Nonlinear Dynamics manuscript No.
(will be inserted by the editor)

Improved gradient descent algorithms for time-delay rational state-space systems: intelligent search method and momentum method

Jing Chen[1] · **Quanmin Zhu**[2] · **Manfeng Hu**[1] · **Liuxiao Guo**[1] · **Pritesh Narayan**[2]

the date of receipt and acceptance should be inserted later

Abstract This study proposes two improved gradient descent parameter estimation algorithms for rational state-space models with time-delay. These two algorithms, based on intelligent search method and momentum method, can simultaneously estimate the time-delay and parameters without the matrix eigenvalue calculation in each iteration. Compared with the traditional gradient descent algorithm, the improved algorithms come with two advantages: having quicker convergence rates and less computational efforts, particularly meaningful for those large-scale systems. A simulated example is selected to illustrate the efficiency of the proposed algorithms.

Keywords Rational model · Time-delay · Gradient descent · Intelligent search method · Momentum method

1 Introduction

Nonlinear systems have various descriptive models, e.g., bilinear models [1], input nonlinear models and rational models [2,3]. The rational model is a general class of nonlinear models which are expressed by a ratio of two polynomials [4,5]. It has several advantages, e.g., a comprehensive model set includes almost all the other nonlinear models, a more concise structure when compared with the polynomial expansion, fast power to catch up large deviations quickly in the represented dynamic systems [6,7]. This type of models widely represent dynamic systems appeared in natural and man-made domains, such as chemical engineering, life science and economic operation [8,9]. Further, the rational model has been selected as a foundation for developing new control design methods to deal with complex nonlinear dynamic systems, with an assumption of the parameters of the model known a priori [10,11]. Thus, the parameter estimation plays a decisive role in such model based controller design [12,13].

If a considered system is polynomial nonlinear, a great many methods can be applied for identifying the models [14–16]. However, the identification of rational model is more challenging because of the denominator polynomial. Recently, least squares (LS) algorithms for rational models have been well investigated. For example, in [17], an implicit LS algorithm is proposed for rational models, where the nonlinear model is transformed into an implicit linear in the parameters model. In [18], a nonlinear LS algorithm is developed for rational models, the parameter estimates can globally converge to the true values. Note that the LS algorithm requires the matrix

This work is supported by the National Natural Science Foundation of China (No. 61973137) and the Funds of the Science and Technology on Near-Surface Detection Laboratory (No. TCGZ2019A001).

1. J. Chen (Corresponding author), M.F. Hu, L.X. Guo
School of Science, Jiangnan University, Wuxi 214122, PR China
E-mail: chenjing1981929@126.com, humanfeng@jiangnan.edu.cn, guoliuxiao@jiangnan.edu.cn
2. Q.M. Zhu, P. Narayan
Department of Engineering Design and Mathematics, University of the West of England, Bristol BS16 1QY, UK
E-mail: quan.zhu@uwe.ac.uk, Pritesh.Narayan@uwe.ac.uk

inversion calculation in each iteration, which leads to heavy computational efforts especially for large-scale systems [19–21]. Chen et al studied a maximum likelihood based hierarchical identification principle algorithm for rational models, by which the parameters in the numerator and denominator are iteratively estimated, and then the computational efforts can be reduced [22]. However, it has an assumption that the noise-to-output ratio is small enough.

The gradient descent (GD) algorithms, including the stochastic gradient algorithms [23] and the gradient-based iterative algorithms [24,25], consist of two steps: the direction devising and the step-size calculating [26]. The gradient-based iterative algorithms perform the matrix eigenvalue calculation instead of the matrix inversion calculation in each iteration, thus have less computational efforts when compared with the LS algorithms [27]. Unfortunately, it brings some other issues, e.g., computing the eigenvalue of a high-dimensional matrix is challenging/impossible, has slower convergence rates because of its zigzagging nature [28]. It is natural to put forward such a question: can any improved GI algorithms be developed for the rational model parameter estimation, which have less computational efforts, no matrix eigenvalue calculation and quicker convergence rates. This is the insight and motivation of the paper.

State-space model is widely studied in research and adopted in wide ranges of applications. Many identification methods have been developed for linear state-space models and polynomial nonlinear state-space models [29,30]. In [31], Li proposed an on-line algorithm for a nonlinear system, which has a linear state-space subsystem. In [32], Gu et al derived an expectation maximization algorithm for linear state-space systems with time-delay and missing outputs. In [33], Xu et al developed a multi-innovation estimation algorithm for a state space system with d-step state-delay, where the delay is known in prior. Surprisingly, to the best of our knowledge, there is only scattered work reported in the literature on the identification of rational state-space models. It should be noted that the rational state-space model identification is an open and promising problem, and it is hoped that the methods proposed in this paper will provide a concise analytical solution for the reference of future studies.

In this paper, two improved GD algorithms are proposed for time-delay rational state-space models via the intelligent search method and momentum method. By using the intelligent search method, no matrix eigenvalue calculation will be involved in each iteration in the GD algorithm. Furthermore, based on the momentum method, the convergence rates can be increased. The main contributions are summarized as follows.

1. Study two improved GD algorithms for time-delay rational state-space models, where the parameters and the time-delay can be iteratively estimated.
2. Use the intelligent search method to avoid calculating the matrix eigenvalue, thus the proposed algorithms can be extended to large-scale rational models.
3. Apply the momentum method to the time-delay rational state-space models, which can increase the convergence rate of the traditional GD algorithm.

Briefly, for the rest of the study, Section 2 introduces the time-delay rational state-space model. Section 3 studies the traditional GD algorithm. Section 4 proposes two improved GD algorithms. Section 5 provides an illustrative example. Finally, Section 6 summarizes the study.

2 Rational state-space model with time-delay

Consider the following rational state-space model,

$$x(t) = \frac{\mathbf{f}(x(t-1), u(t))}{\mathbf{g}(x(t-1), u(t))},$$
$$y(t) = x(t-\tau) + v(t),$$

where $x(t)$ is the unmeasurable state, $u(t)$, $y(t)$ are the input and output data and both are measurable, $v(t)$ is a Gaussian white noise satisfies $v(t) \sim N(0, \delta^2)$, τ is the time-delay, $\mathbf{f}(x(t-1), u(t))$ and $\mathbf{g}(x(t-1), u(t))$ are two nonlinear functions, and can be written by

$$\mathbf{f}(x(t-1), u(t)) = a_1 f_1(x(t-1), u(t)) + a_2 f_2(x(t-1), u(t)) + \cdots + a_n f_n(x(t-1), u(t)),$$
$$\mathbf{g}(x(t-1), u(t)) = b_1 g_1(x(t-1), u(t)) + b_2 g_2(x(t-1), u(t)) + \cdots + b_m g_m(x(t-1), u(t)),$$

where the structures $f_i, g_j, i = 1, \cdots, n, j = 1, \cdots, m$ are known in prior, while the parameters a_i and b_j are unknown.

