Modelling and Inverting Complex-Valued Wiener Systems

Xia Hong, Sheng Chen and Chris J. Harris

***Abstract*— We develop a complex-valued (CV) B-spline neural network approach for efficient identification and inversion of CV Wiener systems. The CV nonlinear static function in the Wiener system is represented using the tensor product of two univariate B-spline neural networks. With the aid of a least squares parameter initialisation, the Gauss-Newton algorithm effectively estimates the model parameters that include the CV 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. An accurate inverse of the CV Wiener system is then obtained, in which the inverse of the CV nonlinear static function of the Wiener system is calculated efficiently using the Gaussian-Newton algorithm based on the estimated B-spline neural network model, with the aid of the De Boor recursions. The effectiveness of our approach for identification and inversion of CV Wiener systems is demonstrated using the application of digital predistorter design for high power amplifiers with memory.**

I. Introduction

Complex-valued (CV) neural networks have been studied theoretically and applied in nonlinear signal and data processing [1]–[11]. Note that most 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. A number of analytic functions were introduced for the fully CV multilayer perceptrons [4]. A full CV radial basis function nework was introduced in [8] for regression and classification. Alternatively, two RV neural networks can be used, one processing the real part and the other processing the imaginary part of the CV signal/system. A much more challenging problem is to invert a CV nonlinear system, which is required in many communication signal processing applications. This is an under-researched area, and a few existing methods, such as the algorithm given in [10], are not very effective in tackling practical CV signal processing problems.

Many practical communication applications involve propagating CV signals through CV nonlinear dynamic systems that is represent by the Wiener model. For example, at the transmitter of wireless systems, the transmitted signal is distorted by the high power amplifier (HPA) that can be characterised by the CV Wiener model [12]. Some nonlinear communication channels can also be modelled by a linear filter followed by a CV static nonlinear function. Accurate identification of a CV Wiener model is required in these applications. Moreover, an accurate inverse of the CV Wiener system based on the estimated model is necessary, such as in digital predistorter design for compensating the distortions of the HPA at the transmitter [13]–[17] and deconvolution or equalisation at the receiver [2], [3]. Our previous work [18] developed an efficient B-spline neural network approach for identification of CV Wiener systems.

Our method [18] represents the CV nonlinear static function in the Wiener system using the tensor product of two univariate B-spline neural networks. By minimising the sum of squared errors between the model output and the system output, the Gauss-Newton algorithm, coupled with a least squares (LS) parameter initialisation, is readily applicable for the parameter estimation of the proposed CV Wiener model, which naturally incorporates the De Boor recursions for both the B-spline curves and first order derivatives [19]. Our method extends the B-spline model from the RV domain to accommodate general CV Wiener systems, and our proposed CV B-spline function model has a significant advantage over many other modelling paradigms in that it enables stable and efficient evaluations of functional and derivative values. It is clear different from the existing CV neural network based on spline functions [3], [20], [21], in both model representation and identification algorithm.

The contribution of this work is to develop an effective method for inverting the CV Wiener system. We demonstrate that the B-spline neural network scheme for modelling of CV Wiener systems proposed in [18] has a further significant advantage in that it can directly be utilised to calculate an accurate inverse of the CV Wiener system in an very efficient way. In particular, the inverse of the CV nonlinear static function in the Wiener model is calculated efficiently 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 inversion of CV Wiener systems is illustrated using the application of digital predistorter design for broadband communication systems that employ power-efficient nonlinear HPA transmitter.

We represent a CV number $x \in \mathbb{C}$ either by the rectangular form $x = x_R + \mathrm{j}x_I$, where $\mathrm{j} = \sqrt{-1}$, $x_R = \Re[x]$ and $x_I = \Im[x]$, or by the polar form $x = |x| \cdot \exp(\mathrm{j}\angle^x)$ with $|x|$ denoting the amplitude of x and $\angle^x$ its phase.

II. Identification and Inversion Algorithms

Consider the Wiener system consisting of a cascade of two subsystems, a linear filter of order L that represents the

X. Hong is with School of Systems Engineering, University of Reading, Reading RG6 6AY, U.K. (E-mail: x.hong@reading.ac.uk).

S. Chen and C.J. Harris are with Electronics and Computer Science, Faculty of Physical and Applied Sciences, University of Southampton, Southampton SO17 1BJ, U.K. (E-mails: sqc@ecs.soton.ac.uk, cjh@ecs.soton.ac.uk).

S. Chen is also with Faculty of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia.

memory effect on the input signal $x(k) \in \mathbb{C}$, followed by a nonlinear memoryless function $\Psi(\bullet) : \mathbb{C} \rightarrow \mathbb{C}$, as follows

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

$$y(k) = \Psi\left(w(k)\right) + \xi(k), \tag{2}$$

where $y(k) \in \mathbb{C}$ is the system output, and $\xi(k)$ is a CV white noise with $E\big[|\xi_R(k)|^2\big] = E\big[|\xi_I(k)|^2\big] = \sigma_\xi^2$. The z transfer function of the linear filter is defined by $H(z) = \sum\limits_{i=0}^{L} h_i z^{-i}$, with the CV coefficient vector given by $\mathbf{h} = [h_1 \ h_2 \cdots h_L]^{\mathrm{T}}$. We assume that $h_0 = 1$. If this is not true, h_0 can be absorbed into the CV static nonlinearity $\Psi(\bullet)$, and the linear filter's coefficients are re-scaled as h_i/h_0 for $0 \le i \le L$.

