Sparse augmented Lagrangian algorithm for system identification

DOI:
10.1016/j.neucom.2018.11.019

Document Version
Accepted author manuscript

Sparse Augmented Lagrangian Algorithm For System Identification

Xiaoquan Tang

School of Automation, Huazhong University of Science and Technology, Wuhan, Hubei, China

Long Zhang

School of Electrical and Electronic Engineering, University of Manchester Manchester, M13 9PL, United Kingdom

Xiaolin Wang

School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China

Abstract

A huge class of nonlinear dynamic systems can be approximated by the Nonlinear AutoRegressive with eXogenous inputs (NARX) models. This paper proposes a novel method, Sparse Augmented Lagrangian (SAL), for NARX model variable selection and parameter estimation. Firstly, Split Augmented Lagrangian Shrinkage Algorithm (SALSA) is applied to produce some intermediate models with subsampling technique, and then only the model terms with high selecting probability are chosen into the final model, followed by the model parameter estimation via SALSA. The model sparsity and algorithm convergence can be guaranteed through theoretical analysis. Two nonlinear examples and one real-world application from the process industry are used to demonstrate the effectiveness and advantages of the proposed method in comparison to several popular methods.

Keywords: System Identification, Stability Selection, NARX, SALSA

1. Introduction

A huge class of nonlinear systems can be well represented using NARX models. When all the model parameters are given a prior, the NARX model can be converted into a linear-in-the-parameters structure model and then a lot of algorithms can be used to determine the model structure. The model structure determination is the key challenge for system identification since it is difficult to select variables especially when the terms are highly correlated [1, 2, 3]. To address this problem, many variable selection methods have been proposed to select a subset from the candidate dictionary.

Selecting important terms from the predetermined dictionary can be considered as building a sparse model with good generalization performance [4]. This is the typical subset selection algorithm and forward selection is one of the most popular categories. The forward selection algorithms begin with the empty model, and then gradually select one term at a time from the predetermined dictionary until certain stopping criterion is satisfied [5]. The popular forward selection algorithms include orthogonal matching pursuits (OMP) [6], orthogonal least squares (OLS) [7], etc. However, the forward selection algorithms may not be optimal [8],

various modified algorithms were proposed for improving model compactness, such as optimized orthogonal matching pursuits (OOMP) [9], iterative orthogonal forward regression (iOFR) [10], etc.

Alternatively, l_1 and l_2 regularization methods are often used to build sparse models, and l_1 regularization tends to find a more sparse solution comparing with l_2 regularization. The Least absolute shrinkage and selection operator (Lasso) is one of the most popular l_1 optimization algorithms, which can prevent some redundant model terms being included into the final model. This is achieved by not only minimizing the residual sum of squares but also the sum of the absolute value of model term coefficients. However, the obtained model may be suboptimal especially when the columns of the predetermined dictionary are highly correlated. In addition, the l_1 minimization methods may not be computationally efficient especially when they employ third party solvers, such as the matlab software for convex optimization [11].

To find a more sparse solution, various improved regularization methods have been proposed [12, 13]. Most modified algorithms are iteratively l_1 reweighted methods since it has been shown that weighted l_1 minimization methods tend to perform better than conventional l_1 regularization technique under certain conditions [13]. There are also some proposed methods under the Bayesian framework to produce a sparse solution by introducing hyper prior [14, 15]. Recently, SALSA algorithm is originally proposed to solve the l_1 minimization problem in the field of image processing [16, 17, 18]. It converts the original optimization problem into several subproblems which can be addressed separately without the third party solver. Actually, SALSA is an instance of Alternating Direction Method of Multipliers which has been widely used as a distributed optimization and statistical learning method [19]. The SALSA combines augmented Lagrangian and dual decomposition method, which can address constrained optimization problems in a computationally efficient manner. SALSA has not been used for NARX modelling. In this paper, it is found that SALSA may not build a sparse NARX model. This may be due to the high correlation among NARX model terms.

