DF2011 DSP21(2) Identification methods for Hammerstein nonlinear systems

2 AUTHORS, INCLUDING:

Jie Ding
Missouri University of Science and Tec…

86 PUBLICATIONS **1,369** CITATIONS

SEE PROFILE

Digital Signal Processing 21 (2011) 215–238

Contents lists available at ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

Identification methods for Hammerstein nonlinear systems ☆

Feng Ding [a,b,*], Xiaoping Peter Liu [c], Guangjun Liu [d]

[a] Key Laboratory of Advanced Process Control for Light Industry (Ministry of Education), Jiangnan University, Wuxi 214122, PR China
[b] Control Science and Engineering Research Center, Jiangnan University, Wuxi 214122, PR China
[c] Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada K1S 5B6
[d] Department of Aerospace Engineering, Ryerson University, Toronto, Canada M5B 2K3

ARTICLE INFO

Article history:
Available online 13 July 2010

Keywords:
Recursive identification
Parameter estimation
Stochastic gradient
Least squares
Projection algorithm
Newton iterations
Hammerstein models
Wiener models

ABSTRACT

This paper considers the identification problems of the Hammerstein nonlinear systems. A projection and a stochastic gradient (SG) identification algorithms are presented for the Hammerstein nonlinear systems by using the gradient search method. Since the projection algorithm is sensitive to noise and the SG algorithm has a slow convergence rate, a Newton recursive and a Newton iterative identification algorithms are derived by using the Newton method (Newton–Raphson method), in order to reduce the sensitivity of the projection algorithm to noise, and to improve convergence rates of the SG algorithm. Furthermore, the performances of these approaches are analyzed and compared using a numerical example, including the parameter estimation errors, the stationarity and convergence rates of parameter estimates and the computational efficiency.

1. Introduction

The parameter estimation methods have many important applications in system identification, signal processing, adaptive control, e.g., [1–9]. This considers the identification problem for Hammerstein systems.

The Hammerstein system consists of a static nonlinear block followed by a linear dynamic subsystem as shown in Fig. 1, e.g. in [10], where $y(t)$ is the measured output, $u(t)$ and $\bar{u}(t)$ are the input and output of the nonlinear block, respectively, $\{v(t)\}$ is a white noise sequence with zero mean and variance σ^2, and $A(z)$ and $B(z)$ are polynomials, of degrees n_a and n_b, in the unit backward shift operator z^{-1} $[z^{-1}y(t) = y(t-1)]$, with

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}, \quad \text{and}$$
$$B(z) = b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + \cdots + b_{n_b} z^{-n_b}.$$

The inner variables $\bar{u}(t)$, $x(t)$ and $w(t)$ are unmeasurable and $f(\cdot)$ is a static nonlinear function, including the polynomial nonlinearity [11–14], the piecewise nonlinearity [15], the monotonous (odd) nonlinearity [10,16], etc. The objective of identification is to estimate the coefficients a_i and b_i of the polynomials $A(z)$ and $B(z)$ and the parameters of the nonlinear function $f(\cdot)$ by using the available input–output data $\{u(t), y(t)\}$.

☆ This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), in in part by the Canada Research Chair Program and in part by the National Natural Science Foundation of China (No. 60973043). This research was completed while the first author visited Carleton University (Ottawa, Canada) and Ryerson University (Toronto, Canada).

* Corresponding author at: Control Science and Engineering Research Center, Jiangnan University, Wuxi 214122, PR China.
E-mail addresses: fding@jiangnan.edu.cn (F. Ding), xpliu@sce.carleton.ca (X.P. Liu), gjliu@ryerson.ca (G. Liu).

1051-2004/$ – see front matter
doi:10.1016/j.dsp.2010.06.006

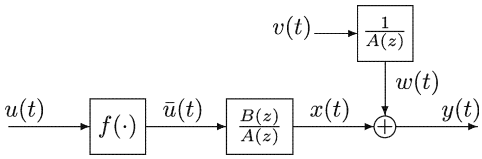

Fig. 1. The Hammerstein system.

Much work has been performed on the parametric model identification of Hammerstein systems. Some approaches assumed that the nonlinearity is a two-segment piecewise linear or multi-segment piecewise-linear function [15,17], a discontinuous function [18–20] or hard nonlinearities [21]. Some methods assumed that the nonlinearity is monotonous and odd [10,16], or invertible [22], and thus the corresponding Hammerstein identification problem can be dealt with using the iterative identification technique [10,16]. The others usually assumed that the nonlinearity $\bar{u} = f(u)$ is a polynomial of a known order in the input [11–15] or is generally expressed by the sum of the nonlinear (orthogonal or nonorthogonal) basis functions of a known basis $\boldsymbol{f} := (f_1, f_2, \ldots, f_{n_c})$ and unknown coefficients c_i [10,23–29]

$$\begin{aligned}\bar{u}(t) &= f\big(u(t)\big) = c_1 f_1\big(u(t)\big) + c_2 f_2\big(u(t)\big) + \cdots + c_{n_c} f_{n_c}\big(u(t)\big) \\ &= \boldsymbol{f}\big(u(t)\big)\boldsymbol{c},\end{aligned} \tag{1}$$

where $\boldsymbol{c} := [c_1, c_2, \ldots, c_{n_c}]^{\mathrm{T}}$.

Existing parametric identification of Hammerstein systems includes the over-parameterization method, the subspace method, the separable least squares method, the blind method, the iterative method, and so on [16].

1. **The over-parameterization method**
   The over-parameterization method is to use the expression of the nonlinearity as a sum of the basis functions to over-parameterize the Hammerstein system, e.g., [12], so that the final parameter vector or cost function will involve cross-products between parameters in the static nonlinearity and those in the linear dynamical subsystem. In other words, the unknown parameters and parameter products appear to be linear in the parameter vector [12,30] and then any linear estimation algorithm can be applied. But the difficulty is that the dimension of the resulting unknown parameter vector is usually very large and computational load increases [12,24–27,30,31].
2. **The subspace method**
   The subspace method is an extension of linear cases to nonlinear cases. In this literature, Verhaegen and Westwick [32] extended the MOESP family of subspace model identification schemes to the Hammerstein-type nonlinear system by assuming that the static nonlinearity has the polynomial structure or is the sum of nonlinear functions with known basis. Goethals, Pelckmans, Suykens and De Moor extended the numerical algorithms of subspace state space identification (N4SID) for linear systems to Hammerstein nonlinear systems by using the oblique projection in the N4SID algorithm as a set of componentwise least squares support vector machines regression problems [33].
3. **The separable least squares method**
   The separable nonlinear least squares approach is to write one set of variables as a function of the other set and the identification problem of Hammerstein nonlinear systems can be transformed into solving two optimization problems [10,21,34–37]. Bai used the separable least squares method to form two cost functions and studied the identification methods for the Hammerstein systems with hard input (nonsmooth) nonlinearities of known structures based on the first-order necessary and sufficient conditions [21]. Golub and Pereyra applied the projection method for solving separable nonlinear least squares problems [34]. Bruls, Chou, Haverkamp and Verhaegen exploited the separable least squares technique in the identification of both linear and nonlinear systems based on the prediction error formulation using the nonlinear output error model and innovation model [36].
4. **The blind identification method**
   The blind method is to use the technique of blind system identification to identify the linear part of the Hammerstein system without requiring the structure of unknown nonlinearity. The basic idea is to fast sample at the output and the linear part can be identified based only on the output measurements [38]. The results have been extended to Hammerstein–Wiener systems [22]. Recently, Wang, Sano, Shook, Chen and Huang studied the blind identification approach of a closed-loop Hammerstein system using the fast output samples [39]; Vanbeylen, Pintelon and Schoukens studied blind maximum likelihood identification problems of Hammerstein systems with the Gaussian noise input, invertible nonlinearity and no output measurement error [40].
5. **The iterative identification method**
   The iterative method is very important for solving matrix equations [41–47]. The iterative technique can be used to study system identification problems. For example, Ding et al. presented the gradient based and least squares based iterative identification methods for OE and OEMA systems [48], the least squares based and gradient based iterative algorithms for identifying Box–Jenkins models with finite measurement data [49,50], and the hierarchical gradient based and hierarchical least squares based iterative identification algorithms for multivariable systems [51–54]; Liu and Lu proposed the least squares based iterative identification algorithm for a class of multirate systems [55].

The iterative identification of Hammerstein systems can be traced back to 1960's in Narendra and Gallman's work [11]. This class of iterative approaches divides the parameters into two sets, linear part and nonlinear part. One calculates the optimal value for one set while the other set is fixed. Then, two sets are switched. This approach was first proposed by Narendra and Gallman has been extensively studied [10,18,56,57]. For example, Haist, Chang and Luus gave an iterative algorithm of identifying the Hammerstein model with correlated noises [13]; Bai and Li studied the convergence properties of the iterative algorithm of the finite impulse response Hammerstein model with odd nonlinearity and showed that the iterative algorithm with normalization is generally convergent with the i.i.d. input signal [16]; Ding et al. presented the least squares based and gradient based iterative estimation algorithms for Hammerstein nonlinear ARMAX systems [24,25]; Liu and Bai reported a normalized iterative approach for identifying the Hammerstein models with odd nonlinearity [10].

6. **Other identification methods**
   In addition to the contributions mentioned above, a lot of work on Hammerstein system identification exists in the literature. For example, Vörös studied the parameter identification problems of Hammerstein systems with discontinuous nonlinearities, two-segment nonlinearities and multi-segment nonlinearities using the key term separation principle [15,17–19]. Recently, Wang et al. proposed an auxiliary model-based RELS and MI-ELS algorithms for Hammerstein OEMA systems and an auxiliary model-based recursive generalized least squares parameter estimation for Hammerstein OEAR systems using the key term separation principle [58,59]. Schoukens, Widanage, Godfrey and Pintelon studied the initial estimates for the dynamic part of a Hammerstein model and showed that ARMAX or Box–Jenkins models result in better initial estimates than ARX or output-error models [60]. Giri, Rochdi, Chaoui and Brouri addressed the identification problem of Hammerstein systems in presence of hysteresis-backlash and hysteresis-relay nonlinearities and separately estimated the parameters of the linear subsystem and the nonlinear element using the least-squares like estimators [61].
   Recently, some new identification methods are developed for linear systems and can be extended to nonlinear systems. They are the auxiliary model based identification methods [62–68], the hierarchical identification methods [51–53,69, 70], the multi-innovation identification methods [65,67,71–80], the interactive least squares algorithm [81] and the interactive stochastic gradient algorithm [82]. Also, Ding et al. proposed the auxiliary model based recursive least squares algorithm for Hammerstein output error systems [26].

The main contributions of this paper are as follows:

- To derive gradient based identification algorithms for the Hammerstein systems by using the gradient search method.
- To present a Newton recursive and a Newton iterative identification algorithms for the Hammerstein systems using the Newton method (Newton–Raphson method).
- To analyze and compare the performances of the proposed approaches using a numerical example, including the convergence rates and the estimation errors of the algorithms for finite measurement data.

Although the proposed algorithms are developed for Hammerstein ARX systems with white noises, the basic idea can be extended to identify Hammerstein systems with colored noises [24,26], Wiener systems [83,84], or Hammerstein–Wiener systems [27,30]. The proposed algorithms differ from the least squares or gradient based iterative/recursive ones in [24–27] using the over-parameterization method.

The rest of this paper is organized as follows. Section 2 describes the identification problem formulation for Hammerstein nonlinear systems. Sections 3 and 4 derive the projection and stochastic gradient algorithms for Hammerstein systems. Sections 5 and 6 derive the Newton recursive and Newton iterative algorithm for Hammerstein systems. Section 7 provides an illustrative example and compares the performances of the proposed algorithms. Finally, we offer some concluding remarks in Section 8.

2. The identification model of the Hammerstein system

Let us introduce some notations first. The superscript T denotes the matrix transpose; $\mathbf{1}_n$ represents an n-dimensional column vector whose elements are 1; the norm of a matrix $\boldsymbol{X}$ is defined by $\|\boldsymbol{X}\|^2 = \mathrm{tr}[\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}]$; $\widehat{\boldsymbol{X}}(t)$ stands for the estimate of $\boldsymbol{X}$ at time t.

The Hammerstein system in Fig. 1 can be expressed as

$$A(z)y(t) = B(z)\bar{u}(t) + v(t). \tag{2}$$

Define the parameter vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ of the linear subsystem and $\boldsymbol{c}$ of the nonlinear part as

$$\boldsymbol{a} := [a_1, a_2, \ldots, a_{n_a}]^{\mathrm{T}} \in \mathbb{R}^{n_a},$$
$$\boldsymbol{b} := [b_1, b_2, \ldots, b_{n_b}]^{\mathrm{T}} \in \mathbb{R}^{n_b},$$
$$\boldsymbol{c} := [c_1, c_2, \ldots, c_{n_c}]^{\mathrm{T}} \in \mathbb{R}^{n_c},$$

$$\boldsymbol{\theta} := \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \\ \boldsymbol{c} \end{bmatrix} \in \mathbb{R}^n, \quad n := n_a + n_b + n_c,$$

and the output information vector $\boldsymbol{\varphi}(t)$ and input information matrix $\boldsymbol{F}(t)$ as

$$\boldsymbol{\varphi}(t) := \big[-y(t-1), -y(t-2), \ldots, -y(t-n_a)\big]^{\mathrm{T}} \in \mathbb{R}^{n_a}, \tag{3}$$

$$\boldsymbol{F}(t) := \begin{bmatrix} \boldsymbol{f}(u(t-1)) \\ \boldsymbol{f}(u(t-2)) \\ \vdots \\ \boldsymbol{f}(u(t-n_b)) \end{bmatrix} \in \mathbb{R}^{n_b \times n_c}. \tag{4}$$

From (2) and (1), we have the identification model of the Hammerstein systems as follows:

$$y(t) = \boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c} + v(t). \tag{5}$$

Note that for any pair $\alpha\boldsymbol{b}$ and $\boldsymbol{c}/\alpha$ ($\alpha \neq 0$), the system in (5) has the identity input–output relationship. To have identifiability, we adopt the normalization constraint on $\boldsymbol{c}$ for model (5). Without loss of generality, we adopt the following assumption.

Assumption 1. $\|\boldsymbol{c}\| = 1$, and the first nonzero entry of $\boldsymbol{c}$ is positive. That is, the first coefficient of the function $f(\cdot)$ is positive, i.e., $c_1 > 0$ [10].

By the over-parameterization of the system in (5), we get a new regression model

$$y(t) = \boldsymbol{\zeta}^{\mathrm{T}}(t)\boldsymbol{\vartheta} + v(t), \tag{6}$$

which contains the products of the parameters, where the over-parameter vector $\boldsymbol{\vartheta}$ and information vector $\boldsymbol{\zeta}(t)$ are defined by

$$\boldsymbol{\vartheta} := \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \otimes \boldsymbol{c} \end{bmatrix} \in \mathbb{R}^{n_a + n_b n_c},$$

$$\boldsymbol{\zeta}(t) := \big[\boldsymbol{\varphi}^{\mathrm{T}}(t), \boldsymbol{f}\big(u(t-1)\big), \boldsymbol{f}\big(u(t-2)\big), \ldots, \boldsymbol{f}\big(u(t-n_b)\big)\big]^{\mathrm{T}} \in \mathbb{R}^{n_a + n_b n_c},$$

$\boldsymbol{b} \otimes \boldsymbol{c}$ represents the Kronecker product of $\boldsymbol{b}$ and $\boldsymbol{c}$.

Although the stochastic gradient algorithm [85]

$$\begin{bmatrix} \hat{\boldsymbol{a}}(t) \\ \widehat{\boldsymbol{b} \otimes \boldsymbol{c}}(t) \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{a}}(t-1) \\ \widehat{\boldsymbol{b} \otimes \boldsymbol{c}}(t-1) \end{bmatrix} + \frac{\boldsymbol{\zeta}(t)}{r(t)} \left\{ y(t) - \boldsymbol{\zeta}^{\mathrm{T}}(t) \begin{bmatrix} \hat{\boldsymbol{a}}(t-1) \\ \widehat{\boldsymbol{b} \otimes \boldsymbol{c}}(t-1) \end{bmatrix} \right\}, \tag{7}$$

$$r(t) = r(t-1) + \big\|\boldsymbol{\zeta}(t)\big\|^2, \quad r(0) = 1 \tag{8}$$

can identify the over-parameterization model in (6), they require estimating $n_a + n_b n_c$ parameters in $\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \otimes \boldsymbol{c} \end{bmatrix} \in \mathbb{R}^{n_a + n_b n_c}$, which is greater than $n = n_a + n_b + n_c$ parameters in the original system in (5) for $n_b, n_c > 2$. The resulting problem is that the corresponding identification algorithms require more computational burden.

