International Journal of Systems Science
Vol. 44, No. 9, September 2013, 1666–1674

Taylor & Francis
Taylor & Francis Group

System identification of Wiener systems with B-spline functions using De Boor recursion

X. Hong[a]*, R.J. Mitchell[a] and S. Chen[bc]

[a]*School of Systems Engineering, University of Reading, Reading RG6 6AY, UK;* [b]*School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK;* [c]*Faculty of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia*

(*Received 6 December 2010; final version received 29 January 2012*)

In this article a simple and effective algorithm is introduced for the system identification of the Wiener system using observational input/output data. The nonlinear static function in the Wiener system is modelled using a B-spline neural network. The Gauss–Newton algorithm is combined with De Boor algorithm (both curve and the first order derivatives) for the parameter estimation of the Wiener model, together with the use of a parameter initialisation scheme. Numerical examples are utilised to demonstrate the efficacy of the proposed approach.

Keywords: B-spline; De Boor recursion; Wiener system; system identification

1. Introduction

A popular approach to nonlinear systems identification is to use the so-called block-oriented nonlinear models which comprise the linear dynamic models and static or memoryless nonlinear functions (Bai 1998; Zhu 2002; Schoukens, Nemeth, Crama, Rolain, and Pintelon 2003; Hsu, Vincent, and Poolla 2006). The Hammerstein model, comprising a nonlinear static functional transformation followed by a linear dynamical model, has been applied to nonlinear plant/process modelling in a wide range of engineering problems (Balestrino, Landi, Ould-Zmirli, and Sani 2001; Bloemen, Van Den Boom, and Verbruggen 2001; Turunen, Tanttu, and Loula 2003). The Hammerstein model has been widely researched (Billings and Fakhouri 1979; Stoica and Söderström 1982; Greblicki and Pawlak 1986; Greblicki 1989, 2002; Lang 1997; Verhaegen and Westwick 1996; Bai and Fu 2002; Chen 2004; Chaoui, Giri, Rochdi, Haloua, and Naitali 2005; Hong and Mitchell 2007). Alternatively, the Wiener model comprises a linear dynamical model followed by a nonlinear static functional transformation. This is a reasonable model for any linear systems with a nonlinear measurement device, or some industrial/biological systems (Hunter and Korenberg 1986; Kalafatis, Arifinand, Wang, and Cluett 1995; Kalafatis, Wang, and Cluett 1997; Zhu 1999; Gomez, Jutan, and Baeyens 2004; Hagenblad, Ljung, and Wills 2008). The model characterization/representation of the unknown nonlinear static function in the Wiener model is fundamental to its identification and control. Various approaches have been developed in order to capture the *a priori* unknown nonlinearity including the nonparametric method (Greblicki 1992), subspace model identification methods (Westwick 1996; Gomez et al. 2004), fuzzy modelling (Skrjanc, Blazic, and Agamennoni 2005) and the parametric method (Kalafatis et al. 1995, 1997; Bai 1998; Zhu 1999). For the parametric method, the unknown nonlinear function is restricted by some parametric representation with a finite number of parameters. In particular, the nonlinear subsystem often has a predetermined linear in the parameters model structure. A nonlinear polynomial function of a known finite degrees is usually considered to be appropriate in approximating the unknown function (Verhaegen and Westwick 1996; Chaoui et al. 2005) based on the Stone–Weierstrass theorem. Conventional nonlinear optimisation algorithms can be applied to determine the unknown parameters.

The spline curves consist of many polynomial pieces offering versatility. The use of piecewise linearity (Wigren 1993, 1994) and various spline functions (Zhu 1999; Hughes and Westwick 2005) in the modelling of the Wiener system have been researched. Given its best conditioning property, the B-spline curve has been widely used in computer graphics and computer-aided geometric design (CAGD) (Farin 1994). The early work on the construction of B-spline curve is mathematically involved and numerically unstable (De Boor 1978). The De Boor algorithm uses recurrence relations and is numerically stable (De Boor 1978). The B-spline basis functions for nonlinear systems modelling have been widely applied (Kavli 1993; Brown and Harris

*Corresponding author. Email: x.hong@reading.ac.uk

ISSN 0020–7721 print/ISSN 1464–5319 online

http://dx.doi.org/10.1080/00207721.2012.669863
http://www.tandfonline.com

1994; Harris, Hong, and Gan 2002). In this article we model the nonlinear static function in the Wiener system using a B-spline neural network. We point out that there are clear differences between the proposed approach to other spline functions-based methods (Zhu 1999; Hughes and Westwick 2005). It is shown that by minimising the mean square error (MSE) between the model output and the system output, the Gauss–Newton algorithm is readily applicable for the parameter estimation in the proposed model. The Gauss–Newton algorithm is combined with the De Boor algorithm (both curve and the first-order derivative) for the parameter estimation of the Wiener model, following a parameter initialisation scheme. The proposed model based on B-spline functions with the De Boor recursion has several advantages over many existent Wiener system modelling paradigms. First, unlike B-spline functions, the spline functions used in the Wiener system modelling (Zhu 1999; Hughes and Westwick 2005) do not have the property of partition of unity (convexity), which is a desirable property in achieving numerical stability. Second, the proposed algorithm based on the De Boor recursion enables stable and efficient evaluations of functional and derivative values, as required in nonlinear optimisation algorithms, e.g. the Gauss–Newton algorithm used in this article. Final, rather than just using the most commonly used cubic splines, the modeller has the freedom/flexibility to cope with different model setting such as the number of knots and polynomial order.

