Modified multi-direction iterative algorithm for separable nonlinear models with missing data

Jing Chen, Manfeng Hu, Yawen Mao, Quanmin Zhu

***Abstract*—Multi-direction iterative (MUL-DI) algorithm is an efficient algorithm for large-scale models, and it establishes a theoretical linkage between least squares (LS) and gradient descent (GD) algorithms. However, it involves Givens transformation and dense matrix calculation in each iteration, which leads to heavy computational efforts. In this letter, a modified MUL-DI algorithm is proposed for separable nonlinear models with missing data. Several directions are designed using a diagonal matrix, and their corresponding step-sizes are obtained based on LS algorithm. Compared with the traditional algorithms, the algorithm proposed in this letter has the following advantages: (1) has a faster convergence rate; (2) has a simple cost function; (3) is more robust to the condition number; (4) has less computational efforts. A simulation example shows the effectiveness of the modified MUL-DI algorithm.**

***Index Terms*—Multi-direction iterative algorithm, separable nonlinear model, missing data, convergence rate**

I. Introduction

SEPARABLE nonlinear (SN) model is a special kind of nonlinear models, in which the parameters are decomposed into two parts depending on if the model is linear or nonlinear with respect to the parameters [1]–[4]. This kind of models widely exist in modern society, e.g., in ecological environment [5], [6], in engineering practice [7], [8]. The identification of SN model is more difficult when compared with the linear and polynomial nonlinear model identification, for the reason that the two parameter parts are coupled with each other [9].

LS algorithm is a classical algorithm which has faster convergence rates for a linear-parameter-model with low-order. However, it is inefficient for systems with complex nonlinear structures or with high-order [10]–[12]. The GD algorithm is an ideal choice for systems with complex nonlinear structures. It does not require solving a derivative function, but its convergence rates are quite slow [13]. To increase the convergence rates, a plethora of modified GD algorithms are developed [14]. For example, the forgetting factor GD algorithm [15], the steepest GD algorithm [16], and the conjugate GD algorithm [17]. There is a bottleneck issue of these modified GD algorithms: if the condition number of the information matrix is large, whatever the step-size is, the convergence rate is always slow [18].

For the unique structure of the SN model, the variable projection (VP) algorithm can be more efficient than the traditional GD (T-GD) algorithm [19]. The basic idea of the VP algorithm is to transform the nonlinear model into a linear model with low-order. Although the low-order model has fewer parameters than those of the SN model, its cost function is more complex, which leads to a difficulty in designing an optimal step-size in each iteration [2]. The multi-direction iterative (MUL-DI) algorithm, which has a simple cost function, was first proposed in [20]. It designs several directions in each iteration. The convergence rates of the MUL-DI algorithm are improved with the increased number of directions. When the number of directions is larger than 1, the MUL-DI algorithm can get around the bottleneck issue caused by the condition number. However, the MUL-DI algorithm requires Givens transformation to calculate the optimal step-sizes and needs a dense matrix to generate the directions, which leads to heavy computational efforts.

In this letter, a modified MUL-DI algorithm is proposed for SN models with missing data. Based on the residual error in each iteration, several directions are designed using a diagonal matrix, and their corresponding step-sizes are calculated using LS algorithm rather than Givens transformation method. Therefore, the computational efforts can be reduced. Compared with the traditional algorithms, the algorithm proposed in this letter has the following contributions: (1) has faster convergence rates than the VP and T-GD algorithms; (2) has a simpler cost function than the VP algorithm ; (3) is more robust to the condition number of the information matrix than that of the T-GD algorithm; (4) has less computational efforts than the traditional MUL-DI algorithm.

The rest of this letter is organized as follows. Section II defines the SN model and the T-GD algorithm. Section III proposes the modified MUL-DI algorithm. Section IV provides a simulation example to demonstrate the main results. Section V concludes the letter and discusses possible future study directions.

II. Problem formulation

Firstly setup some notations. The norm of a matrix $\mathbf{X}$ is defined as $\|\mathbf{X}\| = \sqrt{\lambda_{\max}[\mathbf{X}\mathbf{X}^{\mathrm{T}}]}$; $\lambda_{\max}[\mathbf{X}\mathbf{X}^{\mathrm{T}}]$ and $\lambda_{\min}[\mathbf{X}\mathbf{X}^{\mathrm{T}}]$ mean the maximum and minimum eigenvalues of matrix $\mathbf{X}\mathbf{X}^{\mathrm{T}}$, respectively; the norm of a vector $\mathbf{z} = [z_1, z_2, \cdots, z_n]^{\mathrm{T}} \in \mathbb{R}^n$ is defined as $\|\mathbf{z}\| = (\sum_{i=1}^{n} {z_i}^2)^{\frac{1}{2}}$; the superscript T denotes the matrix transpose.

