CHAPTER 1

COMPLEX-VALUED B-SPLINE NEURAL NETWORKS FOR MODELLING AND INVERSE OF WIENER SYSTEMS

Xia Hong[1], Sheng Chen[2,3] and Chris J. Harris[2]

[1] University of Reading, Reading RG6 6AY, UK
[2] University of Southampton, Southampton SO17 1BJ, UK
[3] King Abdulaziz University, Jeddah 21589, Saudi Arabia

1.1 INTRODUCTION

Communication signal processing applications often involve complex-valued (CV) functional representations for signals and systems. CV artificial neural networks have been studied theoretically and applied widely in nonlinear signal and data processing [1–11]. Note that most artificial neural networks cannot be automatically extended from the real-valued (RV) domain to the CV domain because the resulting model would in general violate Cauchy-Riemann conditions, and this means that the training algorithms become unusable. A number of analytic functions were introduced for the fully CV multilayer perceptrons (MLP) [4]. A fully CV radial basis function (RBF) nework was introduced in [8] for regression and classification applications. Alternatively, the problem can be avoided by using two RV artificial neural networks, one processing the real part and the other processing the imaginary part of the CV signal/system. A even more challenging problem is the inverse of a CV nonlinear system, which is typically found in practical applications. This is an under-researched area, and a few existing methods, such as the algorithm proposed in [10], are not very effective in tackling practical CV signal processing problems. In order to develop an efficient approach for modelling and inverse of CV Wiener systems, we have turned to the RV signal processing field for motivations and inspirations.

A popular approach to nonlinear systems identification in the RV domain is to use block-oriented nonlinear models which comprise the linear dynamic models and static or memoryless nonlinear functions [12–17]. Specifically, the Wiener model, which comprises a linear dynamical model followed by a nonlinear static transformation, offers a reasonable model for linear systems with a nonlinear measurement device that are widely found in industrial and biological systems [18–23]. The model representation of the unknown nonlinear static function in the Wiener model is fundamental to its identification, control and/or other applications. Various approaches have been developed in order to capture the *a priori* unknown nonlinearity in the Wiener system, including the nonparametric method [24], subspace model identification methods [22], fuzzy modelling [25] and the parametric method [13, 20, 21]. With its best conditioning property, the B-spline curve has been widely used in computer graphics and computer aided geometric design [26]. The B-spline curves consist of many polynomial pieces, offering versatility. In particular, the De Boor algorithm [27], which uses numerically stable recurrence relations, offers a highly efficient means of constructing B-spline curve. The B-spline basis functions for RV nonlinear systems modelling have been widely applied [28–31].

Many practical communication applications involve propagating CV signals through CV nonlinear dynamic systems that can be represent by the Wiener model. For example, at the transmitter of broadband communication systems, the transmitted signal is distorted by the high power amplifier (HPA) with memory that can be characterised by the CV Wiener model [32, 33]. Also some nonlinear communication channels can usually be represented by a finite duration impulse response (FIR) filter followed by a CV static nonlinear function, namely, a CV Wiener model. Accurate identification of a CV Wiener model is often the first successful step in these applications. Moreover, an accurate inverse of the estimated CV Wiener model is required, such as in digital predistorter design for compensating the distortions of the Wiener HPA at the transmitter [34–39] and deconvolution or equalisation at the receiver [2,3]. Our previous work [40] has developed an efficient B-spline neural network approach for general modelling of CV Wiener systems, which represents the CV nonlinear static function in the Wiener system using the tensor product from two univariate B-spline neural networks. This novel approach is different from the existing CV neural network based on spline functions [3, 41, 42], in both model representation and identification algorithms [40]. By minimising the mean square error (MSE) between the model output and the system output, the Gauss-Newton algorithm, coupled with a simple least squares (LS) parameter initialisation, is readily applicable for the parameter estimation in the proposed CV model, which naturally incorporates the De Boor recursions for both the B-spline curves and first order derivatives.

The significance of the proposed method [40] is twofold. Firstly, it extends the B-spline model to accommodate general CV Wiener systems. Secondly the proposed model based on B-spline functions has a significant advantage over many other modelling paradigms in that it enables stable and efficient evaluations of functional and derivative values, as required in the Gauss-Newton optimisation algorithm. The additional contribution of our current work is to develop an effective inverse of the CV Wiener system so as to complete the whole task for identification and inverse of the generic Wiener system. We demonstrate that the B-spline neural network scheme for modelling of CV Wiener systems proposed in [40] has a further advantage in that an accurate inverse of the CV Wiener system can directly be achieved from the estimated Wiener model in an very efficient way. In particular, the inverse of the CV nonlinear static function in the Wiener model is calculated effectively using the Gauss-Newton algorithm based on the inverse of De Boor algorithm, which again utilises naturally the B-spline curve and first order derivative recursions. The effectiveness of the proposed approach for identification and inverse of CV Wiener systems is illustrated using the application of digital predistorter design for broadband communication systems that employ power-efficient nonlinear HPA transmitter.

1.2 IDENTIFICATION AND INVERSE OF COMPLEX-VALUED WIENER SYSTEMS

Throughout this contribution, a CV number $x \in \mathbb{C}$ is represented either by the rectangular form $x = x_R + \mathrm{j}x_I$, where $\mathrm{j} = \sqrt{-1}$, while $x_R = \Re[x]$ and $x_I = \Im[x]$ denote the real and imaginary parts of x, or alternatively by the polar form $x = |x| \cdot \exp(\mathrm{j}\angle^x)$ with $|x|$ denoting the amplitude of x and $\angle^x$ its phase.

1.2.1 The complex-valued Wiener system

The generic CV Wiener system considered in this study consists of a cascade of two subsystems, an FIR filter of order L that represents the memory effect on the input signal $x(k) \in \mathbb{C}$, followed by a nonlinear memoryless function $\Psi(\bullet) : \mathbb{C} \to \mathbb{C}$. The system is represented by

$$w(k) = \sum_{i=0}^{L} h_i x(k-i), \; h_0 = 1, \tag{1.1}$$

$$y(k) = \Psi\left(w(k)\right) + \xi(k), \tag{1.2}$$

where $y(k) \in \mathbb{C}$ is the system output, and $\xi(k)$ is a CV white noise sequence independent of $x(k)$ and with $E\left[|\xi_R(k)|^2\right] = E\left[|\xi_I(k)|^2\right] = \sigma_\xi^2$. The z transfer function of the FIR filter is defined by

$$H(z) = \sum_{i=0}^{L} h_i z^{-i}, \ h_0 = 1, \tag{1.3}$$

with the CV coefficient vector given by $\mathbf{h} = [h_1 \ h_2 \cdots h_L]^{\mathrm{T}} \in \mathbb{C}^L$. Note that, without loss of generality, we assume that $h_0 = 1$. If this is not the case, h_0 can always be absorbed into the CV static nonlinearity $\Psi(\bullet)$, and the linear filter's coefficients are re-scaled as h_i/h_0 for $0 \leq i \leq L$.

Without lose of generality, the following assumptions are made regarding the CV Wiener system (1.1) and (1.2).

Assumption 1: $\Psi(\bullet)$ is a one to one mapping, i.e. it is an invertible and continuous function.

Assumption 2: $y_R(k)$, $y_I(k)$, $w_R(k)$, $w_I(k)$ $x_R(k)$ and $x_I(k)$ are upper and lower bounded by some finite real values.

For practical applications, these two assumptions typically hold. Our aim is to identify the above Wiener system, i.e. given the input-output data set $D_N = \{x(k), y(k)\}_{k=1}^{K}$, to identify the underlying nonlinear function $\Psi(\bullet)$ and to estimate the FIR filter parameters $\mathbf{h}$, as well as to provide an accurate inverse of the above Wiener system based on the identified model. Note that the signal $w(k)$ between the two subsystems are unavailable. We will use the CV B-spline neural network approach proposed in [40] for an efficient identification of this Wiener system and then develop an effective algorithm for an accurate inverse of this Wiener system based on the estimated Wiener model $\widehat{\Psi}(\bullet)$ and $\widehat{\mathbf{h}}$.

1.2.2 Complex-valued B-spline neural network

The CV B-spline neural network proposed in [40] is adopted to represent the mapping $\widehat{y} = \widehat{\Psi}(w_R + \mathrm{j}w_I) : \mathbb{C} \rightarrow \mathbb{C}$ that is the estimate of the underlying CV nonlinear function $\Psi(\bullet)$. Assume that $U_{\min} < w_R < U_{\max}$ and $V_{\min} < w_I < V_{\max}$, where $U_{\min}$, $U_{\max}$, $V_{\min}$ and $V_{\max}$ are known finite real values.

A set of univariate B-spline basis functions based on w_R is parametrised by the order $(P_o - 1)$ of a piecewise polynomial and 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 N_R basis functions. Then the knot vector is specified by $(N_R + P_o + 1)$ knot values, $\{U_0, U_1, \cdots, U_{N_R+P_o}\}$, with

$$\begin{aligned} U_0 < U_1 < \cdots < U_{P_o-2} < U_{P_o-1} = U_{\min} < U_{P_o} < \cdots \\ < U_{N_R} < U_{N_R+1} = U_{\max} < U_{N_R+2} < \cdots < U_{N_R+P_o}. \end{aligned} \tag{1.4}$$

At each end, there are $P_o - 1$ external knots that are outside the input region and one boundary knot. As a result, the number of internal knots is $N_R + 1 - P_o$. Given the set of predetermined knots (1.4), the set of N_R B-spline basis functions can be formed by using the De Boor recursion [27], yielding

$$B_l^{(\Re,0)}(w_R) = \begin{cases} 1, & \text{if } U_{l-1} \le w_R < U_l, \\ 0, & \text{otherwise,} \end{cases} \quad 1 \le l \le N_R + P_o, \tag{1.5}$$

$$B_l^{(\Re,p)}(w_R) = \frac{w_R - U_{l-1}}{U_{p+l-1} - U_{l-1}} B_l^{(\Re,p-1)}(w_R) + \frac{U_{p+l} - w_R}{U_{p+l} - U_l} B_{l+1}^{(\Re,p-1)}(w_R),$$
$$\text{for } l = 1, \cdots, N_R + P_o - p \text{ and } p = 1, \cdots, P_o. \tag{1.6}$$

