Neurocomputing 82 (2012) 216–223

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

The system identification and control of Hammerstein system using non-uniform rational B-spline neural network and particle swarm optimization ☆

Xia Hong [a,*], Sheng Chen [b,c]

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

ARTICLE INFO

Article history:
Received 24 August 2011
Received in revised form
26 October 2011
Accepted 13 November 2011
Communicated by K. Li
Available online 27 December 2011

Keywords:
B-spline
NURB neural networks
De Boor algorithm
Hammerstein model
Pole assignment controller
Particle swarm optimization
System identification

ABSTRACT

In this paper a new system identification algorithm is introduced for Hammerstein systems based on observational input/output data. The nonlinear static function in the Hammerstein system is modelled using a non-uniform rational B-spline (NURB) neural network. The proposed system identification algorithm for this NURB network based Hammerstein system consists of two successive stages. First the shaping parameters in NURB network are estimated using a particle swarm optimization (PSO) procedure. Then the remaining parameters are estimated by the method of the singular value decomposition (SVD). Numerical examples including a model based controller are utilized to demonstrate the efficacy of the proposed approach. The controller consists of computing the inverse of the nonlinear static function approximated by NURB network, followed by a linear pole assignment controller.

1. Introduction

The Hammerstein model, comprising a nonlinear static functional transformation followed by a linear dynamical model, has been widely researched [1–9]. It is a popular nonlinear plant/process modelling approach for a wide range of biological/engineering problems [10–13]. For example, it is a suitable model for signal processing applications involving any nonlinear distortion followed by a linear filter, the modelling of the human heart in order to regulate the heart rate during treadmill exercises [14] and the modelling of hydraulic actuator friction dynamics [15].

The model characterization/representation of the unknown nonlinear static function is fundamental to the identification of Hammerstein model. Various approaches have been developed in order to capture the *a priori* unknown nonlinearity by use of both parametric [8,9] and nonparametric methods [6,7,16]. In the parametric approaches the unknown nonlinear function is restricted by some parametric representation with a finite number of parameters. In particular, the nonlinear subsystem often has a predetermined linear in the parameters model structure. The special structure of Hammerstein models can be exploited to develop hybrid parameter estimation algorithms [3,9,17]. It has been shown that the Bernstein basis used in Bezier curve is the best conditioned and the most stable among any other polynomial basis [18]. Similar to Bezier curve, both the uniform/nonrational B-spline curve and the non-uniform/rational B-spline (NURB) curve have also been widely used in computer graphics and computer aided geometric design (CAGD) [19]. These curves consist of many polynomial pieces, offering much more versatility than do Bezier curves while maintaining the same advantage of the best conditioning property. The early work on the construction of B-spline basis functions is mathematically involved and numerically unstable [20]. De Boor algorithm uses recurrence relations and is numerically stable [20]. NURB is a generalization of the uniform, nonrational B-splines, and offers much more versatility and powerful approximation capability. Both B-spline and NURB curves can be evaluated quickly using De Boor algorithms. The system identification algorithm for the Hammerstein model has been introduced based on the Bezier–Bernstein approximation and the inverse of de Casteljau's algorithm [21,22]. The nonrational B-spline neural networks have been widely applied for

☆This work was partly supported by UK EPSRC.
* Corresponding author. Tel.: +44 118 378 8222.
E-mail addresses: x.hong@reading.ac.uk (X. Hong), sqc@ecs.soton.ac.uk (S. Chen).

0925-2312/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2011.11.016

nonlinear dynamical systems [23–25], including the modelling and control of Hammerstein systems [26].

Alternatively rational models have been well researched in the context of modelling and control of general nonlinear dynamically systems [27–31]. These models provide a more concise description to some systems than polynomial models, and can be more appropriate models for certain applications. However the structure detection and parameter estimation are very challenging and need special treatments [27–31]. In comparison to nonrational B-splines neural networks, the rational functions used in [27–31] are not based on polynomial pieces, so they may be less versatile due to its global nature.

Note that for the identification of the Hammerstein model based on the uniform, nonrational B-splines neural network, the optimization of model output with respect to the number/location of knots is an intractable mixed integer problem. With the number of knots and their location determined, conventional nonlinear optimization algorithms are applicable for determining the parameters in the B-spline function based Hammerstein model. If there is severe local nonlinearity, the location of knots need to be empirically set by the user by inserting more knots at higher density in regions with high curvatures. These regions should be identified by trial and error during an iterative modelling process. Clearly this trial and error approach cannot yield the optimum solution.

The NURB neural network possesses a much more powerful modelling capability than a conventional nonrational B-spline neural network because of the extra shaping parameters. This motivates us to propose the use of NURB neural networks to model the nonlinear static function in the Hammerstein system. The positiveness constraints are imposed on the shaping parameters in the NURB model in order to avoid singularity in the model. The assertion is that the severe local nonlinearity can be approximated better by optimizing the shaping parameters in the NURB neural networks, hence the need of optimizing the number/location of knots could be relaxed. This alleviates tractability issue, e.g. for optimizing number/location of knots, because all the adjustable parameters are continuous variables and the conventional nonlinear optimization algorithms are applicable for their estimation. However the joint estimation of all parameters in the Hammerstein based on NURB neural networks, subject to constraints, is still difficult. This motivates us to develop a hybrid parameter estimation algorithm that is simple to implement, by exploiting the special structure of resultant NURB neural network based Hammerstein model.

This paper introduces a hybrid system identification consisting two successive stages. We note that the model output can be represented as a linear in the parameters model once the shaping parameters are fixed. This means that the mean squares error due to the shaping parameters can be easily obtained using the least squares method, without explicitly estimating the other parameters. In the proposed algorithm the shaping parameters in NURB neural networks are estimated using the particle swarm optimization (PSO) as the first step, in which the mean square error is used as the cost function. The PSO [32,33] constitutes a population based stochastic optimization technique, which was inspired by the social behaviour of bird flocks or fish schools. The algorithm commences with random initialisation of a swarm of individuals, referred to as particles, within the specific problem's search space. It then endeavours to find a globally optimum solution by gradually adjusting the trajectory of each particle towards its own best location and towards the best position of the entire swarm at each optimization step. The PSO method is popular owing to its simplicity in implementation, ability to rapidly converge to a "reasonably good" solution and to "steer clear" of local minima. It has been successfully applied to wide-ranging optimization problems [34–38]. In order to satisfy the shaping parameter constraints, the normalisation are applied in PSO as appropriate. Once the shaping parameters are determined. The remaining parameters can be estimated by Bai's overparametrization approach [3], or the Gauss–Newton algorithm subject to constraints as proposed in [22]. We used Bai's overparametrization approach [3] in this study.

For completeness a model based controller is utilized to demonstrate the efficacy of the proposed approach. A popular treatment of handling the Hammerstein model is to remove the nonlinearity via an inversion [39–41]. In this study, the controller consists of computing the inverse of the nonlinear static function approximated by NURB, followed by a linear pole assignment controller. The linearization of the closed loop system is achieved by inserting the inverse of the identified static nonlinearity via the inverse of De Boor algorithm [26] which was introduced for the control of B-spline based Hammerstein systems. It is shown that the inverse of De Boor algorithm [26] is also applicable to NURB based Hammerstein systems.

