DE GRUYTER OPEN

Int. J. Appl. Math. Comput. Sci., *2015, Vol. 25, No. 4, 833–847*
DOI: 10.1515/amcs-2015-0060

amcs

NONLINEAR STATE–SPACE PREDICTIVE CONTROL WITH ON–LINE LINEARISATION AND STATE ESTIMATION

MACIEJ ŁAWRYŃCZUK [a]

[a]Institute of Control and Computation Engineering
Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: M.Lawrynczuk@ia.pw.edu.pl

This paper describes computationally efficient model predictive control (MPC) algorithms for nonlinear dynamic systems represented by discrete-time state-space models. Two approaches are detailed: in the first one the model is successively linearised on-line and used for prediction, while in the second one a linear approximation of the future process trajectory is directly found on-line. In both the cases, as a result of linearisation, the future control policy is calculated by means of quadratic optimisation. For state estimation, the extended Kalman filter is used. The discussed MPC algorithms, although disturbance state observers are not used, are able to compensate for deterministic constant-type external and internal disturbances. In order to illustrate implementation steps and compare the efficiency of the algorithms, a polymerisation reactor benchmark system is considered. In particular, the described MPC algorithms with on-line linearisation are compared with a truly nonlinear MPC approach with nonlinear optimisation repeated at each sampling instant.

Keywords: process control, model predictive control, nonlinear state-space models, extended Kalman filter, on-line linearisation.

1. Introduction

In the classical control techniques (e.g., PID, LQR) a model of the process is only used off-line for controller synthesis, while in model predictive control (MPC) it is used on-line for prediction of the future process trajectory. The control policy is calculated in MPC at each sampling instant from an optimisation problem which aims to minimise the predicted control errors (Camacho and Bordons, 1999; Maciejowski, 2002; Rawlings and Mayne, 2009; Tatjewski, 2007). The MPC approach has many advantages, among which it is necessary to point out its unique ability to take into account constraints imposed on process variables as well as the ability to efficiently control multiple-input multiple-output processes and systems with difficult dynamic properties (e.g., delayed ones). As a result, the MPC algorithms have been successfully used in numerous advanced applications (Qin and Badgwell, 2003). The most important current research fields are: optimality, stability and robustness (Mayne, 2014), on-line set-point optimisation for MPC (Tatjewski, 2010), economic MPC (Ellis *et al.*, 2014) and fault-tolerant MPC (Patan and Korbicz, 2012).

In the majority of applications, input-output models are used, typically linear ones. Applications of state-space models are less frequent. On the other hand, state-space models are more general and able to describe a very wide class of dynamic systems. The treatment of deterministic disturbances which affect the process is very important to guarantee offset-free control. The standard method is to augment the process state by the states of deterministic disturbances (Maciejowski, 2002; Gonzalez *et al.*, 2008; Maeder and Morari, 2010; Muske and Badgwell, 2002; Pannocchia and Bemporad, 2007; Pannocchia and Rawlings, 2003; Rawlings and Mayne, 2009). As a result, one obtains extended state and output equations. It is also possible to use the velocity form state-space model, in which the extended state consists of state increments and the output signals (Maciejowski, 2002; Wang, 2007).

An interesting alternative was suggested recently by Tatjewski (2014), who described the MPC algorithm for dynamic systems based on linear time-invariant models with a new, straightforward MPC controller-observer structure. Such an MPC algorithm is able to compensate for the deterministic constant-type disturbances affecting the process, possibly with white noise added under (asymptotically) constant set-point values. The constant-type disturbances considered are crucial in process control because they include unavoidable modelling errors or piecewise-constant disturbances. Relatively simple disturbance models are used. In comparison with the cited augmented state formulations, the methodology presented by Tatjewski (2014) only requires estimation of the process state.

The objective of this paper is to use the disturbance handling mechanism employed in the case of linear systems and described by Tatjewski (2014) in nonlinear MPC based on a general class of nonlinear state-space models (it is only assumed that the model is differentiable). Because direct application of a nonlinear model leads to a nonlinear MPC optimisation problem, two successive on-line linearisation approaches are discussed. In the first one the model is linearised on-line and used for prediction, in the second one a linear approximation of the future process trajectory is directly found on-line. In both the cases the future control policy is calculated by means of easy-to-solve quadratic optimisation problems. For state estimation, the extended Kalman filter is used. This paper also extends the work of Ławryńczuk (2014), where a few MPC algorithms with on-line linearisation are discussed for dynamic systems represented by neural models. In order to illustrate implementation steps and compare the efficiencies of the discussed MPC algorithms, a polymerisation reactor benchmark system is considered. In particular, the MPC algorithms with on-line linearisation are compared with a truly nonlinear MPC approach with nonlinear optimisation repeated at each sampling instant. The proof that the algorithms guarantee offset-free control is given.

This paper is structured as follows. First, in Section 2, the MPC problem is defined. Next, in Section 3, two nonlinear MPC algorithms are discussed, i.e., MPC with on-line model linearisation and MPC with on-line trajectory linearisation. Section 4 details implementation steps and compares the efficiency of the discussed MPC algorithms for a polymerisation reactor benchmark system. Finally, Section 5 concludes the paper.

2. Nonlinear state-space predictive control problem formulation

2.1. State-space process description. The nonlinear state-space model of a dynamic process is

$$x(k+1) = f(x(k), u(k)), \tag{1a}$$

$$y(k) = g(x(k)), \tag{1b}$$

where $x(k) \in \mathbb{R}^{n_\mathrm{x}}$ denotes the state vector, $u \in \mathbb{R}^{n_\mathrm{u}}$ is the vector of input (manipulated) variables and $y \in \mathbb{R}^{n_\mathrm{y}}$ is the vector of output (controlled) variables. The nonlinear functions $f\colon \mathbb{R}^{n_\mathrm{x}+n_\mathrm{u}} \to \mathbb{R}^{n_\mathrm{x}}$ and $g\colon \mathbb{R}^{n_\mathrm{x}} \to \mathbb{R}^{n_\mathrm{y}}$ are assumed to be continuously differentiable. The state equation may be rewritten for the sampling instant k,

$$x(k) = f(x(k-1), u(k-1)), \tag{2a}$$

$$y(k) = g(x(k)). \tag{2b}$$

2.2. Nonlinear state-space predictive control. The core idea of MPC is to calculate repeatedly on-line not only the values of the manipulated variables for the current sampling instant k, but also some future control sequence. Usually, the vector of increments

$$\triangle \boldsymbol{u}(k) = \begin{bmatrix} \triangle u(k|k) \\ \vdots \\ \triangle u(k+N_\mathrm{u}-1|k) \end{bmatrix} \tag{3}$$

is calculated on-line. The symbol $\triangle u(k+p|k)$ denotes the increment in the manipulated variables for the sampling instant $k+p$ calculated at the current instant k. N_u is the control horizon. It is assumed that $\triangle u(k+p|k) = 0$ for $p \geq N_\mathrm{u}$. The increments are defined by

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

where the control signals for the sampling instant $k+p$ found at the current instant k are denoted by $u(k+p|k)$. The vector of decision variables (3) is successively found on-line as a solution to an optimisation problem in which differences between the set-point trajectory $y^{\mathrm{sp}}(k+p|k)$ and the predicted output values $\hat{y}(k+p|k)$ over the prediction horizon $N \geq N_\mathrm{u}$ are minimised. Typically, the following quadratic cost-function is used:

$$J(k) = \sum_{p=1}^{N} \| y^{\mathrm{sp}}(k+p|k) - \hat{y}(k+p|k) \|^2_{\boldsymbol{M}_p} + \sum_{p=0}^{N_\mathrm{u}-1} \| \triangle u(k+p|k) \|^2_{\boldsymbol{\Lambda}_p}, \tag{4}$$

where $\|x\|^2_A = x^{\mathrm{T}} A x$, the second part of the cost-function minimises excessive control increments, the and weighting matrices $\boldsymbol{M}_p \succeq 0$ and $\boldsymbol{\Lambda}_p \succ 0$ (usually diagonal) are of dimensions $n_\mathrm{y} \times n_\mathrm{y}$ and $n_\mathrm{u} \times n_\mathrm{u}$, respectively. Although the whole sequence of future control increments (3) is calculated at each sampling instant of the MPC algorithm, only its first n_u elements are applied to the process at the current sampling instant k, i.e., $u(k) = \triangle u(k|k) + u(k-1)$, and the whole procedure is repeated in the next sampling instant $(k+1)$.

The consecutive predictions of the output variables over the whole prediction horizon, i.e., the vectors $\hat{y}(k+1|k), \ldots, \hat{y}(k+N|k)$, are calculated from a dynamic model of the process. In this work the nonlinear state-space model (1a)–(1b) is used.

As mentioned in Introduction, the MPC approach makes it possible to efficiently take account of some constraints. A typical MPC optimisation problem solved on-line at each sampling instant is

$$\min_{\triangle \boldsymbol{u}(k)} \{J(k)\},$$

subject to (5)

$$u^{\min} \le u(k+p|k) \le u^{\max}, \quad p=0,\ldots,N_{\mathrm{u}}-1,$$
$$\triangle u^{\min} \le \triangle u(k+p|k) \le \triangle u^{\max}, \quad p=0,\ldots,N_{\mathrm{u}}-1,$$
$$y^{\min} \le \hat{y}(k+p|k) \le y^{\max}, \quad p=1,\ldots,N,$$

where the vectors $u^{\min}$, $u^{\max} \in \mathbb{R}^{n_{\mathrm{u}}}$ define the constraints imposed on the magnitude of the manipulated variables, the vectors $\triangle u^{\min}$, $\triangle u^{\max} \in \mathbb{R}^{n_{\mathrm{u}}}$ define the constraints imposed on the increments of the manipulated variables, and the vectors $y^{\min}$, $y^{\max} \in \mathbb{R}^{n_{\mathrm{y}}}$ define the constraints imposed on the magnitude of the predicted output variables.