In application, the first element $b_1 g_1(x(t-1), u(t))$ is usually assumed to be equal to 1, e.g., in [5,7]. Then the state model can be simplified as

$$x(t) = a_1 f_1(x(t-1), u(t)) + a_2 f_2(x(t-1), u(t)) + \cdots + a_n f_n(x(t-1), u(t)) - b_2 x(t) g_2(x(t-1), u(t)) - \cdots - b_m x(t) g_m(x(t-1), u(t)).$$

Define the parameter vector and the information vector as

$$\boldsymbol{\vartheta} = [a_1, \cdots, a_n, b_2, \cdots, b_m]^{\mathrm{T}} \in \mathbb{R}^{m+n-1},$$
$$\boldsymbol{\phi}(t) = [f_1(x(t-1), u(t)), \cdots, f_n(x(t-1), u(t)), -x(t) g_2(x(t-1), u(t)), \cdots, -x(t) g_m(x(t-1), u(t))]^{\mathrm{T}} \in \mathbb{R}^{m+n-1}.$$

Then the state-space model is written by

$$x(t) = \boldsymbol{\phi}^{\mathrm{T}}(t)\boldsymbol{\vartheta},$$
$$y(t) = x(t-\tau) + v(t),$$

and can be turned into the following regression model

$$y(t) = \boldsymbol{\phi}^{\mathrm{T}}(t-\tau)\boldsymbol{\vartheta} + v(t). \tag{1}$$

The true value of the time-delay τ is unknown, but its upper bound and lower bound are known. For example, when the network using router information protocol, the data in this network may encounter a delay between $[0, 15]$. Assume that the time-delay $\tau \in [0, M]$ and collect L input and output data

$$Y(L) = [y(L), y(L-1), \cdots, y(1)]^{\mathrm{T}} \in \mathbb{R}^{L},$$
$$U(L) = [u(L), u(L-1), \cdots, u(1)]^{\mathrm{T}} \in \mathbb{R}^{L},$$
$$\boldsymbol{\Phi}(L-\tau) = [\boldsymbol{\phi}(L-\tau), \boldsymbol{\phi}(L-1-\tau), \cdots, \boldsymbol{\phi}(1-\tau)]^{\mathrm{T}} \in \mathbb{R}^{L\times(m+n-1)}. \tag{2}$$

It follows that

$$Y(L) = \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta} + V(L), \tag{3}$$

where

$$V(L) = [v(L), v(L-1), \cdots, v(1)]^{\mathrm{T}} \in \mathbb{R}^{L}.$$

The focus of this paper is to use the measurable input and output data to estimate the parameters and time-delay.

3 Traditional gradient descent algorithm

In the traditional gradient descent (T-GD) algorithm, the direction and the step-size are two decisive factors in algorithm devising [34,35]. The direction, usually termed as negative gradient direction, is determined first, then its corresponding step-size is obtained by computing the greatest eigenvalue of a matrix.

Define the cost function

$$J(\boldsymbol{\vartheta}) = \frac{1}{2}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}]^{\mathrm{T}}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}].$$

The negative gradient direction is computed by

$$\frac{-\partial J(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}}|_{\boldsymbol{\vartheta}=\boldsymbol{\vartheta}_{k-1}} = \boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}],$$

and the parameter estimates are updated by

$$\boldsymbol{\vartheta}_k = \boldsymbol{\vartheta}_{k-1} + \gamma_k \boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}], \tag{4}$$

where γ_k is the step-size, $\boldsymbol{\vartheta}_{k-1}$ is the parameter vector estimate in iteration $k-1$.

Once the direction is determined, we would compute the corresponding step-size γ_k. Notice that a small step-size will lead to a slow convergence rate, while a large one may result in a divergent algorithm. Therefore, a suitable step-size plays an important role in the T-GD algorithm devising.

The cost function in iteration k is written by

$$J(\boldsymbol{\vartheta}_k) = \frac{1}{2}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_k]^{\mathrm{T}}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_k],$$

and we want to find a step-size γ_k which can ensure

$$\frac{1}{2}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_k]^{\mathrm{T}}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_k] \leqslant \frac{1}{2}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}]^{\mathrm{T}}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}].$$

Substituting Equation (4) into the left side of the above equation yields

$$\begin{aligned}
&[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_k] = [Y(L) - \boldsymbol{\Phi}(L-\tau)[\boldsymbol{\vartheta}_{k-1} + \gamma_k \boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}]]] = \\
&[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}] - \gamma_k \boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}] = \\
&[I - \gamma_k \boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)][Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}]. \qquad (5)
\end{aligned}$$

To keep

$$\|Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_k\| \leqslant \|Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}_{k-1}\|,$$

one should guarantee that

$$\|I - \gamma_k \boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)\| < 1.$$

It follows that the step-size should be chosen as follows

$$0 < \gamma_k < \frac{2}{\lambda_{max}[\boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)]}, \qquad (6)$$

where $\lambda_{max}[\boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)]$ means the greatest eigenvalue of the matrix $\boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)$.

However, the T-GD algorithm is ineffective for this time-delay rational model, because the information matrix $\boldsymbol{\Phi}(L-\tau)$ contains unknown variables $x(t-\tau), \cdots, x(1)$ and time-delay τ. To overcome this difficulty, we assume that the parameter vector estimate in iteration $k-1$ is $\boldsymbol{\vartheta}_{k-1}$. Then the unknown variables in iteration k are replaced by

$$\hat{x}_k(t) = \boldsymbol{\phi}_k^{\mathrm{T}}(t)\boldsymbol{\vartheta}_{k-1},\ t = 1, \cdots, L,$$

where

$$\begin{aligned}
\boldsymbol{\phi}_k(t) = [&f_1(\hat{x}_k(t-1), u(t)), \cdots, f_n(\hat{x}_k(t-1), u(t)), -\hat{x}_{k-1}(t)g_2(\hat{x}_k(t-1), u(t)), \cdots, \\
&-\hat{x}_{k-1}(t)g_m(\hat{x}_k(t-1), u(t))]^{\mathrm{T}}.
\end{aligned}$$

Let

$$\begin{aligned}
\xi_k^j &= \|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k^j)\boldsymbol{\vartheta}_{k-1}\|, \quad \tau_k^j = 0, 1, \cdots, M, \\
\boldsymbol{\Phi}_k(L-\tau_k^j) &= [\boldsymbol{\phi}_k(L-\tau_k^j), \boldsymbol{\phi}_k(L-1-\tau_k^j), \cdots, \boldsymbol{\phi}_k(1-\tau_k^j)]^{\mathrm{T}},
\end{aligned}$$

and

$$\xi_k^m = \min\{\xi_k^0, \xi_k^1, \cdots, \xi_k^M\},$$

which means that the time-delay estimate in iteration k is $\tau_k = m$.

It follows that the T-GD algorithm for the time-delay rational state-space model is summarized as follows

$$\begin{aligned}
\boldsymbol{\vartheta}_k &= \boldsymbol{\vartheta}_{k-1} + \gamma_k \boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}], \\
0 &< \gamma_k < \frac{2}{\lambda_{max}[\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)]}, \\
\hat{x}_k(t) &= \boldsymbol{\phi}_k^{\mathrm{T}}(t)\boldsymbol{\vartheta}_{k-1},\ t = 1, \cdots, L,
\end{aligned}$$

$$\begin{aligned}\boldsymbol{\Phi}_k(L-\tau_k^j)&=[\boldsymbol{\phi}_k(L-\tau_k^j),\boldsymbol{\phi}_k(L-1-\tau_k^j),\cdots,\boldsymbol{\phi}_k(1-\tau_k^j)]^{\mathrm{T}},\ \tau_k^j=0,1,\cdots,M,\\ \xi_k^j&=\|Y(L)-\boldsymbol{\Phi}_k(L-\tau_k^j)\boldsymbol{\vartheta}_{k-1}\|,\quad \tau_k^j=0,1,\cdots,M,\\ \xi_k^m&=\min\{\xi_k^0,\xi_k^1,\cdots,\xi_k^M\},\quad \tau_k=m.\end{aligned}$$

Remark 1: In the T-GD algorithm, one should perform the matrix eigenvalue calculation in each iteration to get a suitable step-size [36,37]. It is problematic/imossible to get the greatest eigenvalue when the order of the matrix $[\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)]$ is large.

