Nonlinear Dyn (2019) 95:1639–1660
https://doi.org/10.1007/s11071-018-4650-y

ORIGINAL PAPER

Wiener structures for modeling and nonlinear predictive control of proton exchange membrane fuel cell

Maciej Ławryńczuk · **Dirk Söffker**

Received: 17 May 2018 / Accepted: 3 November 2018 / Published online: 14 November 2018

Abstract The proton exchange membrane (PEM) fuel cell is a nonlinear dynamic system which cannot be precisely described and controlled using a linear model. This work has two objectives: (a) it discusses model selection for the PEM and (b) it develops two nonlinear computationally efficient model predictive control (MPC) algorithms for the PEM. Three Wiener model types of different orders of dynamics and complexity of the nonlinear steady-state block are compared. The model consisting of three dynamic blocks and a neural network with five hidden nodes is chosen. To obtain simple MPC quadratic optimization problems, a linear approximation of the model or a linear approximation of the predicted trajectory is repeatedly found. The first MPC scheme gives very good control accuracy, whereas the second MPC scheme leads to the same trajectories as those possible in the "ideal" MPC scheme with full online nonlinear optimization.

Financial support received by the first author from the German Academic Exchange Service (DAAD) is acknowledged.

M. Ławryńczuk (✉)
Institute of Control and Computation Engineering, Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland
e-mail: M.Lawrynczuk@ia.pw.edu.pl

D. Söffker
Chair of Dynamics and Control, Engineering Faculty, University of Duisburg-Essen, Duisburg, Germany
e-mail: soeffker@uni-due.de

Keywords Proton exchange membrane fuel cell · Model predictive control · Wiener models · Neural networks

1 Introduction

Currently, the transport sector relies on the combustion engine which uses fossil fuels. Alas, it results in emission of greenhouse gases, which leads to serious environmental problems, i.e., air pollution, global warming, climate changes and destruction of the ozone layer. Zero-emission electric vehicles are available, and their popularity grows. The majority of electric vehicles use batteries for energy storage. An interesting alternative is to use a fuel cell for energy generation. Fuel cells are electrochemical devices that convert chemical energy stored in hydrocarbon fuels (usually hydrogen) directly into electrical energy [1]. They have many important advantages: high electrical efficiency, very low emission and quiet operation. Moreover, since fuel cells do not have moving parts, their life cycle is very long. Fuel cells may be produced in different scales: from microwatts to megawatts, which makes them useful in numerous applications. Lastly, hydrogen necessary for fuel cells may be easily produced from different sources (i.e., biomass, coal or natural gas) so dependence on imported oil may be significantly reduced.

Among existing types of fuel cells [1], the proton exchange membrane (PEM) fuel cells are preferred not only for mobile and vehicle applications, includ-

ing cars, scooters, bicycles, boats and underwater vessels [2,3], but also for stationary ones. This is because of low operation temperature (usually between 60 and 80 °C) which gives a fast start-up, simple and compact design as well as reliable operation. Since solid electrolyte is used, no electrolyte leakage is possible. The PEM fuel cells are considered to be very promising power sources, and they are expected to become sound alternatives to conventional power generation methods.

It is necessary to point out that control of PEM fuel cells is a challenging task. Although there are examples of classical control methods applied to the PEM fuel cell, e.g., a linear state feedback controller [4] or a sliding mode controller (SMC) [5], the process is inherently nonlinear and the linear controllers may give control accuracy below expectations. Hence, a number of nonlinear control strategies have been applied to the PEM fuel cell process: an adaptive proportional–integral–derivative (PID) algorithm whose parameters are tuned online by a fuzzy logic system [6,7] or by a neural network [8], an adaptive PID algorithm with a fuzzy logic feedforward compensator [9], a nonlinear state feedback controller [10], a fuzzy controller [11] and a lookup table [12]. Fractional complex-order controllers may be also used [13,14]. Recently, model predictive control (MPC) algorithms [15–17] have been applied for the PEM fuel cell. In the literature, it is possible to find two categories of MPC:

1. The fully fledged nonlinear MPC (for prediction a nonlinear model is used) which requires solving a nonlinear optimization problem at each sampling instant online [18–21]. Such an approach may be very computationally demanding, and its practical application may be impossible.
2. The classical MPC algorithm based on a fixed (parameter constant) linear model [22,23]. In this approach, online calculations in MPC are not demanding (quadratic optimization is used), but the resulting control quality may be not satisfactory because the process is nonlinear and the linear model used in MPC is only a very rough approximation of the process.

The contribution of this work is twofold:

1. Two nonlinear MPC algorithms for the PEM fuel cell are described. In contrast to the algorithms presented in [18–21], in both algorithms computationally simple quadratic optimization is used, and full nonlinear optimization is not necessary. It is possible because a linear approximation of the model or a linear approximation of the predicted trajectory is found online. The discussed algorithms are compared to the MPC scheme with nonlinear optimization in terms of control accuracy and computational time, and inefficiency of the linear MPC scheme is shown.
2. Both described MPC algorithms use Wiener models of the PEM process, and effectiveness of three structures is compared. The choice of the Wiener model, composed of a linear dynamic block connected in series with a nonlinear steady-state one [24], is motivated by two factors. Firstly, the Wiener model is a natural representation of the PEM process. Secondly, the Wiener model is used in a simple way in the described MPC algorithms; the first of them is motivated by the serial structure of the Wiener model.

A neural network [25] is used in the steady-state part of the Wiener model. Neural networks are very efficient for modeling of dynamic systems, e.g., [26], and control. There are many different control methods based on neural networks: e.g., adaptive PID control [8], adaptive state control [27], adaptive sliding mode control [28], robust feedback control [29], the zeroing dynamics method [30,31] and MPC [26]. Neural networks may be also used for online matrix calculations [32] and optimization [33].

This paper is structured as follows. Section 2 describes the PEM process and its fundamental model. Next, in Sect. 3 three Wiener structures are characterized. Section 4 details two nonlinear MPC algorithms for the PEM process. Section 5 discusses identification of different types of models and MPC of the PEM process. Section 6 concludes the paper.

2 Proton exchange membrane fuel cell

In the general case, the model of the PEM fuel cell is quite complicated [4]. Hence, the tendency is to use simpler model for development of a control system [34–39].

2.1 PEM fuel cell system description

In this work, the PEM fuel cell model introduced in [39] and further discussed in [40–42] is considered. The

PEM process has one manipulated variable (the input of the process) which is the input methanol flow rate q (mol s^{-1}), one disturbance (the uncontrolled input) I which is the external current load (A) and one controlled variable (the output of the process) which is the stack output voltage V (V). The partial pressures of hydrogen, oxygen and water are denoted by p_{H_2}, p_{O_2} and p_{H_2O}, respectively (atm). The input hydrogen flow, the hydrogen reacted flow and the oxygen input flow are denoted by $q^{in}_{H_2}$, $q^{r}_{H_2}$ and $q^{in}_{O_2}$, respectively (mol s^{-1}).

2.2 PEM fuel cell continuous-time fundamental model

The fundamental continuous-time model of the PEM system is defined by the following continuous-time equations. The pressure of hydrogen is

$$p_{H_2} = \frac{1/K_{H_2}}{\tau_{H_2}s + 1}\left(q^{in}_{H_2} - 2K_rI_r\right), \tag{1}$$

where K_{H_2} and τ_{H_2} denote the valve molar constant for hydrogen and the response time of hydrogen flow, respectively. The input hydrogen flow obtained from the reformer is

$$q^{in}_{H_2} = \frac{CV}{(\tau_1 s + 1)(\tau_2 s + 1)}q, \tag{2}$$

where q is the methane flow rate, CV, τ_1 and τ_2 are constants. Hence, from Eqs. (1) and (2), the pressure of hydrogen is

$$p_{H_2} = \frac{1/K_{H_2}}{\tau_{H_2}s + 1}\left(\frac{CV}{(\tau_1 s + 1)(\tau_2 s + 1)}q - 2K_r\right). \tag{3}$$

The pressure of oxygen is

$$p_{O_2} = \frac{1/K_{O_2}}{\tau_{O_2}s + 1}\left(q^{in}_{O_2} - K_rI_r\right), \tag{4}$$

where K_{O_2} and τ_{O_2} denote the valve molar constant for oxygen and the response time of oxygen flow, respectively. The input flow rate of oxygen is

$$q^{in}_{O_2} = (1/\tau_{H-O})q^{in}_{H_2}, \tag{5}$$

where τ_{H-O} is the ration of hydrogen to oxygen. Using Eq. (2), the pressure of oxygen is

$$p_{O_2} = \frac{1/K_{O_2}}{\tau_{O_2}s + 1}\left(\frac{CV/\tau_{H-O}}{(\tau_1 s + 1)(\tau_2 s + 1)}q - K_rI_r\right). \tag{6}$$

The pressure of water is

$$p_{H_2O} = \frac{1/K_{H_2O}}{\tau_{H_2O}s + 1}q^{r}_{H_2}, \tag{7}$$

where K_{H_2O} and τ_{H_2O} denote the valve molar constant for water and the response time of water flow, respectively. The hydrogen flow that reacts is

$$q^{r}_{H_2} = 2K_rI_r. \tag{8}$$

Hence, from Eqs. (7) and (8), the pressure of water is

$$p_{H_2O} = \frac{2K_r/K_{H_2O}}{\tau_{H_2O}s + 1}I. \tag{9}$$

Finally, the stack output voltage is

$$V = E - \eta_{act} - \eta_{ohmic}. \tag{10}$$

From the Nernst's equation

$$E = N_0\left[E_0 + \frac{RT}{2F}\ln\frac{p_{H_2}\sqrt{p_{O_2}}}{p_{H_2O}}\right], \tag{11}$$

where N_0, E_0, R_0, T_0 and F_0 denote the number of cells in series in the stack, the ideal standard potential, the universal gas constant, the absolute temperature and the Faraday's constant, respectively. The activation loses are defined by

$$\eta_{act} = B\log(CI), \tag{12}$$

where B and C are constants. The ohmic losses are

$$\eta_{ohmic} = R^{int}I, \tag{13}$$

where R^{int} is the internal resistance.

The continuous-time fundamental model consists of Eqs. (2), (3), (6), (8)–(13). The values of parameters are given in Table 1. Table 2 gives the values of process

Table 1 Parameters of the fundamental continuous-time model of the PEM system

Parameter	Value	Unit	Description
B	0.04777	$1\ \mathrm{A}^{-1}$	Activation voltage constant
C	0.0136	V	Activation voltage constant
CV	2	–	Conversion factor
E_0	0.6	V	No-load voltage
F	96485	$\mathrm{C\ mol}^{-1}$	Faraday's constant
$K_{\mathrm{H_2}}$	4.22×10^{-3}	$\mathrm{mol\ s}^{-1}\ \mathrm{atm}^{-1}$	Hydrogen valve constant
$K_{\mathrm{H_2O}}$	7.716×10^{-3}	$\mathrm{mol\ s}^{-1}\ \mathrm{atm}^{-1}$	Water time constant
$K_\mathrm{r} = N_0/4F_0$	2.2802×10^{-3}	$\mathrm{mol\ s}^{-1}\ \mathrm{A}^{-1}$	Constant
$K_{\mathrm{O_2}}$	2.11×10^{-2}	$\mathrm{mol\ s}^{-1}\ \mathrm{atm}^{-1}$	Oxygen time constant
N_0	88	–	Number of cells
R_0	8.314	$\mathrm{J\ mol}^{-1}\ \mathrm{K}^{-1}$	Universal gas constant
R^{int}	0.00303	Ω	Internal resistance
T_0	343	K	Absolute temperature
$\tau_1 = \tau_1$	2	s	Reformer time constants
$\tau_{\mathrm{H_2}}$	3.37	s	Hydrogen time constant
$\tau_{\mathrm{H-O}}$	1.168	–	Hydrogen–oxygen flow ratio
$\tau_{\mathrm{H_2O}}$	18.418	s	Water time constant
$\tau_{\mathrm{O_2}}$	6.74	s	Oxygen time constant

Table 2 Values of process variables for the initial operating point

Variable	Value	Unit
I	100	A
q	0.5	$\mathrm{mol\ s}^{-1}$
$p_{\mathrm{H_2}}$	5.9102	atm
$p_{\mathrm{O_2}}$	39.4959	atm
$p_{\mathrm{H_2O}}$	22.6160	atm
V	56.6179	V

variables for the initial operating point. The values of process input and disturbance signals are constrained

$$0.1\ \mathrm{mol\ s}^{-1} \leq q \leq 2\ \mathrm{mol\ s}^{-1}, \tag{14}$$

$$50\ \mathrm{A} \leq I \leq 150\ \mathrm{A}. \tag{15}$$

3 Wiener models of PEM fuel cell

It can be noted that the continuous-time fundamental model of the discussed PEM fuel cell is characterized by linear dynamic transfer functions (Eqs. 3, 6, 9), but the stack voltage is defined by the nonlinear steady-state Nernst's equation (11). This means that the outputs of the linear dynamic part of the model are inputs of the nonlinear steady-state one. Hence, it is straightforward to use the Wiener structure as an empirical model of the considered PEM fuel cell.

For model identification, the manipulated variable of the process, q, the disturbance, I, and the output, V, are scaled

$$u = q - \bar{q},\ h = 0.01(I - \bar{I}),\ y = V - \overline{V}, \tag{16}$$

where $\bar{q}$, $\bar{I}$ and $\overline{V}$ denote values of process variables at the initial operating point (Table 2).