We assume that $\Psi(\bullet)$ is a one to one mapping. Furthermore, $y_R(k)$, $y_I(k)$, $w_R(k)$, $w_I(k)$, $x_R(k)$ and $x_I(k)$ are upper and lower bounded by some finite real values. Given the input-output data set $D_N = \{x(k), y(k)\}_{k=1}^{K}$, our aim is to identify the above Wiener system, i.e. to identify the underlying nonlinear function $\Psi(\bullet)$ and to estimate the linear filter parameter vector $\mathbf{h}$, as well as to invert this Wiener system. Note that the signal $w(k)$ between the two subsystems are unavailable. We first use the CV B-spline neural network approach proposed in [18] for an efficient identification of this Wiener system, and then develop an algorithm for an accurate inverse of this Wiener system based on the estimated Wiener model $\widehat{\Psi}(\bullet)$ and $\widehat{\mathbf{h}}$.

A. Complex-valued B-spline neural network

The CV B-spline neural network [18] is used 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 $V_{\min} < w_I < V_{\max}$, with some finite real values $U_{\min}$, $U_{\max}$, $V_{\min}$ and $V_{\max}$.

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

$$\begin{aligned} & U_0 < U_1 < \cdots < U_{P_o-2} < U_{P_o-1} = U_{\min} < U_{P_o} < \cdots \\ & \quad < U_{N_R} < U_{N_R+1} = U_{\max} < U_{N_R+2} < \cdots < U_{N_R+P_o}. \end{aligned} \tag{3}$$

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 (3), the set of N_R B-spline basis functions are formed by using the De Boor recursion [19] as follows: for $1 \le l \le N_R + P_o$,

$$B_l^{(\Re,0)}(w_R) = \begin{cases} 1, & \text{if } U_{l-1} \le w_R < U_l, \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

and for $l = 1, \cdots, N_R + P_o - p$ and $p = 1, \cdots, P_o$,

$$\begin{aligned} B_l^{(\Re,p)}(w_R) = & \frac{w_R - U_{l-1}}{U_{p+l-1} - U_{l-1}} B_l^{(\Re,p-1)}(w_R) \\ & + \frac{U_{p+l} - w_R}{U_{p+l} - U_l} B_{l+1}^{(\Re,p-1)}(w_R). \end{aligned} \tag{5}$$

The derivatives of $B_l^{(\Re,P_o)}(w_R)$ for $1 \le l \le N_R$ can also be computed recursively according to [19]

$$\begin{aligned} \frac{dB_l^{(\Re,P_o)}(w_R)}{dw_R} = & \frac{P_o}{U_{P_o+l-1} - U_{l-1}} B_l^{(\Re,P_o-1)}(w_R) \\ & - \frac{P_o}{U_{P_o+l} - U_l} B_{l+1}^{(\Re,P_o-1)}(w_R). \end{aligned} \tag{6}$$

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

$$\begin{aligned} & V_0 < V_1 < \cdots < V_{P_o-2} < V_{P_o-1} = V_{\min} < V_{P_o} < \cdots \\ & \quad < V_{N_I} < V_{N_I+1} = V_{\max} < V_{N_I+2} < \cdots < V_{N_I+P_o}. \end{aligned} \tag{7}$$

Again, there are $2P_o - 2$ external knots, two boundary knot and $N_I + 1 - P_o$internal knots. The set of N_I B-spline basis functions are similarly constructed by the De Boor recursion as follows: for $1 \le m \le N_I + P_o$,

$$B_m^{(\Im,0)}(w_I) = \begin{cases} 1, & \text{if } V_{m-1} \le w_I < V_m, \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

and for $m = 1, \cdots, N_I + P_o - p$ and $p = 1, \cdots, P_o$,

$$\begin{aligned} B_m^{(\Im,p)}(w_I) = & \frac{w_I - V_{m-1}}{V_{p+m-1} - V_{m-1}} B_m^{(\Im,p-1)}(w_I) \\ & + \frac{V_{p+m} - w_I}{V_{p+m} - V_m} B_{m+1}^{(\Im,p-1)}(w_I), \end{aligned} \tag{9}$$

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

$$\begin{aligned} \frac{dB_m^{(\Im,P_o)}(w_I)}{dw_I} = & \frac{P_o}{V_{P_o+m-1} - V_{m-1}} B_m^{(\Im,P_o-1)}(w_I) \\ & - \frac{P_o}{V_{P_o+m} - V_m} B_{m+1}^{(\Im,P_o-1)}(w_I). \end{aligned} \tag{10}$$

Using the tensor product between the two sets of univariate B-spline basis functions [22], $B_l^{(\Re,P_o)}(w_R)$ and $B_m^{(\Im,P_o)}(w_I)$, a set of new B-spline basis functions $B_{l,m}^{(P_o)}(w)$ is formed, giving rise to the CV B-spline neural network

$$\begin{aligned} \widehat{y} = \widehat{\Psi}(w) = & \sum_{l=1}^{N_R} \sum_{m=1}^{N_I} B_{l,m}^{(P_o)}(w) \omega_{l,m} \\ = & \sum_{l=1}^{N_R} \sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(w_R) B_m^{(\Im,P_o)}(w_I) \omega_{l,m}, \end{aligned} \tag{11}$$

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. Obviously,

$$\widehat{y}_R = \sum_{l=1}^{N_R} \sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(w_R) B_m^{(\Im,P_o)}(w_I) \omega_{R_{l,m}}, \tag{12}$$

$$\widehat{y}_I = \sum_{l=1}^{N_R} \sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(w_R) B_m^{(\Im,P_o)}(w_I) \omega_{I_{l,m}}. \tag{13}$$