2. The Hammerstein system

The Hammerstein system, as shown in Fig. 1, consists of a cascade of two subsystems, a nonlinear memoryless function $\Psi(\bullet)$ as the first subsystem, followed by a linear dynamic part as the second subsystem. The system can be represented by

$$y(t)=\hat{y}(t)+\xi(t)=-a_1y(t-1)-a_2y(t-2)-\cdots-a_{n_a}y(t-n_a) +b_1v(t-1)+\cdots+b_{n_b}v(t-n_b)+\xi(t) \quad (1)$$

$$v(t-j)=\Psi(u(t-j)), \quad j=1,\ldots,n_b \quad (2)$$

where $y(t)$ is the system output and $u(t)$ is the system input. $\xi(t)$ is assumed to be a white noise sequence independent of $u(t)$ with zero mean and variance of σ^2. $v(t)$ is the output of nonlinear subsystem and the input to the linear subsystem. a_j's, b_j's are parameters of the linear subsystem. n_a and n_b are assumed known system output and input lags. Denote $\mathbf{a}=[a_1,\ldots,a_{n_a}]^T\in\Re^{n_a}$ and $\mathbf{b}=[b_1,\ldots,b_{n_b}]^T\in\Re^{n_b}$. It is assumed that $A(q^{-1})=1+a_1q^{-1}+\cdots+a_{n_a}q^{-n_a}$ and $B(q^{-1})=b_1q^{-1}+\cdots+b_{n_b}q^{-n_b}$ are coprime polynomials of q^{-1}, where q^{-1} denotes the backward shift operator. The gain of the linear subsystem is given by

$$G=\lim_{q\to1}\frac{B(q^{-1})}{A(q^{-1})}=\frac{\sum_{j=1}^{n_b}b_j}{1+\sum_{j=1}^{n_a}a_j} \quad (3)$$

The two objectives of the work are that of the system identification and the subsequent controller design for the identified model. The objective of system identification for the above Hammerstein model is that, given an observational input/output data set $D_N=\{y(t),u(t)\}_{t=1}^N$, to identify $\Psi(\bullet)$ and to estimate the parameters a_j, b_j in the linear subsystems. Note that the signals between the two subsystems are unavailable.

Without significantly losing generality the following assumptions are initially made about the problem:

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

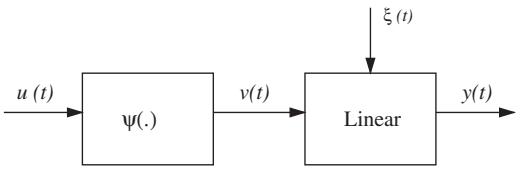

Fig. 1. The Hammerstein system.

Assumption 2. $u(t)$ is bounded by $U_{min} < u(t) < U_{max}$, where U_{min} and U_{max} are assumed known finite real values.

Assumption 3. The persistence excitation condition is given by

$$\text{rank}\begin{pmatrix} u(n_a+n_b) & \cdots & u(n_a+1) & y(n_a+n_b) & \cdots & y(n_b+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u(N-1) & \cdots & u(N-n_b) & y(N-1) & \cdots & y(N-n_a) \end{pmatrix} = n_a+n_b \tag{4}$$

3. Modelling of Hammerstein system using NURB neural network

In this work the non-uniform rational B-spline (NURB) neural network is adopted in order to model $\Psi(\bullet)$. De Boor's algorithm is a fast and numerically stable algorithm for evaluating B-spline basis functions [20]. Univariate B-spline basis functions are parameterized by the order of a piecewise polynomial of order k, and also by a knot vector which is a set of values defined on the real line that break it up into a number of intervals. Supposing that there are d basis functions, the knot vector is specified by $(d+k)$ knot values, $\{U_1, U_2, \ldots, U_{d+k}\}$. At each end there are k knots satisfying the condition of being external to the input region, and as a result the number of internal knots is $(d-k)$. Specifically

$$\begin{aligned} U_1 < U_2 < U_k = U_{min} < U_{k+1} < U_{k+2} \\ < \cdots < U_d < U_{max} = U_{d+1} < \cdots < U_{d+k} \end{aligned} \tag{5}$$

Given these predetermined knots, a set of d B-spline basis functions can be formed by using the De Boor recursion [20], given by

$$\mathcal{B}_j^{(0)}(u) = \begin{cases} 1 & \text{if } U_j \le u < U_{j+1} \\ 0 & \text{otherwise} \end{cases} \qquad j = 1, \ldots, (d+k) \tag{6}$$

$$\left.\begin{aligned} \mathcal{B}_j^{(i)}(u) &= \frac{u-U_j}{U_{i+j}-U_j}\mathcal{B}_i^{(i-1)}(u) \\ &+ \frac{U_{i+j+1}-u}{U_{i+j+1}-U_{j+1}}\mathcal{B}_{j+1}^{(i-1)}(u), \\ j &= 1, \ldots, (d+k-i) \end{aligned}\right\} \quad i = 1, \ldots, k \tag{7}$$

We model $\Psi(\bullet)$ as the NURB neural network in the form of

$$\Psi(u) = \sum_{j=1}^{d} \mathcal{N}_j^{(k)}(u)\omega_j \tag{8}$$

with

$$\mathcal{N}_j^{(k)}(u) = \frac{\lambda_j \mathcal{B}_j^{(k)}(u)}{\sum_{j=1}^{d} \lambda_j \mathcal{B}_j^{(k)}(u)} \tag{9}$$

where ω_j's are weights, $\lambda_j > 0$'s the shaping parameters that are to be determined. Denote $\boldsymbol{\omega} = [\omega_1, \ldots, \omega_d]^T \in \Re^d$. $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_d]^T \in \Re^d$. For uniqueness we set the constraint $\sum_{j=1}^{d} \lambda_j = 1$. Note that due to the piecewise nature of B-spline functions, there are only $(k+1)$ basis functions with non-zero values for any point u. Hence the computational cost for the evaluation of $\Psi(u)$ based on the De-Boor algorithm is determined by the polynomial order k, rather than the number of knots, and this is in the order of $O(k^2)$.

The optimization of model output with respect to the number/location of knots is an intractable mixed integer problem. With the number of knots and their location determined, conventional nonlinear optimization algorithms are applicable for determining the weights and the shaping parameters. Note that if $\lambda_j = 1/d$ $(\forall j)$ the NURB network based Hammerstein systems becomes a nonrational B-spline based Hammerstein systems [26], for which the system identification can be carried out iteratively in practice. The number and locations of knots are predetermined to produce a model as small as possible that can still provide good modelling capability. The model performance may not be particularly sensitive to the location of knots if these are evenly spread out, and there is no severe local nonlinearity. However, if there is severe local nonlinearity, the location of knots need to be empirically set by the user by inserting more knots at higher density in regions with high curvatures. These regions can be identified by trial and error during the iterative modelling process.

Our algorithm involves estimating the weights and the shaping parameters in the NURB model. Note that the proposed NURB neural network possesses a much more powerful modelling capability than a nonrational B-spline network because of the extra shaping parameters. The assumption is that even if there is severe local nonlinearity it is possible to improve modelling accuracy by adjusting the associated shaping parameters. This is advantageous because all the parameters are continuous variables that can be solved by nonlinear optimization, compared to presetting the knots by trial and error which does not yield to the optimum.

