Multi-direction gradient iterative algorithm: a unified framework for gradient iterative and least squares algorithms

Jing Chen, Junxia Ma*, Min Gan, Quanmin Zhu

***Abstract*—In this study, a multi-direction-based gradient iterative (GI) algorithm for Hammerstein systems with irregular sampling data is proposed. The algorithm updates the parameter estimates using several orthogonal directions at each iteration. The convergence rate is significantly improved with an increasing number of directions. The convergence property and two simulation examples are provided to demonstrate the effectiveness of the proposed algorithm. In addition, the multi-direction-based GI algorithm establishes a relationship between the traditional GI and least squares (LS) algorithms. Thus, our algorithm that combines the LS and GI algorithms constructs an identification framework for a significantly wider class of systems.**

***Index Terms*—Hammerstein system, multi-direction, GI algorithm, irregular sampling, computational load, convergence rate**

I. Introduction

The least squares (LS) and gradient iterative (GI) algorithms are two classical types of parameter estimation methods, which are widely applied in system identification [1], [2]. The basic idea of the LS algorithm is to obtain an analytical solution of a derived derivative function whose primitive function is typically defined as the error between the true and predicted outputs [3]. The LS algorithm can obtain the solution in only one iteration via the current input-output data. However, a matrix inversion is involved in the LS algorithm, which leads to heavy computational loads, particularly for large-scale [4] or hidden-variable systems [5]. Moreover, a derivative function can sometimes be unsolvable when the considered models have complex structures [6], such as the exponential autoregressive model [7] and rational model [8].

To avoid performing a matrix inversion and to reduce the computational power required by the LS algorithm, the GI algorithm is considered as a good alternative. The direction and step-size are two decisive factors in the GI algorithm setup [9], [10]. The direction, typically known as negative gradient, can cause the estimates to move toward the true values, while the step-size can determine the convergence rate. Note that the GI algorithm does not require the derivative function calculation, thus it can be extended to complex nonlinear model identification. However, the zigzagging nature of the GI algorithm leads to slow convergence rates. In addition, the eigenvalue calculation makes the GI algorithm inefficient for large-scale system identification [11]. Therefore, the design of modified GI algorithms that have faster convergence rates and no eigenvalue calculation remains an open and challenging research question [12].

There are two ways to improve the estimation efficiency of GI algorithms. The first is to determine an optimal direction rather than the negative gradient direction, which can be achieved by adopting the conjugate gradient iterative (CGI) [13], [14] and the multi-innovation-based GI (MI-GI) algorithms [15]. The other is to obtain a suitable step-size at each iteration, which can be achieved by adopting the forgetting factor GI [16] and multi-step-size GI algorithms [17]. The CGI algorithm assigns a linear combination of two orthogonal neighboring directions in a new direction. However, when the considered systems have high-order or hidden variables, the CGI algorithm can be inefficient. Unlike the CGI algorithm, the MI-GI algorithm performs several innovations (directions) at each iteration to increase the convergence rate, but these innovations are not orthogonal, leading to the lack of confidence in the number of innovations. Therefore, the convergence rates of the MI-GI algorithm may not be improved with an increasing number of innovations. Motivated by the CGI and MI-GI algorithms, this study focuses on direction devising. The multi-direction-based GI (MUL-D-GI) algorithm designed here is expected to provide a concise analytical solution for future studies.

Nonlinear systems widely exist in process control industries. Different types of nonlinear models are used to describe the dynamics of complex systems, such as the Hammerstein model, [18], the rational model [19], and the output nonlinear model [20], [21]. This paper develops an MUL-D-GI algorithm for Hammerstein models. Several orthogonal directions are involved at each iteration, which can improve the convergence rate of the GI algorithm. Furthermore, the optimal number of directions for each iteration and relationships among the GI, MUL-D-GI, and LS algorithms are determined. In summary, the contributions of this paper are as follows.

(1) The MUL-D-GI algorithm has a faster convergence rate than that of the GI algorithm, and can thus be extended to large-scale systems.

(2) The MUL-D-GI algorithm has a lower computational load than that of the LS algorithm, and can thus be applied to nonlinear models with complex structures.

(3) The MUL-D-GI algorithm establishes a link between the GI and LS algorithms, thus constructing an identification framework for a considerable wider class of systems.

(4) The properties of the MUL-D-GI algorithm are presented, serving as a guide for researchers to select the optimal number of directions on a case-by-case basis.

The remainder of this paper is organized as follows: Section II introduces the Hammerstein models and traditional identification algorithms; Section III investigates the two-direction-based GI algorithm; Section IV derives the multi-direction-based GI algorithm and states its properties; Section V provides two simulation examples; Finally, Section VI summarizes the study and discusses future directions.

J. Chen is with the School of Science, Jiangnan University, Wuxi 214122, PR China (chenjing1981929@126.com).

J.X. Ma is with Key Laboratory of Advanced Process Control for Light Industry (Ministry of Education), Jiangnan University, Wuxi 214122, PR China (junxia.20@163.com).

M. Gan is with the College of Computer Science and Technology, Qingdao University, Qingdao 266071, PR China (aganmin@aliyun.com).

Q.M. Zhu is with the Department of Engineering Design and Mathematics, University of the West of England, Bristol BS16 1QY, UK (quan.zhu@uwe.ac.uk).

This work is supported by the National Natural Science Foundation of China (Nos. 62073082, 61973137 and 61803183), the Natural Science Foundation of Jiangsu Province (No. BK20201339), the Fundamental Research Funds for the Central Universities (No. JUSRP22016), and the Foundation of the Science and Technology on Near-Surface Detection Laboratory (No. TCGZ2019A001).

II. Hammerstein models and traditional identification algorithms

Consider the following Hammerstein model

$$\boldsymbol{\alpha}(z)y(t)=\boldsymbol{\beta}(z)g(u(t))+v(t), \tag{1}$$

where $y(t)$ is the output, $u(t)$ is the input with bounded values that is persistently excited, and $v(t)$ is a Gaussian white noise satisfying $v(t)\sim N(0,\sigma^2)$. $\boldsymbol{\alpha}(z)$ and $\boldsymbol{\beta}(z)$ are polynomials, which are expressed as

$$\begin{aligned}\boldsymbol{\alpha}(z)&=1+\alpha_1 z^{-1}+\cdots+\alpha_q z^{-q},\\ \boldsymbol{\beta}(z)&=\beta_0+\beta_1 z^{-1}+\cdots+\beta_m z^{-m},\end{aligned}$$

where $z^{-i}u(t)=u(t-i)$ and $z^{-i}y(t)=y(t-i)$, respectively. The structure of the nonlinear function $g(u(t))$ is known a priori and can be expressed as follows

$$g(u(t))=r_1 g_1(u(t))+\cdots+r_h g_h(u(t)).$$

In system identification, $\boldsymbol{\alpha}(z)$ is assumed to be stable, which implies that the bounded input $\{u(t)\}$ can generate a bounded output $\{y(t)\}$. The Hammerstein model is simplified as

$$\begin{aligned}y(t)&=\boldsymbol{\phi}^{\mathrm{T}}(t)\boldsymbol{\vartheta}+v(t),\\ \boldsymbol{\phi}(t)&=[-y(t-1),\cdots,-y(t-q),g_1(u(t)),\cdots,g_h(u(t)),\\ &\quad g_1(u(t-1)),\cdots,g_h(u(t-1)),\cdots,g_1(u(t-m)),\cdots,\\ &\quad g_h(u(t-m))]^{\mathrm{T}}\in\mathbb{R}^n,\ n=q+(m+1)h,\\ \boldsymbol{\vartheta}&=[\alpha_1,\cdots,\alpha_q,\beta_0 r_1,\cdots,\beta_0 r_h,\beta_1 r_1,\cdots,\\ &\quad \beta_1 r_h,\cdots,\beta_m r_1,\cdots,\beta_m r_h]^{\mathrm{T}}\in\mathbb{R}^n.\end{aligned}$$

Product terms exist in the parameter vector. Thus, to obtain a unique estimate for each parameter, either β_0 or r_1 should be fixed. Here, we assume that $\beta_0=1$. Then, the parameter vector is

$$\begin{aligned}\boldsymbol{\vartheta}=[&\alpha_1,\cdots,\alpha_q,r_1,\cdots,r_h,\beta_1 r_1,\cdots,\\ &\beta_1 r_h,\cdots,\beta_m r_1,\cdots,\beta_m r_h]^{\mathrm{T}}.\end{aligned}$$

Once the parameter vector estimate is obtained, we can get all the unknown parameter estimates by using the over-parametrization method [22], [23]. However, the product terms in the parameter vector may lead to multiple unknown parameters, which makes the traditional methods inefficient, especially for large-scale Hammerstein models.