The derivatives of the B-spline basis functions $B_l^{(\Re,P_o)}(w_R)$ for $1 \le l \le N_R$ can also be computed recursively according to

$$\begin{aligned} \frac{dB_l^{(\Re,P_o)}(w_R)}{dw_R} =& \frac{P_o}{U_{P_o+l-1} - U_{l-1}} B_l^{(\Re,P_o-1)}(w_R) \\ & - \frac{P_o}{U_{P_o+l} - U_l} B_{l+1}^{(\Re,P_o-1)}(w_R). \end{aligned} \tag{1.7}$$

Similarly, a set of univariate B-spline basis functions based on w_I can be established. Suppose that the order of the piecewise polynomial is again predetermined as $(P_o - 1)$ and there are N_I basis functions. Then the knot vector is defined on the imaginary line in a similar manner, which is specified by the $(N_I + P_o + 1)$ knot values, $\{V_0, V_1, \cdots, V_{N_I+P_o}\}$. Specifically,

$$\begin{aligned} V_0 < V_1 < \cdots < V_{P_o-2} < V_{P_o-1} = V_{\min} < V_{P_o} < \cdots \\ < V_{N_I} < V_{N_I+1} = V_{\max} < V_{N_I+2} < \cdots < V_{N_I+P_o}. \end{aligned} \tag{1.8}$$

Again, at each end, there are $P_o - 1$ external knots that are outside the input region and one boundary knot. Consequently, the number of internal knots is $N_I + 1 - P_o$. Similarly, the set of N_I B-spline basis functions are constructed by the De Boor recursion [27] as

$$B_m^{(\Im,0)}(w_I) = \begin{cases} 1, & \text{if } V_{m-1} \le w_I < V_m, \\ 0, & \text{otherwise,} \end{cases} \quad 1 \le m \le N_I + P_o, \tag{1.9}$$

$$B_m^{(\Im,p)}(w_I) = \frac{w_I - V_{m-1}}{V_{p+m-1} - V_{m-1}} B_m^{(\Im,p-1)}(w_I) + \frac{V_{p+m} - w_I}{V_{p+m} - V_m} B_{m+1}^{(\Im,p-1)}(w_I),$$
$$\text{for } m = 1, \cdots, N_I + P_o - p \text{ and } p = 1, \cdots, P_o, \tag{1.10}$$

while the derivatives of the B-spline basis functions $B_m^{(\Im,P_o)}(w_I)$ for $1 \le m \le N_I$ are computed recursively according to

$$\begin{aligned} \frac{dB_m^{(\Im,P_o)}(w_I)}{dw_I} =& \frac{P_o}{V_{P_o+m-1} - V_{m-1}} B_m^{(\Im,P_o-1)}(w_I) \\ & - \frac{P_o}{V_{P_o+m} - V_m} B_{m+1}^{(\Im,P_o-1)}(w_I). \end{aligned} \tag{1.11}$$

Using the tensor product between the two sets of univariate B-spline basis functions [30], $B_l^{(\Re,P_o)}(w_R)$ for $1 \le l \le N_R$ and $B_m^{(\Im,P_o)}(w_I)$ for $1 \le m \le N_I$, a set of new B-spline basis functions $B_{l,m}^{(P_o)}(w)$ can be formed and used in the CV B-spline neural network, giving rise to

$$\widehat{y} = \widehat{\Psi}(w) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_{l,m}^{(P_o)}(w)\omega_{l,m} = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(w_R) B_m^{(\Im,P_o)}(w_I)\omega_{l,m}, \tag{1.12}$$

where $\omega_{l,m} = \omega_{R_{l,m}} + \mathrm{j}\omega_{I_{l,m}} \in \mathbb{C}$, $1 \le l \le N_R$ and $1 \le m \le N_I$, are the CV weights. The CV B-spline neural network (1.12) can obviously be decomposed as the following two RV B-spline neural networks

$$\widehat{y}_R = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(w_R) B_m^{(\Im,P_o)}(w_I)\omega_{R_{l,m}}, \tag{1.13}$$

$$\widehat{y}_I = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(w_R) B_m^{(\Im,P_o)}(w_I)\omega_{I_{l,m}}. \tag{1.14}$$

Because of the piecewise nature of B-spline functions, for any point evaluation, there are only P_o basis functions with nonzero values for each of the real and imaginary parts, leading to P_o^2 nonzero terms in both (1.13) and (1.14). This is advantageous as P_o can be set to a quite low value. The complexity of the De Boor recursion is in the order of P_o^2, $\mathcal{O}(P_o^2)$. Thus the computational cost of calculating both (1.13) and (1.14) scales up to about three times of the De Boor recursion, including evaluation of both real and imaginary parts as well as the tensor product calculation. Notably, additional cost for derivative evaluation is minimal, as (1.7) and (1.11) are a byproduct of the De Boor recursion. Also there are only P_o nonzero first order derivative terms in each of (1.7) and (1.11). Compared with other CV neural networks based on different spline functions [3, 41, 42], our approach is clearly different in terms of model representation and identification algorithm. The advantages of our CV B-spline neural network are discussed in [40].

1.2.3 Wiener system identification

The Schematic of CV Wiener system identification is depicted in Fig. 1.1. For the chosen two sets of knots, (1.4) and (1.8), and the polynomial degree P_o, denote the weight vector of the CV B-spline neural network (1.12) as $\boldsymbol{\omega} = \big[\omega_{1,1}\ \omega_{1,2} \cdots \omega_{l,m} \cdots \omega_{N_R,N_I}\big]^{\mathrm{T}} \in \mathbb{C}^N$, where $N = N_R N_I$. Given a set of training input-output data $\{\mathbf{x}(k), y(k)\}_{k=1}^{K}$, where $\mathbf{x}(k) = [x(k)\ x(k-1) \cdots x(k-L)]^{\mathrm{T}}$, the task is to estimate the parameter vector $\boldsymbol{\theta} = \big[\theta_1\ \theta_2 \cdots \theta_{2(N+L)}\big]^{\mathrm{T}}$ of the Wiener model, defined as

$$\boldsymbol{\theta} = \big[\boldsymbol{\omega}_R^{\mathrm{T}}\ \boldsymbol{\omega}_I^{\mathrm{T}}\ \widehat{\mathbf{h}}_R^{\mathrm{T}}\ \widehat{\mathbf{h}}_I^{\mathrm{T}}\big]^{\mathrm{T}} \in \mathbb{R}^{2(N+L)}, \tag{1.15}$$

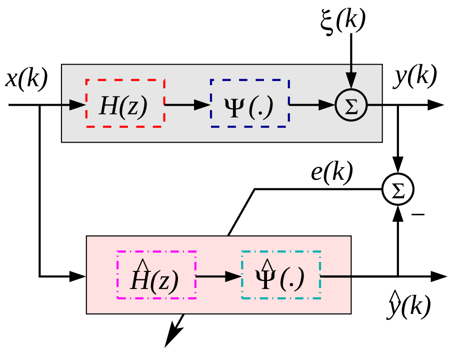

Figure 1.1 Schematic of Wiener system identification.

where $\widehat{\mathbf{h}} = \widehat{\mathbf{h}}_R + \mathrm{j}\widehat{\mathbf{h}}_I$ denotes the estimate of $\mathbf{h} = \mathbf{h}_R + \mathrm{j}\mathbf{h}_I$ and $\boldsymbol{\omega} = \boldsymbol{\omega}_R + \mathrm{j}\boldsymbol{\omega}_I$.

The CV B-spline neural network used in representing $\Psi(\bullet)$ is given by

$$\widehat{y}(k) = \widehat{\Psi}\left(\widehat{w}(k)\right) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k))\omega_{l,m}, \tag{1.16}$$

which is equivalent to the two RV B-spline neural networks

$$\widehat{y}_R(k) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k))\omega_{R_{l,m}}, \tag{1.17}$$

$$\widehat{y}_I(t) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k))\omega_{I_{l,m}}, \tag{1.18}$$

where

$$\begin{aligned}\widehat{w}(k) = \left[1\ \widehat{\mathbf{h}}^{\mathrm{T}}\right]\mathbf{x}(k) =& \Big(x_R(k) + \sum_{i=1}^{L}\left(\widehat{h}_{R_i}x_R(k-i) - \widehat{h}_{I_i}x_I(k-i)\right)\Big)\\ &+\mathrm{j}\Big(x_I(k) + \sum_{i=1}^{L}\left(\widehat{h}_{R_i}x_I(k-i) + \widehat{h}_{I_i}x_R(k-i)\right)\Big).\end{aligned} \tag{1.19}$$

Define the error between the desired output $y(k)$ and the Wiener model output $\widehat{y}(k)$ as $e(k) = y(k) - \widehat{y}(k)$, yielding the sum of squared errors (SSE) cost function

$$J_{\mathrm{SSE}}(\boldsymbol{\theta}) = \sum_{k=1}^{K}|e(k)|^2 = \sum_{k=1}^{K}\left(e_R^2(k) + e_I^2(k)\right). \tag{1.20}$$

We apply the Gauss-Newton algorithm to minimise the cost function (1.20).

