Nonlinear Dyn (2013) 73:449–462
DOI 10.1007/s11071-013-0800-4

ORIGINAL PAPER

A novel APSO-aided maximum likelihood identification method for Hammerstein systems

Jianliang Sun · Xinggao Liu

Received: 25 October 2012 / Accepted: 26 January 2013 / Published online: 9 February 2013

Abstract Identification of Hammerstein nonlinear models has received much attention due to its ability to describe a wide variety of nonlinear systems. In this paper the maximum likelihood estimator which was originally derived for linear systems is extended to work for Hammerstein nonlinear systems in colored-noise environment. The maximum likelihood estimate is known to be statistically efficient, but can lead to complex nonlinear multidimensional optimization problem; traditional methods solve this problem at the computational cost of evaluating second derivatives. To overcome these shortcomings, a particle swarm optimization (PSO) aided maximum likelihood identification algorithm (Maximum Likelihood-Particle Swarm Optimization, ML-PSO) is first proposed to integrate PSO's simplicity in implementation and computation, and its ability to quickly converge to a reasonably good solution. Furthermore, a novel adaptive strategy using the evolution state estimation technique is proposed to improve PSO's performance (maximum likelihood-adaptive particle swarm optimization, ML-APSO). A simulation example shows that ML-APSO method outperforms ML-PSO and traditional recursive least square method in various noise conditions, and thus proves the effectiveness of the proposed identification scheme.

Keywords Hammerstein system · Maximum likelihood principle · Adaptive particle swarm optimization

1 Introduction

Nonlinear models are commonly used to describe the behavior of many industrial processes. The so called block-oriented models have turned out to be very useful for the estimation of nonlinear systems, namely the Hammerstein model, Wiener model, Hammerstein-Wiener model. Large amounts of work on the identification and control of these models and other nonlinear models have been proposed [1–11]. The Hammerstein model which consists of a static nonlinear block followed by a linear time-invariant subsystem is the focus of this paper.

There exist a large number of works on the topic of identification of Hammerstein systems in the literature. Most of the methods can be roughly classified as parametric and nonparametric methods. The method is called 'parametric' if both linear and nonlinear subsystems are described with the use of finite number of unknown parameters. For parametric methods, Narendra and Gallman (1966) proposed an iterative algorithm [12], however, this algorithm is not suitable for the general case with colored noise and non-FIR linear

J. Sun · X. Liu (✉)
State Key Laboratory of Industrial Control Technology, Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, P.R. China
e-mail: lxg@zjuem.zju.edu.cn

blocks; Vörös presented a key term separation technique to identify Hammerstein systems with discontinuous nonlinearities such as dead-zones [13]; Recently, Ding et al. presented an iterative least-squares algorithm and a recursive least-squares (RLS) algorithm for Hammerstein systems [14], Yu et al. and Ding et al. presented a class of stochastic gradient identification algorithms for linear or nonlinear system identification [15–19]. And also some new nonparametric identification methods are presented, such as the using of neural network to model the static nonlinear part [20, 21], and identification without explicit parameterization of nonlinearity driven by piece-wise constant inputs [22].

Compared with the methods mentioned above, maximum likelihood technique has superior statistical performance, and has the unique advantage to operate directly on time-domain data and simultaneously estimate all of the system parameters in a statistically optimal sense [23]. The Maximum Likelihood Estimator was first introduced by Fisher, and has been widely used and discussed since then. Ljung [24] applied the maximum likelihood approach to dynamical systems. He derived the likelihood function based on the prediction error approach and showed that the maximum likelihood method can be seen as a special case of the prediction error criterion. The main objective of this paper is to investigate maximum likelihood-based parametric identification methods for a discrete-time single-input and single-output (SISO) Hammerstein model from input data and noise contaminated output data observed from a finite time interval.

Maximum likelihood (ML) method has excellent asymptotic performance but on the other hand, usually leads to nonlinear, multidimensional optimization problems. So ML method has seen little practical use because of the apparent complexity of the computations necessary to find the maximum of the likelihood function [25]. To solve such optimization problems, several techniques have been proposed. The first is the newton-type optimization, which is sensitive to the initial estimates and involves high-order derivatives for the parameters; Amemiya (1974) has developed a class of estimators for nonlinear structural models that requires the minimization of a quadratic distance function, the distance function contains instrumental variables but no explicit Jacobian matrix, but this still needs tremendous computation work; another technique is using genetic algorithm (GA), but GA is computationally expensive compared to the particle swarm optimization (PSO); Moreover, PSO takes less time for each function evaluation as it does not use many of GA operators such as mutation, crossover, and selection operator. PSO is a stochastic and simple evolutionary optimization technique based on social-psychological model of social influence and social learning; it imitates the “fly nature” of n-dimensional swarm of particles through a problem space, in search of a single optimum or multiple optima. PSO is one of the most successful swarm intelligence techniques currently in existences and has already been applied successfully to image analysis, data clustering, neural network training etc. PSO method is popular owing to its simplicity in implementation, ability to quickly converge to a reasonably good solution and robustness against local minima [26].

Unfortunately, few papers combine PSO algorithm and maximum likelihood estimation to address nonlinear identification problem [27]. A particle swarm optimization aided Maximum Likelihood identification method (ML-PSO method) for Hammerstein system is therefore explored in this work. Although PSO algorithm is easy to implement and has been empirically shown to perform well on many optimization problems, standard PSO algorithm can get trapped in the local optima when solving complex multimodal problems while the convergence rate decreased considerably in the later period of evolution and the performance depends largely on initial parameter setting. For these reasons, an adaptive particle swarm optimization (APSO) algorithm with faster convergence speed and high accuracy is presented.

Most of the contributions assume that the systems under consideration are the nonlinear ARX models, or equation-error-like models [28, 29], and few address parametric model identification methods for the Hammerstein nonlinear ARMAX systems with colored output noises, which will be studied in this work.

The layout of the remainder of the paper is as follows. In Sect. 2, the problem of interest is described. Section 3 provides the Maximum Likelihood approach to this problem. Section 4 shows the standard PSO algorithm and presents an adaptive PSO algorithm to optimize the target function obtained by ML principle. Section 5 shows the overall identification scheme by combining APSO and maximum likelihood estimate. Section 6 provides illustrative example to show the effectiveness of the algorithm proposed. Finally, concluding remarks are offered in Sect. 7.

2 Nonlinear ARMAX modeling and problem statement

In a deterministic setting, the linear part of the system is characterized by a rational transfer function and the system output $y(t)$ is exactly observed. However, in practice the system itself may be random and the observations may be corrupted by noises. So, it is of practical importance to consider stochastic Hammerstein systems as shown in Fig. 1, which is composed of a nonlinear memoryless block $f(\cdot)$ followed by a linear subsystem. $u(t)$ is the system input, $y(t)$ is the system output, and $v(t)$ is white noise sequence. The true output $x(t)$, colored noise $w(t)$ and the inner variable $\bar{u}(t)$ which is the output of the nonlinear block are immeasurable. $N(z)$ is the transfer function of the noise model, and $G(z)$ is the transfer function of the linear part in the model.