4 Two improved gradient descent algorithms

In this section, two improved gradient descent algorithms are developed for the rational state-space systems with time-delay. First, an intelligent search based gradient descent (IS-GD) algorithm is proposed which can avoid the matrix eigenvalue calculation. Then, to increase the convergence rate of the IS-GD algorithm, a momentum IS-GD (M-IS-GD) algorithm is derived.

4.1 Intelligent search based gradient descent algorithm

The particle swarm optimization (PSO) algorithm is an intelligent algorithm which is originally introduced by Kennedy and Eberhart [38]. The key to the PSO algorithm is initialized with a number of random estimates, termed as particles. Each particle is assigned its own velocity and is iteratively moved through the problem space. Inspired by the PSO algorithm, this paper develops an intelligent search method, in which a number of random step-sizes are involved, and each step-size can yield a residual error between the true output and the estimated output. The step-size with the smallest residual error is the best one in this iteration.

Assume that the step-sizes in iteration k are $\gamma_k^1,\cdots,\gamma_k^l$ and $\gamma_k^1<\gamma_k^2<\cdots<\gamma_k^l$, where l is the number of the step-sizes in iteration k. Then we can get l corresponding parameter vector estimates as

$$\boldsymbol{\vartheta}_k^s=\boldsymbol{\vartheta}_{k-1}^b+\gamma_k^s\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b],\quad s=1,2,\cdots,l, \tag{7}$$

where $\boldsymbol{\vartheta}_{k-1}^b$ means the best parameter vector estimate in iteration $k-1$. The residual error of each step-size is computed by

$$\epsilon_k^s=\|Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k^s\|. \tag{8}$$

Let

$$\epsilon_k^{min}=\min\{\epsilon_k^1,\epsilon_k^2,\cdots,\epsilon_k^l\},$$

and its associated estimate is $\boldsymbol{\vartheta}_k^{min}$.

Remark 2: The parameter vector estimate $\boldsymbol{\vartheta}_k^{min}$ can ensure $J(\boldsymbol{\vartheta}_k^{min})\leqslant J(\boldsymbol{\vartheta}_{k-1}^{min})$. However, its corresponding step-size γ_k^{min} is not optimal in iteration k, for the reason that all the step-sizes are chosen randomly.

There are two ways to obtain the best parameter vector estimate $\boldsymbol{\vartheta}_k^b$:

(1) $\boldsymbol{\vartheta}_k^b=\boldsymbol{\vartheta}_k^{min}$, which means that the parameter vector estimate is determined by only one step-size.

(2) Define

$$\epsilon_k^{max}=\max\{\epsilon_k^1,\epsilon_k^2,\cdots,\epsilon_k^l\},$$

and the weight of the sth step-size is computed by

$$w_k^s=\frac{\epsilon_k^{max}+1-\epsilon_k^s}{\sum\limits_{j=1}^{l}[\epsilon_k^{max}+1-\epsilon_k^j]}.$$

Then the best parameter vector estimate $\boldsymbol{\vartheta}_k^b$ can be written by

$$\boldsymbol{\vartheta}_k^b = \sum_{j=1}^{l} w_k^s \boldsymbol{\vartheta}_k^s.$$

In this case, all the step-sizes are involved in updating the parameters, and the parameter vector estimate with the smallest residual error has the largest weight.

Notice that more than one step-sizes are utilized to update the parameters in each iteration, thus this improved GD algorithm is named as intelligent search based gradient descent (IS-GD) algorithm, and can be listed as follows

$$\boldsymbol{\vartheta}_k^b = \boldsymbol{\vartheta}_k^{min}, \tag{9}$$

$$\boldsymbol{\vartheta}_k^s = \boldsymbol{\vartheta}_{k-1}^b + \gamma_k^s \boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b], \tag{10}$$

$$\hat{x}_k(t) = \boldsymbol{\phi}_k^{\mathrm{T}}(t)\boldsymbol{\vartheta}_{k-1}^b,\ t = 1, \cdots, L, \tag{11}$$

$$\boldsymbol{\Phi}_k(L-\tau_k^j) = [\boldsymbol{\phi}_k(L-\tau_k^j), \boldsymbol{\phi}_k(L-1-\tau_k^j), \cdots, \boldsymbol{\phi}_k(1-\tau_k^j)]^{\mathrm{T}},\ \tau_k^j = 0, 1, \cdots, M, \tag{12}$$

$$\epsilon_k^s = \|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k^s\|, \tag{13}$$

$$\boldsymbol{\vartheta}_k^{min} = \arg\min_{\boldsymbol{\vartheta}_k^s}\{\epsilon_k^1, \epsilon_k^2, \cdots, \epsilon_k^l\}, \tag{14}$$

$$\gamma_k^s = random(0, d), \quad s = 1, \cdots, l, \tag{15}$$

$$\xi_k^j = \|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k^j)\boldsymbol{\vartheta}_{k-1}\|, \quad \tau_k^j = 0, 1, \cdots, M, \tag{16}$$

$$\xi_k^m = \min\{\xi_k^0, \xi_k^1, \cdots, \xi_k^M\}, \quad \tau_k = m. \tag{17}$$

The steps of computing $\boldsymbol{\vartheta}_k^b$ and τ_k by the IS-GD algorithm are summarized as:

1. Initialization: Let $y(i) = 0, u(i) = 0,\ i \leqslant 0$, and give a small positive number ε.
2. Let $k = 1$ and assign $\boldsymbol{\vartheta}_k^b = \mathbf{1}/p_0$, where $\mathbf{1}$ is a column vector whose entries are all unity and $p_0 = 10^6$.
3. Assume the time-delay $\tau \in [0, M]$ and the step-size $\gamma \in [0, d]$, where M and d are known in prior.
4. Collect $y(1), \cdots, y(L), u(1), \cdots, u(L)$.
5. Compute $\hat{x}_k(t), t = 1, \cdots, L$ by Equation (11).
6. Form $\boldsymbol{\Phi}_k(L-\tau_k^j), \tau_k^j = 0, 1, \cdots, M$ by (12).
7. Compute $\xi_k^j, j = 0, 1, \cdot, M$ by Equation (16), and then determine τ_k by (17).
8. Form $\gamma_k^s, s = 1, \cdots, l$ by (15).
9. Compute $\boldsymbol{\vartheta}_k^s, s = 1 \cdots, l$ by (10).
10. Compute $\epsilon_k^s, s = 1 \cdots, l$ by (13).
11. Determine $\boldsymbol{\vartheta}_k^{min}$ by (14).
12. Update $\boldsymbol{\vartheta}_k^b$ by (9).
13. Compare $\boldsymbol{\vartheta}_k^b$ and $\boldsymbol{\vartheta}_{k-1}^b$: if $\|\boldsymbol{\vartheta}_k^b - \boldsymbol{\vartheta}_{k-1}^b\| \leqslant \varepsilon$, then obtain $\boldsymbol{\vartheta}_k^b$ and stop the procedure; otherwise, increase k by 1 and go to step 5.