Systems with irregular sampling data are common in modern engineering because of the absence of online measurements for certain quality variables [24]–[26]. In this paper, we assume that the input data $u(t), t=1,2,\cdots,L$ are sampled at a fixed interval Δt, while the output data $y(T_i), i=1,2,\cdots,N$ are irregularly sampled at time instant $t=T_i\Delta t$, T_i is an integer, and $L>n$ (that means the length of the sampled data must be larger than the number of unknown parameters) [3], [27].

Define

$$\begin{aligned}Y(L)&=[y(L),y(L-1),\cdots,y(1)]^{\mathrm{T}}\in\mathbb{R}^L,\\ \boldsymbol{\Phi}(L)&=[\boldsymbol{\phi}(L),\boldsymbol{\phi}(L-1).\cdots,\boldsymbol{\phi}(1)]^{\mathrm{T}}\in\mathbb{R}^{n\times L},\\ V(L)&=[v(L),v(L-1),\cdots,v(1)]^{\mathrm{T}}\in\mathbb{R}^L.\end{aligned}$$

The Hammerstein model is transformed into

$$Y(L)=\boldsymbol{\Phi}^{\mathrm{T}}(L)\boldsymbol{\vartheta}+V(L). \tag{2}$$

Since the output vector $Y(L)$ and the information matrix $\boldsymbol{\Phi}(L)$ contain unknown inner variables $y(T_i+j)$ $(i=1,\cdots,N-1, j=1,2,\cdots,T_{i+1}-T_i-1)$, the traditional GI algorithm is inefficient for the Hammerstein model. To overcome this difficulty, at iteration $k-1$, the output estimates $\hat{y}_{k-1}(T_i+j)$ estimated via the parameter estimates $\boldsymbol{\vartheta}_{k-1}$ are typically employed to replace the true outputs $y(T_i+j)$ [28], [29].

The GI algorithm for Hammerstein models with random sampled data is given as follows

$$\begin{aligned}\boldsymbol{\vartheta}_k&=\boldsymbol{\vartheta}_{k-1}+\hat{\boldsymbol{\Phi}}_{k-1}(L)[\hat{Y}_{k-1}(L)-\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}_{k-1}]\gamma_{k-1},\\ \hat{y}_{k-1}(T_i+j)&=\hat{\boldsymbol{\phi}}^{\mathrm{T}}_{k-1}(T_i+j)\boldsymbol{\vartheta}_{k-1},\ i=1,\cdots,N-1,\\ &\quad j=1,\cdots,T_{i+1}-T_i-1,\\ \hat{Y}_{k-1}(L)&=[y(L),\cdots,\hat{y}_{k-1}(T_i+j),\cdots,y(1)]^{\mathrm{T}},\\ \hat{\boldsymbol{\Phi}}_{k-1}(L)&=[\hat{\boldsymbol{\phi}}_{k-1}(L),\cdots,\hat{\boldsymbol{\phi}}_{k-1}(T_i+j),\cdots,\hat{\boldsymbol{\phi}}_{k-1}(1)]^{\mathrm{T}},\\ \hat{\boldsymbol{\phi}}_{k-1}(t)&=[-y(t-1),\cdots,-\hat{y}_{k-1}(T_i+j),\cdots,-y(t-n),\\ &\quad g_1(u(t)),\cdots,g_h(u(t)),g_1(u(t-1)),\cdots,\\ &\quad g_h(u(t-1)),\cdots,g_1(u(t-m)),\cdots,g_h(u(t-m))]^{\mathrm{T}}.\end{aligned}$$

γ_{k-1} is the step-size at iteration $k-1$, and satisfies

$$0<\gamma_{k-1}<\frac{2}{\lambda_{max}[\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)]},$$

where $\lambda_{max}[A]$ is the maximum eigenvalue of the matrix A [30].

Remark 1: Although the GI algorithm can estimate the parameters and the missing outputs, it has two disadvantages: (1) the convergence rate of the GI algorithm is slow because of its zigzagging nature; (2) the calculation of the maximum eigenvalue of a high-order matrix is challenging.

On the other hand, the LS algorithm for this Hammerstein model is summarized as

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

Remark 2: The LS algorithm can obtain the parameter estimates with faster convergence rates when compared with the GI algorithm, but it involves two challenges: (1) the matrix inverse calculation, especially for large-scale systems; (2) the assumption that the derivative function must have an analytical solution.

Remark 3: Since the information matrix $\hat{\boldsymbol{\Phi}}_{k-1}(L)$ varies with the output estimates, the step-size γ_{k-1} (GI) or the matrix inversion (LS) must be calculated at each iteration to maintain convergence of the algorithms, leading to heavy computational loads.

III. Two-direction-based GI algorithm

Inspired by the CGI algorithm, a novel GI algorithm that uses two directions at each iteration is developed in this section. This algorithm, named the two-direction-based GI (TD-GI) algorithm, avoids the maximum eigenvalue calculation and has a faster convergence rate.

A. One-direction-based GI algorithm

Define the cost function at iteration k as

$$J(\boldsymbol{\vartheta}^{od}_k)=\frac{1}{2}\|\hat{Y}_{k-1}(L)-\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^{od}_k\|^2, \tag{3}$$

where the index od means $one-direction$. Next, we will compute $\boldsymbol{\vartheta}^{od}_k$ based on the parameter vector estimate $\boldsymbol{\vartheta}^{od}_{k-1}$.

The gradient descent direction is given as

$$d_{k-1}=\hat{\boldsymbol{\Phi}}_{k-1}(L)[\hat{Y}_{k-1}(L)-\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^{od}_{k-1}]. \tag{4}$$

Then, the parameter estimates at iteration k can be computed by

$$\boldsymbol{\vartheta}^{od}_k=\boldsymbol{\vartheta}^{od}_{k-1}+d_{k-1}\gamma_{k-1}.$$

Substituting the above equation into Equation (3) yields

$$\begin{aligned}&J(\gamma_{k-1})\\ &=\frac{1}{2}\|\hat{Y}_{k-1}(L)-\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^{od}_{k-1}-\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d_{k-1}\gamma_{k-1}\|^2.\end{aligned} \tag{5}$$

Taking the derivative of $J(\gamma_{k-1})$ with respect to γ_{k-1} and then equating it to zero yields

$$\gamma_{k-1}=\frac{d^{\mathrm{T}}_{k-1}d_{k-1}}{d^{\mathrm{T}}_{k-1}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d_{k-1}}. \tag{6}$$

Remark 4: Although the one-direction-based GI (OD-GI) algorithm has the same structure as the GI algorithm, their step-size calculation procedures are different. The OD-GI algorithm does not require the calculation of the maximum eigenvalue, and can thus be extended to large-scale systems.

B. Two-direction-based GI algorithm

In the TD-GI algorithm, the optimal direction is a linear combination of the two neighboring negative gradient directions.

The two neighboring negative gradient directions are orthogonal; thus, the two directions at iteration $k-1$ are computed as follows

$$\begin{aligned} d^1_{k-1} &= d_{k-1},\\ d^2_{k-1} &= Qd^1_{k-1}-(Qd^1_{k-1},d^1_{k-1})\frac{d^1_{k-1}}{\|d^1_{k-1}\|^2}, \end{aligned}$$

in which $Q \neq E$ and $Q \in \mathbb{R}^{n\times n}$ is a nonsingular matrix, E is an identity matrix, and $(Qd^1_{k-1},d^1_{k-1}) = [d^1_{k-1}]^{\mathrm{T}}Qd^1_{k-1}$. Clearly, $[d^1_{k-1}]^{\mathrm{T}}d^2_{k-1}=0$. The parameter estimates at iteration k using the TD-GI algorithm are computed as

$$\boldsymbol{\vartheta}^{td}_k = \boldsymbol{\vartheta}^{td}_{k-1} + [d^1_{k-1}, d^2_{k-1}]\gamma_{k-1}, \tag{7}$$

where the index td means $two-direction$, and

$$\gamma_{k-1} = \begin{bmatrix} r^1_{k-1}\\ r^2_{k-1} \end{bmatrix}.$$

Substituting Equation (7) into Equation (3), and taking the derivative of $J(\gamma_{k-1})$ with respect to γ_{k-1} and then equating it to zero yields

$$\gamma_{k-1} = \begin{bmatrix} r^1_{k-1}\\ r^2_{k-1} \end{bmatrix} = \begin{bmatrix} a & b\\ b & c \end{bmatrix}^{-1} \begin{bmatrix} [d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}\\ 0 \end{bmatrix} = \begin{bmatrix} \frac{c[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ac-b^2}\\ \frac{-b[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ac-b^2} \end{bmatrix}, \tag{8}$$

where

$$\begin{aligned} a &= [d^1_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d^1_{k-1},\\ b &= [d^1_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d^2_{k-1},\\ c &= [d^2_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d^2_{k-1}. \end{aligned}$$