The linear dynamical block in Fig. 1 is an ARMAX subsystem, so the nonlinear model in Fig. 1 has the following IO relationship:

$$y(t) = x(t) + w(t)$$

$$x(t) = G(z)\bar{u}(t) = \frac{B(z)}{A(z)}\bar{u}(t)$$

$$w(t) = N(z)v(t) = \frac{D(z)}{A(z)}v(t)$$

This can be transformed as

$$y(k) = -\sum_{i=1}^{n_a} a_i y(k-i) + \sum_{i=1}^{n_b} b_i \bar{u}(k-i) + \sum_{i=1}^{n_d} d_i v(k-i) + v(k) \tag{1}$$

where $v(t)$ is a white noise sequence with the normal distribution $v(k) \sim N(0, \sigma_v^2)$ for the nonparametric $f(\cdot)$, the value $f(u)$ is estimated for any fixed u. In the parametric case, $f(\cdot)$ either is expressed by a linear combination of known basis functions with unknown coefficients, or is a piecewise linear function with unknown joints and slopes, and hence identification of the nonlinear block in this case is equivalent to estimating unknown parameters. The nonlinear part is considered as a nonlinear function of a known basis $(w_1, w_2, \ldots, w_{n_c})$ with coefficients $(c_1, c_2, \ldots, c_{n_c})$ in this paper:

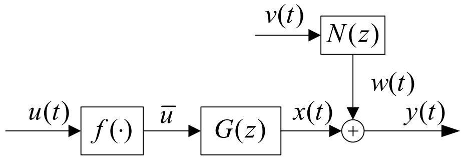


Fig. 1 The discrete-time SISO Hammerstein system

$$f\big(u(k)\big) = \bar{u}(k) = c_1\omega_1\big(u(k)\big) + c_2\omega_2\big(u(k)\big) + \cdots + c_{n_c}\omega_{n_c}\big(u(k)\big) = \sum_{i=1}^{n_c} c_i\omega_i\big(u(k)\big) \tag{2}$$

Notice that the parameterization is actually not unique. In order to get a unique parameter estimate, without loss of generality, one of the gains of $f(\cdot)$ must be fixed. Here the first coefficient of the nonlinear function is assumed to equal 1, i.e. $c_1 = 1$ [30].

Substituting (2) into (1) gives

$$y(k) = -\sum_{i=1}^{n_a} a_i y(k-i) + \sum_{j=1}^{n_b} b_j \sum_{i=1}^{n_c} c_i\omega_i\big(u(k-i)\big) + \sum_{i=1}^{n_d} d_i v(k-i) + v(k) \tag{3}$$

Define the parameter vector as

$$\boldsymbol{\theta} = [a_1, a_2, \ldots, a_{n_a}, b_1, b_2, \ldots, b_{n_b}, c_1, c_2, \ldots, c_{n_c}, d_1, d_2, \ldots, d_{n_d}] \tag{4}$$

3 Maximum likelihood identification

The problem of estimating the model of the form (3) from available input and noise-corrupted output observations is now considered. The proposed method is based on the Maximum Likelihood principle, as it is well-known that maximum likelihood estimation approaches are characterized by optimal asymptotic properties under rather mild conditions [31, 32].

Under the condition of independent observation, for a given set of measurements $\mathbf{u}_L := \{u(1), u(2), \ldots, u(L-1)\}$ and $\mathbf{y}_L := \{y(1), y(2), \ldots, y(L)\}$, the conditional probability density function of the observation vector is

$$\begin{aligned} p(\mathbf{y}_L|\mathbf{u}_{L-1}, \boldsymbol{\theta}) &= p\big(y(L)|\mathbf{y}_{L-1}, \mathbf{u}_{L-1}, \boldsymbol{\theta}\big) \\ &\quad \times p\big(y(L-1)|\mathbf{y}_{L-2}, \mathbf{u}_{L-1}, \boldsymbol{\theta}\big) \\ &\quad \times \cdots \times p\big(y(1)|\mathbf{y}(0), \mathbf{u}(0), \boldsymbol{\theta}\big) \\ &= \prod_{k=1}^{L} p\big(y(k)|\mathbf{y}_{k-1}, \mathbf{u}_{k-1}, \boldsymbol{\theta}\big) \end{aligned} \tag{5}$$

Substituting (3) into (5), we have

$$\begin{aligned}
&p(\mathbf{y}_L|\mathbf{u}_{L-1},\boldsymbol{\theta})\\
&\quad=\prod_{k=1}^{L} p\Bigg(v(k)-\sum_{i=1}^{n_a} a_i y(k-i)\\
&\qquad+\sum_{j=1}^{n_b} b_j \sum_{i=1}^{n_c} c_i\omega_i\big(u(k-i)\big)\\
&\qquad+\sum_{i=1}^{n_d} d_i v(k-i)|\mathbf{y}_{k-1},\mathbf{u}_{k-1},\boldsymbol{\theta}\Bigg)
\end{aligned}$$

Since $y(\cdot)$, $u(\cdot)$ and $v(\cdot)$ before time $k-1$ has already been determined when the observation time arrives k, and that $v(k)$ is uncorrelated with $\boldsymbol{y}_{k-1}, \boldsymbol{u}_{k-1}$ and $\boldsymbol{\theta}$, (5) can be rewritten as

$$\begin{aligned}
&p(\mathbf{y}_L|\mathbf{u}_{L-1},\boldsymbol{\theta})\\
&\quad=\prod_{k=1}^{L} p\big(v(k)\big)+const\\
&\quad=\prod_{k=1}^{L}\big(2\pi\sigma_v^2\big)^{-\frac{1}{2}}\exp\left(-\frac{1}{2\sigma_v^2}v^2(k)\right)+const\\
&\quad=\big(2\pi\sigma_v^2\big)^{-\frac{L}{2}}\exp\left(-\frac{1}{2\sigma_v^2}\sum_{k=1}^{L}v^2(k)\right)+const
\end{aligned}\tag{6}$$

where *const* denotes a constant which can be determined by the previous data, $\tilde{\sigma}_v^2$ is the estimated noise variance of $v(k)$.

The log-likelihood function is thus given by

$$\begin{aligned}
&L(\mathbf{y}_L|\mathbf{u}_{L-1},\boldsymbol{\theta})\\
&\quad:=\log L(\mathbf{y}_L|\mathbf{u}_{L-1},\boldsymbol{\theta})=\log p(\mathbf{y}_L|\mathbf{u}_{L-1},\boldsymbol{\theta})\\
&\quad=const-\frac{L}{2}\log 2\pi-\frac{L}{2}\log\sigma_v^2-\frac{1}{2\sigma_v^2}\sum_{k=1}^{L}v^2(k)
\end{aligned}\tag{7}$$

where

$$\begin{aligned}
v(k)=y(k)&+\sum_{i=1}^{n_a}a_i y(k-i)-\sum_{i=1}^{n_b}b_i\bar{u}(k-i)\\
&-\sum_{i=1}^{n_c}c_i\omega_i\big(u(k)\big)-\sum_{i=1}^{n_d}d_i v(k-i)
\end{aligned}\tag{8}$$

According to maximum likelihood principle, the maximum likelihood estimate $\tilde{\sigma}_v^2$ of noise variance σ_v^2 maximize the log-likelihood function, which leads to

$$\frac{\partial L(\mathbf{y}_L|\mathbf{u}_{L-1},\boldsymbol{\theta})}{\partial\sigma_v^2}\bigg|_{\tilde{\sigma}_v^2}=0\tag{9}$$

