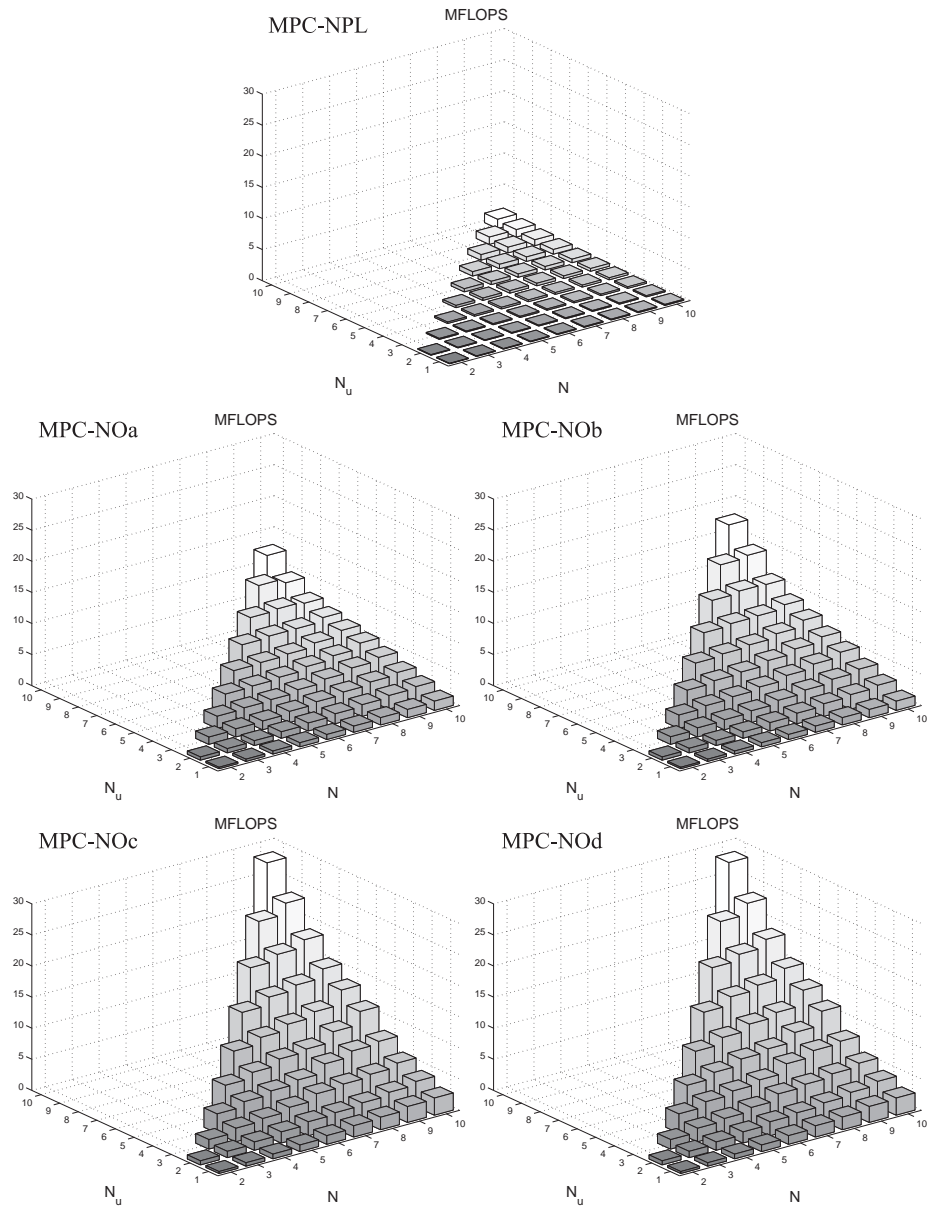
**Fig. 4.** Simulation results of the MPC-NPL (dotted) and MPC-NO (solid) algorithms with the same neural network model

**Table 1.** Influence of control and prediction horizons on floating point operation (MFLOPS) in the MPC-NPL algorithm and the MPC-NO algorithm with different initial points