Therefore, the TD-GI algorithm is summarized as follows

$$\begin{aligned} \boldsymbol{\vartheta}^{td}_k &= \boldsymbol{\vartheta}^{td}_{k-1} + [d^1_{k-1}, d^2_{k-1}]\gamma_{k-1},\\ d^1_{k-1} &= \hat{\boldsymbol{\Phi}}_{k-1}(L)[\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^{td}_{k-1}],\\ d^2_{k-1} &= Qd^1_{k-1}-(Qd^1_{k-1},d^1_{k-1})\frac{d^1_{k-1}}{\|d^1_{k-1}\|^2},\\ \gamma_{k-1} &= \begin{bmatrix} \frac{c[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ac-b^2}\\ \frac{-b[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ac-b^2} \end{bmatrix}. \end{aligned}$$

Remark 5: In the TD-GI algorithm, the direction is a linear combination of the two orthogonal directions. Therefore, the convergence rate is faster than that of the OD-GI algorithm. The proof is presented in the following subsection.

C. Relationship between OD-GI and TD-GI algorithms

Assume that the parameter vector estimate at iteration $k-1$ satisfies $\boldsymbol{\vartheta}^{od}_{k-1} = \boldsymbol{\vartheta}^{td}_{k-1} = \boldsymbol{\vartheta}_{k-1}$.

Case 1: The cost function of the OD-GI algorithm : $J(\boldsymbol{\vartheta}^{od}_k)$
The cost function of the OD-GI algorithm at iteration k is

$$\begin{aligned} &J(\boldsymbol{\vartheta}^{od}_k) = J(\boldsymbol{\vartheta}_{k-1}) -\\ &[\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d_{k-1}\gamma_{k-1}\\ &+\frac{1}{2}d^{\mathrm{T}}_{k-1}\hat{\boldsymbol{\Phi}}_{k-1}\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d_{k-1}\gamma^2_{k-1}. \end{aligned}$$

Since $[\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L) = d^{\mathrm{T}}_{k-1}$, the above equation can be simplified as

$$\begin{aligned} &J(\boldsymbol{\vartheta}^{od}_k) = J(\boldsymbol{\vartheta}_{k-1}) - d^{\mathrm{T}}_{k-1}d_{k-1}\gamma_{k-1}\\ &+\frac{1}{2}d^{\mathrm{T}}_{k-1}\hat{\boldsymbol{\Phi}}_{k-1}\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d_{k-1}\gamma^2_{k-1}. \end{aligned} \tag{9}$$

Substituting Equation (6) into Equation (9) yields

$$J(\boldsymbol{\vartheta}^{od}_k) = J(\boldsymbol{\vartheta}_{k-1}) - \frac{1}{2}d^{\mathrm{T}}_{k-1}d_{k-1}\gamma_{k-1}, \tag{10}$$

which implies that

$$J(\boldsymbol{\vartheta}^{od}_k) \leqslant J(\boldsymbol{\vartheta}_{k-1}).$$

Case 2: The cost function of the TD-GI algorithm: $J(\boldsymbol{\vartheta}^{td}_k)$
The cost function of the TD-GI algorithm at iteration k is

$$\begin{aligned} &J(\boldsymbol{\vartheta}^{td}_k) = J(\boldsymbol{\vartheta}_{k-1}) -\\ &[\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)[d^1_{k-1}, d^2_{k-1}]\gamma_{k-1}\\ &+\frac{1}{2}\gamma^{\mathrm{T}}_{k-1}[d^1_{k-1}, d^2_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)[d^1_{k-1}, d^2_{k-1}]\gamma_{k-1}. \end{aligned}$$

Since d^1_{k-1} is orthogonal to d^2_{k-1}, the above equation can be simplified as

$$\begin{aligned} &J(\boldsymbol{\vartheta}^{td}_k) = J(\boldsymbol{\vartheta}_{k-1}) -\\ &[[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}, 0]\gamma_{k-1} + \frac{1}{2}\gamma^{\mathrm{T}}_{k-1}\begin{bmatrix} [d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}\\ 0 \end{bmatrix}. \end{aligned} \tag{11}$$

Substituting Equation (8) into Equation (11) yields

$$J(\boldsymbol{\vartheta}^{td}_k) = J(\boldsymbol{\vartheta}_{k-1}) - \frac{1}{2}[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}\frac{c[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ac-b^2}. \tag{12}$$

This demonstrates that

$$J(\boldsymbol{\vartheta}^{td}_k) \leqslant J(\boldsymbol{\vartheta}_{k-1}).$$

Remark 6: Let $\boldsymbol{\vartheta}_k = \boldsymbol{\vartheta}^{od}_k$ or $\boldsymbol{\vartheta}^{td}_k$; then, the cost function satisfies $J(\boldsymbol{\vartheta}_k) \leqslant J(\boldsymbol{\vartheta}_{k-1})$, which implies that both the OD-GI and TD-GI algorithms converge.

According to Equations (6) and (8), we have

$$\begin{aligned} \frac{1}{2}d^{\mathrm{T}}_{k-1}d_{k-1}r_{k-1} &= \frac{1}{2}d^{\mathrm{T}}_{k-1}d_{k-1}\frac{d^{\mathrm{T}}_{k-1}d_{k-1}}{a}\\ &\leqslant \frac{1}{2}[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}\frac{c[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ac-b^2}. \end{aligned}$$

It gives rise to

$$J(\boldsymbol{\vartheta}^{td}_k) \leqslant J(\boldsymbol{\vartheta}^{od}_k). \tag{13}$$

Remark 7: Inequality (13) indicates that the TD-GI algorithm has a faster convergence rate than that of the OD-GI algorithm.

IV. Multi-direction-based GI algorithm

Remark 7 shows that the TD-GI algorithm has a faster convergence rate than that of the OD-GI algorithm. Section IV focuses on whether the convergence rate can be improved by increasing the number of directions.

A. Three-direction-based GI algorithm

Let the direction be $[d^1_{k-1}, d^2_{k-1}, d^3_{k-1}]$, $[d^i_{k-1}]^{\mathrm{T}}d^j_{k-1} = 0$, $i \neq j$. The parameter estimates $\boldsymbol{\vartheta}^{thd}_k$ are updated by

$$\boldsymbol{\vartheta}^{thd}_k = \boldsymbol{\vartheta}^{thd}_{k-1} + [d^1_{k-1}, d^2_{k-1}, d^3_{k-1}]\gamma_{k-1},$$

where the index thd means $three-direction$. The associated step-size is computed by

$$\gamma_{k-1} = \begin{bmatrix} a & b & d\\ b & c & f\\ d & f & e \end{bmatrix}^{-1} \begin{bmatrix} [d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}\\ 0\\ 0 \end{bmatrix}, \tag{14}$$

where

$$\begin{aligned} a &= [d^1_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d^1_{k-1},\\ b &= [d^1_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d^2_{k-1},\\ c &= [d^2_{k-1}]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}(L)\hat{\boldsymbol{\Phi}}^{\mathrm{T}}_{k-1}(L)d^2_{k-1}, \end{aligned}$$

$$\begin{aligned} d &= [d^1_{k-1}]^{\mathrm{T}}\hat{\mathbf{\Phi}}_{k-1}(L)\hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)d^3_{k-1},\\ e &= [d^3_{k-1}]^{\mathrm{T}}\hat{\mathbf{\Phi}}_{k-1}(L)\hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)d^3_{k-1},\\ f &= [d^2_{k-1}]^{\mathrm{T}}\hat{\mathbf{\Phi}}_{k-1}(L)\hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)d^3_{k-1}. \end{aligned}$$

Let $\boldsymbol{\vartheta}^{td}_{k-1} = \boldsymbol{\vartheta}^{thd}_{k-1} = \boldsymbol{\vartheta}_{k-1}$, the following theorem can be obtained.