A. SN model with missing data

Consider the following SN model,

$$y(t) = f(\mathbf{a}, u(t))\mathbf{c} + v(t),$$

where $y(t)$ is the output; $u(t)$ is the input; $v(t)$ is a Gaussian white noise that satisfies $v(t) \sim N(0, \sigma^2)$; $\mathbf{a} \in \mathbb{R}^m$ is the nonlinear parameter vector and $\mathbf{c} \in \mathbb{R}^n$ is the linear parameter vector; $f(\cdot)$ is a nonlinear function with known structure. Assume that we have collected L input and output data, and those output data in the sampling instants $o_1, o_2, \cdots, o_p$ are measurable, while the other output data in the sampling instants $m_1, m_2, \cdots, m_q$ are missing, $p + q = L$, $p > (m + n)$ and $p > q$.

Two standard tools are usually applied to systems with missing data: (1) the lifting technique, e.g., the polynomial transformation algorithm [21], its key is to transform the system into a lifted system with high-order, and then to estimate the parameters based on all the measurable data; (2) the imputation technique, e.g., the imputation based LMS algorithm [22], it firstly estimates the missing data, and then updates the parameters based on the measurable data and estimated data. Since the first tool requires identifying more

J. Chen, M.F. Hu and Y.W. Mao are with the School of Science, Jiangnan University, Wuxi 214122, PR China (chenjing1981929@126.com, humanfeng@jiangnan.edu.cn, myw0530@163.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 was supported by the National Natural Science Foundation of China (No. 61973137) and the Natural Science Foundation of Jiangsu Province (No. BK20201339).

parameters and is only available for noise-free systems, this letter focuses on the second one.

Define

$$\begin{aligned}\mathbf{Y}(L)&=[y(L),y(L-1),\cdots,y(1)]^{\mathrm{T}}\in\mathbb{R}^{L},\\ \mathbf{F}(\mathbf{a},u(L))&=[f(\mathbf{a},u(L)),f(\mathbf{a},u(L-1)),\cdots,f(\mathbf{a},u(1))]^{\mathrm{T}}\in\mathbb{R}^{L\times n},\\ \mathbf{V}(L)&=[v(L),v(L-1),\cdots,v(1)]^{\mathrm{T}}\in\mathbb{R}^{L}.\end{aligned}$$

We have

$$\mathbf{Y}(L)=\mathbf{F}(\mathbf{a},u(L))\mathbf{c}+\mathbf{V}(L). \tag{1}$$

B. Traditional GD algorithm

Define the cost function

$$J(\mathbf{a},\mathbf{c})=\|\mathbf{Y}(L)-\mathbf{F}(\mathbf{a},u(L))\mathbf{c}\|^2.$$

The negative direction is

$$d_k=-\left[\begin{array}{c}\frac{\partial J(\mathbf{a},\mathbf{c})}{\partial\mathbf{a}}\\ \frac{\partial J(\mathbf{a},\mathbf{c})}{\partial\mathbf{c}}\end{array}\right].$$

The parameters are updated by

$$\left[\begin{array}{c}\hat{\mathbf{a}}_k\\ \hat{\mathbf{c}}_k\end{array}\right]=\left[\begin{array}{c}\hat{\mathbf{a}}_{k-1}\\ \hat{\mathbf{c}}_{k-1}\end{array}\right]+\gamma d_k,$$

where $\hat{\mathbf{a}}_k$, $\hat{\mathbf{c}}_k$ mean the estimates in iteration k, and γ is the step-size.

Remark 1. Although the T-GD algorithm can obtain the parameter estimates without solving a derivative equation. It brings another challenging issue: the step-size choosing problem. A larger step-size may make the T-GD algorithm divergent, while a smaller one can lead to slow convergence rates.

If the considered model is linear, and is written by

$$\mathbf{Y}(L)=\mathbf{F}(L)\mathbf{c}+\mathbf{V}(L), \tag{2}$$

where $\mathbf{F}(L)\in\mathbb{R}^{L\times n}$ is the information matrix which constitutes of the input data before the sampling instant L. Using the T-GD algorithm to obtain the parameter estimates $\hat{\mathbf{c}}_k$ yields

$$\hat{\mathbf{c}}_k=\hat{\mathbf{c}}_{k-1}+\gamma\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\mathbf{c}_{k-1}].$$

To keep the T-GD algorithm convergent, the step-size γ should satisfy

$$0<\gamma<\frac{2}{\lambda_{max}[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]}.$$

In [23], the optimal step-size was given as follows:

$$\gamma_{op}=\frac{2}{\lambda_{max}[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]+\lambda_{min}[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]}.$$

For the optimal step-size, the convergence factor is

$$\|\hat{\mathbf{e}}_k\|=\mu\|\hat{\mathbf{e}}_{k-1}\|=\frac{\tau-1}{\tau+1}\|\hat{\mathbf{e}}_{k-1}\|,$$

where τ is the condition number of the information matrix $[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]\in\mathbb{R}^{n\times n}$, and μ is the convergence factor.