The Gauss-Newton algorithm First denote $\boldsymbol{\varepsilon} = [\varepsilon_1\ \varepsilon_2 \cdots \varepsilon_{2K}]^{\mathrm{T}} \in \mathbb{R}^{2K}$ as

$$\boldsymbol{\varepsilon} = [e_R(1)\ e_R(2)\cdots e_R(K)\ e_I(1)\ e_I(2)\cdots e_I(K)]^{\mathrm{T}}. \tag{1.21}$$

By denoting the iteration step with the superscript $^{(\tau)}$ and with an initial value $\boldsymbol{\theta}^{(0)}$, the iteration formula is given by

$$\boldsymbol{\theta}^{(\tau)} = \boldsymbol{\theta}^{(\tau-1)} - \mu\Big(\big(\mathbf{J}^{(\tau)}\big)^{\mathrm{T}}\mathbf{J}^{(\tau)}\Big)^{-1}\big(\mathbf{J}^{(\tau)}\big)^{\mathrm{T}}\boldsymbol{\varepsilon}\big(\boldsymbol{\theta}^{(\tau-1)}\big), \tag{1.22}$$

where $\mu > 0$ is the step size, and $\mathbf{J}^{(\tau)}$ denotes the Jacobian of $\boldsymbol{\varepsilon}\big(\boldsymbol{\theta}^{(\tau-1)}\big)$, which is given by

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \varepsilon_1}{\partial \theta_1} & \frac{\partial \varepsilon_1}{\partial \theta_2} & \cdots & \frac{\partial \varepsilon_1}{\partial \theta_{2(N+L)}} \\ \frac{\partial \varepsilon_2}{\partial \theta_1} & \frac{\partial \varepsilon_2}{\partial \theta_2} & \cdots & \frac{\partial \varepsilon_2}{\partial \theta_{2(N+L)}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \varepsilon_{2K}}{\partial \theta_1} & \frac{\partial \varepsilon_{2K}}{\partial \theta_2} & \cdots & \frac{\partial \varepsilon_{2K}}{\partial \theta_{2(N+L)}} \end{bmatrix}. \tag{1.23}$$

The partial derivatives in the Jacobian (1.23) can be calculated as follows. For $1 \le k \le K$,

$$\frac{\partial \varepsilon_k}{\partial \theta_q} = \begin{cases} \frac{\partial e_R(k)}{\partial \omega_{R_{l,m}}} = -B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k)), \\ \qquad q = l\cdot m, 1 \le l \le N_R, 1 \le m \le N_I, \\ \frac{\partial e_R(k)}{\partial \omega_{I_{l,m}}} = 0, q = N + l\cdot m, 1 \le l \le N_R, 1 \le m \le N_I, \\ \frac{\partial e_R(k)}{\partial \widehat{h}_{R_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(k))}{d\widehat{w}_R(k)}B_m^{(\Im,P_o)}(\widehat{w}_I(k))x_R(k-i) \\ \qquad +B_l^{(\Re,P_o)}(\widehat{w}_R(k))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(k))}{d\widehat{w}_I(k)}x_I(k-i)\Big)\omega_{R_{l,m}}, \\ \qquad q = 2N + i, 1 \le i \le L, \\ \frac{\partial e_R(k)}{\partial \widehat{h}_{I_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(-\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(k))}{d\widehat{w}_R(k)}B_m^{(\Im,P_o)}(\widehat{w}_I(k))x_I(k-i) \\ \qquad +B_l^{(\Re,P_o)}(\widehat{w}_R(k))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(k))}{d\widehat{w}_I(k)}x_R(k-i)\Big)\omega_{R_{l,m}}, \\ \qquad q = 2N + L + i, 1 \le i \le L, \end{cases} \tag{1.24}$$

but for $K+1 \le k \le 2K$ and $t = k - K$,

$$\frac{\partial \varepsilon_k}{\partial \theta_q} = \begin{cases} \frac{\partial e_I(t)}{\partial \omega_{R_{l,m}}} = 0, q = l\cdot m, 1 \le l \le N_R, 1 \le m \le N_I, \\ \frac{\partial e_I(t)}{\partial \omega_{I_{l,m}}} = -B_l^{(\Re,P_o)}(\widehat{w}_R(t))B_m^{(\Im,P_o)}(\widehat{w}_I(t)), \\ \qquad q = N + l\cdot m, 1 \le l \le N_R, 1 \le m \le N_I, \\ \frac{\partial e_I(t)}{\partial \widehat{h}_{R_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(t))}{d\widehat{w}_R(t)}B_m^{(\Im,P_o)}(\widehat{w}_I(t))x_R(t-i) \\ \qquad +B_l^{(\Re,P_o)}(\widehat{w}_R(t))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(t))}{d\widehat{w}_I(t)}x_I(t-i)\Big)\omega_{I_{l,m}}, \\ \qquad q = 2N + i, 1 \le i \le L, \\ \frac{\partial e_I(t)}{\partial \widehat{h}_{I_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(-\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(t))}{d\widehat{w}_R(t)}B_m^{(\Im,P_o)}(\widehat{w}_I(t))x_I(t-i) \\ \qquad +B_l^{(\Re,P_o)}(\widehat{w}_R(t))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(t))}{d\widehat{w}_I(t)}x_R(t-i)\Big)\omega_{I_{l,m}}, \\ \qquad q = 2N + L + i, 1 \le i \le L. \end{cases} \tag{1.25}$$

It is seen that the De Boor algorithm, (1.5)–(1.7) and (1.9)–(1.11), is applied in evaluating all entries in the Jacobian. Effectively, this enables stable and efficient evaluations of B-spline functional and derivative values, which could be very difficult for many other nonlinear models, including some spline functions based nonlinear models. The iterative procedure (1.22) is terminated when $\boldsymbol{\theta}^{(\tau)}$ converges or when a predetermined sufficiently large number of iterations has been reached.

Parameter initialisation for the Gauss-Newton algorithm As the cost function (1.20) is highly nonlinear in the parameters, the solution of the Gauss-Newton algorithm depends on the initial condition. It is important to properly initialise $\boldsymbol{\theta}^{(0)}$ so that it is as close as possible to an optimal solution. A simple and effective LS parameter initialisation scheme was introduced in [40], which we adopt in this study.

Initialisation of the linear filter parameters. Denote an estimate of the linear filter parameter vector as $\widetilde{\mathbf{h}} = \left[\widetilde{h}_1\ \widetilde{h}_2 \cdots \widetilde{h}_L\right]^{\mathrm{T}}$ and the inverse function of $\Psi(\bullet)$ as $\varphi(\bullet) = \Psi^{-1}(\bullet) : \mathbb{C} \to \mathbb{C}$. Consider now using the proposed CV B-spline neural network to model $\varphi(\bullet)$. For notational simplicity, assume that the polynomial degree used is still denoted as $P_o - 1$ and the numbers of basis functions used in the modeling of the real and imaginary parts are still denoted as N_R and N_I, respectively. With the two knot vectors for the real and imaginary parts being set based on $y_R(k)$ and $y_I(k)$, respectively, we have an estimate of $\varphi(\bullet)$

$$\widetilde{\varphi}\left(y(k)\right) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_{l,m}^{(P_o)}\left(y(k)\right)\alpha_{l,m}, \tag{1.26}$$

where $\alpha_{l,m} \in \mathbb{C}$, $1 \le l \le N_R$ and $1 \le m \le N_I$, are CV weights. Let the error between $\widetilde{w}(k)$ and $\widetilde{\varphi}(y(k))$ be defined as $\epsilon(k) = \widetilde{w}(k) - \widetilde{\varphi}(y(k))$, where

$$\widetilde{w}(k) = x(k) + \sum_{i=1}^{L}\widetilde{h}_i x(k-i) \tag{1.27}$$

is used as the target for $\widetilde{\varphi}(y(k))$. Thus,

$$\begin{aligned} x(k) &= -\sum_{i=1}^{L}\widetilde{h}_i x(k-i) + \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_{l,m}^{(P_o)}(y(k))\alpha_{l,m} + \epsilon(k) \\ &= \left(\mathbf{p}(\underline{\mathbf{x}}(k))\right)^{\mathrm{T}}\boldsymbol{\vartheta} + \epsilon(k), \end{aligned} \tag{1.28}$$

where

$$\begin{aligned} \underline{\mathbf{x}}(k) =& \left[x(k-1)\ x(k-2) \cdots\ x(k-L)\ y(k)\right]^{\mathrm{T}}, \\ \mathbf{p}\left(\underline{\mathbf{x}}(k)\right) =& \big[-x(k-1)\ -x(k-2) \cdots -x(k-L) \\ & B_{1,1}^{(P_o)}(y(k))\ B_{1,2}^{(P_o)}(y(k)) \cdots B_{N_R,N_I}^{(P_o)}(y(k))\big]^{\mathrm{T}} \\ =& \left[p_1\left(\underline{\mathbf{x}}(k)\right)\ p_2\left(\underline{\mathbf{x}}(k)\right) \cdots p_{N+L}\left(\underline{\mathbf{x}}(k)\right)\right]^{\mathrm{T}} \in \mathbb{C}^{N+L}, \end{aligned}$$

$$\begin{aligned}\boldsymbol{\vartheta} &= \left[\widetilde{h}_1\ \widetilde{h}_2 \cdots \widetilde{h}_L\ \alpha_{1,1}\ \ \alpha_{1,2} \cdots \alpha_{l,m} \cdots \alpha_{N_R,N_I}\right]^{\mathrm{T}} \\ &= \left[\vartheta_1\ \vartheta_2 \cdots \vartheta_{N+L}\right]^{\mathrm{T}} \in \mathbb{C}^{N+L}.\end{aligned}$$

Over the training data set, (1.28) can be written in the matrix form as

$$\overline{\mathbf{x}} = \mathbf{P}\boldsymbol{\vartheta} + \boldsymbol{\epsilon}, \tag{1.29}$$

where $\overline{\mathbf{x}} = [x(1)\ x(2) \cdots x(K)]^{\mathrm{T}}$, $\boldsymbol{\epsilon} = [\epsilon(1)\ \epsilon(2) \cdots \epsilon(K)]^{\mathrm{T}}$, and $\mathbf{P}$ is the regression matrix defined as $\mathbf{P} = [\mathbf{p}(\underline{\mathbf{x}}(1))\ \mathbf{p}(\underline{\mathbf{x}}(2)) \cdots \mathbf{p}(\underline{\mathbf{x}}(K))]^{\mathrm{T}}$. The LS solution for the parameter vector $\boldsymbol{\vartheta}$ is readily given as

$$\boldsymbol{\vartheta}_{\mathrm{LS}} = \left(\mathbf{P}^{\mathrm{H}}\mathbf{P}\right)^{-1}\mathbf{P}^{\mathbf{H}}\overline{\mathbf{x}}. \tag{1.30}$$

The sub-vector of the resulting $\boldsymbol{\vartheta}_{\mathrm{LS}}$, consisting of its first L CV elements, forms our initial estimate $\widehat{\mathbf{h}}^{(0)} = \widehat{\mathbf{h}}_R^{(0)} + \mathrm{j}\widehat{\mathbf{h}}_I^{(0)}$, which are used as the last $2L$ RV elements of $\boldsymbol{\theta}^{(0)}$ in the parameter initialisation for the Gauss-Newton iteration procedure.

