Modelling and control of Hammerstein system using B-spline approximation and the inverse of De Boor algorithm

X. Hong\textsuperscript{a*}, R.J. Mitchell\textsuperscript{a} and S. Chen\textsuperscript{b}

\textsuperscript{a}School of Systems Engineering, University of Reading, Reading RG6 6AY, UK; \textsuperscript{b}School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK

(Received 2 June 2010; final version received 23 January 2011)

In this article a simple and effective controller design is introduced for the Hammerstein systems that are identified based on observational input/output data. The nonlinear static function in the Hammerstein system is modelled using a B-spline neural network. The controller is composed by computing the inverse of the B-spline approximated nonlinear static function, and a linear pole assignment controller. The contribution of this article is the inverse of De Boor algorithm that computes the inverse efficiently. Mathematical analysis is provided to prove the convergence of the proposed algorithm. Numerical examples are utilised to demonstrate the efficacy of the proposed approach.

Keywords: B-spline; De Boor algorithm; Hammerstein model; pole assignment controller; system identification

1. Introduction

The Hammerstein model, comprising a nonlinear static functional transformation followed by a linear dynamical model, has been applied to nonlinear plant/process modelling in a wide range of biological/engineering problems (Hunter and Korenberg 1986; Balestrino, Landi, Ould-Zmirli and Sani 2001; Bloemen, van den Boom, and Verbruggen 2001; Turunen, Tanttu, and Loula 2003). 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 (Su 2007), and the modelling of hydraulic actuator friction dynamics (Kwak, Yagle, and Levitt 1998). The Hammerstein model has been widely researched (Billings and Fakhouri 1979; Stoica and Söderström 1982; Greblicki and Pawlak 1986; Greblicki 1989, 2002; Verhaegen and Westwick 1996; Lang 1997; Bai and Fu 2002; Chen 2004; Chaoui, Giri, Rochdi, Haloua, and Naitali 2005). The model characterisation/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 \textit{a priori} unknown nonlinearity by use of both parametric (Verhaegen and Westwick 1996; Chaoui et al. 2005) and nonparametric methods (Lang 1997; Greblicki 2002; Chen 2004). It has been shown that the Bernstein basis is the best conditioned and the most stable among all other polynomial basis (Farouki and Goodman 1996). The inverse of de Casteljau’s algorithm was introduced to identify the Bezier–Bernstein neural network using the Bernstein approximation and from observational data (Hong and Harris 2000). Recently a new identification algorithm (Hong and Mitchell 2007) for the Hammerstein model has been introduced based on the Bezier–Bernstein approximation and the inverse of de Casteljau’s algorithm. Alternatively, the special structure of Hammerstein models can be exploited to develop hybrid parameter estimation algorithms (Bai 1998; Bai and Fu 2002; Chaoui et al. 2005). Similar to Bezier curve, the B-spline curve has also been widely used in computer graphics and computer aided geometric design (CAGD; Farin 1994). B-spline 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 curve is mathematically involved and numerically unstable (De Boor 1978). De Boor algorithm uses recurrence relations and is numerically stable (De Boor 1978). The B-spline basis functions for nonlinear systems modelling have been widely applied (Kavli 1993; Brown and Harris 1994; Harris, Hong and Gan 2002).

Model based control for the Hammerstein system has been well studied (Anbumani, Patnaik, and Sarma 1981; Bloemen, van den Boom, and Verbruggen 2000;