The total number of decision variables of the MPC algorithm is $n_{\mathrm{u}}N_{\mathrm{u}}$ while the number of constraints is $4n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}}N$. In practice, however, it is necessary to take into account the fact that satisfaction of the hard output constraints $y^{\min} \le \hat{y}(k+p|k) \le y^{\max}$ may be not possible (in such a case the feasible set of the MPC optimisation problem (5) is empty). A conceptually better method is to use soft output constraints (Maciejowski, 2002; Tatjewski, 2007). The predicted values of the output variables may temporarily violate the original hard constraints, but this enforces the existence of the feasible set. The MPC optimisation problem with soft output constraints is

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

subject to

$$u^{\min} \le u(k+p|k) \le u^{\max}, \quad p=0,\ldots,N_{\mathrm{u}}-1,$$
$$\triangle u^{\min} \le \triangle u(k+p|k) \le \triangle u^{\max}, \quad p=0,\ldots,N_{\mathrm{u}}-1,$$
$$y^{\min} - \varepsilon^{\min}(k+p) \le \hat{y}(k+p|k) \le y^{\max} + \varepsilon^{\max}(k+p), \quad p=1,\ldots,N,$$
$$\varepsilon^{\min}(k+p) \ge 0, \quad \varepsilon^{\max}(k+p) \ge 0, \quad p=1,\ldots,N. \tag{6}$$

When necessary (i.e., when the feasible set is empty), the original hard output constraints are temporarily relaxed. The vectors $\varepsilon^{\min}(k+p)$, $\varepsilon^{\max}(k+p) \in \mathbb{R}^{n_{\mathrm{y}}}$, which determine the degree of constraint violation for consecutive sampling instants over the prediction horizon ($p = 1,\ldots,N$), are the additional decision variables of the MPC optimisation problem (6). They have positive values only when the corresponding hard constraints are violated. The number of decision variables of the resulting MPC algorithm is $n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}}N$, the number of constraints is $4n_{\mathrm{u}}N_{\mathrm{u}} + 4n_{\mathrm{y}}N$ and $\rho^{\min}, \rho^{\max} > 0$ are penalty coefficients. In order to reduce computational complexity, one may assume that the same vectors $\varepsilon^{\min}(k)$, $\varepsilon^{\max}(k) \in \mathbb{R}^{n_{\mathrm{y}}}$ are used over the whole prediction horizon. In such a case, the number of decision variables drops to $n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}}$ while the number of constraints is $4n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}}N + 2n_{\mathrm{y}}$.

3. Nonlinear state-space predictive control with on-line linearisation and quadratic optimisation

3.1. Prediction model. In this work it is assumed that the state vector is affected by a state disturbance vector $\nu(k) \in \mathbb{R}^{n_{\mathrm{x}}}$ and the outputs by a disturbance vector $d(k) \in \mathbb{R}^{n_{\mathrm{y}}}$. In such a case, the nonlinear state-space model (1a)–(1b) becomes

$$x(k+1) = f(x(k), u(k)) + \nu(k),$$
$$y(k) = g(x(k)) + d(k).$$

The unknown vector $\nu(k)$ may be assessed as the difference between the measured state $x(k)$ and the state calculated from the state equation for the sampling instant k,

$$\nu(k) = x(k) - f(x(k-1), u(k-1)).$$

Moreover, it is assumed that the same state disturbance acts on the process over the whole prediction horizon, which means that

$$\nu(k+1|k) = \ldots = \nu(k+N|k) = \nu(k). \tag{7}$$

The state disturbance model presented above was discussed by Tatjewski (2014; 2007), but only for linear systems described by the state equation $x(k+1) = \boldsymbol{A}x(k)+\boldsymbol{B}u(k)+\nu(k)$. The predicted state vector for the sampling instant $k+p$ is calculated at the current instant k from

$$\hat{x}(k+p|k) = x(k+p|k) + \nu(k), \tag{8}$$

where $x(k+p|k)$ denotes the state vector obtained from the state equation (2a). For $p=1$, one has

$$\hat{x}(k+1|k) = f(x(k), u(k|k)) + \nu(k), \tag{9}$$

and for $p = 2, \dots, N$, the following recurrent formula is used:

$$\begin{aligned}\hat{x}(k+p|k) = f(\hat{x}(k+p-1|k), u(k+p-1|k)) \\ + \nu(k).\end{aligned} \tag{10}$$

Since it is assumed that the state vector is not measured, the state estimate $\tilde{x}(k)$ is used in place of $x(k)$. The vector $\nu(k)$ is then calculated from

$$\nu(k) = \tilde{x}(k) - f(\tilde{x}(k-1), u(k-1)), \tag{11}$$

and Eqn. (9) becomes

$$\hat{x}(k+1|k) = f(\tilde{x}(k), u(k|k)) + \nu(k). \tag{12}$$

The unmeasured output disturbance vector is calculated similarly to the state one (Tatjewski, 2014; 2007) as the difference between the measured output vector $y(k)$ and the output calculated from the output equation for the sampling instant k,

$$\begin{aligned}d(k) &= y(k) - g(x(k) + \nu(k)) \\ &= y(k) - g(f(x(k-1), u(k-1)) + \nu(k)).\end{aligned}$$

Similarly to Eqn. (7), it is assumed that the same output disturbance acts on the process over the whole prediction horizon, i.e.,

$$d(k+1|k) = \dots = d(k+N|k) = d(k). \tag{13}$$

Similarly to Eqn. (8), the predicted output vector for the sampling instant $k+p$ is calculated at the current instant k from

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

where $y(k+p|k)$ denotes the output vector obtained from the output equation (2b). One has

$$\hat{y}(k+p|k) = g(\hat{x}(k+p|k)) + d(k). \tag{14}$$

When the state is not measured but estimated, the output disturbance vector is calculated from

$$\begin{aligned}d(k) &= y(k) - g(\tilde{x}(k) + \nu(k)) \\ &= y(k) - g(f(\tilde{x}(k-1), u(k-1)) + \nu(k)).\end{aligned} \tag{15}$$

Taking into account Eqns. (12), (10) and (14), the output predictions are

$$\hat{y}(k+1|k) = g(f(\tilde{x}(k), u(k|k)) + \nu(k)) + d(k) \tag{16}$$

and

$$\begin{aligned}\hat{y}(k+p|k) = g(f(\hat{x}(k+p-1|k), u(k+p-1|k)) \\ + \nu(k)) + d(k)\end{aligned} \tag{17}$$

for $p = 2, \dots, N$. It is necessary to emphasise the fact that the state may be measured or estimated, but the output vector must always be measured.

To summarise, the state prediction equations (12) and (10) are very similar to the output prediction ones (16) and (17): in both the cases the predicted vector is a sum of a model output and of a disturbance estimate. The DMC prediction model is used (originally applied in the dynamic matrix control algorithm (Tatjewski, 2007)), in which it is assumed that the same disturbance acts on the process over the whole prediction horizon (Eqns. (7) and (13)).

3.2. Predictive control optimisation problem reformulation. Defining the following set-point trajectory and the predicted output trajectory vectors of length $n_{\mathrm{y}}N$:

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

the additional variables vectors of length $n_{\mathrm{y}}N$,

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

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

and the weighting matrices $\boldsymbol{M} = \mathrm{diag}(\boldsymbol{M}_1, \dots, \boldsymbol{M}_N)$ of dimensions $n_{\mathrm{y}}N \times n_{\mathrm{y}}N$ and $\boldsymbol{\Lambda} = \mathrm{diag}(\boldsymbol{\Lambda}_0, \dots, \boldsymbol{\Lambda}_{N_{\mathrm{u}}-1})$ of dimensions $n_{\mathrm{u}}N_{\mathrm{u}} \times n_{\mathrm{u}}N_{\mathrm{u}}$, the minimised cost-function used in the optimisation task (6) becomes

$$\begin{aligned}J(k) = \|\boldsymbol{y}^{\mathrm{sp}}(k) - \hat{\boldsymbol{y}}(k)\|^2_{\boldsymbol{M}} + \|\triangle \boldsymbol{u}(k)\|^2_{\boldsymbol{\Lambda}} \\ + \rho^{\min}\left\|\boldsymbol{\varepsilon}^{\min}\right\|^2 + \rho^{\max}\left\|\boldsymbol{\varepsilon}^{\max}\right\|^2.\end{aligned}$$

Defining the input constraints vectors of length $n_{\mathrm{u}}N_{\mathrm{u}}$,

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

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

and the output constraints vectors of length $n_{\mathrm{y}}N$,

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

the general MPC optimisation problem with soft output constraints (6) can be expressed as

$$\min_{\substack{\triangle \boldsymbol{u}(k)\\ \boldsymbol{\varepsilon}^{\min}(k)\\ \boldsymbol{\varepsilon}^{\max}(k)}} \Big\{ \|\boldsymbol{y}^{\mathrm{sp}}(k) - \hat{\boldsymbol{y}}(k)\|^2_{\boldsymbol{M}} + \|\triangle \boldsymbol{u}(k)\|^2_{\boldsymbol{\Lambda}} + \rho^{\min} \left\|\boldsymbol{\varepsilon}^{\min}(k)\right\|^2 + \rho^{\max} \left\|\boldsymbol{\varepsilon}^{\max}(k)\right\|^2 \Big\}, \tag{18}$$

subject to

$$\begin{gathered} \boldsymbol{u}^{\min} \le \boldsymbol{u}(k) \le \boldsymbol{u}^{\max}, \\ \triangle \boldsymbol{u}^{\min} \le \triangle \boldsymbol{u}(k) \le \triangle \boldsymbol{u}^{\max}, \\ \boldsymbol{y}^{\min} - \boldsymbol{\varepsilon}^{\min}(k) \le \hat{\boldsymbol{y}}(k) \le \boldsymbol{y}^{\max} + \boldsymbol{\varepsilon}^{\max}(k), \\ \boldsymbol{\varepsilon}^{\min}(k) \ge 0, \quad \boldsymbol{\varepsilon}^{\max}(k) \ge 0. \end{gathered}$$

If the nonlinear model (1a)–(1b) is used in MPC, the output predictions are nonlinear functions of the decision variables of the algorithm, i.e., the calculated control moves $\triangle \boldsymbol{u}(k)$. In consequence, the MPC optimisation problems (5), (6) or (18) are nonlinear tasks which must be solved on-line in real time. Although in some applications, e.g., in active noise control (Bismor, 2015), on-line nonlinear optimisation is used, in general it may be time-consuming and very difficult from a computational point of view (e.g., non-convex or multi-modal problems). That is why two computationally efficient alternatives with on-line linearisation are considered, which result in quadratic optimisation problems. They may be efficiently solved on-line.

In the first case the model is successively linearised on-line for the current operating point, while in the second case a linear approximation of the predicted output trajectory is found on-line. Although the necessity of the computational efficiency of MPC has been recognised in the literature (Tatjewski, 2007; Tatjewski and Ławryńczuk, 2006), the algorithms with simple successive on-line model linearisation are predominant, (e.g., Colin *et al.*, 2007; Mu *et al.*, 2005). More advanced MPC algorithms with on-line linearisation and the nonlinear free trajectory are described, e.g., by Arahal *et al.* (1998), Ławryńczuk (2007), or Tatjewski and Ławryńczuk (2006). In all of these works input-output models are used. MPC algorithms for state-space models and with simple on-line model linearisation are described by Lee and Ricker (1994), algorithms with successive linearisation and a nonlinear free trajectory are described by Ławryńczuk (2014) and Megías *et al.* (1999), linearisation along the set-point vector is analysed by Kuure-Kinsey *et al.* (2006), while more advanced linearisation methods are thoroughly discussed by Ławryńczuk (2014). Unfortunately, in many cases the state estimation problem is not addressed (Ławryńczuk, 2014; Megías *et al.*, 1999), which is crucial in the case of the state-space approach. Finally, for state-space models it is possible to formulate an MPC scheme with a Newton-type optimisation algorithm (de Oliveira and Biegler, 1995). An alternative to on-line linearisation is MPC with feedback linearisation (Deng *et al.*, 2009).

3.3. State-space predictive control with on-line model linearisation: The MPC algorithm with nonlinear prediction and linearisation (MPC-NPL).