With specified knots and over the estimation data set D_N, $\boldsymbol{\lambda}, \boldsymbol{\omega}, \mathbf{a}, \mathbf{b}$ may be jointly estimated via

$$\min_{\boldsymbol{\lambda},\boldsymbol{\omega},\mathbf{a},\mathbf{b}} \left\{ J = \sum_{t=1}^{N} (y - \hat{y}(t, \boldsymbol{\lambda}, \boldsymbol{\omega}, \mathbf{a}, \mathbf{b}))^2 \right\} \tag{10}$$

subject to

$$\lambda_j \ge 0 \quad \forall j, \quad \boldsymbol{\lambda}^T \mathbf{1} = 1 \text{ and } G = 1 \tag{11}$$

in which $G=1$ is imposed for unique solution. We point out that this is still a very difficult nonlinear optimization problem due to the mixed constraints, and this motivates us to propose the following hybrid procedure. It is proposed that the shaping parameters λ_j's are found using the PSO, as the first step of system identification, followed by the estimation of the remaining parameters.

4. The system identification of Hammerstein system based on NURB using PSO

4.1. The basic idea

Initially consider using NURB approximation with a specified shape parameter vector $\boldsymbol{\lambda}$, the model predicted output $\hat{y}(t)$ in (1) can be written as

$$\begin{aligned} \hat{y}(t) = &-a_1 y(t-1) - a_2 y(t-2) - \cdots - a_{n_a} y(t-n_a) + b_1 \sum_{j=1}^{d} \omega_j \mathcal{N}_j^{(k)}(t-1) \\ &+ \cdots + b_{n_b} \sum_{j=1}^{d} \omega_j \mathcal{N}_j^{(k)}(t-n_b) \end{aligned} \tag{12}$$

Over the estimation data set $D_N = \{y(t), u(t)\}_{t=1}^{N}$, (1) can be rewritten in a linear regression form

$$y(t) = [\mathbf{p}(\mathbf{x}(t))]^T \boldsymbol{\vartheta} + \xi(t) \tag{13}$$

where $\mathbf{x}(t) = [-y(t-1), \ldots, -y(t-n_a), u(t-1), \ldots, u(t-n_b)]^T$ is system input vector of observables with assumed known dimension of (n_a+n_b), $\boldsymbol{\vartheta} = [\mathbf{a}^T, (b_1\omega_1), \ldots, (b_1\omega_d), \ldots (b_{n_b}\omega_1), \ldots, (b_{n_b}\omega_{n_b})]^T \in \Re^{n_a + d \cdot n_b}$,

$$\begin{aligned} &\mathbf{p}(\mathbf{x}(t)) = [-y(t-1), \ldots, -y(t-n_a), \\ &\mathcal{N}_1^{(k)}(t-1), \ldots, \mathcal{N}_d^{(k)}(t-1), \ldots \mathcal{N}_1^{(k)}(t-n_b), \ldots, \mathcal{N}_d^{(k)}(t-n_b)]^T \end{aligned} \tag{14}$$

(13) can be rewritten in the matrix form as

$$\mathbf{y} = \mathbf{P}\boldsymbol{\vartheta} + \boldsymbol{\Xi} \tag{15}$$

where $\mathbf{y} = [y(1), \ldots, y(N)]^T$ is the output vector. $\boldsymbol{\Xi} = [\xi(1), \ldots, \xi(N)]^T$, and $\mathbf{P}$ is the regression matrix

$$\mathbf{P}=\begin{bmatrix} p_1(\mathbf{x}(1)) & p_2(\mathbf{x}(1)) & \cdots & p_{n_a+d\cdot n_b}(\mathbf{x}(1)) \\ p_1(\mathbf{x}(2)) & p_2(\mathbf{x}(2)) & \cdots & p_{n_a+d\cdot n_b}(\mathbf{x}(2)) \\ \cdots & \cdots & \cdots & \cdots \\ p_1(\mathbf{x}(N)) & p_2(\mathbf{x}(N)) & \cdots & p_{n_a+d\cdot n_b}(\mathbf{x}(N)) \end{bmatrix} \tag{16}$$

The parameter vector $\boldsymbol{\vartheta}$ can be found as the least squares solution of

$$\boldsymbol{\vartheta}_{LS}=\mathbf{B}^{-1}\mathbf{P}^T\mathbf{y} \tag{17}$$

provided that $\mathbf{B}=\mathbf{P}^T\mathbf{P}$ is of full rank. Alternatively if this condition is violated, i.e. $Rank(\mathbf{B})=r<n_a+d\cdot n_b$, then performing the eigenvalue decomposition $\mathbf{BQ}=\mathbf{Q}\Sigma$, where $\Sigma=\mathrm{diag}[\sigma_1,\ldots,\sigma_r,0,\ldots,0]$ with $\sigma_1>\sigma_2>\cdots>\sigma_r>0$. $\mathbf{Q}=[\mathbf{q}_1,\ldots,\mathbf{q}_{n_a+d\cdot n_b}]$, followed by truncating the eigenvectors corresponding to zero eigenvalues, we have

$$\boldsymbol{\vartheta}_{LS}^{svd}=\sum_{i=1}^{r}\frac{\mathbf{y}^T\mathbf{P}\mathbf{q}_i}{\sigma_i}\mathbf{q}_i \tag{18}$$

Thus the mean square error can be readily computed from

$$J(\boldsymbol{\lambda})=[\mathbf{y}-\mathbf{P}\boldsymbol{\vartheta}_{LS}^{svd}]^T[\mathbf{y}-\mathbf{P}\boldsymbol{\vartheta}_{LS}^{svd}]/N \tag{19}$$

for any specified $\boldsymbol{\lambda}$. Notice that it is computationally simple to evaluate $J(\boldsymbol{\lambda})$ due to the fact that the model has a linear in the parameter structure for a given $\boldsymbol{\lambda}$. This is an important observation for simplifying the algorithm design. This suggests that we can optimize $\boldsymbol{\lambda}$ as the first task. The information of other models parameters are implicit in $\boldsymbol{\vartheta}_{LS}^{svd}$ and dependent on $\boldsymbol{\lambda}$. We point out that at this stage other models parameters are not estimated which would be much more computationally involved but unnecessary.

4.2. Particle swarm optimization for estimating the shaping parameters λ_j's

In the following we propose to apply the PSO algorithm [32,33], and aim to solve

$$\boldsymbol{\lambda}_{\mathrm{opt}}=\arg\min_{\boldsymbol{\lambda}\in\prod_{j=1}^{d}\Lambda_j} J(\boldsymbol{\lambda}) \quad \text{s.t. } \boldsymbol{\lambda}^T\mathbf{1}=1 \tag{20}$$

where $\mathbf{1}$ denotes a vector of all ones with appropriate dimension.

$$\prod_{j=1}^{d}\Lambda_j=\prod_{j=1}^{d}[0,1] \quad \text{s.t. } \boldsymbol{\lambda}^T\mathbf{1}=1 \tag{21}$$

defines the search space. A swarm of particles, $\{\boldsymbol{\lambda}_i^{(l)}\}_{i=1}^{S}$, that represent potential solutions are "flying" in the search space $\prod_{j=1}^{d}\Lambda_j$, where S is the swarm size and index l denotes the iteration step. The algorithm is summarized as follows.