\textsuperscript{*}Corresponding author. Email: x.hong@reading.ac.uk

ISSN 0020–7721 print/ISSN 1464–5319 online
© 2012 Taylor & Francis
http://dx.doi.org/10.1080/00207721.2011.564320
http://www.tandfonline.com
Bloemen et al. 2001). A popular treatment of handling
the Hammerstein model is to remove the nonlinearity
via an inversion (Fruzzetti, Palazoglou, and Mcdonald
1997; Patwardhan, Lakshminarayanan, and Shah
1998; Bloemen et al. 2000). This enables the celebrated
self tuning control methods to be readily applicable
(Astrom and Wittenmark 1989). The implementation
of model based control for a priori unknown
Hammerstein model requires system identification
including the modelling and identification of the
nonlinear static function. Different nonlinear model
representations result in variations in controller design
algorithms. For example, in Anbumani et al. (1981)
and Zhu, Warwick, and Douce (1991) the nonlinear
static function is based on an explicit polynomial
function of the input. The optimal control law satisfies
a polynomial equation of the input, which is then
found via root solving. In Hong and Mitchell (2006),
the closed-loop system is linearised by inserting the
inverse of the identified static nonlinearity, and
the nonlinear subsystems’ inverse is calculated using
the inverse of de Casteljau’s algorithm.

Computationally efficient and numerically stable
algorithms are in general desirable in nonlinear system
identification and control. In this article the control for
Hammerstein system as characterised by the B-spline
polynomial based model is investigated through the
removal of the nonlinearity via the inversion of the
B-spline neural network, and the application of a pole
assignment controller. Note that for the system iden-
tification of the resultant model representation, Bai’s
overparameterisation approach is directly applicable
(Bai and Fu 2002). In this article we used the Gauss–
Newton algorithm subject to constraints as proposed
in Hong and Mitchell (2007). The linearisation of the
closed-loop system is achieved by inserting the inverse
of the identified static nonlinearity via the inverse of
De Boor algorithm. In spite of being an iterative
numerical one, the proposed algorithm is shown to
have a very low computational cost. In addition, the
inverse of De Boor algorithm is numerically stable as
it builds upon De Boor algorithm (De Boor 1978).
Convergence analysis based on the Lyapunov method
is included.

2. Modelling of the Hammerstein system based
on B-spline functions

2.1. The Hammerstein system

The Hammerstein system, as shown in Figure 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.

$\psi(\bullet)$ $v(t)$ $\text{Linear}$ $y(t)$ $u(t)$ $\xi(t)$

Figure 1. The Hammerstein system.

The system can be represented by

$$y(t) = \psi(t) + \xi(t)$$
$$= -a_1 y(t-1) - a_2 y(t-2) - \ldots - a_n y(t-n_a)
+ b_1 v(t-1) + \ldots + b_m v(t-n_b) + \xi(t), \quad (1)$$
$$v(t-j) = \Psi(u(t-j)), \quad j = 1, \ldots, n_b, \quad (2)$$

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

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

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 $a_i$, $b_j$ in the linear subsystems. Note that the
signals between the two subsystems are unavailable.
Without 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.

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

In this work the B-spline basis functions are
adopted in order to model $\Psi(\bullet)$. Specifically, the
B-spline basis functions are initially formed by using
the De Boor algorithm (de Boor 1978) for the input
data sets.
2.2. Modelling of $\Psi(\bullet)$ using B-spline function approximation

Univariate B-spline basis functions are parameterised 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:

\[
\begin{align*}
U_1 &< U_2 < U_k, \\
= U_{\min} &< U_{k+1} < U_{k+2} < \cdots < U_d < U_{\max} \\
= U_{d+1} &< \cdots < U_{d+k}.
\end{align*}
\]

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

\[
\mathcal{B}_j^0(u) = \begin{cases} 
1 & \text{if } U_j \leq u < U_{j+1}, \\
0 & \text{otherwise},
\end{cases} \quad j = 1, \ldots, (d + k)
\]

\[
\mathcal{B}_j^i(u) = \frac{u - U_j}{U_{j+i} - U_j} \mathcal{B}_j^i(u) + \frac{U_{j+i+1} - u}{U_{j+i+1} - U_{j+1}} \mathcal{B}_{j+1}^{i-1}(u),
\]

\[
i = 1, \ldots, k.
\]

We model $\Psi(\bullet)$ as a B-spline neural network (Harris et al. 2002), in the form

\[
\Psi(u) = \sum_{j=1}^{d} \mathcal{B}_j^i(u) \omega_j,
\]

where $\omega_j$'s are weights to be determined. Denote $\omega = [\omega_1, \ldots, \omega_d] \in \mathbb{R}^d$. 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 $O(k^2)$.

