

Identification of Wiener models for dynamic and steady-state performance with application to solid oxide fuel cell

Maciej Ławryńczuk 

Institute of Control and Computation Engineering, Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland

Correspondence

Maciej Ławryńczuk, Institute of Control and Computation Engineering, Faculty of Electronics and Information Technology, Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland.
Email: M.Lawrynczuk@ia.pw.edu.pl

Abstract

This work is concerned with identification of Wiener models (a linear dynamic part connected in series with a nonlinear dynamic one). A neural network with one hidden layer is used as the nonlinear block of the model, two network configurations are considered. For model identification three algorithms are described. In the first case model accuracy only in transient conditions is considered, only the dynamic data is used for model training. In the next two algorithms model accuracy in both transient and steady-state conditions is considered, dynamic and steady-state data sets are used. The steady-state model errors are taken into account by an additional term in the minimized cost-function or by additional inequality constraints. For comparison of discussed algorithms and model structures, identification of a Wiener model of a solid oxide fuel cell (SOFC) process is considered. It is shown that the best results are obtained by the algorithm 2 which minimizes at the same time both dynamic and steady-state model errors, additional constraints used in the algorithm 3 are computationally quite demanding.

KEYWORDS

model identification, neural networks, solid oxide fuel cell, Wiener model

1 | INTRODUCTION

Cascade Wiener models are composed of a linear dynamic part connected in series with a nonlinear steady-state block [1]. Such structures turn out to be very precise models of numerous dynamic systems, e.g. chemical reactors [2], distillation columns [3], separation processes [4], hydraulic systems [5], and even biological systems [6]. Unlike numerous black-box models, e.g. neural

networks, Wiener models typically are characterized by a smaller number of parameters. Moreover, they may be very efficiently used in control algorithms, e.g. [7], and in fault detection, e.g. [1]. Unlike black-box models, block-oriented models have a clear physical interpretation, since the model gain is calculated easily as a multiplication of the constant gain of the linear dynamic block and the gain of the nonlinear steady-state block, which depends on the current operating point.

For identification of Wiener models different approaches have been proposed: correlation methods [8], non-parametric regression methods [9,10] and nonlinear optimization methods [4]. An excellent review of identification algorithms and applications of Wiener systems in process modelling and fault detection is given in [1].

During model identification model parameters are determined in such a way that the difference between model output and a data set is minimized. Typically, dynamic model performance is optimized, i.e. model accuracy only in transient conditions is considered. Unfortunately, such an approach may result in a model whose steady-state properties are far from the properties of the process, as demonstrated in this work. This work is concerned with identification of Wiener models for dynamic and steady-state performance. Two multiple-input single-output (MISO) Wiener structures are considered (structures A and B) with different configurations of the nonlinear part in which a neural network with one hidden layer is used. For model identification three algorithms are described (algorithms 1, 2 and 3). In the first case model accuracy only in transient conditions is considered, only the dynamic data is used for model training. In the second algorithm model accuracy in both transient and steady-state conditions is optimized, dynamic and steady-state data sets are used. The steady-state model errors are taken into account by an additional term in the minimized cost-function. In the third approach model accuracy only in transient conditions is optimized, but additional inequality constraints are used to obtain precise steady-state model properties.

For comparison of the discussed algorithms and model structures, identification of a Wiener model of a solid oxide fuel cell (SOFC) process is considered. Fuel cells are very promising sources of energy since they are able to directly convert the chemical energy stored in hydrocarbon fuels (e.g. hydrogen) into electrical energy [11]. Among different types of fuel cells, SOFCs [11,12] may be distinguished since they have many advantages: high electrical efficiency, fuel flexibility, low emission, quiet operation and relatively low cost. SOFCs are expected to become sound alternatives to conventional power generation schemes not only for domestic, but also for commercial and industrial sectors. For economically efficient and safe operation of SOFCs precise control is necessary. Recently, a number of model-based control methods have been developed for different types of fuel cells: observer-based state feedback control [13], optimal control [14], extremum seeking control cooperating with model reference adaptive control [15], adaptive control [16], model predictive control (MPC) [17-19] and fractional order controllers [20]. A precise model of the process is necessary for developing all these algorithms.

2 | STRUCTURES OF WIENER MODELS

In this work identification of MISO dynamic processes is considered with n_u input variables $u_1, \dots, u_{n_u}$ and one output y .

2.1 | Structure A

The first structure of the Wiener model (the structure A) is depicted in Figure 1. It consists of n_u single-input single-output (SISO) linear dynamic blocks connected in series with a SISO nonlinear steady-state block, $v_1 \dots, v_{n_u}$ denote outputs of the consecutive linear blocks. The linear blocks of the model are described by the following difference equations:

$$v_j(k) = \sum_{i=\tau^j}^{n_B^j} b_i^j u_j(k-i) - \sum_{i=1}^{n_A^j} a_i^j v_j(k-i) \quad (1)$$

where $j = 1, \dots, n_u$. The integers n_A^j, n_B^j, τ^j define the order of dynamics, $\tau^j \leq n_B^j, j = 1, \dots, n_u$. The constant parameters of the linear dynamic blocks are denoted by the real numbers a_i^j ($i = 1, \dots, n_A^j, j = 1, \dots, n_u$) and b_i^j ($i = \tau^j, \dots, n_B^j, j = 1, \dots, n_u$). The sum of output signals of the SISO linear dynamic blocks is taken as the input v of the steady-state one. Hence, the nonlinear steady-state part of the model is described by the general equation

$$y(k) = g\left(\sum_{j=1}^{n_u} v_j(k)\right) \quad (2)$$

As the differentiable function $g : \mathbb{R} \rightarrow \mathbb{R}$ a neural network with one input, one hidden layer with K units and one linear output is used [21]. Its output is

$$y(k) = w_0^2 + \sum_{l=1}^K w_l^2 \varphi\left(w_{l,0}^1 + w_{l,1}^1 \sum_{j=1}^{n_u} v_j(k)\right) \quad (3)$$

where $\varphi : \mathbb{R} \rightarrow \mathbb{R}$ is a nonlinear transfer function (e.g. hyperbolic tangent). The weights of the network are denoted by $w_{l,m}^1, l = 1, \dots, K, m = 0, 1$ and $w_l^2, l = 0, \dots, K$, for the first and the second layers, respectively. The overall number of weights is $3K + 1$.