Remark 2. Once the information matrix $[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]$ is fixed, no matter what the step-size is, the best convergence factor of the T-GD algorithm is $\frac{\tau-1}{\tau+1}$. Therefore, if the information matrix is ill-conditioned or singular, the T-GD algorithm is inefficient.

Remark 3. Owing to the special structure of the SN model, the information matrix in the T-GD algorithm is changing in each iteration, thus, we cannot find a fixed interval in which the step-size can be chosen to keep the T-GD algorithm convergent.

III. Modified MUL-direction iterative algorithm

The VP algorithm proposed in [2], [8] has a complex structure, while the T-GD algorithm has a quite slow convergence rate. To deal with these dilemmas, the MUL-direction iterative (MUL-DI) algorithm is introduced in this section.

A. Basic idea of the modified MUL-DI algorithm

For the linear model (2), using the LS algorithm to estimate the parameters yields

$$\begin{aligned}\hat{\mathbf{c}}_k&=[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]^{-1}\mathbf{F}^{\mathrm{T}}(L)\mathbf{Y}(L)\\ &=\hat{\mathbf{c}}_{k-1}+[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]^{-1}\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}],\end{aligned} \tag{3}$$

where $\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]\in\mathbb{R}^n$.

The T-GD algorithm is listed as follows:

$$\hat{\mathbf{c}}_k=\hat{\mathbf{c}}_{k-1}+\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]\gamma. \tag{4}$$

The differences between the T-GD and LS algorithms are shown in Table I.

TABLE I: The differences between T-GD and LS algorithms

Algorithm	T-GD	LS
Direction	$\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]$	$[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]^{-1}$
Step-size	γ	$\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]$
Number of Directions	1	n
Number of Step-sizes	1	n

When the number of the unknown parameters is n, the LS algorithm can obtain the optimal estimate in only one iteration based on n directions and n corresponding step-sizes; on the other hand, the GD algorithm updates the parameters using only one direction and one step-size. Therefore, the LS algorithm has faster convergence rates but heavier computational efforts, while the GD algorithm has less computational efforts but slower convergence rates.

Remark 4. Although the modified GD algorithms, e.g., the forgetting factor GD algorithm, the steepest GD algorithm, and the conjugate GD algorithm, can increase the convergence rates. They cannot reduce the condition number of the information matrix, which means that these algorithms are sensitive to the condition number.

Remark 5. The multi-innovation algorithm can increase the convergence rates by designing several innovations in each iteration [4], [7], [19], where the innovations can be regarded as directions. However, the two innovations/directions are dependent on each other, and then their corresponding estimates will be correlated, which leads to less confidence of the number of the directions. In addition, for the related innovations, designing the optimal step-sizes for them is impossible.

A natural question arises, can we develop a novel algorithm which can update the parameters through l ($1\leqslant l\leqslant n$) directions and l step-sizes in each iteration, and each direction can be assigned an optimal step-size. This algorithm, termed as MUL-DI algorithm, will have faster convergence rates and be robust to the condition number. It has the following three steps:

(1) compute the residual error as an original direction;

(2) construct several orthogonal directions based on the original direction;

(3) design the optimal step-sizes for each direction.

Assume that the estimate in iteration $k-1$ is $\hat{\mathbf{c}}_{k-1}$, the original direction in iteration k is

$$\mathbf{d}_{k,1}=\frac{\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]}{\|\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L)-\mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]\|}.$$

Use Arnoldi's method to design the remaining $l-1$ directions,

$$\mathbf{d}_{k,i}=\mathbf{N}\mathbf{d}_{k,i-1}-\sum_{j=1}^{i-1}(\mathbf{N}\mathbf{d}_{k,i-1},\mathbf{d}_{k,j})\mathbf{d}_{k,j}, \tag{5}$$

where $\mathbf{N}\in\mathbb{R}^{n\times n}\neq\mathbf{I}$ is a non-singular diagonal matrix. This is different from the work in [20], where $\mathbf{N}=[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]$ is a dense matrix. Once $\mathbf{d}_{k,i}$ is obtained, let

$$\mathbf{d}_{k,i}=\frac{\mathbf{d}_{k,i}}{\|\mathbf{d}_{k,i}\|}.$$

Then, the l directions are written by

$$\mathbf{D}_{k,l} = [\mathbf{d}_{k,1}, \mathbf{d}_{k,2}, \cdots, \mathbf{d}_{k,l}] \in \mathbb{R}^{n\times l}. \quad (6)$$