Sampling technique is another popular technique to enhance the performance of subset selection methods. For example, the Markov chain Monte Carlo (MCMC) sampling method is used for NARX model identification to obtain a more accurate solution [20]. Most recently, stability selection was proposed as a promising but a general approach to improve model sparsity. The stability selection was originally proposed in statistics, which solves the structure determination problem by using subsampling approach. Specifically, only the highly selected terms in those intermediate models generated with subsampling technique will be selected into the final model. Stability selection can not only help to select the number of variables but also provide a new variable selection scheme [21].

In this paper, SAL method is proposed to build a more compact model without using third party solver. With random subsampling technique, SALSA is used to produce some intermediate models, then only choose the highly selected terms of those intermediate models into the final model. Finally, SALSA is applied again for parameters estimation. The proposed SAL method enjoys the advantages of both stability selection and SALSA, which are summarized as follows

- First, SAL is easy to implement since it is under the SALSA framework, which means SAL converts

the l_1 optimization problem into several subproblems to be separately solved without the third party solver.

- Second, stability selection can be used to produce a more compact model which only includes the highly selected terms.
- Third, with the random subsampling technique, it is not necessary to use all the collected data for system identification. This is especially useful to reduce computational time when processing the large data.

The model sparsity and algorithm convergence can be guaranteed through theoretical analysis. Two nonlinear examples and one real-world application from the process industry are used to show the effectiveness of the proposed method.

The paper is structured as follows. In section 2, NARX model and SALSA are briefly reviewed. Then the idea of SAL is introduced. Section 4 gives the theoretical analysis of SAL. In the next section, two nonlinear examples and one real data set from the process industry system are used to show the performance of SAL. Finally, we give some conclusions.

2. Preliminary

2.1. NARX model

The linear-in-the-parameters structure for NARX can be described by a combination of some unknown functions given by [7] :

$$\begin{aligned} y(t) &= f(y(t-1), \cdots, y(t-n_y), u(t-1), \cdots, u(t-n_u)) + \xi(t) \\ &= f(\mathbf{x}(t)) + \xi(t) \end{aligned} \tag{1}$$

where $u(t)$, $y(t)$ and $\xi(t)$ represent system input, output and noise at the time interval $t = 1, 2, \cdots, N$, respectively, with N being the length of training data. n_u and n_y are the largest lags of input and output. $f(\cdot)$ is the unknown function. For convenience, rewrite the model input $\mathbf{x}(t) = \{y(t-1), \cdots, y(t-n_y), u(t-1), \cdots, u(t-n_u)\}$ as $\mathbf{x}(t) = \{x_1(t), \cdots, x_r(t)\}$ with $r = n_u + n_y$.

The NARX model given by (1) can be rewritten as a linear weighted sum of some unknown nonlinear functions:

$$y(t) = \sum_{i=1}^{M} p_i\big(\mathbf{x}(t)\big)\Theta_i + \xi(t) \tag{2}$$

where p_i is the nonlinear function and Θ_i is the coefficient to be estimated for $i = 1, \cdots, M$. The equation (2) can be represented as the matrix format

$$\mathbf{y} = \mathbf{P}\boldsymbol{\Theta} + \xi \tag{3}$$

with

$$\mathbf{y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix}, \ \mathbf{\Theta} = \begin{bmatrix} \Theta_1 \\ \Theta_2 \\ \vdots \\ \Theta_M \end{bmatrix}, \ \xi = \begin{bmatrix} \xi(1) \\ \xi(2) \\ \vdots \\ \xi(N) \end{bmatrix} \tag{4}$$

Here $\mathbf{y}$, $\mathbf{\Theta}$ and ξ represent the system output, parameter being estimated and residual, respectively. The $N \times M$ candidate dictionary $\mathbf{P}$ is given as

$$\mathbf{P} = [\mathbf{p}_1, \cdots, \mathbf{p}_M] \tag{5}$$

with $\mathbf{p}_i = [p_i(\mathbf{x}(1)), \cdots, p_i(\mathbf{x}(N))]^T$

To produce a sparse solution, the regression problem (3) is generally solved from the viewpoint of the following l_1 minimization problem

$$\widehat{\mathbf{\Theta}} = \arg\min_{\mathbf{\Theta}} \{ \frac{1}{2} \|\mathbf{y} - \mathbf{P}\mathbf{\Theta}\|_2^2 + \lambda \|\mathbf{\Theta}\|_1 \} \tag{6}$$

In recent years, SALSA is proposed as an alternative method to solve the problem (6) by converting the optimization problem into several suboptimization problems solved separately without the third party solver.