3.1 Wiener model: structure A

Figure 1 depicts the first structure of the Wiener model (structure A). It consists of a linear dynamic block connected in series with a nonlinear steady-state one. The linear block has two inputs (u, h) and one output, v, which is an auxiliary variable. The linear block is characterized by the equation

$$v(k) = \sum_{i=1}^{n_\mathrm{B}^1} b_i^1 u(k-i) + \sum_{i=0}^{n_\mathrm{B}^2} b_i^2 h(k-i) - \sum_{i=1}^{n_\mathrm{A}} a_i v(k-i). \tag{17}$$

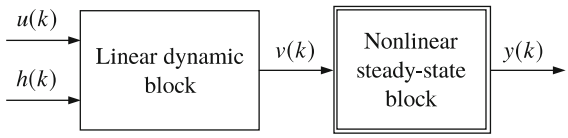

Fig. 1 Structure A of the Wiener model

The integers n_A and n_B^j, $j = 1, 2$ define the order of the model dynamics. The constant parameters of the linear dynamic blocks are denoted by the real numbers a_i $(i = 1, \ldots, n_A)$, b_i^1 $(i = 1, \ldots, n_B^1)$ and b_i^2 $(i = 0, \ldots, n_B^1)$. It is important to note that the signal $v(k)$ depends directly on the signal $h(k)$ since the current, I, has an immediate impact on the voltage, V. The output signal of the linear dynamic block is taken as the input of the steady-state one. The nonlinear steady-state part of the model is described by the general equation

$$y(k) = g\left(v(k)\right). \tag{18}$$

As the differentiable function $g: \mathbb{R} \to \mathbb{R}$ a neural network with one input, one hidden layer with K units and one output is used [25]. The model output is

$$y(k) = w_0^2 + \sum_{l=1}^{K} w_l^2 \varphi\left(w_{l,0}^1 + w_{l,1}^1 v(k)\right), \tag{19}$$

where $\varphi: \mathbb{R} \longrightarrow \mathbb{R}$ is a nonlinear transfer function (e.g., hyperbolic tangent). Weights of the network are denoted by $w_{l,m}^1$, $l = 1, \ldots, K$, $m = 0, 1$ and w_l^2, $l = 0, \ldots, K$, for the first and the second layers, respectively. The total number of weights is $3K+1$. All parameters of the Wiener model, i.e., the parameters of the dynamic part and weights of the neural network, are determined from an identification procedure. During identification, the model error defined by

$$E = \sum_{k=1}^{n_{\text{samples}}} \left(y_{\text{mod}}(k) - y(k)\right)^2 \tag{20}$$

is minimized, where $y_{\text{mod}}(k)$ and $y(k)$ denote the model output and the training data, respectively, n_{samples} denotes the number of data samples. Since the model is nonlinear, optimization of the model parameters is a nonlinear optimization task which is solved off-line. For this purpose, the sequential quadratic programming (SQP) algorithm is used [43], which makes it possible to take into account constraints during optimization. To enforce stability of the Wiener model, the poles of linear dynamic block are optimized subject to stability constraints. (All poles must belong to the unit circle.) Next, from the optimized poles the model coefficients a_i (Eq. 17) are calculated. The values of b_i^j, $w_{l,m}^1$ and w_l^2 are directly calculated (optimized) with no constraints.

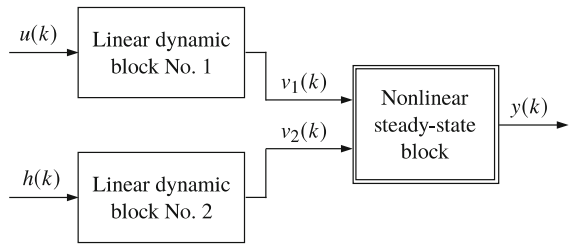

Fig. 2 Structure B of the Wiener model

3.2 Wiener model: structure B

Figure 2 depicts the second structure of the Wiener model (structure B). It consists of two linear dynamic blocks and a nonlinear steady-state one, but unlike structure A, it has two inputs. The outputs of the linear blocks are characterized by the equations

$$v_1(k) = \sum_{i=1}^{n_B^1} b_i^1 u(k-i) - \sum_{i=1}^{n_A^1} a_i^1 v_1(k-i), \tag{21}$$

$$v_2(k) = \sum_{i=0}^{n_B^2} b_i^2 h(k-i) - \sum_{i=1}^{n_A^2} a_i^2 v_2(k-i). \tag{22}$$

The integers n_A^j, n_B^j, $j = 1, 2$ define the order of the model dynamics. The constant parameters of the linear dynamic blocks are denoted by the real numbers a_i^1 $(i = 1, \ldots, n_A^1)$, a_i^2 $(i = 1, \ldots, n_A^2)$, b_i^1 $(i = 1, \ldots, n_B^1)$ and b_i^2 $(i = 0, \ldots, n_B^1)$. The signal $v_2(k)$ depends on the signal $h(k)$ since the current, I, has an immediate impact on the voltage, V. The nonlinear steady-state block is described by the general equation

$$y(k) = g\left(v_1(k), v_2(k)\right). \tag{23}$$

A neural network with two inputs, one hidden layer with K units and one output is used. The model output is

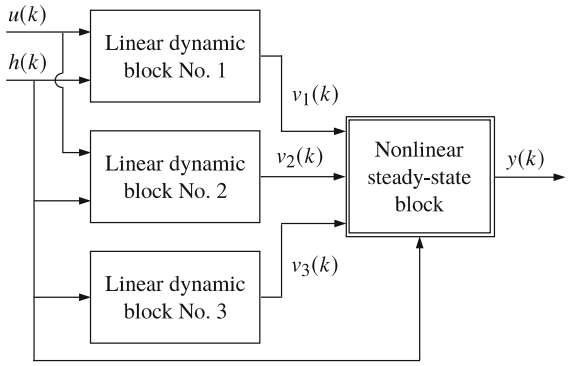

Fig. 3 Structure C of the Wiener model

$$y(k) = w_0^2 + \sum_{l=1}^{K} w_l^2 \varphi\left(w_{l,0}^1 + w_{l,1}^1 v_1(k) + w_{l,2}^1 v_2(k)\right). \tag{24}$$

The weights of the network are denoted by $w_{l,j}^1$, $l = 1, \ldots, K$, $j = 0, 1, 2$ and w_l^2, $l = 0, \ldots, K$, for the first and the second layers, respectively. The overall number of weights is $4K + 1$. The parameters of the second type of the Wiener model are also obtained by minimization of the model error function (20) subject to the constraints to enforce model stability (the poles of Eqs. 21 and 22 must belong to the unit circle).

3.3 Wiener model: structure C

Figure 3 depicts the third structure of the Wiener model (structure C). It has three linear dynamic blocks. They are characterized by the equations

$$v_1(k) = \sum_{i=1}^{n_B^{11}} b_i^{11} u(k-i) + \sum_{i=1}^{n_B^{12}} b_i^{12} h(k-i) - \sum_{i=1}^{n_A^1} a_i^1 v_1(k-i), \tag{25}$$

$$v_2(k) = \sum_{i=1}^{n_B^{21}} b_i^{21} u(k-i) + \sum_{i=1}^{n_B^{22}} b_i^{22} h(k-i) - \sum_{i=1}^{n_A^2} a_i^2 v_2(k-i), \tag{26}$$

$$v_3(k) = \sum_{i=1}^{n_B^3} b_i^3 h(k-i) - \sum_{i=1}^{n_A^3} a_i^3 v_3(k-i). \tag{27}$$

The integers n_A^j for $j = 1, 2, 3$, n_B^{ij} for $i = 1, 2$, $j = 1, 2$ and n_B^3 define the order of the model dynamics. The constant parameters of the linear dynamic blocks are denoted by the real numbers a_i^j ($i = 1, \ldots, n_A^j$, $j = 1, 2, 3$), b_i^{jl} ($i = 1, \ldots, n_B^j$, $j = 1, 2$, $l = 1, 2$) and b_i^3 ($i = 1, \ldots, n_B^3$). Unlike two previously discussed model structures, in structure C the steady-state block has an additional input which is the value of the disturbance signal, h, measured at the current sampling instant, k. The nonlinear steady-state block is described by the general equation

$$y(k) = g\left(v_1(k), v_2(k), v_3(k), h(k)\right). \tag{28}$$

A neural network with four inputs, one hidden layer with K units and one output is used. The model output is

$$y(k) = w_0^2 + \sum_{l=1}^{K} w_l^2 \varphi\left(w_{l,0}^1 + \sum_{j=1}^{3} w_{l,j}^1 v_j(k) + w_{l,4}^1 h(k)\right). \tag{29}$$

Weights of the network are denoted by $w_{l,j}^1$, $l = 1, \ldots, K$, $j = 0, \ldots, 4$ and w_l^2, $l = 0, \ldots, K$, for the first and the second layers, respectively. The overall number of weights is $6K + 1$. In this case, model parameters are also obtained by minimization of the model error function (20) subject to the constraints to enforce model stability (the poles of Eqs. 25–27 must belong to the unit circle).

4 Nonlinear model predictive control algorithms of PEM fuel cell

4.1 MPC problem formulation

MPC is an advanced control technique in which a dynamic model is used repeatedly online to predict the future behavior of the controlled process and an optimization procedure finds the best possible control policy [15–17]. The MPC approach has important advantages: It offers good control quality and takes into account all existing constraints imposed on process variables. As a result, MPC algorithms have been applied to numerous technological processes, mainly in industrial process control [44], e.g., chemical reactors [45], but also for control: computer networks [46],

unmanned vehicles [28], antilock brake systems in automobiles [47], unmanned helicopters [27], tractor–trailer vehicles [48], overhead cranes [49], underwater vehicles [50], active steering systems in cars [29] and even chaotic systems [51].

Let u denote the scaled input (manipulated) variable of the process and y denote the scaled output (controlled) variable (Eq. 16). In MPC algorithms [15–17] at each consecutive sampling instant k, a future control policy for a control horizon, N_u, is calculated. Typically, increments of the manipulated variable are found

$$\triangle \boldsymbol{u}(k) = [\triangle u(k|k) \ldots \triangle u(k+N_\mathrm{u}-1|k)]^\mathrm{T}, \tag{30}$$

where $\triangle u(k|k) = u(k|k) - u(k-1)$, $\triangle u(k+p|k) = u(k+p|k) - u(k+p-1|k)$ for $p = 1, \ldots, N_\mathrm{u}-1$. It is assumed that $\triangle u(k+p|k) = 0$ for $p \geq N_\mathrm{u}$, i.e., $u(k+p|k) = u(k+N_\mathrm{u}-1|k)$ for $p \geq N_\mathrm{u}$. The control increments (30) are calculated at each sampling instant from an optimization problem in which the predicted control errors are minimized. They are defined as the differences between the set-point trajectory, $y^\mathrm{sp}(k+p|k)$, and the predicted trajectory, $\hat{y}(k+p|k)$, over the prediction horizon $N \geq N_\mathrm{u}$. The MPC cost function is

$$J(k) = \sum_{p=1}^{N} (y^\mathrm{sp}(k+p|k) - \hat{y}(k+p|k))^2 + \lambda \sum_{p=0}^{N_\mathrm{u}-1} (\triangle u(k+p|k))^2. \tag{31}$$

A dynamic model of the controlled process is used to calculate online the predicted values of the process output variable. The second part of the MPC cost function is a penalty term used to limit increments of the manipulated variable. An advantage of MPC algorithms is the fact that all existing constraints may be taken into account in an optimization task solved at each sampling instant online to find the optimal increments of the manipulated variable (30). Assuming that there are constraints imposed on the value and the rate of change of the manipulated variables as well as on the predicted values of process output, the rudimentary MPC optimization problem is

$$\min_{\triangle u(k|k),\ldots,\triangle u(k+N_\mathrm{u}-1|k)} \{J(k)\},$$

subject to

$$\begin{aligned} u^\mathrm{min} &\leq u(k+p|k) \leq u^\mathrm{max}, \quad p = 0, \ldots, N_\mathrm{u}-1, \\ -\triangle u^\mathrm{max} &\leq \triangle u(k+p|k) \leq \triangle u^\mathrm{max}, \quad p = 0, \ldots, N_\mathrm{u}-1, \\ y^\mathrm{min} &\leq \hat{y}(k+p|k) \leq y^\mathrm{max}, \quad p = 1, \ldots, N. \end{aligned} \tag{32}$$

The minimal and maximal possible values of the manipulated variable are denoted by u^min and u^max, respectively, the maximal allowed increment of that variable is denoted by $\triangle u^\mathrm{max}$, and the minimal and maximal possible values of the predicted controlled variable are denoted by y^min and y^max, respectively. At each sampling instant, the whole sequence of control increments (30), of length N_u, is calculated, but only the first element is applied to the process, i.e., $u(k) = \triangle u(k|k) + u(k-1)$. At the next sampling instant, $k+1$, the whole procedure is repeated.