Assume that their corresponding step-sizes are

$$\mathbf{R}_{k,l} = [\gamma_{k,1}, \gamma_{k,2}, \cdots, \gamma_{k,l}]^{\mathrm{T}}.$$

Then, the estimate $\hat{\mathbf{c}}_k$ using the modified MUL-DI (M-MUL-DI) algorithm is written by

$$\hat{\mathbf{c}}_k = \hat{\mathbf{c}}_{k-1} + \mathbf{D}_{k,l}\mathbf{R}_{k,l}. \quad (7)$$

Define the cost function

$$J(\mathbf{R}_{k,l}) = \|\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1} - \mathbf{F}(L)\mathbf{D}_{k,l}\mathbf{R}_{k,l}\|^2. \quad (8)$$

Let

$$\|\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]\| = \alpha.$$

Taking the derivative of $J(\mathbf{R}_{k,l})$ with respect to $\mathbf{R}_{k,l}$ and equating it to zero yield

$$\mathbf{R}_{k,l} = [\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}]^{-1}\mathbf{D}_{k,l}^{\mathrm{T}}\alpha\mathbf{d}_{k,1}. \quad (9)$$

Remark 6. By using the M-MUL-DI algorithm, each direction has its own optimal step-size. In addition, we can choose different numbers of directions in each iteration on a case by case basis.

Remark 7. In the M-MUL-DI algorithm, the computational efforts are generally reduced in two ways: (1) apply a diagonal matrix $\mathbf{N}$ rather than a dense matrix $[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]$ to generate l directions [20]; (2) use the LS algorithm instead of the Givens transformation method to calculate the step-sizes [24].

B. Properties of the M-MUL-DI algorithm

(1) The matrix $D_{k,l}$

Based on Equations (8) and (9), the parameters updated using the M-MUL-DI algorithm are written by

$$\begin{aligned}\hat{\mathbf{c}}_k &= \hat{\mathbf{c}}_{k-1} + \mathbf{D}_{k,l}[\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}]^{-1}\mathbf{D}_{k,l}^{\mathrm{T}}\alpha\mathbf{d}_{k,1}\\ &= \hat{\mathbf{c}}_{k-1} + \mathbf{D}_{k,l}[\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}]^{-1}\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\times\\ &\quad [\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]. \end{aligned} \quad (10)$$

Case 1: rank $[\mathbf{D}_{k,l}] = l = n$

When $l = n$, Equation (10) is simplified as

$$\begin{aligned}\hat{\mathbf{c}}_k &= \hat{\mathbf{c}}_{k-1} + \mathbf{D}_{k,l}\mathbf{D}_{k,l}^{-1}[\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]^{-1}[\mathbf{D}_{k,l}^{\mathrm{T}}]^{-1}\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\times\\ &\quad [\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1}]\\ &= \hat{\mathbf{c}}_{k-1} + [\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)]^{-1}\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1}].\end{aligned}$$

In this case, the M-MUL-DI algorithm is the same as the LS algorithm.

Case 2: rank $[\mathbf{D}_{k,l}] = l = 1$

There is only one direction in each iteration, that is

$$\mathbf{D}_{k,l} = d_{k,1}.$$

Substituting the above equation into Equation (10) yields

$$\begin{aligned}\hat{\mathbf{c}}_k &= \hat{\mathbf{c}}_{k-1} + \mathbf{d}_{k,1}[\mathbf{d}_{k,1}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{d}_{k,1}]^{-1}\mathbf{d}_{k,1}^{\mathrm{T}}\alpha\mathbf{d}_{k,1}\\ &= \hat{\mathbf{c}}_{k-1} + \frac{\mathbf{d}_{k,1}\mathbf{d}_{k,1}^{\mathrm{T}}}{\mathbf{d}_{k,1}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{d}_{k,1}}\alpha\mathbf{d}_{k,1}.\end{aligned}$$

Since $\mathbf{d}_{k,1}^{\mathrm{T}}\mathbf{d}_{k,1} = 1$, we have

$$\hat{\mathbf{c}}_k = \hat{\mathbf{c}}_{k-1} + \frac{1}{\mathbf{d}_{k,1}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{d}_{k,1}}\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1}].$$

In this case, the step-size is

$$\gamma_k = \frac{1}{\mathbf{d}_{k,1}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{d}_{k,1}}.$$

The step-size of the M-MUL-DI algorithm is adaptively chosen in each iteration. Therefore, when $l = 1$, the M-MUl-DI algorithm is equivalent to the T-GD algorithm. However, the M-MUL-DI algorithm can be more efficient than the T-GD algorithm for the reason that the step-size is adaptively chosen.