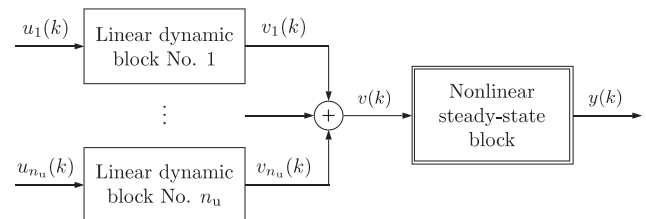


FIGURE 1 The structure A of the Wiener model

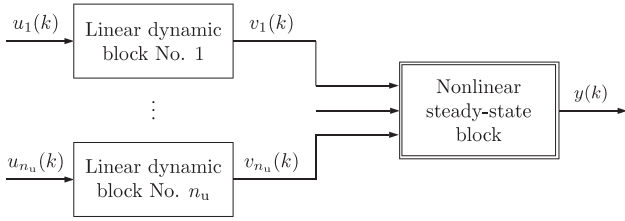


FIGURE 2 The structure B of the Wiener model

2.2 | Structure B

The second structure of the Wiener model (the structure B) is depicted in Figure 2. Similarly to the structure A, it also consists of n_u SISO linear dynamic blocks described by Equation 1. In contrast to the previous structure, a MISO nonlinear steady-state block is used. It is described by the general equation

$$y(k) = g(v_1(k), \dots, v_{n_u}(k)) \quad (4)$$

As the differentiable function $g : \mathbb{R} \rightarrow \mathbb{R}$ a neural network with n_u inputs, one hidden layer with K units and one output is used. Its output is

$$y(k) = w_0^2 + \sum_{l=1}^K w_l^2 \varphi \left(w_{l,0}^1 + \sum_{j=1}^{n_u} w_{l,j}^1 v_j(k) \right) \quad (5)$$

The weights of the network are denoted by $w_{l,m}^1$, $l = 1, \dots, K$, $m = 0, \dots, n_u$ and w_l^2 , $l = 0, \dots, K$, for the first and the second layers, respectively. The difference between the structure A and B is the fact that in the first case the neural network has only one input ($v = v_1 + \dots + v_{n_u}$) whereas in the second one there are n_u inputs ($v_1, \dots, v_{n_u}$). The overall number of weights is $K(n_u + 1) + K + 1$.

3 | IDENTIFICATION OF WIENER MODELS

During model identification the values of b_i^j , $w_{l,m}^1$ and w_l^2 are directly calculated from an optimization problem the objective of which is to minimize model errors. In order to enforce model stability poles of transfer functions corresponding to the discrete-time difference equations (1) are determined rather than directly the coefficients α_i^j . Let the poles be denoted by r_i^j for $i = 1, \dots, n_A^j$, $j = 1, \dots, n_u$. The relation between the poles and the coefficients α_i^j are described by the general functions

$$\alpha_i^j = \alpha_i^j(r_1^j, \dots, r_{n_A^j}^j) \quad (6)$$

where $i = 1, \dots, n_A^j$, $j = 1, \dots, n_u$. The coefficients α_i^j may be easily calculated for the poles r_i^j . For the first-order dynamic blocks (i.e. $n_A^j = 1$, $j = 1, \dots, n_u$)

$$\alpha_1^j = \alpha_1^j(r_1^j) = r_1^j \quad (7)$$

for the second-order blocks (i.e. $n_A^j = 2$, $j = 1, \dots, n_u$)

$$\begin{aligned} \alpha_1^j &= \alpha_1^j(r_1^j, r_2^j) = -r_1^j - r_2^j \\ \alpha_2^j &= \alpha_2^j(r_1^j, r_2^j) = r_1^j r_2^j \end{aligned} \quad (8)$$

etc. All model identification constrained optimization problems are solved by means of the sequential quadratic programming (SQP) algorithm [22]. Analytical derivatives of the minimized objective function are provided for the SQP solver for all algorithms.

3.1 | Algorithm 1

In the first identification algorithm the parameters of the model are determined considering only model accuracy in transient conditions, using dynamic data. The objective of model identification is to find all parameters by minimization of the model error which is defined as

$$E^{\text{dyn}} = \frac{1}{n_{\text{dyn}}} \sum_{k=1}^{n_{\text{dyn}}} (y_{\text{mod}}(k) - y(k))^2 \quad (9)$$

The symbols $y_{\text{mod}}(k)$ and $y(k)$ denote the model output and the process output (i.e. a data sample from the available data set) for the sampling instant k , respectively, the number of dynamic training patterns is denoted by n_{dyn} . Model parameters are determined from the following constrained optimization task

$$\begin{aligned} \min_{r_i^j, b_i^j, w_{l,m}^1, w_l^2} E^{\text{dyn}} \\ \text{subject to} \\ -1 + \delta \leq r_i^j \leq 1 - \delta \end{aligned} \quad (10)$$

During calculation of the minimized cost function, E^{dyn} , the coefficients α_i^j are found from the poles r_i^j using Equations (7) or (8). The constraints $-1 + \delta \leq r_i^j \leq 1 - \delta$ are used to enforce stability of the linear dynamic blocks, where δ is a small positive number, e.g. 10^{-4} . The coefficients b_i^j and the weights of the neural network are not constrained.

3.1.1 | Model structure A

The gradients of the objective function (9) with respect to the poles are

$$\frac{\partial E^{\text{dyn}}}{\partial r_i^j} = \frac{2}{n_{\text{dyn}}} \sum_{k=1}^{n_{\text{dyn}}} (y_{\text{mod}}(k) - y(k)) \frac{\partial y_{\text{mod}}(k)}{\partial r_i^j} \quad (11)$$

where $i = 1, \dots, n_A^j$, $j = 1, \dots, n_u$. The gradients with respect to the coefficients b_i^j are

$$\frac{\partial E^{\text{dyn}}}{\partial b_i^j} = \frac{2}{n_{\text{dyn}}} \sum_{k=1}^{n_{\text{dyn}}} (y_{\text{mod}}(k) - y(k)) \frac{\partial y_{\text{mod}}(k)}{\partial b_i^j} \quad (12)$$

where $i = 1, \dots, n_B^j, j = 1, \dots, n_u$. The gradients with respect to the weights $w_{l,m}^1$ are

$$\frac{\partial E^{\text{dyn}}}{\partial w_{l,m}^1} = \frac{2}{n_{\text{dyn}}} \sum_{k=1}^{n_{\text{dyn}}} (y_{\text{mod}}(k) - y(k)) \frac{\partial y_{\text{mod}}(k)}{\partial w_{l,m}^1} \quad (13)$$

where $l = 1, \dots, K, m = 0, 1$. The gradients with respect to the weights w_l^2 are

$$\frac{\partial E^{\text{dyn}}}{\partial w_l^2} = \frac{2}{n_{\text{dyn}}} \sum_{k=1}^{n_{\text{dyn}}} (y_{\text{mod}}(k) - y(k)) \frac{\partial y_{\text{mod}}(k)}{\partial w_l^2} \quad (14)$$

where $l = 0, \dots, K$. The partial derivative used in the right side of Equation (11) is derived by differentiating Equations (1), (3) and (6)

$$\begin{aligned} \frac{\partial y_{\text{mod}}(k)}{\partial r_i^j} &= - \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} w_{l,1}^1 \\ &\quad \sum_{n=1}^{n_A^j} \frac{\partial \alpha_n^j(r_1^j, \dots, r_{n_A^j}^j)}{\partial r_i^j} v_j(k-n) \end{aligned} \quad (15)$$

where the sum of inputs of the l^{th} hidden node is

$$z_l(k) = w_{l,0}^1 + w_{l,1}^1 \sum_{j=1}^{n_u} v_j(k) \quad (16)$$

for all $l = 1, \dots, K$. If hyperbolic tangent is used as nonlinear transfer functions φ in the hidden layer of the neural network, one has

$$\frac{d\varphi(z_l(k))}{dz_l(k)} = 1 - \tanh^2(z_l(k)) \quad (17)$$

The derivatives of the function $\alpha_i^j(r_1^j, \dots, r_{n_A^j}^j)$ (Equation 6) depend on the dynamic order defined by the integer numbers n_A^j . If $n_A^j = 1$, from Equation (7) one has

$$\frac{\partial \alpha_1^j(r_1^j)}{\partial r_1^j} = 1 \quad (18)$$

If $n_A^j = 2$, from Equation (8) one has

$$\begin{aligned} \frac{\partial \alpha_1^j(r_1^j, r_2^j)}{\partial r_n^j} &= -1, \quad n = 1, 2 \\ \frac{\partial \alpha_2^j(r_1^j, r_2^j)}{\partial r_1^j} &= r_2^j, \quad \frac{\partial \alpha_2^j(r_1^j, r_2^j)}{\partial r_2^j} = r_1^j \end{aligned} \quad (19)$$