For further transformations, the MPC optimization problem (32) is expressed in a convenient vector–matrix notation

$$\begin{aligned} &\min_{\triangle \boldsymbol{u}(k)} \left\{ \left\| \boldsymbol{y}^\mathrm{sp}(k) - \hat{\boldsymbol{y}}(k) \right\|^2 + \|\triangle \boldsymbol{u}(k)\|^2_{\boldsymbol{\Lambda}} \right\}, \\ &\text{subject to} \\ &\boldsymbol{u}^\mathrm{min} \leq \boldsymbol{J}\triangle \boldsymbol{u}(k) + \boldsymbol{u}(k-1) \leq \boldsymbol{u}^\mathrm{max}, \\ &\quad -\triangle \boldsymbol{u}^\mathrm{max} \leq \triangle \boldsymbol{u}(k) \leq \triangle \boldsymbol{u}^\mathrm{max}, \\ &\quad \boldsymbol{y}^\mathrm{min} \leq \hat{\boldsymbol{y}}(k) \leq \boldsymbol{y}^\mathrm{max}, \end{aligned} \tag{33}$$

where the norms are $\|\boldsymbol{x}\|^2 = \boldsymbol{x}^\mathrm{T}\boldsymbol{x}$, $\|\boldsymbol{x}\|^2_{\boldsymbol{A}} = \boldsymbol{x}^\mathrm{T}\boldsymbol{A}\boldsymbol{x}$, the predicted trajectory of the output variable over the prediction horizon and the set-point trajectory are vectors of length N

$$\hat{\boldsymbol{y}}(k) = \left[\hat{y}(k+1|k) \ldots \hat{y}(k+N|k)\right]^\mathrm{T}, \tag{34}$$

$$\boldsymbol{y}^\mathrm{sp}(k) = \left[y^\mathrm{sp}(k+1|k) \ldots y^\mathrm{sp}(k+N|k)\right]^\mathrm{T}. \tag{35}$$

The vectors of length N_u are: $\boldsymbol{u}^\mathrm{min} = \left[u^\mathrm{min} \ldots u^\mathrm{min}\right]^\mathrm{T}$, $\boldsymbol{u}^\mathrm{max} = \left[u^\mathrm{max} \ldots u^\mathrm{max}\right]^\mathrm{T}$, $\triangle \boldsymbol{u}^\mathrm{max} = \left[\triangle u^\mathrm{max} \ldots \triangle u^\mathrm{max}\right]^\mathrm{T}$, $\boldsymbol{u}(k-1) = [u(k-1) \ldots u(k-1)]^\mathrm{T}$; the vectors of length N are: $\boldsymbol{y}^\mathrm{min} = \left[y^\mathrm{min} \ldots y^\mathrm{min}\right]^\mathrm{T}$, $\boldsymbol{y}^\mathrm{max} = \left[y^\mathrm{max} \ldots y^\mathrm{max}\right]^\mathrm{T}$. The matrix $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda, \ldots, \lambda)$ and the lower ones matrix $\boldsymbol{J}$ (its diagonal and below diagonal entries are equal to 1, and the entries over the diagonal are equal to 0) are of dimensionality $N_\mathrm{u} \times N_\mathrm{u}$.

The Wiener models of the PEM fuel cell are nonlinear, so the predicted values of the controlled variable are nonlinear functions of the currently calculated sequence of increments of the manipulated variable, Eq. (30). Hence, the general MPC optimization problem (32) becomes a nonlinear task which must

be solved at each sampling instant online. To reduce the computational complexity, two alternatives are discussed in the following part of the article.

4.2 Nonlinear MPC algorithm with nonlinear prediction and simplified model linearization (MPC-NPSL)

In the first approach regarding computationally efficient MPC of the PEM fuel cell, a linear approximation of the Wiener model is successively calculated online at each sampling instant k and next used for finding the predicted trajectory of the process output [26]. The model is linearized for the current operating conditions. During linearization, the serial structure of the Wiener model is used. Although all three model types may be used, the MPC-NPSL algorithm for the most complex Wiener structure C shown in Fig. 3 is detailed. The algorithm for structures A and B may be easily derived from the given description.

Online linearization is performed in a simple way: The time-varying gains of the nonlinear steady-state blocks of the model are calculated for the current operating point, and next, taking into account the serial structure of the model, the gain of the whole model is updated. From Eq. (29), the time-varying gains of the v_i to y channels ($i = 1, 2, 3$) of the nonlinear block can be obtained as

$$K_i(k) = \frac{\mathrm{d}y(k)}{\mathrm{d}v_i(k)} = \sum_{l=1}^{K} w_l^2 \frac{\mathrm{d}z_l(k)}{\mathrm{d}z_l(k)} w_{l,i}^1, \tag{36}$$

where $z_l(k) = w_{l,0}^1 + \sum_{j=1}^{3} w_{l,j}^1 v_j(k) + w_{l,4}^1 h(k)$. If the hyperbolic tangent is used as the activation function of the hidden layer of the neural network, i.e., $\varphi = \tanh(\cdot)$, it results in

$$\frac{\mathrm{d}z_l(k)}{\mathrm{d}z_l(k)} = 1 - \tanh^2\left(z_l(k)\right). \tag{37}$$

Taking into account the serial structure of Wiener model C shown in Fig. 3, a linear approximation of the model output signal is

$$y(k) = K_1(k)v_1(k) + K_2(k)v_2(k) + K_3(k)v_3(k). \tag{38}$$

Remembering that the dynamic blocks are linear with constant parameters (Eqs. 25–27), the model used in the MPC-NPSL algorithm (Eq. 38) is linear, but time-varying. In all MPC algorithms based on constant linear models, the predicted trajectory of the output (Eq. 34) is a linear combination of the decision variables of MPC [17]. Using the concept of linear MPC and taking into account the time-varying linear approximation of Wiener structure C defined by Eq. (38), the prediction equation is obtained

$$\hat{\boldsymbol{y}}(k) = (K_1(k)\boldsymbol{G}_1(k) + K_2(k)\boldsymbol{G}_2(k))\triangle\boldsymbol{u}(k) + \boldsymbol{y}^0(k). \tag{39}$$

The predicted output trajectory, $\hat{\boldsymbol{y}}(k)$, is a sum of two parts: the forced trajectory, $(K_1(k)\boldsymbol{G}_1(k) + K_2(k)\boldsymbol{G}_2(k))\triangle\boldsymbol{u}(k)$, and the free trajectory (a vector of length N)

$$\boldsymbol{y}^0(k) = \left[y^0(k+1|k) \ldots y^0(k+N|k)\right]^{\mathrm{T}}. \tag{40}$$

The first trajectory depends on the currently calculated future control increments, whereas the second one depends only on the past. It is straightforward to notice from Eqs. (25)–(27) that the manipulated variable, u, influences only the first two intermediate model variables, v_1 and v_2. Hence, only the channels $u - v_1 - y$ and $u - v_2 - y$ are considered in the forced trajectory. The constant matrices of dimensionality $N \times N_{\mathrm{u}}$

$$\boldsymbol{G}_n = \begin{bmatrix} s_1^n & 0 & \ldots & 0 \\ s_2^n & s_1^n & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N^n & s_{N-1}^n & \ldots & s_{N-N_{\mathrm{u}}+1}^n \end{bmatrix} \tag{41}$$

comprise step response matrices of the channels $u - v_1$ ($n = 1$) and $u - v_2$ ($n = 2$), respectively. They are calculated off-line in the classical way [17]. For this purpose, the first two dynamic blocks (Eqs. 25, 26) are taken into account without any influence of the disturbance signal, h.

In the MPC-NPSL algorithm, it is also necessary to find the free trajectory, $\boldsymbol{y}^0(k)$, (Eq. 40). It is calculated at each sampling instant not from the simplified linearized model (38), but from the full nonlinear Wiener model. From Eq. (29), one has

$$y^0(k+p|k) = w_0^2 + \sum_{l=1}^{K} w_l^2 \varphi\Big(w_{l,0}^1$$

$$+\sum_{j=1}^{3} w^1_{l,j} v^0_j(k+p|k) + w^1_{l,4} h_{\text{meas}}(k+p|k)\Big) + d(k). \quad (42)$$

The free trajectories of the variables v_1, v_2 and v_3 are denoted by v^0_1, v^0_2 and v^0_3, respectively. The free trajectory of the variable v_1 is obtained from Eq. (25)

$$\begin{aligned} v^0_1(k+p|k) = &\sum_{i=1}^{I^1_{\text{uf}}(p)} b^{11}_i u(k-1) \\ &+ \sum_{i=I^1_{\text{uf}}(p)+1}^{n^{11}_{\text{B}}} b^{11}_i u(k-i+p) \\ &+ \sum_{i=1}^{n^{12}_{\text{B}}} b^{12}_i h_{\text{meas}}(k-i+p|k) \\ &- \sum_{i=1}^{I^1_{\text{vf}}(p)} a^1_i v_1(k-i+p|k) \\ &- \sum_{i=I_{\text{vf}^1}(p)+1}^{n^1_{\text{A}}} a^1_i v_1(k-i+p), \end{aligned} \quad (43)$$

where $I^1_{\text{uf}}(p) = \min(p, n^{11}_{\text{B}})$, $I^1_{\text{vf}}(p) = \min(p-1, n^1_{\text{A}})$. The free trajectory of the variable v_2 is obtained from Eq. (26)

$$\begin{aligned} v^0_2(k+p|k) = &\sum_{i=1}^{I^2_{\text{uf}}(p)} b^{21}_i u(k-1) \\ &+ \sum_{i=I^2_{\text{uf}}(p)+1}^{n^{21}_{\text{B}}} b^{21}_i u(k-i+p) \\ &+ \sum_{i=1}^{n^{22}_{\text{B}}} b^{22}_i h_{\text{meas}}(k-i+p|k) \\ &- \sum_{i=1}^{I^2_{\text{vf}}(p)} a^2_i v_2(k-i+p|k) \\ &- \sum_{i=I_{\text{vf}^2}(p)+1}^{n^2_{\text{A}}} a^2_i v_2(k-i+p), \end{aligned} \quad (44)$$

where $I^2_{\text{uf}}(p) = \min(p, n^{21}_{\text{B}})$, $I^2_{\text{vf}}(p) = \min(p-1, n^2_{\text{A}})$. The free trajectory of the variable v_3 is obtained from Eq. (27)

$$\begin{aligned} v^0_3(k+p|k) = b^3_i h_{\text{meas}}(k-i+p|k) \cdot &\\ - \sum_{i=1}^{I^3_{\text{vf}}(p)} a^3_i v_3(k-i+p|k) &\\ - \sum_{i=I_{\text{vf}^3}(p)+1}^{n^2_{\text{A}}} a^3_i v_3(k-i+p), & \end{aligned} \quad (45)$$

where $I^3_{\text{vf}}(p) = \min(p-1, n^3_{\text{A}})$. The measured value of the disturbance is typically known up to the current sampling instant, i.e.,

$$h_{\text{meas}}(k+p|k) = \begin{cases} 0.01(I(k+p) - \bar{I}) & \text{when} \quad p < 0, \\ 0.01(I(k) - \bar{I}) & \text{when} \quad p \geq 0. \end{cases} \quad (46)$$

The unmeasured disturbance acting on the process output, $d(k)$, used in the free trajectory (Eq. 42), is calculated as difference between the value of the output signal measured at the current sampling instant, $y(k)$, and process output estimated from the model. Using Eq. (29), one obtains

$$\begin{aligned} d(k) = y(k) &\\ -w^2_0 - \sum_{l=1}^{K} w^2_l \varphi &\left(w^1_{l,0} + \sum_{j=1}^{3} w^1_{l,j} v_j(k) + w^1_{l,4} h(k) \right). \end{aligned} \quad (47)$$

Because in the MPC-NPSL algorithm for prediction of the predicted output trajectory (Eq. (39)) a linear function of the future control increments $\Delta \boldsymbol{u}(k)$ is used, the general MPC optimization problem (33) is transformed to the MPC-NPSL quadratic programming task

$$\begin{aligned} \min_{\Delta \boldsymbol{u}(k)} &\Big\{ \big\| \boldsymbol{y}^{\text{sp}}(k) - (K_1(k)\boldsymbol{G}_1(k) + K_2(k)\boldsymbol{G}_2(k))\Delta \boldsymbol{u}(k) \\ &\quad - \boldsymbol{y}^0(k)\boldsymbol{u}(k) \big\|^2 + \|\Delta \boldsymbol{u}(k)\|^2_{\boldsymbol{\Lambda}} \Big\}, \\ \text{subject to} &\\ \boldsymbol{u}^{\min} &\leq \boldsymbol{J}\Delta \boldsymbol{u}(k) + \boldsymbol{u}(k-1) \leq \boldsymbol{u}^{\max}, \\ -\Delta \boldsymbol{u}^{\max} &\leq \Delta \boldsymbol{u}(k) \leq \Delta \boldsymbol{u}^{\max}, \\ \boldsymbol{y}^{\min} &\leq (K_1(k)\boldsymbol{G}_1(k) + K_2(k)\boldsymbol{G}_2(k))\Delta \boldsymbol{u}(k) \\ &\quad + \boldsymbol{y}^0(k) \leq \boldsymbol{y}^{\max}. \end{aligned} \quad (48)$$

4.3 Nonlinear MPC algorithm with nonlinear prediction and linearization along the trajectory(MPC-NPLT)

In the second approach to computationally efficient MPC of the PEM fuel cell, a linear approximation of the predicted trajectory of the process output is directly calculated online, at each sampling instant k [26]. It should be noted that in the MPC-NPSL algorithm a linear approximation of the model is successively found online and next used for calculation of the predicted trajectory of the controlled variable. This approach uses a constant linearized model over the whole prediction horizon, which is a disadvantage.