$$\frac{\partial \varepsilon_k}{\partial \theta_q} = \begin{cases} \frac{\partial e_R(k)}{\partial \omega_{R_{l,m}}} = -B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k)), q = l\cdot m, 1\le l\le N_R, 1\le m\le N_I, \\ \frac{\partial e_R(k)}{\partial \omega_{I_{l,m}}} = 0, q = N + l\cdot m, 1\le l\le N_R, 1\le m\le N_I, \\ \frac{\partial e_R(k)}{\partial \widehat{h}_{R_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(k))}{d\widehat{w}_R(k)}B_m^{(\Im,P_o)}(\widehat{w}_I(k))x_R(k-i) + B_l^{(\Re,P_o)}(\widehat{w}_R(k))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(k))}{d\widehat{w}_I(k)}x_I(k-i)\Big)\omega_{R_{l,m}}, \\ \qquad q = 2N+i, 1\le i\le L, \\ \frac{\partial e_R(k)}{\partial \widehat{h}_{I_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(-\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(k))}{d\widehat{w}_R(k)}B_m^{(\Im,P_o)}(\widehat{w}_I(k))x_I(k-i) + B_l^{(\Re,P_o)}(\widehat{w}_R(k))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(k))}{d\widehat{w}_I(k)}x_R(k-i)\Big)\omega_{R_{l,m}}, \\ \qquad q = 2N+L+i, 1\le i\le L, \end{cases} \tag{20}$$

$$\frac{\partial \varepsilon_k}{\partial \theta_q} = \begin{cases} \frac{\partial e_I(t)}{\partial \omega_{R_{l,m}}} = 0, q = l\cdot m, 1\le l\le N_R, 1\le m\le N_I, \\ \frac{\partial e_I(t)}{\partial \omega_{I_{l,m}}} = -B_l^{(\Re,P_o)}(\widehat{w}_R(t))B_m^{(\Im,P_o)}(\widehat{w}_I(t)), q = N + l\cdot m, 1\le l\le N_R, 1\le m\le N_I, \\ \frac{\partial e_I(t)}{\partial \widehat{h}_{R_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(t))}{d\widehat{w}_R(t)}B_m^{(\Im,P_o)}(\widehat{w}_I(t))x_R(t-i) + B_l^{(\Re,P_o)}(\widehat{w}_R(t))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(t))}{d\widehat{w}_I(t)}x_I(t-i)\Big)\omega_{I_{l,m}}, \\ \qquad q = 2N+i, 1\le i\le L, \\ \frac{\partial e_I(t)}{\partial \widehat{h}_{I_i}} = -\sum\limits_{l=1}^{N_R}\sum\limits_{m=1}^{N_I}\Big(-\frac{dB_l^{(\Re,P_o)}(\widehat{w}_R(t))}{d\widehat{w}_R(t)}B_m^{(\Im,P_o)}(\widehat{w}_I(t))x_I(t-i) + B_l^{(\Re,P_o)}(\widehat{w}_R(t))\frac{dB_m^{(\Im,P_o)}(\widehat{w}_I(t))}{d\widehat{w}_I(t)}x_R(t-i)\Big)\omega_{I_{l,m}}, \\ \qquad q = 2N+L+i, 1\le i\le L. \end{cases} \tag{21}$$

B. Wiener system identification

For the two chosen sets of knots, (3) and (7), and the polynomial degree P_o, denote the weight vector of the CV B-spline neural network (11) 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 block 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 $\boldsymbol{\theta} = \big[\boldsymbol{\omega}_R^{\mathrm{T}}\ \boldsymbol{\omega}_I^{\mathrm{T}}\ \widehat{\mathbf{h}}_R^{\mathrm{T}}\ \widehat{\mathbf{h}}_I^{\mathrm{T}}\big]^{\mathrm{T}} \in \mathbb{R}^{2(N+L)}$, 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 represents $\Psi(\bullet)$ by

$$\begin{aligned} \widehat{y}(k) &= \widehat{\Psi}\left(\widehat{w}(k)\right) \\ &= \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k))\omega_{l,m}, \end{aligned} \tag{14}$$

which is equivalent to the two RV B-spline models

$$\widehat{y}_R(k) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k))\omega_{R_{l,m}}, \tag{15}$$

$$\widehat{y}_I(t) = \sum_{l=1}^{N_R}\sum_{m=1}^{N_I} B_l^{(\Re,P_o)}(\widehat{w}_R(k))B_m^{(\Im,P_o)}(\widehat{w}_I(k))\omega_{I_{l,m}}, \tag{16}$$

where $\widehat{w}(k) = \big[1\ \widehat{\mathbf{h}}^{\mathrm{T}}\big]\mathbf{x}(k) = \Big(x_R(k) + \sum\limits_{i=1}^{L}\big(\widehat{h}_{R_i}x_R(k-i) - \widehat{h}_{I_i}x_I(k-i)\big)\Big) + \mathrm{j}\Big(x_I(k) + \sum\limits_{i=1}^{L}\big(\widehat{h}_{R_i}x_I(k-i) + \widehat{h}_{I_i}x_R(k-i)\big)\Big)$. Define the error $e(k) = y(k) - \widehat{y}(k)$, yielding the sum of squared errors (SSE) cost function

$$J_{\mathrm{SSE}}(\boldsymbol{\theta}) = \sum_{k=1}^{K}|e(k)|^2 = \sum_{k=1}^{K}\big(e_R^2(k) + e_I^2(k)\big). \tag{17}$$

Gauss-Newton algorithm: Denote $\boldsymbol{\varepsilon} = [\varepsilon_1\ \varepsilon_2\cdots\varepsilon_{2K}]^{\mathrm{T}} = [e_R(1)\ e_R(2)\cdots e_R(K)\ e_I(1)\ e_I(2)\cdots e_I(K)]^{\mathrm{T}} \in \mathbb{R}^{2K}$. By denoting the iteration step with the superscript $^{(\tau)}$ and with an initial value $\boldsymbol{\theta}^{(0)}$, the iteration formula is given by

$$\boldsymbol{\theta}^{(\tau)} = \boldsymbol{\theta}^{(\tau-1)} - \mu\Big(\big(\mathbf{J}^{(\tau)}\big)^{\mathrm{T}}\mathbf{J}^{(\tau)}\Big)^{-1}\big(\mathbf{J}^{(\tau)}\big)^{\mathrm{T}}\boldsymbol{\varepsilon}\big(\boldsymbol{\theta}^{(\tau-1)}\big), \tag{18}$$