Remark 3: Instead of computing the matrix eigenvalue in each iteration, the IS-GD algorithm chooses the step-sizes from a step-size pool, thus it can be extended to large-scale system identification.

Remark 4: Although the IS-GD algorithm does not require the matrix eigenvalue calculation, it brings two challenging issues: one is how to choose a suitable constant d, another is how to determine the best number of the step-sizes in each iteration.

In application, we often choose a large upper bound d first, when the parameter estimates converge to the true values, the step-size becomes smaller. Therefore, a changing d_k is usually utilized to obtain the step-sizes, e.g.,

$$d_k = \frac{k}{\sum_{i=1}^{k} i} d.$$

Remark 5: Assume that the upper bound is d_k, if the smallest step-size γ_k^1 cannot keep

$$\|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k^1\| \leqslant \|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b\|,$$

the new upper bound d_k^{new} is assigned as $d_k^{new} = 0.5d_k^{old}$. On the other hand, if the largest step-size γ_k^l can guarantee that

$$\|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k^l\| \leqslant \|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b\|,$$

we would choose a larger d_k to get a better step-size, e.g, $d_k^{new} = 1.5d_k^{old}$.

4.2 Momentum based gradient descent algorithm

The above subsection is to use the random step-size to avoid the eigenvalue calculation. Notice that the negative gradient direction is not the optimal direction in each iteration because of its zigzagging nature, the convergence rates of the IS-GD and T-GD algorithms are slow. The conjugate gradient descent algorithm is an improved GD algorithm, which can get a better direction in each iteration with the cost of more computational efforts. Based on the conjugate gradient descent algorithm, this subsection proposes a momentum based gradient descent algorithm to increase the convergence rates.

Rewrite the cost function as follows

$$J(\boldsymbol{\vartheta}) = \frac{1}{2}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}]^{\mathrm{T}}[Y(L) - \boldsymbol{\Phi}(L-\tau)\boldsymbol{\vartheta}].$$

The negative gradient direction in iteration k is written by

$$\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b].$$

According to the conjugate gradient method, the optimal direction in iteration k is a linear combination of the negative gradient direction in iteration k and the one in iteration $k-1$, that is the best direction d_k^b can be computed by

$$d_k^b = \alpha\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b] + \beta\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b], \tag{18}$$

where α and β are the weights of the negative gradient directions in k and $k-1$, respectively.

Equation (18) demonstrates that: when the neighbouring gradients have the same direction, the new direction will increase the convergence rate because of the momentum; when the neighbouring gradients have the opposite directions, the new direction will reduce vibration of the parameter estimation errors.

Then the parameter estimates are computed by

$$\boldsymbol{\vartheta}_k = \boldsymbol{\vartheta}_{k-1} + \bar{r}_{k,1}\alpha\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b] + \bar{r}_{k,2}\beta\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b].$$

For simplicity, the above equation can be transformed into

$$\boldsymbol{\vartheta}_k = \boldsymbol{\vartheta}_{k-1} + r_{k,1}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b] + r_{k,2}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b],$$

where $r_{k,1} = \bar{r}_{k,1}\alpha$ and $r_{k,2} = \bar{r}_{k,2}\beta$.

Once the direction is determined, one should compute the step-sizes. There are two ways to calculate the step-sizes:

(1) Assume that the two gradient directions have the same step-size, that is

$$\boldsymbol{\vartheta}_k = \boldsymbol{\vartheta}_{k-1} + r_k\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b] + r_k\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b].$$

The cost function can be written by

$$J(\boldsymbol{\vartheta}_k) = \frac{1}{2}[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k]^{\mathrm{T}}[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k].$$

Define

$$d_k^p = \boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b],$$
$$d_{k-1}^p = \boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L) - \boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b].$$

Let

$$\frac{\partial J(\boldsymbol{\vartheta}_k)}{\partial r_k}=0.$$

It gives rise to

$$r_k=\left\{[d_k^p+d_{k-1}^p]^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)[d_k^p+d_{k-1}^p]\right\}^{-1}[d_k^p+d_{k-1}^p]^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b].$$

Remark 6: Although the step-size is the optimal one when both the two neighbouring negative gradients have the same step-size, its computational efforts are heavy.

Remark 7: It is noted that the two directions usually have different step-sizes (weights), the assumption that both the two directions have the same step-size is problematic.

(2) The two gradient directions have different step-sizes, that is

$$\boldsymbol{\vartheta}_k=\boldsymbol{\vartheta}_{k-1}+r_{k,1}d_k^p+r_{k,2}d_{k-1}^p.$$

In this case, taking the derivative of $J(r_{k,1},r_{k,2})$ with respect to $r_{k,1},r_{k,2}$ and then equating them to zero yield

$$\begin{cases}r_{k,1}(d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_k^p+r_{k,2}(d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_{k-1}^p=\\ (d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b]\\ r_{k,1}(d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_k^p+r_{k,2}(d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_{k-1}^p=\\ (d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b].\end{cases}$$

Let

$$\begin{cases}(d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_k^p=c_k\\ (d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_{k-1}^p=d_k\\ (d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_k^p=(d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)\boldsymbol{\Phi}_k(L-\tau_k)d_{k-1}^p=l_k\\ Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b=e_k\\ Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b=e_{k-1}.\end{cases}$$

It follows that the two step-sizes can be computed by

$$r_{k,1}=\frac{(d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)e_{k-1}l_k-(d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)e_kd_k}{l_k^2-c_kd_k},$$

$$r_{k,2}=\frac{(d_k^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)e_kl_k-(d_{k-1}^p)^{\mathrm{T}}\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)e_{k-1}c_k}{l_k^2-c_kd_k}.$$

Remark 8: Since each direction is assigned its own step-size, the two-step-size momentum based gradient descent (T-M-GD) algorithm has a quicker convergence rate than that of the one-step-size momentum based gradient descent (O-M-GD) algorithm, but with the cost of more computational efforts.

Remark 9: When the estimates converge to the true values, the values of c_k,d_k and l_k satisfy $c_k=d_k=l_k$. It is impossible to compute the step-sizes. For this reason, we often use the T-M-GD algorithm first and then follow the O-M-GD algorithm.

Since the T-M-GD algorithm has heavy computational efforts and sometimes may be ineffective, we can use the IS-GD method to choose the two step-sizes in each iteration. The two-step-size momentum intelligent search based gradient descent (T-M-IS-GD) algorithm is listed as follows:

$$\begin{aligned}\boldsymbol{\vartheta}_k^b&=\boldsymbol{\vartheta}_k^{min},\\ \boldsymbol{\vartheta}_k^s&=\boldsymbol{\vartheta}_{k-1}^b+r_k^s\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b]+\\ &\quad r_{k-1}^s\boldsymbol{\Phi}_k^{\mathrm{T}}(L-\tau_k)[Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_{k-2}^b],\\ \hat{x}_k(t)&=\boldsymbol{\phi}_k^{\mathrm{T}}(t)\boldsymbol{\vartheta}_{k-1}^b,\ t=1,\cdots,L,\\ \boldsymbol{\Phi}_k(L-\tau_k^j)&=[\boldsymbol{\phi}_k(L-\tau_k^j),\boldsymbol{\phi}_k(L-1-\tau_k^j),\cdots,\boldsymbol{\phi}_k(1-\tau_k^j)]^{\mathrm{T}},\ \tau_k^j=0,1,\cdots,M,\\ \epsilon_k^s&=\|Y(L)-\boldsymbol{\Phi}_k(L-\tau_k)\boldsymbol{\vartheta}_k^s\|,\\ \boldsymbol{\vartheta}_k^{min}&=\arg\min_{\boldsymbol{\vartheta}_k^s}\{\epsilon_k^1,\epsilon_k^2,\cdots,\epsilon_k^l\},\end{aligned}$$

$$r_k^s \in random(0,d), \quad r_{k-1}^s \in random(0,d), \quad s=1,2,\cdots,l,$$
$$\xi_k^j = \|Y(L) - \boldsymbol{\Phi}_k(L-\tau_k^j)\boldsymbol{\vartheta}_{k-1}\|, \quad \tau_k^j = 0,1,\cdots,M,$$
$$\xi_k^m = \min\{\xi_k^0, \xi_k^1, \cdots, \xi_k^M\}, \quad \tau_k = m.$$