In the MPC-NPLT algorithm, linearization of the output trajectory is performed for an assumed future trajectory of the input variable $\boldsymbol{u}^{\text{traj}}(k) = \left[u^{\text{traj}}(k|k) \ldots u^{\text{traj}}(k+N_{\text{u}}-1|k)\right]^{\text{T}}$. In this work, that trajectory is defined as the last $N_{\text{u}}-1$ elements of the optimal trajectory calculated at the previous instant and not applied to the process

$$\boldsymbol{u}^{\text{traj}}(k) = \begin{bmatrix} u(k|k-1) \\ \vdots \\ u(k+N_{\text{u}}-3|k-1) \\ u(k+N_{\text{u}}-2|k-1) \\ u(k+N_{\text{u}}-2|k-1) \end{bmatrix}. \tag{49}$$

The last element is repeated twice since the control increment for the sampling instant $k+N_{\text{u}}-1$ is not calculated in the previous sampling instant. Using Taylor's series expansion method, a linear approximation of the nonlinear output trajectory $\hat{\boldsymbol{y}}(k)$ (Eq. 34) along the input trajectory $\boldsymbol{u}^{\text{traj}}(k)$, i.e., linearization of the function $\hat{\boldsymbol{y}}(\boldsymbol{u}(k)) \colon \mathbb{R}^{N_{\text{u}}} \rightarrow \mathbb{R}^{N}$ is

$$\hat{\boldsymbol{y}}(k) = \hat{\boldsymbol{y}}^{\text{traj}}(k) + \boldsymbol{H}(k)(\boldsymbol{u}(k) - \boldsymbol{u}^{\text{traj}}(k)), \tag{50}$$

where $\boldsymbol{u}(k) = [u(k|k) \ldots u(k+N_{\text{u}}-1|k)]^{\text{T}}$ is the future trajectory of the manipulated variable corresponding to the calculated increments, $\triangle\boldsymbol{u}(k)$, (Eq. 30). The predicted trajectory of the controlled variable, $\hat{\boldsymbol{y}}^{\text{traj}}(k)$, corresponds to the assumed trajectory of the manipulated variable, $\boldsymbol{u}^{\text{traj}}(k)$. The $N \times N_{\text{u}}$ matrix of the derivatives of the predicted trajectory of the controlled variable with respect of the assumed trajectory of the manipulated variable has the structure

$$\begin{aligned} \boldsymbol{H}(k) &= \left.\frac{\mathrm{d}\hat{\boldsymbol{y}}(k)}{\mathrm{d}\boldsymbol{u}(k)}\right|_{\substack{\hat{\boldsymbol{y}}(k)=\hat{\boldsymbol{y}}^{\text{traj}}(k) \\ \boldsymbol{u}(k)=\boldsymbol{u}^{\text{traj}}(k)}} \\ &= \begin{bmatrix} \dfrac{\partial \hat{y}^{\text{traj}}(k+1|k)}{\partial u^{\text{traj}}(k|k)} & \cdots & \dfrac{\partial \hat{y}^{\text{traj}}(k+1|k)}{\partial u^{\text{traj}}(k+N_{\text{u}}-1|k)} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial \hat{y}^{\text{traj}}(k+N|k)}{\partial u^{\text{traj}}(k|k)} & \cdots & \dfrac{\partial \hat{y}^{\text{traj}}(k+N|k)}{\partial u^{\text{traj}}(k+N_{\text{u}}-1|k)} \end{bmatrix}. \end{aligned} \tag{51}$$

The linear approximation of the predicted process trajectory (50) may be expressed as a function of the future trajectory of the increments of the manipulated variable, which is repeatedly calculated in MPC at each sampling instant

$$\begin{aligned} \hat{\boldsymbol{y}}(k) = \boldsymbol{H}(k)\boldsymbol{J}\triangle\boldsymbol{u}(k) + \hat{\boldsymbol{y}}^{\text{traj}}(k) \\ + \boldsymbol{H}(k)(\boldsymbol{u}(k-1) - \boldsymbol{u}^{\text{traj}}(k)). \end{aligned} \tag{52}$$

Using the linear approximation of the predicted trajectory of the controlled variable (Eq. 52), from the rudimentary MPC optimization problem (33) the following MPC-NPLT quadratic optimization task is obtained

$$\begin{aligned} &\min_{\triangle\boldsymbol{u}(k)} \Big\{ \big\| \boldsymbol{y}^{\text{sp}}(k) - \boldsymbol{H}(k)\boldsymbol{J}\triangle\boldsymbol{u}(k) - \hat{\boldsymbol{y}}^{\text{traj}}(k) \\ &\qquad \boldsymbol{H}(k)(\boldsymbol{u}(k-1) - \boldsymbol{u}^{\text{traj}}(k)) \big\|^2 \|\triangle\boldsymbol{u}(k)\|^2_{\boldsymbol{\Lambda}} \Big\}, \\ &\text{subject to} \\ &\qquad \boldsymbol{u}^{\min} \leq \boldsymbol{J}\triangle\boldsymbol{u}(k) + \boldsymbol{u}(k-1) \leq \boldsymbol{u}^{\max}, \\ &\qquad -\triangle\boldsymbol{u}^{\max} \leq \triangle\boldsymbol{u}(k) \leq \triangle\boldsymbol{u}^{\max}, \\ &\qquad \boldsymbol{y}^{\min} \leq \boldsymbol{H}(k)\boldsymbol{J}\triangle\boldsymbol{u}(k) + \hat{\boldsymbol{y}}^{\text{traj}}(k) \\ &\qquad\qquad + \boldsymbol{H}(k)(\boldsymbol{u}(k-1) - \boldsymbol{u}^{\text{traj}}(k)) \leq \boldsymbol{y}^{\max}. \end{aligned} \tag{53}$$

The predicted trajectory $\hat{\boldsymbol{y}}^{\text{traj}}(k)$ and the matrix $\boldsymbol{H}(k)$ are calculated directly from the full nonlinear model of the process, without any simplification. Although all three model types may be used, the MPC-NPLT algorithm for the most complex Wiener structure C shown in Fig. 3 is detailed. From Eq. (29), the predicted trajectory is

$$y^{\text{traj}}(k+p|k) = w_0^2 + \sum_{l=1}^{K} w_l^2 \varphi\Bigg(w_{l,0}^1 + \sum_{j=1}^{3} w_{l,j}^1 v_j^{\text{traj}}(k+p|k) + w_{l,4}^1 h_{\text{meas}}(k+p|k)\Bigg) + d(k). \tag{54}$$

From Eq. (25), the predicted trajectory of the variable v_1 is

$$v_1^{\text{traj}}(k+p|k) = \sum_{i=1}^{I_{\text{uf}}^1(p)} b_i^{11} u^{\text{traj}}(k-i+p|k) + \sum_{i=I_{\text{uf}}^1(p)+1}^{n_{\text{B}}^1} b_i^{11} u(k-i+p) + \sum_{i=0}^{n_{\text{B}}^{12}} b_i^{12} h_{\text{meas}}(k-i+p|k) - \sum_{i=1}^{I_{\text{vf}}^1(p)} a_i^1 v_1^{\text{traj}}(k-i+p|k) - \sum_{i=I_{\text{vf}}^1(p)+1}^{n_{\text{A}}} a_i^1 v_1(k-i+p). \tag{55}$$

From Eq. (26), the predicted trajectory of the variable v_2 is

$$v_2^{\text{traj}}(k+p|k) = \sum_{i=1}^{I_{\text{uf}}^2(p)} b_i^{21} u^{\text{traj}}(k-i+p|k) + \sum_{i=I_{\text{uf}}^2(p)+1}^{n_{\text{B}}^2} b_i^{21} u(k-i+p) + \sum_{i=0}^{n_{\text{B}}^{22}} b_i^{22} h_{\text{meas}}(k-i+p|k) - \sum_{i=1}^{I_{\text{vf}}^2(p)} a_i^2 v_2^{\text{traj}}(k-i+p|k) - \sum_{i=I_{\text{vf}}^2(p)+1}^{n_{\text{A}}} a_i^2 v_1(k-i+p). \tag{56}$$

From Eq. (27), the predicted trajectory of the variable v_3 is

$$v_3^{\text{traj}}(k+p|k) = \sum_{i=0}^{n_{\text{B}}^3} b_i^3 h_{\text{meas}}(k-i+p|k) - \sum_{i=1}^{I_{\text{vf}}^3(p)} a_i^3 v_3^{\text{traj}}(k-i+p|k) - \sum_{i=I_{\text{vf}}^3(p)+1}^{n_{\text{A}}} a_i^3 v_3(k-i+p). \tag{57}$$

The unmeasured disturbance is assessed in the same way as in the MPC-NPSL algorithm (Eq. 47).

The entries of the matrix $\boldsymbol{H}(k)$ are determined differentiating Eq. (54) which leads to

$$\frac{\partial y^{\text{traj}}(k+p|k)}{\partial u^{\text{traj}}(k+r|k)} = \sum_{l=1}^{K} w_l^2 \frac{\mathrm{d}z_l^{\text{traj}}(k+p|k)}{\mathrm{d}z_l^{\text{traj}}(k+p|k)} \times \sum_{j=1}^{2} w_{l,j}^1 \frac{\partial v_j^{\text{traj}}(k+p|k)}{\partial u^{\text{traj}}(k+r|k)}, \tag{58}$$

for all $p = 1, \ldots, N$, $r = 0, \ldots, N_{\text{u}} - 1$, where $z_l(k+p|k) = w_{l,0}^1 + \sum_{j=1}^{3} w_{l,j}^1 v_j(k+p|k) + w_{l,4}^1 h_{\text{meas}}(k)$. The first derivative in Eq. (58) depends on the transfer function used in the neural network, for the hyperbolic tangent the general Eq. (37) is used. The partial derivatives are calculated differentiating Eqs. (55) and (56). In general, for $j = 1, 2$, one has

$$\frac{\partial v_j^{\text{traj}}(k+p|k)}{\partial u^{\text{traj}}(k+r|k)} = \sum_{i=1}^{I_{\text{uf}}^j(p)} b_i^{j1} \frac{\partial u^{\text{traj}}(k-i+p|k)}{\partial u^{\text{traj}}(k+r|k)} - \sum_{i=1}^{I_{\text{vf}}^j(p)} a_i^j \frac{\partial v_j^{\text{traj}}(k-i+p|k)}{\partial u^{\text{traj}}(k+r|k)}. \tag{59}$$

The first partial derivative in the right side of Eq. (59) is

$$\frac{\partial u^{\text{traj}}(k+p|k)}{\partial u^{\text{traj}}(k+r|k)} = \begin{cases} 1 & \text{if } p = r \text{ or } (p > r \text{ and } r = N_{\text{u}} - 1), \\ 0 & \text{otherwise,} \end{cases} \tag{60}$$

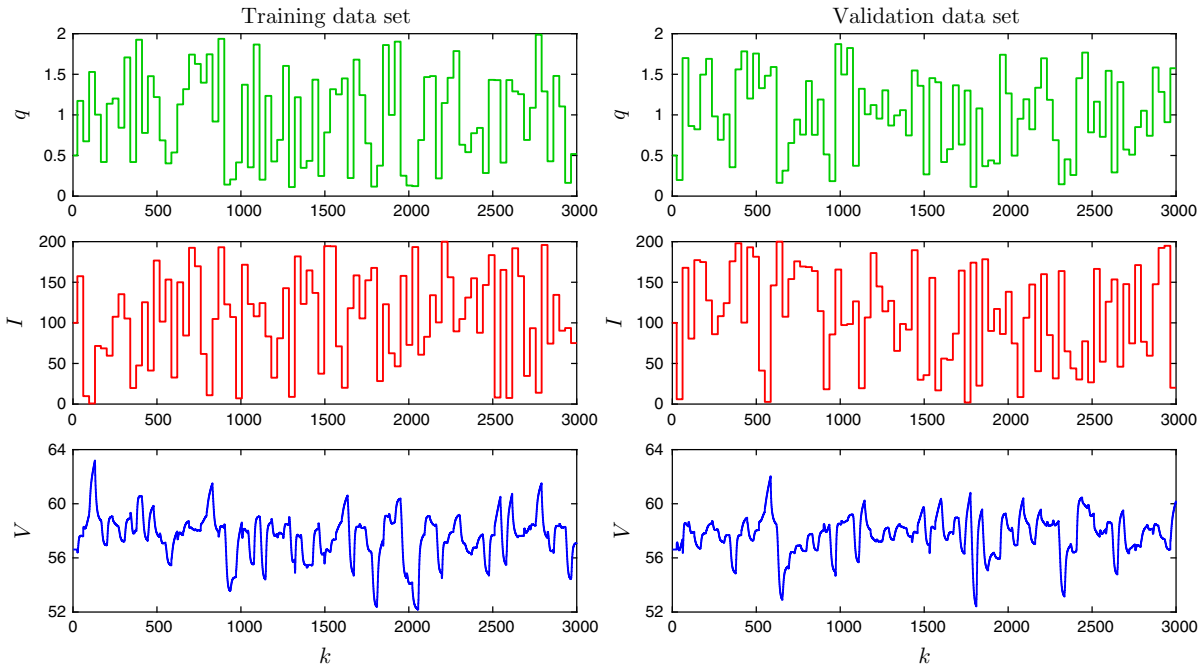

Fig. 4 Training (left) and validation (right) data sets

and the second one is calculated recurrently for the consecutive values of the indices p and r.