Case 3: $1 <$ rank $[\mathbf{D}_{k,l}] = l < n$

According to Equations (8) and (9), one can get

$$J(\hat{\mathbf{c}}_k) \leqslant J(\hat{\mathbf{c}}_{k-1}).$$

In addition, the original direction in iteration $k+1$ is

$$\mathbf{d}_{k+1,1} = \mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_k].$$

For simplicity, we do not need to normalize it here. Multiplying both sides of the above equation by $\mathbf{D}_{k,l}$ yields

$$\begin{aligned}\mathbf{d}_{k+1,1}^{\mathrm{T}}\mathbf{D}_{k,l} &= \left[\mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_{k-1} - \mathbf{F}(L)\mathbf{D}_{k,l}\mathbf{R}_{k,l}]\right]^{\mathrm{T}}\times\\ &\quad \mathbf{D}_{k,l}\\ &= \left[\alpha\mathbf{d}_{k,1} - \mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}\mathbf{R}_{k,l}\right]^{\mathrm{T}}\mathbf{D}_{k,l}.\end{aligned}$$

Substituting Equation (9) into the above equation yields

$$\begin{aligned}\mathbf{d}_{k+1,1}^{\mathrm{T}}\mathbf{D}_{k,l} = \Big[&\alpha\mathbf{D}_{k,l}\mathbf{e}_1 - \alpha\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}\times\\ &[\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}]^{-1}\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{D}_{k,l}\mathbf{e}_1\Big]^{\mathrm{T}}\mathbf{D}_{k,l}, \end{aligned} \quad (11)$$

where $\mathbf{e}_1 = [1, 0, 0, \cdots, 0]^{\mathrm{T}} \in \mathbb{R}^l$.

Since $\mathbf{d}_{k,i}$ is orthogonal to $\mathbf{d}_{k,j}$ $(i \neq j)$. It follows that $\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{D}_{k,l} = \mathbf{I} \in \mathbb{R}^{l\times l}$. Then, Equation (11) is simplified as

$$\begin{aligned}\mathbf{d}_{k+1,1}^{\mathrm{T}}\mathbf{D}_{k,l} &= \alpha\mathbf{e}_1^{\mathrm{T}}\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{D}_{k,l} - \alpha\mathbf{e}^{\mathrm{T}}[\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}]^{-1}\times\\ &\quad \mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}(L)\mathbf{F}(L)\mathbf{D}_{k,l}\\ &= \mathbf{0}. \end{aligned} \quad (12)$$

It shows that the new direction is orthogonal to the previous directions.

(2) Convergence properties of the M-MUL-DI algorithm

When $l = n$, the M-MUL-DI algorithm is the same as the LS algorithm. Based on [25], for a convex cost function, the LS algorithm is convergent, and its convergence factor is $\mu = 0$.

When $l = 1$, the M-MUL-DI algorithm is equivalent to the T-GD algorithm. From [25], it shows that the T-GD algorithm is always convergent for a convex cost function, and its convergence factor satisfies $\frac{\tau-1}{\tau+1} \leqslant \mu < 1$.

When $1 < l < n$, the inequality $J(\hat{\mathbf{c}}_k) \leqslant J(\hat{\mathbf{c}}_{k-1})$ ensures that the M-MUL-DI algorithm is convergent; Equation (12) means that the new direction is orthogonal to the previous directions, which can guarantee the estimates asymptotically converge to the true values. That is, the M-MUL-DI algorithm will never stop running until the new original direction $\mathbf{d}_{k+1,1} = \mathbf{F}^{\mathrm{T}}(L)[\mathbf{Y}(L) - \mathbf{F}(L)\hat{\mathbf{c}}_k] = \mathbf{0}$. In addition, its convergence factor satisfies $0 < \mu < \frac{\tau-1}{\tau+1}$.

C. M-MUL-DI algorithm for the SN model with missing data

For the SN model, the original direction in iteration k is

$$\mathbf{d}_{k,1} = \begin{bmatrix} \mathbf{c}_{k-1}\mathbf{F}'(L, \mathbf{a}_{k-1}) \\ \mathbf{F}(L, \mathbf{a}_{k-1}) \end{bmatrix}[\mathbf{Y}(L) - \mathbf{F}(L, \mathbf{a}_{k-1})\mathbf{c}_{k-1}].$$

Define

$$\begin{aligned}\mathbf{M} = &\begin{bmatrix} \mathbf{c}_{k-1}\mathbf{F}'(L, \mathbf{a}_{k-1}) \\ \mathbf{F}(L, \mathbf{a}_{k-1}) \end{bmatrix}\times\\ &[[\mathbf{c}_{k-1}\mathbf{F}'(L, \mathbf{a}_{k-1})]^{\mathrm{T}}, [\mathbf{F}(L, \mathbf{a}_{k-1})]^{\mathrm{T}}] \in \mathbb{R}^{(m+n)\times(m+n)}.\end{aligned}$$