Theorem 1: Assume that the cost function at iteration k of the three-direction-based GI (ThD-GI) algorithm is

$$J(\boldsymbol{\vartheta}^{thd}_k) = \frac{1}{2}\|\hat{Y}_{k-1}(L) - \hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^{thd}_k\|^2,$$

and the cost function at iteration k of the TD-GI algorithm is

$$J(\boldsymbol{\vartheta}^{td}_k) = \frac{1}{2}\|\hat{Y}_{k-1}(L) - \hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^{td}_k\|^2.$$

Then, the following inequality holds

$$J(\boldsymbol{\vartheta}^{thd}_k) \leqslant J(\boldsymbol{\vartheta}^{td}_k).$$

Proof: The cost function $J(\boldsymbol{\vartheta}^{thd}_k)$ is computed by

$$\begin{aligned} &J(\boldsymbol{\vartheta}^{thd}_k) = J(\boldsymbol{\vartheta}_{k-1}) - \\ &\frac{1}{2}[d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}\frac{[ce-f^2][d^1_{k-1}]^{\mathrm{T}}d^1_{k-1}}{ace+2bfd-d^2c-b^2e-f^2a}. \end{aligned}$$

Note that

$$\frac{ce-f^2}{ace+2bfd-d^2c-b^2e-f^2a} \geqslant \frac{c}{ac-b^2}.$$

This demonstrates that

$$J(\boldsymbol{\vartheta}^{thd}_k) \leqslant J(\boldsymbol{\vartheta}^{td}_k),$$

which implies that the GI algorithm with three directions has a faster convergence rate than that of the GI algorithm with two directions. ■

Two questions naturally arise: whether the number of directions have an upper bound; whether the convergence rates are improved with an increasing number of directions. The following two subsections focus on these aspects.

B. Multi-direction-based GI algorithm

Assume that the direction at iteration $k-1$ is $\mathbf{d}_{k-1} = [d^1_{k-1}, d^2_{k-1}, \cdots, d^l_{k-1}]$, $[d^i_{k-1}]^{\mathrm{T}}d^j_{k-1} = 0, i \neq j$. The parameter estimates at iteration k are computed by

$$\boldsymbol{\vartheta}^l_k = \boldsymbol{\vartheta}^l_{k-1} + [d^1_{k-1}, d^2_{k-1}, \cdots, d^l_{k-1}]\gamma_{k-1}, \tag{15}$$

where

$$\gamma_{k-1} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,l} \\ a_{1,2} & a_{2,2} & \cdots & a_{2,l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1,l} & a_{2,l} & \cdots & a_{l,l} \end{bmatrix}^{-1} \begin{bmatrix} [d^1_{k-1}]^{\mathrm{T}}d^1_{k-1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{16}$$

$$a_{i,j} = [d^i_{k-1}]^{\mathrm{T}}\hat{\mathbf{\Phi}}_{k-1}(L)\hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)d^j_{k-1}. \tag{17}$$

To prove that $J(\boldsymbol{\vartheta}^l_k) \leqslant J(\boldsymbol{\vartheta}^{l-1}_k)$, the following lemmas are given.

For simplicity, let the matrix $A \in \mathbb{R}^{n\times n}$ be expressed by $[a_{i,j}]_{1:n,1:n}$, and $a_{i,j}$ be the element in the ith row and jth column. $\{1:n, 1:n\}$ indicates that the first and last elements are $a_{1,1}$ and $a_{n,n}$, respectively.

Lemma 1: Assume that $A = [a_{i,j}]_{1:n,1:n}$, $\mathrm{rank}(A) = n$, $n \geqslant 2$, and $B = A^{-1}$ is written as $B = [b_{i,j}]_{1:n,1:n}$. Then, the first element $b_{1,1}$ in B is

$$b_{1,1} = \frac{|C|}{|A|},$$

where $C = [a_{i,j}]_{2:n,2:n} \in \mathbb{R}^{(n-1)\times(n-1)}$.

Proof: Based on the matrix theory, matrix B can be computed as

$$B = \frac{A^*}{|A|},$$

where A^* is the adjoint matrix of A. Then, it follows that

$$b_{1,1} = \frac{|C|}{|A|}.$$

■

Lemma 2: Assume that $A \in \mathbb{R}^{n\times n}, n \geqslant 2$ is a symmetric positive-determined (SPD) matrix. There exists a nonsingular matrix $P \in \mathbb{R}^{n\times n}$, which maintains the following equality

$$P^{\mathrm{T}}AP = E,$$

where $E \in \mathbb{R}^{n\times n}$ is an identity matrix.

(The derivation is straightforward and hence omitted.)

Lemma 3: Assume that $\mathbf{x} = [x_1, x_2, \cdots, x_l] \in \mathbb{R}^{n\times l}$, $\mathrm{rank}(\mathbf{x}) = l$, $2 \leqslant l < n$, and $A \in \mathbb{R}^{n\times n}$ is an SPD matrix. Then,

$$C = \mathbf{x}^{\mathrm{T}}A\mathbf{x}$$

is also an SPD matrix.

(The proof of Lemma 3 is given in Appendix A.)

Lemma 4: Assume that all the matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{(n-1)\times(n-1)}$, $C \in \mathbb{R}^{(n-1)\times(n-1)}$, and $D \in \mathbb{R}^{(n-2)\times(n-2)}, n \geqslant 3$ are symmetric and positive definite, and are written as $A = [a_{i,j}]_{1:n,1:n}$, $B = [a_{i,j}]_{2:n,2:n}$, $C = [a_{i,j}]_{1:n-1,1:n-1}$, and $D = [a_{i,j}]_{2:n-1,2:n-1}$. Then, the following inequality holds

$$\frac{|B|}{|A|} \geqslant \frac{|D|}{|C|}.$$

(Refer to the detailed derivation in Appendix B.)

Let $\boldsymbol{\vartheta}^{l-1}_{k-1} = \boldsymbol{\vartheta}^l_{k-1} = \boldsymbol{\vartheta}_{k-1}$. Then, based on Lemmas 1-4, the following theorem is obtained.

Theorem 2: For the Hammerstein model proposed in (2), the parameter vector estimate $\boldsymbol{\vartheta}^l_k$ based on direction $\mathbf{d}^l_{k-1} = [d^1_{k-1}, d^2_{k-1}, \cdots, d^l_{k-1}]$, $l \geqslant 3$ is updated by Equations (15)-(17), while the parameter vector estimate based on direction $\mathbf{d}^{l-1}_{k-1} = [d^1_{k-1}, d^2_{k-1}, \cdots, d^{l-1}_{k-1}]$ is $\boldsymbol{\vartheta}^{l-1}_k$. The cost functions using these two estimates are $J(\boldsymbol{\vartheta}^l_k)$ and $J(\boldsymbol{\vartheta}^{l-1}_k)$, respectively. Then, the following inequality holds

$$J(\boldsymbol{\vartheta}^l_k) \leqslant J(\boldsymbol{\vartheta}^{l-1}_k).$$

(The detailed derivation is presented in Appendix C.)

Remark 8: Theorem 2 shows that having more directions leads to faster convergence rates, albeit at the cost of heavier computational loads.