5 Simulation results

5.1 Model identification of PEM fuel cell

The objective of this subsection is to find precise black box models of the PEM fuel cell. A linear model and three discussed Wiener structures (A, B and C) are considered. All models are assessed in terms of the SSE error (Eq. 20) and the number of model parameters. During model identification two data sets are used: the training data set and the validation one. The first of them is used only to find parameters of models, whereas the second one is used only to assess generalization ability of models, i.e., how the model reacts when it is excited by a different data set than that used for identification. To obtain those two sets of data, the continuous-time fundamental model of the PEM process (defined by Eqs. 2, 3, 6, 8–13) is simulated. The model system of differential equations is solved by Runge–Kutta method of order 45. As the process input and disturbance signals random sequences from the range characterized by Eqs. (14) and (15) are used. The process signals (i.e., the manipulated variable, q, the disturbance, I, and the controlled variable, V) are sampled with the sampling period equal to 1 s. The training and validation data sets are shown in Fig. 4, both sets consist of 3000 samples of the process manipulated variable, the disturbance and the output. Since identification of nonlinear Wiener models is a nonlinear optimization problem, training is repeated as many as 10 times for each model configuration and the results presented next are the best obtained.

At first linear models of the process are considered. They have the following structure

$$v(k) = \sum_{i=1}^{n_{\mathrm{B}}^1} b_i^1 u(k-i) + \sum_{i=0}^{n_{\mathrm{B}}^2} b_i^2 h(k-i) - \sum_{i=1}^{n_{\mathrm{A}}} a_i y(k-i). \tag{61}$$

Table 3 compares training and validation errors of linear models of the first, the second, the third and the fourth order (the order is defined as an integer number $n_{\mathrm{B}}^1 = n_{\mathrm{B}}^2 = n_{\mathrm{A}}$). As a compromise between model accuracy and complexity, the third-order model is chosen. Figure 5 compares the validation data set versus the output of the chosen linear model. The model is stable,

Table 3 Training error, E_{train}, and the validation error, E_{val}, for the linear model; the errors of the chosen model are emphasized

Model order	E_{train}	E_{val}
First order	1.4509×10^3	8.1537×10^2
Second order	1.4482×10^3	8.4352×10^2
Third order	$\mathbf{1.0560 \times 10^3}$	$\mathbf{5.4921 \times 10^2}$
Fourth order	9.5139×10^2	4.9939×10^2

but not precise since there are significant differences between the model output and the data.

Next, Wiener structure A is considered. Table 4 presents training and validation errors for models of different orders (the order is defined as an integer number $n_{\text{B}}^1 = n_{\text{B}}^2 = n_{\text{A}}$) and different numbers of hidden nodes, K. All compared models are of very low quality, only slightly better than the linear models (Table 3). Model complexity (defined by the order of dynamics and the number of hidden nodes) has practically no influence on model accuracy. For further comparison, the third-order model containing five hidden nodes is chosen. The top part of Fig. 6 compares the validation data set versus the output of Wiener model A. Slightly better results in comparison with the linear structure (Fig. 5) can be obtained, but still significant differences between model output and data are present.

The training and validation errors of Wiener structure B are given in Table 5 for models of different orders (the order is defined as an integer number

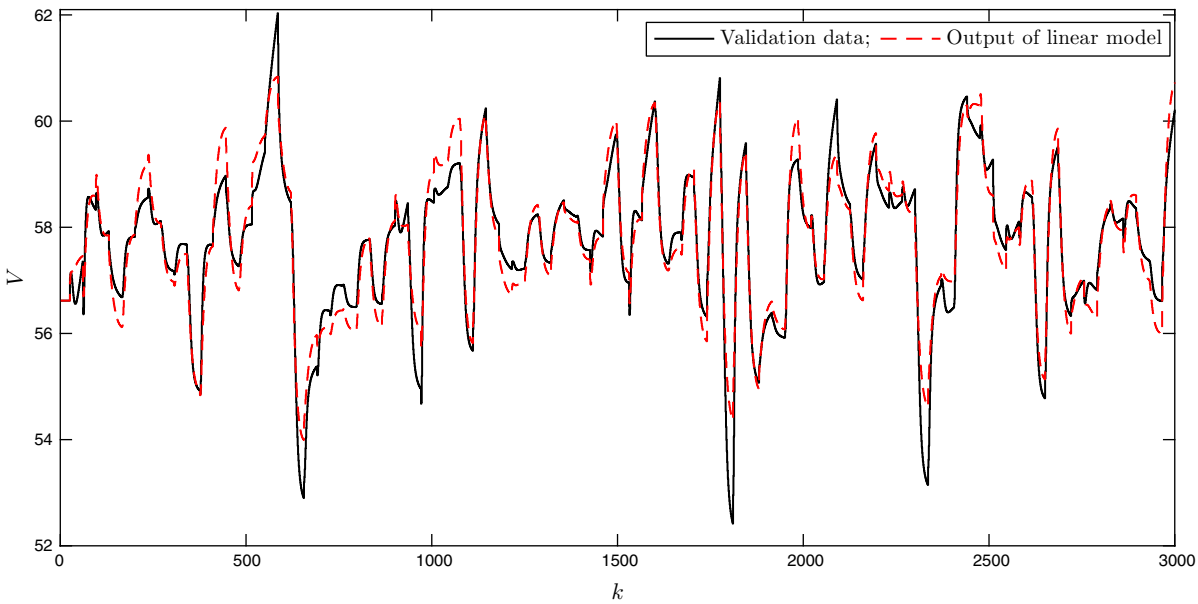

Fig. 5 Validation data set versus the output of the third-order linear model

Table 4 Training error, E_{train}, and the validation error, E_{val}, for structure A of the Wiener model; K is the number of hidden nodes of the neural steady-state block; the errors of the chosen model are emphasized

K	First order		Second order		Third order		Fourth order	
	E_{train}	E_{val}	E_{train}	E_{val}	E_{train}	E_{val}	E_{train}	E_{val}
1	6.6838×10^2	4.5577×10^2	5.0341×10^2	4.2012×10^2	6.6000×10^2	4.5332×10^2	5.4235×10^2	4.6676×10^2
5	5.1987×10^2	3.9975×10^2	4.4502×10^2	4.2231×10^2	$\mathbf{3.9532 \times 10^2}$	$\mathbf{4.0533 \times 10^2}$	4.0093×10^2	3.9105×10^2
10	5.3683×10^2	3.9015×10^2	4.6211×10^2	4.2378×10^2	4.0344×10^2	4.0245×10^2	3.8761×10^2	3.9924×10^2
15	4.9882×10^1	4.0700×10^2	4.3372×10^2	4.2657×10^2	3.7396×10^2	3.8778×10^2	5.2775×10^2	3.5779×10^2
20	5.0761×10^2	3.9746×10^2	4.7123×10^2	3.9891×10^2	4.6384×10^2	3.9448×10^2	3.7140×10^2	3.7399×10^2

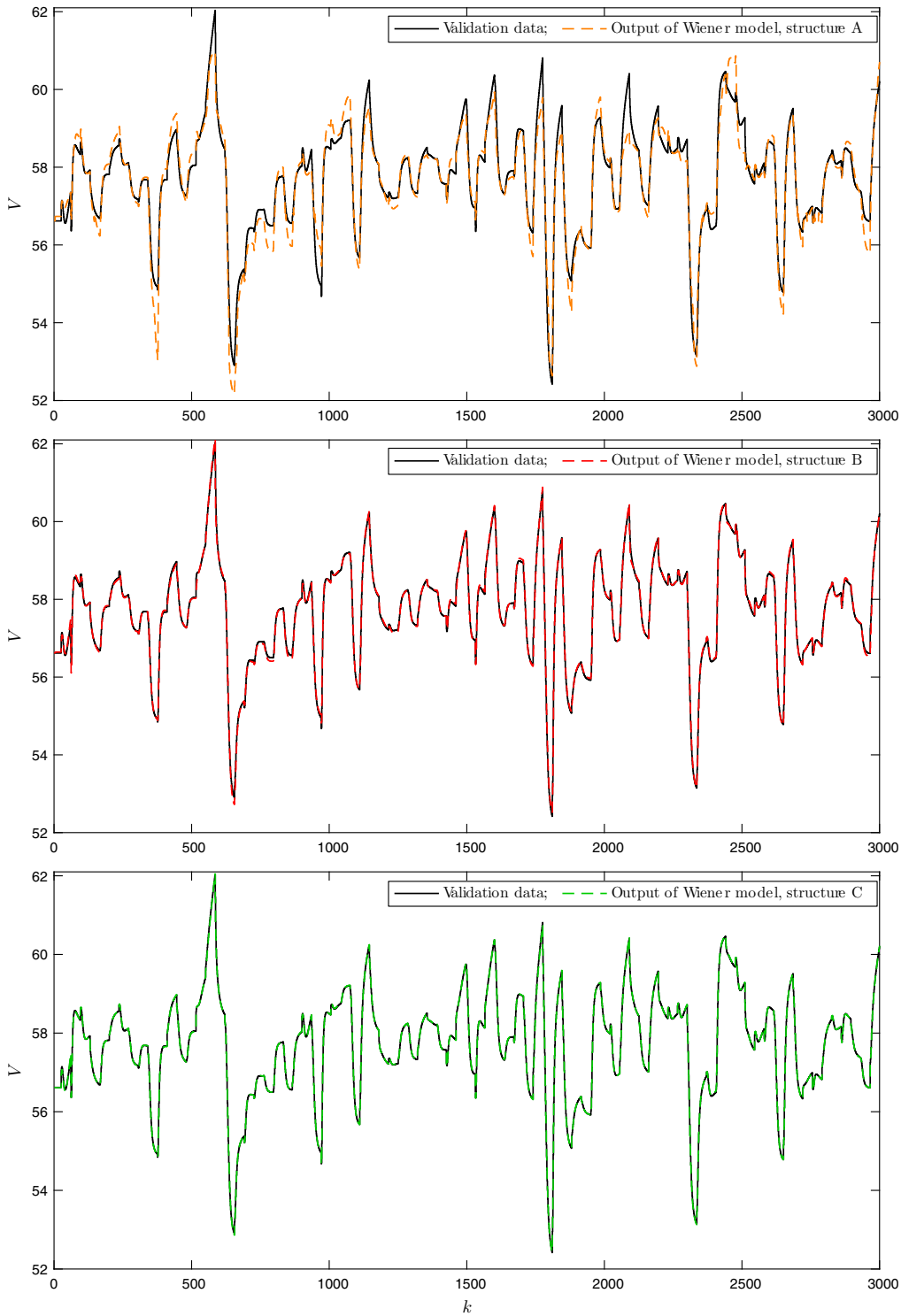

Fig. 6 Validation data set versus the output of the chosen third-order Wiener models (structures A, B and C) containing $K = 5$ hidden nodes

Table 5 Training error, E_{train}, and the validation error, E_{val}, for structure B of the Wiener model; K is the number of hidden nodes of the neural steady-state block; the errors of the chosen model are emphasized

K	First order		Second order		Third order		Fourth order	
	E_{train}	E_{val}	E_{train}	E_{val}	E_{train}	E_{val}	E_{train}	E_{val}
1	5.8138×10^{2}	4.8800×10^{2}	5.4311×10^{2}	4.6994×10^{2}	5.4305×10^{2}	4.7308×10^{2}	5.4290×10^{2}	4.7271×10^{2}
5	9.7195×10^{1}	7.5512×10^{1}	1.2616×10^{1}	1.1287×10^{1}	$\mathbf{1.1623 \times 10^{1}}$	$\mathbf{1.1491 \times 10^{1}}$	1.1399×10^{1}	1.0650×10^{1}
10	9.6519×10^{1}	7.5056×10^{1}	1.4029×10^{1}	1.2614×10^{1}	1.0858×10^{1}	1.0808×10^{1}	9.7289×10^{0}	1.0733×10^{1}
15	8.8093×10^{1}	8.7314×10^{1}	1.0710×10^{1}	1.1261×10^{1}	1.1962×10^{1}	1.1709×10^{1}	1.0402×10^{1}	1.0221×10^{1}
20	8.6768×10^{1}	9.1449×10^{1}	1.3866×10^{1}	1.2104×10^{1}	1.1185×10^{1}	1.1996×10^{1}	9.1614×10^{0}	1.0818×10^{1}

Table 6 Training error, E_{train}, and the validation error, E_{val}, for structure C of the Wiener model; K is the number of hidden nodes of the neural steady-state block; the errors of the chosen model are emphasized