Thus we have

$$\tilde{\sigma}_v^2=\frac{1}{L}\sum_{k=1}^{L}v^2(k)\tag{10}$$

Substituting (10) into (7) yields

$$L(\mathbf{y}_L|\mathbf{u}_{l-1},\boldsymbol{\theta})=const-\frac{L}{2}\log\frac{1}{L}\sum_{k=1}^{L}v^2(k)\tag{11}$$

Maximizing (11) is equivalent to minimizing

$$V(\bar{\boldsymbol{\theta}}_{\mathrm{ML}})=\frac{1}{L}\sum_{k=1}^{L}v^2(k)|_{\bar{\boldsymbol{\theta}}_{\mathrm{ML}}}\tag{12}$$

where $\bar{\boldsymbol{\theta}}_{\mathrm{ML}}$ is the maximum likelihood estimate of $\boldsymbol{\theta}$. Thus, the identification problem of Hammerstein system is equivalent to this optimization problem of (12) under the constraint of (8). Obviously, $V(\bar{\boldsymbol{\theta}}_{\mathrm{ML}})$ is a nonlinear, multidimensional function with respect to $\bar{\boldsymbol{\theta}}_{\mathrm{ML}}$. To solve such an optimization problem, a novel adaptive particle swarm optimization method is proposed to estimate the system parameters.

4 Adaptive particle swarm optimization

PSO was first introduced by Kennedy and Eberhart [33]. PSO employs a natural animal behavior such as bird flocking, fish schooling to yield the best of the characteristics among the population. In PSO algorithm, each particle represents a potential solution, and the status of a particle is characterized by its position and velocity. The position and velocity of each particle are initialized randomly within searching space. Each particle flies in the search space with velocity and position which are dynamically adjusted according to its own as well as the population's best status. During the evolutionary process, the position and velocity of particle i are updated as

$$\begin{aligned}
v_i(t+1)=\omega v_i(t)&+rand_1c_1\big(Lbest_i-p_i(t)\big)\\
&+rand_2c_2\big(Gbest_i-p_i(t)\big)
\end{aligned}\tag{13}$$

$$p_i(t+1)=p_i(t)+v_i(t+1)\tag{14}$$

where $p_i(t)$ is the position of ith particle in tth iteration, $Lbest_i$ is the best previous position of this particle, $Gbest_i$ is the best previous position among all the particles in tth iterations. ω is the inertia weight to control the learning rate [34], c_1 and c_2 are acceleration coefficients and are known as the cognitive and social parameters, respectively. Finally, $rand_1$ and $rand_2$ are two random numbers in the range [0, 1]. After calculating the velocity, the new position of every particle can be worked out by (14). The PSO algorithm is repeated using (13) and (14) which are updated at each iteration, until the predefined number of iterations is reached.

Due to the popularity of PSO in practical problem solving, theoretical studies and performance improvements of the algorithm have become attractive and important. Convergence analysis and stability studies have been reported by Clerc and Kennedy [35], Kadirkamanathan et al. [36], Trelea [37]. Meanwhile, much research on performance improvements has been reported, including parameter studies, topological structures [38], and combination with auxiliary operations.

Shi and Eberhart proposed an ω linearly decreasing with the iterative generations as

$$\omega = \omega_{\max} - (\omega_{\max} - \omega_{\min})\frac{g}{G} \tag{15}$$

where g is the generation index representing the current number of revolutionary generations and G is a predefined maximum number of generations. This strategy can improve the algorithm's performance greatly. Besides, Clerc proposed a constriction coefficient to maintain the balance between exploration and exploitation in PSO.

In addition to the inertia weight and the constriction factor, the acceleration coefficients c_1 and c_2 are also important parameters in PSO. Suganthan [39] showed that using flexible values of c_1 and c_2 rather than a fixed value could yield better performance. Ratnaweera et al. [40] proposed a PSO algorithm with linearly time-varying acceleration coefficients, where a larger c_1 and a smaller c_2 were set at the beginning and gradually reversed during the search h. In this paper, a modified adaptive PSO algorithm according to the Evolution State Estimation (ESE) is presented.

During a PSO process, the population distribution characteristics vary not only with the generation number but also with the evolutionary state. The algorithm is based on the assumption that at an early stage, the elite particles scatter around the whole searching space like any other particles, the population distribution is dispersive; at the converging stage of the algorithm, the elite particles are surrounded by the other particles, at this stage, groups of particles clustering together centering the locally or globally optimal area; at the jumping out stage of the algorithm, the elite particles are far away from ordinary particles, attracting them to move towards better positions. Thus the states of the process can be classified into four sets S1, S2, S3, and S4, which represent the states of exploration, exploitation, convergence, and jumping out, respectively.

The problem now is how to detect which stage of the evolutionary process is when executing the algorithm. Here a more simple and effective evolutionary stage assessing factor is presented compared with [41].

First, denote the sum of the distance of the globally best particle and other particles as

$$d_g = \sum_{i=1}^{P} \|p_g - p_i\| \tag{16}$$

where P is the swarm size, g is the particle index of the globally best one.

Next, define D_g as

$$D_g = \left\| \sum_{i=1}^{P} (p_g - p_i) \right\| \tag{17}$$

It is obvious that D_g is always smaller than d_g. At the early stage of evolution, D_g is slightly less smaller than d_g; at the converging stage, D_g is far more less than d_g; at the jumping out stage, there is little distinction between D_g and d_g.

Then we have the evolutionary factor defined as

$$f = \frac{d_g}{D_g}, \quad f \in [0, 1] \tag{18}$$

An ESE method based on this evolutionary factor is shown in Fig. 2, S1 to S4 denotes the four stages during the evolutionary process. Each link between two different states signifies the condition under which the transition can be carried out.

After defining the evolutionary factor and knowing which stage the particle swarm is in, the inertia weight and acceleration can be improved using an adaptive strategy. Many researchers have advocated that the value of ω should be large in the exploration stage and small in the exploitation stage. The evolutionary factor f shares some characteristics with the inertia weight in that f is also relatively large during the exploration state and becomes relatively small in the convergence stage. Hence the inertia weight can be adjusted as

$$\begin{aligned} &\omega = \bigl(1 + 2\exp(-1.4f)\bigr)^{-1}, \\ &\quad f \in [0, 1],\ \omega \in [0.3, 0.7] \end{aligned} \tag{19}$$

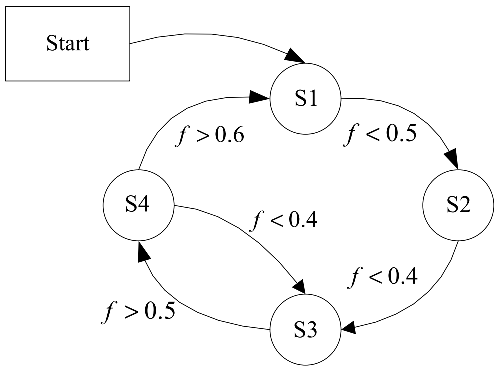


Fig. 2 ESE scheme based on evolutionary factor

The acceleration coefficients are

$$c_1^{t+1} = \begin{cases} c_1^t + (2.2 - c_1^t)d_1, & s \in \{1, 4\} \\ c_1^t + (1.8 - c_1^t)d_2, & s \in \{2, 3\} \end{cases} \tag{20}$$

$$c_2^t = 4 - c_1^t \tag{21}$$

where t is iterative variable, s is evolutionary state, d_1 and d_2 are parametric control variables.