Using the Taylor series expansion method, the linear approximation of the nonlinear state-space model (2a) is

$$\begin{aligned} x(k) &= f(\bar{x}(k-1), \bar{u}(k-1)) \\ &\quad + \boldsymbol{A}(k)(x(k-1) - \bar{x}(k-1)) \\ &\quad + \boldsymbol{B}(k)(u(k-1) - \bar{u}(k-1)), \\ y(k) &= g(\bar{x}(k)) + \boldsymbol{C}(k)(x(k) - \bar{x}(k)), \end{aligned}$$

where the measurement vectors $\bar{x}(k-1) = \tilde{x}(k-1)$, $\bar{x}(k) = \tilde{x}(k)$ and $\bar{u}(k-1)$ define the current operating point of the process, whereas the vectors $x(k-1)$, $x(k)$ and $u(k-1)$ are arguments (independent variables) of the linearised model. Thanks to a proper change of variables with respect to the current operating point, one obtains

$$x(k) = \boldsymbol{A}(k)x(k-1) + \boldsymbol{B}(k)u(k-1), \tag{19a}$$

$$y(k) = \boldsymbol{C}(k)x(k). \tag{19b}$$

The matrices of the linearised model, of dimensions $n_{\mathrm{x}} \times n_{\mathrm{x}}$, $n_{\mathrm{x}} \times n_{\mathrm{u}}$ and $n_{\mathrm{y}} \times n_{\mathrm{x}}$, respectively, are calculated analytically on-line from the general equations,

$$\begin{aligned} \boldsymbol{A}(k) &= \left.\frac{\partial f(x(k-1), u(k-1))}{\partial x(k-1)}\right|_{\substack{x(k-1)=\bar{x}(k-1)\\ u(k-1)=\bar{u}(k-1)}}, \\ \boldsymbol{B}(k) &= \left.\frac{\partial f(x(k-1), u(k-1))}{\partial u(k-1)}\right|_{\substack{x(k-1)=\bar{x}(k-1)\\ u(k-1)=\bar{u}(k-1)}}, \\ \boldsymbol{C}(k) &= \left.\frac{\mathrm{d}g(x(k))}{\mathrm{d}x(k)}\right|_{x(k)=\bar{x}(k)}. \end{aligned} \tag{20}$$

The linearised state equation (19a) may be used to calculate from the prediction equations (9) and (10) the predicted states

$$\begin{aligned} \hat{x}(k+1|k) &= \boldsymbol{A}(k)\tilde{x}(k) + \boldsymbol{B}(k)u(k|k) + \nu(k), \\ \hat{x}(k+2|k) &= \boldsymbol{A}(k)\hat{x}(k+1|k) + \boldsymbol{B}(k)u(k+1|k) + \nu(k), \\ \hat{x}(k+3|k) &= \boldsymbol{A}(k)\hat{x}(k+2|k) + \boldsymbol{B}(k)u(k+2|k) + \nu(k). \\ &\vdots \end{aligned}$$

The state predictions may be expressed as functions of the increments in the future control increments which are the decision variables of the predictive control algorithm,

$$\begin{aligned}\hat{x}(k+1|k) &= \boldsymbol{B}(k)\triangle u(k|k) + \dots,\\ \hat{x}(k+2|k) &= (\boldsymbol{A}(k)+\boldsymbol{I})\boldsymbol{B}(k)\triangle u(k|k)\\ &\quad + \boldsymbol{B}(k)\triangle u(k+1|k) + \dots,\\ \hat{x}(k+3|k) &= (\boldsymbol{A}^2(k)+\boldsymbol{A}(k)+\boldsymbol{I})\boldsymbol{B}(k)\triangle u(k|k)\\ &\quad + (\boldsymbol{A}(k)+\boldsymbol{I})\boldsymbol{B}(k)\triangle u(k+1|k)\\ &\quad + \boldsymbol{B}(k)\triangle u(k+2|k) + \dots, \qquad (21)\\ &\ \ \vdots\end{aligned}$$

It is assumed, similarly to MPC algorithms based on time-invariant linear models (Camacho and Bordons, 1999; Tatjewski, 2007), that the predicted trajectory is a sum of the forced trajectory which depends only on the future (on the future control moves $\triangle\boldsymbol{u}(k)$) and the free state trajectory which depends only on the past. From Eqn. (21), the state vector predicted over the prediction horizon, of length $n_{\mathrm{x}}N$, is

$$\hat{\boldsymbol{x}}(k) = \begin{bmatrix}\hat{x}(k+1|k)\\ \vdots\\ \hat{x}(k+N|k)\end{bmatrix} = \boldsymbol{P}(k)\triangle\boldsymbol{u}(k) + \boldsymbol{x}^0(k), \qquad (22)$$

where the structure of the matrix $\boldsymbol{P}(k)$, of dimensionality $n_{\mathrm{x}}N \times n_{\mathrm{u}}N_{\mathrm{u}}$, is defined by Eqn. (23) and the state free trajectory vector

$$\boldsymbol{x}^0(k) = \begin{bmatrix}x^0(k+1|k)\\ \vdots\\ x^0(k+N|k)\end{bmatrix},$$

is of length $n_{\mathrm{x}}N$. It is calculated from Eqns. (12) and (10) taking into account only the past, i.e., assuming that $u(k+p|k) = u(k-1)$ for $p = 1,\dots,N$. One obtains

$$x^0(k+1|k) = f(\tilde{x}(k), u(k-1)) + \nu(k), \qquad (24)$$

and

$$x^0(k+p|k) = f(x^0(k+p-1|k), u(k-1)) + \nu(k), \qquad (25)$$

for $p = 2,\dots,N$. In order to calculate the predicted output trajectory it is also assumed, in the same way it is done for state prediction in Eqn. (22), that the predicted output trajectory is a sum of forced and free trajectories.

From the linearised output equation (19b), one has

$$\hat{\boldsymbol{y}}(k) = \widetilde{\boldsymbol{C}}(k)\boldsymbol{P}(k)\triangle\boldsymbol{u}(k) + \boldsymbol{y}^0(k), \qquad (26)$$

where the matrix $\widetilde{\boldsymbol{C}}(k) = \mathrm{diag}(\boldsymbol{C}(k),\dots,\boldsymbol{C}(k))$ is of dimensions $n_{\mathrm{y}}N \times n_{\mathrm{x}}N$. The output free trajectory vector

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

is of length $n_{\mathrm{y}}N$. It is calculated from Eqn. (14) taking into account only the past, i.e., the free state trajectory

$$y^0(k+p|k) = g(x^0(k+p)) + d(k). \qquad (27)$$

On-line model linearisation leads to the output prediction equation (26), which is a linear function of the calculated future control policy $\triangle\boldsymbol{u}(k)$. This means that the general MPC optimisation problem (18) can be transformed to the following quadratic optimisation task:

$$\min_{\substack{\triangle\boldsymbol{u}(k)\\ \boldsymbol{\varepsilon}^{\min}(k)\\ \boldsymbol{\varepsilon}^{\max}(k)}} \Big\{ \left\| \boldsymbol{y}^{\mathrm{sp}}(k) - \widetilde{\boldsymbol{C}}(k)\boldsymbol{P}(k)\triangle\boldsymbol{u}(k) - \boldsymbol{y}^0(k) \right\|_{\boldsymbol{M}}^2 + \left\|\triangle\boldsymbol{u}(k)\right\|_{\boldsymbol{\Lambda}}^2 + \rho^{\min}\left\|\boldsymbol{\varepsilon}^{\min}(k)\right\|^2 + \rho^{\max}\left\|\boldsymbol{\varepsilon}^{\max}(k)\right\|^2 \Big\}, \qquad (28)$$

subject to

$$\begin{gathered}\boldsymbol{u}^{\min} \le \boldsymbol{J}\triangle\boldsymbol{u}(k) + \boldsymbol{u}(k-1) \le \boldsymbol{u}^{\max},\\ \triangle\boldsymbol{u}^{\min} \le \triangle\boldsymbol{u}(k) \le \triangle\boldsymbol{u}^{\max},\\ \boldsymbol{y}^{\min} - \boldsymbol{\varepsilon}^{\min}(k) \le \widetilde{\boldsymbol{C}}(k)\boldsymbol{P}(k)\triangle\boldsymbol{u}(k) + \boldsymbol{y}^0(k)\\ \le \boldsymbol{y}^{\max} + \boldsymbol{\varepsilon}^{\max}(k),\\ \boldsymbol{\varepsilon}^{\min}(k) \ge 0, \quad \boldsymbol{\varepsilon}^{\max}(k) \ge 0,\end{gathered}$$

where the vector

$$\boldsymbol{u}(k-1) = \begin{bmatrix}u(k-1)\\ \vdots\\ u(k-1)\end{bmatrix} \qquad (29)$$

is of length $n_{\mathrm{u}}N_{\mathrm{u}}$ and the matrix

$$\boldsymbol{J} = \begin{bmatrix}\boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \dots & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}}\\ \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \dots & \boldsymbol{0}_{n_{\mathrm{u}}\times n_{\mathrm{u}}}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}} & \dots & \boldsymbol{I}_{n_{\mathrm{u}}\times n_{\mathrm{u}}}\end{bmatrix} \qquad (30)$$

is of dimensions $n_{\mathrm{u}}N_{\mathrm{u}} \times n_{\mathrm{u}}N_{\mathrm{u}}$.

Algorithm 1 summarises calculations carried out on-line at each sampling instant k of the MPC-NPL approach. The proof that the algorithm guarantees offset-free control is given in Appendix.

3.4. State-space predictive control with on-line trajectory linearisation: The MPC algorithm with nonlinear prediction and linearisation along the predicted trajectory (MPC-NPLPT). The MPC-NPL algorithm calculates at each sampling instant on-line a linear approximation of the nonlinear state-space model taking into account the values of the (estimated) state $\tilde{x}(k)$, $\tilde{x}(k-1)$ and the most recent available control signals $u(k-1)$. The same linearised model is recurrently used for state and output prediction calculation for the whole prediction horizon. Because the linearised model describes well properties of the process only in some neighbourhood

$$\boldsymbol{P}(k)=\begin{bmatrix} \boldsymbol{B}(k) & \dots & \boldsymbol{0}_{n_{\mathrm{x}}\times n_{\mathrm{u}}} \\ (\boldsymbol{A}(k)+\boldsymbol{I}_{n_{\mathrm{x}}\times n_{\mathrm{x}}})\boldsymbol{B}(k) & \dots & \boldsymbol{0}_{n_{\mathrm{x}}\times n_{\mathrm{u}}} \\ \vdots & \ddots & \vdots \\ \left(\sum_{i=1}^{N_{\mathrm{u}}-1}\boldsymbol{A}^i(k)+\boldsymbol{I}_{n_{\mathrm{x}}\times n_{\mathrm{x}}}\right)\boldsymbol{B}(k) & \dots & \boldsymbol{B}(k) \\ \left(\sum_{i=1}^{N_{\mathrm{u}}}\boldsymbol{A}^i(k)+\boldsymbol{I}_{n_{\mathrm{x}}\times n_{\mathrm{x}}}\right)\boldsymbol{B}(k) & \dots & (\boldsymbol{A}(k)+\boldsymbol{I}_{n_{\mathrm{x}}\times n_{\mathrm{x}}})\boldsymbol{B}(k) \\ \vdots & \ddots & \vdots \\ \left(\sum_{i=1}^{N-1}\boldsymbol{A}^i(k)+\boldsymbol{I}_{n_{\mathrm{x}}\times n_{\mathrm{x}}}\right)\boldsymbol{B}(k) & \dots & \left(\sum_{i=1}^{N-N_{\mathrm{u}}}\boldsymbol{A}^i(k)+\boldsymbol{I}_{n_{\mathrm{x}}\times n_{\mathrm{x}}}\right)\boldsymbol{B}(k) \end{bmatrix}, \tag{23}$$

Algorithm 1. MPC-NPL algorithm with state estimation.