K	First order		Second order		Third order		Fourth order	
	E_{train}	E_{val}	E_{train}	E_{val}	E_{train}	E_{val}	E_{train}	E_{val}
1	6.8804×10^{2}	4.8451×10^{2}	5.4451×10^{2}	4.6874×10^{2}	5.4365×10^{2}	4.5806×10^{2}	5.8198×10^{2}	4.7912×10^{2}
2	2.8987×10^{2}	1.7401×10^{2}	6.2320×10^{1}	4.6059×10^{1}	5.1074×10^{1}	4.0937×10^{1}	5.5591×10^{1}	4.2298×10^{1}
3	2.7854×10^{2}	1.4884×10^{2}	2.1169×10^{1}	1.0964×10^{1}	6.3563×10^{0}	5.2184×10^{0}	6.4498×10^{0}	5.2912×10^{0}
4	2.9195×10^{2}	1.4232×10^{2}	1.7371×10^{1}	7.9965×10^{0}	1.7434×10^{0}	1.8839×10^{0}	1.7913×10^{0}	1.8252×10^{0}
5	2.9557×10^{2}	1.4212×10^{2}	1.6731×10^{1}	7.1024×10^{0}	$\mathbf{7.0345 \times 10^{-1}}$	$\mathbf{1.0348 \times 10^{0}}$	7.0913×10^{-1}	1.1003×10^{0}
6	2.9577×10^{2}	1.4142×10^{2}	1.6480×10^{1}	6.9003×10^{0}	2.3153×10^{-1}	2.7509×10^{-1}	2.2592×10^{-1}	2.6913×10^{-1}
7	2.9541×10^{2}	1.4120×10^{2}	1.6201×10^{1}	6.4680×10^{0}	1.1082×10^{-1}	1.7223×10^{-1}	1.2672×10^{-1}	1.7559×10^{-1}
8	3.0817×10^{2}	1.4282×10^{2}	1.6223×10^{1}	6.6127×10^{0}	7.9624×10^{-2}	1.5681×10^{-1}	8.0613×10^{-2}	1.6129×10^{-1}
9	3.2279×10^{2}	1.4431×10^{2}	1.6425×10^{1}	6.3947×10^{0}	6.5134×10^{-2}	1.1477×10^{-1}	6.5002×10^{-2}	1.1701×10^{-1}
10	3.2980×10^{2}	1.4455×10^{2}	1.6866×10^{1}	6.4134×10^{0}	1.2087×10^{-2}	2.1544×10^{-2}	1.1920×10^{-2}	2.1791×10^{-2}
15	3.2881×10^{2}	1.4465×10^{2}	1.6962×10^{1}	6.5556×10^{0}	2.2084×10^{-3}	1.5153×10^{-2}	2.2302×10^{-3}	1.5902×10^{-2}
20	3.2618×10^{2}	1.4466×10^{2}	1.7744×10^{1}	7.1142×10^{0}	1.6085×10^{-3}	1.1335×10^{-2}	1.6482×10^{-3}	1.1742×10^{-2}

$n_{\text{B}}^{1} = n_{\text{B}}^{2} = n_{\text{A}}^{1} = n_{\text{A}}^{2}$) and different numbers of hidden nodes, K. In comparison with the linear model (Table 3) and Wiener structure A (Table 4), Wiener structure B has significantly lower errors. For further comparisons, the third-order model containing five hidden nodes is chosen. The middle part of Fig. 6 compares the validation data set versus the output of Wiener model B. Unlike structure A, the model output signal is similar to that of the validation data, the differences are small.

Finally, Wiener structure C is considered. Training and validation errors of the model are given in Table 6 for models of different orders (where the order is defined as an integer number $n_{\text{B}}^{ij} = n_{\text{A}}^{i} = n_{\text{B}}^{3} = n_{\text{A}}^{3}$ for $i = 1, 2$, $j = 1, 2$) and different numbers of hidden nodes, K. In comparison with Wiener structures A and B (Tables 4, 5), structure C has significantly lower errors. Furthermore, there is a direct influence of the number of hidden nodes on model accuracy (the more hidden nodes, the lower the errors). It is interesting to notice that the third-order models are characterized by very similar errors as the fourth-order ones, whereas the first- and second-order structures are significantly worse. As a compromise between accuracy and complexity the third-order model containing five hidden nodes is chosen. The bottom part of Fig. 6 compares the validation data set versus the output of Wiener model C. It can be checked that in this case it is practically impossible to seen any differences between the validation data and the model output (which are present in the case of structures A and B).

The accuracy of the considered third-order linear model and all Wiener structures (with five hidden nodes) is compactly presented in Fig. 7 which depicts the relation between the validation data versus the

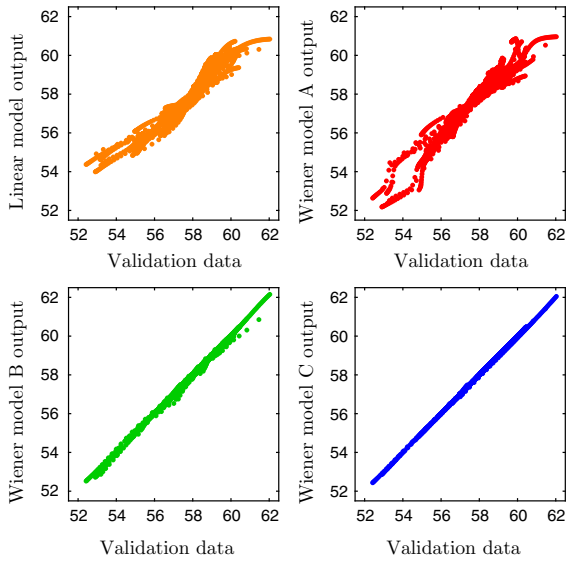

Fig. 7 Relation between the validation data versus the outputs of the chosen third-order linear model and the Wiener structures (structures A, B and C) containing $K = 5$ hidden nodes

model outputs. The linear model and Wiener structure A are imprecise, Wiener structure B gives much better results, whereas Wiener structure C is excellent. (The relation between the data and the model output forms a line the slope of which is 45 degrees.)

5.2 Model predictive control of PEM fuel cell

The objective of this subsection is to compare performance of MPC algorithms based on the models found in the previous subsection. The following MPC algorithms are compared:

1. The classical generalized predictive control (GPC) MPC algorithm [52]. The third-order linear model is used for prediction.
2. The fully fledged MPC-NO algorithm with nonlinear optimization repeated at each sampling instant. The Wiener models (A, B and C) are used for prediction.
3. The computationally efficient MPC schemes: the MPC-NPSL algorithm with simplified linearization and the MPC-NPLT one with trajectory linearization. Wiener model C is used in both MPC approaches. Both algorithms need solving online a quadratic optimization problem at each sampling instant.

All models (linear and nonlinear) used in all MPC algorithms are of the third order. As the simulated process the continuous-time fundamental model consists of Eqs. (2), (3), (6), (8)–(13) is used. In all simulations, the horizons of all compared MPC algorithms are the same: $N = 10$ and $N_u = 3$. The objective of all MPC algorithms is to control the process in such a way that the output, V, is close to the constant set point $V^{sp} = \bar{V}$ irrespective of the changes of the disturbance, I. The scenario of disturbance changes is

$$I(k) = \begin{cases} \bar{I} & \text{for } k < 5, \\ \bar{I} + 25 & \text{for } 5 \leq k < 40, \\ \bar{I} - 25 & \text{for } 40 \leq k < 80, \\ \bar{I} + 50 & \text{for } 80 \leq k < 120, \\ \bar{I} - 50 & \text{for } 120 \leq k < 160, \\ \bar{I} & \text{for } 160 \leq k < 200. \end{cases} \tag{62}$$

The magnitude of the manipulated variable is constrained: $u^{min} = 0.1$, $u^{max} = 2$.

At first, the GPC algorithm based on the linear model is considered. Simulation results for different values of the penalty factor λ are depicted in Fig. 8. Unfortunately, for the smallest value of that coefficient, i.e., for $\lambda = 1$, there are very strong oscillations of the process input and output variables. When the penalty coefficient is increased, for $\lambda = 25$ and $\lambda = 50$, the oscillations are damped, but the trajectory of the process output is slow. When $\lambda = 100$, no oscillations combined with a long rise time are observed. It means that the GPC algorithm is unable to compensate fast for changes of the disturbance.

Due to the underlying linear model, the GPC algorithm gives poor control results when applied for prediction. It seems to be straightforward to consider a nonlinear model in MPC. At first, Wiener structure A is used in the fully fledged MPC-NO algorithm. Although the MPC-NO algorithm is computationally too demanding to be used in practice, in simulations it shows whether or not the model may be used for long-range prediction in MPC. Figure 9 depicts simulation results of the MPC-NO algorithm based on Wiener model A for different values of the penalty factor λ. (The same values are used as in the case of the GPC algorithm.) Since Wiener model A is imprecise (Table 4 and Fig. 6), the obtained control quality is poor. For the smallest value $\lambda = 1$, there are some damped oscilla-

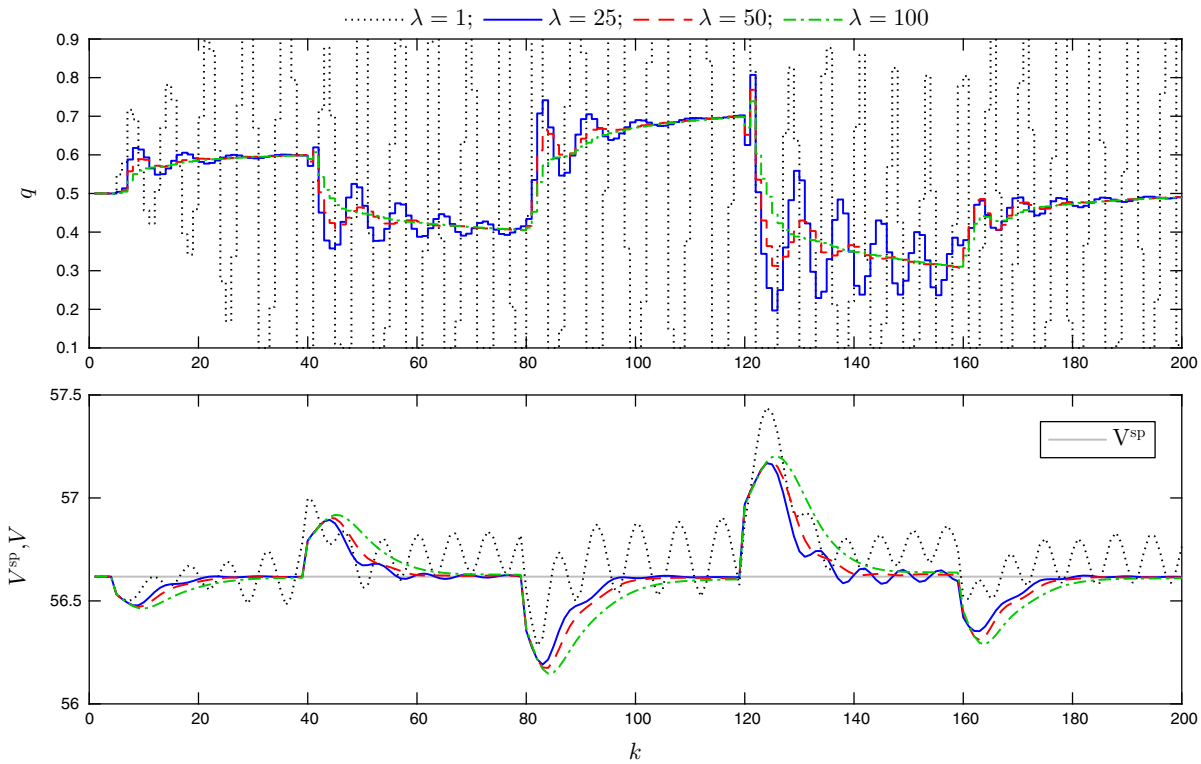

Fig. 8 Simulation results: the GPC algorithm for different values of the penalty factor λ

tions which may be eliminated when the penalty coefficient is increased. Unfortunately, it results in very slow trajectories, as those in the case of the GPC strategy. One may conclude that Wiener model A is not precise enough to be used in MPC.

Next, Wiener models B and C are discussed to be used in MPC. Simulations results of the MPC-NO algorithm based on these models are shown in Fig. 10. It is possible to formulate two observations. First of all, unlike the GPC algorithm and the MPC-NO strategy based on Wiener model A, when applied for prediction in MPC, both B and C Wiener models result in good control, i.e., it is possible to compensate fast for changes of the disturbance. Secondly, it should be noticed that the MPC-NO algorithm gives slightly better, but noticeable, results when Wiener model C is used. In this case, overshoot is smaller and the required set point is achieved faster. This results from the use of the more precise model C (instead of B) (Tables 5, 6).