The missing outputs at iteration $k-1$ are approximated by $\hat{y}_{k-1}(T_i+j) = \hat{\boldsymbol{\phi}}^{\mathrm{T}}_{k-1}(T_i+j)\boldsymbol{\vartheta}_{k-1}$. Then, the steps of the MUL-D-GI algorithm are listed as follows:

C. Relationship between the two neighboring directions in the MUL-D-GI algorithm

Case 1: $l = n$

Let the direction be $[d^1_{k-1}, \cdots, d^n_{k-1}]$. Then, the parameter estimates are

$$\boldsymbol{\vartheta}^n_k = \boldsymbol{\vartheta}^n_{k-1} + [d^1_{k-1}, \cdots, d^n_{k-1}]\gamma_{k-1}.$$

The source direction at iteration k is

$$\begin{aligned} d^1_k &= \hat{\mathbf{\Phi}}_{k-1}(L)[\hat{Y}_{k-1}(L) - \hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)\boldsymbol{\vartheta}^n_k] \\ &= d^1_{k-1} - \hat{\mathbf{\Phi}}_{k-1}(L)\hat{\mathbf{\Phi}}^{\mathrm{T}}_{k-1}(L)[d^1_{k-1}, \cdots, d^n_{k-1}]\gamma_{k-1} \end{aligned}$$

```
MUL-D-GI algorithm
Initialise $\boldsymbol{\vartheta}_0^l = \mathbf{1}/p_0$, $Q \in \mathbb{R}^{n\times n}, Q \neq E$ is a nonsingular
        matrix, $l$ is an integer
Collect measurable data $u(1), \cdots, u(L)$ and $y(1), \cdots, y(L)$
repeat
  for $k = 1, 2, \cdots$, do
    Compute $\hat{y}_{k-1}(T_i + j), i = 1, \cdots, N-1,$
            $j = 1, \cdots, T_{i+1} - T_i - 1$
    Form $\hat{Y}_{k-1}(L)$
    Form $\hat{\boldsymbol{\phi}}_{k-1}(t), t = 1, \cdots, L$, then form $\hat{\mathbf{\Phi}}_{k-1}(L)$
    Compute $d_{k-1}^i, i = 1, \cdots, l$
    Compute $\gamma_{k-1}$
    Update $\boldsymbol{\vartheta}_k^l$
  end
until convergence
```

$$= d_{k-1}^1 - \{[d_{k-1}^1, \cdots, d_{k-1}^n]^{\mathrm{T}}\}^{-1} \begin{bmatrix} [d_{k-1}^1]^{\mathrm{T}} d_{k-1}^1 \\ 0 \\ \cdots \\ 0 \end{bmatrix}. \quad (18)$$

Define

$$D_n = [d_{k-1}^1, \cdots, d_{k-1}^n].$$

Then, we have

$$D_n^{\mathrm{T}} D_n = \mathrm{diag}[\lambda_1, \cdots, \lambda_n],$$

where $\lambda_i = [d_{k-1}^i]^{\mathrm{T}} d_{k-1}^i$. This gives rise to

$$[D_n^{\mathrm{T}}]^{-1} = D_n \left\{\mathrm{diag}[\lambda_1, \cdots, \lambda_n]\right\}^{-1}. \quad (19)$$

Substituting Equation (19) into Equation (18) yields,

$$d_k^1 = d_{k-1}^1 - d_{k-1}^1 = \mathbf{0}.$$

Remark 9: When the number of directions is equal to the number of unknown parameters, the MUL-D-GI algorithm can yield the optimal parameter estimates in one iteration via the current input-output data, that is, the upper bound of the directions is $l = n$.

Case 2: $l < n$

Assume that all outputs are measurable, and $l < n$. The direction at iteration $k-1$ is

$$\mathbf{d}_{k-1} = [d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l].$$

The source direction at iteration k is

$$d_k^1 = \mathbf{\Phi}(L)[Y(L) - \mathbf{\Phi}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_k] \neq \mathbf{0}.$$

Multiplying the above equation on both sides with $\mathbf{d}_{k-1}^{\mathrm{T}}$ yields

$$\begin{aligned} \mathbf{d}_{k-1}^{\mathrm{T}} d_k^1 &= \mathbf{d}_{k-1}^{\mathrm{T}} \mathbf{\Phi}(L)[Y(L) - \mathbf{\Phi}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_k] \\ &= \mathbf{d}_{k-1}^{\mathrm{T}} \mathbf{\Phi}(L)[Y(L) - \mathbf{\Phi}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_{k-1} - \mathbf{\Phi}^{\mathrm{T}}(L) \times \\ &\quad [d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]\gamma_{k-1}] \\ &= [[d_{k-1}^1]^{\mathrm{T}} d_{k-1}^1, 0, \cdots, 0]^{\mathrm{T}} - [[d_{k-1}^1]^{\mathrm{T}} d_{k-1}^1, 0, \cdots, 0]^{\mathrm{T}} \times \\ &\quad \mathbf{\Phi}(L)\mathbf{\Phi}^{\mathrm{T}}(L)[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]\gamma_{k-1}. \quad (20) \end{aligned}$$

According to Equations (16) and (17), the above equation can be simplified as

$$\begin{aligned} \mathbf{d}_{k-1}^{\mathrm{T}} d_k^1 &= [[d_{k-1}^1]^{\mathrm{T}} d_{k-1}^1, 0, \cdots, 0]^{\mathrm{T}} - [[d_{k-1}^1]^{\mathrm{T}} d_{k-1}^1, 0, \cdots, 0]^{\mathrm{T}} \\ &= [0, 0, \cdots, 0]^{\mathrm{T}}. \quad (21) \end{aligned}$$

Remark 10: When the system does not have hidden variables, the source direction d_k^1 at iteration k is orthogonal to the previous direction $\mathbf{d}_{k-1}$. This implies that the MUL-D-GI algorithm is convergent.

Remark 11: When $l = 1$, the MUL-D-GI algorithm is equivalent to the GI algorithm. In contrast, when $l = n$, the MUL-D-GI algorithm can be regarded as the LS algorithm. Therefore, the MUL-D-GI algorithm establishes a bridge that links the GI and LS algorithms; as depicted in Fig. 1.

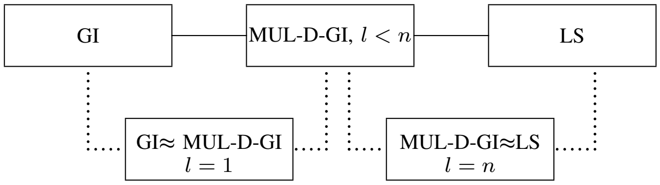

Fig. 1. Relationship among these three algorithms

V. Simulation Examples

A. Example 1

Consider the Hammerstein model in [31],

$$\begin{aligned} &\boldsymbol{\alpha}(z)y(t) = \boldsymbol{\beta}(z)g(u(t)) + v(t),\ g(u(t)) = 0.9u(t) + 0.4u^2(t), \\ &\boldsymbol{\alpha}(z) = 1 + \alpha_1 z^{-1} = 1 + 0.8z^{-1},\ \boldsymbol{\beta}(z) = 1 + \beta_1 z^{-1} = 1 + 0.3z^{-1}, \\ &\boldsymbol{\phi}(t) = [-y(t-1), u(t), u^2(t), u(t-1), u^2(t-1)]^{\mathrm{T}}, \\ &\boldsymbol{\vartheta} = [\alpha_1, r_1, r_2, \beta_1 r_1, \beta_1 r_2]^{\mathrm{T}} = [0.8, 0.9, 0.4, 0.27, 0.12]^{\mathrm{T}}. \end{aligned}$$

The input $\{u(t)\}$ satisfies $N(0,1)$, $\{v(t)\}$ is a white noise satisfying $N(0, 0.1^2)$, and the number of the input-output data is $L = 500$. The outputs at the sampling instants $1, 3, 5, \cdots, 499$ are measurable, while the others are not.

Apply the traditional GI algorithm ($\gamma_{k-1} = \frac{1}{\lambda_{max}[\hat{\mathbf{\Phi}}_{k-1}(L)\hat{\mathbf{\Phi}}_{k-1}^{\mathrm{T}}(L)]}$) and the MUL-D-GI algorithm (with different directions: OD-GI, TD-GI, and ThD-GI) to the Hammerstein model. The estimation errors $\delta = \frac{\|\boldsymbol{\vartheta}_k - \boldsymbol{\vartheta}\|}{\|\boldsymbol{\vartheta}\|}$ are depicted in Fig. 2. The parameter estimates and their errors using the traditional GI and OD-GI algorithms are presented in Table I. The output estimates ($t = 450 - 500$) using the traditional GI and OD-GI algorithms are shown in Fig. 3.

Furthermore, the parameter estimation errors of the three algorithms for a batch of dynamic data are depicted in Fig. 4. The length of this batch of dynamic data is fixed, but the data window moves forward dynamically. This implies that when new sampling data are collected, the oldest data are removed. Therefore, the data participating in the iterative algorithm are a batch of the latest data.