The objective of this paper is to study and present new identification methods to estimate the parameter vectors $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ in the system in (2) instead of $\boldsymbol{\vartheta}$ by using the gradient search method and Newton method.

3. The projection identification algorithm

In the following, we derive the projection algorithm for identifying the Hammerstein systems using the gradient search method.

Define the cost function:

$$J_1(\boldsymbol{\theta}) = J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) = \big[y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}\big]^2. \tag{9}$$

The gradients of J_1 with respect to $\boldsymbol{\theta}$ is

$$\mathrm{grad}_{\theta}\big[J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\big] = \begin{bmatrix} \frac{\partial J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})}{\partial \boldsymbol{a}} \\ \frac{\partial J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})}{\partial \boldsymbol{b}} \\ \frac{\partial J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})}{\partial \boldsymbol{c}} \end{bmatrix} = -2 \begin{bmatrix} \boldsymbol{\varphi}(t) \\ \boldsymbol{F}(t)\boldsymbol{c} \\ \boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{b} \end{bmatrix} \big[y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}\big].$$

Let $\hat{\boldsymbol{\theta}}(t) := \begin{bmatrix} \hat{\boldsymbol{a}}_t \\ \hat{\boldsymbol{b}}_t \\ \hat{\boldsymbol{c}}_t \end{bmatrix}$ be the estimate of $\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \\ \boldsymbol{c} \end{bmatrix}$ at time t. Define the generalized information vector $\boldsymbol{\xi}(t)$ and innovation $e(t)$ as

$$\boldsymbol{\xi}(t) := \begin{bmatrix} \boldsymbol{\varphi}(t) \\ \boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1} \\ \boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1} \end{bmatrix} \in \mathbb{R}^{n_a+n_b+n_c},$$

$$e(t) := y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\hat{\boldsymbol{a}}_{t-1} - \hat{\boldsymbol{b}}_{t-1}^{\mathrm{T}}\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}.$$

For the optimization problems in (9), minimizing J_1 and using the negative gradient search lead to the following recursive algorithm:

$$\begin{aligned} \hat{\boldsymbol{\theta}}(t) &= \hat{\boldsymbol{\theta}}(t-1) - \frac{\mu_t}{2}\operatorname{grad}_{\theta}\big[J_1(\hat{\boldsymbol{a}}_{t-1}, \hat{\boldsymbol{b}}_{t-1}, \hat{\boldsymbol{c}}_{t-1})\big] \\ &= \hat{\boldsymbol{\theta}}(t-1) + \mu_t\boldsymbol{\xi}(t)\big[y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\hat{\boldsymbol{a}}_{t-1} - \hat{\boldsymbol{b}}_{t-1}^{\mathrm{T}}\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}\big] \\ &= \hat{\boldsymbol{\theta}}(t-1) + \mu_t\boldsymbol{\xi}(t)e(t), \end{aligned} \tag{10}$$

or

$$\begin{aligned} \begin{bmatrix} \hat{\boldsymbol{a}}_t \\ \hat{\boldsymbol{b}}_t \\ \hat{\boldsymbol{c}}_t \end{bmatrix} &= \begin{bmatrix} \hat{\boldsymbol{a}}_{t-1} \\ \hat{\boldsymbol{b}}_{t-1} \\ \hat{\boldsymbol{c}}_{t-1} \end{bmatrix} + \mu_t \begin{bmatrix} \boldsymbol{\varphi}(t)e(t) \\ \boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}e(t) \\ \boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t) \end{bmatrix} \\ &= \begin{bmatrix} \hat{\boldsymbol{a}}_{t-1} + \mu_t\boldsymbol{\varphi}(t)e(t) \\ \hat{\boldsymbol{b}}_{t-1} + \mu_t\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}e(t) \\ \hat{\boldsymbol{c}}_{t-1} + \mu_t\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t) \end{bmatrix}, \end{aligned} \tag{11}$$

where $\mu_t > 0$ is the step-size or convergence factor.

Assume that $e(t) \neq 0$ and $\boldsymbol{\xi}(t) \neq 0$ (otherwise, let $\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1)$). In order to determine μ_t using the gradient search method, substituting $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}(t)$ into (9) gets

$$\begin{aligned} g(\mu_t) &:= J_1\big[\hat{\boldsymbol{\theta}}(t)\big] \\ &= \big\{y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\big[\hat{\boldsymbol{a}}_{t-1} + \mu_t\boldsymbol{\varphi}(t)e(t)\big] - \big[\hat{\boldsymbol{b}}_{t-1} + \mu_t\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}e(t)\big]^{\mathrm{T}}\boldsymbol{F}(t)\big[\hat{\boldsymbol{c}}_{t-1} + \mu_t\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t)\big]\big\}^2 \\ &= \big[y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\hat{\boldsymbol{a}}_{t-1} - \mu_t\big\|\boldsymbol{\varphi}(t)\big\|^2e(t) - \hat{\boldsymbol{b}}_{t-1}^{\mathrm{T}}\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1} - \mu_t\hat{\boldsymbol{b}}_{t-1}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t) \\ &\quad - \mu_t\hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}e(t) - \mu_t^2\hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{F}(t)\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e^2(t)\big]^2 \\ &= \big[e(t) - \mu_t\big\|\boldsymbol{\varphi}(t)\big\|^2e(t) - \mu_t\big\|\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}\big\|^2e(t) - \mu_t\big\|\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}\big\|^2e(t) - \mu_t^2\hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{F}(t)\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e^2(t)\big]^2 \\ &= \big[1 - \mu_t\big\|\boldsymbol{\varphi}(t)\big\|^2 - \mu_t\big\|\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}\big\|^2 - \mu_t\big\|\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1}\big\|^2 - \mu_t^2\hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{F}(t)\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t)\big]^2e^2(t) \\ &= \big[1 - \mu_t\big\|\boldsymbol{\xi}(t)\big\|^2 - \mu_t^2\hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{F}(t)\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t)\big]^2e^2(t) \\ &=: \big[1 - \mu_t\big\|\boldsymbol{\xi}(t)\big\|^2 - \mu_t^2\kappa_t\big]^2e^2(t), \end{aligned}$$

where

$$\kappa_t := \hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{F}(t)\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1}e(t). \tag{12}$$

The best step-size μ_t can be obtained by minimizing $g(\mu_t)$. If $\kappa_t = 0$, then minimizing $g(\mu_t)$ gives the best step-size:

$$\mu_t = \frac{1}{\|\boldsymbol{\xi}(t)\|^2}; \tag{13}$$

otherwise, μ_t is the positive solution of the following equation:

$$\frac{\partial J_1(\mu_t)}{\partial \mu_t} = 2\big[\mu_t^2\kappa_t + \mu_t\big\|\boldsymbol{\xi}(t)\big\|^2 - 1\big]\big[2\mu_t\kappa_t + \big\|\boldsymbol{\xi}(t)\big\|^2\big] = 0,$$

such that $g(\mu_t) = \min$. In the case with $\kappa_t \neq 0$, the best step-size is given by

$$\mu_t = \frac{\sqrt{\|\boldsymbol{\xi}(t)\|^4 + 4\kappa_t} - \|\boldsymbol{\xi}(t)\|^2}{2\kappa_t} = \frac{2}{\sqrt{\|\boldsymbol{\xi}(t)\|^4 + 4\kappa_t} + \|\boldsymbol{\xi}(t)\|^2}.$$

When $\kappa_t = 0$, the above equation reduces to (13). That is

$$\mu_t = \begin{cases} \frac{1}{\|\boldsymbol{\xi}(t)\|^2}, & \text{if } \kappa_t = 0, \\ \frac{\sqrt{\|\boldsymbol{\xi}(t)\|^4 + 4\kappa_t} - \|\boldsymbol{\xi}(t)\|^2}{2\kappa_t}, & \text{otherwise}, \end{cases}$$

which is equivalent to

$$\mu_t = \frac{2}{\sqrt{\|\boldsymbol{\xi}(t)\|^4 + 4\kappa_t} + \|\boldsymbol{\xi}(t)\|^2}.$$

Thus, for any κ_t, the step-size is given by the above equation.

Therefore, **the projection identification algorithm for the Hammerstein models** (H-Proj algorithm for short) can be summarized as follows:

$$\hat{\boldsymbol{\theta}}(t) = \begin{cases} \hat{\boldsymbol{\theta}}(t-1), & \text{if } e(t) = 0 \text{ or } \boldsymbol{\xi}(t) = 0, \\ \hat{\boldsymbol{\theta}}(t-1) + \mu_t \boldsymbol{\xi}(t) e(t), & \text{otherwise}, \end{cases} \tag{14}$$

$$\mu_t = \frac{2}{\sqrt{\|\boldsymbol{\xi}(t)\|^4 + 4\kappa_t} + \|\boldsymbol{\xi}(t)\|^2}, \tag{15}$$

$$\kappa_t = \hat{\boldsymbol{c}}_{t-1}^{\mathrm{T}} \boldsymbol{F}^{\mathrm{T}}(t) \boldsymbol{F}(t) \boldsymbol{F}^{\mathrm{T}}(t) \hat{\boldsymbol{b}}_{t-1} e(t), \tag{16}$$

$$e(t) = y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t) \hat{\boldsymbol{a}}_{t-1} - \hat{\boldsymbol{b}}_{t-1}^{\mathrm{T}} \boldsymbol{F}(t) \hat{\boldsymbol{c}}_{t-1}, \tag{17}$$

$$\boldsymbol{\xi}(t) = \begin{bmatrix} \boldsymbol{\varphi}(t) \\ \boldsymbol{F}(t) \hat{\boldsymbol{c}}_{t-1} \\ \boldsymbol{F}^{\mathrm{T}}(t) \hat{\boldsymbol{b}}_{t-1} \end{bmatrix}, \tag{18}$$

$$\boldsymbol{\varphi}(t) = \big[-y(t-1), -y(t-2), \ldots, -y(t-n_a)\big]^{\mathrm{T}}, \tag{19}$$

$$\boldsymbol{F}(t) = \begin{bmatrix} f_1(u(t-1)) & f_2(u(t-1)) & \cdots & f_{n_c}(u(t-1)) \\ f_1(u(t-2)) & f_2(u(t-2)) & \cdots & f_{n_c}(u(t-2)) \\ \vdots & \vdots & & \vdots \\ f_1(u(t-n_b)) & f_2(u(t-n_b)) & \cdots & f_{n_c}(u(t-n_b)) \end{bmatrix}. \tag{20}$$

Normalize $\hat{\boldsymbol{c}}_t$ with the first positive element, i.e.,

$$\hat{\boldsymbol{c}}_t = \operatorname{sgn}\big[\hat{\boldsymbol{\theta}}_{n_a+n_b+1}(t)\big] \frac{[\hat{\boldsymbol{\theta}}(t)](n_a+n_b+1:n)}{\|[\hat{\boldsymbol{\theta}}(t)](n_a+n_b+1:n)\|}, \tag{21}$$

and set

$$\big[\hat{\boldsymbol{\theta}}(t)\big](n_a+n_b+1:n) = \hat{\boldsymbol{c}}_t, \tag{22}$$

where $\operatorname{sgn}[\hat{\boldsymbol{\theta}}_i(t)]$ represents the sign of the ith element of $\hat{\boldsymbol{\theta}}(t)$ and the ":" operation in Matlab is used.

Eq. (14) can be equivalently rewritten as

$$\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1) + \mu_t \boldsymbol{\xi}(t) e(t).$$

The H-Proj algorithm in (14)–(22) for the Hammerstein models is more complicated than the projection algorithm for linear systems in [85]. Referring to the projection algorithm for the linear systems in [85], Eqs. (14)–(16) can be simplified as

$$\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1) + \frac{\boldsymbol{\xi}(t)}{1 + \|\boldsymbol{\xi}(t)\|^2} e(t). \tag{23}$$

Adding unity in the denominator on the right-hand side is to deal with the case with $\|\boldsymbol{\xi}(t)\| = 0$.

Eqs. (17) to (23) form **the simplified projection algorithm for the Hammerstein models** (the H-S-Proj algorithm for short).

4. The stochastic gradient algorithm

For the stochastic Hammerstein systems, the H-Proj algorithm is sensitive to noise like the projection algorithm for the linear systems because the gain vector $\mu_t \boldsymbol{\xi}(t)$ in (14) or $\frac{\boldsymbol{\xi}(t)}{1+\|\boldsymbol{\xi}(t)\|^2}$ in (23) does not approach zero. In order to adjust the gain vector of the algorithm and referring to the stochastic gradient algorithm for the linear systems in [85], we take the

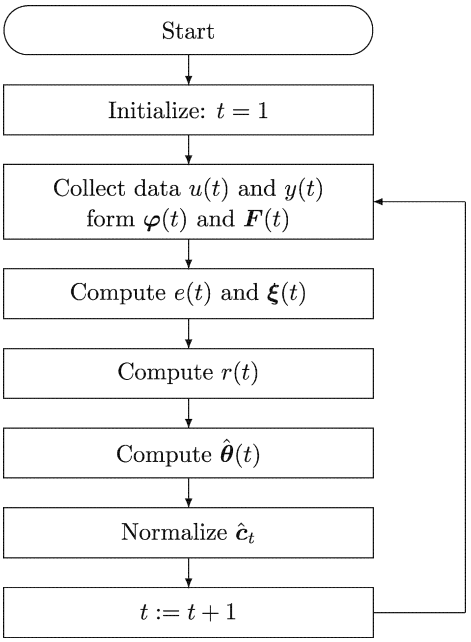

Fig. 2. The flowchart of computing $\hat{\boldsymbol{\theta}}(t)$ in the H-FF-SG algorithm.

convergence factor to be $\mu_t = \frac{1}{r(t)}$ and

$$\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1) + \frac{\boldsymbol{\xi}(t)}{r(t)} e(t), \tag{24}$$

$$r(t) = \lambda r(t-1) + \left\|\boldsymbol{\xi}(t)\right\|^2, \quad 0 \leqslant \lambda \leqslant 1,\ r(0) = 1, \tag{25}$$

where λ is the forgetting factor. Eqs. (24), (25) and (17)–(22) form **the stochastic gradient algorithm with the forgetting factor for the Hammerstein models** (H-FF-SG). When $\lambda = 0$, the H-FF-SG algorithm reduces to the projection algorithm; when $\lambda = 1$, the H-FF-SG algorithm becomes **the stochastic gradient algorithm for the Hammerstein models** (H-SG).

The computation procedure of the H-FF-SG algorithm in (24), (25) and (17)–(22) is summarized as follows:

1. Given forgetting factor λ and $r(0) = 1$. To initialize: let $t = 1$, $\hat{\boldsymbol{\theta}}(0) = \begin{bmatrix} \hat{\boldsymbol{a}}_0 \\ \hat{\boldsymbol{b}}_0 \\ \hat{\boldsymbol{c}}_0 \end{bmatrix}$ be an arbitrary real vector with $\|\hat{\boldsymbol{c}}_0\| = 1$.
2. Collect the input–output data $u(t)$ and $y(t)$ and form $\boldsymbol{\varphi}(t)$ by (19) and $\boldsymbol{F}(t)$ by (20).
3. Compute $e(t)$ by (17) and $\boldsymbol{\xi}(t)$ by (18).
4. Compute $r(t)$ by (25).
5. Compute $\hat{\boldsymbol{\theta}}(t)$ by (24).
6. Normalize $\hat{\boldsymbol{c}}_t$ by (21) and (22) with the first positive element.
7. Increase t by 1 and go to step 2.

The flowchart of computing the parameter estimates $\hat{\boldsymbol{\theta}}(t)$ in the H-FF-SG algorithm is shown in Fig. 2.

5. The Newton recursive identification algorithm

This section derives the Newton recursive identification algorithm for the Hammerstein systems using the Newton method. The basic idea is introducing the stacked output vector and stacked information matrices and defining a new cost function.