Taking into account simulation results presented in Figs. 8, 9 and 10, it can be concluded that Wiener model C results in strongly improved control quality when used in the MPC-NO algorithm. It should be noted that the MPC-NO algorithm requires solving a nonlinear optimization problem at each sampling instant online. In order to reduce computational complexity, two alternatives are considered: the MPC-NPLT algorithm with trajectory linearization and the MPC-NPSL algorithm with simplified linearization. Both algorithms result in quadratic optimization problems; nonlinear optimization is not necessary. Figure 11 compares trajectories of the MPC-NO algorithm with those obtained in the MPC-NPLT and MPC-NPSL strategies. Two observations may be discussed. Firstly, the MPC-NPLT algorithm with trajectory linearization gives practically the same process trajectories as the "ideal" MPC-NO strategy; it is impossible to see any differences. It is a beneficial feature of the MPC-NPLT algorithm, since it is significantly less computationally demanding, but leads to the same control performance as the MPC-NO strategy. Secondly, the MPC-NPSL algorithm is stable, and it only gives slightly larger overshoot than the MPC-NO and MPC-NPLT algorithms. It should be noted that the MPC-NPSL algorithm uses for prediction a linear

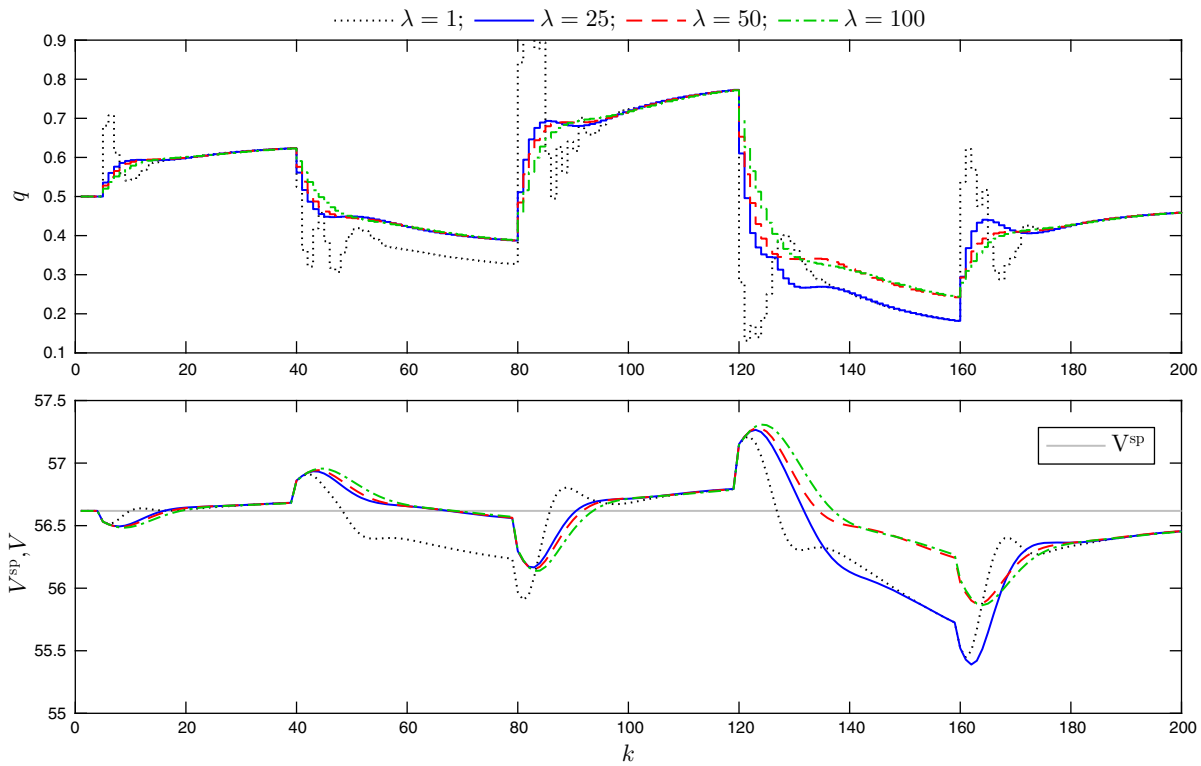

Fig. 9 Simulation results: the MPC-NO algorithm based on Wiener model A for different values of the penalty factor λ

approximation of the model which is obtained in a simple way, and quite complicated trajectory linearization is not necessary.

Figure 12 depicts simulation results of the three compared nonlinear MPC algorithms based on Wiener model C (MPC-NO, MPC-NPLT and MPC-NPSL), but now the increments of the manipulated variable are constrained, $\triangle u^{\max} = 0.1$. Due to the additional constraints, the manipulated variable does not change as quickly as in Fig. 11, but the trajectories of the process output are slower. In this case, the observations of algorithms' performance are the same as before, i.e., the MPC-NPLT algorithm gives the trajectories practically the same as the MPC-NO one, whereas the MPC-NPSL algorithm gives only slightly larger overshoot.

To further compare MPC algorithms whose trajectories are depicted in Figs. 11 and 12, two performance indices are calculated after completing simulations. The first one

$$E_1 = \sum_{k=1}^{200} (V^{\mathrm{sp}}(k) - V(k))^2, \tag{63}$$

measures the sum of squared differences between the required set point, $V^{\mathrm{sp}}(k)$, and the actual process output, $V(k)$, for the whole simulation horizon. The second one

$$E_2 = \sum_{k=1}^{200} (V^{\mathrm{MPC-NO}}(k) - V(k))^2, \tag{64}$$

measures the sum of squared differences between the process output controlled by the "ideal" MPC-NO algorithm, $V^{\mathrm{MPC-NO}}(k)$, and the process output controlled by a compared MPC algorithm, $V(k)$. The obtained numerical values of the performance indices E_1 and E_2 are given in Table 7. In general, the values of E_1 for the MPC-NO and MPC-NPLT algorithms are the same, which indicates that the measure E_2 for the MPC-NPLT algorithm is low, close to 0. When the MPC-NO algorithm is compared with the MPC-NPSL one, there are much noticeable differences, but still leading to control accuracy very similar to that possible when the MPC-NO and MPC-NPLT strategies are used. Two cases are considered (corresponding to Figs. 11, 12):

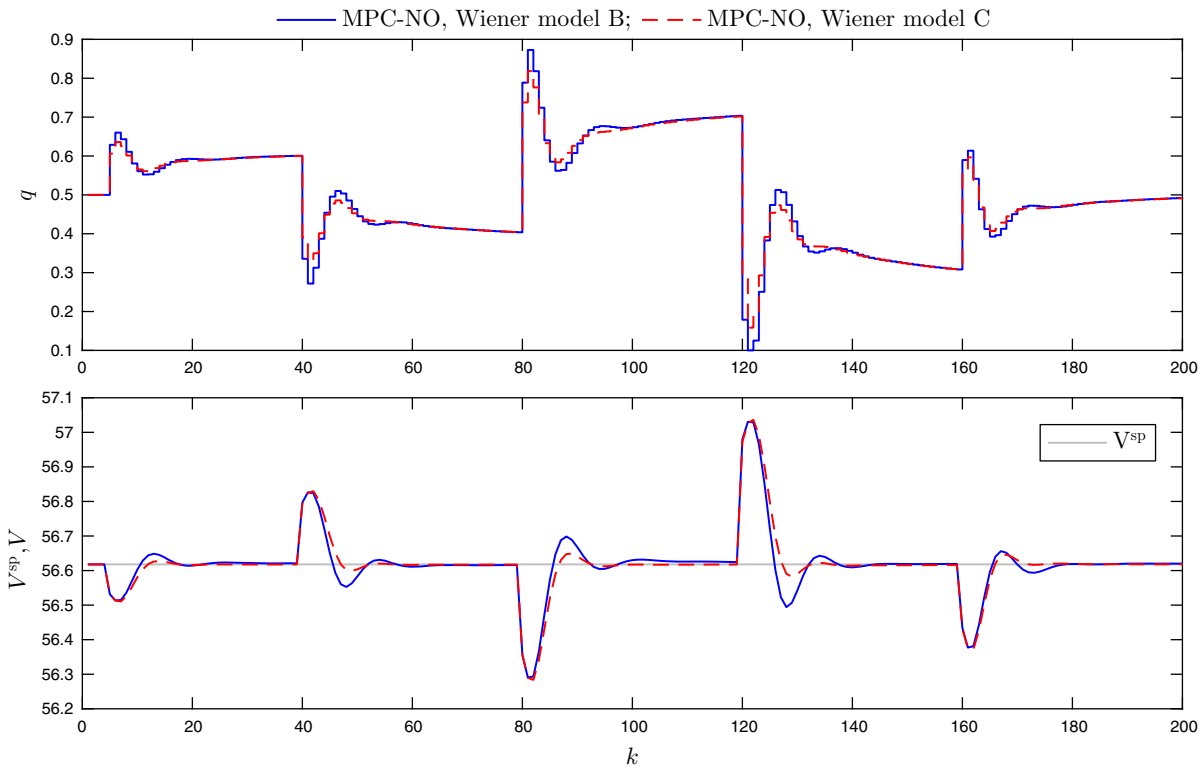

Fig. 10 Simulation results: the MPC-NO algorithm based on Wiener models B and C, $\lambda = 1$

when the rate of change of the manipulated variable is constrained or not. When the rate constraints are present, all trajectories (of the process input and output) are slower. Table 7 additionally specifies calculation time (scaled) of the MPC algorithms. Two general observations may be made. Firstly, the MPC-NPSL and MPC-NPLT algorithms are many times computationally efficient in comparison with the MPC-NO one. The MPC-NPSL scheme is somehow less demanding than the MPC-NPLT one, but this difference is not big since computational complexity is mostly influenced by the quadratic optimization subroutine. Secondly, introduction of the additional constraints imposed on the rate of change of the manipulated variable "helps" the optimization routine to slightly faster find the solution.

6 Conclusions

The PEM fuel cell is a nonlinear dynamic system. A linear model cannot describe process behavior precisely. Moreover, when such a model is used in MPC, obtained control accuracy is not acceptable.

In this work, effectiveness of three Wiener models of the PEM fuel cell is discussed. In all models, the nonlinear steady-state block is represented by a neural network, whereas the linear dynamic part is different. The model consisting of three dynamic blocks and a neural network with five hidden nodes is chosen for control.

The second contribution of this work is the development of two computationally efficient MPC algorithms for the PEM process. In both MPC algorithms, computationally not complicated quadratic optimization is used online, and nonlinear optimization is not necessary. In the first MPC algorithm, a time-varying linear approximation of the model is used for prediction, whereas in the second one a linear approximation of the predicted process trajectory is calculated repeatedly online. The MPC-NPSL algorithm with online simple model linearization gives very good results, very similar to those possible in the complex algorithm using trajectory linearization and computationally demanding MPC with online nonlinear optimization. It is important to note the fact that model linearization in MPC

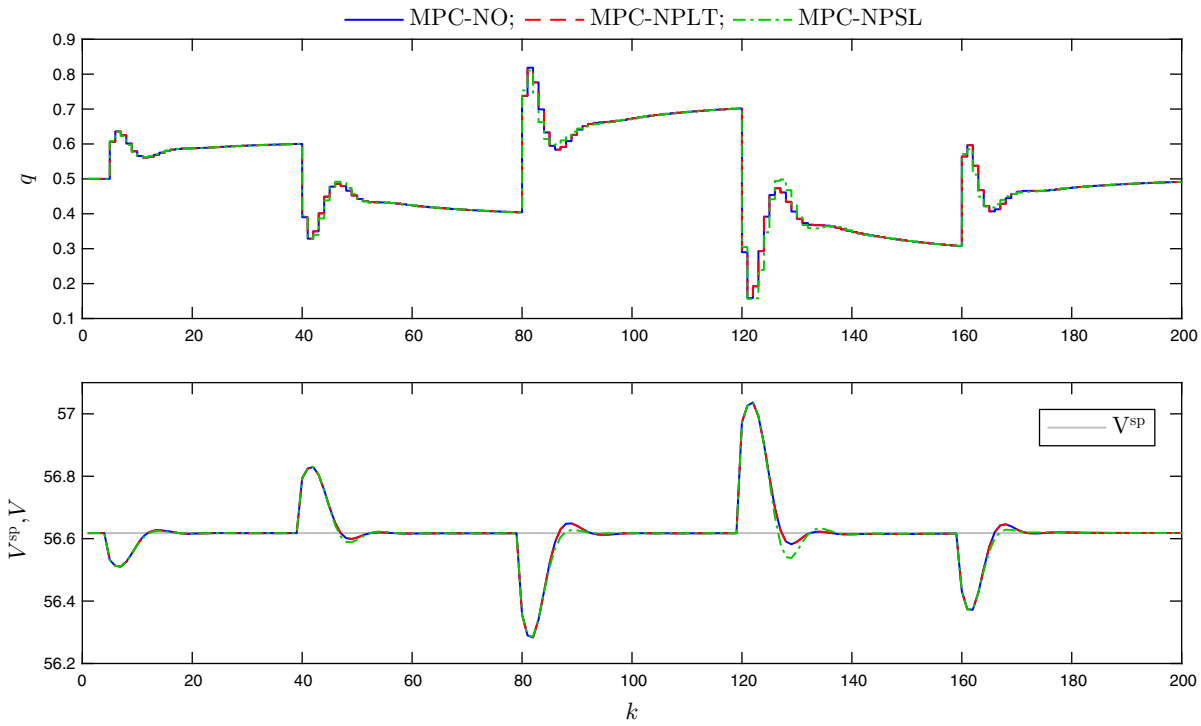

Fig. 11 Simulation results: the MPC-NO, MPC-NPLT and MPC-NPSL algorithms based on Wiener model C, $\lambda = 1$