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

$$\mathbf{J} = \begin{bmatrix} \frac{\partial\varepsilon_1}{\partial\theta_1} & \frac{\partial\varepsilon_1}{\partial\theta_2} & \cdots & \frac{\partial\varepsilon_1}{\partial\theta_{2(N+L)}} \\ \frac{\partial\varepsilon_2}{\partial\theta_1} & \frac{\partial\varepsilon_2}{\partial\theta_2} & \cdots & \frac{\partial\varepsilon_2}{\partial\theta_{2(N+L)}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial\varepsilon_{2K}}{\partial\theta_1} & \frac{\partial\varepsilon_{2K}}{\partial\theta_2} & \cdots & \frac{\partial\varepsilon_{2K}}{\partial\theta_{2(N+L)}} \end{bmatrix}. \tag{19}$$

The partial derivatives in the Jacobian (19) are calculated at the top of this page: for $1 \le k \le K$, they are given in (20), while for $K+1 \le k \le 2K$ and $t = k - K$, they are given in (21). It is seen that the De Boor algorithm, (4)–(6) and (8)–(10), is applied in evaluating all entries in the Jacobian effectively. The iterative procedure (18) is terminated when $\boldsymbol{\theta}^{(\tau)}$ converges or when a predetermined sufficiently large number of iterations has been reached.

Parameter initialisation: As the cost function (17) is highly nonlinear in the parameters, it is important to properly initialise $\boldsymbol{\theta}^{(0)}$. A simple and effective LS parameter initialisation scheme introduced in [18] is adopted.

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

$$\widetilde{\varphi}\left(y(k)\right)=\sum_{l=1}^{N_R}\sum_{m=1}^{N_I}B_{l,m}^{(P_o)}\left(y(k)\right)\alpha_{l,m}, \qquad (22)$$

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 $\widetilde{w}(k)=x(k)+\sum\limits_{i=1}^{L}\widetilde{h}_i x(k-i)$ is used as the target for $\widetilde{\varphi}(y(k))$. Thus,

$$\begin{aligned}x(k)&=-\sum_{i=1}^{L}\widetilde{h}_i x(k-i)+\sum_{l=1}^{N_R}\sum_{m=1}^{N_I}B_{l,m}^{(P_o)}(y(k))\alpha_{l,m}+\epsilon(k)\\&=\left(\mathbf{p}(\underline{\mathbf{x}}(k))\right)^{\mathrm{T}}\boldsymbol{\vartheta}+\epsilon(k), \qquad (23)\end{aligned}$$

with $\underline{\mathbf{x}}(k)=\left[x(k-1)\cdots\ x(k-L)\ y(k)\right]^{\mathrm{T}}$, $\mathbf{p}\left(\underline{\mathbf{x}}(k)\right)=\left[-x(k-1)\cdots-x(k-L)\quad B_{1,1}^{(P_o)}(y(k))\quad B_{1,2}^{(P_o)}(y(k))\cdots B_{N_R,N_I}^{(P_o)}(y(k))\right]^{\mathrm{T}}=\left[p_1\left(\underline{\mathbf{x}}(k)\right)\ p_2\left(\underline{\mathbf{x}}(k)\right)\cdots p_{N+L}\left(\underline{\mathbf{x}}(k)\right)\right]^{\mathrm{T}}$ $\in\mathbb{C}^{N+L}$, and $\boldsymbol{\vartheta}=\left[\widetilde{h}_1\cdots\widetilde{h}_L\ \alpha_{1,1}\ \ \alpha_{1,2}\cdots\alpha_{l,m}\cdots\alpha_{N_R,N_I}\right]^{\mathrm{T}}$ $=\left[\vartheta_1\ \vartheta_2\cdots\vartheta_{N+L}\right]^{\mathrm{T}}\in\mathbb{C}^{N+L}$.

Over the training data set, (23) can be written in the matrix form as $\overline{\mathbf{x}}=\mathbf{P}\boldsymbol{\vartheta}+\boldsymbol{\epsilon}$, 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

$$\boldsymbol{\vartheta}_{\mathrm{LS}}=\left(\mathbf{P}^{\mathrm{H}}\mathbf{P}\right)^{-1}\mathbf{P}^{\mathrm{H}}\overline{\mathbf{x}}. \qquad (24)$$

The first L CV elements of $\boldsymbol{\vartheta}_{\mathrm{LS}}$ forms the 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.

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

$$\widehat{\widetilde{w}}(k)=x(k)+\sum_{i=1}^{L}\widehat{h}_i^{(0)}x(k-i). \qquad (25)$$

Using the CV B-spline neural network (11) to model $\Psi(\bullet)$ based on the training data set $\{\widehat{\widetilde{w}}(k),y(k)\}_{k=1}^{K}$ yields

$$\begin{aligned}y(k)&=\sum_{l=1}^{N_R}\sum_{m=1}^{N_I}B_{l,m}^{(P_o)}\left(\widehat{\widetilde{w}}(k)\right)\omega_{l,m}+\widehat{e}(k)\\&=\left(\mathbf{q}\left(\widehat{\widetilde{w}}(k)\right)\right)^{\mathbf{T}}\boldsymbol{\omega}+\widehat{e}(k), \qquad (26)\end{aligned}$$

where $\mathbf{q}\left(\widehat{\widetilde{w}}(k)\right)=\left[B_{1,1}^{(P_o)}\left(\widehat{\widetilde{w}}(k)\right)\quad B_{1,2}^{(P_o)}\left(\widehat{\widetilde{w}}(k)\right)\cdots B_{N_R,N_I}^{(P_o)}\left(\widehat{\widetilde{w}}(k)\right)\right]^{\mathrm{T}}=\left[q_1\left(\widehat{\widetilde{w}}(k)\right)\ q_2\left(\widehat{\widetilde{w}}(k)\right)\cdots,q_N\left(\widehat{\widetilde{w}}(k)\right)\right]^{\mathrm{T}}$ $\in\mathbb{R}^N$. Over the training data set, (26) can be written in the matrix form $\mathbf{y}=\mathbf{Q}\boldsymbol{\omega}+\widehat{\mathbf{e}}$, with $\mathbf{y}=[y(1)\cdots y(K)]^{\mathrm{T}}$, $\widehat{\mathbf{e}}=[\widehat{e}(1)\cdots\widehat{e}(K)]^{\mathrm{T}}$ and $\mathbf{Q}=\left[\mathbf{q}\left(\widehat{\widetilde{w}}(1)\right)\cdots\mathbf{q}\left(\widehat{\widetilde{w}}(K)\right)\right]^{\mathrm{T}}$. The LS solution for $\boldsymbol{\omega}\in\mathbb{C}^N$