Using the Newton method to solve identification optimization problems, the stacked data have to be used because the following Hessian matrix $\boldsymbol{H}[J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})]$ of the cost function J_1 is singular,

$$\boldsymbol{H}\big[J_1(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\big] = 2 \begin{bmatrix} \boldsymbol{\varphi}(t)\boldsymbol{\varphi}^{\mathrm{T}}(t) & \boldsymbol{\varphi}(t)\boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t) & \boldsymbol{\varphi}(t)\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t) \\ \boldsymbol{F}(t)\boldsymbol{c}\boldsymbol{\varphi}^{\mathrm{T}}(t) & \boldsymbol{F}(t)\boldsymbol{c}\boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t) & \boldsymbol{h}_{23}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, t) \\ \boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{b}\boldsymbol{\varphi}^{\mathrm{T}}(t) & \boldsymbol{h}_{32}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, t) & \boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{b}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t) \end{bmatrix} \in \mathbb{R}^{n \times n},$$

where

$$\begin{aligned}\boldsymbol{h}_{23}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},t) &:= -\frac{\partial}{\partial \boldsymbol{c}}\{\boldsymbol{F}(t)\boldsymbol{c}[y(t)-\boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}]\}\\ &= -\boldsymbol{F}(t)[y(t)-\boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}]+\boldsymbol{F}(t)\boldsymbol{c}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\in\mathbb{R}^{n_b\times n_c},\end{aligned}$$

$$\begin{aligned}\boldsymbol{h}_{32}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},t) &:= -\frac{\partial}{\partial \boldsymbol{b}}\{\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{b}[y(t)-\boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}]\}\\ &= -\boldsymbol{F}^{\mathrm{T}}(t)[y(t)-\boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}]+\boldsymbol{F}^{\mathrm{T}}(t)\boldsymbol{b}\boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\\ &= \boldsymbol{h}_{23}^{\mathrm{T}}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},t)\in\mathbb{R}^{n_c\times n_b}.\end{aligned}$$

Consider the newest p data and define stacked output vector $\boldsymbol{Y}(p,t)$ and stacked matrices:

$$\boldsymbol{Y}(p,t):=\begin{bmatrix}y(t)\\ y(t-1)\\ \vdots\\ y(t-p+1)\end{bmatrix}\in\mathbb{R}^p,\qquad \boldsymbol{\Phi}_0(p,t):=\begin{bmatrix}\boldsymbol{\varphi}^{\mathrm{T}}(t)\\ \boldsymbol{\varphi}^{\mathrm{T}}(t-1)\\ \vdots\\ \boldsymbol{\varphi}^{\mathrm{T}}(t-p+1)\end{bmatrix}\in\mathbb{R}^{p\times n_a},$$

$$\boldsymbol{\Phi}(\boldsymbol{c},t):=\begin{bmatrix}\boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t)\\ \boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t-1)\\ \vdots\\ \boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(t-p+1)\end{bmatrix}\in\mathbb{R}^{p\times n_b},\qquad \boldsymbol{\Psi}(\boldsymbol{b},t):=\begin{bmatrix}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\\ \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-1)\\ \vdots\\ \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-p+1)\end{bmatrix}\in\mathbb{R}^{p\times n_c}. \tag{26}$$

Introduce a new cost function:

$$J_2(\boldsymbol{\theta})=J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}):=\big\|\boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\boldsymbol{a}-\boldsymbol{\Psi}(\boldsymbol{b},t)\boldsymbol{c}\big\|^2 \tag{27}$$

$$=\big\|\boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\boldsymbol{a}-\boldsymbol{\Phi}(\boldsymbol{c},t)\boldsymbol{b}\big\|^2, \tag{28}$$

which is equivalent to the following cost function defined by using the data of the dynamical window with length p:

$$J_3(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})=\sum_{i=t-p+1}^{t}[y(i)-\boldsymbol{\varphi}^{\mathrm{T}}(i)\boldsymbol{a}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(i)\boldsymbol{c}]^2. \tag{29}$$

That is $J_2(\boldsymbol{\theta})=J_3(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})$. If we take $t=N$ and $p=N$ (N is the data length), then they are the least squares cost functions in [10].

Compute the gradient of J_2 with respect to $\boldsymbol{\theta}$:

$$\begin{aligned}\mathrm{grad}_\theta[J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})] &= \begin{bmatrix}\frac{\partial J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\partial \boldsymbol{a}}\\ \frac{\partial J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\partial \boldsymbol{b}}\\ \frac{\partial J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\partial \boldsymbol{c}}\end{bmatrix}=-2\begin{bmatrix}\boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\\ \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c},t)\\ \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b},t)\end{bmatrix}[\boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\boldsymbol{a}-\boldsymbol{\Psi}(\boldsymbol{b},t)\boldsymbol{c}]\\ &= -2\begin{bmatrix}\boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\\ \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c},t)\\ \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b},t)\end{bmatrix}[\boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\boldsymbol{a}-\boldsymbol{\Phi}(\boldsymbol{c},t)\boldsymbol{b}].\end{aligned}$$

Define the generalized information matrix $\boldsymbol{\Xi}(t)$ and the innovation vector $\boldsymbol{E}(p,t)$ as

$$\boldsymbol{\Xi}(t):=\begin{bmatrix}\boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\\ \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{t-1},t)\\ \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{t-1},t)\end{bmatrix}\in\mathbb{R}^{n\times p},$$

$$\begin{aligned}\boldsymbol{E}(p,t) &:= \boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\hat{\boldsymbol{a}}_{t-1}-\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t)\hat{\boldsymbol{c}}_{t-1}\\ &= \boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\hat{\boldsymbol{a}}_{t-1}-\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t)\hat{\boldsymbol{b}}_{t-1}\in\mathbb{R}^p.\end{aligned} \tag{30}$$

Thus, we have

$$\mathrm{grad}_\theta[J_2(\hat{\boldsymbol{a}}_{t-1},\hat{\boldsymbol{b}}_{t-1},\hat{\boldsymbol{c}}_{t-1}]=-2\begin{bmatrix}\boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\\ \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{t-1},t)\\ \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{t-1},t)\end{bmatrix}[\boldsymbol{Y}(p,t)-\boldsymbol{\Phi}_0(p,t)\hat{\boldsymbol{a}}_{t-1}-\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t)\hat{\boldsymbol{c}}_{t-1}]$$

$$= -2\begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{t-1},t) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{t-1},t) \end{bmatrix}\big[\boldsymbol{Y}(p,t) - \boldsymbol{\Phi}_0(p,t)\hat{\boldsymbol{a}}_{t-1} - \boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t)\hat{\boldsymbol{b}}_{t-1}\big]$$
$$= -2\boldsymbol{\Xi}(t)\boldsymbol{E}(p,t). \tag{31}$$

Compute the Hessian matrix of the cost function J_2:

$$\boldsymbol{H}\big[J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})\big] = \frac{\partial\,\mathrm{grad}_\theta[J_2(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})]}{\partial\boldsymbol{\theta}^{\mathrm{T}}}$$
$$= 2\begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\boldsymbol{\Phi}_0(p,t) & \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\boldsymbol{\Phi}(\boldsymbol{c},t) & \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\boldsymbol{\Psi}(\boldsymbol{b},t) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c},t)\boldsymbol{\Phi}_0(p,t) & \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c},t)\boldsymbol{\Phi}(\boldsymbol{c},t) & \boldsymbol{H}_{23}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},t) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b},t)\boldsymbol{\Phi}_0(p,t) & \boldsymbol{H}_{23}^{\mathrm{T}}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},t) & \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b},t)\boldsymbol{\Psi}(\boldsymbol{b},t) \end{bmatrix} \in \mathbb{R}^{n\times n},$$

where

$$\boldsymbol{H}_{23}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},t) := -\frac{\partial}{\partial\boldsymbol{c}}\big\{\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c},t)\big[\boldsymbol{Y}(p,t) - \boldsymbol{\Phi}_0(p,t)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b},t)\boldsymbol{c}\big]\big\}$$
$$= -\frac{\partial}{\partial\boldsymbol{c}}\big\{\big[\boldsymbol{F}(t)\boldsymbol{c},\boldsymbol{F}(t-1)\boldsymbol{c},\ldots,\boldsymbol{F}(t-p+1)\boldsymbol{c}\big]\big[\boldsymbol{Y}(p,t) - \boldsymbol{\Phi}_0(p,t)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b},t)\boldsymbol{c}\big]\big\}$$
$$= -\frac{\partial}{\partial\boldsymbol{c}}\left\{\sum_{i=0}^{p-1}\boldsymbol{F}(t-i)\boldsymbol{c}\big[y(t-i) - \boldsymbol{\varphi}^{\mathrm{T}}(t-i)\boldsymbol{a} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-i)\boldsymbol{c}\big]\right\}$$
$$= -\sum_{i=0}^{p-1}\big\{\boldsymbol{F}(t-i)\big[y(t-i) - \boldsymbol{\varphi}^{\mathrm{T}}(t-i)\boldsymbol{a} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-i)\boldsymbol{c}\big] - \boldsymbol{F}(t-i)\boldsymbol{c}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-i)\big\}$$
$$= \sum_{i=0}^{p-1}\big\{\boldsymbol{F}(t-i)\big[-y(t-i) + \boldsymbol{\varphi}^{\mathrm{T}}(t-i)\boldsymbol{a} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-i)\boldsymbol{c}\big] + \boldsymbol{F}(t-i)\boldsymbol{c}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-i)\big\}$$
$$= \sum_{i=0}^{p-1}\big\{\boldsymbol{F}(t-i)\big[-y(t-i) + \boldsymbol{\varphi}^{\mathrm{T}}(t-i)\boldsymbol{a} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t-i)\boldsymbol{c}\big]\big\} + \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c},t)\boldsymbol{\Psi}(\boldsymbol{b},t) \in \mathbb{R}^{n_b\times n_c}.$$

Using the Newton method and minimizing $J_2(\boldsymbol{\theta})$, we have

$$\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1) - \big\{\boldsymbol{H}\big[J_2(\hat{\boldsymbol{a}}_{t-1},\hat{\boldsymbol{b}}_{t-1},\hat{\boldsymbol{c}}_{t-1})\big]\big\}^{-1}\mathrm{grad}_\theta\big[J_2(\hat{\boldsymbol{a}}_{t-1},\hat{\boldsymbol{b}}_{t-1},\hat{\boldsymbol{c}}_{t-1})\big].$$

Thus, **the Newton recursive algorithm for the Hammerstein models** (the H-NR algorithm for short) is expressed as

$$\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1) + \boldsymbol{\Omega}^{-1}(t)\boldsymbol{\Xi}(t)\boldsymbol{E}(p,t), \tag{32}$$
$$\boldsymbol{\Omega}(t) = \frac{1}{2}\boldsymbol{H}\big[J_2(\hat{\boldsymbol{a}}_{t-1},\hat{\boldsymbol{b}}_{t-1},\hat{\boldsymbol{c}}_{t-1})\big]$$
$$= \begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\boldsymbol{\Phi}_0(p,t) & \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t) & \boldsymbol{\Phi}_0^{\mathrm{T}}(p,t)\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{t-1},t)\boldsymbol{\Phi}_0(p,t) & \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{t-1},t)\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t) & \boldsymbol{\Omega}_{23}(t) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{t-1},t)\boldsymbol{\Phi}_0(p,t) & \boldsymbol{\Omega}_{23}^{\mathrm{T}}(t) & \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{t-1},t)\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t) \end{bmatrix}, \tag{33}$$
$$\boldsymbol{\Omega}_{23}(t) = \sum_{i=0}^{p-1}\big\{\boldsymbol{F}(t-i)\big[-y(t-i) + \boldsymbol{\varphi}^{\mathrm{T}}(t-i)\hat{\boldsymbol{a}}_{t-1} + \hat{\boldsymbol{b}}_{t-1}^{\mathrm{T}}\boldsymbol{F}(t-i)\hat{\boldsymbol{c}}_{t-1}\big]\big\} + \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{t-1},t)\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t), \tag{34}$$
$$\boldsymbol{\Xi}(t) = \big[\boldsymbol{\Phi}_0(p,t),\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t),\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t)\big], \tag{35}$$
$$\boldsymbol{E}(p,t) = \boldsymbol{Y}(p,t) - \boldsymbol{\Phi}_0(p,t)\hat{\boldsymbol{a}}_{t-1} - \boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t)\hat{\boldsymbol{b}}_{t-1}, \tag{36}$$
$$\boldsymbol{Y}(p,t) = \big[y(t),y(t-1),\ldots,y(t-p+1)\big]^{\mathrm{T}}, \tag{37}$$
$$\boldsymbol{\Phi}_0(p,t) = \big[\boldsymbol{\varphi}(t),\boldsymbol{\varphi}(t-1),\ldots,\boldsymbol{\varphi}(t-p+1)\big]^{\mathrm{T}}, \tag{38}$$
$$\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t) = \big[\boldsymbol{F}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{t-1},\boldsymbol{F}^{\mathrm{T}}(t-1)\hat{\boldsymbol{b}}_{t-1},\ldots,\boldsymbol{F}^{\mathrm{T}}(t-p+1)\hat{\boldsymbol{b}}_{t-1}\big]^{\mathrm{T}}, \tag{39}$$
$$\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t) = \big[\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{t-1},\boldsymbol{F}(t-1)\hat{\boldsymbol{c}}_{t-1},\ldots,\boldsymbol{F}(t-p+1)\hat{\boldsymbol{c}}_{t-1}\big]^{\mathrm{T}}, \tag{40}$$
$$\boldsymbol{\varphi}(t) = \big[-y(t-1),-y(t-2),\ldots,-y(t-n_a)\big]^{\mathrm{T}}, \tag{41}$$

Table 1
The dimensions of the Newton recursive algorithm variables.

Item	Variables	Dimensions
1	Input variable	$u(t) \in \mathbb{R}^1$
2	Output variable	$y(t) \in \mathbb{R}^1$
3	Base functions	$f_i(u(t-j)) \in \mathbb{R}^1$, $i = 1, 2, \ldots, n_c$
4	Parameter vectors	$\boldsymbol{a} \in \mathbb{R}^{n_a}$, $\boldsymbol{b} \in \mathbb{R}^{n_b}$, $\boldsymbol{c} \in \mathbb{R}^{n_c}$
5	Parameter vector	$\boldsymbol{\theta} \in \mathbb{R}^{n_a+n_b+n_c}$
6	Output information vector	$\boldsymbol{\varphi}(t) \in \mathbb{R}^{n_a}$
7	Input information matrix	$\boldsymbol{F}(t-i) \in \mathbb{R}^{n_b \times n_c}$, $i = 0, 1, \ldots, p-1$
8	Information matrix	$\boldsymbol{\Xi}(t) \in \mathbb{R}^{n \times p}$
9	Innovation vector	$\boldsymbol{E}(p,t) \in \mathbb{R}^p$
10	Hessian matrices	$\boldsymbol{\Omega}(t) \in \mathbb{R}^{n \times n}$, $\boldsymbol{\Omega}_{23}(t) \in \mathbb{R}^{n_b \times n_c}$
11	Stacked output vector	$\boldsymbol{Y}(p,t) \in \mathbb{R}^p$
12	Output information matrix	$\boldsymbol{\Phi}_0(p,t) \in \mathbb{R}^{p \times n_a}$
13	Generalized information matrices	$\boldsymbol{\Phi}(\boldsymbol{c},t) \in \mathbb{R}^{p \times n_b}$, $\boldsymbol{\Psi}(\boldsymbol{b},t) \in \mathbb{R}^{p \times n_c}$

$$\boldsymbol{F}(t) = \begin{bmatrix} f_1(u(t-1)) & f_2(u(t-1)) & \cdots & f_{n_c}(u(t-1)) \\ f_1(u(t-2)) & f_2(u(t-2)) & \cdots & f_{n_c}(u(t-2)) \\ \vdots & \vdots & & \vdots \\ f_1(u(t-n_b)) & f_2(u(t-n_b)) & \cdots & f_{n_c}(u(t-n_b)) \end{bmatrix}, \tag{42}$$

$$\hat{\boldsymbol{c}}_t = \operatorname{sgn}\big[\hat{\boldsymbol{\theta}}_{n_a+n_b+1}(t)\big] \frac{[\hat{\boldsymbol{\theta}}(t)](n_a+n_b+1:n)}{\|[\hat{\boldsymbol{\theta}}(t)](n_a+n_b+1:n)\|}, \tag{43}$$

$$\big[\hat{\boldsymbol{\theta}}(t)\big](n_a+n_b+1:n) = \hat{\boldsymbol{c}}_t. \tag{44}$$

Since there exists the inverse matrix $\boldsymbol{\Omega}^{-1}(t)$ in (32), the stacked data length p should be sufficient large such that $\boldsymbol{\Omega}(t)$ is always a nonsingular matrix for all t.