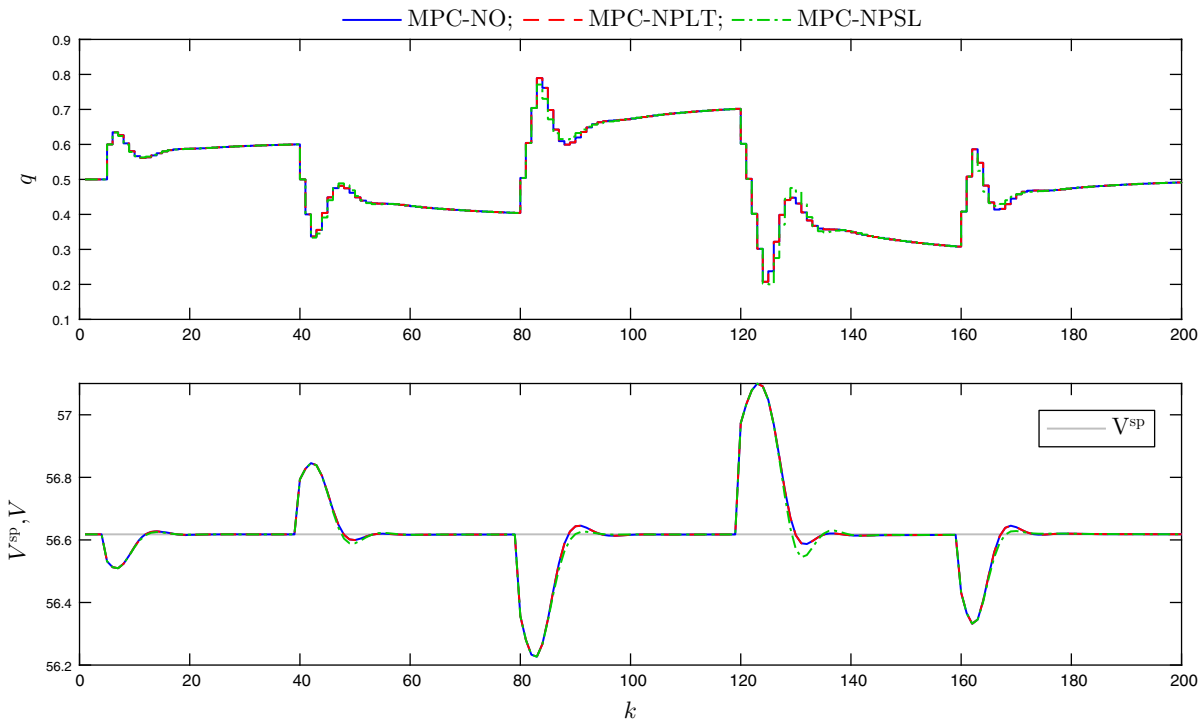

Fig. 12 Simulation results: the MPC-NO, MPC-NPLT and MPC-NPSL algorithms based on Wiener model C; the increments of the manipulated variable are constrained, $\triangle u^{\max} = 0.1$, $\lambda = 1$

Table 7 Comparison of control performance criteria, E_1 and E_2, as well as calculation time for MPC-NPSL, MPC-NPLT and MPC-NO algorithms based on Wiener model C; $\lambda = 1$

Algorithm	$\triangle u^{\max}$	E_1	E_2	Calculation time (%)
MPC-NPSL	No	1.6561×10^0	2.8791×10^{-2}	7.84
MPC-NPLT	No	1.5779×10^0	2.6979×10^{-10}	9.13
MPC-NO	No	1.5779×10^0	0	100.00
MPC-NPSL	Yes	2.7934×10^0	2.5633×10^{-2}	7.36
MPC-NPLT	Yes	2.6894×10^0	6.7151×10^{-11}	8.44
MPC-NO	Yes	2.6894×10^0	0	97.89

is performed online in a simple way, which is possible because of the specialized serial structure of the Wiener model. Moreover, as a nonlinear block a neural network is used which leads to good approximation accuracy and easiness of model utilization in MPC. The MPC-NPLT algorithm with trajectory linearization gives practically the same control accuracy as that possible in the MPC-NO approach with nonlinear optimization, but requires more demanding calculations. (Trajectory linearization is more demanding than model linearization.)

Although the Wiener models and the MPC algorithms are developed for a particular type of the PEM, one may consider alternative ones, e.g., [4,34–38]. The described MPC algorithms may be used in complex power management optimization systems for the PEM fuel cell [53].

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Larminie, J., Dicks, A.: Fuel Cell Systems Explained. Wiley, Hoboken (2000)
2. Barbir, F.: PEM Fuel Cells: Theory and Practice. Academic Press, London (2013)
3. Özbek, M.: Modeling, Simulation, and Concept Studies of a Fuel Cell Hybrid Electric Vehicle Powertrain. Ph.D. Thesis, University of Duisburg-Essen, Duisburg (2010)
4. Pukrushpan, J.T., Stefanopoulou, A.G., Peng, H.: Control of Fuel Cell Power Systems: Principles, Modeling. Analysis and Feedback Design. Springer, London (2004)
5. Kunusch, C., Puleston, P., Mayosky, M.: Sliding-Mode Control of PEM Fuel Cells. Springer, London (2012)
6. Baroud, Z., Benmiloud, M., Benalia, A., Ocampo-Martinez, C.: Novel hybrid fuzzy-PID control scheme for air supply in PEM fuel-cell-based systems. Int. J. Hydrogen Energy **42**, 10435–10447 (2017)
7. Ou, K., Wang, Y.-X., Li, Z.-Z., Shen, Y.-D., Xuan, D.-J.: Feedforward fuzzy-PID control for air flow regulation of PEM fuel cell system. Int. J. Hydrogen Energy **40**, 11686–11695 (2015)
8. Damoura, C., Benne, M., Lebreton, C., Deseure, J., Grondin-Perez, B.: Real-time implementation of a neural model-based self-tuning PID strategy for oxygen stoichiometry control in PEM fuel cell. Int. J. Hydrogen Energy **39**, 12819–12825 (2014)
9. Beirami, H., Shabestari, A.Z., Zerafat, M.M.: Optimal PID plus fuzzy controller design for a PEM fuel cell air feed system using the self-adaptive differential evolution algorithm. Int. J. Hydrogen Energy **40**, 9422–9434 (2015)
10. Hong, L., Chen, J., Liu, Z., Huang, L., Wu, Z.: A nonlinear control strategy for fuel delivery in PEM fuel cells considering nitrogen permeation. Int. J. Hydrogen Energy **42**, 1565–1576 (2017)
11. Meidanshahi, V., Karimi, G.: Dynamic modeling, optimization and control of power density in a PEM fuel cell. Appl. Energy **93**, 98–105 (2012)
12. Özbek, M., Wang, S., Marx, M., Söffker, D.: Modeling and control of a PEM fuel cell system: a practical study based on experimental defined component behavior. J. Process Control **23**, 282–293 (2013)
13. Shahiri, M., Ranjbar, A., Karami, M.R., Ghaderi, R.: New tuning design schemes of fractional complex-order PI controller. Nonlinear Dyn. **84**, 1813–1835 (2016)
14. Shahiri, M., Ranjbar, A., Karami, M.R., Ghaderi, R.: Robust control of nonlinear PEMFC against uncertainty using fractional complex order control. Nonlinear Dyn. **80**, 1785–1800 (2015)
15. Camacho, E.F., Bordons, C.: Model Predictive Control. Springer, London (1999)
16. Maciejowski, J.M.: Predictive Control with Constraints. Prentice Hall, Englewood Cliffs (2002)
17. Tatjewski, P.: Advanced Control of Industrial Processes, Structures and Algorithms. Springer, London (2007)

18. Hähnel, C., Aul, V., Horn, J.: Power control for efficient operation of a PEM fuel cell system by nonlinear model predictive control. IFAC-PapersOnLine **48**, 174–179 (2015)
19. Rosanas-Boeta, N., Ocampo-Martinez, C., Kunusch, C.: On the anode pressure and humidity regulation in PEM fuel cells: a nonlinear predictive control approach. IFAC-PapersOnLine **48**, 434–439 (2015)
20. Schultze, M., Horn, J.: Modeling, state estimation and nonlinear model predictive control of cathode exhaust gas mass flow for PEM fuel cells. Control Eng. Pract. **43**, 76–86 (2016)
21. Ziogou, C., Papadopoulou, S., Georgiadis, M.C., Voutetakis, S.: On-line nonlinear model predictive control of a PEM fuel cell system. J. Process Control **23**, 483–492 (2013)
22. Barzegari, M.M., Alizadeh, E., Pahnabi, A.H.: Grey-box modeling and model predictive control for cascade-type PEMFC. Energy **127**, 611–622 (2017)
23. Panos, C., Kouramas, K.I., Georgiadis, M.C., Pistikopoulos, E.N.: Modelling and explicit model predictive control for PEM fuel cell systems. Chem. Eng. Sci. **67**, 15–25 (2012)
24. Janczak, A.: Identification of Nonlinear Systems Using Neural Networks and Polynomial Models. A Block-Oriented Approach. Lecture Notes in Control and Information Sciences, vol. 310. Springer, Berlin (2004)
25. Haykin, S.: Neural Networks-A Comprehensive Foundation. Prentice Hall, Upper Saddle River (2008)
26. Ławryńczuk, M.: Computationally Efficient Model Predictive Control Algorithms: A Neural Network Approach. Studies in Systems, Decision and Control, vol. 3. Springer, Cham (2014)
27. Zhu, B.: Nonlinear adaptive neural network control for a model-scaled unmanned helicopter. Nonlinear Dyn. **78**, 1695–1708 (2014)
28. Guo, J., Luo, Y., Li, K.: Adaptive neural-network sliding mode cascade architecture of longitudinal tracking control for unmanned vehicles. Nonlinear Dyn. **87**, 2497–2510 (2017)
29. Eski, İ., Temürlenk, A.: Design of neural network-based control systems for active steering system. Nonlinear Dyn. **73**, 1443–1454 (2013)
30. Chen, D., Zhang, Y., Li, S.: Tracking control of robot manipulators with unknown models: a Jacobian-matrix-adaption method. IEEE Trans. Ind. Inform. **14**, 3044–3053 (2018)
31. Chen, D., Zhang, Y.: Robust zeroing neural-dynamics and its time-varying disturbances suppression model applied to mobile robot manipulators. IEEE Trans. Neural Netw. Learing Syst. **29**, 4385–4397 (2018)
32. Zhang, Y., Chen, K., Tan, H.-Z.: Performance analysis of gradient neural network exploited for online time-varying matrix inversion. IEEE Trans. Autom. Control **54**, 1940–1945 (2009)
33. Yan, Z., Wang, J.: Nonlinear model predictive control based on collective neurodynamic optimization. IEEE Trans. Neural Netw. Learing Syst. **26**, 840–850 (2015)
34. Benchouia, N.E., Derghal, A., Mahmah, B., Madi, B., Khochemane, L., Aoul, L.H.: An adaptive fuzzy logic controller (AFLC) for PEMFC fuel cell. Int. J. Hydrog. Energy **40**, 13806–13819 (2015)
35. Barzegari, M.M., Dardel, M., Alizadeh, E., Ramiar, A.: Reduced-order model of cascade-type PEM fuel cell stack with integrated humidifiers and water separators. Energy **113**, 683–692 (2016)
36. Hatziadoniu, C.J., Lobo, A.A., Pourboghrat, F., Daneshdoost, M.: A simplified dynamic model of grid-connected fuel-cell generators. IEEE Trans. Power Deliv. **17**, 467–473 (2002)
37. Suh, K.W.: Modeling, Analysis and control of fuel cell hybrid power systems. Ph.D. Thesis, University of Michigan, Ann Arbor (2016)
38. Talj, R.J., Hissel, D., Ortega, R., Becherif, M., Hilairet, M.: Experimental validation of a PEM fuel-cell reduced-order model and a moto-compressor higher order sliding-mode control. IEEE Trans. Ind. Electron. **57**, 1906–1913 (2010)
39. Uzunoglu, M., Alam, M.S.: Dynamic modeling, design and simulation of a combined PEM fuel cell and ultracapacitor system for stand-alone residential applications. IEEE Trans. Energy Conv. **21**, 767–775 (2006)
40. Uzunoglu, M., Alam, M.S.: Dynamic modeling, design and simulation of a PEM fuel cell/ultra-capacitor hybrid system for vehicular applications. Energy Conv. Manag. **48**, 1544–1553 (2009)
41. Erdinc, O., Vural, B., Uzunoglu, M., Ates, Y.: Modeling and analysis of an FC/UC hybrid vehicular power system using a wavelet-fuzzy logic based load sharing and control algorithm. Int. J. Hydrog. Energy **34**, 5223–5233 (2009)
42. Kisacikoglu, M.C., Uzunoglu, M., Alam, M.S.: Load sharing using fuzzy logic control in a fuel cell/ultracapacitor hybrid vehicle. Int. J. Hydrog. Energy **34**, 1497–1507 (2009)
43. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, Berlin (2006)
44. Qin, S.J., Badgwell, T.A.: A survey of industrial model predictive control technology. Control Eng. Pract. **11**, 733–764 (2003)
45. Zhang, J., Chin, K.-S., Ławryńczuk, M.: Nonlinear model predictive control based on piecewise linear Hammerstein models. Nonlinear Dyn. **92**, 1001–1021 (2018)
46. Marami, B., Haeri, M.: Implementation of MPC as an AQM controller. Comput. Commun. **33**, 227–239 (2010)
47. Sardarmehni, T., Rahmani, R., Menhaj, M.B.: Robust control of wheel slip in anti-lock brake system of automobiles. Nonlinear Dyn. **76**, 125–138 (2014)
48. Yue, M., Hou, X., Gao, R., Chen, J.: Trajectory tracking control for tractor-trailer vehicles: a coordinated control approach. Nonlinear Dyn. **91**, 1061–1074 (2018)
49. Wu, Z., Xia, X., Zhu, B.: Model predictive control for improving operational efficiency of overhead cranes. Nonlinear Dyn. **79**, 2639–2657 (2015)
50. Gao, J., Puguo, W., Li, T., Proctor, A.: Optimization-based model reference adaptive control for dynamic positioning of a fully actuated underwater vehicle. Nonlinear Dyn. **87**, 2611–2623 (2017)
51. Longge, Z., Xiangjie, L.: The synchronization between two discrete-time chaotic systems using active robust model predictive control. Nonlinear Dyn. **74**, 905–910 (2013)
52. Clarke, D.W., Mohtadi, C.: Properties of generalized predictive control. Automatica **25**, 859–875 (1989)
53. Moulik, B., Söffker, D.: Optimal rule-based power management for online, real-time applications in HEVs with multiple sources and objectives: a review. Energies **8**, 9049–9063 (2015)