$$\boldsymbol{\omega}_{\mathrm{LS}}=\left(\mathbf{Q}^{\mathrm{T}}\mathbf{Q}\right)^{-1}\mathbf{Q}^{\mathrm{T}}\mathbf{y} \qquad (27)$$

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 $\boldsymbol{\theta}^{(0)}$.

C. Wiener system inverse

For the CV Wiener system (1) and (2), there are two types of inversion as depicted in Fig. 1. The "pre-inverse" can be found for example in the digital predistorter design for compensating the Wiener HPA [13]–[17], while the "post-inverse" is typically found in the deconvolution or equalisation applications [2], [3]. 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 main difference 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.

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 II-B, the estimate $\widehat{\Psi}(\bullet)$ for $\Psi(\bullet)$ is 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 (15) and (16), we have

$$\widehat{x}_R(k)=\sum_{l=1}^{N_R}\sum_{m=1}^{N_I}B_l^{(\Re,P_o)}(v_R(k))B_m^{(\Im,P_o)}(v_I(k))\omega_{R_{l,m}}, \quad (28)$$

$$\widehat{x}_I(t)=\sum_{l=1}^{N_R}\sum_{m=1}^{N_I}B_l^{(\Re,P_o)}(v_R(k))B_m^{(\Im,P_o)}(v_I(k))\omega_{I_{l,m}}. \quad (29)$$

Define $\zeta(k)=x(k)-\widehat{x}(k)$ and the squared error (SE) $S(k)=\zeta_R^2(k)+\zeta_I^2(k)$. 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 $S(k)$. We propose to use the following Gauss-Newton algorithm to solve this optimisation problem.

Denoting the iteration step with ${}^{(\tau)}$ and giving a randomly chosen $v^{(0)}(k)$ 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

$$\begin{aligned}\begin{bmatrix}v_R^{(\tau)}(k)\\ v_I^{(\tau)}(k)\end{bmatrix}&=\begin{bmatrix}v_R^{(\tau-1)}(k)\\ v_I^{(\tau-1)}(k)\end{bmatrix}-\eta\left(\left(\mathbf{J}_v^{(\tau)}\right)^{\mathrm{T}}\mathbf{J}_v^{(\tau)}\right)^{-1}\\&\quad\times\left(\mathbf{J}_v^{(\tau)}\right)^{\mathrm{T}}\begin{bmatrix}\zeta_R^{(\tau-1)}(k)\\ \zeta_I^{(\tau-1)}(k)\end{bmatrix}, \qquad (30)\end{aligned}$$

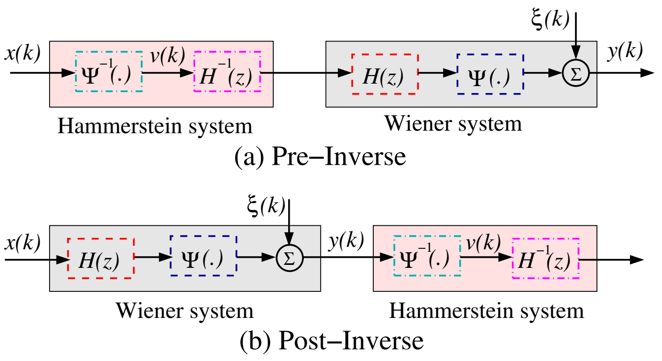

Fig. 1. Schematic of inverse for Wiener system.

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

$$\mathbf{J}_v^{(\tau)} = \left[\begin{array}{cc} \frac{\partial \zeta_R(k)}{\partial v_R(k)} & \frac{\partial \zeta_R(k)}{\partial v_I(k)} \\ \frac{\partial \zeta_I(k)}{\partial v_R(k)} & \frac{\partial \zeta_I(k)}{\partial v_I(k)} \end{array} \right]_{|v(k)=v^{(\tau)}(k)} . \tag{31}$$

The entries in (31) are given by

$$\left\{ \begin{array}{l} \frac{\partial \zeta_R(k)}{\partial v_R(k)} = -\sum\limits_{l=1}^{N_R} \sum\limits_{m=1}^{N_I} \frac{dB_l^{(\Re,P_o)}(v_R(k))}{dv_R(k)} B_m^{(\Im,P_o)}(v_I(k)) \omega_{R_{l,m}}, \\ \frac{\partial \zeta_R(k)}{\partial v_I(k)} = -\sum\limits_{l=1}^{M_R} \sum\limits_{m=1}^{N_I} B_l^{(\Re,P_o)}(v_R(k)) \frac{dB_m^{(\Im,P_o)}(v_I(k))}{dv_I(k)} \omega_{R_{l,m}}, \\ \frac{\partial \zeta_I(k)}{\partial v_R(k)} = -\sum\limits_{l=1}^{N_R} \sum\limits_{m=1}^{N_I} \frac{dB_l^{(\Re,P_o)}(v_R(k))}{dv_R(k)} B_m^{(\Im,P_o)}(v_I(k)) \omega_{I_{l,m}}, \\ \frac{\partial \zeta_I(k)}{\partial v_I(k)} = -\sum\limits_{l=1}^{N_R} \sum\limits_{m=1}^{N_I} B_l^{(\Re,P_o)}(v_R(k)) \frac{dB_m^{(\Im,P_o)}(v_I(t))}{dv_I(k)} \omega_{I_{l,m}}, \end{array} \right. \tag{32}$$

for which the De Boor algorithm, (4)–(6) and (8)–(10), is used for their efficient calculation. The algorithm is terminated when $S(k) < \rho$, where ρ is a preset precision, e.g. $\rho = 10^{-8}$, or when τ reaches a preset maximum value.