2.3. The system identification algorithm

With the B-spline approximation, the model predicted output $\hat{y}(t)$ in (1) can be written as

\[
\hat{y}(t) = -a_1 y(t-1) - a_2 y(t-2) - \cdots - a_{n_y} y(t-n_y)
\]

\[
+ b_1 \sum_{j=1}^{d} \omega_j \mathcal{B}_j^i(t-1) + \cdots + b_{n_y} \sum_{j=1}^{d} \omega_j \mathcal{B}_j^i(t-n_y). \quad (8)
\]

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

\[
y(t) = [p(x(t))]^T \theta + \xi(t), \quad (9)
\]

where $x(t) = [-y(t-1), \ldots, -y(t-n_y), u(t-1), \ldots, u(t-n_y)]^T$ is system input vector of observables with assumed known dimension of $(n_x+n_y)$, $\theta = [\alpha^T, (b_1 \omega_1), \ldots, (b_1 \omega_d), \ldots, (b_{n_y} \omega_1), \ldots, (b_{n_y} \omega_d)]^T \in \mathbb{R}^{n_x+n_y}$,

\[
p(x(t)) = [-y(t-1), \ldots, -y(t-n_y), B_1^{(k)}(t-1), \ldots, B_d^{(k)}(t-1), \ldots, B^{(k)}_{d}(t-n_y), \ldots, B^{(k)}_{d}(t-n_y)]^T
\]

(9) can be rewritten in the matrix form as

\[
y = P \theta + \Xi, \quad (11)
\]

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

\[
P = \begin{bmatrix}
p_1(x(1)) & p_2(x(1)) & \cdots & p_{n_x+n_y}(x(1)) \\
p_1(x(2)) & p_2(x(2)) & \cdots & p_{n_x+n_y}(x(2)) \\
& \cdots & \ddots & \cdots \\
p_1(x(N)) & p_2(x(N)) & \cdots & p_{n_x+n_y}(x(N))
\end{bmatrix}
\]

(12)

The parameter vector $\theta$ can be found as the least squares solution of

\[
\theta_{LS} = P^{-1} P^T y
\]

provided that $B = P^T P$ is of full rank. Alternatively, if this condition is violated, i.e. $\text{Rank}(B) = r < n_x + d \cdot n_y$, then performing the singular value decomposition (SVD) $BQ = \Sigma Q$, where $\Sigma = \text{diag}[\sigma_1, \ldots, \sigma_r, 0, \ldots, 0]$, $Q = [q_1, \ldots, q_r, \ldots, q_{n_x+n_y}]$, followed by truncating the eigenvectors corresponding to zero eigenvalues, we have

\[
\theta_{LS} = \sum_{i=1}^{r} \frac{y^T P q_i}{\sigma_i} q_i
\]

(14)

This procedure produces our final estimate of $\hat{a}$, which is simply taken as the subvector of the resultant $\theta_{LS}^{\text{rad}}$, consisting of its first $n_y$ elements. Clearly, information on $\hat{b}$ and $\hat{\omega}$ are contained in $\theta_{LS}^{\text{rad}}$. Hence, it is straightforward to recover these based on Bai’s approach using singular value decomposition (SVD: Bai and Fu 2002). Alternatively, the parameter estimation for $\hat{b}$ and $\hat{\omega}$ can be obtained using our previous work (Hong and Mitchell 2007). This is outlined below and in the summary in Appendix A. Consider that a sequence $z(t)$ is generated, based on the derived parameter estimates $\hat{a}$, as an auxiliary model.
output sequence, given by
\[ z(t) = y(t) + \hat{a}_1 y(t-1) + \hat{a}_2 y(t-2) + \cdots + \hat{a}_n y(t-n). \]  
(15)