(a) *Swarm initialisation.* Set the iteration index $l=0$ and randomly generate $\{\boldsymbol{\lambda}_i^{(l)}\}_{i=1}^{S}$ in the search space $\prod_{j=1}^{d}\Lambda_j$. These are obtained by randomly set each element of $\{\boldsymbol{\lambda}_i^{(l)}\}_{i=1}^{S}$ as *rand*() (denoting the uniform random number between 0 and 1), followed normalizing them by

$$\boldsymbol{\lambda}_i^{(0)}=\boldsymbol{\lambda}_i^{(0)}\Big/\sum_{j=1}^{d}\boldsymbol{\lambda}_i^{(0)}\big|_j \tag{22}$$

where $\bullet|_j$ denotes the jth element of $\bullet$, so that $\{\boldsymbol{\lambda}_i^{(0)}\}^T\mathbf{1}=1$ is valid.

(b) *Swarm evaluation.* The cost of each particle $\boldsymbol{\lambda}_i^{(l)}$ is obtained as $J(\boldsymbol{\lambda}_i^{(l)})$. Each particle $\boldsymbol{\lambda}_i^{(l)}$ remembers its best position visited so far, denoted as $\mathbf{pb}_i^{(l)}$, which provides the cognitive information. Every particle also knows the best position visited so far among the entire swarm, denoted as $\mathbf{gb}^{(l)}$, which provides the social information. The cognitive information $\{\mathbf{pb}_i^{(l)}\}_{i=1}^{S}$ and the social information $\mathbf{gb}^{(l)}$ are updated at each iteration

For ($i=1$; $i\le S$; i++)
  If ($J(\boldsymbol{\lambda}_i^{(l)})<J(\mathbf{pb}_i^{(l)})$) $\mathbf{pb}_i^{(l)}=\boldsymbol{\lambda}_i^{(l)}$;
End for;
$i^*=\arg\min_{1\le i\le S} J(\mathbf{pb}_i^{(l)})$;
If ($J(\mathbf{pb}_{i^*}^{(l)})<J(\mathbf{gb}^{(l)})$) $\mathbf{gb}^{(l)}=\mathbf{pb}_{i^*}^{(l)}$;

(c) *Swarm update.* Each particle $\boldsymbol{\lambda}_i^{(l)}$ has a velocity, denoted as $\boldsymbol{\gamma}_i^{(l)}$, to direct its "flying". The velocity and position of the ith particle are updated in each iteration according to

$$\boldsymbol{\gamma}_i^{(l+1)}=\mu_0*\boldsymbol{\gamma}_i^{(l)}+rand()*\mu_1*(\mathbf{pb}_i^{(l)}-\boldsymbol{\lambda}_i^{(l)})+rand()*\mu_2*(\mathbf{gb}^{(l)}-\boldsymbol{\lambda}_i^{(l)}) \tag{23}$$

$$\boldsymbol{\lambda}_i^{(l+1)}=\boldsymbol{\lambda}_i^{(l)}+\boldsymbol{\gamma}_i^{(l+1)} \tag{24}$$

where μ_0 is the inertia weight, μ_1 and μ_2 are the two acceleration coefficients. In order to avoid excessive roaming of particles beyond the search space [37], a velocity space

$$\prod_{j=2}^{d}\Upsilon_j=\prod_{j=2}^{d}[-\Upsilon_{j,\max},\Upsilon_{j,\max}] \tag{25}$$

is imposed on $\boldsymbol{\gamma}_i^{(l+1)}$ so that

If ($\boldsymbol{\gamma}_i^{(l+1)}|_j>\Upsilon_{j,\max}$) $\boldsymbol{\gamma}_i^{(l+1)}|_j=\Upsilon_{j,\max}$;
If ($\boldsymbol{\gamma}_i^{(l+1)}|_j<-\Upsilon_{j,\max}$) $\boldsymbol{\gamma}_i^{(l+1)}|_j=-\Upsilon_{j,\max}$.

Moreover, if the velocity as given in Eq. (23) approaches zero, it is reinitialised proportional to $\Upsilon_{j,\max}$ with a small factor v

$$\text{If } (\boldsymbol{\gamma}_i^{(l+1)}|_j==0)\boldsymbol{\gamma}_i^{(l+1)}|_j=\pm rand()*v*\Upsilon_{j,\max} \tag{26}$$

In order to ensure each element of $\boldsymbol{\lambda}_i^{(l+1)}$ that it satisfies the constraint and stays in the space, we modified constraint check in the PSO as follows:

If ($\boldsymbol{\lambda}_i^{(l+1)}|_j<0$) $\boldsymbol{\lambda}_i^{(l+1)}|_j=0$;

then

$$\boldsymbol{\lambda}_i^{(l+1)}=\boldsymbol{\lambda}_i^{(l+1)}\Big/\sum_{j=1}^{d}\boldsymbol{\lambda}_i^{(l+1)}|_j \tag{27}$$

Note that the normalization step that we introduced here does not affect the cost function value, rather it effectively keeps the solution stay inside the bound.

(d) *Termination condition check.* If the maximum number of iterations, $I_{\max}$, is reached, terminate the algorithm with the solution $\mathbf{gb}^{(I_{\max})}$; otherwise, set $l=l+1$ and go to Step (b).

Ratnaweera et al. [35] reported that using a time varying acceleration coefficient (TVAC) enhances the performance of PSO. We adopt this mechanism, in which μ_1 is reduced from 2.5 to 0.5 and μ_2 varies from 0.5 to 2.5 during the iterative procedure

$$\mu_1=(0.5-2.5)*l/I_{\max}+2.5$$

$$\mu_2=(2.5-0.5)*l/I_{\max}+0.5 \tag{28}$$

The reason for good performance of this TVAC mechanism can be explained as follows. At the initial stages, a large cognitive component and a small social component help particles to

wander around or better exploit the search space, avoiding local minima. In the later stages, a small cognitive component and a large social component help particles to converge quickly to a global minimum. We use $\mu_0 = rand()$ at each iteration.

The search space as given in Eq. (21) is defined by the specific problem to be solved, and the velocity limit $\Upsilon_{j,\max}$ is empirically set. An appropriate value of the small control factor v in Eq. (26) for avoiding zero velocity is empirically found to be $v = 0.1$ for our application.

4.3. Estimating the parameter vectors $\boldsymbol{\omega}$,$\mathbf{a}$,$\mathbf{b}$ using ϑ_{LS}^{svd}

In this section we describe the second stage of Bai's two stage identification algorithm [3] which can be used to recover $\boldsymbol{\omega}$,$\mathbf{a}$,$\mathbf{b}$ from $\vartheta_{LS}^{svd}(\boldsymbol{\lambda}_{\text{opt}})$ based on the result of PSO above. Our final estimate of $\hat{\mathbf{a}}$, which is simply taken as the subvector of the resultant $\vartheta_{LS}^{svd}(\boldsymbol{\lambda}_{\text{opt}})$, consisting of its first n_a elements.