The computation procedure of the Newton recursive algorithm for the Hammerstein models is summarized as follows:

1. Given p. To initialize: let $t = 1$, $\hat{\boldsymbol{\theta}}(0) = \begin{bmatrix} \hat{\boldsymbol{a}}_0 \\ \hat{\boldsymbol{b}}_0 \\ \hat{\boldsymbol{b}}_0 \end{bmatrix}$ be an arbitrary real vector with $\|\hat{\boldsymbol{c}}_0\| = 1$.
2. Collect the input–output data $u(t)$ and $y(t)$ and form $\boldsymbol{\varphi}(t)$ by (41), $\boldsymbol{F}(t)$ by (42) and $\boldsymbol{Y}(p,t)$ by (37).
3. Form $\boldsymbol{\Phi}_0(p,t)$ by (38), compute and form $\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{t-1},t)$ by (39) and $\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{t-1},t)$ by (40).
4. Form $\boldsymbol{\Xi}(t)$ by (35) and compute $\boldsymbol{E}(p,t)$ by (36).
5. Compute $\boldsymbol{\Omega}_{23}(t)$ by (34) and $\boldsymbol{\Omega}(t)$ by (33).
6. Compute $\hat{\boldsymbol{\theta}}(t)$ by (32).
7. Normalize $\hat{\boldsymbol{c}}_t$ by (43) and (44) with the first positive element.
8. Increase t by 1 and go to step 2.

A summary of the variables and their dimensions in the Newton recursive algorithm is shown in Table 2 for convenience. The flowchart of computing the parameter estimates $\hat{\boldsymbol{\theta}}(t)$ in the Newton recursive algorithm is shown in Fig. 3.

6. The Newton iterative algorithm

In this paper, in order to distinguish on-line from off-line calculation, we use *iterative* with index k, e.g., $\hat{\boldsymbol{\theta}}_k$ as the estimate of $\boldsymbol{\theta}$, for off-line/iterative algorithms (k is the iterative variable), and *recursive* with index t for on-line/recursive ones, e.g., $\hat{\boldsymbol{\theta}}(t)$ as the estimate of $\boldsymbol{\theta}$ (t is the time variable). We imply that a recursive algorithm can be on-line implemented, but an iterative one cannot.

The recursive estimation algorithms discussed in the preceding sections can be used for on-line identification since the recursive variable is the time t. In theory, the recursive algorithm can compute the parameter estimates for $t = \infty$. Since the gain vector $\boldsymbol{\Omega}^{-1}(t)\boldsymbol{\Xi}(t)$ in (32) in the Newton recursive algorithm does not approach zero, the parameter estimates are sensitive to noise and cannot be stationary or converge to true parameters even for the data length $t \to \infty$. Also, in practice, we can collect only a finite number of input–output measurement data. How to use the finite data to improve the parameter estimation accuracy is the goal of this section. This motivates us to develop new iterative algorithms for the Hammerstein models. The following discusses the Newton iterative algorithm for the Hammerstein systems.

Assume that the measurement data are $\{u(t), y(t)\colon t = 1, 2, \ldots, L\}$ (L represents the data length) and define the stacked output vector $\boldsymbol{Y}(L)$ and stacked matrices:

$$\boldsymbol{Y}(L) := \big[y(1), y(2), \ldots, y(L)\big] \in \mathbb{R}^L,$$

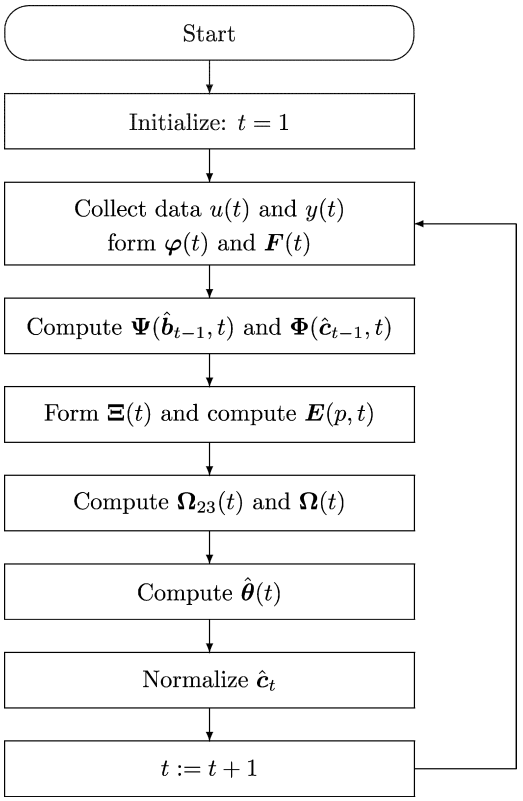

Fig. 3. The flowchart of computing $\hat{\boldsymbol{\theta}}(t)$ in the Newton recursive algorithm.

$$\boldsymbol{\Phi}_0(L) := \big[\boldsymbol{\varphi}(1), \boldsymbol{\varphi}(2), \ldots, \boldsymbol{\varphi}(L)\big]^{\mathrm{T}} \in \mathbb{R}^{L\times n_a},$$

$$\boldsymbol{\Phi}(\boldsymbol{c}) := \begin{bmatrix} \boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(1) \\ \boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(2) \\ \vdots \\ \boldsymbol{c}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}(L) \end{bmatrix} \in \mathbb{R}^{L\times n_b}, \qquad \boldsymbol{\Psi}(\boldsymbol{b}) := \begin{bmatrix} \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(1) \\ \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(2) \\ \vdots \\ \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(L) \end{bmatrix} \in \mathbb{R}^{L\times n_c}.$$

In the iterative methods, we use all measurement data from $t = 1$ to $t = L$ to form the cost function:

$$\begin{aligned} J_4(\boldsymbol{\theta}) = J_4(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) &:= \big\|\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Phi}(\boldsymbol{c})\boldsymbol{b}\big\|^2 \\ &= \big\|\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b})\boldsymbol{c}\big\|^2, \end{aligned} \tag{45}$$

which is the least squares cost functions in [10].

Compute the gradient of J_4 with respect to $\boldsymbol{\theta}$:

$$\begin{aligned} \mathrm{grad}_a\big[J_4(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\big] &= -2\boldsymbol{\Phi}_0^{\mathrm{T}}(L)\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Phi}(\boldsymbol{c})\boldsymbol{b}\big], \\ \mathrm{grad}_b\big[J_4(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\big] &= -2\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c})\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Phi}(\boldsymbol{c})\boldsymbol{b}\big], \\ \mathrm{grad}_c\big[J_4(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\big] &= -2\boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b})\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b})\boldsymbol{c}\big]. \end{aligned}$$

Thus, we have

$$\begin{aligned} \mathrm{grad}_\theta\big[J_4(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\big] &= \begin{bmatrix} \frac{\partial J_4(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\partial \boldsymbol{a}} \\ \frac{\partial J_4(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\partial \boldsymbol{b}} \\ \frac{\partial J_4(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\partial \boldsymbol{c}} \end{bmatrix} = -2\begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(L) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c}) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b}) \end{bmatrix}\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b})\boldsymbol{c}\big] \\ &= -2\begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(L) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c}) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b}) \end{bmatrix}\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Phi}(\boldsymbol{c})\boldsymbol{b}\big]. \end{aligned} \tag{46}$$

Let k be an iterative variable, $\hat{\boldsymbol{\theta}}_k := \begin{bmatrix} \hat{\boldsymbol{a}}_k \\ \hat{\boldsymbol{b}}_k \\ \hat{\boldsymbol{c}}_k \end{bmatrix}$ be the iterative estimate of $\boldsymbol{\theta} := \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \\ \boldsymbol{c} \end{bmatrix}$ at iteration k. Compute the Hessian matrix of the cost function J_4:

$$\boldsymbol{H}\big[J_4(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})\big] = 2\begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(L)\boldsymbol{\Phi}_0(L) & \boldsymbol{\Phi}_0^{\mathrm{T}}(L)\boldsymbol{\Phi}(\boldsymbol{c}) & \boldsymbol{\Phi}_0^{\mathrm{T}}(L)\boldsymbol{\Psi}(\boldsymbol{b}) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c})\boldsymbol{\Phi}_0(L) & \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c})\boldsymbol{\Phi}(\boldsymbol{c}) & \boldsymbol{M}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b})\boldsymbol{\Phi}_0(L) & \boldsymbol{M}^{\mathrm{T}}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) & \boldsymbol{\Psi}^{\mathrm{T}}(\boldsymbol{b})\boldsymbol{\Psi}(\boldsymbol{b}) \end{bmatrix} \in \mathbb{R}^{n\times n}, \tag{47}$$

where

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) &:= -\frac{\partial}{\partial \boldsymbol{c}}\big\{\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c})\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b})\boldsymbol{c}\big]\big\} \\ &= -\frac{\partial}{\partial \boldsymbol{c}}\big\{\big[\boldsymbol{F}(1)\boldsymbol{c}, \boldsymbol{F}(2)\boldsymbol{c}, \ldots, \boldsymbol{F}(L)\boldsymbol{c}\big]\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\boldsymbol{a} - \boldsymbol{\Psi}(\boldsymbol{b})\boldsymbol{c}\big]\big\} \\ &= -\frac{\partial}{\partial \boldsymbol{c}}\left\{\sum_{t=1}^{L}\boldsymbol{F}(t)\boldsymbol{c}\big[y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}\big]\right\} \\ &= -\sum_{t=1}^{L}\big\{\boldsymbol{F}(t)\big[y(t) - \boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}\big] - \boldsymbol{F}(t)\boldsymbol{c}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\big\} \\ &= \sum_{t=1}^{L}\big\{\boldsymbol{F}(t)\big[-y(t) + \boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{a} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{F}(t)\boldsymbol{c}\big]\big\} + \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{c})\boldsymbol{\Psi}(\boldsymbol{b}) \in \mathbb{R}^{n_b\times n_c}. \end{aligned}$$

Using the Newton method and minimizing J_4, we have

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} - \big\{\boldsymbol{H}\big[J_4(\hat{\boldsymbol{a}}_{k-1},\hat{\boldsymbol{b}}_{k-1},\hat{\boldsymbol{c}}_{k-1})\big]\big\}^{-1}\operatorname{grad}_{\theta}\big[J_4(\hat{\boldsymbol{a}}_{k-1},\hat{\boldsymbol{b}}_{k-1},\hat{\boldsymbol{c}}_{k-1})\big].$$

Substituting $\boldsymbol{a} = \hat{\boldsymbol{a}}_{k-1}$, $\boldsymbol{b} = \hat{\boldsymbol{b}}_{k-1}$ and $\boldsymbol{c} = \hat{\boldsymbol{c}}_{k-1}$ into the gradient $\operatorname{grad}_{\theta}[J_4(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})]$ in (46) and Hessian matrix $\boldsymbol{H}[J_4(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})]$ in (47), we can summary **the Newton iterative method for the Hammerstein models** (the H-NI algorithm for short) as follows:

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} + \boldsymbol{\Omega}_k^{-1}\begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(L) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{k-1}) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{k-1}) \end{bmatrix}\big[\boldsymbol{Y}(L) - \boldsymbol{\Phi}_0(L)\hat{\boldsymbol{a}}_{k-1} - \boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{k-1})\hat{\boldsymbol{b}}_{k-1}\big], \tag{48}$$

$$\boldsymbol{\Omega}_k = \begin{bmatrix} \boldsymbol{\Phi}_0^{\mathrm{T}}(L)\boldsymbol{\Phi}_0(L) & \boldsymbol{\Phi}_0^{\mathrm{T}}(L)\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{k-1}) & \boldsymbol{\Phi}_0^{\mathrm{T}}(L)\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{k-1}) \\ \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{k-1})\boldsymbol{\Phi}_0(L) & \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{k-1})\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{k-1}) & \boldsymbol{M}(\hat{\boldsymbol{\theta}}_{k-1}) \\ \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{k-1})\boldsymbol{\Phi}_0(L) & \boldsymbol{M}^{\mathrm{T}}(\hat{\boldsymbol{\theta}}_{k-1}) & \boldsymbol{\Psi}^{\mathrm{T}}(\hat{\boldsymbol{b}}_{k-1})\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{k-1}) \end{bmatrix}, \tag{49}$$

$$\boldsymbol{M}(\hat{\boldsymbol{\theta}}_{k-1}) = \sum_{t=1}^{L}\big\{\boldsymbol{F}(t)\big[-y(t) + \boldsymbol{\varphi}^{\mathrm{T}}(t)\hat{\boldsymbol{a}}_{k-1} + \hat{\boldsymbol{b}}_{k-1}^{\mathrm{T}}\boldsymbol{F}(t)\hat{\boldsymbol{c}}_{k-1}\big]\big\} + \boldsymbol{\Phi}^{\mathrm{T}}(\hat{\boldsymbol{c}}_{k-1})\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{k-1}), \tag{50}$$

$$\boldsymbol{Y}(L) := \big[y(1), y(2), \ldots, y(L)\big], \tag{51}$$

$$\boldsymbol{\Phi}_0(L) := \big[\boldsymbol{\varphi}(1), \boldsymbol{\varphi}(2), \ldots, \boldsymbol{\varphi}(L)\big]^{\mathrm{T}}, \tag{52}$$

$$\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{k-1}) = \big[\boldsymbol{F}^{\mathrm{T}}(1)\hat{\boldsymbol{b}}_{k-1}, \boldsymbol{F}^{\mathrm{T}}(2)\hat{\boldsymbol{b}}_{k-1}, \ldots, \boldsymbol{F}^{\mathrm{T}}(L)\hat{\boldsymbol{b}}_{k-1}\big]^{\mathrm{T}}, \tag{53}$$

$$\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{k-1}) = \big[\boldsymbol{F}(1)\hat{\boldsymbol{c}}_{k-1}, \boldsymbol{F}(2)\hat{\boldsymbol{c}}_{k-1}, \ldots, \boldsymbol{F}(L)\hat{\boldsymbol{c}}_{k-1}\big]^{\mathrm{T}}, \tag{54}$$

$$\boldsymbol{\varphi}(t) = \big[-y(t-1), -y(t-2), \ldots, -y(t-n_a)\big]^{\mathrm{T}}, \quad t = 1, 2, \ldots, L, \tag{55}$$

$$\boldsymbol{F}(t) = \begin{bmatrix} f_1(u(t-1)) & f_2(u(t-1)) & \cdots & f_{n_c}(u(t-1)) \\ f_1(u(t-2)) & f_2(u(t-2)) & \cdots & f_{n_c}(u(t-2)) \\ \vdots & \vdots & & \vdots \\ f_1(u(t-n_b)) & f_2(u(t-n_b)) & \cdots & f_{n_c}(u(t-n_b)) \end{bmatrix}, \tag{56}$$

$$\hat{\boldsymbol{c}}_k = \operatorname{sgn}\big[\hat{\boldsymbol{\theta}}_k(n_a+n_b+1)\big]\frac{\hat{\boldsymbol{\theta}}_k(n_a+n_b+1:n)}{\|\hat{\boldsymbol{\theta}}_k(n_a+n_b+1:n)\|}, \tag{57}$$

$$\hat{\boldsymbol{\theta}}_k(n_a+n_b+1:n) = \hat{\boldsymbol{c}}_k. \tag{58}$$

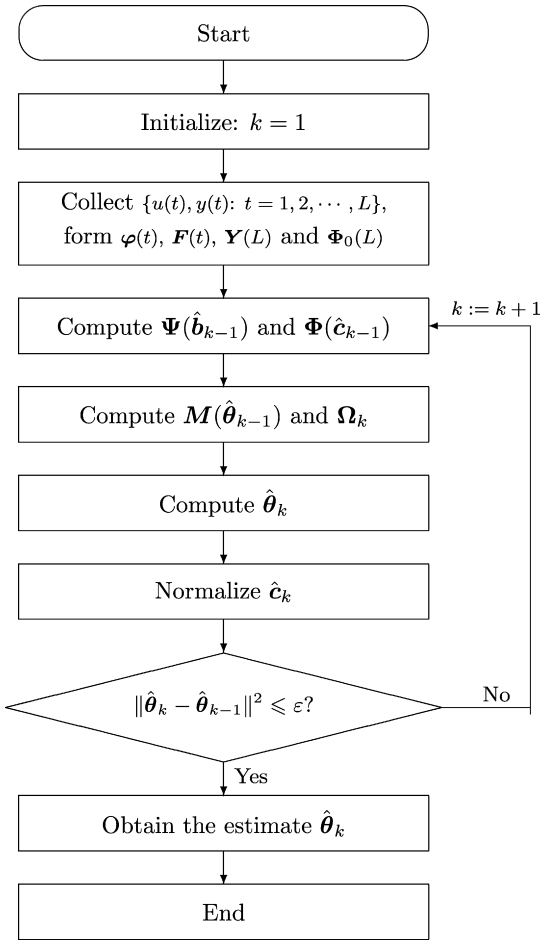

Fig. 4. The flowchart of computing the iterative estimate $\hat{\boldsymbol{\theta}}_k$.

The H-NI algorithm uses the fixed data batch with the data length L. When the data length $L \to \infty$, Liu and Bai reported an iterative identification algorithm based on the cost function J_4 in (45) and showed that the iterative parameter estimation errors for the Hammerstein systems converge to zero [10].