Step 1. The current state vector $\tilde{x}(k)$ is estimated, the output vector $y(k)$ is measured.

Step 2. The matrices $\boldsymbol{A}(k)$, $\boldsymbol{B}(k)$, $\boldsymbol{C}(k)$ of the local linear approximation (19a)–(19b) of the nonlinear model (1a)–(1b) are calculated for the current operating point of the process from Eqns. (20).

Step 3. The matrix $\boldsymbol{P}(k)$ is calculated from Eqn. (23).

Step 4. The state disturbance vector $\nu(k)$ is estimated from Eqn. (11), the output disturbance vector $d(k)$ is calculated from Eqn. (15).

Step 5. The nonlinear model of the process (1a)–(1b) is used to find nonlinear free state trajectory $\boldsymbol{x}^0(k)$ from Eqns. (24) and (25) and the nonlinear free output trajectory $\boldsymbol{y}^0(k)$ from Eqn. (27).

Step 6. The quadratic optimisation problem (28) is solved to calculate the future control increments vector $\triangle\boldsymbol{u}(k)$.

Step 7. The first n_{u} elements of the sequence $\triangle\boldsymbol{u}(k)$ are applied to the process, i.e., $u(k)=\triangle u(k|k)+u(k-1)$.

Step 8. At the next sampling instant, i.e., $k:=k+1$, the algorithm goes to Step 1.

of the operating point, it may give insufficient control accuracy when the set-point trajectory changes rapidly and significantly.

In order to increase prediction accuracy (and control accuracy), a more advanced approach is discussed, in which the predicted trajectory is linearised for some future control scenario, not for an operating point defined by past signals. Furthermore, trajectory linearisation and optimisation of the future control scenario are repeated a few times at each sampling instant in the internal iterations.

Let t be the index of internal iterations ($t=1,\dots,t_{\max}$). In the internal iteration t, the predicted nonlinear output trajectory

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

is linearised along the future input trajectory

$$\boldsymbol{u}^{t-1}(k)=\begin{bmatrix}u^{t-1}(k|k)\\ \vdots \\ u^{t-1}(k+N|k)\end{bmatrix},$$

found in the previous internal iteration ($t-1$). The input trajectory $\boldsymbol{u}^{t-1}(k)$ corresponds to the predicted output trajectory $\hat{\boldsymbol{y}}^{t-1}(k)$; i.e., the output trajectory may be calculated for the input one from the model of the process. Taking into account the general output prediction equations (14), (16) and (17), it is calculated from

$$\begin{aligned}\hat{y}^{t-1}(k+1|k)&=g(\hat{x}^{t-1}(k+1|k))+d(k)\\ &=g(f(\tilde{x}(k),u^{t-1}(k|k))+\nu(k))\\ &\quad+d(k)\end{aligned} \tag{31}$$

and

$$\begin{aligned}\hat{y}^{t-1}(k+p|k)&=g(\hat{x}^{t-1}(k+p|k))+d(k)\\ &=g(f(\hat{x}^{t-1}(k+p-1|k),\\ &\qquad u^{t-1}(k+p-1|k))+\nu(k))\\ &\quad+d(k),\end{aligned} \tag{32}$$

for $p=2,\dots,N$. The input trajectory $\boldsymbol{u}^{t-1}(k)$, along which the output trajectory is linearised on-line, may be initialised using the last $n_{\mathrm{u}}(N_{\mathrm{u}}-1)$ elements of the future control sequence calculated at the previous sampling instant ($k-1$) and not applied to the process.

Using the Taylor series expansion method, the linear approximation of the nonlinear output trajectory $\hat{\boldsymbol{y}}^t(k)$ along the input trajectory $\boldsymbol{u}^{t-1}(k)$, i.e., linearisation of the function $\hat{\boldsymbol{y}}^t(\boldsymbol{u}^t(k))\colon \mathbb{R}^{n_{\mathrm{u}}N_{\mathrm{u}}}\rightarrow\mathbb{R}^{n_{\mathrm{y}}N}$, is

$$\hat{\boldsymbol{y}}^t(k)=\hat{\boldsymbol{y}}^{t-1}(k)+\boldsymbol{H}^t(k)(\boldsymbol{u}^t(k)-\boldsymbol{u}^{t-1}(k)). \tag{33}$$

The control values $\boldsymbol{u}^t(k)$ calculated at the current internal iteration t are the arguments of the function $\hat{\boldsymbol{y}}^t(k)$, whereas the vectors $\boldsymbol{u}^{t-1}(k)$ and $\hat{\boldsymbol{y}}^{t-1}(k)$ are fixed for the current initial iteration. The matrix of the derivatives of the predicted output trajectory with respect to the future control signals is of dimensions $n_{\mathrm{y}}N \times n_{\mathrm{u}}N_{\mathrm{u}}$ and it has the general structure

$$\begin{aligned}\boldsymbol{H}^t(k) &= \left.\frac{\mathrm{d}\hat{\boldsymbol{y}}(k)}{\mathrm{d}\boldsymbol{u}(k)}\right|_{\substack{\hat{\boldsymbol{y}}(k)=\hat{\boldsymbol{y}}^{t-1}(k)\\ \boldsymbol{u}(k)=\boldsymbol{u}^{t-1}(k)}} = \frac{\mathrm{d}\hat{\boldsymbol{y}}^{t-1}(k)}{\mathrm{d}\boldsymbol{u}^{t-1}(k)}\\ &= \begin{bmatrix} \frac{\partial \hat{y}^{t-1}(k+1|k)}{\partial u^{t-1}(k|k)} & \cdots & \frac{\partial \hat{y}^{t-1}(k+1|k)}{\partial u^{t-1}(k+N_{\mathrm{u}}-1|k)}\\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}^{t-1}(k+N|k)}{\partial u^{t-1}(k|k)} & \cdots & \frac{\partial \hat{y}^{t-1}(k+N|k)}{\partial u^{t-1}(k+N_{\mathrm{u}}-1|k)}\end{bmatrix}.\end{aligned} \tag{34}$$

From the output prediction equations (31) and (32), it is possible to calculate the entries of the matrix $\boldsymbol{H}^t(k)$. The general formulae for trajectory linearisation are

$$\begin{aligned}\frac{\partial \hat{y}^{t-1}(k+1|k)}{\partial u^{t-1}(k+r|k)} &= \frac{\partial g(\hat{x}^{t-1}(k+1|k))}{\partial u^{t-1}(k+r|k)}\\ &\quad\times \frac{\partial f(\tilde{x}(k), u^{t-1}(k|k))}{\partial u^{t-1}(k+r|k)},\end{aligned} \tag{35}$$

for $r = 0, \dots, N_{\mathrm{u}} - 1$, and

$$\begin{aligned}&\frac{\partial \hat{y}^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}\\ &= \frac{\partial g(\hat{x}^{t-1}(k+p|k))}{\partial u^{t-1}(k+r|k)}\\ &\quad\times \frac{\partial f(\hat{x}^{t-1}(k+p-1|k), u^{t-1}(k+p-1|k))}{\partial u^{t-1}(k+r|k)},\end{aligned} \tag{36}$$

for $p = 2, \dots, N$, $r = 0, \dots, N_{\mathrm{u}} - 1$. Using the vector $\boldsymbol{u}(k-1)$ defined by Eqn. (29) and the matrix $\boldsymbol{J}$ defined by Eqn. (30), the linear approximation of the nonlinear predicted output trajectory (33) becomes

$$\begin{aligned}\hat{\boldsymbol{y}}^t(k) &= \boldsymbol{H}^t(k)\boldsymbol{J}\triangle\boldsymbol{u}^t(k) + \hat{\boldsymbol{y}}^{t-1}(k)\\ &\quad + \boldsymbol{H}^t(k)(\boldsymbol{u}(k-1) - \boldsymbol{u}^{t-1}(k)).\end{aligned} \tag{37}$$

On-line trajectory linearisation leads to the output prediction equation (37), which is a linear function of the future control policy $\triangle\boldsymbol{u}^t(k)$ calculated for the current sampling instant k and at the internal iteration t. Furthermore, the general MPC optimisation problem (18) can be transformed to the following quadratic optimisation task:

$$\begin{aligned}\min_{\substack{\triangle\boldsymbol{u}^t(k)\\ \boldsymbol{\varepsilon}^{\min}(k)\\ \boldsymbol{\varepsilon}^{\max}(k)}} \Big\{ &\left\|\boldsymbol{y}^{\mathrm{sp}}(k) - \boldsymbol{H}^t(k)\boldsymbol{J}\triangle\boldsymbol{u}^t(k) - \hat{\boldsymbol{y}}^{t-1}(k)\right.\\ &\left. -\boldsymbol{H}^t(k)(\boldsymbol{u}(k-1) - \boldsymbol{u}^{t-1}(k))\right\|^2\\ &+ \left\|\triangle\boldsymbol{u}^t(k)\right\|^2_{\boldsymbol{\Lambda}}\\ &+ \rho^{\min}\left\|\boldsymbol{\varepsilon}^{\min}(k)\right\|^2 + \rho^{\max}\left\|\boldsymbol{\varepsilon}^{\max}(k)\right\|^2 \Big\},\end{aligned}$$

subject to (38)

$$\begin{aligned}\boldsymbol{u}^{\min} \le \boldsymbol{J}\triangle\boldsymbol{u}^t(k) + \boldsymbol{u}(k-1) &\le \boldsymbol{u}^{\max},\\ \triangle\boldsymbol{u}^{\min} \le \triangle\boldsymbol{u}^t(k) &\le \triangle\boldsymbol{u}^{\max},\\ \boldsymbol{y}^{\min} - \boldsymbol{\varepsilon}^{\min}(k) \le \boldsymbol{H}^t(k)\boldsymbol{J}&\triangle\boldsymbol{u}^t(k) + \hat{\boldsymbol{y}}^{t-1}(k)\\ &+ \boldsymbol{H}^t(k)(\boldsymbol{u}(k-1) - \boldsymbol{u}^{t-1}(k))\\ &\le \boldsymbol{y}^{\max} + \boldsymbol{\varepsilon}^{\max}(k),\\ \boldsymbol{\varepsilon}^{\min}(k) \ge 0, \quad &\boldsymbol{\varepsilon}^{\max}(k) \ge 0.\end{aligned}$$

More than one internal iteration may be necessary if the set-point changes in the consecutive sampling instants are significant, i.e., when

$$\sum_{p=0}^{N_0}(y^{\mathrm{sp}}(k-p) - y(k-p))^2 \ge \delta_{\mathrm{y}}. \tag{39}$$

If the difference between the future control increments calculated in two consecutive internal iterations is not significant, i.e., when

$$\left\|\triangle\boldsymbol{u}^t(k) - \triangle\boldsymbol{u}^{t-1}(k)\right\|^2 < \delta_{\mathrm{u}}, \tag{40}$$

the internal iterations are terminated. The parameters N_0, $\delta_{\mathrm{y}} > 0$ and $\delta_{\mathrm{u}} > 0$ are adjusted experimentally.