2. The Wiener system and B-spline neural network

2.1. *The Wiener system*

The Wiener system consists of a cascade of two subsystems, a linear filter as the first subsystem, followed by a nonlinear memoryless function $\Psi(\bullet)$: $\mathcal{R} \to \mathcal{R}$ as the second subsystem. The system can be represented by

$$\begin{aligned} v(t) = \frac{z^{-d}B(z)}{A(z)}u(t) &= b_0u(t-d) + b_1u(t-d-1) \\ &\quad + \cdots + b_{n_b}u(t-d-n_b) \\ &\quad - a_1v(t-1) - a_2v(t-2) - \cdots - a_{n_a}v(t-n_a), \quad b_0 = 1 \end{aligned} \tag{1}$$

$$\begin{aligned} y(t) &= \Psi(v(t)) + C(z)\xi(t) \\ &= \Psi(v(t)) + c_1\xi(t-1) + c_2\xi(t-2) \\ &\quad + \cdots + c_{n_c}\xi(t-n_c) + \xi(t) \end{aligned} \tag{2}$$

with z transfer functions $A(z)$, $B(z)$ and $C(z)$ which are defined by

$$A(z) = \sum_{j=0}^{n_a} a_j z^{-j}, \qquad a_0 = 1 \tag{3}$$

$$B(z) = \sum_{j=0}^{n_b} b_j z^{-j}, \qquad b_0 = 1 \tag{4}$$

$$C(z) = \sum_{j=0}^{n_c} c_j z^{-j}, \qquad c_0 = 1 \tag{5}$$

$u(t) \in \mathcal{R}$ is the system input and $y(t) \in \mathcal{R}$ is the system output. $\xi(t)$ is assumed to be a white noise sequence independent of $u(t)$, with zero mean and variance σ^2. $v(t) \in \mathcal{R}$ is the output of the linear filter subsystem and the input to the nonlinear subsystem. a_j, b_j are the coefficients of the linear filter. $d \geq 1$ is an assumed known positive integer representing the delay of the system. c_j are the coefficients of the linear filter to the noise. n_a, n_b and n_c are assumed known positive integers. We denote $\mathbf{a} = [a_1, \ldots, a_{n_a}]^T \in \mathcal{R}^{n_a}$, $\mathbf{b} = [b_1, \ldots, b_{n_b}]^T \in \mathcal{R}^{n_b}$ and $\mathbf{c} = [c_1, \ldots, c_{n_c}]^T \in \mathcal{R}^{n_c}$. The objective of system identification for the above Wiener model is that, given an observational input/output data set $D_N = \{y(t), u(t)\}_{t=1}^N$, to identify $\Psi(\bullet)$ and to estimate the parameters a_j, b_j in the linear subsystems and c_j in the noise filter. Note that this model is a special case of the general Wiener systems (Hagenblad et al. 2008).

Without significant loss of generality, the following assumptions are initially made about the problem.

Assumption 1: *Persistence excitation condition is given by*

$$\operatorname{rank}\begin{pmatrix} u(n_b) & \cdots & u(1) & y(n_a) & \cdots & y(1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u(N-1-d) & \cdots & u(N-n_b-d) & y(N-1) & \cdots & y(N-n_a) \end{pmatrix} = n_a + n_b \tag{6}$$

Assumption 2: *$A(z)$ and $C(z)$ have all zeros inside the unit circle.*

Assumption 3: *$v(t)$ is bounded by $V_{\min} \leq v(t) \leq V_{\max}$, where $V_{\min}$ and $V_{\max}$ are finite real values.*

Remarks:

- Assumption 1 is necessary irrespective of the model representation and identification algorithm. Persistence excitation is the essential condition to the input signal for the sake of the identifiability.
- Assumption 2 represents the conditions on the stability and the identifiability of the system.

Note that for the simplest case $\Psi(\bullet)=1$, the proposed algorithm reduces to the prediction error method (Soderström and Stoica 1989), in which condition on $C(z)$ is required.

- Since only the input–output data are available and no internal signals are available, we fix $b_0=1$ so that identification is possible regarding uniqueness of parameter estimators. Otherwise any pair of $\{\lambda[b_0,\ldots,b_{n_b}]^T,\ \Psi(\bullet)/\lambda\}$, for $\lambda\neq 0$, provides the identical input–output measurements. Due to the constraint $b_0=1$, although the signals between the two subsystems are unavailable, Assumption 3 is valid. $V_{\min}$ and $V_{\max}$ need not to be known precisely and they can be set based on an auxiliary signal $\{\hat{v}(t)\}$ as defined in (21) in the modelling process. Note that if the nonlinear subsystem is modelled using other local basis functions, e.g. piecewise linear models or radial basis functions (RBF), there is a need to impose constraints on the range of $v(t)$, and determine the required parameters for the associated models (knots or centres).

In this work the B-spline basis functions are adopted in order to model $\Psi(\bullet)$. Specifically, the De Boor algorithm (De Boor 1978) is used in the construction of the B-spline basis functions, as described below.

2.2. *Modelling of $\Psi(\bullet)$ using B-spline function approximation with De Boor's algorithm*

De Boor's algorithm is a fast and numerically stable algorithm for evaluating B-spline spline curves. Univariate B-spline basis functions are parameterised by the order of a piecewise polynomial of order $(k-1)$ and also by a knot vector which is a set of values defined on the real line that break it up into a number of intervals. Suppose that there are M basis functions, the knot vector is specified by $(M+k)$ knot values, $\{V_1, V_2, \ldots, V_{M+k}\}$. At each end there are k knots satisfying the condition of being external to the input region, and as a result the number of internal knots is $(M-k)$. Specifically

$$\begin{aligned} V_1 < V_2 < V_k < V_{\min} < V_{k+1} < V_{k+2} < \cdots < V_M \\ < V_{\max} < V_{M+1} < \cdots < V_{M+k}. \end{aligned} \tag{7}$$

Given these predetermined knots, a set of M B-spline basis functions can be formed by using the De Boor recursion (De Boor 1978), given by

$$\mathcal{B}_j^{(0)}(v)=\begin{cases} 1 & \text{if } V_j \le v \le V_{j+1} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$$\left.\begin{aligned} j=1,\ldots,(M+k) \\ \mathcal{B}_j^{(i)}(v)=\frac{v-V_j}{V_{i+j}-V_j}\mathcal{B}_j^{(i-1)}(v)+\frac{V_{i+j+1}-v}{V_{i+j+1}-V_{j+1}}\mathcal{B}_{j+1}^{(i-1)}(v), \\ j=1,\ldots,(M+k-i) \\ i=1,\ldots,k \end{aligned}\right\} \tag{9}$$

Notably the first-order derivatives of the B-spline function has a similar recursion

$$\frac{\mathrm{d}}{\mathrm{d}v}\mathcal{B}_j^{(k)}(v)=\frac{k}{V_{k+j}-V_j}\mathcal{B}_j^{(k-1)}(v)-\frac{k}{V_{k+j+1}-V_{j+1}}\mathcal{B}_{j+1}^{(k-1)}(v), \quad j=1,\ldots,M \tag{10}$$

Note that the early work on the construction of B-spline curve is mathematically involved. Hence, another advantage of using De Boor's recursion is the flexibility in terms of the evaluations of functional and derivative values, since it can cope with different setting such as number of knots, and polynomial order.