\( z(t) \) can be interpreted as an intermediate model residual after the model output \( y(t) \) has been explained by the \( n \) regressors of \( y(t-1), \ldots, y(t-n) \). Then consider approximating \( z(t) \) using the following model
\[
\hat{z}(t) = b_1 \sum_{j=1}^{d} \omega_j B_j^{(k)}(t-1) + \cdots + b_m \sum_{j=1}^{d} \omega_j B_j^{(k)}(t-n),
\]
\[ = g(x(t), b, \omega). \]  
(16)

By setting the objective function as
\[ J_g(b, \omega) = \frac{1}{N} \sum_{t=1}^{N} [z(t) - g(x(t), b, \omega)]^2, \]
subject to
\[ \sum_{j=1}^{n} \hat{b}_j = 1, \]
the Gauss–Newton algorithm subject to constraints as proposed in Hong and Mitchell (2007) is used in this work. For completeness, see Appendix A.

3. The controller design combining the inverse of De Boor algorithm and pole assignment

3.1. The proposed inverse of De Boor algorithm

Using parameter estimates \( \hat{\omega}_j \) (Appendix A), the output of the nonlinear subsystem is represented by
\[ v = \hat{\Psi}(u) = \sum_{j=1}^{d} B_j^{(k)}(u) \hat{\omega}_j. \]  
(17)

In this section we consider the problem of finding its inverse, \( u = \hat{\Psi}^{-1}(v) \), given that \( v \) lies in the region between two points, \( \hat{\Psi}(U_{\text{min}}) \) and \( \hat{\Psi}(U_{\text{max}}) \), i.e. finding the root of the polynomial equation \( v = \hat{\Psi}(u) = \sum_{j=1}^{d} B_j^{(k)}(u) \hat{\omega}_j \). In order to solve this problem, the inverse of De Boor algorithm is introduced below.

Initially, a sequence in the domain of \( v \) is generated as
\[ V_i = \sum_{j=1}^{d} B_j^{(k)}(U_i) \hat{\omega}_j, \quad i = 1, 2, \ldots, (d+k). \]  
(18)

Note that \( v = \hat{\Psi}(u) \) is a one-to-one mapping, and this means that the resultant sequence due to the internal knots \( V_1, \ldots, V_d \) is either increasing or decreasing.

The algorithm is as follows.

1. Given \( v \), and the sequence \( \{V_i\} \), initially find
\[ l = \text{arg}(v - V_i)(v - V_{i+1}) < 0, \]
\[ i = k, k + 1, \ldots, (d - 1). \]  
(19)

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

3. The \( (m+1) \)th step is given by
\[
\hat{u}^{(m+1)} = \hat{u}^{(m)} + \Delta \hat{u}^{(m)}
\]
\[ = \hat{u}^{(m)} + \eta \cdot \text{sign} \left[ \frac{V_d - V_k}{U_d - U_k} \right] (v - \hat{\Psi}(u^{(m)})), \]  
(20)

where
\[ \text{sign}(s) = \begin{cases} 1 & \text{if } s \geq 0, \\ -1 & \text{if } s < 0, \end{cases} \]  
(21)

\( 0 < \eta < 1 \) is the learning rate, that is preset empirically. \( \hat{\Psi}(u^{(m)}) \) is calculated using De Boor algorithm ((5) and (6) and (17)).

4. Set \( m = m + 1 \), repeat Step 3 and 4, until \( \frac{\Delta \hat{u}^{(m)}}{U_l - U_{l+1}} < \varepsilon \), where \( \varepsilon > 0 \) is a predetermined small number in order to achieve the required precision, e.g. \( \varepsilon = 10^{-3} \). Or the iteration can be terminated when \( m \) reaches a predetermined maximum value.

Theorem 1: The sequence \( \{u^{(0)}, \ldots, u^{(m)}, \ldots\} \) converges to the unique solution of \( u = \hat{\Psi}^{-1}(v) \), as \( m \to \infty \), if \( \eta < \frac{2}{\|\hat{\Psi}(u^{(m)})\|} \), where \( \hat{\Psi}(u^{(m)}) = \frac{\partial \hat{\Psi}}{\partial u}_{u=u^{(m)}} \).