Since the output vector $\mathbf{Y}(L)$ contains unknown outputs, we use the parameter estimates in iteration $k-1$ to compute these missing data,

$$\hat{y}_k(m_j) = f(\mathbf{a}_{k-1}, m_j)\mathbf{c}_{k-1},\ j = 1, 2, \cdots, q.$$

The M-MUL-DI algorithm for the SN model with missing data is written as

$$\begin{bmatrix} \mathbf{a}_k \\ \mathbf{c}_k \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{k-1} \\ \mathbf{c}_{k-1} \end{bmatrix} + \mathbf{D}_{k,l}\mathbf{R}_{k,l}, \tag{13}$$

$$\mathbf{D}_{k,l} = [\mathbf{d}_{k,1}, \mathbf{d}_{k,2}, \cdots, \mathbf{d}_{k,l}] \in \mathbb{R}^{(m+n)\times l}, \tag{14}$$

$$\mathbf{R}_{k,l} = [\mathbf{D}_{k,l}^{\mathrm{T}}\mathbf{M}\mathbf{D}_{k,l}]^{-1}\mathbf{D}_{k,l}^{\mathrm{T}}\begin{bmatrix} \mathbf{c}_{k-1}\mathbf{F}'(L, \mathbf{a}_{k-1}) \\ \mathbf{F}(L, \mathbf{a}_{k-1}) \end{bmatrix} \times [\hat{\mathbf{Y}}_k(L) - \mathbf{F}(L, \mathbf{a}_{k-1})\mathbf{c}_{k-1}], \tag{15}$$

$$\hat{y}_k(m_j) = f(\mathbf{a}_{k-1}, m_j)\mathbf{c}_{k-1}, \ j = 1, 2, \cdots, q, \tag{16}$$

$$\hat{\mathbf{Y}}_k = [y(L), \cdots, \hat{y}_k(m_q), \cdots, y(1)]^{\mathrm{T}}. \tag{17}$$

The steps of the M-MUL-DI algorithm are listed as follows:

M-MUL-DI algorithm

Initialize $\mathbf{a_0} = \mathbf{1/p_0}$, $\mathbf{c_0} = \mathbf{1/p_0}$
Collect measurable data $u(1), \cdots, u(L)$ and $y(o_1), \cdots, y(o_p)$
Repeat
  for $k = 1, 2, \cdots,$ do
    Compute $\hat{y}_k(m_j), j = 1, \cdots, q$ based on (16)
    Form $\hat{\mathbf{Y}}_k$ based on (17)
    Compute $\mathbf{D}_{k,l}$ according to (14)
    Compute $\mathbf{R}_{k,l}$ according to (15)
    Update $\mathbf{a}_k$ and $\mathbf{c}_k$ using (13)
  end
until convergence

Remark 8: In this letter, we consider that the system has missing output data. If the system has missing input data, the M-MUL-DI algorithm is invalid. In this case, we can try to combine the M-MUL-DI algorithm with the imputation model based method in [22] to deal with this problem.

IV. Example

Consider an SN model in [5],

$$y(t) = c_1 e^{-a_2 u^2(t-1)} \cos(a_3 u(t-1)) + c_2 e^{-a_1 u^2(t-1)} \times \cos(a_2 u(t-2)) + c_3 e^{-a_4 u^2(t-1)} \sin(a_1 u(t-3)) + v(t),$$
$$\boldsymbol{\theta} = [\mathbf{c}^{\mathrm{T}}, \mathbf{a}^{\mathrm{T}}]^{\mathrm{T}}, \ \mathbf{c} = [c_1, c_2, c_3]^{\mathrm{T}} = [2, 3, 2]^{\mathrm{T}},$$
$$\mathbf{a} = [a_1, a_2, a_3, a_4]^{\mathrm{T}} = [1, 1.5, 3, 0.8]^{\mathrm{T}},$$

where $\{u(t)\}$ is an input sequence with zero mean and unit variance, $\{v(t)\}$ is taken as a white noise sequence with zero mean and variance $\sigma^2 = 0.10^2$. In the simulation, we generate $L = 800$ pairs of input-output data, where the data in the sampling instants $4, 8, 12, \cdots, 4i, \cdots, 796$ are missing.

Apply the T-GD, VP and M-MUL-DI ($l = 3$, $\mathbf{N} = \mathrm{diag}\{1, 2, 3, 4, 5, 6, 7\}$) algorithms for the SN model. The parameter estimates and their estimation errors $\tau := \|\boldsymbol{\theta}_k - \boldsymbol{\theta}\|/\|\boldsymbol{\theta}\|$ are shown in Fig. 1 and Table II. The predicted and true outputs are shown in Fig. 2. In addition, use the M-MUL-DI and traditional MUL-DI (T-MUL-DI) algorithms ($l = 3$) for the SN model (**Intel(R) Core(TM) i7-8550U: 1.80GHz, 2.00GHz; RAM: 8.0 GB; Windows 10**). The elapsed times and estimation errors are shown in Table III.