We model $\Psi(\bullet)$ in (2) as

$$\hat{\Psi}(v)=\sum_{j=1}^{M}\mathcal{B}_j^{(k)}(v)\omega_j \tag{11}$$

where $\hat{}$ denotes the estimate, ω_j's are weights to be determined. We denote $\boldsymbol{\omega}=[\omega_1,\ldots,\omega_M]^T\in\Re^M$. Note that model (11) satisfies the property of partition of unity (convexity), i.e. $(\mathcal{B}_j^{(k)}(v)\ge 0,\ \sum_{j=1}^{M}\mathcal{B}_j^{(k)}(v)=1)$, which is a desirable property in achieving numerical stability (Farouki and Goodman 1996).

The optimisation of model output with respect to the number/location of knots is an intractable mixed integer problem. With the number of knots and their location determined, conventional nonlinear optimisation algorithms are applicable. In practice, the number of knots are predetermined to produce a model as small as possible that can still provide good modelling capability. The model performance is not particularly sensitive to the location of knots if these are evenly spread out. If there is severe local nonlinearity, the location of knots can be empirically set by the user by inserting more knots at higher density in regions with high curvatures. These regions can be identified by trial and error.

Note that due to the piecewise nature of B-spline functions, there are only k basis functions with non-zero values, and non-zero first-order derivatives, for any point v. Hence, the computational cost for the evaluation of $\Psi(v)$ based on the De Boor algorithm is to do with k, rather than the number of knots, and this is in the order of $O(k^2)$. The evaluation of the first-order derivatives can be regarded as a byproduct, with the additional computational cost in the order of $O(k)$.

3. The proposed system identification algorithm

Let the prediction error (Soderström and Stoica 1989) between the Wiener system output $y(t)$ and $\hat{y}(t)$, the model predicted output for $y(t)$, be denoted by $e(t)=y(t)-\hat{y}(t)$, and $\mathbf{e}=[e(1),e(2),\ldots,e(N)]^T$. With the B-spline approximation, the model predicted output $\hat{y}(t)$ can be written as

$$\hat{y}(t)=\Psi(v(t,\mathbf{a},\mathbf{b}),\boldsymbol{\omega})+\sum_{j=1}^{n_c}c_j e(t-j)$$
$$=\sum_{j=1}^{M}\mathcal{B}_j^{(k)}(v(t,\mathbf{a},\mathbf{b}))\omega_j+\sum_{j=1}^{n_c}c_j e(t-j) \quad (12)$$

The specific system identification task is to jointly estimate $\mathbf{a},\mathbf{b},\mathbf{c}$ and $\boldsymbol{\omega}$. This could be achieved by minimising

$$V=\sum_{t=1}^{N}[e(t)]^2 \quad (13)$$

via the Gauss–Newton algorithm. Note that

$$E\left[\frac{V}{N}\right]\rightarrow\sigma^2+\text{approximation error}\ \text{ if }\ \mathbf{a},\ \mathbf{b},\ \mathbf{c},\ \boldsymbol{\omega}\ \text{ are convergent} \quad (14)$$

Note that, in practice, the approximation error is often negligible as it is a very small finite term in comparison to the variance of noise σ^2. The solution obtained via (13) can be interpreted as the maximum likelihood estimates (MLE) under the condition that $\xi(t)$ is Gaussian.

Remarks:

- Considerable work has been conducted on the convergence issues in the identification of block-oriented nonlinear models, such as the Hammerstein and Wiener systems (Greblicki 1992; Zhang, Iouditski, and Ljung 1996; Bai and Reyland 2008, 2009; Bai and Li 2010). These are mainly based on the separability of the linear part, since the nonlinearities can be subsequently identified. Results have been established based on additional assumptions such as the statistical properties on the input signal (white noise or Gaussian) and/or the nonlinear part (e.g. monotonicity or different types of prior information). We point out that the assumptions used in a convergence proof should not limit the algorithm's application to practical systems, as these additional assumptions are sufficient but may not be necessary. This work could be extended in the further study of the proposed algorithm, if not directly applicable. Moreover, there is still the open problem on what is the least prior information for the identification of Wiener systems (Bai and Reyland 2008, 2009).
- For parameter estimation, the mean square error (MSE) criterion is more often used as it lends to the ease of implementation, e.g. MLE estimation and nonlinear least squares algorithm. Alternatively the normalised mean square error (NMSE), though not often used for parameter estimation, is commonly used to demonstrate the modelling performance.

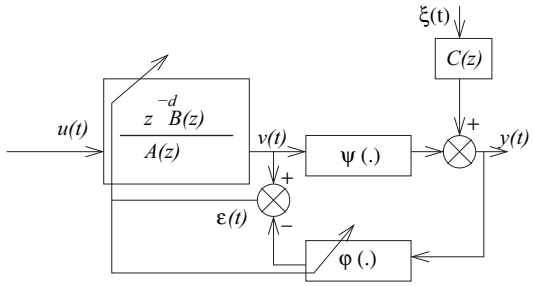

Figure 1. The initialisation for the linear filter parameter vector **a** and **b**'s.

As the objective function of (13) is highly nonlinear, the solution of the Gauss–Newton algorithm is dependent on the initial condition. It is important that **a**, **b**, **c** and **ω** are properly initialized so that they converge to an optimal solution. An initialisation scheme is proposed below.