Inverse of Wiener system's linear filter: The identification algorithm presented in Subsection II-B also provides the estimate of the Wiener system's linear filter $\widehat{H}(z) = 1 + \sum\limits_{i=1}^{L} \widehat{h}_i z^{-i}$. Let the transfer function of the Hammerstein model's linear filter be $G(z) = z^{-\iota} \cdot \sum\limits_{i=0}^{L_g} g_i z^{-i}$, where the delay $\iota = 0$ if $H(z)$ is minimum phase. The Hammerstein model's linear filter $\mathbf{g} = [g_0 \ g_1 \cdots g_{L_g}]^{\mathrm{T}}$ is readily obtained by solving the set of linear equations specified by

$$G(z) \cdot \hat{H}(z) = z^{-\iota}. \tag{33}$$

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 $h_0 = 1$.

III. Application to Digital Predistorter Design

The operation of HPAs in wireless systems may introduce serious memory effects and nonlinear distortions [12], [23], [24], causing intersymbol interference and adjacent channel interference that degrade the system's achievable bit error rate (BER) performance. The problem becomes particularly acute for high bandwidth-efficiency quadrature amplitude modulation (QAM) systems [25]. It is therefore critical to compensate the distortions caused by the HPA with a digital predistorter in the design of a wireless system [13]–[17].

A. High power amplifier model

A widely used model for HPAs is the Wiener model [12]. Without loss of generality, we consider QAM systems [25], where the CV input signal to the HPA, $x(k)$, takes the values from the CV M-QAM symbol set

$$\mathbb{S} = \{d(2l - \sqrt{M} - 1) + \mathrm{j}d(2q - \sqrt{M} - 1), 1 \le l, q \le \sqrt{M}\}, \tag{34}$$

where $2d$ is the minimum distance between symbol points. The memory effect of the Wiener HPA is modelled by the linear filter (1), while the nonlinear saturating distortion of the Wiener HPA is represented by the static nonlinearity (2). In practical HPAs, the noise $\xi(k)$ is negligible, i.e. σ_ξ^2 is zero or extremely small. Two typical CV nonlinearities $\Psi(\bullet)$ of HPAs are the travelling-wave tube (TWT) nonlinearity [23] and the nonlinearity of solid state power amplifiers [24]. We consider the TWT static nonlinearity, but the approach is equally applicable to the other type of nonlinearity.

Express the (unavailable) input signal $w(k)$ to the static nonlinearity $\Psi(\bullet)$ by $w(k) = r(k) \cdot \exp(\mathrm{j}\psi(k))$. 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 $r(k) = |w(k)|$, yielding

$$y(k) = A(r(k)) \cdot \exp(\mathrm{j}(\psi(k) + \Phi(r(k)))). \tag{35}$$

The output amplitude $|y(k)| = A(r(k))$ and the phase $\Phi(r(k)) = \angle^{y(k)} - \psi(k)$ of the HPA's static nonlinearity are specified respectively by [12], [17], [23]

$$A(r) = \left\{ \begin{array}{ll} \frac{\alpha_a r}{1+\beta_a r^2}, & 0 \le r \le r_{\mathrm{sat}}, \\ A_{\max}, & r > r_{\mathrm{sat}}, \end{array} \right. \tag{36}$$

$$\Phi(r) = \frac{\alpha_\phi r^2}{1 + \beta_\phi r^2}, \tag{37}$$

where the saturating input amplitude is defined as

$$r_{\mathrm{sat}} = \frac{1}{\sqrt{\beta_a}}, \tag{38}$$

while the saturation output amplitude is given by

$$A_{\max} = \frac{\alpha_a}{2\sqrt{\beta_a}}. \tag{39}$$

From the underlying physics, $A_{\max} > r_{\mathrm{sat}}$ and the input amplitude $r < R_{\max}$, 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 defined by the input back-off (IBO)

$$\mathrm{IBO} = 10 \cdot \log_{10} \frac{P_{\mathrm{sat}}}{P_{\mathrm{avg}}}, \tag{40}$$

where $P_{\mathrm{sat}} = r_{\mathrm{sat}}^2$ is the saturation input power and P_{avg} is the average power of the signal $w(k)$ at the input of the TWT nonlinearity, which is equal to the average power of $x(k)$ scaled by the linear filter power gain $1 + \|\mathbf{h}\|^2$.

B. A novel digital predistorter design

Based on the technique developed in Section II for identification and inversion of the CV Wiener system, a 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 $\widehat{\Psi}(\bullet)$. Moreover, the measurement $y(k)$ is usually noise free, i.e. $\sigma_\xi^2 = 0.0$. 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 (3) and (7) 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 empirically. 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 II-C based on the inverse of De Door algorithm converges to the unique solution $\widehat{\Psi}^{-1}(\bullet)$. For the M-QAM signal $x(k)$ of (34), $v(k) = \widehat{\Psi}^{-1}(x(k))$ has M distinct values, and these values can be pre-calculated off-line and stored for on-line transmission.

C. Simulation results

We considered the 16-QAM system with the static nonlinearity of the Wiener HPA described by (36) and (37). The parameters of the Wiener HPA were given as

$$\begin{aligned} \mathbf{h}^{\mathrm{T}} &= [0.75 + \mathrm{j}0.2 \;\; 0.15 + \mathrm{j}0.1 \;\; 0.08 + \mathrm{j}0.01], \\ \mathbf{t}^{\mathrm{T}} &= [2.1587 \;\; 1.15 \;\; 4.0 \;\; 2.1]. \end{aligned} \tag{41}$$

The serious nonlinear and memory distortions caused by this HPA are illustrated in Figs. 2 and 3. For IBO= 0 dB, the HPA is operating well into the saturation region.