This dynamic adjusting strategy improves the convergence speed, but also increases the probability of converging to local minimum. So a mutation operator is introduced here to solve this problem. When the algorithm is in converging stage, mutate each particle under a certain probability, so as to jump out the local minimum.

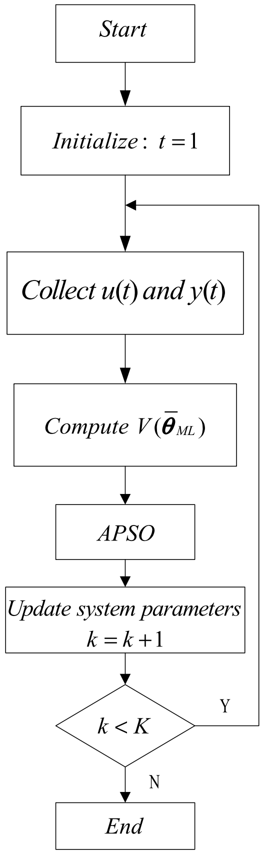
The pseudo code of the mutation operator is as follows:

```
Algorithm input: current evolutionary factor $f$, particle swarm PS
If PS is in converging stage then:
  Compute the mutation rate
  $pr = 0.03 + 0.07 \times (1 - f)^2$
  For each particle $r$ in PS:
          Get a random number $rand$ between [0, 1]
          If $pr < rand$, then:
                  Mutate $r$:
                  $r = rand \times [Max - Min] + Min$
```

5 Identification scheme

To solve the optimization problem in (12), the APSO algorithm listed above is employed to get the optimum. The iterative estimation theory is adopted: when computing the parameter estimates, the unknown noise terms are replaced with their corresponding estimates at the preceding iteration, and the estimates of these unknown noise variables are again computed through the parameter estimates [42].

Fig. 3 Flow chart of APSO algorithm



The flowchart of the identification scheme is as follow. Figure 4 is the flowchart of APSO algorithm, Fig. 3 is the overall identification scheme.

In Fig. 3, K is the iteration times of maximum likelihood estimation. In Fig. 4, i is particle number, F is fitness function, G is the iteration times for APSO. The procedure of ML-APSO algorithm is as follow:

Step 1: Prepare for the algorithm:

1.1. Initialize the system parameters and collect input and output.

1.2. Compute $V(\bar{\boldsymbol{\theta}}_{\mathrm{ML}})$ using data collected in the last step.

Step 2: Employ APSO algorithm to optimize $V(\bar{\boldsymbol{\theta}}_{\mathrm{ML}})$:

2.1. Initialize APSO parameters, such as swarm size, searching space, inertia weight, etc. to form a particle swarm $S = \{s_1, s_2 \ldots, s_p\}$. Each particle represents a possible solution corresponding to all the parameters to be identified.

2.2. Compute evolutionary factor f and get the evolution state estimate, adjust system parameters accordingly.

2.3. For each particle s_i $(i = 1, 2, \ldots, p)$, if s_i is in stage S3, perform the mutation operation; if not, calculate and record fitness of s_i as F_i $(i = 1, 2, \ldots, p)$.

2.4. If F_i is better than $F_{LocalBest}$, record particle i as *LocalBest*; if F_i is better than $F_{GlobalBest}$, record i as *GlobalBest*, and proceed to the next particle.

2.5. After all particles are processed, update particle velocity and position as (13) and (14) indicated.

2.6. If iteration number is smaller than predefined number G, jump to 2.2.

Step 3: Update system parameters with the best particle obtained from APSO

Step 4: If iteration number is smaller than predefined number K, jump to 1.2.

6 Case study

Due to the commonly recognized effectiveness of Ding and Chen's RLS algorithm, Ding's example [14] is taken as the model to demonstrate the improved identification performance of the new algorithm. This is a Hammerstein ARMAX system as follows:

$$
\begin{aligned}
&A(z)y(t) = B(z)\widetilde{u}(t) + D(z)v(t)\\
&\overline{u}(t) = f\big(u(t)\big) = c_1 u(t) + c_2 u^2(t) + c_2 u^3(t)\\
&\qquad = u(t) + 0.5u^2(t) + 0.25u^3(t)\\
&A(z)y(t) = B(z)\overline{u}(t) + D(z)v(t)\\
&B(z) = b_1 z^{-1} + b_2 z^{-2} = 0.85z^{-1} + 0.65z^{-2}\\
&D(z) = 1 + d_1 z^{-1} = 1 - 0.64z^{-1}\\
&\overline{u}(t) = f\big(u(t)\big) = c_1 u(t) + c_2 u^2(t) + c_2 u^3(t)\\
&\qquad = u(t) + 0.5u^2(t) + 0.25u^3(t)\\
&\boldsymbol{\theta} = [a_1, a_2, b_1, b_2, c_2, c_3, d_1]^T
\end{aligned}
$$

$\{u(t)\}$ is taken as a persistent excitation signal sequence with zero mean and unit variance, and $\{v(t)\}$ as a white noise sequence with zero mean and constant variance σ_v^2.

The noise-to-signal-ratio (NSR) is defined by the standard deviation of the ratio of input-free output and noise-free output in Fig. 1, that is,

$$NSR = \sqrt{\frac{\operatorname{var}[w(t)]}{\operatorname{var}[x(t)]}} \times 100\ \%$$

Apply three algorithms (RLS, ML-PSO, and ML-APSO) to this system; the sampling data length is 2000, with NSR is 17.75 % and 28.56 %, respectively. MAPSO and PSO swarm size is 20, the initial correction factor of PSO and APSO is $c1 = 2.0$, $c2 = 1.6$. The platform is as the following: operating system is 64-bit Windows 7, CPU is Intel(R) Xeon(R) CPU E31230 @ 3.20 GHz, RAM is 8.00 GB, MATLAB version is 7.11.0.584(R2010b). The total runtime of ML-APSO algorithm under above conditions is 37.1966 seconds.