Initialisation of the B-spline neural network weights. Given the estimate $\widehat{\mathbf{h}}^{(0)}$, generate the auxiliary signal

$$\widehat{w}(k) = x(k) + \sum_{i=1}^{L} \widehat{h}_i^{(0)} x(k-i). \tag{1.31}$$

Using the CV B-spline neural network (1.12) to model the nonlinear static function $\Psi(\bullet)$ based on the training data set $\{\widehat{w}(k), y(k)\}_{k=1}^{K}$ yields

$$y(k) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_{l,m}^{(P_o)}\big(\widehat{w}(k)\big)\omega_{l,m} + \widehat{e}(k) = \Big(\mathbf{q}\big(\widehat{w}(k)\big)\Big)^{\mathbf{T}}\boldsymbol{\omega} + \widehat{e}(k), \tag{1.32}$$

where

$$\begin{aligned}\mathbf{q}\big(\widehat{w}(k)\big) &= \Big[B_{1,1}^{(P_o)}\big(\widehat{w}(k)\big)\ B_{1,2}^{(P_o)}\big(\widehat{w}(k)\big) \cdots B_{l,m}^{(P_o)}\big(\widehat{w}(k)\big) \cdots B_{N_R,N_I}^{(P_o)}\big(\widehat{w}(k)\big)\Big]^{\mathrm{T}} \\ &= \big[q_1\big(\widehat{w}(k)\big)\ q_2\big(\widehat{w}(k)\big) \cdots, q_N\big(\widehat{w}(k)\big)\big]^{\mathrm{T}} \in \mathbb{R}^N.\end{aligned}$$

Over the training data set, (1.32) can be written in the matrix form

$$\mathbf{y} = \mathbf{Q}\boldsymbol{\omega} + \widehat{\mathbf{e}}, \tag{1.33}$$

with $\mathbf{y} = [y(1) \cdots y(K)]^{\mathrm{T}}$, $\widehat{\mathbf{e}} = [\widehat{e}(1) \cdots \widehat{e}(K)]^{\mathrm{T}}$ and $\mathbf{Q} = \big[\mathbf{q}\big(\widehat{w}(1)\big) \cdots \mathbf{q}\big(\widehat{w}(K)\big)\big]^{\mathrm{T}}$. The LS solution for $\boldsymbol{\omega} \in \mathbb{C}^N$

$$\boldsymbol{\omega}_{\mathrm{LS}} = \Big(\mathbf{Q}^{\mathrm{T}}\mathbf{Q}\Big)^{-1}\mathbf{Q}^{\mathrm{T}}\mathbf{y} \tag{1.34}$$

is used as the initial estimate of $\boldsymbol{\omega}^{(0)} = \boldsymbol{\omega}_R^{(0)} + \mathrm{j}\boldsymbol{\omega}_I^{(0)}$ that forms the first $2N$ RV elements of $\boldsymbol{\theta}^{(0)}$ for the parameter initialisation of the Gauss-Newton algorithm.

The LS estimates $\widehat{\mathbf{h}}^{(0)}$ and $\boldsymbol{\omega}^{(0)}$ are generally not consistent. This is because the B-spline regressors in (1.26) and (1.32) are subject to the output noise which will in general propagate to the parameter estimates, yielding a bias. However, this estimate represents an excellent initialisation for the Gauss-Newton algorithm. The final parameter estimate via minimising (1.20) is optimal in the sense that it is the maximum likelihood estimate in the case that $\xi(t)$ is Gaussian.

1.2.4 Wiener system inverse

For the CV Wiener system (1.1) and (1.2), there are two types of inverse as depicted in Fig. 1.2. The "pre-inverse" can be found for example in the digital predistorter design for compensating the Wiener HPA [34–39], while the "post-inverse" is typically found in the deconvolution or equalisation applications [2, 3]. Note that in either case, the exact inverse of the Wiener system is a Hammerstein system consisting of a nonlinear static function followed by a linear filter. The difference between these two cases is that in the pre-inverse case, the input to the Hammerstein model is a clean, i.e. noise-free, signal, while in the post-inverse case, the input signal to the Hammerstein model is corrupted by the noise. Without significant loss of generality, we consider the pre-inverse case in this study.

Inverse of Wiener system's static nonlinear function Given the CV Wiener system's static nonlinearity $\Psi(\bullet)$, we wish to compute its inverse defined by $v(k) = \Psi^{-1}(x(k))$. This task is identical to find the CV root of $x(k) = \Psi(v(k))$, given $x(k)$. In Subsection 1.2.3, the estimate $\widehat{\Psi}(\bullet)$ for $\Psi(\bullet)$ has been obtained based on the CV B-spline neural network with the aid of the De Boor algorithm. We now show that $\widehat{\Psi}^{-1}(\bullet)$ can be effectively obtained with the aid of the inverse of De Boor algorithm. Given $\widehat{\Psi}(\bullet)$ of (1.17) and (1.18),

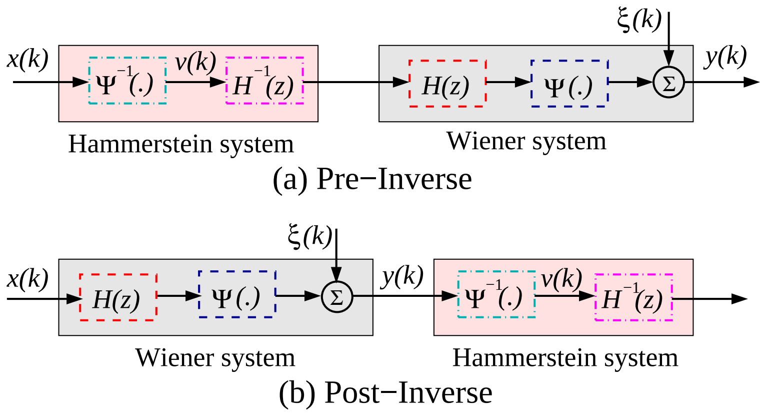

Figure 1.2 Schematic of inverse for Wiener system.

we have

$$\widehat{x}_R(k) = \sum_{l=1}^{N_R} \sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(v_R(k)) B_m^{(\Im,P_o)}(v_I(k)) \omega_{R_{l,m}}, \tag{1.35}$$

$$\widehat{x}_I(t) = \sum_{l=1}^{N_R} \sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(v_R(k)) B_m^{(\Im,P_o)}(v_I(k)) \omega_{I_{l,m}}. \tag{1.36}$$

Define $\zeta(k) = x(k) - \widehat{x}(k)$ and the squared error (SE)

$$S(k) = \zeta_R^2(k) + \zeta_I^2(k). \tag{1.37}$$

If $S(k) = 0$, then $v(k)$ is the CV root of $x(k) = \widehat{\Psi}(v(k))$. Thus, the task is equivalent to the one that minimises the SE (1.37). We propose to use the following Gauss-Newton algorithm to solve this optimisation problem with the aid of the inverse of De Boor algorithm.

By denoting again the iteration step with the superscript $^{(\tau)}$ and giving a random initialisation of $v^{(0)}(k)$ that satisfies $U_{\min} < v_R^{(0)}(k) < U_{\max}$ and $V_{\min} < v_I^{(0)}(k) < V_{\max}$, the iterative procedure is given by

$$\left[\begin{array}{c} v_R^{(\tau)}(k) \\ v_I^{(\tau)}(k) \end{array} \right] = \left[\begin{array}{c} v_R^{(\tau-1)}(k) \\ v_I^{(\tau-1)}(k) \end{array} \right] - \eta \Big(\big(\mathbf{J}_v^{(\tau)}\big)^{\mathrm{T}} \mathbf{J}_v^{(\tau)} \Big)^{-1} \big(\mathbf{J}_v^{(\tau)}\big)^{\mathrm{T}} \left[\begin{array}{c} \zeta_R^{(\tau-1)}(k) \\ \zeta_I^{(\tau-1)}(k) \end{array} \right], \tag{1.38}$$

where $\eta > 0$ is the step size, $\zeta^{(\tau)}(k) = x(k) - \widehat{x}^{(\tau)}(k)$ with $\widehat{x}^{(\tau)}(k) = \widehat{\Psi}\big(v^{(\tau)}(k)\big)$, and $\mathbf{J}_v^{(\tau)}$ is the 2×2 Jacobian matrix given by

$$\mathbf{J}_v^{(\tau)} = \left[\begin{array}{cc} \frac{\partial \zeta_R(k)}{\partial v_R(k)} & \frac{\partial \zeta_R(k)}{\partial v_I(k)} \\ \frac{\partial \zeta_I(k)}{\partial v_R(k)} & \frac{\partial \zeta_I(k)}{\partial v_I(k)} \end{array} \right]_{|v(k)=v^{(\tau)}(k)}. \tag{1.39}$$

The entries in (1.39) are given by

$$\left\{ \begin{array}{l} \frac{\partial \zeta_R(k)}{\partial v_R(k)} = -\sum\limits_{l=1}^{N_R} \sum\limits_{m=1}^{N_I} \frac{dB_l^{(\Re,P_o)}(v_R(k))}{dv_R(k)} B_m^{(\Im,P_o)}(v_I(k)) \omega_{R_{l,m}}, \\ \frac{\partial \zeta_R(k)}{\partial v_I(k)} = -\sum\limits_{l=1}^{M_R} \sum\limits_{m=1}^{N_I} B_l^{(\Re,P_o)}(v_R(k)) \frac{dB_m^{(\Im,P_o)}(v_I(k))}{dv_I(k)} \omega_{R_{l,m}}, \\ \frac{\partial \zeta_I(k)}{\partial v_R(k)} = -\sum\limits_{l=1}^{N_R} \sum\limits_{m=1}^{N_I} \frac{dB_l^{(\Re,P_o)}(v_R(k))}{dv_R(k)} B_m^{(\Im,P_o)}(v_I(k)) \omega_{I_{l,m}}, \\ \frac{\partial \zeta_I(k)}{\partial v_I(k)} = -\sum\limits_{l=1}^{N_R} \sum\limits_{m=1}^{N_I} B_l^{(\Re,P_o)}(v_R(k)) \frac{dB_m^{(\Im,P_o)}(v_I(t))}{dv_I(k)} \omega_{I_{l,m}}, \end{array} \right. \tag{1.40}$$

for which the De Boor algorithm, (1.5)–(1.7) and (1.9)–(1.11), can be used for their calculation efficiently. The algorithm is terminated when $S(k) < \rho$, where ρ is a preset required precision, e.g. $\rho = 10^{-8}$, or when τ reaches a predetermined maximum value.