Rearrange the (n_a+1)th to $(n_a+(d+1)\times n_b)$th elements of $\vartheta_{LS}^{svd}(\boldsymbol{\lambda}_{\text{opt}})$ into a matrix

$$\mathbf{M} = \begin{bmatrix} b_1\omega_0 & b_1\omega_1 & \cdots & b_1\omega_d \\ b_2\omega_0 & b_2\omega_1 & \cdots & b_2\omega_d \\ \cdots & \cdots & \cdots & \cdots \\ b_{n_b}\omega_0 & b_{n_b}\omega_1 & \cdots & b_{n_b}\omega_d \end{bmatrix} = \mathbf{b}\boldsymbol{\omega}^T \in \Re^{n_b\times(d+1)} \tag{29}$$

The matrix $\mathbf{M}$ has rank 1 and its singular value decomposition is of the form

$$\mathbf{M} = \boldsymbol{\Gamma}\begin{bmatrix} \delta_{\mathbf{M}} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}\boldsymbol{\Delta}^T = \boldsymbol{\Gamma}\begin{bmatrix} \delta_{\mathbf{M}} \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}\begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix}\boldsymbol{\Delta}^T \tag{30}$$

where $\boldsymbol{\Gamma} = [\boldsymbol{\Gamma}_1,\ldots,\boldsymbol{\Gamma}_{n_b}] \in \Re^{n_b\times n_b}$ and $\boldsymbol{\Delta} = [\boldsymbol{\Delta}_1,\ldots,\boldsymbol{\Delta}_{d+1}] \in \Re^{(d+1)\times(d+1)}$, where $\boldsymbol{\Gamma}_i$ $(i=1,\ldots,n_b)$ and $\boldsymbol{\Lambda}_i$ $(i=1,\ldots,(d+1))$ are orthonormal vectors. $\delta_{\mathbf{M}}$ is the sole non-zero singular value of $\mathbf{M}$. $\mathbf{b}$ and $\boldsymbol{\omega}$ can be obtained using

$$\hat{\mathbf{b}} = \delta_{\mathbf{M}}\Gamma_1$$

$$\hat{\boldsymbol{\omega}} = \boldsymbol{\Delta}_1 \tag{31}$$

followed by

$$\hat{\mathbf{b}} \leftarrow \beta\hat{\mathbf{b}}$$

$$\hat{\boldsymbol{\omega}} \leftarrow \hat{\boldsymbol{\omega}}/\beta \tag{32}$$

where $\beta = (1+\sum_{j=1}^{n_a}\hat{a}_j)/(\sum_{j=1}^{n_b}\hat{b}_j)$.

Note that the standard Bai's approach as above may suffer a serious numerical problem that the matrix $\mathbf{M}$ turns out to have rank higher than one, resulting in the parameters estimator far from usable. This issue was discussed in [42], in which the modified SVD approach was proposed to address the problem. The more stable modified SVD approach [42] is used in our simulations.

4.4. A summary of the complete system identification algorithm

For completeness, the system identification algorithm is summarized below.

(1) Based on the training data set and any prior knowledge of the system, predetermine the number of basis functions d, the polynomial order k and the input range $[U_{min}, U_{max}]$. Predetermine a set of $(d+k)$ knots within the range according to (5).
(2) Apply the PSO to determine $\hat{\boldsymbol{\lambda}}$ as the optimal shaping parameter vector $\boldsymbol{\lambda}_{\text{opt}}$ according to Section 4.2.
(3) Using the shaping parameters as specified by $\boldsymbol{\lambda}_{\text{opt}}$, find $\vartheta_{LS}^{svd}(\boldsymbol{\lambda}_{\text{opt}})$ based on (18). Subsequently apply the method described in Section 4.3 to find the parameter vector $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$ and $\hat{\boldsymbol{\omega}}$ from $\vartheta_{LS}^{svd}(\boldsymbol{\lambda}_{\text{opt}})$.
(4) Based on $\hat{\boldsymbol{\omega}}$ and $\hat{\boldsymbol{\lambda}}$, the underlying function $\Psi(\bullet)$ for any point within the range $[U_{min},U_{max}]$ can be recovered by applying the De Boor algorithm using (6)–(9).

5. An illustrative example

The Hammerstein system is a suitable model for signal processing applications involving any nonlinear distortion followed by a linear filter, e.g. the modelling of hydraulic actuator friction dynamics [15], liquid level control system for a non-constant cross-sectional area tank [43]. A Hammerstein system is simulated, in which the linear subsystem is $A(q^{-1}) = 1-1.2q^{-1}+0.9q^{-2}$, $B(q^{-1}) = 1.7q^{-1}-q^{-2}$, and the nonlinear subsystem $\Psi(u) = 2\,\text{sign}(u)\sqrt{|u|}$. The variances of the additive noise to the system output is set as 0.01 (low noise) and 0.25 (high noise) respectively. About 1000 training data samples $y(t)$ were generated by using (1) and (2), where $u(t)$ was uniformly distributed random variable $u(t) \in [-1.5,1.5]$. The signal to noise ratio are calculated as 36 dB and 22 dB respectively. The polynomial degree of B-spline basis functions was set as $k=2$ (piecewise quadratic). The knots sequence U_j is set as

$$[-3.2,\ -2.4,\ -1.6,\ -0.8,\ -0.05,\ 0,\ 0.05,\ 0.8,\ 1.6,\ 2.4,\ 3.2]$$

Initially the system identification was carried out as outlined in Section 4.4. In the modified PSO algorithm, we set $S=20$, $I_{\max}=20$, $\Upsilon_{j,\max}=0.025$. The resultant eight NURB basis functions for the two data sets are plotted in Fig. 2. The modelling results are shown in Table 1, for the linear subsystem. Fig. 3 demonstrates for the nonlinear subsystem obtained with $\sigma^2=0.01$ data set. (The plot obtained with $\sigma^2=0.25$ data set has the same appearance except for the external knots sequences.)

The simulations of the pole assignment controller (see Appendix A) was experimented based on a given polynomial $T(q^{-1}) = 1-0.6q^{-1}+0.1q^{-2}$. Under the assumption that the proposed inverse of De Boor algorithm can cancel the nonlinearity in the system which is modelled by the identified NURB model as shown in Fig. 3, and by using parameter estimates given in Table 1, the required controller polynomials are estimated, e.g. for the data set from noise sequence variance at 0.01,

$$F(q^{-1}) = 1-0.4873q^{-1}$$

and

$$G(q^{-1}) = 0.6369-0.4354q^{-1}$$

and we predetermine

$$H(q^{-1}) = 0.5308+0.1834q^{-1}$$

The learning rate was preset as $\eta=0.1$. The maximum value of iteration number m was predetermined as 100. The reference signals $r(t)$ are generated as a series of sinusoidal wave with its magnitude and frequency changing every 200 time steps. Fig. 4(a) and (b) plots the resultant control signal and system response to the reference signal, respectively, when the output noise variance is set at 0.01. It can be concluded that the proposed method has excellent results in terms of system identification as well as the subsequent control for the identified systems.