The computation procedure of the Newton recursive algorithm for the Hammerstein models is summarized as follows:

1. To initialize: let $k = 1$, $\hat{\boldsymbol{\theta}}(0) = \begin{bmatrix} \hat{\boldsymbol{a}}_0 \\ \hat{\boldsymbol{b}}_0 \\ \hat{\boldsymbol{b}}_0 \end{bmatrix}$ be an arbitrary real vector with $\|\hat{\boldsymbol{c}}_0\| = 1$.
2. Collect the input–output data $\{u(t), y(t)\colon t = 1, 2, \ldots, L\}$ and form $\boldsymbol{\varphi}(t)$ by (55), $\boldsymbol{F}(t)$ by (56), $\boldsymbol{Y}(L)$ by (51) and $\boldsymbol{\Phi}_0(L)$ by (52).
3. Form and compute $\boldsymbol{\Psi}(\hat{\boldsymbol{b}}_{k-1})$ by (53) and $\boldsymbol{\Phi}(\hat{\boldsymbol{c}}_{k-1})$ by (54).
4. Compute $\boldsymbol{M}(\hat{\boldsymbol{\theta}}_{k-1})$ by (50) and $\boldsymbol{\Omega}_k$ by (49).
5. Compute $\hat{\boldsymbol{\theta}}_k$ by (48).
6. Normalize $\hat{\boldsymbol{c}}_k$ by (57) and (58) with the first positive element.
7. Compare $(\hat{\boldsymbol{a}}_k, \hat{\boldsymbol{b}}_k, \hat{\boldsymbol{c}}_k)$ with $(\hat{\boldsymbol{a}}_{k-1}, \hat{\boldsymbol{b}}_{k-1}, \hat{\boldsymbol{c}}_{k-1})$: if they are sufficiently close, or for some pre-set small $\varepsilon > 0$, if

$$\|\hat{\boldsymbol{a}}_k - \hat{\boldsymbol{a}}_{k-1}\|^2 + \|\hat{\boldsymbol{b}}_k - \hat{\boldsymbol{b}}_{k-1}\|^2 + \|\hat{\boldsymbol{c}}_k - \hat{\boldsymbol{c}}_{k-1}\|^2 \leqslant \varepsilon,$$

then terminate the procedure and obtain the iterative times k and estimates $(\hat{\boldsymbol{a}}_k, \hat{\boldsymbol{b}}_k, \hat{\boldsymbol{c}}_k)$; otherwise, increase k by 1 and go to step 3.

The flowchart of computing the parameter estimate $\hat{\boldsymbol{\theta}}_k$ is shown in Fig. 4.

A summary of the variables and their dimensions in the Newton recursive algorithm is shown in Table 2 for convenience.

Table 2
The dimensions of the Newton recursive algorithm variables.

Item	Variables	Dimensions
1	Input and output variables	$u(t) \in \mathbb{R}^1$, $y(t) \in \mathbb{R}^1$
2	Base functions	$f_i(u(t-j)) \in \mathbb{R}^1$, $i = 1, 2, \ldots, n_c$
3	Parameter vectors	$\boldsymbol{a} \in \mathbb{R}^{n_a}$, $\boldsymbol{b} \in \mathbb{R}^{n_b}$, $\boldsymbol{c} \in \mathbb{R}^{n_c}$
4	Parameter vector	$\boldsymbol{\theta} \in \mathbb{R}^{n_a+n_b+n_c}$
5	Output information vector	$\boldsymbol{\varphi}(t) \in \mathbb{R}^{n_a}$, $t = 1, 2, \ldots, L$
6	Input information matrix	$\boldsymbol{F}(t) \in \mathbb{R}^{n_b \times n_c}$, $t = 1, 2, \ldots, L$
7	Hessian matrices	$\boldsymbol{\Omega}_k \in \mathbb{R}^{n \times n}$, $\boldsymbol{M}(\theta) \in \mathbb{R}^{n_b \times n_c}$
8	Stacked output vector	$\boldsymbol{Y}(L) \in \mathbb{R}^L$
9	Output information matrix	$\boldsymbol{\Phi}_0(L) \in \mathbb{R}^{L \times n_a}$
10	Generalized information matrices	$\boldsymbol{\Phi}(\boldsymbol{c}) \in \mathbb{R}^{L \times n_b}$, $\boldsymbol{\Psi}(\boldsymbol{b}) \in \mathbb{R}^{L \times n_c}$

It is worth pointing out that the Newton recursive or Newton iterative identification algorithms exist only for nonlinear systems. **The Newton recursive or Newton iterative identification algorithms for linear systems reduce to the least squares estimates**. In fact, for example, consider the linear regression or pseudo-linear regression models [35]:

$$y(t) = \phi^{\mathrm{T}}(t)\boldsymbol{\Theta} + v(t),$$

which includes the over-parameterization identification model in (6), where $\phi(t) \in \mathbb{R}^n$ is the information vector and $\boldsymbol{\Theta} \in \mathbb{R}^n$ is the parameter vector. Define

$$\boldsymbol{Y}(p,t) := \begin{bmatrix} y(t) \\ y(t-1) \\ \vdots \\ y(t-p+1) \end{bmatrix} \in \mathbb{R}^p, \qquad \boldsymbol{\Phi}(p,t) := \begin{bmatrix} \boldsymbol{\phi}^{\mathrm{T}}(t) \\ \boldsymbol{\phi}^{\mathrm{T}}(t-1) \\ \vdots \\ \boldsymbol{\phi}^{\mathrm{T}}(t-p+1) \end{bmatrix} \in \mathbb{R}^{p \times n}, \tag{59}$$

and the cost function:

$$J_5(\boldsymbol{\Theta}) := \big\| \boldsymbol{Y}(p,t) - \boldsymbol{\Phi}(p,t)\boldsymbol{\Theta} \big\|^2.$$

Compute the gradient of J_5 with respect to $\boldsymbol{\Theta}$:

$$\mathrm{grad}_{\Theta}\big[J_5(\boldsymbol{\Theta})\big] = -2\boldsymbol{\Phi}^{\mathrm{T}}(p,t)\big[\boldsymbol{Y}(p,t) - \boldsymbol{\Phi}(p,t)\boldsymbol{\Theta}\big].$$

Compute the Hessian matrix of the cost function J_5:

$$\boldsymbol{H}\big[J_5(\boldsymbol{\Theta}\big] = \frac{\partial\, \mathrm{grad}_{\Theta}[J_5(\boldsymbol{\Theta})]}{\partial \boldsymbol{\Theta}^{\mathrm{T}}} = 2\boldsymbol{\Phi}^{\mathrm{T}}(p,t)\boldsymbol{\Phi}(p,t) \in \mathbb{R}^{n \times n}.$$

Using the Newton method and minimizing $J_5(\boldsymbol{\Theta})$, we have

$$\begin{aligned} \hat{\boldsymbol{\Theta}}(t) &= \hat{\boldsymbol{\Theta}}(t-1) - \big\{\boldsymbol{H}\big[J_5\big(\hat{\boldsymbol{\Theta}}(t-1)\big)\big]\big\}^{-1} \mathrm{grad}_{\Theta}\big[J_5\big(\hat{\boldsymbol{\Theta}}(t-1)\big)\big] \\ &= \hat{\boldsymbol{\Theta}}(t-1) + \big[2\boldsymbol{\Phi}^{\mathrm{T}}(p,t)\boldsymbol{\Phi}(p,t)\big]^{-1} 2\boldsymbol{\Phi}^{\mathrm{T}}(p,t)\big[\boldsymbol{Y}(p,t) - \boldsymbol{\Phi}(p,t)\hat{\boldsymbol{\Theta}}(t-1)\big] \\ &= \big[\boldsymbol{\Phi}^{\mathrm{T}}(p,t)\boldsymbol{\Phi}(p,t)\big]^{-1} \boldsymbol{\Phi}^{\mathrm{T}}(p,t)\boldsymbol{Y}(p,t). \end{aligned}$$

This is the least squares estimate over the data window with the length p.

7. Example and comparisons

7.1. Example

Consider the following Hammerstein nonlinear system:

$$\begin{aligned} &A(z)y(t) = B(z)\bar{u}(t) + v(t), \\ &A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} = 1 - 1.60z^{-1} + 0.80z^{-2}, \\ &B(z) = b_1 z^{-1} + b_2 z^{-2} = 0.85z^{-1} + 0.65z^{-2}, \\ &\bar{u}(t) = f\big(u(t)\big) = c_1 u(t) + c_2 u^2(t) + c_3 u^3(t) \\ &\qquad\qquad = 0.90u(t) + 0.40u^2(t) + 0.1721u^3(t), \\ &\boldsymbol{\theta} = [a_1, a_2, b_1, b_2, c_1, c_2, c_3]^{\mathrm{T}}. \end{aligned}$$

Table 3
The H-Proj estimates and errors ($\sigma^2 = 0$).

t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
100	−1.60659	0.79906	0.89582	0.67547	0.90996	0.38803	0.14632	2.65142
200	−1.59933	0.80032	0.87823	0.67173	0.91047	0.38501	0.15107	1.97937
500	−1.60080	0.80091	0.85866	0.65674	0.90379	0.39550	0.16352	0.68455
1000	−1.59983	0.79993	0.85089	0.65068	0.90035	0.39960	0.17232	0.06630
2000	−1.60000	0.80001	0.85004	0.65003	0.90002	0.39998	0.17316	0.00339
3000	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	0.00010
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 4
The H-S-Proj estimates and errors ($\sigma^2 = 0$).

t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
100	−1.60487	0.79757	0.89544	0.67283	0.91046	0.38675	0.14658	2.59890
200	−1.59916	0.80014	0.87772	0.67138	0.91030	0.38527	0.15138	1.94662
500	−1.60087	0.80106	0.86019	0.65783	0.90437	0.39484	0.16194	0.79711
1000	−1.59959	0.79985	0.85216	0.65165	0.90082	0.39908	0.17109	0.15959
2000	−1.60002	0.80004	0.85014	0.65011	0.90006	0.39993	0.17305	0.01127
3000	−1.60000	0.80000	0.85001	0.65000	0.90000	0.40000	0.17321	0.00036
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 5
The H-Proj estimates and errors ($\sigma^2 = 0.30^2$).

t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
100	−1.57472	0.78616	0.84741	0.69383	0.91839	0.37073	0.13827	3.11146
100	−1.57285	0.78779	0.84726	0.69501	0.92050	0.36737	0.13311	3.35408
200	−1.55049	0.71192	0.87611	0.64823	0.88523	0.42801	0.18215	4.73358
500	−1.57536	0.75328	0.83216	0.57191	0.84707	0.50699	0.15946	6.65094
1000	−1.61803	0.78857	0.78179	0.61459	0.84253	0.47881	0.24675	6.31018
2000	−1.59037	0.77370	0.87921	0.73427	0.92341	0.37933	−0.05846	10.89000
3000	−1.60993	0.81163	0.79425	0.71993	0.89882	0.40472	0.16832	3.93593
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

For this example system, $\|\boldsymbol{c}\| = 1$ and

$$\boldsymbol{F}(t) = \begin{bmatrix} f_1(u(t-1)) & f_2(u(t-1)) & f_3(u(t-1)) \\ f_1(u(t-2)) & f_2(u(t-2)) & f_3(u(t-2)) \end{bmatrix} = \begin{bmatrix} u(t-1) & u^2(u(t-1)) & u^3(u(t-1)) \\ u(t-2) & u^2(u(t-2)) & u^3(u(t-2)) \end{bmatrix}.$$

In simulation, the input $\{u(t)\}$ is taken as a persistent excitation signal sequence with zero mean and unit variance $\sigma_u^2 = 1.00^2$. For the deterministic case, we set $v(t) \equiv 0$ or $\sigma^2 = 0$; for the stochastic case, $\{v(t)\}$ is taken as a white noise sequence with zero mean and variance $\sigma^2 = 0.30^2$ and is independent of $\{u(t)\}$, the corresponding noise-to-signal ratio is $\delta_{\text{ns}} = 16.16\%$, where the noise-to-signal ratio δ_{ns} is defined by the square root of the ratio of the variances of $w(t)$ and $x(t)$ in Fig. 1, i.e.,

$$\delta_{\text{ns}} = \sqrt{\frac{\text{var}[w(t)]}{\text{var}[x(t)]}} \times 100\%,$$

$$w(t) = \frac{1}{A(z)} v(t), \qquad x(t) = \frac{B(z)}{A(z)} \bar{u}(t).$$

We use the proposed algorithms to identify the parameters of this Hammerstein system. The results are as follows:

1. The parameter estimates and their errors of the projection (H-Proj) algorithm in (14)–(22) and simplified projection (H-S-Proj) algorithm in (17)–(23) are shown in Tables 3–4 for $\sigma^2 = 0$ and Tables 5–6 for $\sigma^2 = 0.30^2$; the parameter estimation errors $\delta := \|\hat{\boldsymbol{\theta}}(t) - \boldsymbol{\theta}\|/\|\boldsymbol{\theta}\|$ versus t are shown in Fig. 5 for $\sigma^2 = 0$ and Fig. 6 for $\sigma^2 = 0.30^2$.
2. The parameter estimates and their errors of the stochastic gradient (H-SG) and forgetting factor stochastic gradient (H-FF-SG) algorithms in (24), (25) and (17)–(22) are shown in Tables 7–8 for $\sigma^2 = 0$ and Tables 9–10 for $\sigma^2 = 0.30^2$; the parameter estimation errors versus t are shown in Fig. 7 for $\sigma^2 = 0$ and Fig. 8 for $\sigma^2 = 0.30^2$, where the forgetting factors $\lambda = 1$, $\lambda = 0.95$ and $\lambda = 0.85$.

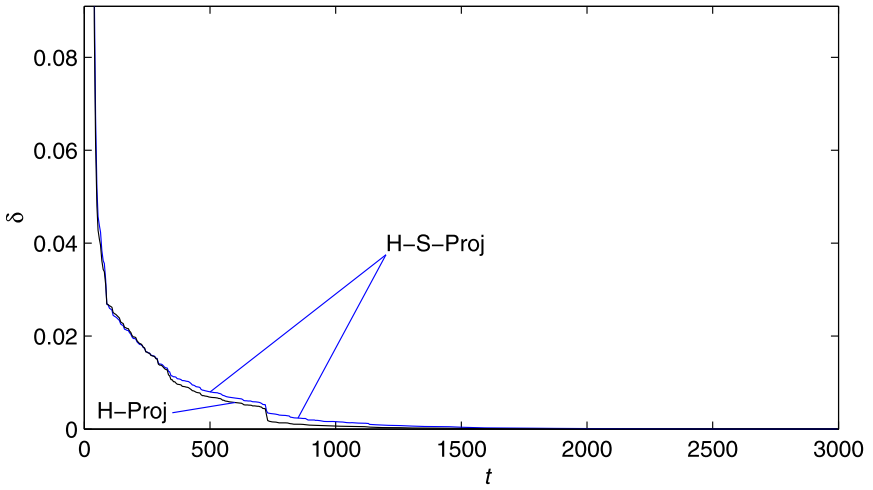

Fig. 5. The H-Proj and H-S-Proj estimation errors δ versus t ($\sigma^2 = 0.00^2$).

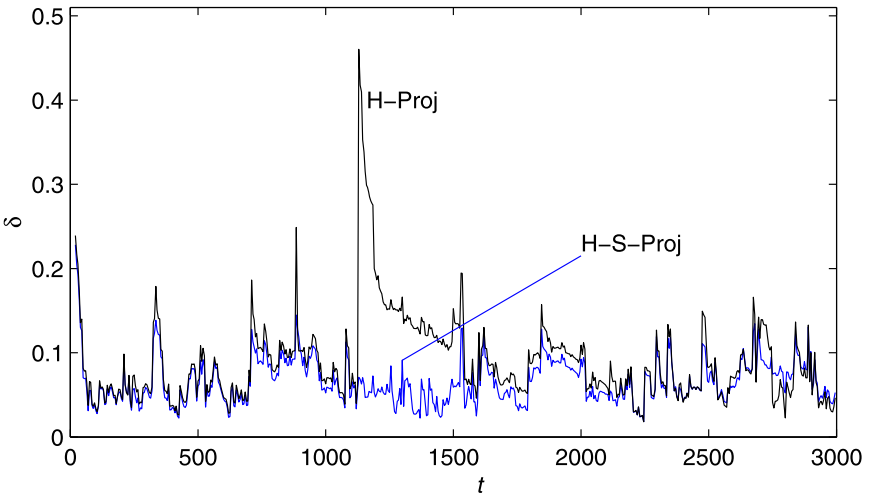

Fig. 6. The H-Proj and H-S-Proj estimation errors δ versus t ($\sigma^2 = 0.30^2$).