3.1. *Initialisation of parameter vectors a and b*

The initialisation of the linear filter is illustrated with reference to Figure 1. We denote $\Psi^{-1}(\bullet)$, any of the inverse functions of $\Psi(\bullet)$, by $\varphi(\bullet)$. Consider using also a B-spline neural network for the modelling of $\varphi(\bullet)$. For convenience, we still denote the polynomial degree as k in the modelling of $\varphi(\bullet)$. The number of basis functions is denoted as d_y. A set of (d_y+k) knots is predetermined (see (7)) based on the domain of the system output $y(t)$, so that there are k external knots outside each side of the boundary of the system output $y(t)$. The model used for modelling $\varphi(\bullet)$ is

$$\hat{\varphi}(y(t))=\sum_{j=1}^{d_y}\mathcal{B}_j^{(k)}(y(t))\alpha_j \quad (15)$$

where $\alpha_j\in\mathcal{R}$, $(j=1,\ldots,d_y)$ are the associated weights. Figure 2 shows the error feedback for parameter optimisation in which $v(t)$ is used as the target for

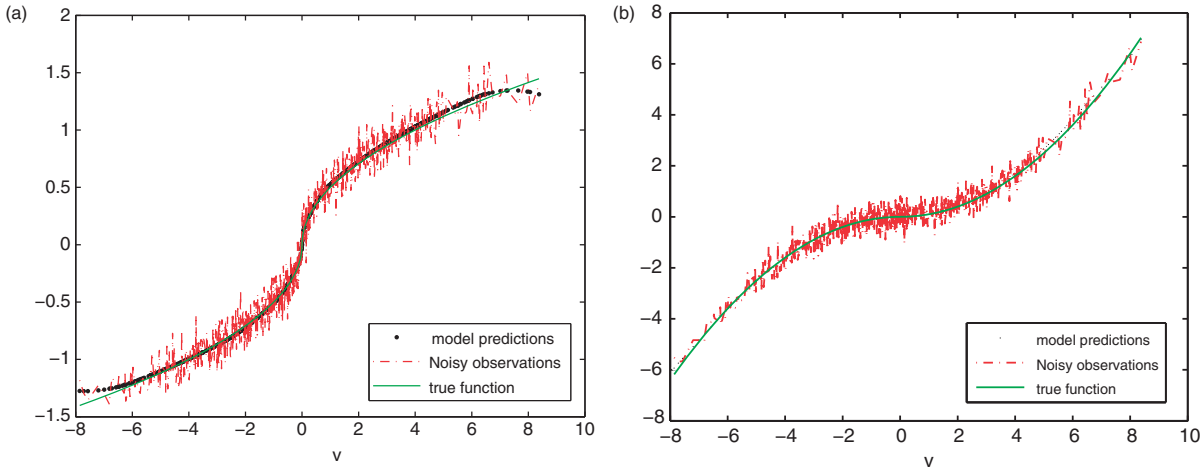

Figure 2. The modelling results (high noise) for the nonlinear function $\Psi(u)$: (a) system 1 and (b) system 2.

$\varphi(y(t))$. We denote the error between $v(t)$ and $\varphi(y(t))$ as $\epsilon(t)$ and let $n = d_y \times (n_a+1) + n_b$. Applying (1), yields

$$\begin{aligned} u(t-d) &= -\sum_{i=1}^{n_b} b_i u(t-d-i) \\ &\quad + \sum_{i=0}^{n_a} a_i \left[\left(\sum_{j=1}^{d_y} \mathcal{B}_j^{(k)}(y(t-i))\alpha_j \right] + \epsilon(t) \right. \\ &= [\mathbf{p}(\mathbf{x}(t))]^T \boldsymbol{\vartheta} + \epsilon(t) \end{aligned} \tag{16}$$

where $\mathbf{x}(t) = [-u(t-d-1), \ldots, -u(t-d-n_b), \ y(t)]^T$, $\boldsymbol{\vartheta} = [\vartheta_1, \ldots, \vartheta_n]^T = [-b_1, \ldots, -b_{n_b}, \alpha_1, \ldots, \alpha_{d_y}, \alpha_1 a_1, \ldots, \alpha_{d_y} a_{n_a}]^T \in \Re^n$. $\mathbf{p}(\mathbf{x}(t)) = [p_1(\mathbf{x}(t)), \ldots, p_n(\mathbf{x}(t))]^T = [-u(t-d-1), \ldots, -u(t-d-n_b), \ \mathcal{B}_1^{(k)}(y(t)), \ldots, \mathcal{B}_{d_y}^{(k)}(y(t)), \ \mathcal{B}_1^{(k)}(y(t-1)) \cdots, \mathcal{B}_j^{(k)}(y(t-i)), \ldots, \mathcal{B}_{d_y}^{(k)}(y(t-n_a))]^T \in \mathcal{R}^n$. Define $\alpha = [\alpha_1, \ldots, \alpha_{d_y}]^T$.

Over the training data set, (16) can be written in matrix form as

$$\mathbf{u} = \mathbf{P}\boldsymbol{\vartheta} + \epsilon \tag{17}$$

where $\mathbf{u} = [u(1-d), \ldots, u(N-d)]^T$. $\epsilon = [\epsilon(1), \ldots, \epsilon(N)]^T$, and $\mathbf{P}$ is the regression matrix $\boldsymbol{P} = [\mathbf{p}(\mathbf{x}(1)), \ldots, \mathbf{p}(\mathbf{x}(N))]^T$. The parameter vector $\boldsymbol{\vartheta}$ can be found as the least squares solution of

$$\boldsymbol{\vartheta}_{LS} = \left(\mathbf{P}^T\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{u} \tag{18}$$

This procedure produces our initial estimate of the vector $\mathbf{b}^{(0)}$, which is simply taken as the subvector of the resultant $\boldsymbol{\vartheta}_{LS}$, consisting of its first n_b elements.

In order to produce our initial estimate of the vector $\mathbf{a}^{(0)}$, the singular value decomposition (SVD) method based on the above $\boldsymbol{\vartheta}_{LS}$ (Bai 1998) is used. We rearrange the last $(n_a+1) \times d_y$ elements of $\boldsymbol{\vartheta}_{LS}$ to form the matrix given by

$$\boldsymbol{\Theta} = \alpha[1 \ \mathbf{a}^T]^T = \begin{pmatrix} \widehat{\alpha_1} & \widehat{\alpha_1 a_1} & \ddots & \widehat{\alpha_1 a_{n_a}} \\ \widehat{\alpha_2} & \widehat{\alpha_2 a_1} & \ddots & \widehat{\alpha_2 a_{n_a}} \\ \vdots & \ddots & \vdots & \\ \widehat{\alpha_{d_y}} & \widehat{\alpha_{d_y} a_1} & \ddots & \widehat{\alpha_{d_y} a_{n_a}} \end{pmatrix} \in \Re^{d_y \times (n_a+1)} \tag{19}$$