Inverse of Wiener system's linear filter The identification algorithm presented in Subsection 1.2.3 also provides the estimate of the Wiener system's linear filter $\widehat{H}(z) = 1 + \sum_{i=1}^{L} \widehat{h}_i z^{-i}$. Let the transfer function of the Hammerstein model's linear filter be

$$G(z) = z^{-\iota} \cdot \sum_{i=0}^{L_g} g_i z^{-i}, \tag{1.41}$$

where the delay $\iota = 0$ if $H(z)$ is minimum phase. The solution of the Hammerstein model's linear filter $\mathbf{g} = [g_0 \; g_1 \cdots g_{L_g}]^{\mathrm{T}}$ can readily be obtained by solving the set of linear equations specified by

$$G(z) \cdot \hat{H}(z) = z^{-\iota}. \tag{1.42}$$

To guarantee an accurate inverse, the length of $\mathbf{g}$ should be chosen to be three to four times of the length of $\mathbf{h}$. Note that $g_0 = 1$ as $h_0 = 1$.

1.3 APPLICATION TO DIGITAL PREDISTORTER DESIGN

HPA is an indispensable component in any wireless communication system. The operation of HPAs in modern wireless systems may introduce serious memory effects and nonlinear distortions [32, 33, 43, 44], causing intersymbol interference and adjacent channel interference that degrade the system's achievable bit error rate (BER) performance. The problem becomes particularly acute, as the recent green-radio initiative [45] places the emphasis on the energy-efficiency aspect of communication. To achieve high energy efficiency, HPAs should operate at their output saturation regions but this operational mode could not accommodate high bandwidth-efficiency single-carrier high-order quadrature amplitude modulation (QAM) signals [46] as well as multi-carrier orthogonal frequency division multiplexing (OFDM) signals [47], which are essential modern transmission technologies. It is therefore critical to compensate the distortions caused by the HPA with a digital predistorter in the design of a wireless system [34–39].

1.3.1 High power amplifier model

A widely used model for HPAs is the Wiener model [32]. Without loss of generality, we consider single-carrier QAM systems [46], but our approach is equally applicable to multi-carrier OFDM systems [47]. The CV input signal to the HPA, $x(k)$, where k denotes the discrete time or symbol index, takes the values from the CV M-QAM symbol set

$$\mathbb{S} = \{d(2l - \sqrt{M} - 1) + \mathrm{j}d(2q - \sqrt{M} - 1), 1 \le l, q \le \sqrt{M}\}, \tag{1.43}$$

where $2d$ is the minimum distance between symbol points. The 16-QAM symbol constellation is illustrated in Fig. 1.3. The memory effect of the Wiener

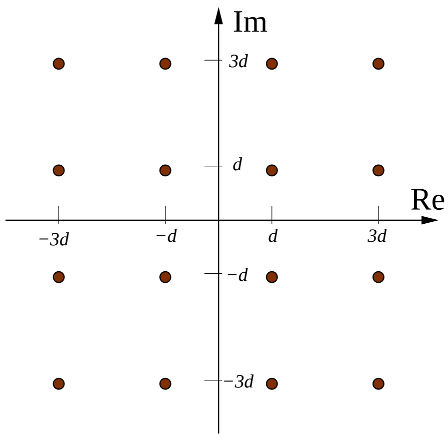

Figure 1.3 16-QAM symbol constellation.

HPA can be modelled by the FIR filter (1.1), while the nonlinear saturating distortion of the Wiener HPA can be represented by the static nonlinearity (1.2). Note that in practical HPAs, the noise $\xi(k)$ is often negligible, that is, σ_ξ^2 is zero or extremely small. Two typical CV nonlinearities $\Psi(\bullet)$ of HPAs are the travelling-wave tube (TWT) nonlinearity [43] and the nonlinearity of solid state power amplifiers [44]. Nonlinear characteristics of these two types of HPAs are similar. The static nonlinearity of the HPA considered in this study is the TWT nonlinearity, but the approach is equally applicable to the other type of nonlinearity.

Express the (unavailable) input signal $w(k)$ to the static nonlinearity part $\Psi(\bullet)$ of the HPA by

$$w(k) = r(k) \cdot \exp(\mathrm{j}\psi(k)), \tag{1.44}$$

with the amplitude $r(k) = |w(k)|$ and phase $\psi(k) = \angle^{w(k)}$. The input signal $w(k)$ is affected by the nonlinear amplitude and phase functions of the HPA, and the output signal $y(k)$ is distorted mainly depending on the input signal amplitude $r(k)$, yielding

$$y(k) = |y(k)| \cdot \exp(\mathrm{j}\angle^{y(k)}) = A(r(k)) \cdot \exp(\mathrm{j}(\psi(k) + \Phi(r(k)))). \tag{1.45}$$

The output amplitude $A(r(k))$ and the phase $\Phi(r(k)) = \angle^{y(k)} - \psi(k)$ of the HPA are specified respectively by [32, 39, 43]

$$A(r) = \begin{cases} \frac{\alpha_a r}{1+\beta_a r^2}, & 0 \le r \le r_{\mathrm{sat}}, \\ A_{\max}, & r > r_{\mathrm{sat}}, \end{cases} \tag{1.46}$$

$$\Phi(r) = \frac{\alpha_\phi r^2}{1 + \beta_\phi r^2}, \tag{1.47}$$

where the saturating input amplitude is defined as

$$r_{\mathrm{sat}} = \frac{1}{\sqrt{\beta_a}}, \tag{1.48}$$

while the saturation output amplitude is given by

$$A_{\max} = \frac{\alpha_a}{2\sqrt{\beta_a}}. \tag{1.49}$$

The underlying physics require that $A_{\max} > r_{\text{sat}}$ and the input amplitude r meets the condition $r < R_{\max}$, where $R_{\max}$ is some large positive number. The TWT nonlinearity is specified by the positive RV parameter vector $\mathbf{t} = [\alpha_a \ \beta_a \ \alpha_\phi \ \beta_\phi]^{\mathrm{T}}$. The operating status of the HPA is specified by the input back-off (IBO), which is defined as

$$\text{IBO} = 10 \cdot \log_{10} \frac{P_{\text{sat}}}{P_{\text{avg}}}, \tag{1.50}$$

where $P_{\text{sat}} = r_{\text{sat}}^2$ is the saturation input power and P_{avg} is the average power of the signal at the input of the TWT nonlinearity. Note that here P_{avg} is defined as the average power of $w(k)$, which is equal to the average power of $x(k)$ scaled by the linear filter power gain $1 + \|\mathbf{h}\|^2$. A small IBO value indicates that the HPA operates in the highly nonlinear saturation region.

1.3.2 A novel digital predistorter design

Based on the technique developed in Section 1.2 for identification and inverse of the CV Wiener system, a novel digital predistorter can readily be designed to compensate the distortions caused by the HPA. Because both the predistorter and the HPA are operating at the transmitter, the input M-QAM signal $x(k)$ to the HPA and the HPA's output signal $y(k)$ are readily available to identify the Wiener HPA model $\widehat{H}(z)$ and $\widehat{\Psi}(\bullet)$ using the Gauss-Newton method based on the De Door algorithm of Subsection 1.2.3. Since the distributions of $x_R(k)$ and $x_I(k)$ are symmetric, the distributions of $w_R(k)$ and $w_I(k)$ are also symmetric. Furthermore, from the underlying physics of the HPA, $R_{\max}$ is known or can easily be found. Therefore, the two knot sequences (1.4) and (1.8) can be chosen to be identical with $U_{\max} = V_{\max} = R_{\max}$, $U_{\min} = V_{\min} = -R_{\max}$ and $N_R = N_I = \sqrt{N}$. In practice, $P_o = 4$ is sufficient, and an appropriate value of $\sqrt{N}$ can be chosen by trail and error. Specifically, the number of internal knots should be sufficient to provide good modelling capability but should not be too large in order to avoid overfitting.

Based on the estimated $\widehat{\Psi}(\bullet) = \widehat{\Psi}_R(\bullet) + \mathrm{j}\widehat{\Psi}_I(\bullet)$, an accurate inverse to $\Psi(\bullet) = \Psi_R(\bullet) + \mathrm{j}\Psi_I(\bullet)$ can readily be obtained. Note that over the input range, $\Psi_R(\bullet)$ and $\Psi_I(\bullet)$ are monotonic. Since $\widehat{\Psi}(\bullet)$ is an accurate estimate of $\Psi(\bullet)$, $\widehat{\Psi}_R(\bullet)$ and $\widehat{\Psi}_I(\bullet)$ can also be assumed to be monotonic over the input range. Therefore, the Gauss-Newton method of Subsection 1.2.4 based on the inverse of De Door algorithm converges to the unique solution $\widehat{\Psi}^{-1}(\bullet)$. For the M-QAM signal (1.43), there are M different symbol points $x(k)$. Thus, $v(k) = \widehat{\Psi}^{-1}(x(k))$ has M distinct values, and these values can be precalculated off-line and stored for on-line transmission. Therefore, our proposed digital predistorter solution has extremely low on-line computational complexity, which is critically important for high-throughput wireless systems.

1.3.3 A simulation example

We considered the single-carrier 16-QAM system with the static nonlinearity of the Wiener HPA described by (1.46) and (1.47). The parameters of the Wiener HPA were given as

$$\begin{aligned} \mathbf{h}^{\mathrm{T}} &= [0.75+\mathrm{j}0.2\ 0.15+\mathrm{j}0.1\ 0.08+\mathrm{j}0.01], \\ \mathbf{t}^{\mathrm{T}} &= [2.1587\ 1.15\ 4.0\ 2.1]. \end{aligned} \tag{1.51}$$

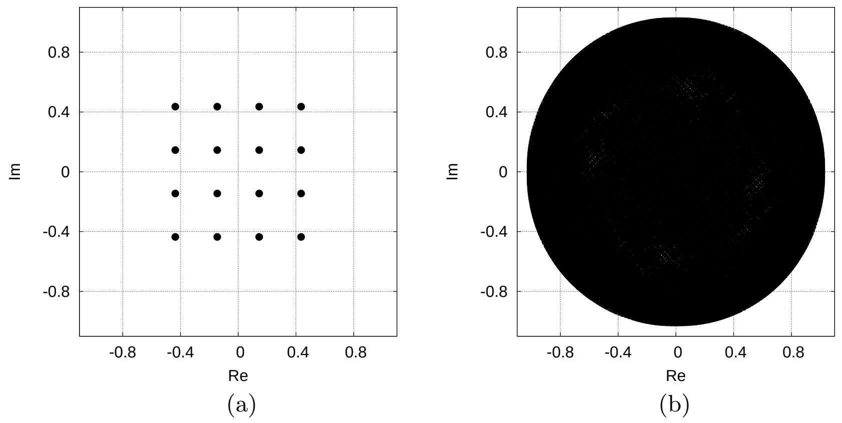

Figure 1.4 The case of IBO= 4 dB: (a) the HPA's input $x(k)$, marked by •, and (b) the HPA's output $y(k)$, marked by $\times$.