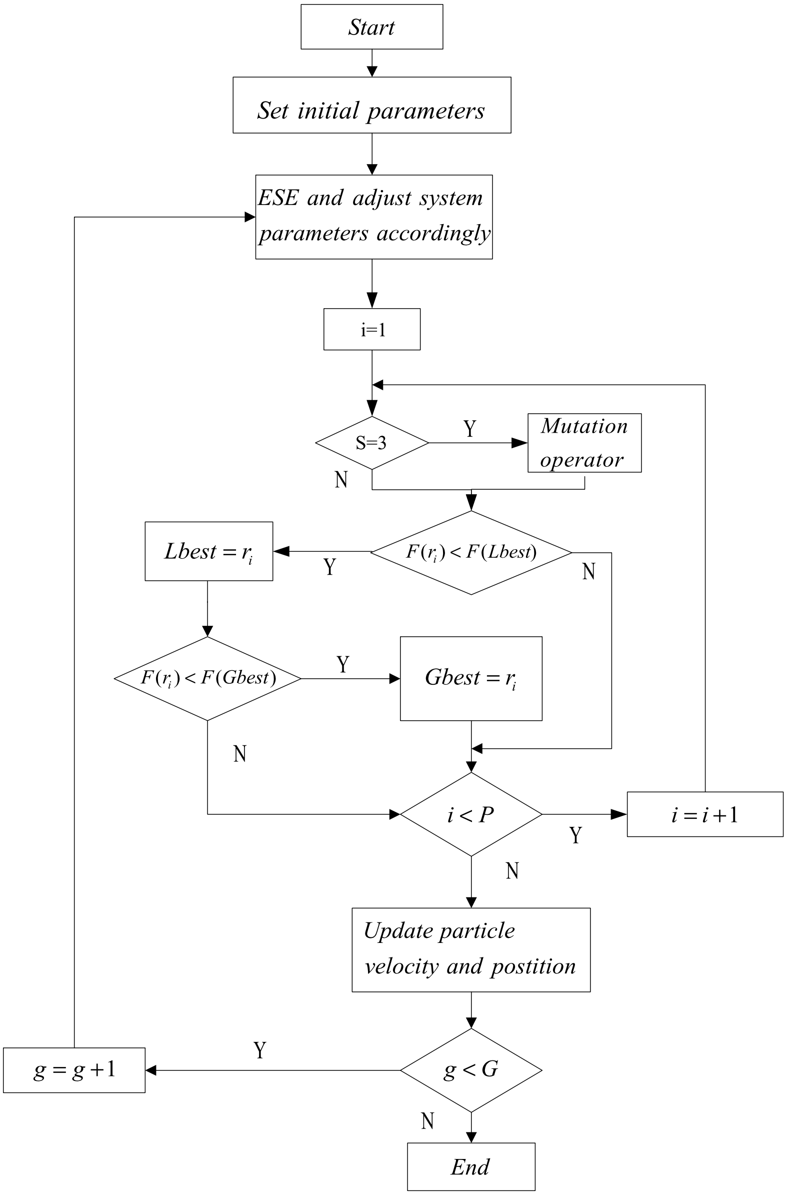
The accuracy of identification of the proposed models is assessed by comparing overall output response of estimated model and true output, and also the relative parameter estimation error which is $\sigma\ \% = \|\overline{\boldsymbol{\theta}}(t) - \boldsymbol{\theta}\| / \|\boldsymbol{\theta}\|$.

The true and estimated output of the last 50 sample times of the three algorithms are shown in Figs. 5–10 under different NSR for RLS, ML-PSO, ML-APSO, respectively. The trajectories of parameters during the identification process of ML-PSO under NSR 17.75 % is shown in Fig. 11.

As is shown in Figs. 5, 6, and 7 that, when NSR is 17.75 %, predicted output (dashed line) using RLS method fits real output (solid line) not so good, several predicted value deviate from real output largely such as values at sample time 1955, 1985 and 1992—see Fig. 5. As to ML-PSO method (Fig. 6), the predicted output fits real output much better than that of RLS, while there are still some deviations between solid line and dashed line at the sample time 1960, 1990, etc. When using ML-APSO method, as Fig. 7 depicted, the predicted output fits real output perfectly, which is huge improvement on PSO and RLS.

To see more about the improved performance of ML-APSO, the three algorithms is evaluated with a different NSR. Figures 8, 9, and 10 show the similar results ($NSR = 28.56\ \%$) are obtained as in Figs. 5, 6, and 7 ($NSR = 17.75\ \%$). Notice that the predicted output we get after applying ML-APSO fits real output much better than RLS does, especially at the

Fig. 4 Flow chart of ML-APSO algorithm



points where the derivatives of output changes dramatically, like points at sample time 1953, 1965, etc.

Figure 11 depicts the process of changing of each parameter during identification using ML-APSO algorithm when NSR is 17.75 %, the dashed lines are real parameters. It can be seen that after merely 80 iterations the estimated parameters converged to real parameters. Noise parameter d_1 has the biggest steady state error while other parameters' steady state error is relatively small.

Tables 1 and 2 list the true parameters and identification result of each parameter for RLS, ML-PSO, and ML-APSO algorithms, respectively, with NSR equals 28.56 % (Table 1) and 17.75 % (Table 2). The last column is the relative error of each estimated parameter. As we can see that ML-PSO yields better estimates than RLS, and ML-APSO yield better estimates than ML-PSO under the same sample times and NSR condition.

From Tables 1 and 2 and Figs. 5–11, the following conclusions can be drawn:

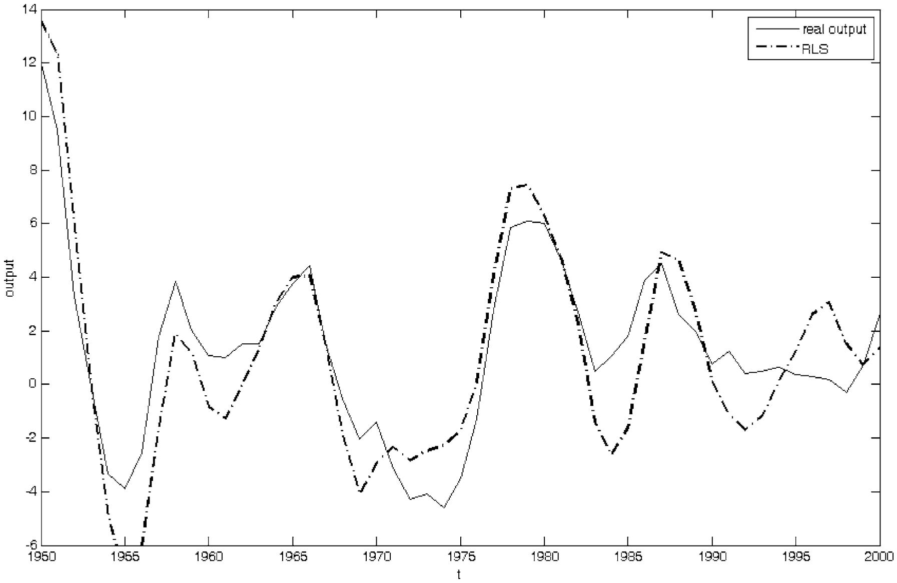


Fig. 5 Real output vs. RLS output ($NSR = 17.75\ \%$)

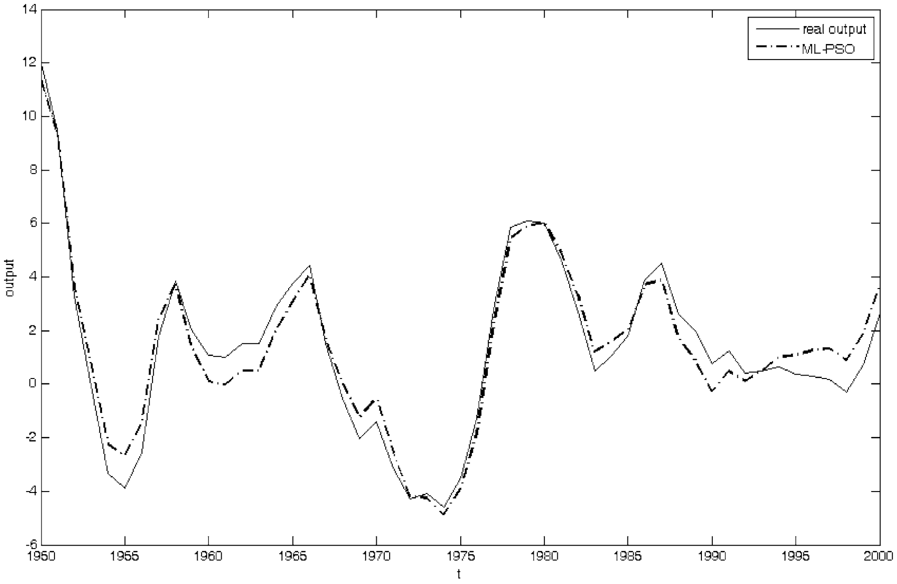


Fig. 6 Real output vs. ML-PSO output ($NSR = 17.75\ \%$)

(1) The strategy of paring maximum likelihood principle and swarm intelligence yields much better output than that of RLS, see Figs. 5–10. Moreover, the adaptive strategy performed on PSO yields better output than standard PSO—see Figs. 6–10.