The idea in Bai (1998) is to determine $\boldsymbol{\alpha}$ and $\mathbf{a}$ from an overparameterised vector $\boldsymbol{\Theta}$ by minimising

$$\|\boldsymbol{\Theta} - \alpha[1 \ \mathbf{a}^T]^T\|_F^2.$$

Construct the SVD of $\boldsymbol{\Theta} = \sum_{i=1}^{\min(d_y, n_a+1)} \sigma_i \boldsymbol{\mu}_i \boldsymbol{\nu}_i^T$, where $\boldsymbol{\mu}_i (i = 1, \ldots, d_y)$ and $\boldsymbol{\nu}_i (i = 1, \ldots, (n_a+1))$ are orthonormal vectors. Let $\boldsymbol{\nu}_1 = [\nu_{1,1}, \nu_{2,1}, \ldots, \nu_{n_a+1,1}]^T$. Using the fact that $\boldsymbol{\Theta}$ has rank one, we obtain

$$\mathbf{a}^{(0)} = [\nu_{2,1}, \ldots, \nu_{n_a+1,1}]^T / \nu_{1,1}. \tag{20}$$

We can also obtain $\boldsymbol{\alpha} = \sigma 1 \nu_{1,1} \boldsymbol{\mu}_1$, but it is no longer used in the remainder of our algorithm. Note that we used the constraint $a_0 = 1$ to get the unique solution that is different from Bai (1998) due to the different constraints.

3.2. *The initialisation of parameter vectors and c and ω*

Consider initially generating an auxiliary signal $\{\hat{v}(t)\}_{t=1}^N$ over training data set $\{u(t), y(t)\}_{t=1}^N$, based on the initialised parameter estimates $\hat{\mathbf{b}}^{(0)}$ and $\hat{\mathbf{a}}^{(0)}$, as

$$\begin{aligned} \hat{v}(t) &= u(t-d) + \hat{b}_1^{(0)} u(t-d) + \cdots + \hat{b}_{n_b}^{(0)} u(t-d-n_b) \\ &\quad - a_1^{(0)} \hat{v}(t-1) - a_2^{(0)} \hat{v}(t-2) - \cdots - a_{n_a}^{(0)} \hat{v}(t-n_a). \end{aligned} \tag{21}$$

Then a block of training data set $\{\hat{v}(t), y(t)\}_{t=1}^{N}$ is used for the initialisation of parameter vectors $\mathbf{c}$ and $\boldsymbol{\omega}$. Using model form (2), in which $v(t)$ is replaced by its estimates $\hat{v}(t)$, and $\xi(t)$ replaced by $e(t)$, we have

$$\begin{aligned} y(t) &= \sum_{j=1}^{M} \mathcal{B}_j^{(k)}(\hat{v}(t))\omega_j + \sum_{j=1}^{n_c} c_j e(t-j) + e(t) \\ &= [\mathbf{q}(\hat{v}(t))]^T\boldsymbol{\omega} + [\tilde{\mathbf{e}}(t)]^T\mathbf{c} + e(t) \end{aligned} \tag{22}$$

where $\mathbf{q}(\hat{v}(t)) = [q_1(\hat{v}(t)), \ldots, q_M(\hat{v}(t))]^T = [\mathcal{B}_1^{(k)}(\hat{v}(t)), \ldots, \mathcal{B}_M^{(k)}(\hat{v}(t))]^T \in \Re^M$ and $\tilde{\mathbf{e}}(t) = [e(t-1), \ldots, e(t-n_c)]^T \in \Re^{n_c}$. Over the training data set, (22) can be written in matrix form

$$\mathbf{y} = [\mathbf{Q} \;\; \tilde{\mathbf{E}}]\begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{c} \end{bmatrix} + \mathbf{e} \tag{23}$$

where $\mathbf{Q} = [\mathbf{q}(\hat{v}(1)), \ldots, \mathbf{q}(\hat{v}(N))]^T$, $\tilde{\mathbf{E}} = [\tilde{\mathbf{e}}(1), \ldots, \tilde{\mathbf{e}}(N)]^T$ and $\mathbf{y} = [y(1), \ldots, y(N)]^T$. The least squares solution of $\boldsymbol{\omega}$ and $\mathbf{c}$ are obtained from the following iterations:

(1) Initialise

$$\begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{c} \end{bmatrix}_{(0)} = \begin{bmatrix} (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{y} \\ \mathbf{0} \end{bmatrix} \tag{24}$$

where $\mathbf{0}$ is a zero vector of size n_c. Calculate $\tilde{\mathbf{e}}_{(0)} = \mathbf{y} - \mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{y}$, and form $\tilde{\mathbf{E}}_{(0)}$. Set the iteration step $\tau = 1$.

(2) Calculate the least square solution given by

$$\begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{c} \end{bmatrix}_{(\tau)} = \left(\begin{bmatrix} \mathbf{Q}^T \\ \tilde{\mathbf{E}}_{(\tau-1)}^T \end{bmatrix} [\mathbf{Q} \;\; \tilde{\mathbf{E}}_{(\tau-1)}] \right)^{-1} \begin{bmatrix} \mathbf{Q}^T\mathbf{y} \\ \tilde{\mathbf{E}}_{(\tau-1)}^T\mathbf{y} \end{bmatrix} \tag{25}$$

(3) Calculate

$$\mathbf{e}_{(\tau)} = \mathbf{y} - [\mathbf{Q} \;\; \tilde{\mathbf{E}}_{(\tau-1)}]\begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{c} \end{bmatrix}_{(\tau)} \tag{26}$$

and form $\tilde{\mathbf{E}}_{(\tau)}$. Set $\tau = \tau + 1$, repeat steps 1 to 3 until a predetermined iteration number (e.g. 10) has reached. Our initial value for $\boldsymbol{\omega}^{(0)}$ and $\mathbf{c}^{(0)}$ is taken from the final value of $\left[\begin{smallmatrix} \omega \\ \mathbf{c} \end{smallmatrix}\right]_{(\tau)}$.