TABLE I

IDENTIFICATION RESULTS FOR THE LINEAR FILTER, $\mathbf{h}$, OF THE HPA.

true parameter vector:
$\mathbf{h}^{\mathrm{T}} = [0.7500 + \mathrm{j}0.2000 \;\; 0.1500 + \mathrm{j}0.1000 \;\; 0.0800 + \mathrm{j}0.0010]$
estimate under IBO= 0 dB:
$\widehat{\mathbf{h}}^{\mathrm{T}} = [0.7502 + \mathrm{j}0.1996 \;\; 0.1499 + \mathrm{j}0.0999 \;\; 0.0800 + \mathrm{j}0.0008]$
estimate under IBO= 4 dB
$\widehat{\mathbf{h}}^{\mathrm{T}} = [0.7502 + \mathrm{j}0.2001 \;\; 0.1501 + \mathrm{j}0.1001 \;\; 0.0800 + \mathrm{j}0.0011]$

Results of HPA identification: Two 16-QAM training sets each containing $K = 3000$ samples were generated given the HPA's parameters (41) and with the HPA operating at IBO= 4 dB and 0 dB, 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 $\sqrt{N} = 8$. For this HPA, we used the knot sequence $\{-12.0, -6.0, -2.0, -\mathbf{1.2}, -0.6, -0.3, 0.0, 0.3, 0.6, \mathbf{1.2}, 2.0, 6.0, 12.0\}$ with $R_{\max} = 1.2$. The Gauss-Newton identification algorithm with the LS parameter initialisation was carried out. The results obtained are summarised in Table I as well as illustrated in Figs. 4 and 5, which confirm that an accurate HPA model was obtained.

It can be seen from Fig. 5 that the estimated response of the B-spline neural network model $\widehat{\Psi}(\bullet)$ exhibit small deviations from the HPA's true amplitude response $A(r)$ and phase response $\Phi(r)$ in the region $r > R_{\max}$. This is because, under the condition of IBO= 4 dB, there were relative few data points which yielded the signal amplitude $r(k) = |w(k)|$ near or over the saturation value r_{sat}. Interestingly, under the operating condition of IBO= 0 dB, the deviation between the estimated response and the true response no longer exists

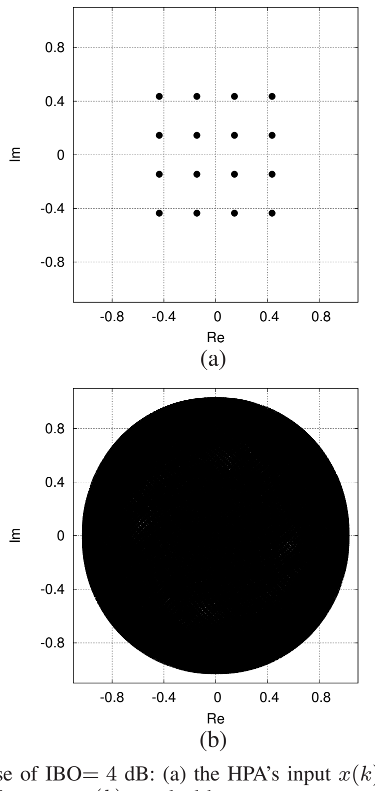

Fig. 2. The case of IBO= 4 dB: (a) the HPA's input $x(k)$, marked by $\bullet$, and (b) the HPA's output $y(k)$, marked by $\times$.

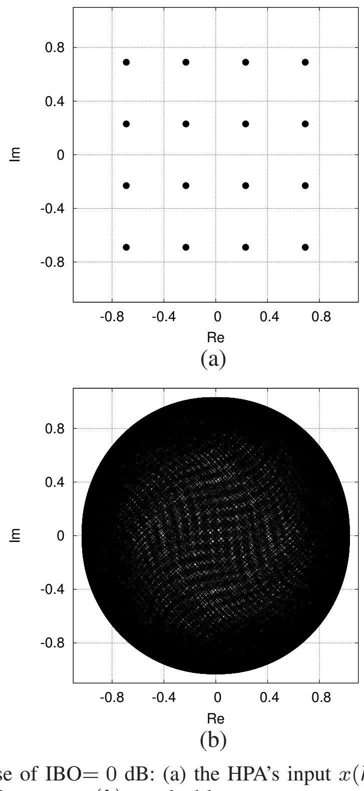

Fig. 3. The case of IBO= 0 dB: (a) the HPA's input $x(k)$, marked by $\bullet$, and (b) the HPA's output $y(k)$, marked by $\times$.

in the region of $r > R_{\max}$, as can be noted from Fig. 4. Instead, small deviation is seen in Fig. 4 between the true phase response and the estimated phase response at the region of small r. This is because there were relative few data points with small signal amplitudes $r(k) = |w(k)|$ under the condition of IBO$= 0$ dB.

Results of digital predistorter solution: We employed the estimated CV B-spline Wiener HPA model 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 symbol constellation using the Gauss-Newton algorithm based on the De Boor inversion. 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. 6 for the HPA's operating conditions of IBO$= 4$ dB and 0 dB, respectively. The achievable performance of the designed predistorter was further assessed using the MSE metric defined by

$$\text{MSE} = 10 \log_{10} \Big(\frac{1}{K_{\text{test}}} \sum_{k=1}^{K_{\text{test}}} |x(k) - y(k)|^2 \Big), \tag{42}$$

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 $\text{SNR} = 10 \log_{10} \big(\text{E}_\text{b}/\text{N}_\text{o}\big)$, 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 (42), and the resulting MSE as the function of IBO is plotted in Fig. 7. The output signal after the HPA was then transmitted over the AWGN channel, and the BER was determined at the receiver. The results obtained are plotted in Fig. 8, in comparison with the benchmark BER of the ideal AWGN channel. It can be seen from Fig. 8 that the BER of the combined predistorter and HPA system is practically indistinguishable from that of the ideal AWGN channel even under the operating condition of IBO $= 0$ dB.