The partial derivatives used in the right sides of Equations (12), (13) and (14) are derived by differentiating Equations (1) and (3), which gives

$$\frac{\partial y_{\text{mod}}(k)}{\partial b_i^j} = \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} w_{l,1}^1 u_j(k-i) \quad (20)$$

and

$$\frac{\partial y_{\text{mod}}(k)}{\partial w_{l,m}^1} = \begin{cases} w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} & \text{if } m = 0 \\ w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} \sum_{j=1}^{n_u} v_j(k) & \text{if } m = 1 \end{cases} \quad (21)$$

and

$$\frac{\partial y_{\text{mod}}(k)}{\partial w_l^2} = \begin{cases} 1 & \text{if } l = 0 \\ \varphi(z_l(k)) & \text{if } l = 1, \dots, K \end{cases} \quad (22)$$

3.1.2 | Model structure B

The general formulas for gradient calculation, i.e. Equations (11)–(13) and (14), are the same for both model structures, the only minor difference is that in the case of the structure B the neural network has n_u inputs. It means that the partial derivative used in the right side of Equation (11) is derived by differentiating Equations (1), (5) and (6)

$$\begin{aligned} \frac{\partial y_{\text{mod}}(k)}{\partial r_i^j} &= - \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} w_{l,j}^1 \\ &\quad \sum_{n=1}^{n_A^j} \frac{\partial \alpha_n^j(r_1^j, \dots, r_{n_A^j}^j)}{\partial r_i^j} v_j(k-n) \end{aligned} \quad (23)$$

where the sum of inputs of the l^{th} hidden node is

$$z_l(k) = w_{l,0}^1 + \sum_{j=1}^{n_u} w_{l,j}^1 v_j(k) \quad (24)$$

for all $l = 1, \dots, K$. The derivatives of the function $\alpha_i^j(r_1^j, \dots, r_{n_A^j}^j)$ (Equation 6) are calculated in the same way it is done in the model structure A, i.e. from Equations (18) or (19). The partial derivatives used in the right sides of Equations (12) and (13) are derived by differentiating Equations (1) and (5), which gives

$$\frac{\partial y_{\text{mod}}(k)}{\partial b_i^j} = \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} w_{l,j}^1 u_j(k-i) \quad (25)$$

and

$$\frac{\partial y_{\text{mod}}(k)}{\partial w_{l,m}^1} = \begin{cases} w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} & \text{if } m = 0 \\ w_l^2 \frac{d\varphi(z_l(k))}{dz_l(k)} v_m(k) & \text{if } m = 1, \dots, n_u \end{cases} \quad (26)$$

One must remember that for the model structure B the gradients with respect to the weights $w_{l,m}^1$ must be calculated for all $l = 1, \dots, K, m = 0, \dots, n_u$. Finally, the partial derivative in the right side of (14) is calculated from Equation (22), but remembering that the signals $z_l(k)$ are defined by Equation (24) for the model structure B.

3.2 | Algorithm 2

In the second identification algorithm model accuracy not only in transient conditions defined by the error E^{dyn} (Equation 9), but also in steady-state conditions is considered. It means that two training data sets are used: the dynamic data set and the steady-state one. Analogously, there are two validation data sets. Model error for steady-state data is defined as

$$E^{\text{ss}} = \frac{1}{n_{\text{ss}}} \sum_{s=1}^{n_{\text{ss}}} (y_{\text{mod}}^{\text{ss}}(s) - y^{\text{ss}}(s))^2 \quad (27)$$

where $y_{\text{mod}}^{\text{ss}}(s)$ and $y^{\text{ss}}(s)$ denote the steady-state model output and the process output (i.e. a data sample from the available data set) in steady-state for the data sample s , respectively, the number of steady-state training patterns is denoted by n_{ss} . Model parameters are determined from the following constrained optimization task

$$\begin{aligned} \min_{r_i^j, b_i^j, w_{l,m}^1, w_l^2} \quad & E = E^{\text{dyn}} + \lambda E^{\text{ss}} \quad (28) \\ \text{subject to} \quad & -1 + \delta \leq r_i^j \leq 1 - \delta \end{aligned}$$

The above optimization problem is very similar to the one used in the identification algorithm 1 (Equation 10), the only difference is the fact that now the objective is to minimize both model errors in dynamic and steady-state conditions, $\lambda > 0$ is a weighting parameter (the bigger its value, the more important the steady-state error).

3.2.1 | Model structure A

The gradients of the objective function $E = E^{\text{dyn}} + \lambda E^{\text{ss}}$, minimized in the optimization problem (28), with respect to the poles are

$$\frac{\partial E}{\partial r_i^j} = \frac{\partial E^{\text{dyn}}}{\partial r_i^j} + \lambda \frac{\partial E^{\text{ss}}}{\partial r_i^j} \quad (29)$$

where for $i = 1, \dots, n_B^j, j = 1, \dots, n_u$. The gradients with respect to the coefficients b_i^j are

$$\frac{\partial E}{\partial b_i^j} = \frac{\partial E^{\text{dyn}}}{\partial b_i^j} + \lambda \frac{\partial E^{\text{ss}}}{\partial b_i^j} \quad (30)$$

where $i = 1, \dots, n_B^j, j = 1, \dots, n_u$. The gradients with respect to the weights $w_{l,m}^1$ are

$$\frac{\partial E}{\partial w_{l,m}^1} = \frac{\partial E^{\text{dyn}}}{\partial w_{l,m}^1} + \lambda \frac{\partial E^{\text{ss}}}{\partial w_{l,m}^1} \quad (31)$$

where $l = 1, \dots, K, m = 0, 1$. The gradients with respect to the weights w_l^2 are

$$\frac{\partial E}{\partial w_l^2} = \frac{\partial E^{\text{dyn}}}{\partial w_l^2} + \lambda \frac{\partial E^{\text{ss}}}{\partial w_l^2} \quad (32)$$

where $l = 0, \dots, K$. The gradients of the first part of the minimized objective function, E^{dyn} , are calculated in the same it is done in the case of the identification algorithm 1 for the structure A (Section 3.1.1). The gradients of the objective function E^{ss} with respect to the poles are