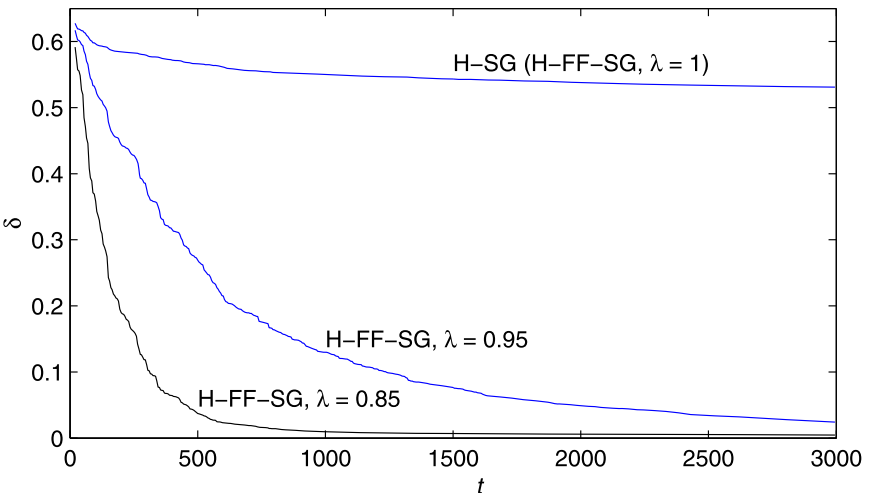

Fig. 7. The H-SG and H-FF-SG estimation errors δ versus t ($\sigma^2 = 0$).

Table 6
The H-S-Proj estimates and errors ($\sigma^2 = 0.30^2$).

t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
100	−1.57472	0.78616	0.84741	0.69383	0.91839	0.37073	0.13827	3.11146
200	−1.55435	0.71647	0.87763	0.65428	0.88527	0.42889	0.17987	4.52391
500	−1.57586	0.75543	0.84165	0.57956	0.85430	0.49692	0.15248	6.04199
1000	−1.61388	0.78741	0.79041	0.62007	0.85436	0.46453	0.23297	5.22921
2000	−1.59035	0.77184	0.85572	0.70857	0.92107	0.38836	−0.02850	9.23804
3000	−1.60852	0.81111	0.83146	0.74900	0.90913	0.39224	0.14013	4.65430
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 7
The H-SG estimates and errors ($\sigma^2 = 0$).

t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
100	−0.85619	−0.00062	0.84473	0.09727	0.97691	−0.12738	−0.17155	59.66361
200	−0.88166	0.01539	0.84446	0.10208	0.97867	−0.12079	−0.16619	58.42434
500	−0.90614	0.05135	0.84641	0.10987	0.98097	−0.11565	−0.15598	56.63592
1000	−0.93067	0.08199	0.84807	0.11600	0.98315	−0.10960	−0.14627	55.00875
2000	−0.95215	0.10133	0.84948	0.12105	0.98474	−0.10456	−0.13912	53.79773
3000	−0.96557	0.11145	0.85039	0.12420	0.98567	−0.10169	−0.13455	53.09535
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 8
The H-FF-SG estimates and errors with $\lambda = 0.95$ and $\lambda = 0.85$ ($\sigma^2 = 0$).

λ	t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
0.95	100	−0.94900	0.12391	0.85106	0.12877	0.98872	−0.09089	−0.11902	52.74182
	200	−1.13275	0.20104	0.85265	0.17888	0.99686	−0.02995	−0.07331	44.40758
	500	−1.35036	0.53092	0.88597	0.31534	0.99673	0.05966	0.05444	26.91878
	1000	−1.54348	0.72502	0.90514	0.46236	0.96692	0.20528	0.15144	12.97127
	2000	−1.61191	0.80208	0.89422	0.57415	0.92721	0.33544	0.16660	4.89490
	3000	−1.61098	0.80726	0.88073	0.61604	0.91199	0.37689	0.16196	2.39861
0.85	100	−1.17768	0.37337	0.87662	0.24837	0.99911	0.03859	0.01685	35.85863
	200	−1.43160	0.55920	0.87631	0.42516	0.97608	0.18662	0.11150	18.98828
	500	−1.58795	0.77773	0.88134	0.60648	0.92342	0.34329	0.17163	3.69170
	1000	−1.60219	0.80160	0.86637	0.64667	0.90596	0.39034	0.16394	0.96832
	2000	−1.60000	0.80033	0.85892	0.65547	0.90289	0.39666	0.16571	0.58891
	3000	−1.59978	0.79981	0.85665	0.65487	0.90239	0.39706	0.16746	0.46484
True values		−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 9
The H-SG estimates and errors ($\sigma^2 = 0.30^2$).

t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
100	−1.01623	0.16329	0.74881	0.09163	0.94613	−0.22056	−0.23706	55.11711
200	−1.03787	0.17630	0.74825	0.09343	0.94948	−0.21347	−0.23005	54.13719
500	−1.06032	0.20608	0.74968	0.09799	0.95359	−0.20711	−0.21855	52.68644
1000	−1.08084	0.23175	0.75045	0.10135	0.95706	−0.20098	−0.20889	51.43020
2000	−1.09987	0.24605	0.75128	0.10430	0.95969	−0.19594	−0.20151	50.51006
3000	−1.11187	0.25447	0.75191	0.10631	0.96126	−0.19307	−0.19676	49.94752
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

3. The parameter estimates and their errors of the Newton recursive (H-NR) algorithm in (32)–(44) are shown in Table 11 with $p = 50$ and $p = 80$ for $\sigma^2 = 0$ and Table 12 with $p = 50$, $p = 160$ and $p = 300$ for $\sigma^2 = 0.30^2$; the parameter estimation errors versus t are shown in Fig. 9 for $\sigma^2 = 0$ and Fig. 10 for $\sigma^2 = 0.30^2$.
4. The parameter estimates and their errors of the Newton iterative (H-NI) algorithm in (48)–(58) are shown in Table 13 for $\sigma^2 = 0$ and Table 14 for $\sigma^2 = 0.30^2$; the parameter estimation errors versus t are shown in Fig. 11 for $\sigma^2 = 0$ and Fig. 12 for $\sigma^2 = 0.30^2$, where the data length $L = 1000$.

7.2. Comparisons

For the proposed identification algorithms for the Hammerstein systems, we have the following conclusions:

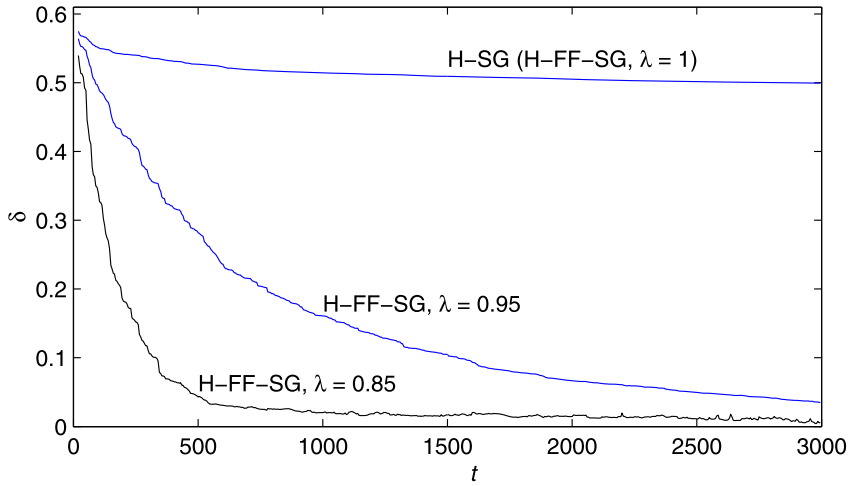

Fig. 8. The H-SG and H-FF-SG estimation errors δ versus t ($\sigma^2 = 0.30^2$).

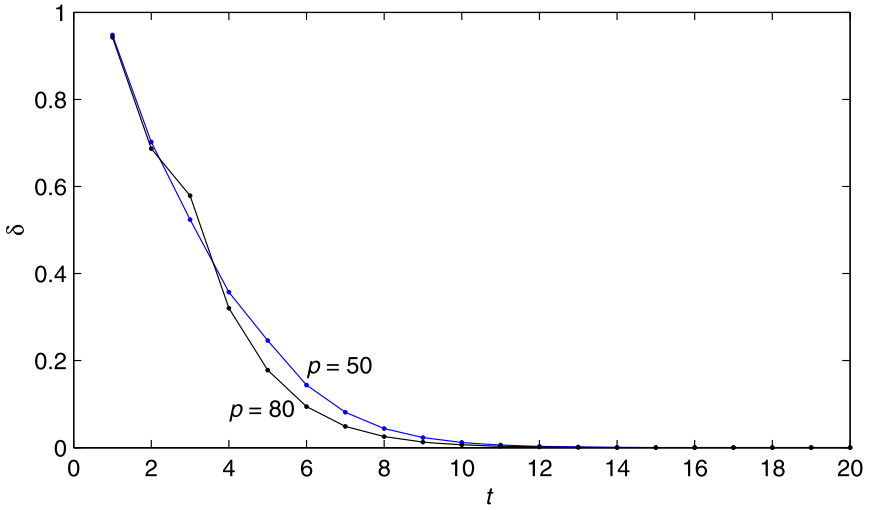

Fig. 9. The H-NR estimation errors δ versus t with $p = 50$ and $p = 80$ ($\sigma^2 = 0$).

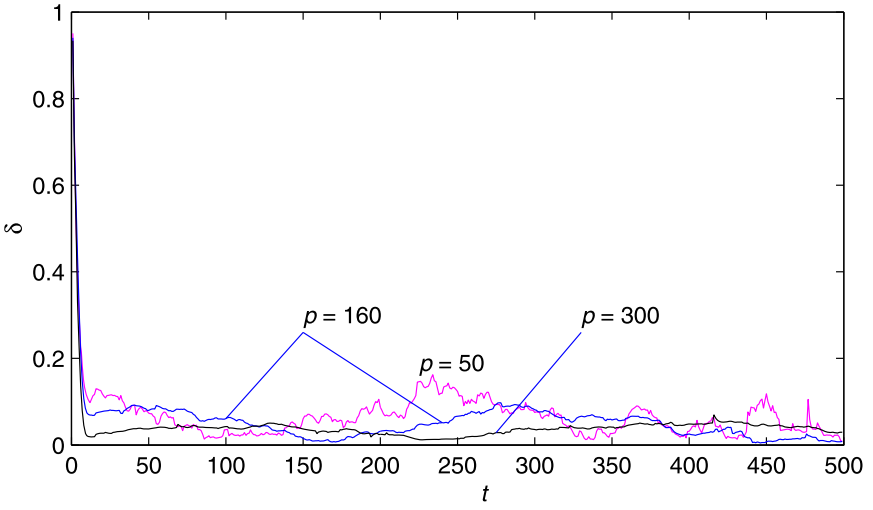

Fig. 10. The H-NR estimation errors δ versus t with $p = 50$, $p = 160$ and $p = 300$ ($\sigma^2 = 0.30^2$).

Table 10
The H-FF-SG estimates and errors with $\lambda = 0.95$ and $\lambda = 0.85$ ($\sigma^2 = 0.30^2$).

λ	t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
0.95	100	−1.08903	0.27277	0.75066	0.10982	0.96916	−0.18043	−0.16784	49.29273
	200	−1.25771	0.32753	0.75280	0.14281	0.98738	−0.11286	−0.11111	42.34862
	500	−1.43164	0.61164	0.79491	0.25166	0.99974	−0.00763	0.02156	28.18609
	1000	−1.58414	0.76468	0.83580	0.38961	0.97809	0.15105	0.14328	16.09548
	2000	−1.62704	0.81664	0.85203	0.52637	0.92911	0.31977	0.18578	6.66439
	3000	−1.61449	0.81757	0.84984	0.58222	0.90970	0.36983	0.18889	3.45009
0.85	100	−1.27605	0.48861	0.78529	0.22202	0.99892	−0.03972	0.02399	33.91136
	200	−1.49710	0.61893	0.80498	0.39841	0.97880	0.15107	0.13828	18.25374
	500	−1.59794	0.78044	0.84224	0.56848	0.91637	0.35298	0.18888	4.28539
	1000	−1.60068	0.81565	0.83392	0.61274	0.89841	0.39580	0.19025	2.02998
	2000	−1.60385	0.81108	0.82618	0.64734	0.89128	0.41609	0.18023	1.43203
	3000	−1.58810	0.80645	0.83955	0.65724	0.90156	0.39675	0.17255	0.81886
True values		−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 11
The H-NR estimates and errors with $p = 50$ and $p = 80$ ($\sigma^2 = 0$).

p	t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
50	1	−1.91450	1.04092	0.76484	−1.47767	0.98398	0.16922	−0.05609	94.80495
	2	−1.69888	0.85952	1.44206	−0.85353	0.92434	0.37327	0.07915	70.20776
	5	−1.62744	0.82781	1.41488	0.66649	0.88682	0.40555	0.22154	24.59888
	10	−1.59994	0.79999	0.87168	0.66660	0.90001	0.40001	0.17314	1.18107
	20	−1.60000	0.80000	0.85002	0.65002	0.90000	0.40000	0.17321	0.00123
80	1	−1.86580	0.97694	0.67220	−1.47184	0.98793	0.15147	−0.03245	94.24593
	2	−1.69885	0.85200	1.24183	−0.86383	0.97853	0.20276	0.03706	68.68114
	5	−1.60903	0.80825	1.24658	0.74241	0.90454	0.36569	0.21927	17.79645
	10	−1.59998	0.80000	0.86206	0.65923	0.90000	0.40002	0.17318	0.65683
	20	−1.60000	0.80000	0.85001	0.65001	0.90000	0.40000	0.17321	0.00069
True values		−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 12
The H-NR estimates and errors with $p = 50$, $p = 160$ and $p = 300$ ($\sigma^2 = 0.30^2$).

p	t	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
50	1	−1.88808	1.02512	0.74419	−1.48653	0.98585	0.15653	−0.05995	95.06669
	2	−1.67399	0.84033	1.48064	−0.85472	0.89315	0.43640	0.10878	70.73733
	5	−1.58825	0.79571	1.54990	0.77119	0.91676	0.38268	0.11452	30.85118
	10	−1.57251	0.77821	0.99552	0.83430	0.92389	0.37570	0.07265	11.24983
	20	−1.56016	0.76888	0.97438	0.85491	0.92902	0.36480	0.06196	11.80371
	50	−1.55078	0.75844	0.90937	0.76998	0.92307	0.37201	0.09770	7.37676
	100	−1.60668	0.80794	0.89717	0.68584	0.90866	0.40025	0.11888	3.52563
	500	−1.59176	0.79816	0.84390	0.65727	0.90747	0.38666	0.16430	0.94237
160	1	−1.88856	0.99916	0.59192	−1.47366	0.95869	0.26542	0.10229	94.03178
	2	−1.72763	0.87349	1.29570	−0.87071	0.80173	0.59767	−0.00576	69.92771
	5	−1.60732	0.81414	1.57616	0.67654	0.92432	0.35689	0.13516	31.55357
	10	−1.58063	0.79012	0.94663	0.75962	0.92203	0.37599	0.09212	7.42451
	20	−1.57664	0.78710	0.94079	0.76079	0.92624	0.36850	0.07930	7.70471
	50	−1.56236	0.76921	0.93737	0.76695	0.92917	0.36005	0.08368	7.99198
	100	−1.57965	0.78578	0.92778	0.73435	0.91991	0.37899	0.10068	6.09845
	500	−1.60561	0.80812	0.84349	0.64275	0.89806	0.39991	0.18320	0.74415
300	1	−1.86854	1.00687	0.58035	−1.45708	0.90542	0.29256	0.30762	93.34303
	2	−1.75379	0.90879	1.02234	−0.86021	0.78465	0.54553	0.29448	66.94053
	5	−1.61243	0.81475	1.36966	0.58305	0.88228	0.42409	0.20427	22.75519
	10	−1.58332	0.78703	0.86125	0.69242	0.90274	0.39778	0.16383	2.15091
	20	−1.58007	0.78590	0.86710	0.69846	0.90759	0.39311	0.14745	2.73729
	50	−1.57682	0.78324	0.87573	0.70989	0.91398	0.38428	0.13024	3.70974
	100	−1.58317	0.78782	0.88937	0.71216	0.91815	0.37804	0.11871	4.24385
	500	−1.60207	0.80497	0.82507	0.61207	0.87907	0.43061	0.20449	2.88294
True values		−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 13
The H-NI estimates and errors ($L = 1000$, $\sigma^2 = 0$).