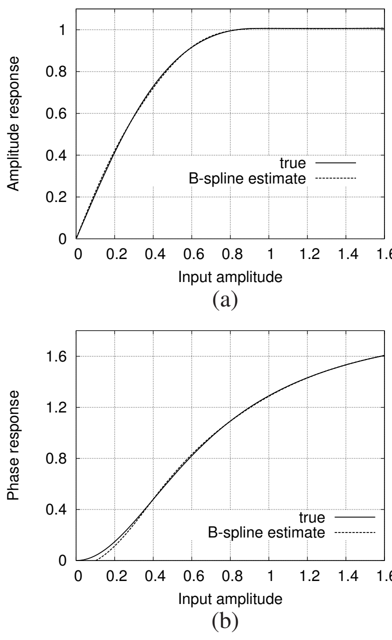

Fig. 4. Comparison of the HPA's static nonlinearity $\Psi(\bullet)$ and the estimated static nonlinearity $\widehat{\Psi}(\bullet)$ under IBO$= 0$ dB: (a) the amplitude response, and (b) the phase response.

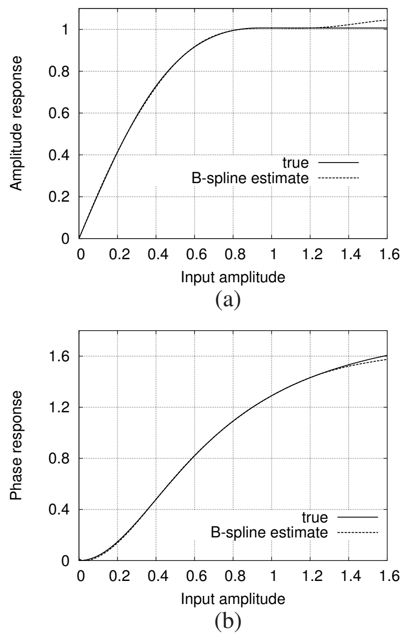

Fig. 5. Comparison of the HPA's static nonlinearity $\Psi(\bullet)$ and the estimated static nonlinearity $\widehat{\Psi}(\bullet)$ under IBO$= 4$ dB: (a) the amplitude response, and (b) the phase response.

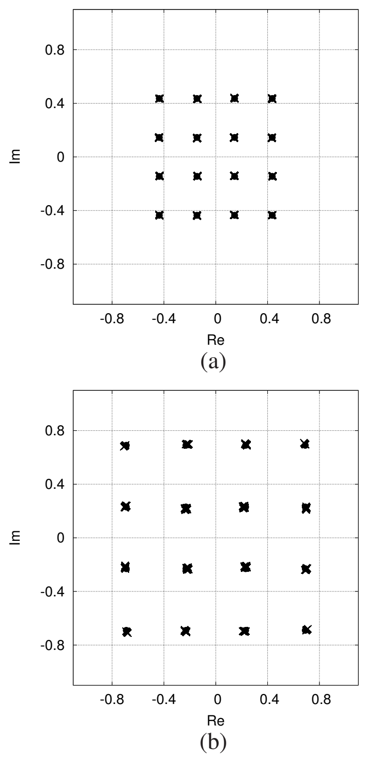

Fig. 6. The output of the combined predistorter and HPA $y(k)$, marked by $\times$, for the 16-QAM input signal $x(k)$, marked by $\bullet$: (a) the IBO of 4 dB, and (b) the IBO of 0.0 dB.

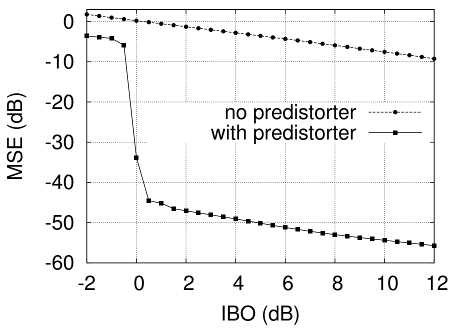

Fig. 7. The mean square error versus IBO performance.

IV. Conclusions

A novel scheme has been proposed for identifying and inverting CV Wiener systems. Firstly, accurate identification of CV Wiener systems has been achieved using the CV B-spline neural network approach based on the Gauss-Newton algorithm, with the aid of a simple LS parameter initialisation. The identification algorithm naturally incorporates the efficient De Boor algorithm with both the B-spline curve and first order derivative recursions. Then an accurate inverting technique has been developed for CV Wiener systems. In particular, the inverse of the CV nonlinear static function in the Wiener system is calculated efficiently using the Gaussian-Newton algorithm based on the estimated B-spline neural network model with the aid of the De Boor 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 inverting CV 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.

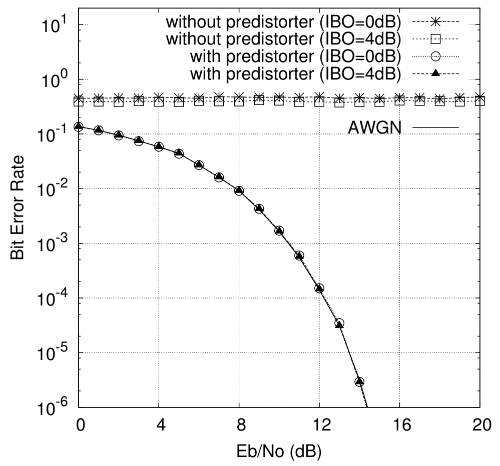

Fig. 8. The bit error rate versus channel SNR performance.

[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, no. 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] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] C. De Boor, *A Practical Guide to Splines*. New York: Spring Verlag, 1978.

[20] 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.

[21] 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.

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

[23] 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.

[24] 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.

[25] 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.