(2) In line with conclusion 1, for the same data length, both of the maximum likelihood-based algorithms (ML-PSO and ML-APSO) yield superior parameter estimations to RLS algorithm proposed by Ding by 0.4042 % and 1.2939 %, respectively, when NSR is 28.6 %; 2.0367 % and 3.2235 % when NSR is 17.75 %—see Tables 1 and 2.

(3) ML-APSO algorithm achieved a smaller parameter estimation error than ML-PSO for 0.8897 % ($NSR = 28.65$) and 1.1868 % (17.75 %)—see Tables 1 and 2.

(4) As the noise-to-signal-ratio decreased from 28.56 % to 17.75 %, the parameter estimation er-

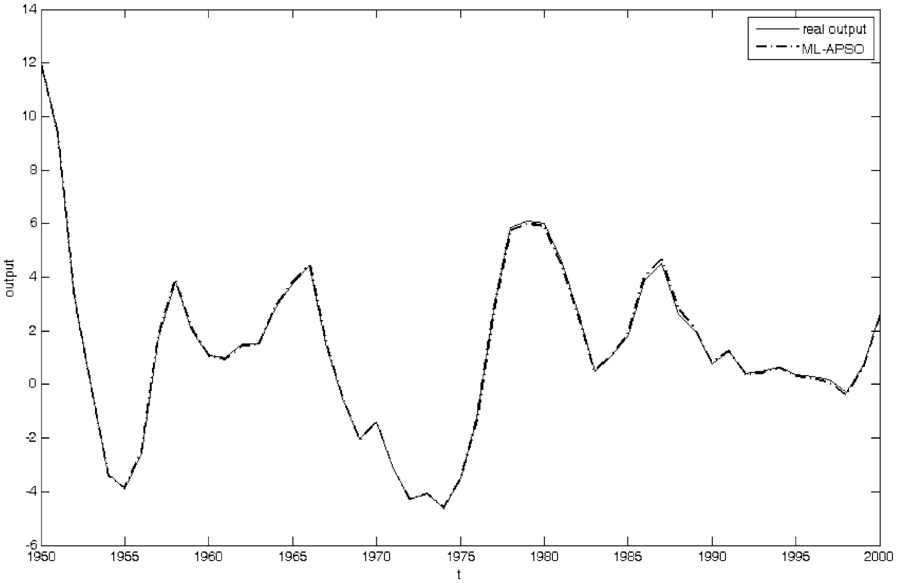


Fig. 7 Real output vs. ML-APSO output ($NSR = 17.75\ \%$)

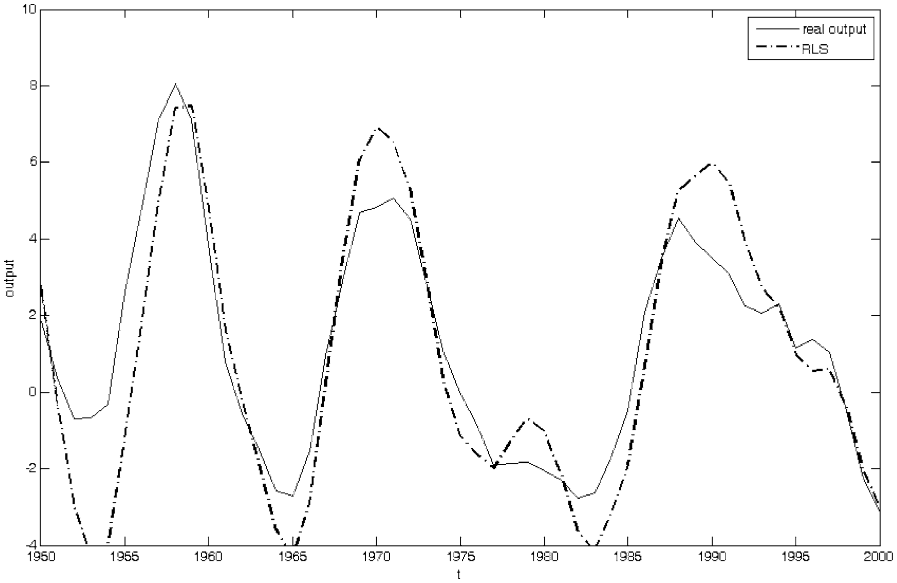


Fig. 8 Real output vs. RLS output ($NSR = 28.56\ \%$)

rors become smaller for all the three algorithms: RLS for 0.9627 %, ML-PSO for 2.5952 %, ML-APSO for 2.8932 %—see Tables 1 and 2.

7 Conclusions

A novel enhanced particle swarm optimization-based maximum likelihood algorithm is applied to estimate the Hammerstein model parameters, where an ESE strategy is presented to get a novel adaptive particle swarm optimization by defining an evolutionary factor and a state transition scheme. The presented algorithm is examined comparatively with other identification methods, due to the superior statistical performance of maximum likelihood principle and the improved searching ability of APSO, it is derived that the proposed algorithm achieves better results than its ML-PSO and RLS counterparts both in terms of

Fig. 9 Real output vs. ML-PSO output ($NSR = 28.56\ \%$)