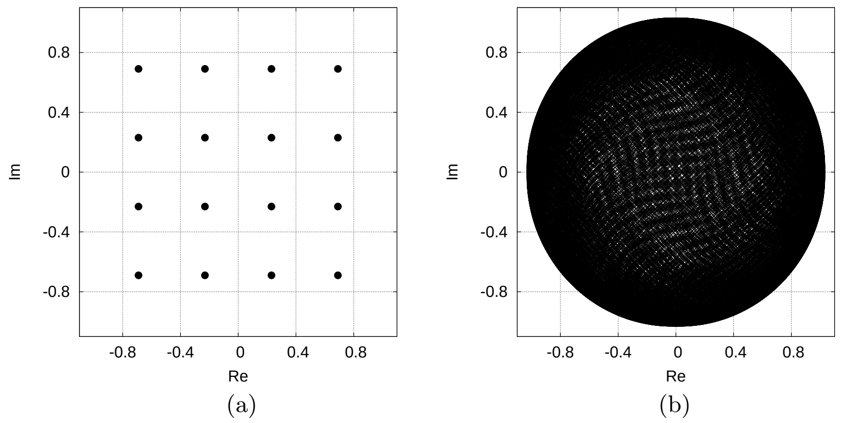

Figure 1.5 The case of IBO= 0 dB: (a) the HPA's input $x(k)$, marked by •, and (b) the HPA's output $y(k)$, marked by $\times$.

The serious nonlinear and memory distortions caused by this memory HPA are illustrated in Figs. 1.4 and 1.5. Note that, for IBO= 0 dB, the HPA is operating well into the saturation region.

Results of Wiener HPA identification The 16-QAM training sets each containing $K = 3000$ data samples were generated given the HPA's parameters (1.51) and with the HPA operating at the IBO values of 4 dB and 0 dB, respectively, where the power of the CV output measurement noise $\xi(k)$ was $2\sigma_{\xi}^2$. Note that since the identification is carried out at the transmitter, both the HPA's input $\mathbf{x}(k)$ and the corresponding HPA's output measurement $y(k)$ are available. Furthermore, the measurement $y(k)$ can usually be considered as noise free. However, to demonstrate the effectiveness of the proposed CV B-spline identification approach, we considered both the noise-free and noisy measurement cases with $2\sigma_{\xi}^2 = 0.0$ and $2\sigma_{\xi}^2 = 0.01$, respectively.

The piecewise cubic polynomial ($P_o = 4$) was chosen as the B-spline basis function, and the number of B-spline basis functions was set to $\sqrt{N} = 8$. For this HPA, we set $R_{\max} = 1.2$, and used the empirically determined knot sequence

$$\{-12.0, -6.0, -2.0, -\mathbf{1.2}, -0.6, -0.3, 0.0, 0.3, 0.6, \mathbf{1.2}, 2.0, 6.0, 12.0\}.$$

The Gauss-Newton identification algorithm with the LS parameter initialisation, as described in Subsection 1.2.3, was carried out. The results obtained are summarised in Table 1.1 and illustrated in Figs. 1.6 to 1.9, which confirm that an accurate CV B-spline neural network model can be obtained for the HPA even in the cases that the measurements $y(k)$ are corrupted by noise.

In order to achieve an accurate identification of a nonlinear system, the nonlinear system should be sufficiently excited over all the amplitudes concerned by the input signal, which is known as the "persistent excitation" condition. Note that, under the identification condition of IBO= 4 dB, there were relative few data points which yielded the signal amplitude $r(k) = |w(k)|$ with

Table 1.1 Identification results for the linear filter part, $\mathbf{h}$, of the HPA.

true parameter vector: $\mathbf{h}^{\mathrm{T}} = \left[0.7500 + \mathrm{j}0.2000\ 0.1500 + \mathrm{j}0.1000\ 0.0800 + \mathrm{j}0.0010\right]$
estimate under IBO= 0 dB and $2\sigma_{\xi}^2 = 0.0$: $\widehat{\mathbf{h}}^{\mathrm{T}} = \left[0.7502 + \mathrm{j}0.1996\ 0.1499 + \mathrm{j}0.0999\ 0.0800 + \mathrm{j}0.0008\right]$
estimate under IBO= 0 dB and $2\sigma_{\xi}^2 = 0.01$: $\widehat{\mathbf{h}}^{\mathrm{T}} = \left[0.7519 + \mathrm{j}0.1963\ 0.1510 + \mathrm{j}0.1000\ 0.0814 + \mathrm{j}0.0014\right]$
estimate under IBO= 4 dB and $2\sigma_{\xi}^2 = 0.0$: $\widehat{\mathbf{h}}^{\mathrm{T}} = \left[0.7502 + \mathrm{j}0.2001\ 0.1501 + \mathrm{j}0.1001\ 0.0800 + \mathrm{j}0.0011\right]$
estimate under IBO= 4 dB and $2\sigma_{\xi}^2 = 0.01$: $\widehat{\mathbf{h}}^{\mathrm{T}} = \left[0.7533 + \mathrm{j}0.1978\ 0.1518 + \mathrm{j}0.1002\ 0.0810 + \mathrm{j}0.0019\right]$

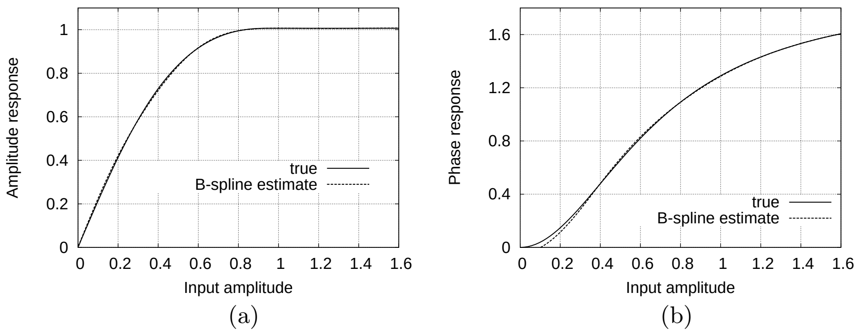

Figure 1.6 Comparison of the HPA's static nonlinearity $\Psi(\bullet)$ and the estimated static nonlinearity $\widehat{\Psi}(\bullet)$ under IBO= 0 dB and $2\sigma_\xi^2 = 0.0$: (a) the amplitude response, and (b) the phase response.

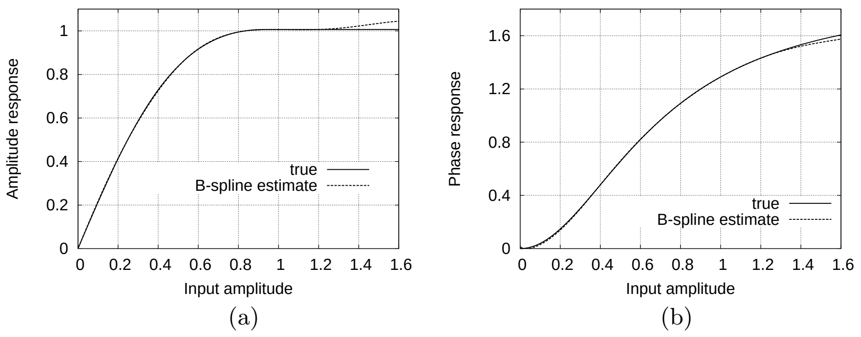

Figure 1.7 Comparison of the HPA's static nonlinearity $\Psi(\bullet)$ and the estimated static nonlinearity $\widehat{\Psi}(\bullet)$ under IBO= 4 dB and $2\sigma_\xi^2 = 0.0$: (a) the amplitude response, and (b) the phase response.

the values near or over the saturation value r_{sat}. Consequently, the amplitude response and phase response of the estimated B-spline neural network $\widehat{\Psi}(\bullet)$ exhibits noticeable deviation from the HPA's true amplitude response $A(r)$ and true phase response $\Phi(r)$ in the region $r > R_{\max}$, as can be seen from Figs. 1.7 and 1.9. This of course does not matter, as this region is well beyond the operating region of the HPA. Interestingly, under the operating condition of IBO= 0 dB, the deviation between the estimated response and the true response at the region of $r > R_{\max}$ is no longer noticeable, as can be noted from Figs. 1.6 and 1.8, because of the better excitation of the input signal. From Figs. 1.7 and 1.9, it can be seen that the noise $\xi(k)$ mainly affects the estimated phase response at the region of the signal amplitude $r(k)$ near zero. Note that this relatively poor accuracy of the estimated phase response under the noisy measurement condition at $r(k)$ near zero does not matter at all. This is because the estimated $\widehat{\Psi}(\bullet)$ is used to design $v(k) = \widehat{\Psi}^{-1}(x(k))$ for the 16-QAM signal $x(k)$, whose amplitude $|x(k)|$ is much larger and is well over this near zero region.

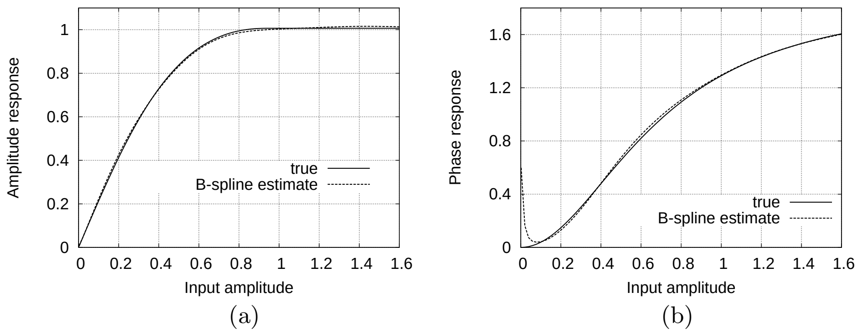

Figure 1.8 Comparison of the HPA's static nonlinearity $\Psi(\bullet)$ and the estimated static nonlinearity $\widehat{\Psi}(\bullet)$ under IBO= 0 dB and $2\sigma_\xi^2 = 0.01$: (a) the amplitude response, and (b) the phase response.