Algorithm	$N$	$N_u = 1$	$N_u = 2$	$N_u = 3$	$N_u = 4$	$N_u = 5$	$N_u = 10$
MPC-NPL	5	0.0998	0.1221	0.1739	0.2469	0.3515	–
MPC-NOa	5	0.6522	1.0488	1.4110	2.1493	2.9892	–
MPC-NOb	5	0.6522	1.0206	1.6195	2.5040	3.6211	–
MPC-NOc	5	1.0558	1.7913	2.5847	3.9007	5.1672	–
MPC-NOd	5	1.0558	1.7913	2.5843	3.9045	5.1707	–
MPC-NPL	10	0.1607	0.1861	0.2427	0.3209	0.4315	1.4956
MPC-NOa	10	1.4705	2.4544	3.3510	4.4030	5.3262	12.5403
MPC-NOb	10	1.4705	2.6686	3.8834	5.2310	6.3347	17.4621
MPC-NOc	10	2.7457	4.0431	5.6171	7.7429	11.1757	28.2162
MPC-NOd	10	2.7504	4.0548	5.6054	7.7607	11.2266	28.2670

**Table 2.** Floating point operation reduction obtained in the MPC-NPL algorithm compared to the MPC-NO algorithm with different initial points for all possible combinations of horizons ( $N_u = 1, \dots, 10$ ,  $N = 2, \dots, 10$ )

Algorithm	minimal	maximal
MPC-NPL vs. MPC-NOa	3.4317	13.6146
MPC-NPL vs. MPC-NOb	3.4340	16.1267
MPC-NPL vs. MPC-NOc	4.5388	25.6879
MPC-NPL vs. MPC-NOd	4.5276	25.8049



**Fig. 5.** Influence of prediction and control horizons on floating point operation (MFLOPS) in the MPC-NPL algorithm and the MPC-NO algorithm with different initial points

terminate in a local minimum, the global solution is always found. Moreover, for the studied polymerisation process, floating point operation reduction obtained in the suboptimal algorithm compared to the MPC-NO algorithm in the best case is bigger than 25 (Table 2).

## Acknowledgement

This work was supported by Polish national budget funds 2005-2007 for science as a research project.

## References

1. Haykin S.: *Neural networks—a comprehensive foundation*. Prentice Hall, Englewood Cliffs. (1999).
2. Henson M. A.: *Nonlinear model predictive control: current status and future directions*, Computers and Chemical Engineering. **23** (1998) 187–202.
3. Hussain M. A.: *Review of the applications of neural networks in chemical process control—simulation and online implementation*, Artificial Intelligence in Engineering. **13** (1999) 55–68.
4. Liu G. P., Kadiramanathan V., Billings S. A.: *Predictive control for non-linear systems using neural networks*, International Journal of Control. **71** (1998) 1119–1132.
5. Ławryńczuk M.: *A family of model predictive control algorithms with artificial neural networks*, International Journal of Applied Mathematics and Computer Science. **17** (2007) 217–232.
6. Ławryńczuk, M. and Tatjewski, P.: *A computationally efficient nonlinear predictive control algorithm with RBF neural models and its application*, Lecture Notes in Artificial Intelligence, Springer. **4585** (2007) 603–612.
7. Ławryńczuk, M., Tatjewski, P.: *An efficient nonlinear predictive control algorithm with neural models and its application to a high-purity distillation process*, Lecture Notes in Artificial Intelligence, Springer. **4029** (2006) 76–85.
8. Maciejowski J. M.: *Predictive control with constraints*, Prentice Hall. Harlow. (2002).
9. Maner B. R., Doyle F. J., Ogunnaike B. A., Pearson R. K.: *Nonlinear model predictive control of a simulated multivariable polymerization reactor using second-order Volterra models*, Automatica. **32** (1996) 1285–1301.
10. Nørgaard M., Ravn O., Poulsen N. K., Hansen L. K.: *Neural networks for modelling and control of dynamic systems*, Springer. London. (2000).
11. Parisini T., Sanguineti M., Zoppoli R.: *Nonlinear stabilization by receding-horizon neural regulators*, International Journal of Control. **70** (1998) 341–362.
12. Piche S., Sayyar-Rodsari B., Johnson D., Gerules M.: *Nonlinear model predictive control using neural networks*, IEEE Control Systems Magazine. **20** (2000) 56–62.
13. Qin S. J., Badgwell T. A.: *A survey of industrial model predictive control technology*, Control Engineering Practice. **11** (2003) 733–764.
14. Tatjewski P.: *Advanced control of industrial processes, Structures and algorithms*, Springer. London. (2007).
15. Tatjewski P., Ławryńczuk M.: *Soft computing in model-based predictive control*, International Journal of Applied Mathematics and Computer Science. **16** (2006) 101–120.