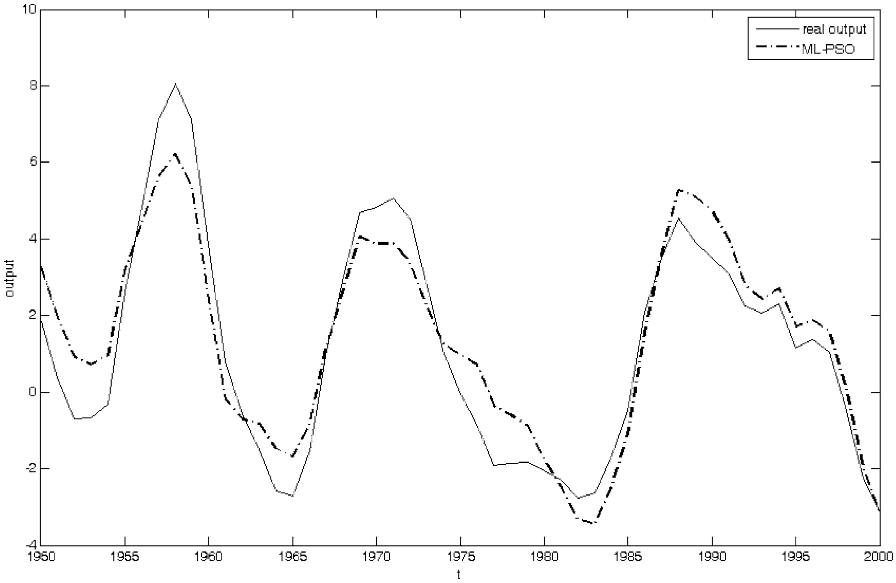
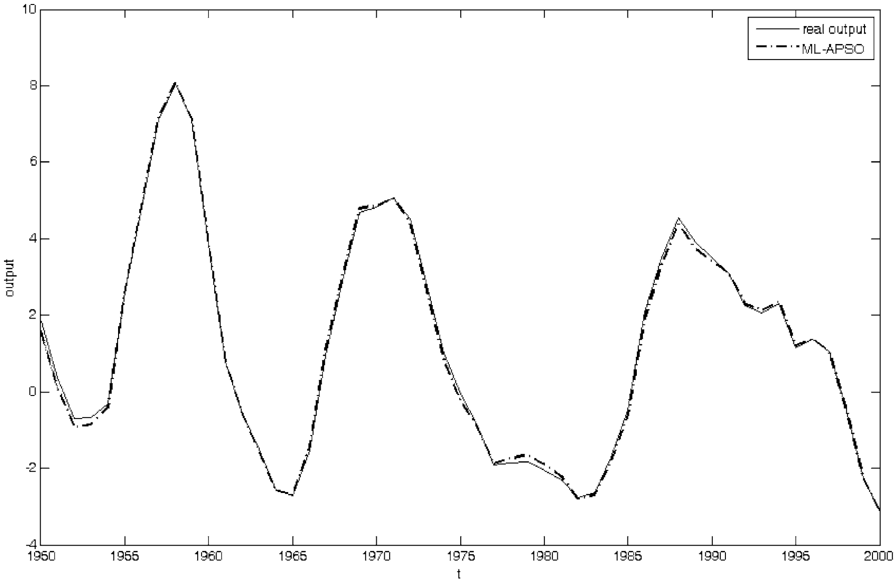


Fig. 10 Real output vs. ML-PSO output ($NSR = 28.56\ \%$)



convergence speed and accuracy under various levels of noise-to-signal ratio. When NSR is 28.65 %, the estimation error of ML-APSO is 1.2949 % superior to RLS method, when NSR is 17.75 %, the error is 3.2235 % superior to RLS. These revealed the effectiveness and advantage of the proposed ML-APSO approach.

8 Nomenclature

$y(t)$	System output
$x(t)$	True output, not corrupted by noise
$v(t)$	White noise
$w(t)$	Colored noise
$\overline{u}(t)$	Inner variable

Fig. 11 Converging process of each parameter ($NSR = 17.75\ \%$)

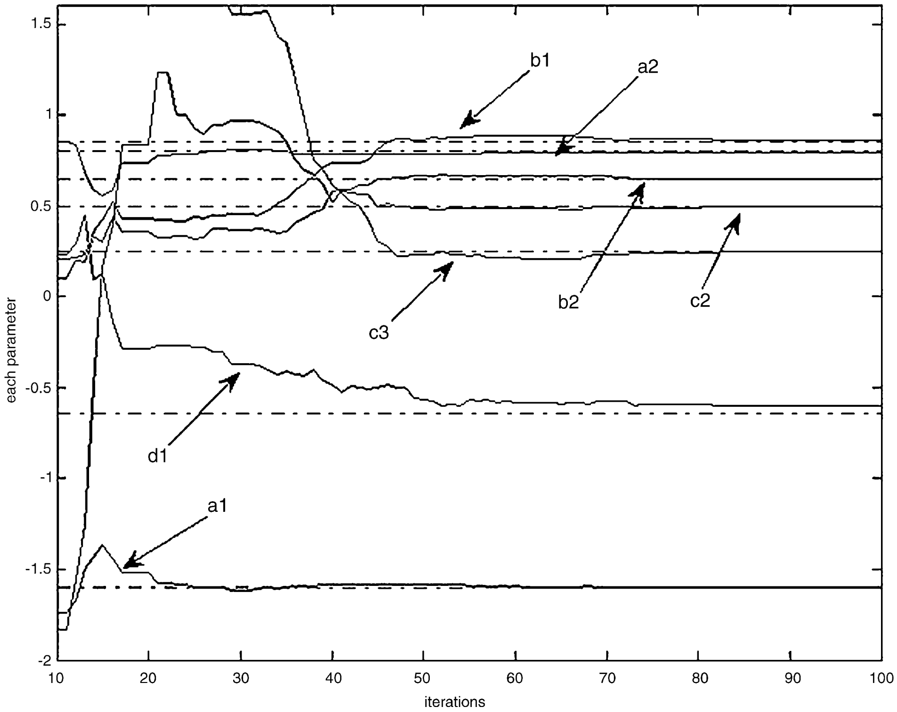


Table 1 Identification results for the parameter estimates ($NSR = 28.56\ \%$)

Method	a_1	a_2	b_1	b_2	c_2	c_3	d_1	σ %
True value	1.6000	0.8000	0.8500	0.6500	0.5000	0.2500	0.6400	0.0000 %
Ref. [14], Automatica, 2005 (RLS)	1.5981	0.8010	0.9303	0.5614	0.5095	0.3085	0.5954	6.1543 %
This work (ML-PSO)	1.5944	0.7969	0.7958	0.6131	0.5582	0.3441	0.6271	5.7501 %
This work (ML-APSO)	1.5946	0.7971	0.8948	0.6786	0.4806	0.1840	0.5939	4.8604 %

Table 2 Identification results for the parameter estimates ($NSR = 17.75\ \%$)

Method	a_1	a_2	b_1	b_2	c_2	c_3	d_1	σ %
True value	1.6000	0.8000	0.8500	0.6500	0.5000	0.2500	0.6400	0.0000 %
Ref. [14], Automatica, 2005 (RLS)	1.6023	0.8017	0.8389	0.6710	0.5002	0.2242	0.5695	5.1916 %
This work (ML-PSO)	1.6029	0.8038	0.8538	0.6245	0.5008	0.2797	0.5811	3.1549 %
This work (ML-APSO)	1.5985	0.7988	0.8614	0.6436	0.4979	0.2531	0.5979	1.9681 %

a_i, b_i, c_i, d_i	System parameters	$\overline{\boldsymbol{\theta}}_{\mathrm{ML}}$	Parameters to be identified by ML method
$f(u(t))$	Nonlinear function		
$w_1, w_2, \ldots, w_{n_c}$	Basis of a nonlinear function	$rand_1, rand_2$	Random numbers between (0, 1)
$const$	A constant number	c_1, c_2	Acceleration coefficients of PSO
σ_v^2	Variance of white noise	$Gbest$	Global best particle
$\tilde{\sigma}_v^2$	Estimated variance of white noise	$Lbest$	Local best particle

p_i Position of particle i

g Generation index representing the current number of revolutionary generations

G Predefined maximum number of generations

d_g The sum of the distance of the globally best particle and other particles

D_g The sum of vectors of the globally best particle and other particles

f Evolutionary factor

ω Inertia weight

NSR Noise-to-Signal-Ratio, defined as $NSR = \sqrt{\frac{\mathrm{var}[v(t)]}{\mathrm{var}[u(t)]}} \times 100\ \%$

Acknowledgements This work was supported by Joint Funds of NSFC-CNPC of China (Grant U1162130), National High Technology Research and Development Program (863, Grant 2006AA05Z226) and Zhejiang Provincial Natural Science Foundation for Distinguished Young Scientists (Grant R4100133).