k	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
1	−1.91215	1.04646	0.55697	−1.49146	0.90408	0.30274	0.30163	95.31471
2	−1.77048	0.92592	0.96584	−0.92865	0.75104	0.54488	0.37289	70.19325
3	−1.72342	0.90522	1.22313	−0.45300	0.56956	0.75850	0.31666	55.39937
4	−1.67366	0.87207	1.35977	0.04396	0.89439	0.37607	0.24216	34.68769
5	−1.62910	0.82632	1.33385	0.44301	0.88308	0.42522	0.19842	22.88990
6	−1.60343	0.80320	1.16896	0.68815	0.90476	0.38560	0.18086	13.91557
7	−1.59852	0.79910	1.01550	0.73008	0.90182	0.39488	0.17546	7.95708
8	−1.59877	0.79919	0.93130	0.70642	0.90030	0.39917	0.17358	4.28129
9	−1.59952	0.79968	0.89104	0.68126	0.90001	0.39998	0.17320	2.23183
10	−1.59985	0.79990	0.87094	0.66612	0.89999	0.40002	0.17319	1.14331
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

Table 14
The H-NI estimates and errors ($L = 1000$, $\sigma^2 = 0.30^2$).

k	a_1	a_2	b_1	b_2	c_1	c_2	c_3	δ (%)
1	−1.90086	1.03876	0.55256	−1.48858	0.91002	0.29643	0.28983	95.09906
2	−1.76320	0.92158	0.96508	−0.92744	0.76782	0.53189	0.35713	69.90109
3	−1.71663	0.90095	1.23628	−0.44921	0.56308	0.76730	0.30692	55.51995
4	−1.66746	0.86913	1.36910	0.05724	0.89049	0.38285	0.24585	34.49005
5	−1.62469	0.82486	1.32363	0.45058	0.87560	0.43652	0.20683	22.40793
6	−1.60067	0.80348	1.15089	0.68333	0.89530	0.40267	0.19053	13.11869
7	−1.59613	0.79972	0.99745	0.72109	0.89193	0.41231	0.18562	7.13130
8	−1.59644	0.79986	0.91366	0.69670	0.89009	0.41698	0.18404	3.55367
9	−1.59719	0.80035	0.87333	0.67133	0.88972	0.41790	0.18376	1.70001
10	−1.59753	0.80057	0.85312	0.65603	0.88969	0.41795	0.18377	1.05279
True values	−1.60000	0.80000	0.85000	0.65000	0.90000	0.40000	0.17321	

1. For the deterministic Hammerstein systems ($v(t) \equiv 0$ or $\sigma^2 = 0$), the parameter estimates given by the projection and simplified projection algorithms can fast converge to the true parameters and their estimation accuracy is very close – see Tables 3 and 4 and Fig. 5 with $\sigma^2 = 0$.
2. For the stochastic Hammerstein systems ($\sigma^2 \neq 0$), the projection and simplified projection algorithms are very sensitive to noise, the variances of their estimation errors are large and the parameter estimation errors cannot converge to zero – see Tables 5 and 6 and Fig. 6 with $\sigma^2 = 0.30^2$, because the gains of these two algorithms do not approach to zero. The simplified projection algorithm has less computational load than the projection algorithm. The simplified projection algorithm seems to have more accurate parameter estimates than the projection algorithm from Fig. 6.
3. For the deterministic or stochastic Hammerstein systems, the H-SG algorithm has very slow convergence rate just like the stochastic gradient algorithm of linear systems in [85] – see Table 7 with $\sigma^2 = 0$ and Table 9 with $\sigma^2 = 0.30^2$, and the top error curves in Figs. 7 and 8.
   Introducing the forgetting factor ($0 < \lambda < 1$) in the H-SG algorithm, the resulting H-FF-SG can speed up the convergence rate and improve the accuracy of the parameter estimates – see Table 8 with $\sigma^2 = 0$ and Table 10 with $\sigma^2 = 0.30^2$, and the error curves in Figs. 7 and 8 with $\lambda = 0.95$ and $\lambda = 0.85$, and the small λ leads to fast convergence rate and highly accurate parameter estimates.
   Therefore, it is very important to introduce the forgetting factor in the H-SG algorithm.
4. From Figs. 7 and 8, it is clear that the rate of change of the parameter estimates (or the estimation error) becomes more stationary with the increase of λ for the H-FF-SG algorithm, but the estimation errors are getting larger. In other words, if we decrease the forgetting factor λ, the convergence rate of the parameter estimation is faster initially, but the variance of the estimation error becomes larger if we further decrease λ. Therefore, a tradeoff is to choose a smaller forgetting factor at the initial period of the operation, and then to let the forgetting factor gradually increase with t and finally approach unity so that more accurate parameter estimates are obtained. For example, in the bottom curves in Figs. 7 and 8, we can obtain fast convergence rate as well as acceptable stationarity in the estimation errors as long as we choose an appropriate forgetting factor, the estimation errors are becoming smaller (in general) as t increases.
5. For the deterministic Hammerstein systems ($v(t) \equiv 0$ or $\sigma^2 = 0$), the parameter estimates given by the Newton recursive (H-NR) algorithm can faster converge to the true parameters than the projection and simplified projection algorithms and has faster tracking performances – see Table 11 and Fig. 9 with $\sigma^2 = 0$.
6. For the stochastic Hammerstein systems ($\sigma^2 \neq 0$), the fluctuation of the parameter estimation given by the H-NR algorithm is large, especially for the small stacked data length p, and its estimation errors cannot converge to zero even if the data length t approaches infinity. The reason is that the gain of the H-NR algorithm does not approach zero. However, as the stacked data length p increases, the parameter estimates are getting more stationary – see Table 12 and Fig. 10 with $p = 50$, $p = 160$ and $p = 300$.

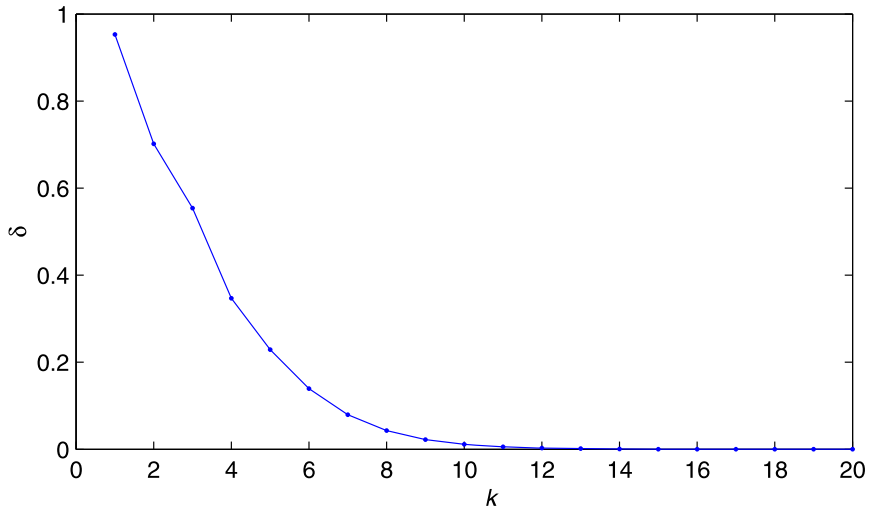

Fig. 11. The H-NI estimation errors δ versus k ($L = 1000$, $\sigma^2 = 0$).

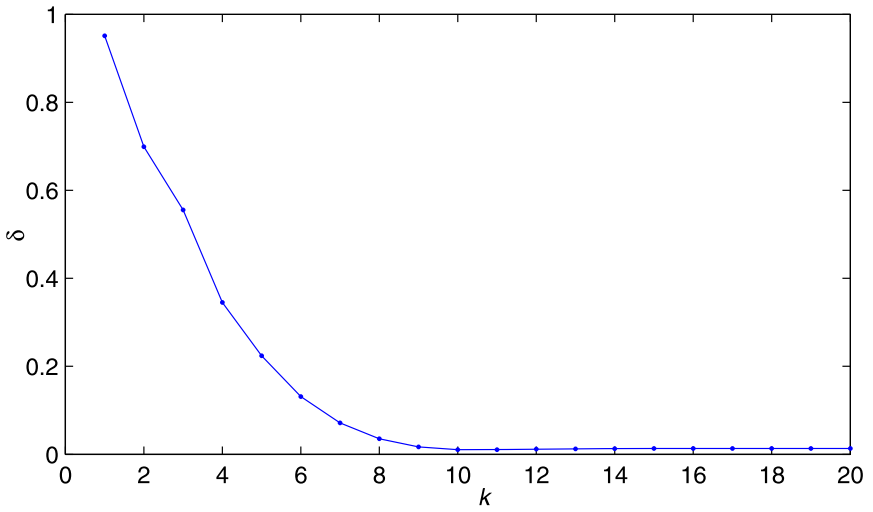

Fig. 12. The H-NI estimation errors δ versus k ($L = 1000$, $\sigma^2 = 0.30^2$).

7. For the deterministic Hammerstein systems ($v(t) \equiv 0$ or $\sigma^2 = 0$), the parameter estimation errors given by the Newton iterative (H-NI) algorithm can faster converge to zero with the iterative variable k increasing – see Table 13 and Fig. 11 with $\sigma^2 = 0$.
8. For the stochastic Hammerstein systems ($\sigma^2 \neq 0$), as the iterative variable k increases, the parameter estimation errors given by the H-NI algorithm fast converge to a small constant depending on the variance σ^2 and data length L – see Table 14 and Fig. 12 with $\sigma^2 = 0.30^2$. When the data length L goes to infinity, this constant (the estimation error) gradually becomes small and approaches zero.

8. Conclusions

This paper studies the identification problems of Hammerstein systems using the gradient search method and Newton method. We derive the projection based and gradient based algorithms and Newton recursive and Newton iterative algorithms. The proposed projection, Newton recursive and Newton iterative algorithms have fast tracking performances for the deterministic Hammerstein systems and their parameter estimation errors become small with the increase of the data length. The projection, simplified projection and Newton recursive algorithms are sensitive to noise for the stochastic Hammerstein systems and their estimation errors have large fluctuations and cannot converge to zero. In order to improve the accuracy of the parameter estimates, the stochastic gradient algorithm with the forgetting factor ($0 < \lambda < 1$) and Newton iterative algorithms are developed to enhance convergence properties for the stochastic Hammerstein systems. Moreover, the Newton recursive and Newton iterative algorithms require computing the matrix inversion and thus have larger compu-

tational load than the others. While the above algorithms are analyzed and compared using the numerical example, some important topics are not discussed in the paper, e.g., the convergence of the algorithms. This is an interesting and very challenging question, which is worth further research.

Other identification methods for linear systems can be extended to nonlinear systems, e.g., [86–104].

References

[1] M.B. Malik, M. Salman, State-space least mean square, Digital Signal Process. 18 (3) (2008) 334–345.
[2] L.L. Han, F. Ding, Multi-innovation stochastic gradient algorithms for multi-input multi-output systems, Digital Signal Process. 19 (4) (2009) 545–554.
[3] J. Ding, Y. Shi, H.G. Wang, F. Ding, A modified stochastic gradient based parameter estimation algorithm for dual-rate sampled-data systems, Digital Signal Process. 20 (4) (2010) 1238–1249.
[4] D.Q. Wang, F. Ding, Input–output data filtering based recursive least squares identification for CARARMA systems, Digital Signal Process. 20 (4) (2010) 991–999.
[5] F. Ding, T. Chen, Parameter estimation for dual-rate systems with finite measurement data, Dynam. Contin. Discrete Impuls. Syst. Ser. B: Appl. Algorithms 11 (1–2) (2004) 101–121.
[6] H.Z. Yang, J. Li, F. Ding, A neural network learning algorithm of chemical process modeling based on the extended Kalman filter, Neurocomputing 70 (4–6) (2007) 625–632.
[7] F. Ding, T. Chen, Least squares based self-tuning control of dual-rate systems, Int. J. Adapt. Control Signal Process. 18 (8) (2004) 697–714.
[8] F. Ding, T. Chen, Z. Iwai, Adaptive digital control of Hammerstein nonlinear systems with limited output sampling, SIAM J. Control Optim. 45 (6) (2006) 2257–2276.
[9] F. Ding, T. Chen, A gradient based adaptive control algorithm for dual-rate systems, Asian J. Control 8 (4) (2006) 314–323.
[10] Y. Liu, E.W. Bai, Iterative identification of Hammerstein systems, Automatica 43 (2) (2007) 346–354.
[11] K.S. Narendra, P.G. Gallman, An iterative method for the identification of nonlinear systems using a Hammerstein model, IEEE Trans. Automat. Control 11 (3) (1966) 546–550.
[12] F. Chang, R. Luus, A noniterative method for identification using Hammerstein model, IEEE Trans. Automat. Control 16 (5) (1971) 464–468.
[13] N.D. Haist, F. Chang, R. Luus, Nonlinear identification in the presence of correlated noise using a Hammerstein model, IEEE Trans. Automat. Control 18 (5) (1973) 553–555.
[14] Y. Zhu, Identification of Hammerstein models for control using ASYM, Int. J. Control 73 (18) (2000) 1692–1702.
[15] J. Vörös, Iterative algorithm for parameter identification of Hammerstein systems with two-segment nonlinearities, IEEE Trans. Automat. Control 44 (11) (1999) 2145–2149.
[16] E.W. Bai, D. Li, Convergence of the iterative Hammerstein system identification algorithm, IEEE Trans. Automat. Control 49 (11) (2004) 1929–1940.
[17] J. Vörös, Modeling and parameter identification of systems with multi-segment piecewise-linear characteristics, IEEE Trans. Automat. Control 47 (1) (2002) 184–188.
[18] J. Vörös, Parameter identification of discontinuous Hammerstein systems, Automatica 33 (6) (1997) 1141–1146.
[19] J. Vörös, Recursive identification of Hammerstein systems with discontinuous nonlinearities containing dead-zones, IEEE Trans. Automat. Control 48 (12) (2003) 2203–2206.
[20] H.F. Chen, Strong consistency of recursive identification for Hammerstein systems with discontinuous piecewise-linear memoryless block, IEEE Trans. Automat. Control 50 (10) (2005) 1612–1617.
[21] E.W. Bai, Identification of linear systems with hard input nonlinearities of known structure, Automatica 38 (5) (2002) 853–860.
[22] E.W. Bai, A blind approach to the Hammerstein–Wiener model identification, Automatica 38 (6) (2002) 967–979.
[23] V. Cerone, D. Regruto, Parameter bounds for discrete-time Hammerstein models with bounded output errors, IEEE Trans. Automat. Control 48 (10) (2003) 1855–1860.
[24] F. Ding, T. Chen, Identification of Hammerstein nonlinear ARMAX systems, Automatica 41 (9) (2005) 1479–1489.
[25] F. Ding, Y. Shi, T. Chen, Gradient-based identification methods for Hammerstein nonlinear ARMAX models, Nonlinear Dynam. 45 (1–2) (2006) 31–43.
[26] F. Ding, Y. Shi, T. Chen, Auxiliary model based least-squares identification methods for Hammerstein output-error systems, Systems Control Lett. 56 (5) (2007) 373–380.
[27] D.Q. Wang, F. Ding, Extended stochastic gradient identification algorithms for Hammerstein–Wiener ARMAX systems, Comput. Math. Appl. 56 (12) (2008) 3157–3164.
[28] J. Chen, Y. Zhang, R.F. Ding, Auxiliary model based multi-innovation algorithms for multivariable nonlinear systems, Math. Comput. Modelling (2010), doi:10.1016/j.mcm.2010.05.026.
[29] Y. Yu, J.B. Zhang, Y.W. Liao, J. Ding, Parameter estimation error bounds for Hammerstein finite impulsive response models, Appl. Math. Comput. 202 (2) (2008) 472–480.
[30] E.W. Bai, An optimal two-stage identification algorithm for Hammerstein–Wiener nonlinear systems, Automatica 34 (3) (1998) 333–338.
[31] F. Ding, T. Chen, Author's reply to "Comments on 'Identification of Hammerstein nonlinear ARMAX systems' ", Automatica 43 (8) (2007) 1497.
[32] M. Verhaegen, D. Westwick, Identifying MIMO Hammerstein systems in the context of subspace model identification methods, Int. J. Control 63 (2) (1996) 331–349.
[33] M.I. Goethals, K. Pelckmans, J.A.K. Suykens, B. De Moor, Subspace identification of Hammerstein systems using least squares support vector machines, IEEE Trans. Automat. Control 50 (10) (2005) 1509–1519.
[34] G. Golub, V. Pereyra, Separable nonlinear least squares: the variable projection method and its applications, Inverse Problems 19 (2) (2003) R1–R26.
[35] L. Ljung, System Identification: Theory for the User, 2nd ed., Prentice–Hall, Englewood Cliffs, NJ, 1999.
[36] J. Bruls, C.T. Chou, B. Haverkamp, M. Verhaegen, Linear and nonlinear system identification using separable least squares, Eur. J. Control 5 (11) (1999) 116–128.
[37] D. Westwick, R. Kearney, Separable least squares identification of nonlinear Hammerstein models: Application to stretch reflex dynamics, Ann. Biomed. Eng. 29 (8) (2001) 707–718.
[38] E.W. Bai, M. Fu, A blind approach to Hammerstein model identification, IEEE Trans. Signal Process. 50 (7) (2002) 1610–1619.
[39] J. Wang, A. Sano, D. Shook, T. Chen, B. Huang, A blind approach to closed-loop identification of Hammerstein systems, Int. J. Control 80 (2) (2007) 302–313.
[40] L. Vanbeylen, R. Pintelon, J. Schoukens, Blind maximum likelihood identification of Hammerstein systems, Automatica 44 (12) (2008) 3139–3146.
[41] F. Ding, T. Chen, Gradient based iterative algorithms for solving a class of matrix equations, IEEE Trans. Automat. Control 50 (8) (2005) 1216–1221.
[42] F. Ding, T. Chen, Iterative least squares solutions of coupled Sylvester matrix equations, Systems Control Lett. 54 (2) (2005) 95–107.
[43] F. Ding, T. Chen, On iterative solutions of general coupled matrix equations, SIAM J. Control Optim. 44 (6) (2006) 2269–2284.
[44] F. Ding, P.X. Liu, J. Ding, Iterative solutions of the generalized Sylvester matrix equations by using the hierarchical identification principle, Appl. Math. Comput. 197 (1) (2008) 41–50.