3.3. *Gauss–Newton algorithm combined with the De Boor recursion*

Note that the initial parameter estimates obtained so far are only near to, but not optimal in minimising (13). This is because the regressors in (15) are subject to the output noise, which will in general propagate to the parameter estimates, yielding biased parameters (Hagenblad et al. 2008). In order to find the optimal value to minimise (13), the Gauss–Newton algorithm can be applied. We denote $\boldsymbol{\theta} = [\boldsymbol{\omega}^T \; \mathbf{a}^T \; \mathbf{b}^T \; \mathbf{c}^T]^T$, and an iteration step variable m by a superscript (m). With an initial $\boldsymbol{\theta}^{(0)}$ (Sections 3.1–3.2), the Gauss–Newton algorithm is given by

$$\boldsymbol{\theta}^{(m)} = \boldsymbol{\theta}^{(m-1)} - \alpha\{[\mathbf{J}^{(m)}]^T\mathbf{J}^{(m)}\}^{-1}[\mathbf{J}^{(m)}]^T\mathbf{e}(\boldsymbol{\theta}^{(m-1)}) \tag{27}$$

where $\mathbf{J} = [\mathbf{J}_{\boldsymbol{\omega}} \; \mathbf{J}_{\mathbf{a}} \; \mathbf{J}_{\mathbf{b}} \; \mathbf{J}_{\mathbf{c}}]$ is the Jacobian of $\mathbf{e}(\boldsymbol{\theta})$, and

$$\mathbf{J}_{\boldsymbol{\omega}} = -\begin{bmatrix} \mathcal{B}_1^{(k)}(v(1)) & \cdots & \mathcal{B}_M^{(k)}(v(1)) \\ \mathcal{B}_1^{(k)}(v(2)) & \cdots & \mathcal{B}_M^{(k)}(v(2)) \\ \vdots & \ddots & \vdots \\ \mathcal{B}_1^{(k)}(v(N)) & \cdots & \mathcal{B}_M^{(k)}(v(N)) \end{bmatrix} \tag{28}$$

$$\mathbf{J}_{\mathbf{a}} = \begin{bmatrix} v(0)f(1) & \cdots & v(1-n_a)f(1) \\ v(1)f(2) & \cdots & v(2-n_a)f(2) \\ \vdots & \ddots & \vdots \\ v(N-1)f(N) & \cdots & v(N-n_a)f(N) \end{bmatrix} \tag{29}$$

$$\mathbf{J}_{\mathbf{b}} = -\begin{bmatrix} u(-d)f(1) & \cdots & u(1-d-n_b)f(1) \\ u(1-d)f(2) & \cdots & u(2-d-n_b)f(2) \\ \vdots & \ddots & \vdots \\ u(N-d)f(N) & \cdots & u(N-d-n_b)f(N) \end{bmatrix} \tag{30}$$

$$\mathbf{J}_{\mathbf{c}} = -\begin{bmatrix} e(0, \boldsymbol{\theta}^{(m-1)}) & \cdots & e(1-n_c, \boldsymbol{\theta}^{(m-1)}) \\ e(1, \boldsymbol{\theta}^{(m-1)}) & \cdots & e(2-n_c, \boldsymbol{\theta}^{(m-1)}) \\ \vdots & \ddots & \vdots \\ e(N-1, \boldsymbol{\theta}^{(m-1)}) & \cdots & e(N-n_c, \boldsymbol{\theta}^{(m-1)}) \end{bmatrix} \tag{31}$$

where $f(t) = \sum_{j=1}^{M} \frac{\mathrm{d}}{\mathrm{d}v}[\mathcal{B}_j^{(k)}(v(t))]\omega_j$, $t = 1, \ldots, N$, $\alpha > 0$ is a small positive step size. Note that in calculating (28)–(31), the De Boor algorithm (8)–(10) is applied in evaluating all entries. In particular, we point out the term $\frac{\mathrm{d}}{\mathrm{d}v}[\mathcal{B}_j^{(k)}(v(t))]$ using (10), it gives exact values at minimum extra computational cost (this is an advantage specific to our B-spline functions with the De Boor recursion, but not Hughes and Westwick (2005) and Zhu (1999). Effectively, this enables stable and efficient evaluations of B-spline functional and derivative values to be possible, which could be problematic for many other nonlinear representation including some spline function-based nonlinear models. The above iteration can be terminated when $\boldsymbol{\theta}^{(m)}$ converges, or by predetermining a sufficiently large number of iterations.

3.4. *A summary of the system identification algorithm*

The system identification algorithm can be summarised as follows.

(1) Predetermine a set of $(d_y + k)$ knots that break the domain of system output $y(t)$ up, with k knots satisfying the condition of being external to system output region at each end.
(2) Form $\mathbf{P}$ and $\mathbf{u}$, and apply (18). Obtain $\mathbf{a}^{(0)}$ and $\mathbf{b}^{(0)}$ using (18)–(20).
(3) Construct an auxiliary signal $\{\hat{v}(t)\}_{t=1}^{N}$ based on (21).
(4) Predetermine a set of $(M + k)$ knots that break the domain $v(t)$ up, with k knots satisfying the condition of being external to region of $v(t)$ at each end.
(5) Obtain $\boldsymbol{\omega}^{(0)}$ and $\mathbf{c}^{(0)}$ using (25)–(26).
(6) Apply the Gauss–Newton algorithm combined with the De Boor recursion (27)–(31).

4. Numerical examples

Two Wiener systems are simulated, in which the linear subsystems are the same for both systems, as $A(z) = 1 - 1.2z^{-1} + 0.5z^{-2}$, $B(z) = 1 + 0.3z^{-1} + 0.8z^{-2} + 0.07z^{-3}$, $C(z) = 1 + 0.3z^{-1}$ and $d = 2$. For the nonlinear subsystem, $\Psi(\bullet)$ is given by

$$\text{System 1: } \Psi(v) = 0.5\,\text{sign}(v)\sqrt{|v|} \tag{32}$$

$$\text{System 2: } \Psi(v) = 0.1\,\text{sign}(v)v^2 \tag{33}$$

respectively. For each system, 1000 training data samples $y(t)$ were generated by using (1) and (2), where $u(t)$ was uniformly distributed random variable $u(t) \in [-1.5, 1.5]$. For system 1, the variances of the additive noise to the system output are set as 0.01^2 (low noise) and 0.1^2 (high noise), respectively. The low/high noise cases correspond to a signal/noise ratio at the output of 37.9329 dB/17.9329 dB, respectively. For system 2, the variances of the additive noise to the system output are set as 0.03^2 (low noise) and 0.3^2 (high noise), respectively. The low/high noise cases correspond to a signal/noise ratio at the output of 33.9050 dB/13.9050 dB, respectively. The polynomial degree of B-spline basis functions was set as two ($k = 3$, piecewise quadratic). The system identification algorithm outlined in Section 3.4 was carried out for both systems with the following predetermined knot sequences.