$$\frac{\partial E^{\text{ss}}}{\partial r_i^j} = \frac{2}{n_{\text{ss}}} \sum_{s=1}^{n_{\text{ss}}} (y_{\text{mod}}(s) - y(s)) \frac{\partial y_{\text{mod}}(s)}{\partial r_i^j} \quad (33)$$

where for $i = 1, \dots, n_B^j, j = 1, \dots, n_u$. The gradients with respect to the coefficients b_i^j are

$$\frac{\partial E^{\text{ss}}}{\partial b_i^j} = \frac{2}{n_{\text{ss}}} \sum_{s=1}^{n_{\text{ss}}} (y_{\text{mod}}(s) - y(s)) \frac{\partial y_{\text{mod}}(s)}{\partial b_i^j} \quad (34)$$

where $i = 1, \dots, n_B^j, j = 1, \dots, n_u$. The gradients with respect to the weights $w_{l,m}^1$ are

$$\frac{\partial E^{\text{ss}}}{\partial w_{l,m}^1} = \frac{2}{n_{\text{ss}}} \sum_{s=1}^{n_{\text{ss}}} (y_{\text{mod}}(s) - y(s)) \frac{\partial y_{\text{mod}}(s)}{\partial w_{l,m}^1} \quad (35)$$

where $l = 1, \dots, K, m = 0, 1$. The gradients with respect to the weights w_l^2 are

$$\frac{\partial E^{\text{ss}}}{\partial w_l^2} = \frac{2}{n_{\text{ss}}} \sum_{s=1}^{n_{\text{ss}}} (y_{\text{mod}}(s) - y(s)) \frac{\partial y_{\text{mod}}(s)}{\partial w_l^2} \quad (36)$$

where $l = 0, \dots, K$. In order to derive the partial derivatives in the right sides of Equations (33)-(36), it is necessary to consider steady-state behavior of the dynamic Wiener model. In steady-state, from Equation (1), one has

$$v_j = u_j \sum_{i=\tau^j}^{n_B^j} b_i^j \left[1 + \sum_{n=1}^{n_A^j} \alpha_n^j (r_1^j, \dots, r_{n_A^j}^j) \right]^{-1} \quad (37)$$

and Equation (3) gives

$$y = w_0^2 + \sum_{l=1}^K w_l^2 \varphi \left(w_{l,0}^1 + w_{l,1}^1 \sum_{j=1}^{n_u} v_j \right) \quad (38)$$

In the steady-state conditions the relations between the poles and the coefficients α_i^j of the linear dynamic part (Equation 6) described by Equations (7) or (8) are true. The partial derivatives used in the right side of Equation 33 are derived by differentiating Equations (37), (38) and (6) which gives

$$\begin{aligned} \frac{\partial y_{\text{mod}}(s)}{\partial r_i^j} = & - \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(s))}{dz_l(s)} w_{l,1}^1 u_j \\ & \sum_{i=\tau^j}^{n_B^j} b_i^j \left[1 + \sum_{n=1}^{n_A^j} \alpha_n^j (r_1^j, \dots, r_{n_A^j}^j) \right]^{-2} \\ & \times \sum_{n=1}^{n_A^j} \frac{\partial \alpha_n^j (r_1^j, \dots, r_{n_A^j}^j)}{\partial r_n^j} \end{aligned} \quad (39)$$

where from Equation 16

$$z_l(s) = w_{l,0}^1 + w_{l,1}^1 \sum_{j=1}^{n_u} v_j(s) \quad (40)$$

The derivatives of the function $\alpha_i^j(r_1^j, \dots, r_{n_A}^j)$ (Equation 6), used in the right side of Equation (39), are given by Equations (18) or (19), they are true no matter if the model is considered in dynamic or steady-state conditions. The partial derivatives used in the right sides of Equations (34) are derived by differentiating Equations (37) and (38), which gives

$$\frac{\partial y_{\text{mod}}(s)}{\partial b_i^j} = \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(s))}{dz_l(s)} w_{l,1}^1 u_j(s) \left[1 + \sum_{n=1}^{n_A} \alpha_n^j(r_1^j, \dots, r_{n_A}^j) \right]^{-1} \quad (41)$$

The partial derivatives used in the right sides of Equations (35) and (36) are calculated from Equations (21) and (22), respectively, but the index k must be replaced by s .

3.2.2 | Model structure B

The general formulas for gradient calculation, i.e. Equations (29)–(36), are the same for both model structures, the only minor difference is that in the case of the structure B the neural network has n_u inputs. Equation 37 is true, but from Equation (5), in place of Equation (38), one obtains

$$y = w_0^2 + \sum_{l=1}^K w_l^2 \varphi \left(w_{l,0}^1 + \sum_{j=1}^{n_u} w_{l,j}^1 v_j \right) \quad (42)$$

The partial derivatives used in the right side of Equation (33) are derived by differentiating Equations (37), (42) and 6 which gives

$$\begin{aligned} \frac{\partial y_{\text{mod}}(s)}{\partial r_i^j} = & - \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(s))}{dz_l(s)} w_{l,j}^1 u_j \sum_{i=r^j}^{n_B^j} b_i^j \\ & \times \left[1 + \sum_{n=1}^{n_A} \alpha_n^j(r_1^j, \dots, r_{n_A}^j) \right]^{-2} \\ & \times \sum_{n=1}^{n_A} \frac{\partial \alpha_n^j(r_1^j, \dots, r_{n_A}^j)}{\partial r_n^j} \end{aligned} \quad (43)$$

where from Equation (24)

$$z_l(s) = w_{l,0}^1 + \sum_{j=1}^{n_u} w_{l,j}^1 v_j(s) \quad (44)$$

The derivatives of the function $\alpha_i^j(r_1^j, \dots, r_{n_A}^j)$ (Equation 6), used in the right side of Equation 43, are also given by Equations (18) or (19) for the structure B. The partial derivatives used in the right side of Equations (34) are derived by differentiating Equations (37) and (38), which gives

$$\begin{aligned} \frac{\partial y_{\text{mod}}(s)}{\partial b_i^j} = & \sum_{l=1}^K w_l^2 \frac{d\varphi(z_l(s))}{dz_l(s)} w_{l,j}^1 u_j(s) \\ & \left[1 + \sum_{n=1}^{n_A} \alpha_n^j(r_1^j, \dots, r_{n_A}^j) \right]^{-1} \end{aligned} \quad (45)$$

The partial derivatives used in the right sides of Equations (35) and (36) are calculated from Equations (21) and (22), respectively, but the index k must be replaced by s .