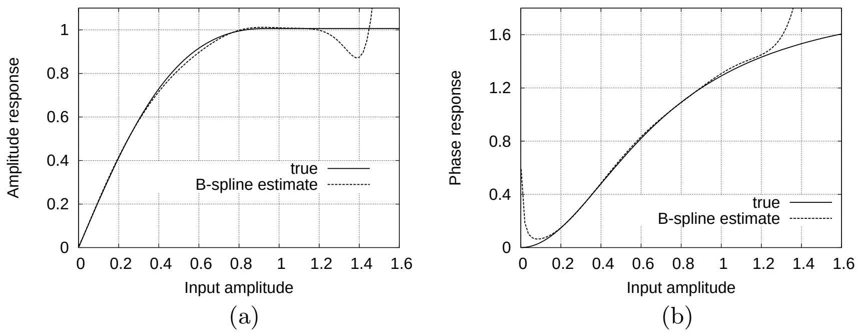

Figure 1.9 Comparison of the HPA's static nonlinearity $\Psi(\bullet)$ and the estimated static nonlinearity $\widehat{\Psi}(\bullet)$ under IBO= 4 dB and $2\sigma_\xi^2 = 0.01$: (a) the amplitude response, and (b) the phase response.

Results of digital predistorter solution We employed the estimated CV B-spline Wiener HPA model obtained under the condition of noise-free measurement ($2\sigma_\xi^2 = 0.0$) to design the predistorter. Note that we only needed to calculate the 16 points of $v(k) = \widehat{\Psi}^{-1}(x(k))$ for the 16-QAM constellation using the Gauss-Newton algorithm based on the De Door inverse, as described in Subsection 1.2.4. The length of the predistorter's inverse filter was set to $L_g = 12$. The outputs of the combined predistorter and Wiener HPA are depicted in Fig. 1.10 for the HPA's operating conditions of IBO= 4 dB and 0 dB, respectively. Compared with the outputs of the HPA as plotted in Fig. 1.4 (b) and Fig. 1.5 (b), it can be seen that the designed predistorter successfully removes the serious distortions caused by the HPA. The achievable performance of the designed predistorter was further assessed using the MSE metric defined by

$$\mathrm{MSE} = 10\log_{10}\Big(\frac{1}{K_{\mathrm{test}}}\sum_{k=1}^{K_{\mathrm{test}}} |x(k) - y(k)|^2\Big), \tag{1.52}$$

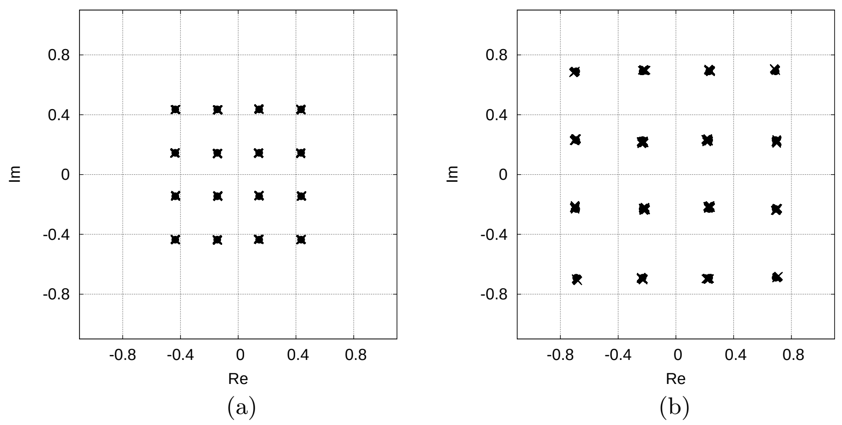

Figure 1.10 The output of the combined predistorter and HPA $y(k)$, marked by $\times$, for the 16-QAM input signal $x(k)$, marked by $\bullet$: (a) the IBO of 4 dB, and (b) the IBO of 0.0 dB.

and the system's BER, where K_{test} was the number of test data, $x(k)$ was the 16-QAM input and $y(k)$ was the output of the combined predistorter and HPA system. The channel signal to noise ratio (SNR) in the simulation was given by $\text{SNR} = 10 \log_{10} \left(\text{E}_\text{b}/\text{N}_\text{o}\right)$, where E_b was defined as the energy per bit and N_o the power of the channel's additive white Gaussian noise (AWGN).

With $K_{\text{test}} = 10^5$, 16-QAM data were passed through the combined predistorter and HPA system to compute the MSE (1.52), and the resulting MSE as the function of IBO is plotted in Fig. 1.11. The output signal after the HPA was then transmitted over the AWGN channel, and the BER was then

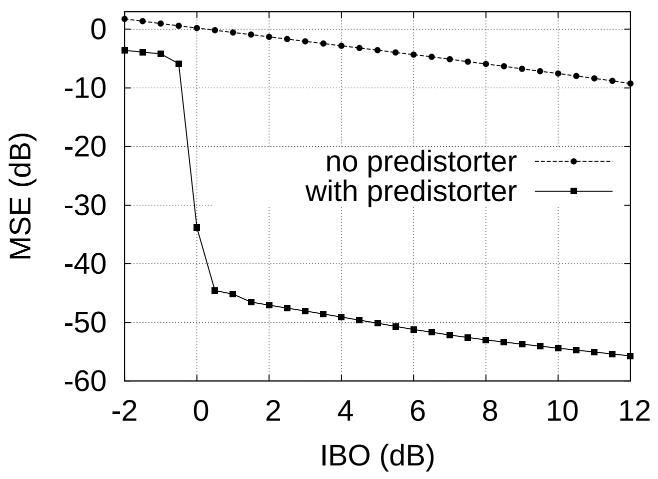

Figure 1.11 The mean square error versus IBO performance.

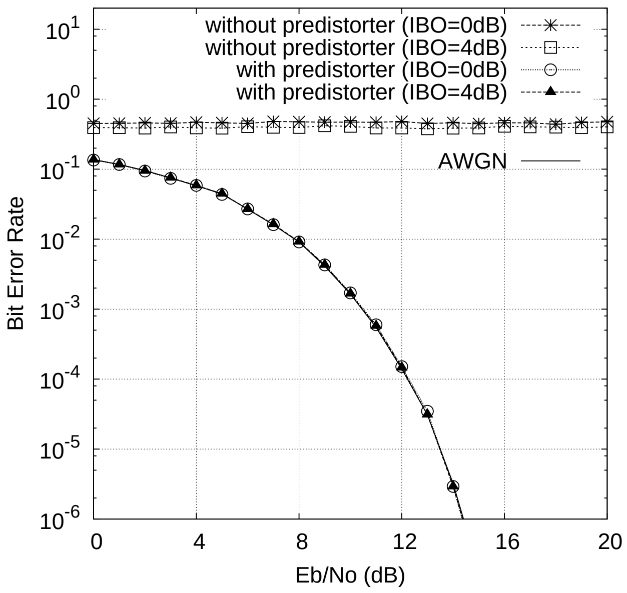

Figure 1.12 The bit error rate versus channel SNR performance.

determined at the receiver. The results so obtained are plotted in Fig. 1.12, in comparison with the benchmark BER curve of the ideal AWGN channel. It can be seen from Fig. 1.12 that the BER performance of the combined predistorter and HPA system is practically indistinguishable from those of the ideal AWGN channel even under the operating condition of IBO = 0 dB. The achievable BER performance of the combined predistorter and HPA system are further illustrated in Fig. 1.13 for the three values of the channel SNR.

1.4 CONCLUSIONS

Identification and inverse of complex-valued Wiener systems have been proposed based on the complex-valued B-spline neural network approach. Our contribution is twofold. Firstly, the complex-valued nonlinear static function in the Wiener system is modelled based on the tensor product from two univariate B-spline neural networks that are constructed using the real and imaginary parts of the system input. The Gauss-Newton algorithm, aided by an least squares parameter initialisation scheme, has been applied to estimate the model parameters that include the complex-valued linear dynamic model coefficients and B-spline neural network weights. The identification algorithm naturally incorporates the efficient De Boor algorithm with both the B-spline curve and first order derivative recursions. Secondly, an accurate inverse technique has been developed for the complex-valued Wiener model. In particular,

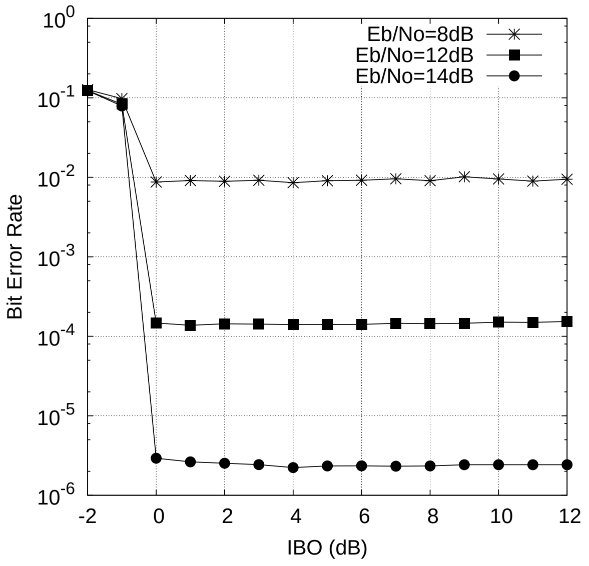

Figure 1.13 The bit error rate versus IBO performance of the combined predistorter and HPA for three values of the channel SNR.

the inverse of the complex-valued nonlinear static function in the Wiener system is calculated effectively using the Gaussian-Newton algorithm based on the estimated B-spline neural network model with the aid of the inverse of De Boor algorithm that again utilises naturally both the B-spline curve and first order derivative recursions. An application to digital predistorter design for high power amplifiers with memory has been used to demonstrate the effectiveness of our approach for modelling and inverse of complex-valued Wiener systems.