From this simulation, we have the following findings: (1) for the SN model identification, the M-MUL-DI algorithm has the fastest convergence rates when $l > 1$, this can be shown in Fig. 1 and Table II; (2) the VP and M-MUL-DI algorithms can well catch the true outputs of the SN model, but the T-GD algorithm cannot, as shown in Fig. 2; (3) the M-MUL-DI algorithm has less elapsed times than those from the T-MUL-DI algorithm, as shown in Table III, that is, the M-MUL-DI algorithm has less computational efforts than those from the T-MUL-DI algorithm.

TABLE II: The parameter estimates and their estimation errors

Algorithm	k	c_1	c_2	c_3	a_1	a_2	a_3	a_4	τ (%)
T-GD	100	0.11	0.10	0.10	0.10	1.12	1.21	0.21	82.02
	200	0.13	0.09	0.10	0.10	1.34	1.42	0.23	80.30
	500	0.18	0.05	0.08	0.10	2.14	1.98	0.29	78.06
VP	100	3.60	0.53	10.62	0.08	1.14	1.51	0.26	170.04
	200	3.24	1.82	2.73	0.61	1.95	2.16	0.46	39.36
	500	2.77	2.26	1.98	0.96	1.79	2.33	0.58	24.12
M-MUL-DI	100	2.85	2.18	1.98	0.96	1.83	2.28	0.57	26.42
	200	2.05	2.95	1.98	1.01	1.52	2.94	0.76	1.94
	500	2.01	2.99	1.97	1.02	1.51	2.98	0.78	0.83
	True Value	2.00	3.00	2.00	1.00	1.50	3.00	0.80	

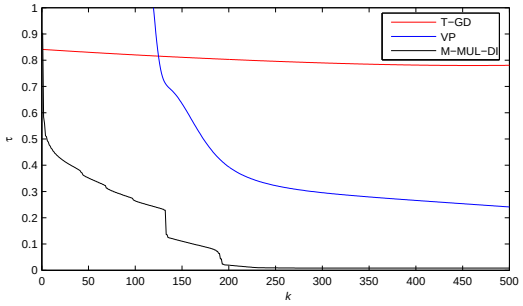

Fig. 1: The parameter estimation errors

TABLE III: The elapsed times and estimation errors

Algorithm	Estimation error	Elapsed time (second)
T-MUL-DI	0.82526%	20.048
M-MUL-DI	0.82643%	16.414

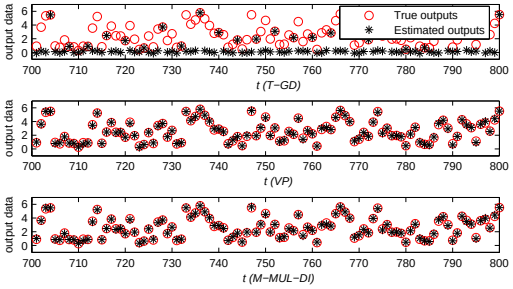

Fig. 2: The true outputs and their estimates

V. Conclusions

An M-MUL-DI algorithm is proposed for SN models with missing data in this letter. The algorithm constructs several orthogonal directions in each iteration, thus it has faster convergence rates than the T-GD algorithm, and has a simpler cost function than the VP algorithm. In addition, it has less computational efforts when compared with the T-MUL-DI algorithm. The benefits of the M-MUL-DI algorithm are demonstrated through a simulation example.

The M-MUL-DI algorithm has some advantages over the traditional algorithms, and can be used in wide range of cutting edge applications. However, some challenging issues need to be further discussed to enrich and enhance the MUL-DI algorithm. For example, how to choose the optimal number of directions in each iteration, and how to avoid the same direction in different iterations. Similar topics will remain as meaningfully challenging issues in future studies.

REFERENCES

[1] G.H. Golub and V. Pereyra, "The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate," *SIAM J. Numer. Anal.*, vol. 10, no. 2, pp. 413-432, 1973.

[2] G.H. Golub and V. Pereyra, "Separable nonlinear least squares: the variable projection method and its applications," *Inverse Probl.*, vol. 19, no. 2, pp. R1-R26, 2003.

[3] L. Xu, F. Ding, L.J. Wan, and J. Sheng, "Separable multi-innovation stochastic gradient estimation algorithm for the nonlinear dynamic responses of systems," *Int. J. Adapt. Control Signal Process.*, vol. 34, no. 7, pp. 937-954, 2020.

[4] H. Xu, F. Ding, and B. Champagne, "Joint parameter and time-delay estimation for a class of nonlinear time-series models," *IEEE Signal Process. Lett.*, vol. 29, pp. 947-951, 2022.