As a new direction, the proposed method can combine the Newton methods [39] and the multi-innovation methods [40] to explore the parameter identification of rational models and other nonlinear stochastic systems [41]-[45].

The algorithm constitutes of the following steps.

T-M-IS-GD algorithm

Initialize $\boldsymbol{\vartheta}_0^b = \mathbf{1}/p_0$, **get** $y(1),\cdots,y(L),u(1),\cdots,u(L)$
repeat
  for $k=1,2,\cdots,$ **do**
    Compute $\hat{x}_k(t), t=1,\cdots,L$
    Form $\boldsymbol{\Phi}_k(L-\tau_k^j), \tau_k^j=0,1,\cdots,M$
    Compute $\xi_k^j, j=0,1,\cdots,M$ **and** τ_k
    Form r_k^s **and** $r_{k-1}^s, s=1,\cdots,l$
    Compute $\boldsymbol{\vartheta}_k^s, s=1,\cdots,l$
    Compute $\epsilon_k^s, s=1\cdots,l$ **and determine** ϵ_k^{min}
    if $\epsilon_k^{min} \leqslant \|Y(L) - \boldsymbol{\Phi}(L-\tau_k)\boldsymbol{\vartheta}_{k-1}^b\|$, **then**
      Let $\boldsymbol{\vartheta}_k^b = \boldsymbol{\vartheta}_k^{min}$
    else
      Let $d=0.5d$, $\boldsymbol{\vartheta}_k^b = \boldsymbol{\vartheta}_{k-1}^b$
  end
until convergence

5 Example

Consider a rational state-space model with time-delay $\tau=1$,

$$x(t) = \frac{a_1x(t-1)+a_2x(t-1)u(t)+a_3u(t)}{1+b_2x^2(t-1)} = \frac{0.2x(t-1)+0.1x(t-1)u(t)+u(t)}{1+x^2(t-1)},$$
$$y(t) = x(t-\tau)+v(t).$$

Then one can get

$$y(t) = 0.2x(t-2)+0.1x(t-2)u(t-1)+u(t-1)-x(t-1)x^2(t-2)+v(t).$$

In the simulation, the input $\{u(t)\}$ is a persistent excitation signal sequence satisfies $u(t) \sim N(0,1)$, and $\{v(t)\}$ is a white noise sequence satisfies $v(t) \sim N(0,0.01)$, the true time-delay is $\tau=1$, and the upper bound of the time-delay is $M=3$. The simulation data ($L=1000$) are shown in Figure 1.

Firstly, apply the T-GD ($\gamma_k = \frac{1}{\lambda_{max}[\boldsymbol{\Phi}(L-\tau)\boldsymbol{\Phi}^{\mathrm{T}}(L-\tau)]}$), IS-GD ($l=50$), O-M-IS-GD and T-M-IS-GD ($l=50$) algorithms to estimate the parameters and time-delay of the model. The estimation errors $\delta := \|\boldsymbol{\vartheta}_k - \boldsymbol{\vartheta}\|/\|\boldsymbol{\vartheta}\|$ versus k are shown in Figure 2. The parameter estimates and the estimation errors are shown in Table 1. The time-delay estimates are illustrated in Figure 3.

Secondly, use the IS-GD algorithm with different l and d to identify the model ($\gamma = \frac{1}{\lambda_{max}[\boldsymbol{\Phi}(L-1)\boldsymbol{\Phi}^{\mathrm{T}}(L-1)]}$). The estimation errors $\delta := \|\boldsymbol{\vartheta}_k - \boldsymbol{\vartheta}\|/\|\boldsymbol{\vartheta}\|$ versus k are shown in Figure 4.

Finally, use the T-M-IS-GD algorithm with different l and d to identify the model. The estimation errors $\delta := \|\boldsymbol{\vartheta}_k - \boldsymbol{\vartheta}\|/\|\boldsymbol{\vartheta}\|$ versus k are shown in Table 2, the relative errors ($\delta_i =$

$\frac{\hat{a}_i - a_i}{a_i}, i = 1, 2, 3$, $\delta_4 = \frac{\hat{b}_2 - b_2}{b_2}$) of each parameter element with different iterations are shown in Figure 5.

Then, we can get the following findings:

(1) Figure 2 and Table 1 show that the M-IS-GD (T-M-IS-GD and O-M-IS-GD) algorithms have the quickest convergence rate, then is the IS-GD algorithm, finally is the T-GD algorithm.

(2) Figure 2 and Table 1 show that the T-M-IS-GD algorithm is more effective than the O-M-IS-GD algorithm, though they have the same computational efforts.

(3) Figure 3 shows that all the algorithms can estimate the time-delay, and the estimates of the T-M-IS-GD algorithm can quickly converge to the true value.

(4) Figures 4, 5 and Table 2 demonstrate that a larger l or d will lead to a quicker convergence rate. However, a larger one will also lead to heavier computational efforts.

6 Conclusions

Two improved GD algorithms are proposed for rational state-space models with time-delay in this paper. The objective of the proposed algorithms is to use the intelligent search method to avoid the matrix eigenvalue calculation, and to apply the momentum method to increase the convergence rate. In addition, these two algorithms can be extended to large-scale system identification for its simple step-size devising method, and can be applied to other literatures [46]-[49], such as information filtering and processing and networked communication systems.

Although the proposed algorithms have several advantages over the T-GD algorithm, they also bring some challenging questions that need to be answered. For example, how many step-sizes should be chosen in each iteration? what is the best upper bound of the step-sizes? These topics remain as open problems.

7 Disclosures

Conflict of Interest: The authors declare that they have no conflict of interest.

Funding: This study was funded by the National Natural Science Foundation of China (No. 61973137) and the Funds of the Science and Technology on Near-Surface Detection Laboratory (No. TCGZ2019A001).

Acknowledgments

The authors would like to thank the Associate Editor and the anonymous reviewers for their constructive and helpful comments and suggestions to improve the quality of this paper.