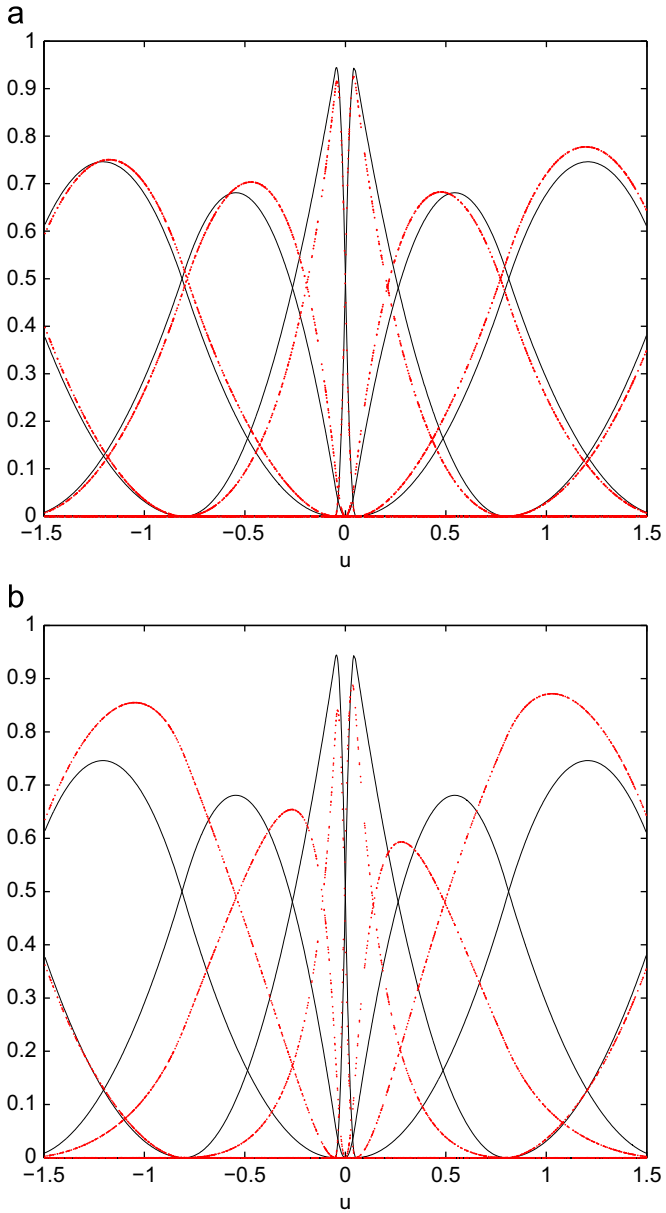

Fig. 2. The resultant B-spline (solid line) and NURB (dotted line) basis functions formed using PSO; (a) $\sigma^2 = 0.01$ and $\sigma^2 = 0.25$.

Table 1
Results of linear subsystem parameter estimation for two systems.

Parameters	a_1	a_2	b_1	b_2
True parameter	−1.2	0.9	1.7	−1
Estimate parameters ($\sigma^2 = 0.01$)	−1.2004	0.9004	1.7077	−1.0076
Estimate parameters ($\sigma^2 = 0.25$)	−1.2015	0.9027	1.7424	−1.0412

6. Conclusions

This article has introduced a new system identification algorithm for the Hammerstein systems based on observational input/output data, using the non-uniform rational B-spline (NURB) neural network. The main contribution is to propose the PSO for the estimation of the shaping parameters in NURB neural networks. For completeness, a model based controller consists of computing the inverse of the nonlinear static function approximated by NURB neural network, followed by a linear pole assignment controller is included. An illustrative example is utilized to demonstrate the efficacy of the proposed approaches.

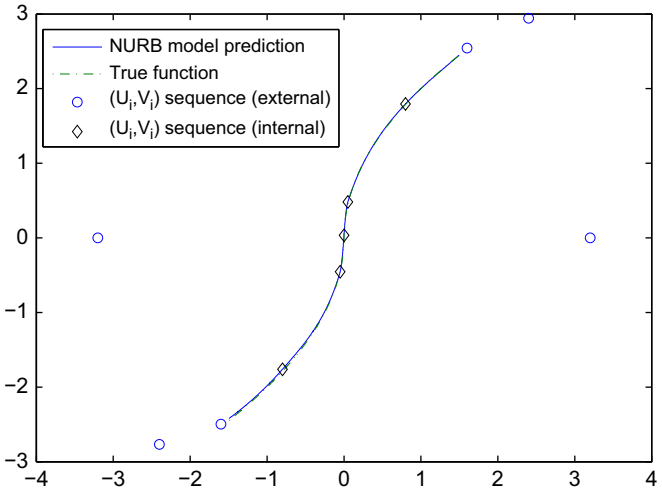

Fig. 3. The modelling result for the nonlinear function $\Psi(u)$ ($\sigma^2 = 0.01$).

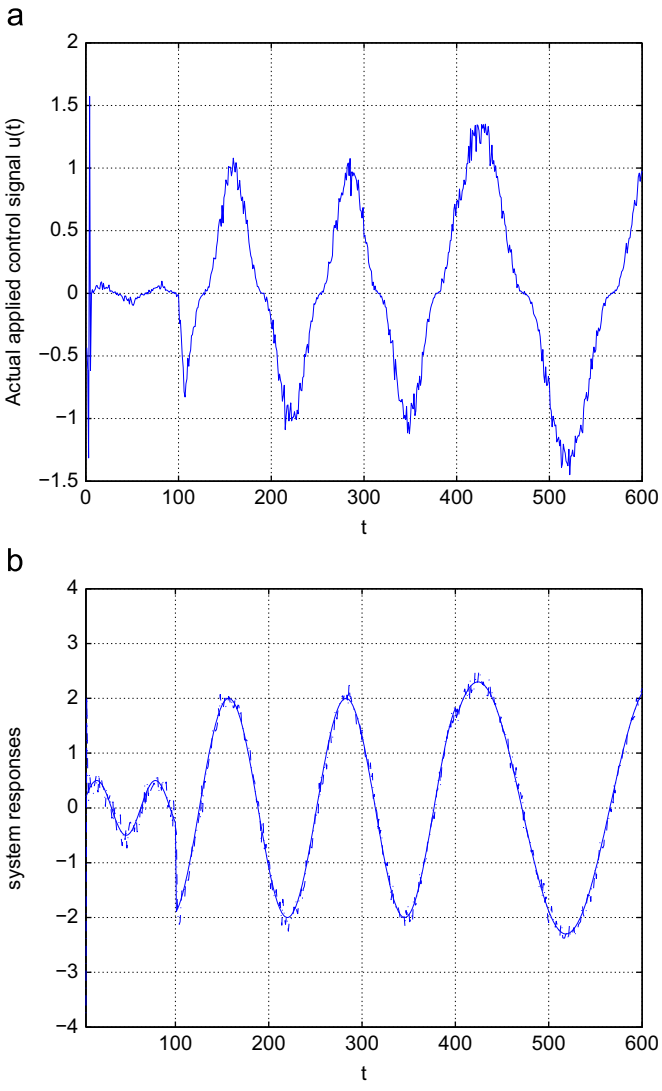

Fig. 4. The results of the pole assignment controller.

Appendix A. The pole assignment controller

The pole assignment controller used in the numerical examples is as shown in Fig. 5, in which

$$F(q^{-1}) = 1 + f_1 q^{-1} + \cdots + f_{n_f} q^{-n_f} \tag{33}$$

$$G(q^{-1}) = g_0 + g_1 q^{-1} + \cdots + g_{n_g} q^{-n_g} \tag{34}$$

$$H(q^{-1}) = h_0 + h_1 q^{-1} + \cdots + h_{n_h} q^{-n_h} \tag{35}$$

where n_f, n_g and n_h are lags in the controller to be determined. Here the problem under study is the control of the Hammerstein system, of which the nonlinear subsystem is modelled as a NURB curve and identified from input/output data. The proposed controller is the pole assignment design scheme for F, G, H [44,45], followed by $\hat{\Psi}^{-1}$, which is calculated using the inverse of the De Boor algorithm as described in Appendix B. We assume that the modelling of $\hat{\Psi}^{-1}$ using the inverse of De Boor algorithm as described in Appendix B can cancel the actual nonlinearity in Hammerstein system. Hence the closed loop description of the system is

$$\underbrace{[AF+BG]}_{\text{closed loop denominator}} \qquad y(t) = BHr(t) \tag{36}$$

where $r(t)$ is a reference signal for the system output $y(t)$ to follow. The dynamics of the closed loop are specified by a stable polynomial

$$AF + BG = T(q^{-1}) = 1 + t_1 q^{-1} + \cdots + t_n q^{-n} \tag{37}$$

The coefficients of polynomials F, G can be solved by setting $n_f = n_b + 1$, $n_g = n_a - 1$, $n \le n_a + n_b + 1$. H can be predetermined as desired subject to

$$\lim_{q \to 1} \frac{B(q^{-1})H(q^{-1})}{T(q^{-1})} = 1 \tag{38}$$

From Fig. 5, it is clear that the actual control input $u(t)$ applied to the Hammerstein system is given by

$$u(t) = \hat{\Psi}^{-1}(\hat{v}(t)) = \hat{\Psi}^{-1}\left(\frac{Hr(t) - Gy(t)}{F}\right) \tag{39}$$