\textbf{Proof:} \( \text{(i) Proof of the existence of the inverse. This is straightforward as it is easy to verify by the intermediate value theorem that there exists a point } U_l < u < U_{l+1}, \text{ such that } v = \hat{\Psi}(u). \)

\( \text{(ii) Proof of convergency. Construct a Lyapunov function in the form of} \)
\[ L_m = (v - \hat{\Psi}(u^{(m)}))^2 \geq 0. \]  
(22)

Since \( \eta \) can be set as arbitrarily small, it is assumed that the following first order Taylor expansion is adequate
\[ \hat{\Psi}(u^{(m+1)}) \approx \hat{\Psi}(u^{(m)}) + \hat{\Psi}'(u^{(m)}) \Delta u^{(m)}. \]  
(23)

Clearly we have
\[ \text{sign} \left[ \frac{V_d - V_k}{U_d - U_k} \right] \hat{\Psi}'(u^{(m)}) = \left| \hat{\Psi}'(u^{(m)}) \right|. \]  
(24)
Hence, the closed-loop description of the system is

\[ y(t) = BHr(t), \]

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

\[ AF + BG = T(q^{-1}) = 1 + t_1q^{-1} + \cdots + t_nq^{-n}. \]
The coefficients of polynomials $F$, $G$ can be solved by setting $n_f = n_b + 1$, $n_g = n_a - 1$, $n \leq n_a + n_b + 1$. $H$ can be predetermined as desired subject to

$$\lim_{q \to 1} \frac{B(q^{-1})H(q^{-1})}{I(q^{-1})} = 1. \quad (31)$$

From Figure 2, it is clear that the actual control input $u(t)$ applied to the Hammerstein system is given by

$$u(t) = \hat{\Psi}^{-1}(\hat{v}(t)) = \hat{\Psi}^{-1}\left(\frac{Hr(t) - Gy(t)}{F}\right) \quad (32)$$

Rewriting (32) in a recursive form yields the following control law

1. $\hat{v}(t) = \sum_{j=0}^{n_a} b_j r(t-j) - \sum_{j=0}^{n_b} g_j y(t-j) - \sum_{j=1}^{n_f} \hat{J}(t-j)$,

2. Find $u(t) = \hat{\Psi}^{-1}(\hat{v}(t))$ using the inverse of De Boor algorithm. \quad (33)

Note that in practice if $\hat{v}(t)$ is out of the region between $\Psi(U_{min})$ and $\Psi(U_{max})$ (which is undesirable), $\hat{v}(t)$ is reset as 0 to avoid this to happen at the next time step. The computational cost of the inverse of De Boor algorithm is that of De Boor algorithm at $O(k^2)$, scaled by the number of iterations. The proposed controller is very efficient. First, because of the piecewise nature of B-spline, the polynomial order $k$ can be kept quite low without losing its approximation performance. Second the number of iterations needed for convergence should be low as the proposed algorithm provides an initial solution that is close to the real inverse.

### 4. Numerical examples

Two Hammerstein systems are simulated, in which the linear subsystems are the same for both systems, as $A(q^{-1}) = 1 - 1.2q^{-1} + 0.9q^{-2}$, $B(q^{-1}) = 1.7q^{-1} - q^{-2}$. For the nonlinear subsystem, $\Psi(\bullet)$ is given by

- System 1: $\Psi(u) = 2 \text{sign}(u)\sqrt{|u|}$, \quad (34)
- System 2: $\Psi(u) = -2 \text{sign}(u)u^2$, \quad (35)

respectively. Note that these nonlinear functions may be associated with physical devices such as nonlinear valves, e.g. the exit flow rate/fluid level in some parts of a pH neutralisation process (Henson and Seborg 1994).