References

1. Zhang, X., Yang, E.F.: State estimation for bilinear systems through minimizing the covariance matrix of the state estimation errors. Int. J. Adapt. Control Signal Process. **33**(7), 1157-1173 (2019).
2. Zhang, X., Ding, F.: Hierarchical parameter and state estimation for bilinear systems. Int. J. Syst. Sci. **51**(2), 275-290 (2020).
3. Ding, F., Liu, X.P., Liu, G.: Identification methods for Hammerstein nonlinear systems. Digit. Signal Process. **21**(2), 215-238 (2011).
4. Billings, S.A., Zhu, Q.M.: Rational model identification using extended least squares algorithm. Int. J. Control **54**(3), 529-546 (1991).
5. Zhu Q.M., Wang, Y., Zhao, D., et al.: Review of rational (total) nonlinear dynamic system modelling, identification, and control. Int. J. Syst. Sci. **46**(12), 2122-2133 (2015).
6. Zhu, Q.M., Yu, D.L., Zhao, D.Y.: An enhanced linear Kalman filter (EnLKF) algorithm for parameter estimation of nonlinear rational models. Int. J. Syst. Sci. **48**(3), 451-461 (2017).
7. Chen, J., Zhu, Q.M., Li, J., Liu, Y.J.: Biased compensation recursive least squares-based threshold algorithm for time-delay rational models via redundant rule. Nonlinear Dyn. **91**(2), 797-807 (2018).
8. Kamenski, D.I., Dimitrov, S.D.: Parameter estimation in differential equations by application of rational functions. Comput. Chem. Eng. **17**, 643-651 (1993).
9. Klipp, E., Herwig, R., Kowald, A.: Systems biology in practice: Concepts, implementation and application. Weinheim, Germany: Wiley-VCH (2005).

10. Geng, X., Zhu, Q., Liu, T., Na, J.: U-model based predictive control for nonlinear processes with input delay. J. Process Control **75**, 156-170 (2019).
11. Li, H.P., Shi, Y., Yan, W.S., Liu, F.Q.: Receding horizon consensus of general linear multi-agent systems with input constraints: An inverse optimality approach. Automatica **91**, 10-16 (2018).
12. Wang, D.Q., Mao, L.: Recasted models based hierarchical extended stochastic gradient method for MIMO nonlinear systems. IET Control Theory Appl. **11**(4), 476-485 (2017).
13. Yu, C.P., Verhaegen, M., Hanson, A.: Subspace identification of local systems in one-dimensional homogeneous networks. IEEE Trans. Automat. Control **63**(4), 1126-1131 (2018).
14. Wang, D.Q., Zhang, S., Gan, M., Qiu, J.L.: A novel EM identification method for Hammerstein systems with missing output data. IEEE Trans. Ind. Inform. **16**(4), 2500-2508 (2020).
15. Wang, D.Q., Li, L.W., Ji, Y., Yan, Y.R.: Model recovery for Hammerstein systems using the auxiliary model based orthogonal matching pursuit method. Appl. Math. Model. **54**, 537-550 (2018).
16. Chen, G.Y., Gan, M., Chen, C.L.P., Li, H.X.: A regularized variable projection algorithm for separable nonlinear least-squares problems. IEEE Trans. Automat. Control **64**(2), 526-537 (2019).
17. Zhu, Q.M.: An implicit least squares algorithm for nonlinear rational model parameter estimation. Appl. Math. Model. **29**(7), 673-689 (2005).
18. Mu, B.Q., Bai, E.W., Zheng, W.X., Zhu, Q.M.: A globally consistent nonlinear least squares estimator for identification of nonlinear rational systems. Automatica **77**, 322-335 (2017).
19. Xu, H., Ding, F., Yang, E.F.: Modeling a nonlinear process using the exponential autoregressive time series model. Nonlinear Dyn. **95**, 2079-2092 (2019).
20. Chen, G.Y., Gan, M.: Generalized exponential Autoregressive models for nonlinear time series: Stationarity, estimation and applications. Inform. Sciences **438**, 46-57 (2018).
21. Li, M.H., Liu, X.M.: Least-squares-based iterative and gradient-based iterative estimation algorithms for bilinear systems. Nonlinear Dyn. **89**(1), 197-211 (2017).
22. Chen, J., Zhu, Q.M., Liu, Y.J.: Maximum likelihood based identification methods for rational models. Int. J. Syst. Sci. **50**(11), 1-13 (2019).
23. Zhang, X.: Recursive parameter estimation methods and convergence analysis for a special class of nonlinear systems. Int. J. Robust Nonlinear Control **30**(4), 1373-1393 (2020).
24. Ding, F., Lv, L., Pan, J., Wan, X.K., Jin, X.B.: Two-stage gradient-based iterative estimation methods for controlled autoregressive systems using the measurement data. Int. J. Control Autom. Syst. **18**(4), 886-896 (2020).
25. Ding, F., Xu, L., Meng, D.D., et al.: Gradient estimation algorithms for the parameter identification of bilinear systems using the auxiliary model. J. Comput. Appl. Math. (2020). DOI: 10.1016/j.cam.2019.112575
26. Wang, D.Q., Yan, Y.R., Liu, Y.J., Ding, J.H.: Model recovery for Hammerstein systems using the hierarchical orthogonal matching pursuit method. J. Comput. Appl. Math. **345**, 135-145 (2019).
27. Gan, M., Chen, G.Y., Chen, L., Chen, C.L.P.: Term selection for a class of nonlinear separable models. IEEE Trans. Neur. Net. Lear. Syst. (2020). DOI: 10.1109/TNNLS.2019.2904952
28. Wan, L.J., Ding, F.: Decomposition-gradient-based iterative identification algorithms for multivariable systems using the multi-innovation theory. Circuits Syst. Signal Process. **38**, 2971-2991 (2019).
29. Ma, J.X., Wu, O., Huang, B., et al.: Expectation maximization estimation for a class of input nonlinear state space systems by using the Kalman smoother. Signal Process. **145**, 295-303 (2018).
30. Zhang, X., Alsaadi, F.E., Hayat, T.: Recursive parameter identification of the dynamical models for bilinear state space systems. Nonlinear Dyn. **89**(4), 2415-2429 (2017).
31. Li, J.H., Zheng, W., Gu, J.P., Hua, L.: A recursive identification algorithm for Wiener nonlinear systems with linear state-space subsystem. Circuits Syst. Signal Process. **37**(6), 2374-2393 (2018).
32. Gu, Y., Liu, J., Li, X., et al.: State space model identification of multirate processes with time-delay using the expectation maximization. J. Frankl. Inst. **356**(3), 1623-1639 (2019).
33. Xu, L., Ding, F., et al.: A multi-innovation state and parameter estimation algorithm for a state space system with d-step state-delay. Signal Process. **140**, 97-103 (2017).
34. Ding, F.: Hierarchical multi-innovation stochastic gradient algorithm for Hammerstein nonlinear system modeling. Appl. Math. Model. **37**(4), 1694-1704 (2013).
35. Ding, F., Liu, X.G., Chu, J.: Gradient-based and least-squares-based iterative algorithms for Hammerstein systems using the hierarchical identification principle. IET Control Theory Appl. **7**(2), 176-184 (2013).
36. Xu, L.: The damping iterative parameter identification method for dynamical systems based on the sine signal measurement. Signal Process. **120**, 660-667 (2016).
37. Xu., L.: The parameter estimation algorithms based on the dynamical response measurement data. Adv. Mech. Eng. **9**(11), 1-12 (2017).
38. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. Proc. of IEEE Int. Conf. Neur. Net. IV, 1942–1948 (1995).
39. Xu, L., Chen, L., Xiong, W.L.: Parameter estimation and controller design for dynamic systems from the step responses based on the Newton iteration. Nonlinear Dyn. **79**(3), 2155-2163 (2015).
40. Xu, L, Xiong, W.L., Alsaedi, A., Hayat, T.: Hierarchical parameter estimation for the frequency response based on the dynamical window data. Int. J. Control Autom. Syst. **16**(4), 1756-1764 (2018).
41. Pan, J., Li, W., Zhang, H.P.: Control algorithms of magnetic suspension systems based on the improved double exponential reaching law of sliding mode control. Int. J. Control Autom. Syst. **16**(6), 2878-2887 (2018).
42. Wan, X.K., Li, Y., Xia, C., et al.: A T-wave alternans assessment method based on least squares curve fitting technique. Measurement **86**, 93-100 (2016).
43. Chang, Y.F., Zhai, G.S., Fu, B., Xiong, L.L.: Quadratic stabilization of switched uncertain linear systems: a convex combination approach. IEEE-CAA J. Autom. Sin. **6**(5), 1116-1126 (2019).