3.3 | Algorithm 3

In the third identification algorithm, similarly to the second one, model accuracy in both transient and steady-state conditions is considered. However, in contrast to the second algorithm, as the minimized cost-function the dynamic model error E^{dyn} is only taken into account, but it is explicitly required that in steady-state the model output is exactly the same as the steady-state training data

$$y_{\text{mod}}^{\text{ss}}(s) = y^{\text{ss}}(s) \quad (46)$$

for all steady-state data samples, i.e. $s = 1, \dots, n_{\text{ss}}$. It leads to the following constrained optimization task

$$\begin{aligned} \min_{r_i^j, b_i^j, w_{l,m}^1, w_l^2} \quad & E^{\text{dyn}} \\ -1 + \delta \leq r_i^j \leq 1 - \delta \\ y_{\text{mod}}^{\text{ss}}(s) = y^{\text{ss}}(s), \quad & s = 1, \dots, n_{\text{ss}} \end{aligned} \quad (47)$$

Although theoretically correct, the optimization problem (47) is not recommended since the nonlinear equality constraints (46) greatly increase computational complexity. In general, equality constraints always are difficult to satisfy, they may even lead to the infeasibility problem, i.e. the set of solutions of the whole optimization task may be empty. Hence, the equality constraints (46) are relaxed by the following equality ones

$$y^{\text{ss}}(s) - \epsilon^{\min} \leq y_{\text{mod}}^{\text{ss}}(s) \leq y^{\text{ss}}(s) + \epsilon^{\max} \quad (48)$$

The values of quotients $\epsilon^{\min} \geq 0$ and $\epsilon^{\max} \geq 0$ define the degree of violation of the original equality constraints (46). The resulting constrained optimization problem is

$$\begin{aligned}
& \min_{\substack{r_i^j, b_i^j, w_{lm}^j, \\ w_l^2, \epsilon^{\min}, \epsilon^{\max}}} E = E^{\text{dyn}} + \lambda^{\min} (\epsilon^{\min})^2 + \lambda^{\max} (\epsilon^{\max})^2 \quad (49) \\
& -1 + \delta \leq r_i^j \leq 1 - \delta \\
& -y_{\text{mod}}^{\text{ss}}(s) + y^{\text{ss}}(s) - \epsilon^{\min} \leq 0, \quad s = 1, \dots, n_{\text{ss}} \\
& y_{\text{mod}}^{\text{ss}}(s) - y^{\text{ss}}(s) - \epsilon^{\max} \leq 0, \quad s = 1, \dots, n_{\text{ss}} \\
& \epsilon^{\min} \geq 0, \quad \epsilon^{\max} \geq 0
\end{aligned}$$

The original hard constraints (46) can be temporarily violated, but it enlarges the admissible set of possible solutions when necessary. The degree of constraint violation is minimized by adding a quadratic penalty to the minimized cost function, $\lambda^{\min} > 0$, $\lambda^{\max} > 0$ are weights. A very similar relaxation technique is commonly used in control algorithms [23,24].

Since the model parameters do not depend on the slack optimization variables, $\epsilon^{\min}$ and $\epsilon^{\max}$, the gradients of the objective function E , minimized in the optimization problem (49), with respect to model parameters are the same as those used in algorithm 1, which is discussed in Sections 3.1.1 and 3.1.2 for the model structures A and B, respectively. Additional derivatives are

$$\frac{\partial E}{\partial \epsilon^{\min}} = 2\lambda^{\min} \epsilon^{\min}, \quad \frac{\partial E}{\partial \epsilon^{\max}} = 2\lambda^{\max} \epsilon^{\max} \quad (50)$$

4 | SIMULATION RESULTS

In this section the effectiveness of the three discussed identification algorithms and two Wiener structures is studied.

4.1 | Solid oxide fuel cell system description

Details of mathematical modeling of SOFCs using first-principle laws are given e.g. in: [11,12,25-27]. The fundamental model of the SOFC introduced in [28] and next discussed in [19,29] is used in this work. The considered SOFC has two manipulated variables (the inputs of the process): the input gas flow rate q_f^{in} (mols⁻¹) and the input oxygen flow rate $q_{O_2}^{\text{in}}$ (mols⁻¹) as well as one disturbance (the uncontrolled input) I_{in} which is the external current load (A). There is one main controlled variable (the output of the process) which is the stack output voltage V_r (V). The values of process inputs are constrained

$$0 \text{ mol s}^{-1} \leq q_f^{\text{in}} \leq 1.7023 \text{ mol s}^{-1} \quad (51)$$

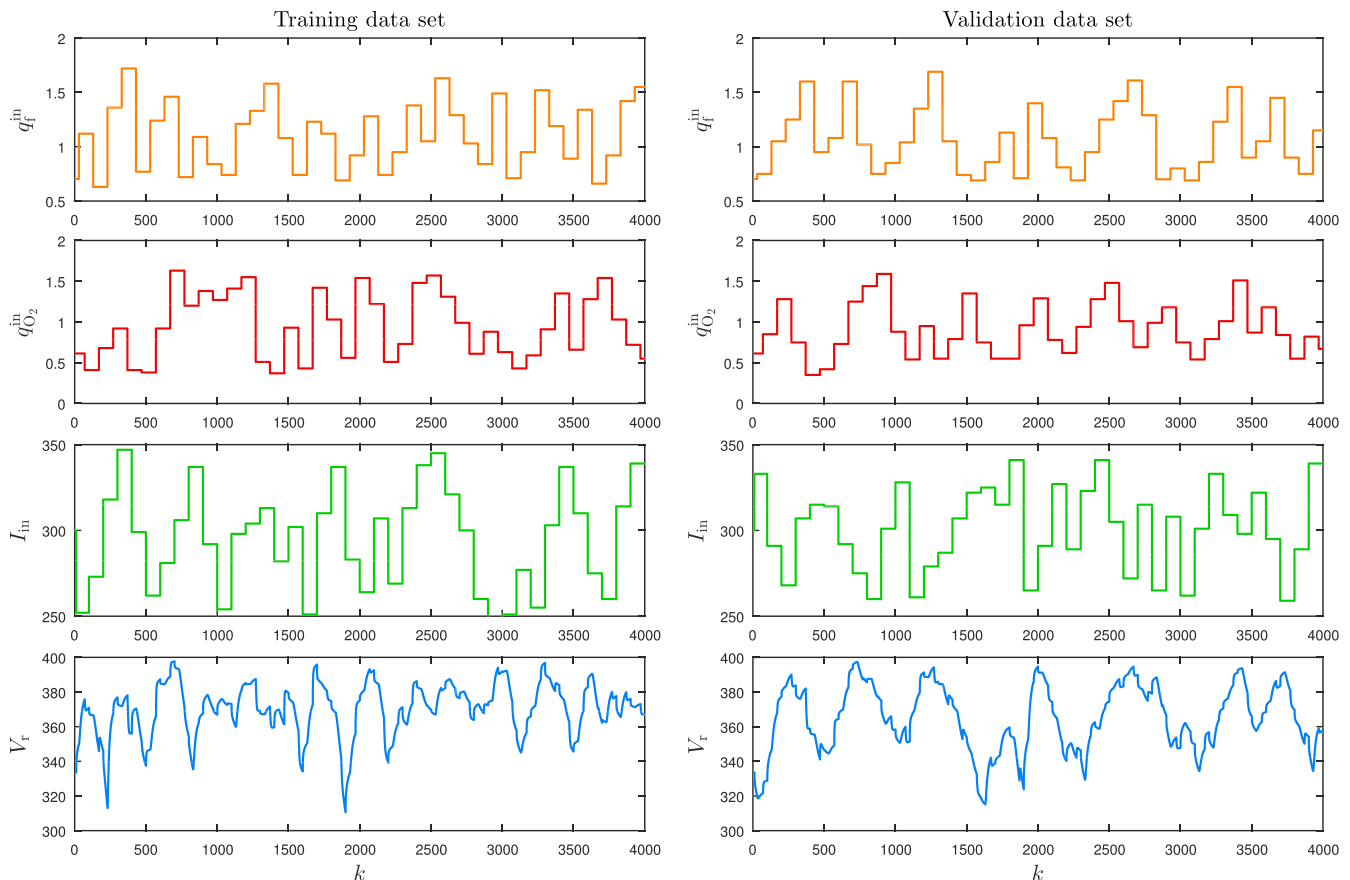


FIGURE 3 Training (left) and validation (right) dynamic data sets [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 1 Errors of the model structure A trained using different algorithms for the training and validation data sets, errors for dynamic and steady-state data are given