TABLE I
Parameter estimates and their estimation errors

Algorithm	k	α_1	r_1	r_2	$\beta_1 r_1$	$\beta_2 r_2$	δ (%)
GI	50	0.7473	0.3525	0.5057	-0.0781	0.1529	50.6860
	200	0.7851	0.7436	0.4263	0.0100	0.1626	23.6356
	400	0.7948	0.8749	0.4039	0.1850	0.1352	6.9213
	500	0.7973	0.8926	0.4015	0.2268	0.1288	3.4384
OD-GI	50	0.7960	0.9003	0.3991	0.2393	0.1233	2.3896
	200	0.7980	0.9045	0.3994	0.2719	0.1184	0.4239
	400	0.7980	0.9045	0.3994	0.2719	0.1184	0.4239
	500	0.7980	0.9045	0.3994	0.2719	0.1184	0.4239
True Values		0.8000	0.9000	0.4000	0.2700	0.1200	

Then, the following findings are obtained:

(1) The parameter estimates asymptotically converge to their true values using both the GI and OD-GI algorithms, as shown in Fig. 2 and Table I;

(2) The convergence rates are improved with an increasing number of directions, as shown in Fig. 2;

(3) The output estimates based on the GI and OD-GI algorithms can capture their true values, see Fig. 3;

(4) The ThD-GI algorithm has the minimum estimation error for the same number of iterations, as depicted in Fig. 4.

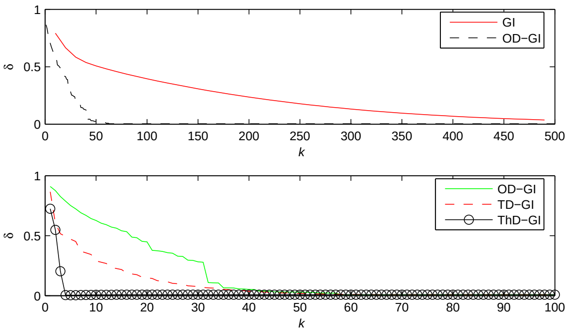

Fig. 2. Parameter estimation errors

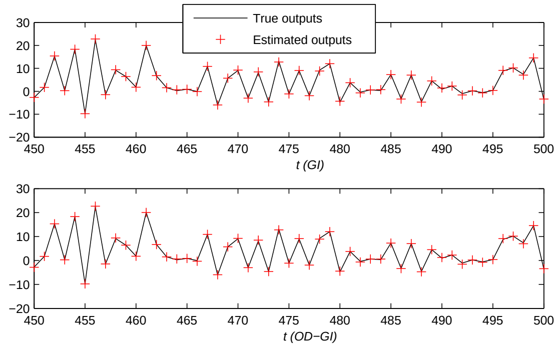

Fig. 3. True outputs and their estimates (t=450-500)

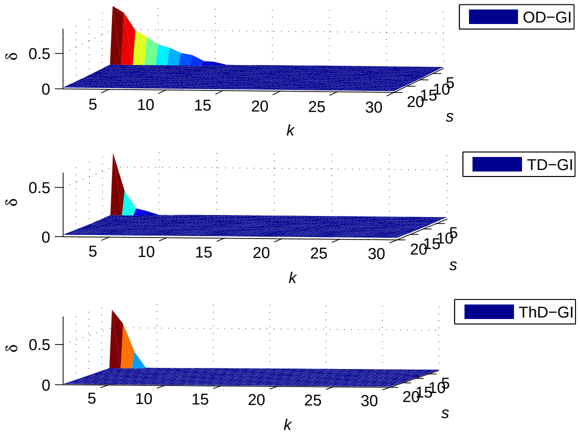

Fig. 4. Parameter estimation error for a batch of dynamic data

B. Example 2

Consider the following Hammerstein model with 17 unknown parameters,

$$
\begin{aligned}
&\boldsymbol{\alpha}(z)y(t) = \boldsymbol{\beta}(z)g(u(t)) + v(t),\\
&\boldsymbol{\alpha}(z) = 1 - 0.2z^{-1},\\
&\boldsymbol{\beta}(z) = 0.4z^{-1} - 0.2z^{-2} + 0.6z^{-3} - 0.3z^{-4} + 0.3z^{-5} +\\
&\qquad 0.2z^{-6} + 0.2z^{-7} - 0.5z^{-8},\\
&g(u(t)) = r_1u(t) + r_2u^2(t) = u(t) + 0.5u^2(t),\\
&\boldsymbol{\phi}(t) = [-y(t-1), u(t-1), u^2(t-1), \cdots, u(t-8), u^2(t-8)]^{\mathrm{T}},\\
&\boldsymbol{\vartheta} = [\alpha_1, \beta_1 r_1, \beta_1 r_2, \cdots, \beta_8 r_1, \beta_8 r_2]^{\mathrm{T}},\\
&u \sim N(0,1), \quad v \sim N(0, 0.1^2).
\end{aligned}
$$

We sample $L = 100$ data points for fast-rate inputs and sample four outputs at every five fast-rate sampling intervals. Thus, the output data $y(5), y(10), y(15), \cdots, y(95), y(100)$ are unmeasurable, while the other input and output data are available.

Monte Carlo simulations (with 100 different noise seeds) were performed based on the LS, GI, and MUL-D-GI (l =3) algorithms. The parameter estimation errors are shown in Fig. 5. The elapsed time of these three algorithms is displayed in Table II (**by Intel(R) Core(TM) i5-7220U: 2.50GHz, 2.71GHz; RAM: 8.0 GB; Windows 10**). This example shows that all three algorithms are robust to noise, and the MUL-D-GI algorithm has the shortest elapsed time.

The computational flops using (only the number of multiplications and divisions) the LS, GI, and MUL-D-GI (l =2, 3 and 17) algorithms are shown in Table III (with 30 iterations). Since we should compute the maximum eigenvalue of the information matrix at each iteration, it will lead to heavy computational efforts of the GI algorithm. Table III demonstrates that the MUL-D-GI algorithm has the smallest computational efforts when $l < n$, and the computational efforts of the MUL-D-GI algorithm increase with an increasing number of l. However, when $l = n$, the MUL-D-GI algorithm has much heavier computational efforts than those of the LS algorithm because of the direction calculation at each iteration.

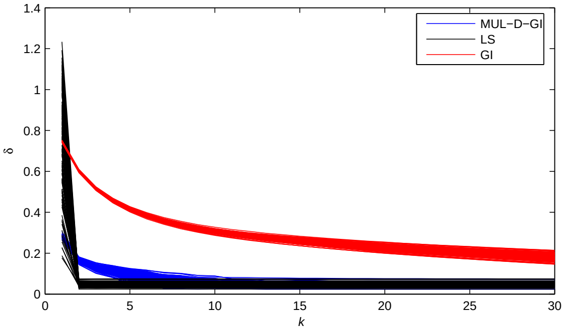

Fig. 5. Parameter estimation errors δ versus k

TABLE II
ELAPSED TIMES

Algorithm	LS	GI	MUL-D-GI ($l = 3$)
Time (second)	4.529	5.575	3.892

TABLE III
THE COMPUTATIONAL EFFORTS

Algorithm	LS	GI	MUL-D-GI ($l = 2$)	($l = 3$)	($l = 17$)
Flop	1 074 060	1 166 040	982 050	994 890	1 420 350

VI. CONCLUSIONS

The MUL-D-GI algorithm proposed in this study benefits from the advantages of both GI and LS methods. Therefore, not only can it be applied to nonlinear systems with complicated structures but it can also be extended to large-scale systems. The properties of the MUL-D-GI algorithm are also stated, which can help researchers and engineers select the optimal number of directions for their system identification applications.

Our study shows that there exists a potential middle ground between the LS and GI methods. Research the properties of this common ground reflects a deep understanding and comprehension of LS and GI algorithms. It is believed that this study is a good outline that provides easy and general solutions for parameter estimation in different types of systems.

Acknowledgements

The authors would like to express their gratitude to the editors and anonymous reviewers for their helpful comments and constructive suggestions regarding the revision of this paper.

Appendix A

Proof of Lemma 3. Let

$$B = [b_1, b_2, \cdots, b_l]^{\mathrm{T}} \in \mathbb{R}^l,$$

and let there be at least one $b_i \neq 0$.

Because x_1, x_2, $\cdots$, and x_l are linearly independent, we have

$$\mathbf{x}B = b_1x_1 + b_2x_2 + \cdots + b_lx_l \neq \mathbf{0},$$

where $\mathbf{0} = [0, 0, \cdots, 0]^{\mathrm{T}} \in \mathbb{R}^n$. It follows that

$$B^{\mathrm{T}}\mathbf{x}^{\mathrm{T}}A\mathbf{x}B = [\mathbf{x}B]^{\mathrm{T}}A\mathbf{x}B.$$

Since A is an SPD matrix and $\mathbf{x}B \neq \mathbf{0}$,

$$[\mathbf{x}B]^{\mathrm{T}}A\mathbf{x}B > 0;$$

that is

$$B^{\mathrm{T}}[\mathbf{x}^{\mathrm{T}}A\mathbf{x}]B > 0,$$

which implies that $\mathbf{x}^{\mathrm{T}}A\mathbf{x}$ is an SPD matrix.