[45] L. Xie, J. Ding, F. Ding, Gradient based iterative solutions for general linear matrix equations, Comput. Math. Appl. 58 (7) (2009) 1441–1448.
[46] J. Ding, Y.J. Liu, F. Ding, Iterative solutions to matrix equations of form AiXBi=Fi, Comput. Math. Appl. 59 (11) (2010) 3500–3507.
[47] F. Ding, Transformations between some special matrices, Comput. Math. Appl. 59 (8) (2010) 2676–2695.
[48] F. Ding, P.X. Liu, G. Liu, Gradient based and least-squares based iterative identification methods for OE and OEMA systems, Digital Signal Process. 20 (3) (2010) 664–677.
[49] Y.J. Liu, D.Q. Wang, F. Ding, Least-squares based iterative algorithms for identifying Box–Jenkins models with finite measurement data, Digital Signal Process. 20 (5) (2010) 1458–1467.
[50] D.Q. Wang, F. Ding, Gradient-based iterative parameter estimation for Box–Jenkins systems, Comput. Math. Appl. (2010), doi:10.1016/j.camwa.2010.06.001.
[51] F. Ding, T. Chen, Hierarchical gradient-based identification of multivariable discrete-time systems, Automatica 41 (2) (2005) 315–325.
[52] F. Ding, T. Chen, Hierarchical least squares identification methods for multivariable systems, IEEE Trans. Automat. Control 50 (3) (2005) 397–402.
[53] L.Y. Wang, F. Ding, P.X. Liu, Consistency of HLS estimation algorithms for MIMO ARX-like systems, Appl. Math. Comput. 190 (2) (2007) 1081–1093.
[54] H.Q. Han, L. Xie, F. Ding, X.G. Liu, Hierarchical least squares based iterative identification for multivariable systems with moving average noises, Math. Comput. Modelling 51 (9–10) (2010) 1213–1220.
[55] X.G. Liu, J. Lu, Least squares based iterative identification for a class of multirate systems, Automatica 46 (3) (2010) 549–554.
[56] P. Stoica, On the convergence of an iterative algorithm used for Hammerstein system identification, IEEE Trans. Automat. Control 26 (4) (1981) 967–969.
[57] P. Stoica, T. Söderström, Instrumental variable methods for identification of Hammerstein systems, Int. J. Control 35 (3) (1982) 459–476.
[58] D.Q. Wang, Y.Y. Chu, F. Ding, Auxiliary model-based RELS and MI-ELS algorithms for Hammerstein OEMA systems, Comput. Math. Appl. 59 (9) (2010) 3092–3098.
[59] D.Q. Wang, Y.Y. Chu, G.W. Yang, F. Ding, Auxiliary model-based recursive generalized least squares parameter estimation for Hammerstein OEAR systems, Math. Comput. Modelling 52 (1–2) (2010) 309–317.
[60] J. Schoukens, W.D. Widanage, K.R. Godfrey, R. Pintelon, Initial estimates for the dynamics of a Hammerstein system, Automatica 43 (7) (2007) 1296–1301.
[61] F. Giri, Y. Rochdi, F.Z. Chaoui, A. Brouri, Identification of Hammerstein systems in presence of hysteresis-backlash and hysteresis-relay nonlinearities, Automatica 44 (3) (2008) 767–775.
[62] F. Ding, T. Chen, Combined parameter and output estimation of dual-rate systems using an auxiliary model, Automatica 40 (10) (2004) 1739–1748.
[63] F. Ding, T. Chen, Identification of dual-rate systems based on finite impulse response models, Int. J. Adapt. Control Signal Process. 18 (7) (2004) 589–598.
[64] F. Ding, T. Chen, Parameter estimation of dual-rate stochastic systems by using an output error method, IEEE Trans. Automat. Control 50 (9) (2005) 1436–1441.
[65] F. Ding, P.X. Liu, G. Liu, Auxiliary model based multi-innovation extended stochastic gradient parameter estimation with colored measurement noises, Signal Process. 89 (10) (2009) 1883–1890.
[66] Y.J. Liu, L. Xie, F. Ding, An auxiliary model based recursive least squares parameter estimation algorithm for non-uniformly sampled multirate systems, Proc. Inst. Mech. Eng. Part I: J. Syst. Control Eng. 223 (4) (2009) 445–454.
[67] L.L. Han, J. Sheng, F. Ding, Y. Shi, Auxiliary model identification method for multirate multi-input systems based on least squares, Math. Comput. Modelling 50 (7–8) (2009) 1100–1106.
[68] F. Ding, J. Ding, Least squares parameter estimation with irregularly missing data, Int. J. Adapt. Control Signal Process. 24 (7) (2010) 540–553.
[69] F. Ding, L. Qiu, T. Chen, Reconstruction of continuous-time systems from their non-uniformly sampled discrete-time systems, Automatica 45 (2) (2009) 324–332.
[70] L.L. Xiang, L.B. Xie, R.F. Ding, Hierarchical least squares algorithms for single-input multiple-output systems based on the auxiliary model, Math. Comput. Modelling 52 (5–6) (2010) 918–924.
[71] F. Ding, T. Chen, Performance analysis of multi-innovation gradient type identification methods, Automatica 43 (1) (2007) 1–14.
[72] F. Ding, Several multi-innovation identification methods, Digital Signal Process. 20 (4) (2010) 1027–1039.
[73] D.Q. Wang, F. Ding, Performance analysis of the auxiliary models based multi-innovation stochastic gradient estimation algorithm for output error systems, Digital Signal Process. 20 (3) (2010) 750–762.
[74] J.B. Zhang, F. Ding, Y. Shi, Self-tuning control based on multi-innovation stochastic gradient parameter estimation, Systems Control Lett. 58 (1) (2009) 69–75.
[75] L.L. Han, F. Ding, Identification for multirate multi-input systems using the multi-innovation identification theory, Comput. Math. Appl. 57 (9) (2009) 1438–1449.
[76] F. Ding, H.B. Chen, M. Li, Multi-innovation least squares identification methods based on the auxiliary model for MISO systems, Appl. Math. Comput. 187 (2) (2007) 658–668.
[77] Y.J. Liu, Y.S. Xiao, X.L. Zhao, Multi-innovation stochastic gradient algorithm for multiple-input single-output systems using the auxiliary model, Appl. Math. Comput. 215 (4) (2009) 1477–1483.
[78] L. Xie, H.Z. Yang, F. Ding, Modeling and identification for non-uniformly periodically sampled-data systems, IET Control Theory Appl. 4 (5) (2010) 784–794.
[79] F. Ding, P. X Liu, G. Liu, Multi-innovation least squares identification for linear and pseudo-linear regression models, IEEE Trans. Syst. Man Cybernet. Part B: Cybernet. 40 (3) (2010) 767–778.
[80] Y.J. Liu, L. Yu, F. Ding, Multi-innovation extended stochastic gradient algorithm and its performance analysis, Circuits Syst. Signal Process. 29 (4) (2010) 649–667.
[81] Y.S. Xiao, Y. Zhang, J. Ding, J.Y. Dai, The residual based interactive least squares algorithms and simulation studies, Comput. Math. Appl. 58 (6) (2009) 1190–1197.
[82] L.Y. Wang, L. Xie, X.F. Wang, The residual based interactive stochastic gradient algorithms for controlled moving average models, Appl. Math. Comput. 211 (2) (2009) 442–449.
[83] A. Hagenblada, L. Ljung, A. Wills, Maximum likelihood identification of Wiener models, Automatica 44 (11) (2008) 2697–2705.
[84] J. Vörös, Parameter identification of Wiener systems with multisegment piecewise-linear nonlinearities, Systems Control Lett. 56 (2) (2007) 99–105.
[85] G.C. Goodwin, K.S. Sin, Adaptive Filtering, Prediction and Control, Prentice–Hall, Englewood Cliffs, NJ, 1984.
[86] F. Ding, X.P. Liu, Y. Shi, Convergence analysis of estimation algorithms of dual-rate stochastic systems, Appl. Math. Comput. 176 (1) (2006) 245–261.
[87] J. Ding, F. Ding, The residual based extended least squares identification method for dual-rate systems, Comput. Math. Appl. 56 (6) (2008) 1479–1487.
[88] Y.S. Xiao, F. Ding, Y. Zhou, M. Li, J.Y. Dai, On consistency of recursive least squares identification algorithms for controlled auto-regression models, Appl. Math. Model. 32 (11) (2008) 2207–2215.
[89] J. Ding, L.L. Han, X.M. Chen, Time series AR modeling with missing observations based on the polynomial transformation, Math. Comput. Modelling 51 (5–6) (2010) 527–536.
[90] F. Ding, H.Z. Yang, F. Liu, Performance analysis of stochastic gradient algorithms under weak conditions, Sci. China Ser. F Inform. Sci. 51 (9) (2008) 1269–1280.

[91] F. Ding, P.X. Liu, H.Z. Yang, Parameter identification and intersample output estimation for dual-rate systems, IEEE Trans. Syst. Man Cybernet. Part A: Syst. Humans 38 (4) (2008) 966–975.
[92] Y. Shi, F. Ding, T. Chen, Multirate crosstalk identification in xDSL systems, IEEE Trans. Commun. 54 (10) (2006) 1878–1886.
[93] Y. Shi, F. Ding, T. Chen, 2-Norm based recursive design of transmultiplexers with designable filter length, Circuits Syst. Signal Process. 25 (4) (2006) 447–462.
[94] Y.J. Liu, J. Sheng, R.F. Ding, Convergence of stochastic gradient algorithm for multivariable ARX-like systems, Comput. Math. Appl. 59 (8) (2010) 2615–2627.
[95] F. Ding, T. Chen, L. Qiu, Bias compensation based recursive least squares identification algorithm for MISO systems, IEEE Trans. Circuits Syst. II: Express Briefs 53 (5) (2006) 349–353.
[96] F. Ding, Y.S. Xiao, A finite-data-window least squares algorithm with a forgetting factor for dynamical modeling, Appl. Math. Comput. 186 (1) (2007) 184–192.
[97] F. Ding, T. Chen, Performance bounds of the forgetting factor least squares algorithm for time-varying systems with finite measurement data, IEEE Trans. Circuits Syst. I: Reg. Papers 52 (3) (2005) 555–566.
[98] F. Ding, T. Chen, Hierarchical identification of lifted state-space models for general dual-rate systems, IEEE Trans. Circuits Syst. 52 (6) (2005) 1179–1187.
[99] F. Ding, Y. Shi, T. Chen, Performance analysis of estimation algorithms of non-stationary ARMA processes, IEEE Trans. Signal Process. 54 (3) (2006) 1041–1053.
[100] F. Ding, Y. Shi, T. Chen, Amendments to "Performance analysis of estimation algorithms of non-stationary ARMA processes", IEEE Trans. Signal Process. 56 (10) (2008) 4983–4984.
[101] Y.S. Xiao, D.Q. Wang, F. Ding, The residual based ESG algorithm and its performance analysis, J. Franklin Inst. Eng. Appl. Math. 347 (2) (2010) 426–437.
[102] J. Ding, F. Ding, S. Zhang, Parameter identification of multi-input, single-output systems based on FIR models and least squares principle, Appl. Math. Comput. 197 (1) (2008) 297–305.
[103] Y.S. Xiao, H.B. Chen, F. Ding, Identification of multi-input systems based on the correlation techniques, Int. J. Syst. Sci. (2010), doi:10.1080/00207720903470189.
[104] F. Ding, G. Liu, X.P. Liu, Partially coupled stochastic gradient identification methods for non-uniformly sampled systems, IEEE Trans. Automat. Control 55 (8) (2010), doi:10.1109/TAC.2010.2050713.

Feng Ding was born in Guangshui, Hubei Province, China. He received the B.Sc. degree from the Hubei University of Technology (Wuhan, China) in 1984, and the M.Sc. and Ph.D. degrees in automatic control both from the Department of Automation, Tsinghua University in 1991 and 1994, respectively.

From 1984 to 1988, he was an Electrical Engineer at the Hubei Pharmaceutical Factory, Xiangfan, China. From 1994 to 2002, he was with the Department of Automation, Tsinghua University, Beijing, China and he was a Research Associate at the University of Alberta, Edmonton, Canada from 2002 to 2005.

He was a Visiting Professor in the Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada from May to December 2008 and a Research Associate in the Department of Aerospace Engineering, Ryerson University, Toronto, Canada, from January to October 2009.

He has been a Professor in the School of Communication and Control Engineering, Jiangnan University, Wuxi, China since 2004. He is a Colleges and Universities "Blue Project" Middle-Aged Academic Leader (Jiangsu, China). His current research interests include model identification and adaptive control. He co-authored the book *Adaptive Control Systems* (Tsinghua University Press, Beijing, 2002), and published over 100 papers on modeling and identification as the first author.

Xiaoping Peter Liu received his B.Sc. and M.Sc. degrees from Northern Jiaotong University, China in 1992 and 1995, respectively, and Ph.D. degree from the University of Alberta, Canada in 2002. He has been with the Department of Systems and Computer Engineering, Carleton University, Canada since July 2002 and he is currently a Canada Research Chair Professor. His interest includes interactive networked systems and teleoperation, haptics, micro-manipulation, robotics, intelligent systems, context-aware intelligent networks, and their applications to biomedical engineering.

Dr. Liu has published more than 150 research articles. He serves as an Associate Editor for several journals including IEEE/ASME Transactions on Mechatronics, IEEE Transactions on Automation Science and Engineering, Intelligent Service Robotics, Int. J. of Robotics and Automation, Control and Intelligent Systems and Int. J. of Advanced Media and Communication. He received a 2007 Carleton Research Achievement Award, 2006 Province of Ontario Early Researcher Award, 2006 Carty Research Fellowship, the Best Conference Paper Award of the 2006 IEEE International Conference on Mechatronics and Automation, and a 2003 Province of Ontario Distinguished Researcher Award. He has served in the organization committees of numerous conferences including being the General Chair of the 2008 IEEE International Workshop on Haptic Audio Visual Environments and their Applications, and the General Chair of 2005 IEEE International Conference on Mechatronics and Automation.

Dr. Liu is a member of the Professional Engineers of Ontario (P. Eng.) and a senior member of IEEE.

Guangjun Liu received his B.E. degree from the University of Science and Technology of China in 1984, M.E. from the Chinese Academy of Sciences in 1987, and Ph.D. from the University of Toronto in 1996. He is currently a Professor and Canada Research Chair in Control Systems and Robotics, with the Department of Aerospace Engineering, Ryerson University, Toronto, Canada. Before becoming a faculty member in 1999, he worked as a systems engineer and design lead for Honeywell Aerospace Canada on the Boeing X-32 program. He was a postdoctoral fellow with Massachusetts Institute of Technology, USA, in 1996 before he joined Honeywell in 1997. Dr. Liu has authored or co-authored more than 130 research articles and 5 patents. His research interests are in the areas of control systems and robotics. Dr. Liu is a licensed member of the Professional Engineers of Ontario, Canada, and a senior member of IEEE.