Identification algorithm	$E_{\text{dyn}}^{\text{train}}$	$E_{\text{ss}}^{\text{train}}$	$E_{\text{dyn}}^{\text{val}}$	$E_{\text{ss}}^{\text{val}}$
Algorithm 1	1.4586×10^1	1.6423×10^2	2.5439×10^1	1.5013×10^2
Algorithm 2	2.3129×10^1	1.1233×10^2	4.1340×10^1	1.1845×10^2
Algorithm 3	2.3158×10^1	1.2105×10^2	3.0193×10^1	1.3437×10^2

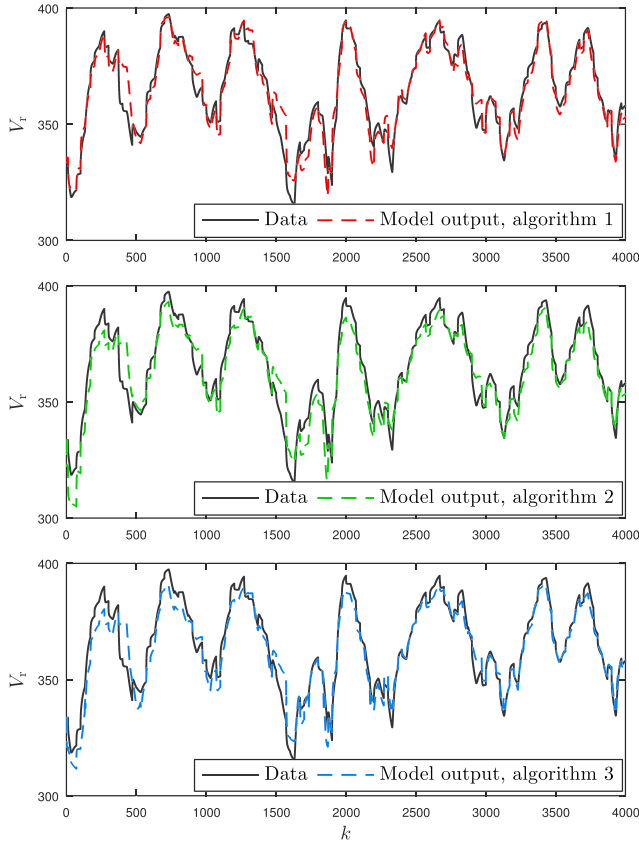


FIGURE 4 The dynamic validation data set (solid line) vs. the output of the model structure A trained using different algorithms (dashed line) [Color figure can be viewed at wileyonlinelibrary.com]

$$0 \text{ mol s}^{-1} \leq q_{\text{O}_2}^{\text{in}} \leq 1.6134 \text{ mol s}^{-1} \quad (52)$$

The value of the disturbance is also limited

$$250 \text{ A} \leq I_{\text{in}} \leq 350 \text{ A} \quad (53)$$

4.2 | Wiener models identification of solid oxide fuel cell

The discussed SOFC is characterized by linear dynamic behavior and nonlinear steady-state behavior. Hence, as the empirical models the Wiener structure is straightforward.

For model identification, the manipulated variables of the process, q_f^{in} and $q_{\text{O}_2}^{\text{in}}$, the disturbance, I_{in} , and the output, V_r , are scaled in the following way:

$$u_1 = q_f^{\text{in}} - \bar{q}_f^{\text{in}}, \quad u_2 = q_{\text{O}_2}^{\text{in}} - \bar{q}_{\text{O}_2}^{\text{in}} \quad (54)$$

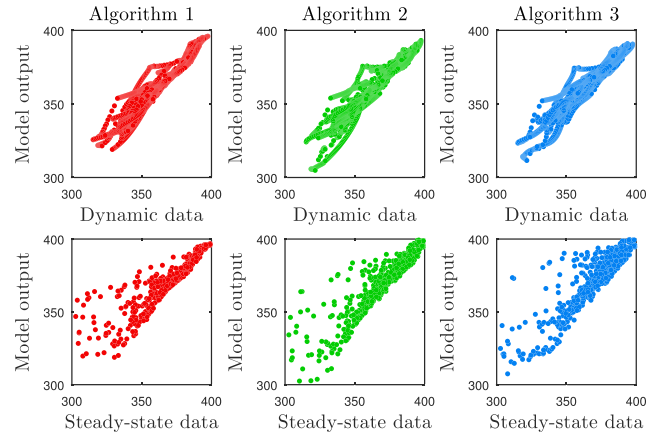


FIGURE 5 The relation between the validation data vs. the outputs of the model structure A trained using different algorithms for dynamic validation data set (top panels) and steady-state validation data set (bottom panels) [Color figure can be viewed at wileyonlinelibrary.com]

$$u_3 = 0.02(I_{\text{in}} - \bar{I}_{\text{in}}), \quad y = V_r - \bar{V}_r \quad (55)$$

where $\bar{q}_f^{\text{in}} = 0.7023 \text{ mol s}^{-1}$, $\bar{q}_{\text{O}_2}^{\text{in}} = 0.6134 \text{ mol s}^{-1}$, $\bar{I}_{\text{in}} = 300 \text{ A}$ and $\bar{V}_r = 3.335865 \times 10^2 \text{ V}$ denote values of process variables at the initial operating point.

The objective is to find a Wiener model of the stack voltage, V_r , taking into account both dynamic and steady-state model accuracy. The fundamental model of the SOFC is treated as “the real process” to generate the data next used for identification of Wiener models. When one has to develop a model for a real process, real data sets recorded during process operation are used for model identification. In general, as many as four data sets are used: dynamic training and validation data sets as well as steady-state training and validation ones. The training data sets are used only for model identification, i.e. to find parameters of models. The validation data sets are used only to assess generalisation ability of models. Firstly, the continuous-time fundamental model is simulated (the resulting system of differential equations is solved by the Runge-Kutta method of order 45). The sampling period is 1 second. The dynamic training and validation data sets are depicted in Figure 3, both sets consist of 4000 samples. As the process inputs and the disturbance signals a series of random sequences from the range characterized by Equations (51)–(53) is used. Secondly, from the fun-

TABLE 2 Errors of the model structure B trained using different algorithms for the training and validation data sets, errors for dynamic and steady-state data are given

Identification algorithm	$E_{\text{dyn}}^{\text{train}}$	$E_{\text{ss}}^{\text{train}}$	$E_{\text{dyn}}^{\text{val}}$	$E_{\text{ss}}^{\text{val}}$
Algorithm 1	2.8125×10^{-1}	5.3972×10^1	9.4749×10^{-1}	3.5639×10^1
Algorithm 2	3.8580×10^{-1}	4.2385×10^{-1}	5.9092×10^{-1}	1.8888×10^0
Algorithm 3	5.6838×10^{-1}	1.6694×10^0	1.5833×10^0	2.2336×10^0