Appendix B

Proof of Lemma 4. Since B is an SPD matrix, based on Lemma 2, there exists a nonsingular matrix $G_1 \in \mathbb{R}^{(n-1)\times(n-1)}$ which can maintain $G_1^{\mathrm{T}}BG_1 = E \in \mathbb{R}^{(n-1)\times(n-1)}$. The matrix A can be rewritten as

$$A = \begin{bmatrix} a_{1,1} & \rho_1^{\mathrm{T}} \\ \rho_1 & B \end{bmatrix},$$

where $\rho_1 = [a_{1,2}, a_{1,3}, \cdots, a_{1,n}]^{\mathrm{T}}$. Introduce the matrix

$$F_1 = \begin{bmatrix} 1 & \mathbf{0}^{\mathrm{T}} \\ \mathbf{0} & G_1 \end{bmatrix},$$

in which $\mathbf{0} = [0, \cdots, 0]^{\mathrm{T}} \in \mathbb{R}^{n-1}$. This gives rise to

$$F_1^{\mathrm{T}}AF_1 = \begin{bmatrix} a_{1,1} & \rho_1^{\mathrm{T}}G_1 \\ G_1^{\mathrm{T}}\rho_1 & E \end{bmatrix}.$$

Assume that

$$H = \begin{bmatrix} 1 & \mathbf{0}^{\mathrm{T}} \\ -G_1^{\mathrm{T}}\rho_1 & E \end{bmatrix},$$

then, it follows that

$$H^{\mathrm{T}}F_1^{\mathrm{T}}AF_1H = \begin{bmatrix} a_{1,1} - \rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1 & \mathbf{0}^{\mathrm{T}} \\ \mathbf{0} & E \end{bmatrix}.$$

Because $|H| = 1$, we obtain

$$\begin{aligned} \frac{|B|}{|A|} &= \frac{|G_1^{\mathrm{T}}||B||G_1|}{|H^{\mathrm{T}}||F_1^{\mathrm{T}}||A||F_1||H|} = \frac{|G_1^{\mathrm{T}}BG_1|}{|H^{\mathrm{T}}F_1^{\mathrm{T}}AF_1H|} \\ &= \frac{1}{a_{1,1} - \rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1}. \end{aligned} \tag{22}$$

In the same way, the determinant $\frac{|D|}{|C|}$ is computed as

$$\frac{|D|}{|C|} = \frac{1}{a_{1,1} - \rho_2^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2},$$

where $G_2 \in \mathbb{R}^{(n-2)\times(n-2)}$, $G_2^{\mathrm{T}}DG_2 = E \in \mathbb{R}^{(n-2)\times(n-2)}$, and $\rho_2 = [a_{1,2}, a_{1,3}, \cdots, a_{1,n-1}]^{\mathrm{T}}$.

In order to prove $\frac{|B|}{|A|} \geqslant \frac{|D|}{|C|}$, we derive

$$\frac{1}{a_{1,1} - \rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1} \geqslant \frac{1}{a_{1,1} - \rho_2^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2}, \tag{23}$$

where $a_{1,1} - \rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1 > 0$, $a_{1,1} - \rho_2^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2 > 0$ and $a_{1,1} > 0$. It shows that (23) is equivalent to

$$\rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1 \geqslant \rho_2^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2.$$

Since $G_1^{\mathrm{T}}BG_1 = E$, it follows that $G_1G_1^{\mathrm{T}} = B^{-1}$. Let

$$B = \begin{bmatrix} D & r \\ r^{\mathrm{T}} & a_{n,n} \end{bmatrix},$$

where

$$r = [a_{2,n}, a_{3,n}, \cdots, a_{n-1,n}]^{\mathrm{T}}.$$

Define

$$P_1 = \begin{bmatrix} G_2 & \mathbf{0}^{\mathrm{T}} \\ \mathbf{0} & 1 \end{bmatrix}.$$

Transform the matrix B into

$$P_1^{\mathrm{T}}BP_1 = \begin{bmatrix} E & G_2^{\mathrm{T}}r \\ r^{\mathrm{T}}G_2 & a_{n,n} \end{bmatrix}.$$

Define

$$P_2 = \begin{bmatrix} E & -G_2^{\mathrm{T}}r \\ \mathbf{0} & 1 \end{bmatrix}.$$

Then, we obtain

$$P_2^{\mathrm{T}}P_1^{\mathrm{T}}BP_1P_2 = \begin{bmatrix} E & 0 \\ \mathbf{0} & a_{n,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}r \end{bmatrix} = \Lambda. \tag{24}$$

Based on the above equation, it gives

$$B^{-1} = P_1P_2\Lambda^{-1}P_2^{\mathrm{T}}P_1^{\mathrm{T}},$$

which means that

$$\rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1 = \rho_1^{\mathrm{T}}B^{-1}\rho_1 = \rho_1^{\mathrm{T}}P_1P_2\Lambda^{-1}P_2^{\mathrm{T}}P_1^{\mathrm{T}}\rho_1 = x^{\mathrm{T}}\Lambda^{-1}x, \tag{25}$$

where

$$\begin{aligned} x = P_2^{\mathrm{T}}P_1^{\mathrm{T}}\rho_1 &= \begin{bmatrix} E & 0 \\ -r^{\mathrm{T}}G_2 & 1 \end{bmatrix}\begin{bmatrix} G_2^{\mathrm{T}} & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix}\begin{bmatrix} \rho_2 \\ a_{1,n} \end{bmatrix} \\ &= \begin{bmatrix} G_2^{\mathrm{T}}\rho_2 \\ a_{1,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2 \end{bmatrix}. \end{aligned} \tag{26}$$

According to Equation (24), the following equality holds

$$|P_2|^2|P_1|^2|B| = a_{n,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}r > 0,$$

and Equation (25) can be transformed into

$$\begin{aligned} \rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1 &= x^{\mathrm{T}}\Lambda^{-1}x = [\rho_2^{\mathrm{T}}G_2, a_{1,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2] \times \\ &\begin{bmatrix} E & 0 \\ 0 & \frac{1}{a_{n,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}r} \end{bmatrix}\begin{bmatrix} G_2^{\mathrm{T}}\rho_2 \\ a_{1,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2 \end{bmatrix} \\ &= \rho_2^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2 + \frac{(a_{1,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2)^2}{a_{n,n} - r^{\mathrm{T}}G_2G_2^{\mathrm{T}}r}, \end{aligned}$$

which implies that

$$\rho_1^{\mathrm{T}}G_1G_1^{\mathrm{T}}\rho_1 \geqslant \rho_2^{\mathrm{T}}G_2G_2^{\mathrm{T}}\rho_2.$$

Therefore, we have

$$\frac{|B|}{|A|} \geqslant \frac{|D|}{|C|}.$$

Appendix C

Proof of Theorem 2. The cost function of the parameter estimates $\hat{\boldsymbol{\vartheta}}_k^l$ is

$$J(\boldsymbol{\vartheta}_k^l) = \frac{1}{2}\|\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_k^l\|^2.$$

Substituting Equation (15) into the above equation yields

$$\begin{aligned} J(\boldsymbol{\vartheta}_k^l) = \frac{1}{2}&\|\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_{k-1}\|^2 - \\ &[\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_{k-1}]^{\mathrm{T}} \times \\ &\hat{\boldsymbol{\Phi}}_{k-1}(L)[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]\gamma_{k-1}^l + \end{aligned}$$

$$\frac{1}{2}[\gamma_{k-1}^l]^{\mathrm{T}}[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]^{\mathrm{T}} \times$$
$$\hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\hat{\boldsymbol{\Phi}}_{k-1}(L)[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]\gamma_{k-1}^l.$$

According to Equations (15)-(17), it gives rise to

$$\begin{aligned} J(\boldsymbol{\vartheta}_k^l) = {} & \frac{1}{2}\|\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_{k-1}\|^2 - \\ & [[d_{k-1}^1]^{\mathrm{T}}d_{k-1}^1, 0, \cdots, 0]\gamma_{k-1}^l + \\ & \frac{1}{2}[\gamma_{k-1}^l]^{\mathrm{T}}[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]^{\mathrm{T}} \times \\ & \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\hat{\boldsymbol{\Phi}}_{k-1}(L)[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]\gamma_{k-1}^l. \end{aligned} \tag{27}$$

Based on Lemma 3, we observe that the matrix

$$[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]^{\mathrm{T}}\hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\hat{\boldsymbol{\Phi}}_{k-1}(L)[d_{k-1}^1, d_{k-1}^2, \cdots, d_{k-1}^l]$$

is an SPD matrix, which implies that its inversion matrix is also an SPD matrix.