44. Geng, L., Xiao, R.B.: Control and backbone identification for the resilient recovery of a supply network utilizing outer synchronization. Appl. Sci. **10**(1), 213 (2020).
45. Tang, L., Liu, G.J., Yang, M., et al.: Joint design and torque feedback experiment of rehabilitation robot. Adv. Mech. Eng. **12**, 1-11 (2020).
46. Zhang, Y., Huang, M.M., Wu, T.Z., Ji, F.: Reconfigurable equilibrium circuit with additional power supply. Int. J. Low-Carbon Tec. **15**(1), 106-111 (2020).
47. Wang, L., Liu, H., Dai, L.V., Liu, Y.W.: Novel method for identifying fault location of mixed lines. Energies **11**(6), 1529 (2018).
48. Liu, H., Zou, Q.X., Zhang, Z.P.: Energy disaggregation of appliances consumptions using ham approach. IEEE Access **7**, 185977-185990 (2019).
49. Zhao, X.L., Lin, Z.Y., et al.: Research on automatic generation control with wind power participation based on predictive optimal 2-degree-of-freedom PID strategy for multi-area interconnected power system. Energies **11**(12), 3325 (2018).

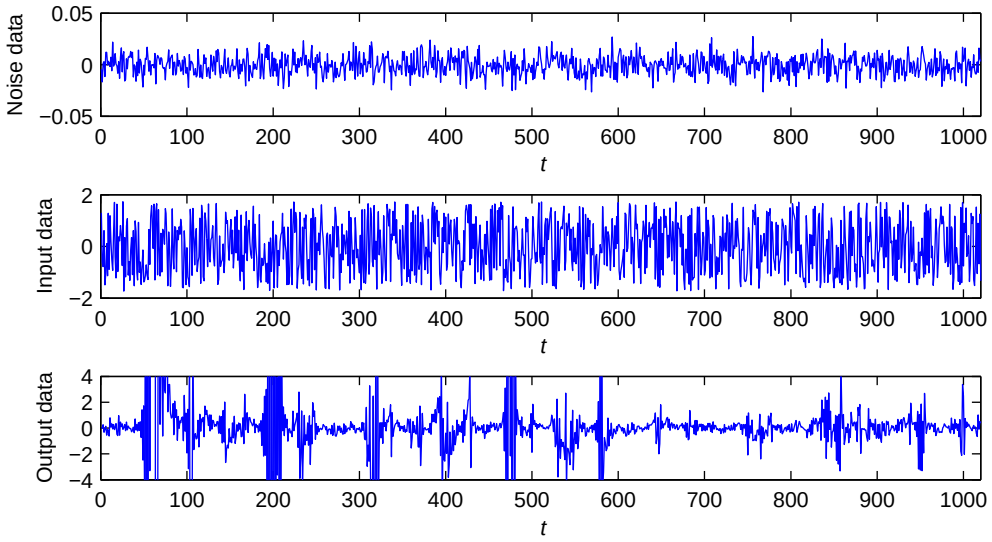

Fig. 1 The simulation data

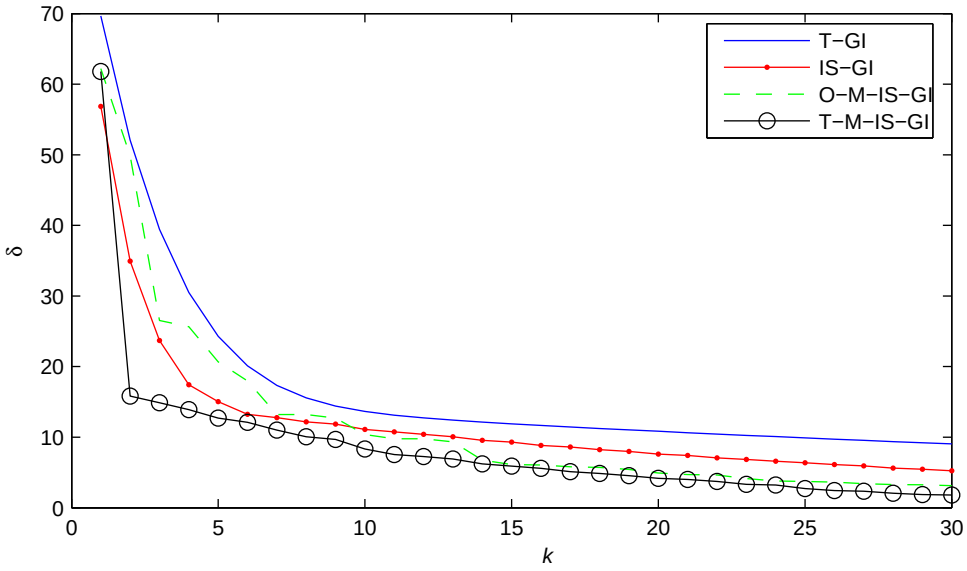

Fig. 2 The parameter estimation errors δ versus k

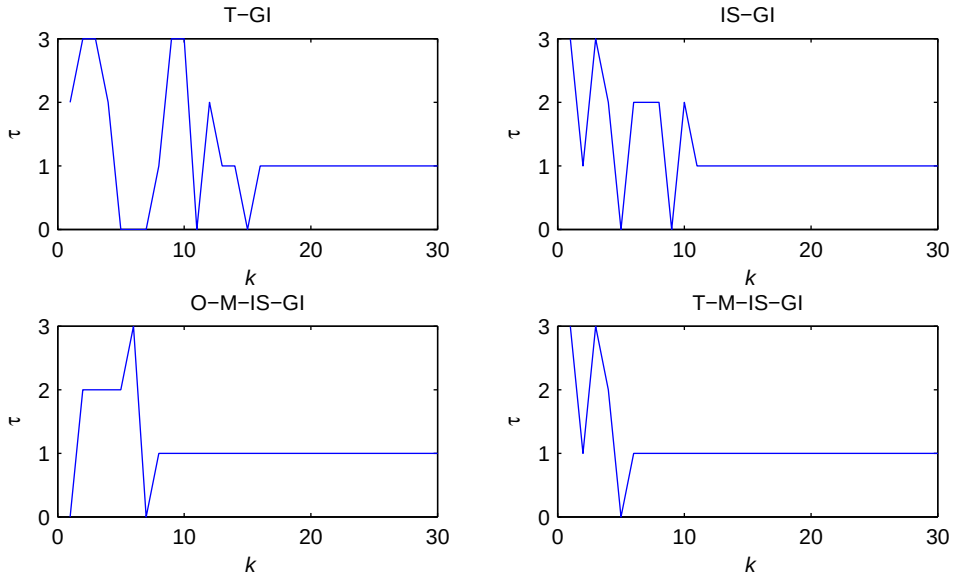

Fig. 3 The time-delay estimates versus k

Table 1 The estimates and errors

Algorithm	k	a_1	a_2	a_3	b_2	δ (%)
T-GD	1	0.24576	0.09987	0.00253	-0.49140	69.66419
	2	0.35230	-0.06167	0.00444	-0.63782	52.06156
	10	0.66990	-0.46207	0.02719	-0.96949	13.64437
	20	0.69866	-0.49734	0.05545	-0.99811	10.83780
	30	0.69996	-0.49911	0.07932	-0.99951	9.04557
IS-GD	1	0.45421	0.18459	0.00467	-0.90822	56.86805
	2	0.43857	-0.33466	0.00655	-0.71014	34.92554
	10	0.69842	-0.50219	0.05208	-0.99677	11.09145
	20	0.69918	-0.50118	0.09851	-0.99821	7.60915
	30	0.69954	-0.50060	0.13000	-0.99896	5.24763
O-M-IS-GD	1	0.33299	0.13532	0.00343	-0.66583	62.19947