[5] M. Gan, C.L.P. Chen, L. Chen, and C.Y. Zhang, "Exploiting the interpretability and forecasting ability of the RBF-AR model for nonlinear time series," *Int. J. Syst. Sci.*, vol. 47, no. 8, pp. 1868-1876, 2016.

[6] N.C. Stenseth, W. Falck, et al., "From ecological patterns to ecological processes: Phase and density-dependencies in Canadian lynx cycle," *Proc. National Academy Sci.*, USA, vol. 95, no. 26, pp. 15430-15435, 1999.

[7] Y.H. Zhou and F. Ding, "Modeling nonlinear processes using the radial basis function-based state-dependent autoregressive models," *IEEE Signal Process. Lett.*, vol. 27, pp. 1600-1604, 2020.

[8] G.Y. Chen, M. Gan, S.Q. Wang, and C.L.P. Chen, "Insights into algorithms of separable nonlinear least squares problems," *IEEE Trans. Image Process.*, vol. 30, pp. 1207-1218, 2021.

[9] H. Xu and E.F. Yang, "Three-stage multi-innovation parameter estimation for an exponential autoregressive time-series model with moving average noise by using the data filtering technique," *Int. J. Robust Nonlinear Control*, vol. 31, no. 1, pp. 166-184, 2021.

[10] X.P. Liu and X.Q. Yang, "Identification of nonlinear state-space systems with skewed measurement noises," *IEEE Trans. Circuits Syst. I.*, 2022. DOI: 10.1109/TCSI.2022.3193444.

[11] G. Pillonetto, F. Dinuzzo, T.S. Chen, G.D. Nicolao, and L. Ljung, "Kernel methods in system identification, machine learning and function estimation: A survey," *Automatica*, vol. 50, pp. 657-682, 2014.

[12] B. Sinquin and M. Verhaegen, "K4SID: Large-scale subspace identification with kronecker modeling," *IEEE Trans. Autom. Control*, vol. 64, no. 3, pp. 960-975, 2019.

[13] D.Q. Wang, L.W. Li, Y. Ji, and Y.R. Yan, "Model recovery for Hammerstein systems using the auxiliary model based orthogonal matching pursuit method," *Appl. Math. Model.*, vol. 54, pp. 537-550, 2018.

[14] M. Jiao, D.Q. Wang, and J.L. Qiu, "A GRU-RNN based momentum optimized algorithm for SOC estimation," *J. Power Sources*, vol. 459, 2020. DOI: 10.1016/j.jpowsour.2020.228051

[15] Y.M. Fan and X.M. Liu, "Data filtering-based multi-innovation forgetting gradient algorithms for input nonlinear FIR-MA systems with piecewise-linear characteristics ," *J. Frankl. Inst.*, vol. 358, no. 18, pp. 9818-9840, 2022.

[16] M. Gerard, B.D. Schutter, and M. Verhaegen, "A hybrid steepest descent method for constrained convex optimization," *Automatica*, vol. 45, no. 2, pp. 525-531, 2009.

[17] L.T. Qi, M.L. Shen, D.L. Wang, and S.Y. Wang, "Robust cauchy kernel conjugate gradient algorithm for non-Gaussian noises," *IEEE Signal Process. Lett.*, vol. 28, pp. 1011-1015, 2021.

[18] J. Chen, D.Q. Wang, Y.J. Liu, and Q.M. Zhu, "Varying infimum gradient descent algorithm for agent-sever systems using different order iterative preconditioning methods," *IEEE Trans. Ind. Inform.*, vol. 18, no. 7, pp. 4436-4446, 2022.

[19] M. Gan, X.X. Chen, F. Ding, G.Y. Chen, and C.L.P. Chen, "Adaptive RBF-AR models based on multi-innovation least squares method," *IEEE Signal Process. Lett.*, vol. 26, no. 8, pp. 1182-1186, 2019.

[20] J. Chen, B. Huang, et al., "A novel reduced-order algorithm for rational model based on Arnoldi process and Krylov subspace," *Automatica*, vol. 129, 2021. DOI: 10.1016/j.automatica.2021.109663

[21] J. Ding, L.L. Han, and X.M. Chen, "Time series AR modeling with missing observations based on the polynomial transformation," *Math. Comput. Model.*, vol. 52, no. 5-6, pp. 527-536, 2010.

[22] S. Mukhopadhyay and A. Mukherjee, "ImdLMS: An imputation based LMS algorithm for linear system identification with missing input data," *IEEE Trans. Signal Process.*, vol. 68, pp. 2370-2385, 2020.

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

[24] J. Chen, J.X. Ma, M. Gan, and Q.M. Zhu, "Multi-direction gradient iterative algorithm: A unified framework for gradient iterative and least squares algorithms," *IEEE Trans. Autom. Control*, 2021. DOI: 10.1109/TAC.2021.3132262

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