REFERENCES

1. S. Chen, S. McLaughlin, and B. Mulgrew, “Complex-valued radial basis function network, Part I: Network architecture and learning algorithms,” *Signal Processing*, vol. 35, no. 1, pp. 19–31, Jan. 1994.

2. S. Chen, S. McLaughlin, and B. Mulgrew, “Complex-valued radial basis function network, Part II: Application to digital communications channel equalisation,” *Signal Processing*, vol. 36, no. 2, pp. 175–188, March 1994.

3. A. Uncini, L. Vecci, P. Campolucci, and F. Piazza, “Complex valued neural networks with adaptive spline activation function for digital radio links nonlinear equalization,” *IEEE Trans. Signal Processing*, vol. 47, no. 2, pp. 505–514, Feb. 1999.

4. T. Kim and T. Adali, "Approximation by fully complex multilayer perceptrons," *Neural Computation*, vol. 15, no. 7, pp. 1641–1666, July 2003.

5. C.-C. Yang and N. K. Bose, "Landmine detection and classification with complex-valued hybrid neural network using scattering parameters dataset," *IEEE Trans. Neural Networks*, vol. 16, no. 3, pp. 743–753, May 2005.

6. M. B. Li, G. B. Guang, P. Saratchandran, and N. Sundararajan, "Fully complex extreme learning machine," *Neurocomputing*, vol. 68, pp. 306–314, Oct. 2005.

7. A. Hirose, *Complex Valued Neural Networks*. Berlin: Springer-Verlag, 2006.

8. S. Chen, X. Hong, C. J. Harris, and L. Hanzo, "Fully complex-valued radial basis function networks: Orthogonal least squares regression and classification," *Neurocomputing*, vol. 71, nos. 16-18, pp. 3421–3433, Oct. 2008.

9. T. Nitta, Ed., *Complex-Valued Neural Networks: Utilizing High-Dimensional Parameters*. New York: Information Science Reference, 2009.

10. A. S. Gangal, P. K. Kalra, and D. S. Chauhan, "Inversion of complex valued neural networks using complex back-propagation algorithm," *Int. J. Mathematics and Computers in Simulation*, vol. 3, no. 1, pp. 1–8, 2009.

11. M. Kobayashi, "Exceptional reducibility of complex-valued neural networks," *IEEE Trans. Neural Networks*, vol. 21, no. 7, pp. 1060–1072, July 2010.

12. S. A. Billings, "Identification of nonlinear systems – a survey," *IEE Proc. D*, vol. 127, no. 6, pp. 272–285, Nov. 1980.

13. E. W. Bai, "An optimal two-stage identification algorithm for Hammerstein-Wiener nonlinear systems," *Automatica*, vol. 34, no. 3, pp. 333–338, March 1998.

14. Y. Zhu, "Estimation of an N-L-N Hammerstein-Wiener model," *Automatica*, vol. 38, no. 9, pp. 1607–1614, Sept. 2002.

15. J. Schoukens, J. G. Nemeth, P. Crama, Y. Rolain, and R. Pintelon, "Fast approximate identification of nonlinear systems," *Automatica*, vol. 39, no. 7, pp. 1267–1274, July 2003.

16. K. Hsu, T. Vincent, and K. Poolla, "A kernel based approach to structured nonlinear system identification part I: algorithms," in *Proc. 14th IFAC Symp. System Identification* (Newcastle, Australia), March 29-31, 2006, 6 pages.

17. K. Hsu, T. Vincent, and K. Poolla, "A kernel based approach to structured nonlinear system identification part II: convergence and consistency," in *Proc. 14th IFAC Symp. System Identification* (Newcastle, Australia), March 29-31, 2006, 6 pages.

18. I. W. Hunter and M. J. Korenberg, "The identification of nonlinear biological systems: Wiener and Hammerstein cascade models," *Biological Cybernetics*, vol. 55, nos. 2-3, pp. 135–144, 1986.

19. A. Kalafatis, N. Arifin, L. Wang, and W. R. Cluett, "A new approach to the identification of pH processes based on the Wiener model," *Chemical Engineering Science*, vol. 50, no. 23, pp. 3693–3701, Dec. 1995.

20. A. D. Kalafatis, L. Wang, and W. R. Cluett, "Identification of Wiener-type nonlinear systems in a noisy environment," *Int. J. Control*, vol. 66, no. 7, pp. 923–941, 1997.

21. Y. Zhu, "Distillation column identification for control using Wiener model," in *Proc. 1999 American Control Conference* (San Diego, USA), June 2-4, 1999, pp. 3462–3466.
22. J. C. Gomez, A. Jutan, and E. Baeyens, "Wiener model identification and predictive control of a pH neutralisation process," *IEE Proc. Control Theory and Applications*, vol. 151, no. 3, pp. 329–338, May 2004.
23. A. Hagenblad, L. Ljung, and A. Wills, "Maximum likelihood identification of Wiener models," *Automatica*, vol. 44, no. 11, pp. 2697–2705, Nov. 2008.
24. W. Greblicki, "Nonparametric identification of Wiener systems," *IEEE Trans. Information Theory*, vol. 38, no. 5, pp. 1487–1493, Sept. 1992.
25. I. Skrjanc, S. Blazic, and O. E. Agamennoni, "Interval fuzzy modeling applied to Wiener models with uncertainties," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol. 35, no. 5, pp. 1092–1095, Oct. 2005.
26. G. Farin, *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide.* Fourth Edition. Boston: Academic Press, 1996.
27. C. De Boor, *A Practical Guide to Splines.* New York: Spring Verlag, 1978.
28. T. Kavli, "ASMOD – an algorithm for adaptive spline modelling of observation data," *Int. J. Control*, vol. 58, no. 4, pp. 947–967, 1993.
29. M. Brown and C. J. Harris, *Neurofuzzy Adaptive Modelling and Control.* Hemel Hempstead: Prentice Hall, 1994.
30. C. J. Harris, X. Hong, and Q. Gan, *Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach.* Berlin: Springer-Verlag, 2002.
31. Y. Yang, L. Guo, and H. Wang, "Adaptive statistic tracking control based on two-step neural networks with time delays," *IEEE Trans. Neural Networks*, vol. 20, no. 3, pp. 420–429, March 2009.
32. C. J. Clark, G. Chrisikos, M. S. Muha, A. A. Moulthrop, and C. P. Silva, "Time-domain envelope measurement technique with application to wideband power amplifier modeling," *IEEE Trans. Microwave Theory and Techniques*, vol. 46, no. 12, pp. 2531–2540, Dec. 1998.
33. J. H. K. Vuolevi, T. Rahkonen, and J. P. A. Manninen, "Measurement technique for characterizing memory effects in RF power amplifiers," *IEEE Trans. Microwave Theory and Techniques*, vol. 49, no. 8, pp. 1383–1389, Aug. 2001.
34. C.-H. Lin, H.-H. Chen, Y.-Y. Wang, and J.-T. Chen, "Dynamically optimum lookup-table spacing for power amplifier predistortion linearization," *IEEE Trans. Microwave Theory and Techniques*, vol. 54, no. 5, pp. 2118–2127, May 2006.
35. B. Ai, Z.-Y. Yang, C.-P. Pan, S.-G. Tang, and T. T. Zhang, "Analysis on LUT based predistortion method for HPA with memory," *IEEE Trans. Broadcasting*, vol. 53, no. 1, pp. 127–131, March 2007.
36. L. Ding, G. T. Zhou, D. R. Morgan, Z. Ma, J. S. Kenney, J. Kim, and C. R. Giardina, "A robust digital baseband predistorter constructed using memory polynomials," *IEEE Trans. Communications*, vol. 52, no. 1, pp. 159–165, Jan. 2004.
37. D. Zhou and V. E. DeBrunner, "Novel adaptive nonlinear predistorters based on the direct learning algorithm," *IEEE Trans. Signal Processing*, vol. 55, no. 1, pp. 120–133, Jan. 2007.

38. V. P. G. Jiménez, Y. Jabrane, A. G. Armada, and B. Ait Es Said, “High power amplifier pre-distorter based on neural-fuzzy systems for OFDM signals,” *IEEE Trans. Broadcasting*, vol. 57, no. 1, pp. 149–158, March 2011.

39. S. Chen, “An efficient predistorter design for compensating nonlinear memory high power amplifier,” *IEEE Trans. Broadcasting*, vol. 57, no. 4, pp. 856–865, Dec. 2011.

40. X. Hong and S. Chen, “Modeling of complex-valued Wiener systems using B-spline neural network,” *IEEE Trans. Neural Networks*, vol. 22, no. 5, pp. 818–825, May 2011.

41. B. Igelnik, “Kolmogorov’s spline complex network and adaptive dynamic modeling of data,” in: T. Nitta, Ed., *Complex-Valued Neural Networks: Utilizing High-Dimensional Parameters.* New York: Information Science Reference: 2009, pp. 56–78.

42. M. Scarpiniti, D. Vigliano, R. Parisi, and A. Unicinis, “Flexible blind signal separation in the complex domain,” in: T. Nitta, Ed., *Complex-Valued Neural Networks: Utilizing High-Dimensional Parameters.* New York: Information Science Reference, 2009, pp. 284–323.

43. A. A. M. Saleh, “Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers,” *IEEE Trans. Communications*, vol. COM-29, no. 11, pp. 1715–1720, Nov. 1981.

44. M. Honkanen and S.-G. Häggman, “New aspects on nonlinear power amplifier modeling in radio communication system simulations,” in *Proc. PIMRC’97* (Helsinki, Finland), Sept. 1-4, 1997, pp. 844–848.

45. P. M. Grant, S. McLaughlin, H. Aghvami, and S. Fletcher, “Green radio – towards sustainable wireless networks,” *Mobile VCE Core 5 Programme* Presentation, April 2009. Available on-line from `http://www.ee.princeton.edu/seminars/iss/Spring2009/slides/grant.pdf`

46. L. Hanzo, S. X. Ng, T. Keller, and W. Webb, *Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-Coded, Turbo-Equalised and Space-Time Coded OFDM, CDMA and MC-CDMA Systems.* Chichester, UK: John Wiley, 2004.

47. L. Hanzo, M. Münster, B. J. Choi, and T. Keller, *OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting.* Chichester, UK: John Wiley, 2003.