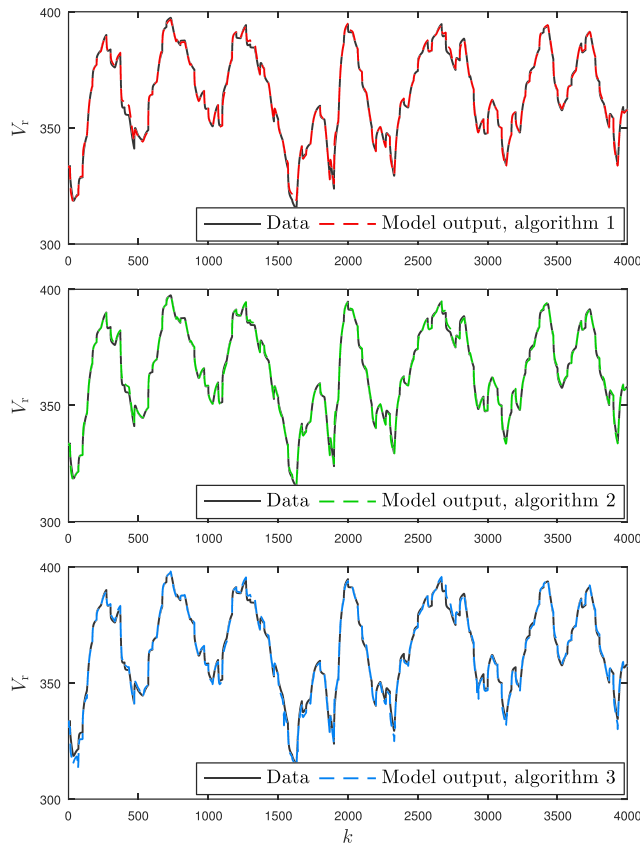


FIGURE 6 The dynamic validation data set (solid line) vs. the output of the model structure B trained using different algorithms (dashed line) [Color figure can be viewed at wileyonlinelibrary.com]

damental dynamic model, the steady-state one is derived. From that model the steady-state training and validation data sets are generated for random values from the range characterized by Equations (51)–(53). Both steady-state data sets consist of 500 samples.

The order of all compared Wiener models is defined by $\tau^j = 1$, $n_A^j = n_B^j = 2$, $j = 1, 2, 3$, which means that three dynamic model blocks shown in Figures 1 and 2 are of the second order. In all cases neural networks with 10 hidden nodes are used ($K = 10$). Since the solved optimization problems may have local minima, each model configuration has been trained 10 times, only the best results are presented. The maximal number of training epochs is 500,

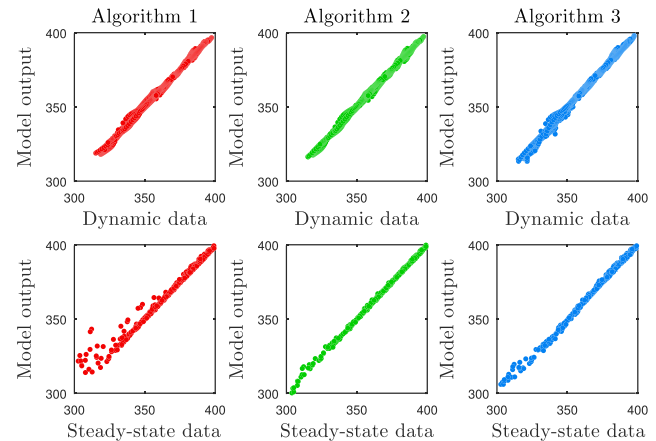


FIGURE 7 The relation between the validation data vs. the outputs of the model structure B trained using different algorithms for dynamic validation data set (top panels) and steady-state validation data set (bottom panels) [Color figure can be viewed at wileyonlinelibrary.com]

but training is stopped when the validation error grows. Additional parameters are: $\lambda = 1$ (algorithm 2), $\lambda^{\min} = \lambda^{\max} = 0.02$ (algorithm 3).

4.2.1 | Effectiveness of model structure A

At first the model structure A and 3 discussed identification algorithms are considered. Table 1 shows model errors for the training and validation data sets, both errors for dynamic and steady-state data are given. The errors for the training set are given for information purposes, the model is always assessed taking into account its validation errors. Figure 4 depicts the dynamic validation data set vs. the output of the model. Figure 5 shows the relation between the validation data vs. the outputs of models for both dynamic and steady-state data sets, respectively (for perfect models the relation figures should consist of points which form lines of the slope 45 degrees). In general, the model A gives very poor results, both dynamic and steady-state validation errors, $E_{\text{dyn}}^{\text{val}}$ and $E_{\text{ss}}^{\text{val}}$, respectively, are big. The steady-state error is the biggest when it is not minimized during model identification (algorithm 1). Unfortunately, in algorithms 2 and 3 the dynamic error is bigger than in the case of the algorithm 1. One may con-

clude that for the considered SOFC process with 3 input variables the simple steady-state nonlinear block with only 1 input is insufficient.

4.2.2 | Effectiveness of model structure B

Next, the model structure B and 3 discussed identification algorithms are considered. Table 2 shows model errors for the training and validation data sets, both errors for dynamic and steady-state data are given. Figure 6 depicts the dynamic validation data set vs. the output of the model. Figure 7 shows the relation between the validation data vs. the outputs of models for both dynamic and steady-state data sets, respectively. In general, the model B is much more precise in comparison with the model A, but the three compared training algorithms give different results. Taking into account the validation errors, the algorithm 1 gives quite a low value of the dynamic error since its identification task is defined as minimization of that error. Unfortunately, during model identification the steady-state error is not considered and it has the biggest value for all 3 compared algorithms. The algorithm 2 during identification takes into account both dynamic and steady-state error and both of them have minimal values in the case of the structure B. It is interesting to note that when compared with the algorithm 1, the algorithm 2 gives not only lower model error for steady-state, but also for dynamic data. The objective of the algorithm 3 is to minimize the dynamic error and take into account the constraints to enforce good steady-state properties of the model. Unfortunately, the necessity of satisfying the additional constraints makes the whole optimization problem computationally demanding and difficult. The obtained model in terms of the dynamic error is the worst one among the B structure, but it is much better when compared to the model A. At the same time, the steady-state error of the model B trained by the algorithm 3 is only somehow worse than that of the model determined by the algorithm 2, much better than the model obtained by the algorithm 1.

Considering the obtained results one may conclude that for the considered Wiener model of the SOFC stack voltage, the best model accuracy both in dynamic and steady-state conditions are obtained in the structure B of the model and when the algorithm 2 is used. It turns out that minimization at the same time the dynamic and steady-state errors is simple, yet very efficient. Incorporation of additional inequality constraints to enforce good model accuracy in steady-state does not lead to better results, but significantly increases computational complexity of the identification procedure.