For system 1, the knot sequence

$$[-2.5, -2, -1.75, -1.25, -1, -0.5, -0.25, 00.25, 0.5, 1, 1.25, 1.75, 2, 2.5]$$

is initially set for $y(t)$ in order to generate basis functions used in (15). The knot sequence

$$[-14, -12, -9, -6, -3, -1, -0.1, 0, 0.1, 1, 3, 6, 9, 12, 14]$$

is used for $v(t)$ in order to generate basis functions used in (11).

For system 2, the knot sequence

$$[-2.5, -2, -1.75, -1.25, -1, -0.5, -0.25, 00.25, 0.5, 1, 1.25, 1.75, 2, 2.5]$$

is initially set for $y(t)$ in order to generate basis functions used in (15). The knot sequence

$$[-14, -12, -9, -8, -6, -3, -1, 0, 1, 3, 6, 8, 9, 12, 14]$$

is used for $v(t)$ in order to generate basis functions used in (11).

The modelling results are shown in Tables 1 and 2 for the two linear subsystems. It is shown that the proposed system identification method is particularly effective at high-output noise level for both systems. Figure 2(a) and (b) show the excellent approximation results for the nonlinear static functions using the B-spline models.

Table 1. Results of linear subsystem parameter estimation (system 1): Panel A – low noise and Panel B – high noise.

	True parameters	Initial estimates	Final estimates
Panel A			
a_1	−1.2	−1.1551	−1.2010
a_2	0.5	0.5190	0.5003
b_1	0.3	0.3068	0.2983
b_2	0.8	0.8030	0.7991
b_3	0.07	0.0766	0.0678
c_1	0.3	0.6177	0.2429
Panel B			
a_1	−1.2	−0.5081	−1.2027
a_2	0.5	−0.2325	0.4997
b_1	0.3	0.4361	0.3011
b_2	0.8	0.7206	0.7915
b_3	0.07	0.2933	0.0634
c_1	0.3	0.8041	0.2959

5. Conclusions

The system identification system is investigated by using the B-spline neural network to model the nonlinear static function in the Wiener system. The parameter estimation is based on the Gauss–Newton algorithm combining with the De Boor algorithm for both curve and the first-order derivatives, following the use of a proposed parameter initialisation scheme. The efficacy of the proposed approach have been demonstrated via numerical examples.

Table 2. Results of linear subsystem parameter estimation (system 2): Panel A – low noise and Panel B – high noise.

	True parameters	Initial estimates	Final estimates
Panel A			
a_1	−1.2	−0.5539	−1.1962
a_2	0.5	0.1249	0.4995
b_1	0.3	0.0963	0.2943
b_2	0.8	0.1564	0.8049
b_3	0.07	0.0626	0.0918
c_1	0.3	0.8023	0.3369
Panel B			
a_1	−1.2	−0.3600	−1.1934
a_2	0.5	0.0491	0.4944
b_1	0.3	0.0841	0.2801
b_2	0.8	0.0986	0.8369
b_3	0.07	0.0502	0.0313
c_1	0.3	0.7988	0.3004

Acknowledgements

X. Hong gratefully acknowledges that part of this work was supported by EPSRC in the UK. We also thank the reviewers for their valuable comments.

Notes on contributors

X. Hong received her university education at National University of Defense Technology, P.R. China (BSc, 1984, MSc, 1987), and University of Sheffield, UK (PhD, 1998), all in Automatic Control. She worked as a Research Assistant in Beijing Institute of Systems Engineering, Beijing, China from 1987 to 1993. She worked as a Research fellow in the Department of Electronics and Computer Science at University of Southampton from 1997 to 2001. She is currently a Reader of Computational Intelligence at School of Systems Engineering, University of Reading. She is actively engaged in research into nonlinear systems identification, data modelling, estimation and intelligent control, neural networks, pattern recognition, learning theory and their applications. She has published over 100 research papers and coauthored a research book. She was awarded a Donald Julius Groen Prize by IMechE in 1999.

R.J. Mitchell received his BSc (Hons) degree in Cybernetics and Control Engineering and the PhD degree in Cybernetics from the Department of Cybernetics, University of Reading, Reading, UK, in 1980 and 1987, respectively. He was appointed Lecturer in Cybernetics in 1983 and is now a Senior Lecturer in Cybernetics and also Senior Tutor in the School of Systems Engineering, University of Reading. He has published four textbooks, edited a custom book on Cybernetics and has over 100 research papers in control engineering, robotics and learning systems.

S. Chen received his BEng degree in Control Engineering from the East China Petroleum Institute in 1982 and the PhD degree in Control Engineering from the City University at London in 1986. In 2005, he was awarded the Doctor of Sciences (DSc) degree by the University of Southampton. He joined the School of Electronics and Computer Science, the University of Southampton in September 1999. He previously held research and academic appointments at the Universities of Sheffield, Edinburgh and Portsmouth. He is a Chartered Engineer (CEng), a Fellow of IET (FIET) and a Fellow of IEEE (FIEEE). His research interests are in adaptive signal processing for communications, wireless communications, modelling and identification of nonlinear systems, learning theory and neural networks, finite-precision digital controller design and networked control systems, evolutionary computation methods and optimisation. In the database of the world's most highly cited researchers in various disciplines, compiled by Institute for Scientific Information (ISI) of the USA, Professor Chen is on the list of the highly cited researchers in the engineering category.

References

Bai, E.W. (1998), 'An Optimal Two-stage Identification Algorithm for Hammerstein–Wiener Nonlinear Systems', *Automatica*, 34, 333–338.

Bai, E.W., and Fu, M.Y. (2002), 'A Blind Approach to Hammerstein Model Identification', *IEEE Transactions on Signal Processing*, 50, 1610–1619.

Bai, E.W., and Li, K. (2010), 'Convergence of the Iterative Algorithm for a General Hammerstein System Identification', *Automatica*, 46, 1891–1896.

Bai, E.W., and Reyland, J. (2008), 'Toward Identification of Wiener Systems with the Least Amount of a Prior Information on the Nonlinearity', *Automatica*, 44, 910–919.

Bai, E.W., and Reyland, J. (2009), 'Toward Identification of Wiener Systems with the Least Amount of a Prior Information: IIR Cases', *Automatica*, 45, 956–964.

Balestrino, A., Landi, A., Ould-Zmirli, M., and Sani, L. (2001), 'Automatic Nonlinear Autotuning Method for Hammerstein Modelling of Electrical Drives', *IEEE Transactions on Industrial Electronics*, 48, 645–655.