Rewriting (39) in a recursive form yields the following control law:

1. $\hat{v}(t) = \sum_{j=0}^{n_h} h_j r(t-j) - \sum_{j=0}^{n_g} g_j y(t-j) - \sum_{j=1}^{n_f} f_j \hat{v}(t-j)$
2. Find $u(t) = \hat{\Psi}^{-1}(\hat{v}(t))$ using the inverse of De Boor algorithm (40)

Note that in practice if $\hat{v}(t)$ is out of the region between $\hat{\Psi}(U_{min})$ and $\hat{\Psi}(U_{max})$, $\hat{v}(t)$ is reset as 0 to avoid this to happen at the next time step.

Appendix B. The inverse of De Boor algorithm

Using estimated weights $\hat{\omega}_j$ and shaping parameters $\hat{\lambda}_j$, the output of the nonlinear subsystem is represented by

$$v = \hat{\Psi}(u) = \frac{\sum_{j=1}^{d} \hat{\omega}_j \hat{\lambda}_j \mathcal{B}_j^{(k)}(u)}{\sum_{j=1}^{d} \hat{\lambda}_j \mathcal{B}_j^{(k)}(u)} \tag{41}$$

The inverse of De Boor algorithm [26] solves the problem of finding its inverse, $u = \hat{\Psi}^{-1}(v)$, given that v lies in the region between two points, $\hat{\Psi}(U_{min})$ and $\hat{\Psi}(U_{max})$. Initially a sequence in the domain of v is generated as

$$V_i = \frac{\sum_{j=1}^{d} \hat{\omega}_j \hat{\lambda}_j \mathcal{B}_j^{(k)}(U_i)}{\sum_{j=1}^{d} \hat{\lambda}_j \mathcal{B}_j^{(k)}(u)}, \quad i = 1, 2, \ldots, (d+k) \tag{42}$$

Note that $v = \hat{\Psi}(u)$ is an one-to-one mapping, and this means that the resultant sequence due to the internal knots $[V_k, \ldots, V_d]$ is either increasing or decreasing.

B.1. The algorithm

(1) Given v, and the sequence $\{V_i\}$, initially find

$$l = \arg\{(v - V_i)(v - V_{i+1}) < 0, i = k, k+1, \ldots, (d-1)\} \tag{43}$$

(2) Initialise $u^{(0)}$ as a random number with $U_l < u^{(0)} < U_{l+1}$.

(3) The $(m+1)$th step is given by

$$u^{(m+1)} = u^{(m)} + \Delta u^{(m)} = u^{(m)} + \eta \cdot \text{sign}\left[\frac{V_d - V_k}{U_d - U_k}\right](v - \hat{\Psi}(u^{(m)})) \tag{44}$$

where

$$\text{sign}(s) = \begin{cases} 1 & \text{if } s \ge 0 \\ -1 & \text{if } s < 0 \end{cases} \tag{45}$$

$0 < \eta \ll 1$ is the learning rate, that is preset empirically. $\hat{\Psi}(u^{(m)})$ is calculated using De Boor algorithm ((6)–(9) and (41)).

(4) Set $m = m+1$, repeat Steps 3 and 4, until $|\Delta u^{(m)}|/(U_d - U_k) < \varepsilon$, where $\varepsilon > 0$ is a predetermined small number in order to achieve the required precision, e.g. $\varepsilon = 10^{-3}$. Or the iteration

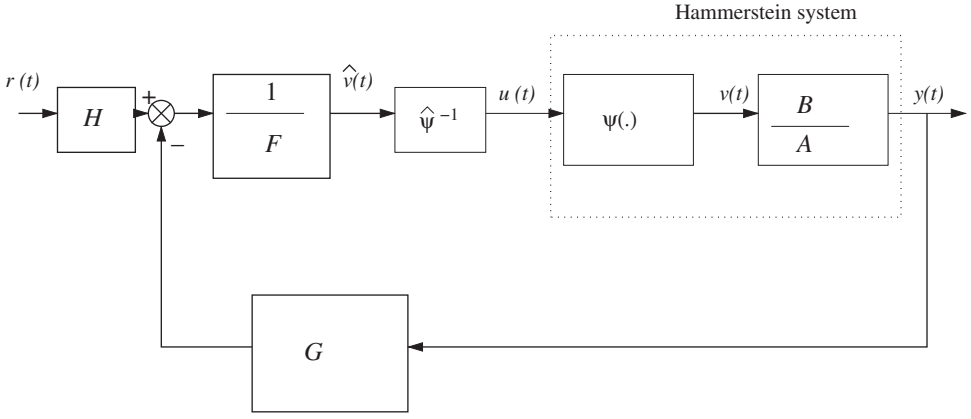

Fig. 5. The control of Hammerstein system using pole assignment and the inverse of De Boor algorithm.

can be terminated when m reaches a predetermined maximum value.

The inverse of De Boor algorithm was introduced for the control of B-spline based Hammerstein systems [26], in which the convergence was analyzed as Theorem 1 in [26]. It is easy to verify that the same procedure and convergence analysis is applicable for NURB approximation based Hammerstein systems (with $\{V_i\}$, $\hat{\Psi}(u)$ evaluation by NURB basis functions as here rather than nonrational B-spline function as in [26]).