References

1. Bai, E.W.: Decoupling the linear and nonlinear parts in Hammerstein model identification. Automatica **40**, 671–676 (2007)
2. Vanbeylen, L., Pintelon, R., Schoukens, J.: Blind maximum likelihood identification of Hammerstein systems. Automatica **44**, 3139–3146 (2008)
3. Ding, F., Liu, X.P., Liu, G.: Identification methods for Hammerstein nonlinear systems. Digit. Signal Process. **21**, 215–238 (2012)
4. Wang, D.Q., Ding, F.: Extended stochastic gradient identification algorithms for Hammerstein–Wiener ARMAX systems. Comput. Math. Appl. **56**, 3157–3164 (2008)
5. Ding, F., Shi, Y., Chen, T.: Auxiliary model based least-square identification methods for Hammerstein output error systems. Syst. Control Lett. **56**, 373–380 (2007)
6. Liu, X., Qian, J.: Modeling, control and optimization of ideal internal thermally coupled distillation columns. Chem. Eng. Technol. **23**, 235–241 (2000)
7. Chen, Y., Liu, X.: Modeling mass transport of propylene polymerization on Ziegler–Natta catalyst. Polymer **46**, 9434–9442 (2005)
8. Wang, D.Q., Ding, F.: Least squares based and gradient based iterative identification for Wiener nonlinear systems. Signal Process. **91**, 1182–1189 (2011)
9. Li, J., Ding, F., Yang, G.: Maximum likelihood least squares identification method for input nonlinear finite impulse response moving average systems. Math. Comput. Model. **55**, 442–450 (2012)
10. Zhu, Y., Liu, X.: Dynamics and control of high purity heat integrated distillation columns. Ind. Eng. Chem. Res. **44**, 8806–8814 (2005)
11. Ding, F., Shi, Y., Chen, T.: Gradient-based identification methods for Hammerstein nonlinear ARMAX models. Nonlinear Dyn. **45**, 31–43 (2005)
12. Narendra, K., Gallman, P.: An iterative method for the identification of nonlinear systems using a Hammerstein model. IEEE Trans. Autom. Control **11**, 546–550 (1966)
13. Vörös, J.: Recursive identification of Hammerstein systems with discontinuous nonlinearities containing dead-zones. IEEE Trans. Autom. Control **48**, 2203–2206 (2003)
14. Ding, F., Chen, T.: Identification of Hammerstein nonlinear ARMAX systems. Automatica **41**, 1479–1489 (2005)
15. Yu, B., Fang, H., Lin, Y., Shi, Y.: Identification of Hammerstein output-error systems with two-segment nonlinearities: algorithm and applications. Control Intell. Syst. **38**, 194–201 (2010)
16. Liu, Y., Xiao, Y., Zhao, X.: Multi-innovation stochastic gradient algorithm for multiple-input single-output systems using the auxiliary model. Appl. Math. Comput. **215**, 1477–1483 (2009)
17. Liu, Y., Sheng, J., Ding, R.: Convergence of stochastic gradient estimation algorithm for multivariable ARX-like systems. Comput. Math. Appl. **59**, 2615–2627 (2010)
18. Ding, F., Liu, G., Liu, X.: Parameter estimation with scarce measurements. Automatica **47**, 1646–1655 (2011)
19. Ding, F.: Hierarchical multi-innovation stochastic gradient algorithm for Hammerstein nonlinear system modeling. Appl. Math. Model. **37**, 1694–1704 (2013)
20. LL-Duwaish, H., Karim, M.: A new method for the identification of Hammerstein Model. Automatica **33**, 1871–1875 (1997)
21. Lang, Z.: A nonparametric polynomial identification algorithm for the Hammerstein system. IEEE Trans. Autom. Control **42**, 1435–1441 (1997)
22. Wang, J., Sano, A., Chen, T., Huang, B.: Identification of Hammerstein systems without explicit parameterization of non-linearity. Int. J. Control **82**, 937–952 (2009)
23. Chen, C.H., Fassois, S.D.: Maximum likelihood identification of stochastic Wiener–Hammerstein-type non-linear systems. Mech. Syst. Signal Process. **6**, 135–153 (1992)
24. Ljung, L.: System Identification: Theory for the User. Prentice Hall, Englewood Cliffs (1987)
25. Hagenblad, A., Ljung, L.: Maximum likelihood identification of Wiener models. Automatica **44**, 2697–2705 (2008)
26. Chen, S., Hong, X., Harris, C.J.: Particle swarm optimization aided orthogonal forward regression of unified data modeling. IEEE Trans. Evol. Comput. **14**, 477–499 (2010)
27. Konstantinos, T., Grigorios, B., Spiridon, L.: Combining evolutionary and stochastic gradient techniques for system identification. J. Comput. Appl. Math. **2**(27), 147–160 (2009)
28. Bai, E.W.: An optimal two-stage identification algorithm for Hammerstein–Wiener nonlinear systems. Automatica **34**, 333–338 (1998)
29. Chang, F., Luus, R.: A noniterative method for identification using Hammerstein model. IEEE Trans. Autom. Control **16**, 464–468 (1971)
30. Bai, E.W.: A blind approach to the Hammerstein–Wiener model identification. Automatica **38**, 967–979 (2002)
31. Astrom, K.J.: Maximum likelihood and prediction error methods. Automatica **16**, 551–574 (1980)

32. Wang, W., Ding, F., Dai, J.: Maximum likelihood least squares identification for systems with autoregressive moving average noise. Appl. Math. Model. **36**, 1842–1853 (2012)
33. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
34. Shi, Y., Eberhart, R.: Fuzzy adaptive particle swarm optimization. In: Proceedings of the 2001 Congress on Evolutionary Computation, pp. 101–106 (2001)
35. Clerc, M., Kennedy, J.: The particle swarm—explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**, 58–73 (2002)
36. Kadirkamanathan, V., Selvarajah, K., Fleming, P.J.: Stability analysis of the particle dynamics in particle swarm optimizer. IEEE Trans. Evol. Comput. **10**, 245–255 (2006)
37. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Inf. Process. Lett. **85**, 317–325 (2003)
38. Kennedy, J., Mendes, R.: Neighborhood topologies in fully-informed and best-of-neighborhood particle swarms. In: Proceedings of the 2003 IEEE SMC Workshop on Soft Computing in Industrial Applications (SMCia03), Binghamton, New York (2003)
39. Suganthan, P.N.: Particle swarm optimizer with neighborhood operator. In: Proceedings of the 1999 Congress on Evolutionary Computation, vol. 3, pp. 1958–1962 (1999)
40. Ratnaweera, A., Halgamuge, S.K., Watson, H.C.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. **8**, 240–255 (2004)
41. Zhan, Z.H., Zhang, J., Li, Y., Chung, H.S.: Adaptive particle swarm optimization. IEEE Trans. Syst. Man Cybern., Part B **39**, 1362–1381 (2009)
42. Liu, X., Lu, J.: Least squares based iterative identification for a class of multirate systems. Automatica **46**, 549–554 (2010)