	2	0.45486	0.07318	0.00512	-0.87319	49.86956
	10	0.69103	-0.49509	0.06202	-1.00492	10.37692
	20	0.70152	-0.50027	0.13440	-0.99661	4.92469
	30	0.69993	-0.49950	0.15808	-1.00025	3.14252
T-M-IS-GD	1	0.33882	0.13769	0.00349	-0.67748	61.79626
	2	0.66858	-0.55250	0.01015	-1.06881	15.81541
	10	0.70445	-0.49546	0.08921	-1.00091	8.31802
	20	0.69864	-0.49949	0.14426	-1.00140	4.18024
	30	0.69969	-0.50028	0.17574	-0.99978	1.81874
True Values		0.70000	-0.50000	0.20000	-1.00000	

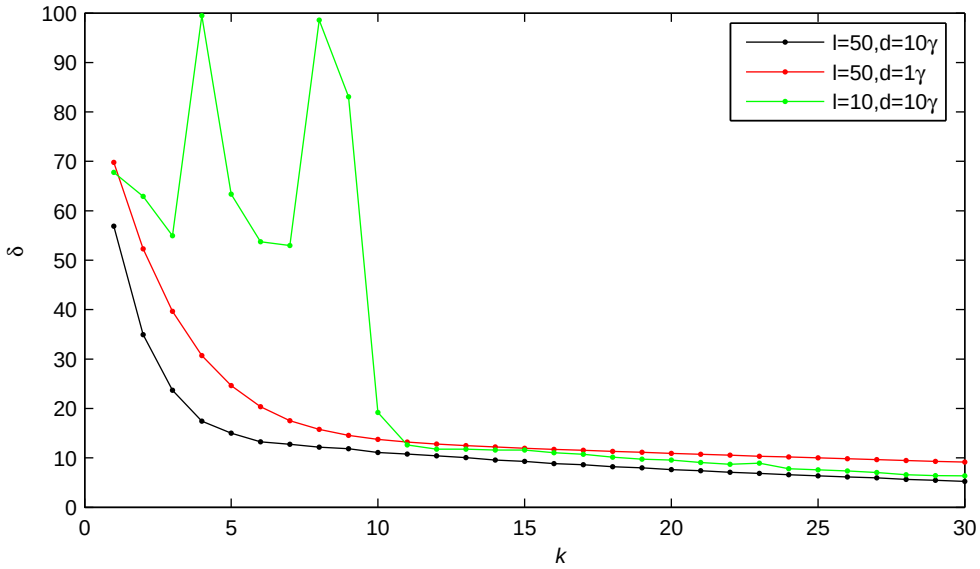

Fig. 4 The parameter estimation errors δ versus k (IS-GD)

Table 2 The T-M-IS-GD algorithm estimates and errors with different l and d

l, d	k	a_1	a_2	a_3	b_2	δ (%)
	1	0.18255	0.07419	0.00188	-0.36502	76.43371
	2	0.30649	0.07864	0.00334	-0.59791	62.26199
$l = 5, d = \gamma$	10	0.66368	-0.45466	0.02158	-0.96331	14.33020
	20	0.69830	-0.49679	0.04640	-0.99773	11.51723
	30	0.69993	-0.49899	0.06735	-0.99943	9.94270
	1	0.33342	0.13550	0.00343	-0.66669	62.16915
	2	0.44505	-0.09621	0.00568	-0.79979	41.45447
$l = 50, d = \gamma$	10	0.69896	-0.49467	0.03845	-0.99873	12.11605
	20	0.69933	-0.50078	0.08737	-0.99841	8.44281
	30	0.70048	-0.49846	0.12099	-1.00054	5.92343
	1	0.05736	0.02331	0.00059	-0.11470	92.11485
	2	0.79935	0.17911	0.00880	-1.55093	67.50518
$l = 5, d = 10\gamma$	10	1.08561	-0.04296	0.09898	-0.93216	45.73897
	20	0.70358	-0.50228	0.13883	-0.99965	4.59598
	30	0.70185	-0.49885	0.17704	-0.99907	1.73016
	1	0.33882	0.13769	0.00349	-0.67748	61.79626
	2	0.66858	-0.55250	0.01015	-1.06881	15.81541
$l = 50, d = 10\gamma$	10	0.70445	-0.49546	0.08921	-1.00091	8.31802
	20	0.69864	-0.49949	0.14426	-1.00140	4.18024
	30	0.69969	-0.50028	0.17574	-0.99978	1.81874
True Values		0.70000	-0.50000	0.20000	-1.00000	

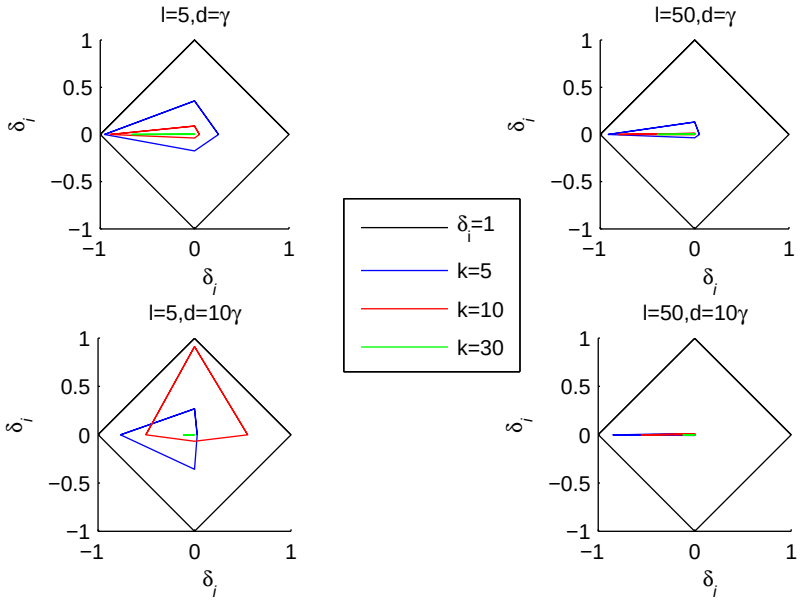

Fig. 5 The relative errors of each parameter vector with different iterations (T-M-IS-GI)