References

[1] S.A. Billings, S.Y. Fakhouri, Nonlinear system identification using the Hammerstein model, Int. J. Syst. Sci. 10 (1979) 567–578.
[2] P. Stoica, T. Söderström, Instrumental variable methods for identification of Hammerstein systems, Int. J. Control 35 (1982) 459–476.
[3] E.W. Bai, M.Y. Fu, A blind approach to Hammerstein model identification, IEEE Trans. Signal Process. 50 (7) (2002) 1610–1619.
[4] W. Greblicki, M. Pawlak, Identification of discrete Hammerstein systems using kernel regression estimate, IEEE Trans. Autom. Control AC-31 (1) (1986) 74–77.
[5] W. Greblicki, Nonparametric orthogonal series identification of Hammerstein systems, Int. J. Syst. Sci. 20 (1989) 2355–2367.
[6] W. Greblicki, Stochastic approximation in nonparametric identification of Hammerstein systems, IEEE Trans. Autom. Control 47 (11) (2002) 1800–1810.
[7] H.F. Chen, Pathwise convergence of recursive identification algorithms for Hammerstein systems, IEEE Trans. Autom. Control 49 (10) (2004) 1873–1896.
[8] M. Verhaegen, D. Westwick, Identifying MIMO Hammerstein systems in the context of subspace model identification, Int. J. Control 63 (2) (1996) 331–349.
[9] F.Z. Chaoui, F. Giri, Y. Rochdi, M. Haloua, A. Naitali, System identification based Hammerstein model, Int. J. Control 78 (6) (2005) 430–442.
[10] I.W. Hunter, M.J. Korenberg, The identification of nonlinear biological systems: Wiener and Hammerstein cascade models, Biol. Cybern. 55 (2–3) (1986) 135–144.
[11] H.H.J. Bloemen, T.J.V.D. Boom, H.B. Verbruggen, Model-based predictive control for Hammerstein–Wiener systems, Int. J. Control 74 (5) (2001) 295–482.
[12] J. Turunen, J.T. Tanttu, P. Loula, Hammerstein model for speech coding, EURASIP J. Appl. Signal Process. 12 (2003) 1238–1249.
[13] A. Balestrino, A. Landi, M. Ould-Zmirli, L. Sani, Automatic nonlinear autotuning method for Hammerstein modelling of electrical drives, IEEE Trans. Ind. Electron. 48 (3) (2001) 645–655.
[14] S.W. Su, Identification and control for heart rate regulation during treadmill exercise, IEEE Trans. Biomed. Eng. 54 (7) (2007) 1238–1246.
[15] B. Kwak, A.E. Yagle, J.A. Levitt, Nonlinear system identification of hydraulic actuator friction dynamics using a Hammerstein model, in: Proceedings of the IEEE ASSP'98, Seattle, WA, 1998, pp. 1933–1936.
[16] Z.Q. Lang, A nonparametric polynomial identification algorithm for the Hammerstein system, IEEE Trans. Autom. Control 42 (October) (1997) 1435–1441.
[17] E.W. Bai, An optimal two-stage identification algorithm for Hammerstein–Wiener nonlinear systems, Automatica 34 (1998) 333–338.
[18] R.T. Farouki, T.N.T. Goodman, On the optimal stability of the Bernstein basis, Math. Comput. 65 (216) (1996) 1553–1566.
[19] G. Farin, Curves and Surfaces for Computer-aided Geometric Design: A Practical Guide, Academic Press, Boston, 1994.
[20] de Boor, A Practical Guide to Splines, Spring Verlag, New York, 1978.
[21] X. Hong, C.J. Harris, Generalised neurofuzzy network modelling algorithms using Bezier Bernstein polynomial functions and additive decomposition, IEEE Trans. Neural Netw. 11 (2000) 889–902.
[22] X. Hong, R.J. Mitchell, A Hammerstein model identification algorithm using Bezier–Bernstein approximation, IET Proc. Control Theory Appl. 1 (4) (2007) 1149–1159.
[23] T. Kavli, ASMOD—an algorithm for adaptive spline modelling of observation data, Int. J. Control 58 (4) (1993) 947–967.
[24] M. Brown, C.J. Harris, Neurofuzzy Adaptive Modelling and Control, Prentice Hall, Hemel Hempstead, 1994.
[25] C.J. Harris, X. Hong, Q. Gan, Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach, Springer-Verlag, 2002.
[26] X. Hong, R.J. Mitchell, S. Chen, Modeling and control of Hammerstein system using B-spline approximation and the inverse of De Boor algorithm, Int. J. Syst. Sci., doi:10.1080/00207721.2011.564320, in press.
[27] S.A. Billings, S. Chen, Identification of non-linear rational systems using a prediction-error estimation algorithm, Int. J. Syst. Sci. 20 (3) (1989) 467–494.
[28] S.A. Billings, Q.M. Zhu, Structure detection algorithm for nonlinear rational models, Int. J. Control 59 (6) (1994) 1439–1463.
[29] K.Z. Mao, S.A. Billings, Q.M. Zhu, A regularised rational model estimation algorithm, Int. J. Syst. Sci. 30 (5) (1999) 455–465.
[30] Q.M. Zhu, A back propagation algorithm to estimate the parameters of nonlinear dynamic rational models, Appl. Math. Modelling 27 (2003) 169–187.
[31] Q.M. Zhu, An implicit least squares algorithm for nonlinear rational model parameter estimation, Appl. Math. Modelling 29 (2005) 673–689.
[32] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of 1995 IEEE International Conference on Neural Networks, vol. 4, Perth, Australia, November 27–December 1, 1995, pp. 1942–1948.
[33] J. Kennedy, R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann, 2001.
[34] D.W. van der Merwe, A.P. Engelbrecht, Data clustering using particle swarm optimization, in: Proceedings of the CEC 2003, Cabberra, Australia, December 8–12, 2003, pp. 215–220.
[35] A. Ratnaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans. Evol. Comput. 8 (June) (2004) 240–255.
[36] M.G.H. Omran, Particle Swarm Optimization Methods for Pattern Recognition and Image Processing, Ph.D. Dissertation, University of Pretoria, Pretoria, South Africa, 2005.
[37] S.M. Guru, S.K. Halgamuge, S. Fernando, Particle swarm optimisers for cluster formation in wireless sensor networks, in: Proceedings of 2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing, Melbourne, Australia, December 5–8, 2005, pp. 319–324.
[38] K.K. Soo, Y.M. Siu, W.S. Chan, L. Yang, R.S. Chen, Particle-swarm-optimization-based multiuser detector for cdma communications, IEEE Trans. Veh. Technol. 56 (September) (2007) 3006–3013.
[39] E. Fruzzetti, A. Palazoglu, K.A. Mcdonald, Nonlinear model predictive control using Hammerstein models, J. Process Control 7 (1) (1997) 31–41.
[40] H.H. Bloemen, T.J. van den Boom, H.B. Verbruggen, Model based predictive control for Hammerstein systems, in: Proceedings of the 39th IEEE Conference on Decision and Control, Sydney, Australia, 2000, pp. 4963–4968.
[41] R.S. Patwardhan, S. Lakshminarayanan, S.L. Shah, Constrained nonlinear MC using Hammerstein and Wiener model: PLS framework, AIChE J. 44 (7) (1998) 1611–1622.
[42] I. Goethals, K. Pelckmans, J.A.K. Suykens, B.D. Moor, Identification of MIMO Hammerstein models using least squares support vector machines, Automatica 41 (2005) 1263–1272.
[43] K.J. Astrom, B. Wittenmark, Adaptive Control, Addison Wesley MS, 1989.
[44] P.E. Wellstead, J.M. Edmunds, D. Pragerand, P. Zanker, Self tuning pole/zero assignment regulator, Int. J. Control 30 (1) (1979) 1–26.
[45] P.E. Wellstead, M.B. Zarrop, Self-Tuning Systems—Control and Signal Processing, John Wiley & Sons, 1991.

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

Sheng Chen received his PhD degree in control engineering from the City University, London, UK, in September 1986. He was awarded the Doctor of Sciences (DSc) degree by the University of Southampton, Southampton, UK, in 2005. From October 1986 to August 1999, he held research and academic appointments at the University of Sheffield, the University of Edinburgh and the University of Portsmouth, all in UK. Since September 1999, he has been with the School of Electronics and Computer Science, University of Southampton, UK. Professor Chen's research interests include wireless communications, adaptive signal processing for communications, machine learning, and evolutionary computation methods. He has published over 400 research papers. Dr. Chen is a Fellow of IET and a Fellow of IEEE. In the database of the world's most highly cited researchers, compiled by Institute for Scientific Information (ISI) of the USA, Dr. Chen is on the list of the highly cited researchers (2004) in the engineering category.