Algorithm 2 summarises calculations carried out on-line at each sampling instant k of the MPC-NPLPT approach. The proof that the algorithm guarantees offset-free control is given in Appendix.

3.5. State estimation.

The classical extended Kalman filter (Simon, 2006) is used for state estimation. For this purpose, the model of the process is

$$\begin{aligned}x(x) &= f(x(k-1), u(k-1)) + w(k-1),\\ y(x) &= g(x(k)) + v(k),\end{aligned}$$

where $w(k-1)$ and $v(k)$ are the process and observation (measurement) noises, respectively. They are assumed to be zero mean, Gaussian and uncorrelated noises with covariance matrices $\boldsymbol{Q}(k-1) = E([w(k-1)w^{\mathrm{T}}(k-1)])$ and $\boldsymbol{R}(k) = E([v(k)v^{\mathrm{T}}(k)])$, respectively. State estimation requires successive on-line model linearisation

$$\begin{aligned}\boldsymbol{F}(k-1) &= \left.\frac{\partial f(x(k-1), u(k-1))}{\partial x(k-1)}\right|_{x(k-1)=\tilde{x}(k-1|k-1)},\\ \boldsymbol{H}(k) &= \left.\frac{\partial g(x(k))}{\partial x(k)}\right|_{x(k)=\tilde{x}(k|k-1)}.\end{aligned}$$

When constant matrices $\boldsymbol{F}$ and $\boldsymbol{H}$ are used, one obtains the Kalman filter for linear systems.

Algorithm 2. MPC-NPLPT algorithm with state estimation.

```
Step 1. The current state vector $\tilde{x}(k)$ is estimated, the output vector $y(k)$ is measured.
Step 2. The state disturbance vector $\nu(k)$ is estimated from Eqn. (11), the output disturbance vector $d(k)$ is calculated from Eqn. (15).
Step 3. The first internal iteration ($t = 1$): the predicted output trajectory $\hat{\boldsymbol{y}}^0(k)$ is calculated for the assumed input trajectory $\boldsymbol{u}^0(k)$ using the nonlinear model from Eqns. (31) and (32).
Step 4. The nonlinear model (1a)–(1b) is used to find the linear approximation of the predicted trajectory $\hat{\boldsymbol{y}}^1(k)$ along the trajectory $\boldsymbol{u}^0(k)$; i.e., the entries of the matrix $\boldsymbol{H}^1(k)$ given by Eqn. (34) are found from Eqns. (35) and (36).
Step 5. The quadratic optimisation problem (38) is solved to calculate the future control increments vector $\triangle\boldsymbol{u}^1(k)$.
Step 6. If the condition (39) is satisfied, the internal iterations are continued for $t = 2, \ldots, t_{\max}$.
    Step 6.1. The predicted output trajectory $\hat{\boldsymbol{y}}^{t-1}(k)$ is calculated for the input trajectory $\boldsymbol{u}^{t-1}(k) = \boldsymbol{J}\triangle\boldsymbol{u}^{t-1}(k)+\boldsymbol{u}(k-1)$ using the nonlinear model from Eqns. (31) and (32).
    Step 6.2. The nonlinear model is used to find the linear approximation of the predicted trajectory $\hat{\boldsymbol{y}}^t(k)$ along the trajectory $\boldsymbol{u}^{t-1}(k)$; i.e., the entries of the matrix $\boldsymbol{H}^t(k)$ given by Eqn. (34) are found from Eqns. (35) and (36).
    Step 6.3. The quadratic optimisation problem (38) is solved to calculate the future control increments vector $\triangle\boldsymbol{u}^t(k)$ for the current internal iteration.
    Step 6.4. If the condition (40) is satisfied or $t > t_{\max}$, the internal iterations are terminated. Otherwise, the internal iteration index is increased, i.e., $t := t + 1$, the algorithms goes to Step 6.1.
Step 7. The first $n_{\rm u}$ elements of the sequence $\triangle\boldsymbol{u}^t(k)$ are applied to the process, i.e., $u(k) = \triangle u^t(k|k) + u(k-1)$.
Step 8. At the next sampling instant, i.e., $k\colon = k + 1$, the algorithm goes to Step 1.
```

4. Simulations

4.1. Polymerisation reactor benchmark. In order to evaluate and compare the discussed MPC algorithms, a polymerisation reaction taking place in a jacketed continuous stirred tank reactor (Doyle *et al.*, 1995) is considered. The reaction is the free-radical polymerisation of methyl methacrylate with azo-bis-isobutyronitrile as the initiator and toluene as the solvent. The output y (number average molecular weight, NAMW) [kg kmol^{-1}] is controlled by manipulating the inlet initiator flow rate u [m^3 h^{-1}]. Under some technological assumptions (Doyle *et al.*, 1995), the continuous-time fundamental model of the polymerisation reactor is comprised of four nonlinear ordinary differential equations and one algebraic output equation,

$$\begin{aligned}
\dot{x}_1(t) &= 60 - 10x_1(t) - 2.4568x_1(t)\sqrt{x_2(t)},\\
\dot{x}_2(t) &= 80u(t) - 10.1022x_2(t),\\
\dot{x}_3(t) &= 0.0024121x_1(t)\sqrt{x_2(t)} + 0.112191x_2(t)\\
&\quad - 10x_3(t),\\
\dot{x}_4(t) &= 245.978x_1(t)\sqrt{x_2(t)} - 10x_4(t),\\
y(t) &= \frac{x_4(t)}{x_3(t)}.
\end{aligned}$$

The initial operating conditions are $u = 0.028328$ m^3 h^{-1}, $y = 20000$ kg kmol^{-1}, $x_1 = 5.3745$ kmol m^{-3}, $x_2 = 2.2433 \times 10^{-1}$ kmol m^{-3}, $x_3 = 3.1308 \times 10^{-3}$ kmol m^{-3}, $x_4 = 6.2616 \times 10^{-1}$ kmol m^{-3}. The continuous-time state-space model is discretised with the sampling time $T_{\rm s} = 1.8$ s by means of Euler's method,

$$\begin{aligned}
x_1(k+1) &= x_1(k) + T_{\rm s}\big(60 - 10x_1(k) - 2.4568\alpha(k)\big),\\
x_2(k+1) &= x_2(k) + T_{\rm s}\big(80u(k) - 10.1022x_2(k)\big),\\
x_3(k+1) &= x_3(k) + T_{\rm s}\big(0.0024121\alpha(k)\\
&\qquad + 0.112191x_2(k) - 10x_3(k)\big),\\
x_4(k+1) &= x_4(k) + T_{\rm s}\big(245.978\alpha(k) - 10x_4(k)\big),\\
y(k) &= \frac{x_4(k)}{x_3(k)},
\end{aligned} \tag{41}$$

where

$$\alpha(k) = x_1(k)\sqrt{x_2(k)}. \tag{42}$$

The following MPC strategies are compared:

(a) the linear MPC algorithm (Tatjewski, 2014; 2007),

(b) the discussed nonlinear MPC-NPL algorithm with successive on-line model linearisation,

(c) the discussed nonlinear MPC-NPLPT algorithm with successive on-line trajectory linearisation,

(d) the "ideal" MPC algorithm with nonlinear optimisation (MPC-NO).

The linear algorithm and algorithms with on-line linearisation require on-line quadratic optimisation. Because the MPC-NO strategy uses for prediction the nonlinear model without any simplifications, it requires on-line nonlinear optimisation. Parameters of all algorithms are the same: $N = 10$, $N_{\rm u} = 3$, $\boldsymbol{M}_p = 1$, $\boldsymbol{\Lambda}_p = 5 \times 10^{10}$, the additional parameters of the MPC-NPLPT algorithm are $N_0 = 3$, $\delta_{\rm y} = 100$ and $\delta_{\rm u} = 0.00001$ (tuning is described elsewhere (Ławryńczuk, 2014)), the constraints are $u^{\min} = 0.003$, $u^{\max} = 0.06$.

The process is simulated by the system described by Eqns. (41) and (42), and the same set of equations is used as the model in all nonlinear MPC algorithms. For state estimation, the Kalman filter is used in the linear MPC algorithm while the extended Kalman filter is used in all nonlinear ones. Their parameters are the initial value of the predicted covariance estimate is $\boldsymbol{P}(1|0) = 100\boldsymbol{I}_{4\times 4}$ and the covariance matrices of process and observation noises are $\boldsymbol{Q} = 0.1\boldsymbol{I}_{4\times 4}$ and $\boldsymbol{R} = 1$, respectively.

4.2. Implementation of the linear MPC algorithm for the benchmark process. The linear MPC algorithm (Tatjewski, 2014) with the state disturbance model (7) and the output disturbance model (13) is used. For prediction the algorithm uses a linear model

$$\begin{aligned} x(k+1) &= \boldsymbol{A}x(k) + \boldsymbol{B}u(k),\\ y(k) &= \boldsymbol{C}x(k), \end{aligned}$$

whose parameters correspond to the nominal operating point of the process. Model matrices are

$$\boldsymbol{A} = \begin{bmatrix} 6.6509\times 10^{-1} & -4.1818\times 10^{-1} & 0 & 0\\ 0 & 6.9693\times 10^{-1} & 0 & 0\\ 3.4274\times 10^{-5} & 3.7763\times 10^{-3} & 0.7 & 0\\ 3.4951 & 4.1868\times 10^{1} & 0 & 0.7 \end{bmatrix},$$

$$\boldsymbol{B} = \begin{bmatrix} 0\\ 2.4\\ 0\\ 0 \end{bmatrix},$$

$$\boldsymbol{C} = \begin{bmatrix} 0 & 0 & -6.3881\times 10^{6} & 3.1940\times 10^{2} \end{bmatrix}.$$