5 | CONCLUSIONS

Efficient advanced control schemes [7,24], process optimization algorithms [24], fault tolerant control [30] and estimation [31] require precise models of considered processes. Typically, during model identification accuracy in dynamic conditions is optimized, but such a model may have poor steady-state performance, which is crucial, e.g. in on-line process set-point optimization [24]. In this work two model identification algorithms are discussed the objective of which is to optimize model accuracy both in dynamic as well as in steady-state conditions. In the first approach (algorithm 2) both dynamic and steady-state model errors are directly taken into account in the minimized cost function. In the second method (algorithm 3) good model behaviour in steady-state is enforced by additional inequality constraints. Implementation details of these algorithms for two Wiener MISO structures are discussed. To compare effectiveness of the algorithms and two model structures, the problem of obtaining a precise model of the SOFC process is considered. The best results are obtained by the algorithm 2 which minimizes at the same time both dynamic and steady-state model errors, additional constraints turn out to be computationally quite demanding and do not lead to better models. The presented approach may be directly used for identification of models of alternative fuel cells, e.g. [32].

The identification algorithms presented in this work are universal since they are developed for the MISO case. Moreover, general ideas are true for other Wiener structures, with different types of the steady-state part. Application e.g. of Radial Basis Function (RBF) neural networks in place of the multi-layer perceptrons is possible. Finally, it is also possible to use a different kind of a nonlinear model, but it must be stressed that it is relatively easy to consider steady-state behavior in the identification algorithm of cascade models, i.e. Hammerstein, Wiener, Hammerstein-Wiener and Wiener-Hammerstein structures [1]. For general black-box models, it may be not possible.

ORCID

Maciej Ławryńczuk  <https://orcid.org/0000-0002-6846-2004>

REFERENCES

1. A. Janczak, *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.
2. Q. Chi et al., *Latent-variable model predictive control strategy for a ph neutralization process*, *Asian J. Control* **17** (2015), 2427–2434.

3. H. H. J. Bloemen et al., *Wiener model identification and predictive control for dual composition control of a distillation column*, J. Process. Control **11** (2001), 601–620.
4. V. Arto, P. Hannu, and A. Halme, *Modeling of chromatographic separation process with wiener-MLP representation*, J. Process. Control **78** (2001), 443–458.
5. T. Knohl and H. Unbehauen, *Adaptive position control of electrohydraulic servo system using ANN*, Mechatronics **10** (2000), 127–143.
6. R. Hodrea, I. Nascu, and H. Vasan, *Wiener model identification for muscle relaxation*, Proc. 2012 IEEE Int. Conf. Automation Quality Testing Robotics, Cluj-Napoca, Romania, **6** (2012), 481–485.
7. M. Ławryńczuk, *Computationally Efficient Model Predictive Control Algorithms: A Neural Network Approach*, Studies Systems Decision and Control, Vol. **3**, Springer, Cham, 2014.
8. S. A. Billings and S. Y. Fakhouri, *Identification of systems containing linear dynamic and static nonlinear elements*, Automatica **18** (1982), 15–26.
9. W. Greblicki, *Nonparametric identification of Wiener systems by orthogonal series*, IEEE Trans. Autom. Control **39** (1994), 2077–2086.
10. G. Mzyk, *Combined Parametric-Nonparametric Identification of Block-Oriented Systems*, Lecture Notes in Control and Information Sciences, Vol. **454**, Springer, Berlin, 2014.
11. J. Larminie and A. Dicks, *Fuel Cell Systems Explained*, John Wiley & Sons, Chichester, 2003.
12. J. Milewski et al., *Advanced Methods of Solid Oxide Fuel Cell Modelling*, Springer, London, 2011.
13. J. Chen et al., *Observer based fuel delivery control for PEM fuel cells with a segmented anode model*, Asian J. Control **20** (2018), 1–15.
14. H.-L. Tsai and J.-M. Lin, *Optimal control of fuel processing systems using generalized linear quadratic Gaussian and loop transfer recovery method*, Asian J. Control **12** (2010), 597–606.
15. R. D. Tehrani and F. Shabani, *Performance improvement of fuel cells using perturbation-based extremum seeking and model reference adaptive control*, Asian J. Control **19** (2017), 2178–2191.
16. A. T. Tahri et al., *Nonlinear adaptive control of a hybrid fuel cell power system for electric vehicles – a Lyapunov stability based approach*, Asian J. Control **18** (2016), 166–177.
17. R. T. Meyer et al., *Hybrid model predictive power management of a fuel cell-battery vehicle*, Asian J. Control **15** (2013), 363–379.
18. Q. Ouyang et al., *Nonlinear MPC controller design for air supply of PEM fuel cell based power systems*, Asian J. Control **19** (2017), 929–940.
19. X. Wang, B. Huang, and T. Chen, *Data-driven predictive control for solid oxide fuel cells*, J. Proc. Control **17** (2007), 103–114.
20. M. Shahiri et al., *Tuning method for fractional complex order controller using standarized k-chart: Application to PEMFC control*, Asian J. Control **18** (2016), 1102–1118.
21. S. Haykin, *Neural Networks and Learning Machines: A comprehensive foundation*, Prentice Hall, Englewood Cliffs, 2008.
22. J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, Berlin New York, 2006.
23. J. M. Maciejowski, *Predictive control with constraints*, Prentice, Hall, Englewood Cliffs, 2002.
24. P. Tatjewski, *Advanced Control of Industrial Processes, Structures and Algorithms*, Springer, London, 2007.
25. S. A. Hajimolana et al., *Mathematical modeling of solid oxide fuel cells: A review*, Renew. Sustain. Energy Rev. **15** (2011), 1893–1917.
26. M. Peksen, *Numerical thermomechanical modelling of solid oxide fuel cells*, Progress Energy Combust. Sci. **48** (2011), 1–20.
27. C. Bao et al., *Macroscopic modeling of solid oxide fuel cell (SOFC) and model-based control of SOFC and gas turbine hybrid system*, Progress Energy Combust. Sci. **66** (2018), 83–140.
28. J. Padullés, G. W. Ault, and J. R. McDonald, *An integrated SOFC plant dynamic model for power systems simulation*, J. Power Sources **86** (2000), 495–500.
29. Y. Zhu and K. Tomasovic, *Development of models for analyzing the load-following performance of microturbines and fuel cells*, Electric Power Syst. Res. **62** (2002), 1–11.
30. M. Witczak, *Fault Diagnosis and Fault-Tolerant Control Strategies for Non-Linear Systems Analytical and Soft Computing Approaches*, Lecture Notes in Electrical Engineering, Vol. **266**, Springer, Cham, 2014.
31. B. Dolenc et al., *State of health estimation and remaining useful life prediction of solid oxide fuel cell stack*, Energy Conv. Manag. **148** (2017), 993–1002.
32. M. Özbek et al., *Modeling and control of a PEM fuel cell system: a practical study based on experimental defined component behavior*, J. Proc. Control **23** (2013), no. 2013, 282–293.

AUTHOR BIOGRAPHY



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

How to cite this article: Ławryńczuk M. Identification of Wiener models for dynamic and steady-state performance with application to solid oxide fuel cell. *Asian J Control*. 2019;1–11. <https://doi.org/10.1002/asjc.2038>