It follows that Equation (27) can be simplified as

$$J(\boldsymbol{\vartheta}_k^l) = \frac{1}{2}\|\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_{k-1}\|^2 - \frac{\rho_{1,1}^l}{2}\{[d_{k-1}^1]^{\mathrm{T}}d_{k-1}^1\}^2, \tag{28}$$

where $\rho_{1,1}^l$ is the first element of the following matrix

$$A^{-1} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,l} \\ a_{1,2} & a_{2,2} & \cdots & a_{2,l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1,l} & a_{2,l} & \cdots & a_{l,l} \end{bmatrix}^{-1} \in \mathbb{R}^{l\times l}. \tag{29}$$

According to Lemma 1, $\rho_{1,1}^l$ is computed as

$$\rho_{1,1}^l = \frac{|B|}{|A|},$$

where $B = [a_{i,j}]_{2:l,2:l} \in \mathbb{R}^{(l-1)\times(l-1)}$ and $a_{i,j} = a_{j,i}$. Similarly, the cost function of the parameter estimates $\hat{\boldsymbol{\vartheta}}_k^{l-1}$ is

$$J(\boldsymbol{\vartheta}_k^{l-1}) = \frac{1}{2}\|\hat{Y}_{k-1}(L) - \hat{\boldsymbol{\Phi}}_{k-1}^{\mathrm{T}}(L)\boldsymbol{\vartheta}_{k-1}\|^2 - \frac{\rho_{1,1}^{l-1}}{2}\{[d_{k-1}^1]^{\mathrm{T}}d_{k-1}^1\}^2,$$

where

$$\begin{aligned} \rho_{1,1}^{l-1} &= \frac{|D|}{|C|}, \\ D &= [a_{i,j}]_{2:l-1,2:l-1} \in \mathbb{R}^{(l-2)\times(l-2)},\ a_{i,j} = a_{j,i}, \\ C &= [a_{i,j}]_{1:l-1,1:l-1} \in \mathbb{R}^{(l-1)\times(l-1)},\ a_{i,j} = a_{j,i}. \end{aligned}$$

Therefore, based on Lemma 4, we have

$$\rho_{1,1}^l \geqslant \rho_{1,1}^{l-1}.$$

It follows that

$$J(\boldsymbol{\vartheta}_k^l) \leqslant J(\boldsymbol{\vartheta}_k^{l-1}).$$

References

[1] P. Stoica and T. Söderström, "Bias correction in least-squares identification," *Int. J. Control*, vol. 35, no. 3, pp. 49-457, 1982.

[2] C.P. Yu, L. Ljung, A. Wills, and M. Verhaegen, "Constrained subspace method for the identification of structured state-space models," *IEEE Trans. Autom. Control*, vol. 65, no. 10, pp. 4201-4214, 2020.

[3] T. Söderström and P. Stoica, *Systen Identification*, Englewood Cliffs, NJ: Prentice-Hall, 1989.

[4] S.A. Billings and Q.M. Zhu, "A structure detection algorithm for nonlinear dynamics rational models," *Int. J. Control*, vol. 59, no. 6, pp. 1439-1463, 1994.

[5] G. Bottegal, A.Y. Aravkin, H. Hjalmarsson, and G. Pillonetto, "Robust EM kernel-based methods for linear system identification," *Automatica*, vol. 67, pp. 114-126, 2016.

[6] W. Greblicki and M. Pawlak, *Nonparametric System Identification*, Cambridge University Press, 2008.

[7] G.Y. Chen et al., "A regularized variable projection algorithm for separable nonlinear least squares problems," *IEEE Trans. Autom. Control*, vol. 64, no. 2, pp. 526-537, 2019.

[8] X.P. Geng, Q.M. Zhu, T. Liu, and J. Na, "U-model based predictive control for nonlinear processes with input delay," *J. Process Control*, vol. 75, pp. 156-170, 2019.

[9] J. Chen et al., "Interval error correction auxiliary model based gradient iterative algorithms for multi-rate ARX models," *IEEE Trans. Autom. Control*, vol. 65, no. 10, pp. 4385-4392, 2020.

[10] S. Magnússon et al., "Convergence of limited communication gradient methods," *IEEE Trans. Autom. Control*, vol. 63, no. 5, pp. 1356-1371, 2018.

[11] A.G. Wu, W.X. Zhang, and Y. Zhang, "An iterative algorithm for discrete periodic Lyapunov matrix equations," *Automatica*, vol. 87, pp. 395-403, 2018.

[12] M. Schmidt, N.L. Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," *Math. Program.*, vol. 162, no. 1-2, pp. 83-112, 2017.

[13] M. Hajarian, "Conjugate gradient-like methods for solving general tensor equation with Einstein product," *J. Frankl. Inst.*, vol. 357, no. 7, pp. 4272-4285, 2020.

[14] D.G. Skariah and M. Arigovindan, "Nested conjugate gradient algorithm with nested preconditioning for non-linear image restoration," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4471-4482, 2017.

[15] F. Ding and T. Chen, "Performance analysis of multi-innovation gradient type identification methods," *Automatica*, vol. 43, no. 1, pp. 1-14, 2007.

[16] H. Xu et al., "Fitting the exponential autoregressive model through recursive search," *J. Frankl. Inst.*, vol. 356, no. 11, pp. 5801-5818, 2019.

[17] J. Chen, Q.M. Zhu, and Y.J. Liu, "Modified Kalman filtering based multi-step-length gradient iterative algorithm for ARX models with random missing outputs," *Automatica*, vol. 118, 2020. https://doi.org/10.1016/j.automatica.2020.109034

[18] E.W. Bai, "Identification of linear systems with hard input nonlinearities of known structure," *Automatica*, vol. 38, no. 5, pp. 853-860, 2002.

[19] S.A. Billings and K.Z. Mao, "Structure detection for nonlinear rational models using genetic algorithms," *Int. J. Syst. Sci.*, vol. 29, no. 3, pp. 223-231, 1998.

[20] A. Hagenblad, L. Ljung, and A. Wills, "Maximum likelihood identification of Wiener models," *Automatica*, vol. 44, no. 11, pp. 2697-2705, 2008.

[21] J. Vörös, "Parameter identification of Wiener systems with multisegment piecewise-linear nonlinearities," *Syst. Control Lett.*, vol. 56, no. 2, pp. 99-105, 2007.

[22] S.A. Billings, *Nonlinear System Identification: NARMAX Methods in the Time, Frequency and Spatio-Temporal Domains*, Wiley, 2014.

[23] F. Ding and T. Chen, "Identification of Hammerstein nonlinear ARMAX systems," *Automatica*, vol. 41, no, 9, pp. 1479-1489, 2005.

[24] J. Chen et al., "Variational Bayesian approach for ARX systems with missing observations and varying time-delays," *Automatica*, vol. 94, pp. 194-204, 2018.

[25] F. Ding, G. Liu, and X. P. Liu, "Parameter estimation with scarce measurements," *Automatica*, vol. 47, no. 8, pp. 1646-1655, 2011.

[26] Y. Zhao, A. Fatehi, and B. Huang, "A data-driven hybrid ARX and Markov-Chain modeling approach to process identification with time varying time delays," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4226-4236, 2017.

[27] F. Ding and T. Chen, "Combined parameter and output estimation of dual-rate systems using an auxiliary model," *Automatica*, vol. 40, no. 10, pp. 1739-1748, 2004.

[28] Q.B. Jin, Z. Wang, and X.P. Liu, "Auxiliary model-based interval-varying multi-innovation least squares identification for multivariable OE-like systems with scarce measurements," *J. Process Control*, vol. 35, no. 11, pp. 154-168, 2015.

[29] Y.J. Wang and F. Ding, "Novel data filtering based parameter identification for multiple-input multiple-output systems using the auxiliary model," *Automatica*, vol. 71, pp. 308-313, 2016.

[30] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, 2003.

[31] D.Q. Wang et al., "Highly efficient identification methods for dual-rate Hammerstein systems," *IEEE Trans. Control Syst. Tech.*, vol. 23, no. 5, pp. 1952-1960, 2015.