4.3. Implementation of the MPC-NPL algorithm for the benchmark process. For the nonlinear state-space model defined by Eqns. (41) and (42), from Eqn. (20) the matrices of the linearised model (19a)–(19b) are

$$\boldsymbol{A}(k) = \begin{bmatrix} a_{1,1}(k) & a_{1,2}(k) & 0 & 0\\ 0 & a_{2,2} & 0 & 0\\ a_{3,1}(k) & a_{3,2}(k) & a_{3,3} & 0\\ a_{4,1}(k) & a_{4,2}(k) & 0 & a_{4,4} \end{bmatrix},$$

$$\boldsymbol{B}(k) = \begin{bmatrix} 0\\ b_{2,1}\\ 0\\ 0 \end{bmatrix},$$

$$\boldsymbol{C}(k) = \begin{bmatrix} 0 & 0 & c_{1,3}(k) & c_{1,4}(k) \end{bmatrix},$$

where

$$\begin{aligned} a_{1,1}(k) &= 1 + T_{\mathrm{s}}(-10 - 2.4568\sqrt{\tilde{x}_2(k-1)}),\\ a_{1,2}(k) &= -1.2284T_{\mathrm{s}}\frac{\tilde{x}_1(k-1)}{\sqrt{\tilde{x}_2(k-1)}},\\ a_{2,2} &= 1 - 10.1022T_{\mathrm{s}},\\ a_{3,1}(k) &= 0.0024121T_{\mathrm{s}}\sqrt{\tilde{x}_2(k-1)},\\ a_{3,2}(k) &= T_{\mathrm{s}}\left(0.00120605\frac{\tilde{x}_1(k-1)}{\sqrt{\tilde{x}_2(k-1)}} + 0.112191\right),\\ a_{3,3} &= 1 - 10T_{\mathrm{s}},\\ a_{4,4} &= 1 - 10T_{\mathrm{s}},\\ a_{4,1}(k) &= 245.978T_{\mathrm{s}}\sqrt{\tilde{x}_2(k-1)},\\ a_{4,2}(k) &= 122.989T_{\mathrm{s}}\frac{\tilde{x}_1(k-1)}{\sqrt{\tilde{x}_2(k-1)}},\\ b_{2,1} &= 2.4,\\ c_{1,3}(k) &= -\frac{\tilde{x}_4(k)}{(\tilde{x}_3(k))^2},\\ c_{1,4}(k) &= \frac{1}{\tilde{x}_3(k)}. \end{aligned}$$

One may note that the the extended Kalman filter matrix $\boldsymbol{F}(k)$ is the same as the model matrix $\boldsymbol{A}(k)$ used in MPC, but $\boldsymbol{H}(k) = \boldsymbol{C}(k-1)$. The state disturbances are estimated from Eqn. (11), which gives

$$\begin{aligned} &\nu(k)\\ &= [\tilde{x}_1(k)\ \tilde{x}_2(k)\ \tilde{x}_3(k)\ \tilde{x}_4(k)]^{\mathrm{T}} \qquad (43)\\ &- \begin{bmatrix} (1-10T_{\mathrm{s}})\tilde{x}_1(k-1) + 60T_{\mathrm{s}}\\ \quad -2.4568T_{\mathrm{s}}\tilde{\alpha}(k-1)\\ (1-10.1022T_{\mathrm{s}})\tilde{x}_2(k-1) + 80T_{\mathrm{s}}u(k-1)\\ 0.112191T_{\mathrm{s}}\tilde{x}_2(k-1) + (1-10T_{\mathrm{s}})\tilde{x}_3(k-1)\\ \quad +0.0024121T_{\mathrm{s}}\tilde{\alpha}(k-1)\\ \tilde{x}_4(k-1)(1-10T_{\mathrm{s}}) + 245.978T_{\mathrm{s}}\tilde{\alpha}(k-1) \end{bmatrix}, \end{aligned}$$

where $\tilde{\alpha}(k-1) = \tilde{x}_1(k-1)\sqrt{\tilde{x}_2(k-1)}$ whereas the output disturbance is estimated from Eqn. (15), which gives

$$\begin{aligned} d(k) = y(k) - \Big(&(1-10T_{\mathrm{s}})\tilde{x}_4(k-1) \qquad (44)\\ &+ 245.978T_{\mathrm{s}}\tilde{\alpha}(k-1) + \nu_4(k)\Big)\\ &\times \Big(0.112191T_{\mathrm{s}}\tilde{x}_2(k-1)\\ &+ (1-10T_{\mathrm{s}})\tilde{x}_3(k-1)\\ &+ 0.0024121T_{\mathrm{s}}\tilde{\alpha}(k-1) + \nu_3(k)\Big)^{-1}. \end{aligned}$$

The nonlinear state free trajectory may be found from Eqns. (10) and (12) using the state equations (41) and remembering that $u(k+p|k) = u(k-1)$ for $p = 0,\ldots,N_{\mathrm{u}}-1$. For the first sampling instant of the prediction horizon one has

$$\begin{aligned} \hat{x}_1^0(k+1|k) = (1-10T_{\mathrm{s}})\tilde{x}_1(k) &+ 60T_{\mathrm{s}}\\ &- 2.4568T_{\mathrm{s}}\tilde{\alpha}(k) + \nu_1(k), \end{aligned}$$

$$\begin{aligned}\hat{x}_2^0(k+1|k) &= (1-10.1022T_{\mathrm{s}})\tilde{x}_2(k)\\ &\quad + 80T_{\mathrm{s}}u(k-1)+\nu_2(k),\\ \hat{x}_3^0(k+1|k) &= 0.112191T_{\mathrm{s}}\tilde{x}_2(k)+(1-10T_{\mathrm{s}})\tilde{x}_3(k)\\ &\quad + 0.0024121T_{\mathrm{s}}\tilde{\alpha}(k)+\nu_3(k),\\ \hat{x}_4^0(k+1|k) &= (1-10T_{\mathrm{s}})\tilde{x}_4(k)\\ &\quad + 245.978T_{\mathrm{s}}\tilde{\alpha}(k)+\nu_4(k),\end{aligned}$$

where $\tilde{\alpha}(k)=\tilde{x}_1(k)\sqrt{\tilde{x}_2(k)}$, whereas for $p=2,\dots,N$ one obtains

$$\begin{aligned}x_1^0(k+p|k) &= (1-10T_{\mathrm{s}})x_1^0(k+p-1|k)+60T_{\mathrm{s}}\\ &\quad - 2.4568T_{\mathrm{s}}\alpha^0(k+p-1|k)+\nu_1(k),\\ x_2^0(k+p|k) &= (1-10.1022T_{\mathrm{s}})x_2^0(k+p-1|k)\\ &\quad + 80T_{\mathrm{s}}u(k+p-1)+\nu_2(k),\\ x_3^0(k+p|k) &= 0.112191T_{\mathrm{s}}x_2^0(k+p-1|k)\\ &\quad + (1-10T_{\mathrm{s}})x_3^0(k+p-1|k)\\ &\quad + 0.0024121T_{\mathrm{s}}\alpha^0(k+p-1|k)+\nu_3(k),\\ x_4^0(k+p|k) &= (1-10T_{\mathrm{s}})x_4^0(k+p-1|k)\\ &\quad + 245.978T_{\mathrm{s}}\alpha^0(k+p-1|k)+\nu_4(k),\end{aligned}$$

where

$$\alpha^0(k+p-1|k)=x_1^0(k+p-1|k)\sqrt{x_2^0(k+p-1|k)}.$$

The output free trajectory is calculated for the state free one, from Eqn. (17) and the output equation (41), which gives

$$\hat{y}^0(k+p|k)=\frac{x_4^0(k+p|k)+\nu_4(k)}{x_3^0(k+p|k)+\nu_3(k)}+d(k),$$

for $p=1,\dots,N$.

4.4. Implementation of the MPC-NPLPT algorithm for the benchmark process.

The state and output disturbances are estimated in the same way it is done in the MPC-NPL algorithm, from Eqns. (43) and (44), respectively. The predicted nonlinear state trajectory for the internal iteration t is found from Eqns. (12) and (10) using the state equations (41). For the first sampling instant of the prediction horizon one has

$$\begin{aligned}\hat{x}_1^t(k+1|k) &= (1-10T_{\mathrm{s}})\tilde{x}_1(k)+60T_{\mathrm{s}}\\ &\quad - 2.4568T_{\mathrm{s}}\tilde{\alpha}(k)+\nu_1(k),\\ \hat{x}_2^t(k+1|k) &= (1-10.1022T_{\mathrm{s}})\tilde{x}_2(k)\\ &\quad + 80T_{\mathrm{s}}u^t(k|k)+\nu_2(k),\\ \hat{x}_3^t(k+1|k) &= 0.112191T_{\mathrm{s}}\tilde{x}_2(k)\\ &\quad + (1-10T_{\mathrm{s}})\tilde{x}_3(k)\\ &\quad + 0.0024121T_{\mathrm{s}}\tilde{\alpha}(k)+\nu_3(k),\\ \hat{x}_4^t(k+1|k) &= (1-10T_{\mathrm{s}})\tilde{x}_4(k)\\ &\quad + 245.978T_{\mathrm{s}}\tilde{\alpha}(k)+\nu_4(k),\end{aligned}$$

where $\tilde{\alpha}(k)=\tilde{x}_1(k)\sqrt{\tilde{x}_2(k)}$, whereas for $p=2,\dots,N$ one obtains

$$\begin{aligned}\hat{x}_1^t(k+p|k) &= (1-10T_{\mathrm{s}})\hat{x}_1^t(k+p-1|k)+60T_{\mathrm{s}}\\ &\quad - 2.4568T_{\mathrm{s}}\alpha^t(k+p-1|k)+\nu_1(k),\\ \hat{x}_2^t(k+p|k) &= (1-10.1022T_{\mathrm{s}})\hat{x}_2^t(k+p-1|k)\\ &\quad + 80T_{\mathrm{s}}u^t(k+p-1)+\nu_2(k),\\ \hat{x}_3^t(k+p|k) &= 0.112191T_{\mathrm{s}}\hat{x}_2^t(k+p-1|k)\\ &\quad + (1-10T_{\mathrm{s}})\hat{x}_3^t(k+p-1|k)\\ &\quad + 0.0024121T_{\mathrm{s}}\alpha^t(k+p-1|k)+\nu_3(k),\\ \hat{x}_4^t(k+p|k) &= (1-10T_{\mathrm{s}})\hat{x}_4^t(k+p-1|k)\\ &\quad + 245.978T_{\mathrm{s}}\alpha^t(k+p-1|k)+\nu_4(k),\end{aligned}$$

where

$$\alpha^t(k+p-1|k)=\hat{x}_1^t(k+p-1|k)\sqrt{\hat{x}_2^t(k+p-1|k)}.$$

The predicted nonlinear output trajectory is calculated from Eqn. (17) and the output equation (41), which gives

$$\hat{y}^t(k+p|k)=\frac{\hat{x}_4^t(k+p|k)+\nu_4(k)}{\hat{x}_3^t(k+p|k)+\nu_3(k)}+d(k),$$

for $p=1,\dots,N$. The linear approximation of the predicted trajectory $\hat{\boldsymbol{y}}^t(k)$ along the trajectory $\boldsymbol{u}^{t-1}(k)$ is calculated from the general Eqns. (35) and (36). For the benchmark model (41), one obtains

$$\begin{aligned}\frac{\partial\hat{y}(k+p|k)}{\partial u^{t-1}(k+r|k)} &= \left(\frac{\partial\hat{x}_4^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}x_3^{t-1}(k+p|k)\right.\\ &\quad \left. - x_4^{t-1}(k+p|k)\frac{\partial\hat{x}_3^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)}\right)\\ &\quad \times\frac{1}{(\hat{x}_3^{t-1}(k+p|k))^2},\end{aligned}$$