Billings, S.A., and Fakhouri, S.Y. (1979), 'Nonlinear System Identification using the Hammerstein Model', *International Journal of Systems Science*, 10, 567–578.

Bloemen, H.H.J., Van Den Boom, T.J., and Verbruggen, H.B. (2001), 'Model-based Predictive Control for Hammerstein–Wiener Systems', *International Journal of Control*, 74, 482–295.

Brown, M., and Harris, C.J. (1994), *Neurofuzzy Adaptive Modelling and Control*, Hemel Hempstead: Prentice Hall.

Chaoui, F.Z., Giri, F., Rochdi, Y., Haloua, M., and Naitali, A. (2005), 'System Identification Based Hammerstein Model', *International Journal of Control*, 78, 430–442.

Chen, H.F. (2004), 'Pathwise Convergence of Recursive Identification Algorithms for Hammerstein Systems', *IEEE Transactions on Automatic Control*, 49, 1873–1896.

De Boor, C. (1978), *A Practical Guide to Splines*, New York: Spring Verlag.

Farin, G. (1994), *Curves and Surfaces for Computer-aided Geometric Design: A Practical Guide*, Boston: Academic Press.

Farouki, R.T., and Goodman, T.N.T. (1996), 'On the Optimal Stability of the Bernstein Basis', *Mathematics of Computation*, 65, 1553–1566.

Gomez, J.C., Jutan, A., and Baeyens, E. (2004), 'Wiener Model Identification and Predictive Control of a pH Neutralisation Process', *IEE Proceedings – Control Theory and Applications*, 151, 329–338.

Greblicki, W. (1989), 'Nonparametric Orthogonal Series Identification of Hammerstein Systems', *International Journal of Systems Science*, 20, 2355–2367.

Greblicki, W. (1992), 'Nonparametric Identification of Wiener Systems', *IEEE Transactions on Information Theory*, 38, 1487–1493.

Greblicki, W. (2002), 'Stochastic Approximation in Nonparametric Identification of Hammerstein Systems', *IEEE Transactions on Automatic Control*, 47, 1800–1810.

Greblicki, W., and Pawlak, M. (1986), 'Identification of Discrete Hammerstein Systems using Kernel Regression Estimate', *IEEE Transactions on Automatic Control*, AC-31, 74–77.

Hagenblad, A., Ljung, L., and Wills, A. (2008), 'Maximum Likelihood Identification of Wiener Models', *Automatica*, 44, 2697–2705.

Harris, C.J., Hong, X., and Gan, Q. (2002), *Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach*, Berlin: Springer-Verlag.

Hong, X., and Mitchell, R.J. (2007), 'A Hammerstein Model Identification Algorithm using Bezier–Bernstein Approximation', *IEE Proceedings: Control Theory and Applications*, 1, 1149–1159.

Hsu, K., Vincent, T., and Poolla, K. (2006), 'A Kernel-Based Approach to Structured Nonlienar System Identification Part I: Algorithms, Part II: Convergence and Consistency', in *Proceedings of 14th IFAC Symposium on System Identification*, Australia.

Hughes, M.C., and Westwick, D.T. (2005), 'Identification of IIR Wiener System with Spline Nonliearitis that have Variable Knots', *IEEE Transactions on Automatic Control*, 50, 1617–1622.

Hunter, I.W., and Korenberg, M.J. (1986), 'The Identification of Nonlinear Biological Systems: Wiener and Hammerstein Cascade Models', *Biological Cybernetics*, 55, 135–144.

Kalafatis, A.D., Arifinand, N., Wang, L., and Cluett, W.R. (1995), 'A New Approach to the Identification of pH Processes on the Wiener Model', *Chemical Engineering Science*, 50, 3693–3701.

Kalafatis, A.D., Wang, L., and Cluett, W.R. (1997), 'Identification of Wiener-type Nonlinear Systems in a Noisy Environment', *International Journal of Control*, 66, 923–941.

Kavli, T. (1993), 'ASMOD – an Algorithm for Adaptive Spline Modelling of Observation Data', *International Journal of Control*, 58, 947–967.

Lang, Z.Q. (1997), 'A Nonparametric Polynomial Identification Algorithm for the Hammerstein System', *IEEE Transactions on Automatic Control*, 42, 1435–1441.

Schoukens, J., Nemeth, J.G., Crama, P., Rolain, Y., and Pintelon, R. (2003), 'Fast Approximate Identification of Nonliear Systems', *Automatica*, 39, 1267–1274.

Skrjanc, I., Blazic, S., and Agamennoni, O.E. (2005), 'Interval Fuzzy Modelling Applied to Wiener Models with Uncertainties', *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics*, 35, 1092–1095.

Soderström, T., and Stoica, P. (1989), *System Identification*, Upper Saddle River, NJ: Prentice Hall.

Stoica, P., and Söderström, T. (1982), 'Instrumental Variable Methods for Identification of Hammerstein Systems', *International Journal of Control*, 35, 459–476.

Turunen, J., Tanttu, J.T., and Loula, P. (2003), 'Hammerstein Model for Speech Coding', *EURASIP Journal of Applied Signal Processing*, 12, 1238–1249.

Verhaegen, M., and Westwick, D. (1996), 'Identifying MIMO Hammerstein Systems in the Context of Subspace Model Identification', *International Journal of Control*, 63, 331–349.

Westwick, D. (1996), 'Identifying MIMO Wiener Systems using Subspace Model Identification Model Identification Methods', *Signal Processing*, 52, 235–258.

Wigren, T. (1993), 'Recursive Prediction Error Identification using the Nonlinear Wiener Model', *Automatica*, 29, 1011–1025.

Wigren, T. (1994), 'Convergence Analysis of Recursive Identification Algorithms Based on the Nonlinear Wiener Model', *IEEE Transactions on Automatic Control*, 39, 2191–2206.

Zhang, Q., Iouditski, A., and Ljung, L. (1996), 'Identification of Wiener System with Monotonous Nonliearity', in *IFAC Symposium on System Identification*, Newcastle, Australia, pp. 166–171.

Zhu, Y. (1999), 'Distillution Column Identification for Control using Wiener Model', in *Proceedings of the American Control Conference*, San Diego, CA, USA, pp. 3462–3466.

Zhu, Y. (2002), 'Estimation of an N–L–N Hammerstein–Wiener Model', *Automatica*, 38, 1607–1614.