The variances of the additive noise to the system output are set as $\sigma^2 = 0.0001$ and 0.01, respectively, for both systems. For each system, 1000 training data samples $r(t)$ were generated by using (1) and (2), where $u(t)$ was uniformly distributed random variable.
The polynomial degree of B-spline basis functions was set as \( k = 2 \) (piecewise quadratic). The knots sequence \( U_j \) is set as \([-3, -2.5, -2, -1, -0.3, 0, 0.3, 1, 2, 2.5, 3]\).

The resultant 8 basis functions are plotted in Figure 3. Initially, the system identification was carried out using the modelling algorithm outlined in Appendix A for each system. The modelling results are shown in Table 1, for the linear subsystem, and in Figure 4(a) and (b) for the nonlinear subsystems.

For both systems, the simulations of the pole assignment controllers were 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 as shown Figure 4, and by using parameter estimates given in Table 1, the required controller polynomials are estimated as

\[
P(q^{-1}) = 1 - 0.4862q^{-1}
\]

and

\[
G(q^{-1}) = 0.6425 - 0.4426q^{-1}
\]

and we predetermine

\[
H(q^{-1}) = 0.5300 + 0.1836q^{-1}
\]

The learning rate was preset as \( \eta = 0.02 \). The maximum value of iteration number \( m \) was predetermined as 200. The performance of the controllers for the two systems were tested. For each system, the reference signals \( r(t) \) are generated as a series of square waves with its magnitude at three different levels every 200 time steps. Figures 5(a) and 6(a) plot the computed control signal applied to each system, respectively. Figures 5(b) and 6(b) plot the system output \( y(t) \) together with the corresponding reference signal \( r(t) \) for both systems, respectively. From these figures, the proposed is shown method to have excellent results in terms of system identification as well as the subsequent control for the identified systems.

Figure 5. The results of the pole assignment controller for System 1.

Figure 6. The results of the pole assignment controller for System 2.
5. Conclusions
This brief article has introduced a simple and effective method for the modelling and control of the Hammerstein system based on observational input/output data. The modelling of the nonlinear static function in the Hammerstein system is based on the B-spline function approximation. The main contribution of this article is the calculation of the inverse of the estimated nonlinear function using a new algorithm, the inverse of De Boor algorithm. This step is used to remove the nonlinearity, lending to the linear controller methods. A pole assignment controller is used for completeness. Mathematical analysis is provided to prove the convergence of the proposed inverse of De Boor algorithm. The efficacy of the proposed approach have been demonstrated via two numerical examples.

Acknowledgements
The authors gratefully acknowledge that part of this work was supported by EPSRC in the UK. They also thank the reviewers for their valuable comments.

Notes on contributors

Xia Hong received her university education at National University of Defense Technology, P.R. China (BSc, 1984; MSc, 1987), and University of Sheffield, UK (PhD, 1998), all in automatic control. She worked as a research assistant in Beijing Institute of Systems Engineering, Beijing, China from 1987–1993. She is now Senior Lecturer in Cybernetics at School of Systems Engineering, University of Reading. She has been demonstrated via two numerical examples.

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

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

References


Appendix A: A summary of the modelling algorithm

(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 \([U_{\text{min}}, U_{\text{max}}]\). Predetermine a set of \((d + k)\) knots according to (4).

(2) Over the training data set \( D_t = \{y(t), u(t)\}_{t=1}^{N} \), construct \( d \) basis functions \( B_{\text{basis}}^{(d)}(u(t)) \) based on (5) and (6). Subsequently, apply the method described in Section 2.3 to find the parameter vector \( \hat{a} \) as the subvector of \( \hat{\theta}_{\text{LS}}^{(d)} \).

(3) Construct the auxiliary model output sequence \( z(t) \) using (15).

(4) Apply the Gauss–Newton algorithm subject to constraint (Hong and Mitchell 2007) to find \( b \) and \( a \).

(5) Based on \( \hat{a} \), the underlying function \( \Phi(*) \) for any point within the range \([U_{\text{min}}, U_{\text{max}}]\) can be recovered by applying the De Boor algorithm (using (5)–(7)).