for $p=1,\dots,N$ and $r=0,\dots,N_{\mathrm{u}}-1$, where

$$\begin{aligned}\frac{\partial\hat{x}_n^{t-1}(k+1|k)}{\partial u^{t-1}(k+r|k)} &= 0,\quad n=1,3,4,\\ \frac{\partial\hat{x}_2^{t-1}(k+1|k)}{\partial u^{t-1}(k+r|k)} &= 80T_{\mathrm{s}}\frac{\partial u^{t-1}(k|k)}{\partial u^{t-1}(k+r|k)},\end{aligned}$$

and

$$\begin{aligned}\frac{\partial\hat{x}_1^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)} &= (1-10T_{\mathrm{s}})\frac{\partial\hat{x}_1^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)}\\ &\quad - 2.4568T_{\mathrm{s}}\frac{\partial\alpha^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)},\end{aligned}$$

$$\frac{\partial \hat{x}_2^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)} = (1-10.1022T_{\rm s})\frac{\partial \hat{x}_2^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)} + 80T_{\rm s}\frac{\partial u^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)},$$

$$\frac{\partial \hat{x}_3^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)} = 0.112191T_{\rm s}\frac{\partial \hat{x}_2^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)} (1-10T_{\rm s})\frac{\partial \hat{x}_3^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)} + 0.0024121T_{\rm s}\frac{\partial \alpha^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)},$$

$$\frac{\partial \hat{x}_4^{t-1}(k+p|k)}{\partial u^{t-1}(k+r|k)} = (1-10T_{\rm s})\frac{\partial \hat{x}_4^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)} + 245.978T_{\rm s}\frac{\partial \alpha^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)},$$

for $p = 2, \ldots, N$, where

$$\frac{\partial \alpha^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)} = \frac{\partial \hat{x}_1^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)}\sqrt{\hat{x}_2^{t-1}(k+p-1|k)} + \frac{\hat{x}_1^{t-1}(k+p-1|k)}{2\sqrt{\hat{x}_2^{t-1}(k+p-1|k)}}\frac{\partial \hat{x}_2^{t-1}(k+p-1|k)}{\partial u^{t-1}(k+r|k)},$$

and

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

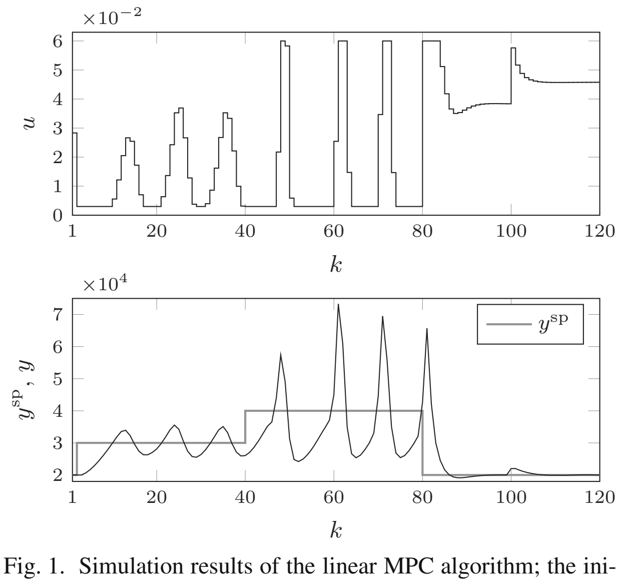

Fig. 1. Simulation results of the linear MPC algorithm; the initial condition of the process is known.

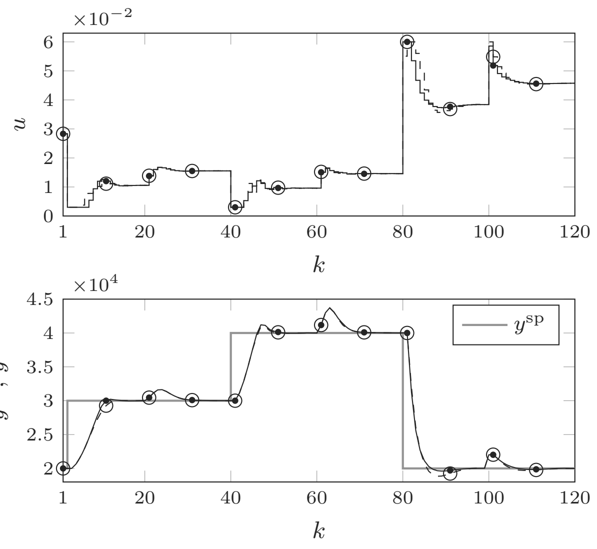

Fig. 2. Simulation results: the MPC-NPL algorithm with on-line model linearisation and quadratic optimisation (*dashed line with circles*), the MPC-NO algorithm with on-line nonlinear optimisation (*solid line with dots*); the initial condition of the process is known.

4.5. Comparison of MPC algorithms for the benchmark process. The set-point trajectory consists of three steps: at the instant $k = 2$ it changes from the nominal operating point ($y = 20000$) to $y^{\rm sp} = 30000$, at the instant $k = 40$ it changes to $y^{\rm sp} = 40000$, and at the instant $k = 80$ it changes to $y^{\rm sp} = 20000$. The unmeasured input step disturbance at the instant $k = 20$ changes from 0 to -0.005, at the instant $k = 60$ it changes from -0.005 to -0.01; the unmeasured output step disturbance at the instant $k = 100$ changes from 0 to 2000.

4.5.1. Case I: Initial condition known, no noise. First it is assumed that the initial condition of the process is known; i.e., the initial conditions of the process and of the filter are the same, and the process is not affected by measurement noise. Simulation results of the linear MPC algorithm are shown in Fig. 1. Unfortunately, for the first two set-point changes, the steady-state is not reached. Next, Fig. 2 compares simulation results of the MPC-NPL algorithm with on-line model linearisation and quadratic optimisation to those of the "ideal" MPC-NO algorithm with on-line nonlinear optimisation. The trajectories of the MPC-NPL algorithm are a little bit slower, but it works well: it follows the set-point changes, and quickly compensates for input and output disturbances, and the steady-state is always reached. Next, Fig. 3 compares simulation results of the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation to those of the MPC-NO algorithm. On-line predicted trajectory linearisation leads to the same control accuracy as in the MPC-NO algorithm. Figure 4 depicts real and

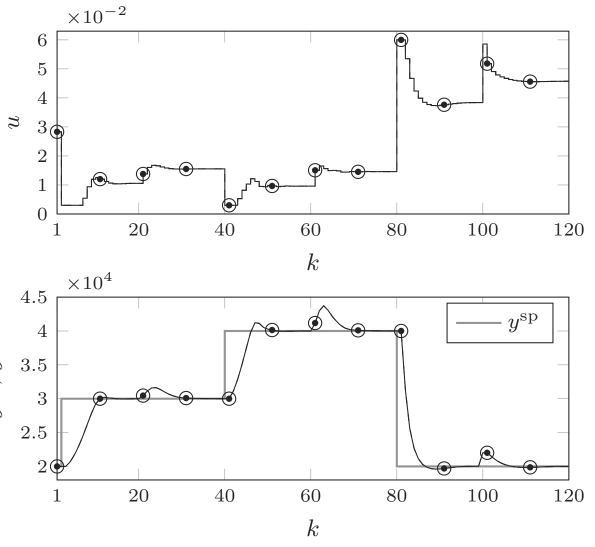

Fig. 3. Simulation results: the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation (*dashed line with circles*), the MPC-NO algorithm with on-line nonlinear optimisation (*solid line with dots*); the initial condition of the process is known.

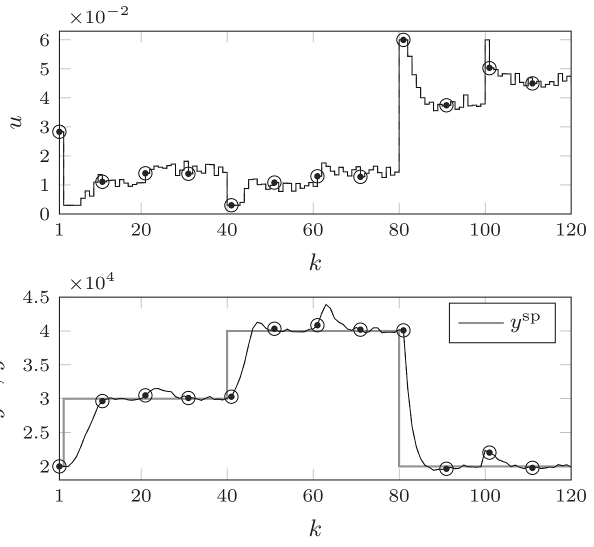

Fig. 5. Simulation results: the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation (*dashed line with circles*), the MPC-NO algorithm with on-line nonlinear optimisation (*solid line with dots*); the initial condition of the process is not known and the output of the process is additionally affected by measurement noise.

estimated state trajectories in the MPC-NPLPT scheme. At the beginning, due to a perfect initial condition of the filter, they are the same, but as the first disturbance appears ($k = 20$), they start to be different. Despite these discrepancies and disturbances, the discussed MPC algorithms provide offset-free control. Table 1 compares the nonlinear MPC algorithms in terms of the sum of squared control errors.

4.5.2. Case II: Initial condition not known, additional output noise. Next, it is assumed that the initial condition of the process is not known, the initial state of the filter is $\tilde{x} = [4\ 0.3\ 0.001\ 40]^{\mathrm{T}}$, and the process output is additionally affected by noise with normal distribution with zero mean and standard deviation 250.

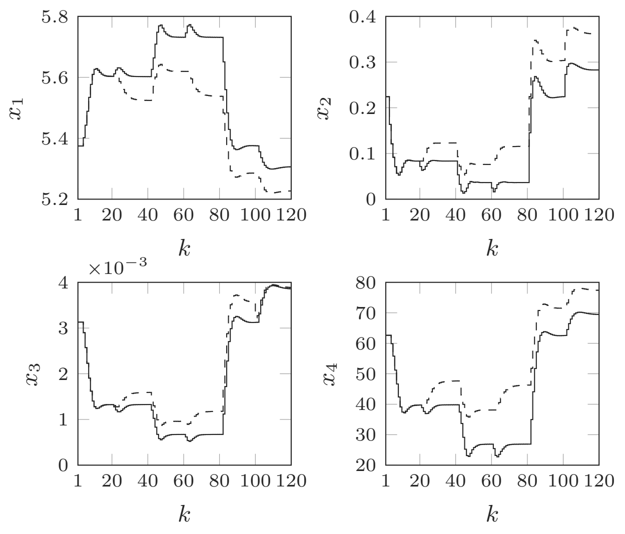

Fig. 4. Real state trajectories (*solid line*) and the estimated state trajectories (*dashed line*) in the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation; the initial condition of the process is known.

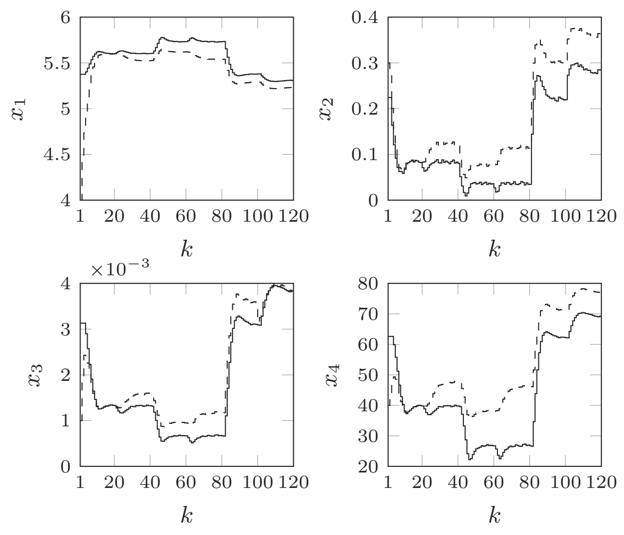

Fig. 6. Real state trajectories (*solid line*) and the estimated state trajectories (*dashed line*) in the MPC-NPLPT algorithm with on-line trajectory linearisation and quadratic optimisation; the initial condition of the process is not known and the output of the process is additionally affected by measurement noise.

Table 1. Comparison of summmarised squared control errors of the nonlinear MPC algorithms.

Algorithm	Case I	Case II
MPC-NPL	1.8827×10^9	2.0045×10^9
MPC-NPLPT	1.8512×10^9	1.8666×10^9
MPC-NO	1.8512×10^9	1.8666×10^9

Figure 5 compares trajectories of MPC-NPLPT and MPC-NO algorithms, Figure 6 depicts real and estimated state trajectories. Also in this case, despite noise, the MPC-NPLPT algorithm gives control accuracy the same as the MPC-NO scheme. All nonlinear MPC algorithms provide offset-free control.

Computational time of the MPC algorithms is as follows: MPC-NPL algorithm—2.33 s, MPC-NPLPT—2.99 s, MPC-NO—14.47 s (Intel Core i5-2540M processor).

5. Conclusions

This work discusses two computationally efficient nonlinear MPC algorithms for dynamic systems described by state-space models and with state estimation. The model or the predicted trajectory is successively linearised on-line, which leads to quadratic optimisation. The presented algorithms are straightforward in implementation because the disturbance handling mechanism used by Tatjewski (2014) in the case of linear systems is adopted. The process state itself is estimated, the disturbance state observer is not used, but the algorithms compensate for deterministic constant-type disturbances. Implementation and efficiency of the algorithms are demonstrated for a polymerisation reactor benchmark for which the MPC algorithm with trajectory linearisation gives the same performance as the MPC scheme with nonlinear optimisation.

References

Arahal, M., Berenguel, M. and Camacho, E. (1998). Neural identification applied to predictive control of a solar plant, *Control Engineering Practice* **6**(3): 333–344.

Bismor, D. (2015). Extension of LMS stability condition over a wide set of signals, *International Journal of Adaptive Control and Signal Processing* **29**(5): 653–670, DOI: 10.1002/acs.2500.

Camacho, E. and Bordons, C. (1999). *Model Predictive Control*, Springer, London.

Colin, G., Chamaillard, Y., Bloch, G. and Corde, G. (2007). Neural control of fast nonlinear systems application to a turbocharged SI engine with VCT, *IEEE Transactions on Neural Networks* **18**(4): 1101–1114.

de Oliveira, N. and Biegler, L. (1995). An extension of Newton-type algorithms for nonlinear process control, *Automatica* **52**(2): 281–286.

Deng, J., Becerra, V.M. and Stobart, R. (2009). Input constraints handling in an MPC/feedback linearization scheme, *International Journal of Applied Mathematics and Computer Science* **19**(2): 219–232, DOI: 10.2478/v10006-009-0018-2.

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

Ellis, M., Durand, H. and Christofides, P. (2014). A tutorial review of economic model predictive control methods, *Journal of Process Control* **24**(8): 1156–1178.

Gonzalez, A., Adam, E. and Marchetti, J. (2008). Conditions for offset elimination in state space receding horizon controllers: A tutorial analysis, *Chemical Engineering and Processing* **47**(12): 2184–2194.

Kuure-Kinsey, M., Cutright, R. and Bequette, B. (2006). Computationally efficient neural predictive control based on a feedforward architecture, *Industrial and Engineering Chemistry Research* **45**(25): 8575–8582.

Ławryńczuk, M. (2007). A family of model predictive control algorithms with artificial neural networks, *International Journal of Applied Mathematics and Computer Science* **17**(2): 217–232, DOI: 10.2478/v10006-007-0020-5.

Ławryńczuk, M. (2014). *Computationally Efficient Model Predictive Control Algorithms: A Neural Network Approach*, Studies in Systems, Decision and Control, Vol. 3, Springer, Heidelberg.

Lee, J. and Ricker, N. (1994). Extended Kalman filter based nonlinear model predictive control, *Industrial and Engineering Chemistry Research* **33**(6): 1530–1541.

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

Maeder, U. and Morari, M. (2010). Offset-free reference tracking with model predictive control, *Automatica* **46**(9): 1469–1476.

Mayne, D. (2014). Model predictive control: Recent developments and future promise, *Automatica* **50**(12): 2967–2986.

Megías, D., Serrano, J. and Ghoumari, M.E. (1999). Extended linearised predictive control: Practical control algorithms for non-linear systems, *Proceedings of the European Control Conference, ECC 1999, Karlsruhe, Germany*, F883.

Mu, J., Rees, D. and Liu, G. (2005). Advanced controller design for aircraft gas turbine engines, *Journal of Process Control* **13**(8): 1001–1015.

Muske, K. and Badgwell, T. (2002). Disturbance modeling for offset-free linear model predictive control, *Journal of Process Control* **12**(5): 617–632.

Pannocchia, G. and Bemporad, A. (2007). Combined design of disturbance model and observer for offset-free model predictive control, *IEEE Transactions on Automatic Control* **52**(6): 1048–1053.

Pannocchia, G. and Rawlings, J. (2003). Disturbance models for offset-free model predictive control, *AIChE Journal* **49**(2): 426–437.

Patan, K. and Korbicz, J. (2012). Nonlinear model predictive control of a boiler unit: A fault tolerant control study, *International Journal of Applied Mathematics and Computer Science* **22**(1): 225–237, DOI: 10.2478/v10006-012-0017-6.

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

Rawlings, J. and Mayne, D. (2009). *Model Predictive Control: Theory and Design*, Nob Hill Publishing, Madison.

Simon, D. (2006). *Optimal State Estimation: Kalman,* H_∞ *and Nonlinear Approaches*, John Wiley & Sons, Hoboken, NJ.

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

Tatjewski, P. (2010). Supervisory predictive control and on-line set-point optimization, *International Journal of Applied Mathematics and Computer Science* **20**(3): 483–495, DOI: 10.2478/v10006-010-0035-1.

Tatjewski, P. (2014). Disturbance modeling and state estimation for offset-free predictive control with state-space models, *International Journal of Applied Mathematics and Computer Science* **24**(2): 313–323, DOI: 10.2478/amcs-2014-0023.

Tatjewski, P. and Ławryńczuk, M. (2006). Soft computing in model-based predictive control, *International Journal of Applied Mathematics and Computer Science* **16**(1): 7–26.

Wang, L. (2007). *Model Predictive Control System Design and Implementation Using MATLAB*, Springer, London.

Maciej Ławryńczuk was born in Warsaw, Poland, in 1972. He received his M.Sc. in 1998, his Ph.D. in 2003, and his D.Sc. in 2013, all in automatic control from the Warsaw University of Technology, Faculty of Electronics and Information Technology. Since 2003 he has been employed at the same university at the Institute of Control and Computation Engineering: in 2003–2004 as a teaching assistant, in 2004–2015 as an assistant professor and since 2015 as an associate professor. He is the author or a co-author of 6 books and more than 100 other publications, including 30 journal articles. His research interests include advanced control algorithms, in particular MPC algorithms, set-point optimisation algorithms, and soft computing methods, especially neural networks, modelling and simulation.

Appendix

To prove that the MPC-NPL algorithm with state estimation guarantees offset-free control, the reasoning is similar to that used for the linear MPC algorithm with the discussed state and output disturbance estimations (Tatjewski, 2014). It is assumed that the set-point is (asymptotically) constant, $y^{\mathrm{sp}}(k) = y^{\mathrm{sp}}$, and feasible at steady-state; all disturbances are constant and the process is asymptotically stable. In steady state, the process state and its estimate stabilise at x^{ss} and $\tilde{x}^{\mathrm{ss}}$, respectively; the solution to the MPC optimisation problem (18) is $\triangle\boldsymbol{u}(k) = \mathbf{0}_{n_{\mathrm{u}}N_{\mathrm{u}},1}$, the cost-function has its lowest possible value 0, and the input of the process stabilises at u^{ss}. From Eqn. (26), it follows that $y^{\mathrm{sp}} = y^0(k+p|k)$ for $p = 1,\dots,N$. From Eqns. (24) and (11), one has

$$\begin{aligned} x^0(k+1|k) &= f(\tilde{x}^{\mathrm{ss}}, u^{\mathrm{ss}}) + \nu(k) \\ &= f(\tilde{x}^{\mathrm{ss}}, u^{\mathrm{ss}}) + \tilde{x}^{\mathrm{ss}} - f(\tilde{x}^{\mathrm{ss}}, u^{\mathrm{ss}}) = \tilde{x}^{\mathrm{ss}}. \end{aligned}$$

Analogously, from Eqns. (25) and (11), for $p = 2,\dots,N$ one obtains

$$\begin{aligned} x^0(k+p|k) &= f(x^0(k+p-1|k), u^{\mathrm{ss}}) + \nu(k) \\ &= f(x^{\mathrm{ss}}, u^{\mathrm{ss}}) + \tilde{x}^{\mathrm{ss}} - f(\tilde{x}^{\mathrm{ss}}, u^{\mathrm{ss}}) = \tilde{x}^{\mathrm{ss}}. \end{aligned}$$

From Eqns. (27) and (15), one has

$$\begin{aligned} y^{\mathrm{sp}} &= y^0(k+p|k) = g(x^0(k+p)) + d(k) \\ &= g(\tilde{x}^{\mathrm{ss}}) + y^{\mathrm{ss}} - g(\tilde{x}^{\mathrm{ss}}) = y^{\mathrm{ss}}, \end{aligned}$$

which means that at steady-state the offset-free stabilisation is guaranteed ($y^{\mathrm{sp}} = y^{\mathrm{ss}}$).

For the MPC-NPLPT algorithm, from Eqn. (33) it follows that $\boldsymbol{u}^t(k) = \boldsymbol{u}^{t-1}(k)$, and only one (initial) internal iteration is necessary, which results in state and output free trajectories. They are calculated in the same way it is done in the MPC-NPL scheme, which means that at steady-state the offset-free stabilisation is guaranteed.

Received: 10 July 2014
Revised: 24 January 2015