2.2. SALSA

2.2.1. Converting into subproblems

The l_1 minimization problem (6) can be rewritten as

$$\min_{\mathbf{\Theta} \in \mathbb{R}^M} f_1(\mathbf{\Theta}) + f_2(\mathbf{\Theta}) \tag{7}$$

with $f_1(\mathbf{\Theta}) = \frac{1}{2}||\mathbf{y} - \mathbf{P}\mathbf{\Theta}||_2^2$ and $f_2(\mathbf{\Theta}) = \lambda \|\mathbf{\Theta}\|_1$. Introduce a new variable vector v to replace $\mathbf{\Theta}$ in function f_2, then we have the following constrained problem

$$\begin{aligned} \min_{\mathbf{\Theta}, v \in \mathbb{R}^M} \quad & f_1(\mathbf{\Theta}) + f_2(v) \\ s.t. \quad & v = \mathbf{\Theta} \end{aligned} \tag{8}$$

According to the constraint of the problem (8), we have $\|\mathbf{\Theta} - v\|_2^2 = \mathbf{0}$, and the optimization problem can be redescribed as follows:

$$\begin{aligned} \min_{\mathbf{\Theta}, v \in \mathbb{R}^M} \quad & f_1(\mathbf{\Theta}) + f_2(v) + \frac{\mu}{2} \|\mathbf{\Theta} - v\|_2^2 \\ s.t. \quad & v - \mathbf{\Theta} = \mathbf{0} \end{aligned} \tag{9}$$

where μ is the Lagrange multiplier. The value of μ has an impact on the solution of problem (9), namely, increasing μ makes the solution gradually approximate that of problem (8).

With Augmented Lagrangian (AL) method, problem (9) can be transformed into the following unconstrained optimization problem

$$L_\mu(\mathbf{\Theta}, v, u) = f_1(\mathbf{\Theta}) + f_2(v) - u^T(\mathbf{\Theta} - v) + \frac{\mu}{2} \|\mathbf{\Theta} - v\|_2^2 \tag{10}$$

where u is the dual variable. Replace u by variable $d = u/\mu$, then problem (10) can be solved by alternately minimizing the following subproblems [16]

$$\widehat{\mathbf{\Theta}}_{k+1} = \arg\min_{\mathbf{\Theta}} f_1(\mathbf{\Theta}) + \frac{\mu}{2}\|\mathbf{\Theta} - v_k - d_k\|_2^2 \tag{11}$$

$$v_{k+1} = \arg\min_{v} f_2(v) + \frac{\mu}{2}\|\widehat{\mathbf{\Theta}}_{k+1} - v - d_k\|_2^2 \tag{12}$$

$$d_{k+1} = d_k - (\widehat{\mathbf{\Theta}}_{k+1} - v_{k+1}) \tag{13}$$

3. The proposed method

Although the original SALSA provides an alternative way to solve the l_1 minimization problem, it may not produce a sparse solution. In this paper, stability selection is introduced to select the highly selected terms in the intermediate models produced by SALSA, leading to a parsimonious model. The proposed SAL algorithm consists of two procedures, namely variable selection and parameter estimation. The first stage of SAL aims to select variables for finding a suitable subset of some possible variables. Then SALSA is further applied for parameters estimation at the second stage. It is worth pointing out that variable selection and parameter estimation may be simultaneously implemented by SALSA at second stage in certain cases, leading to a more compact model by discarding falsely selected terms. The main idea of SAL shows as Fig.1. Firstly, we give the exact solutions of subproblems.

3.1. Solving the subproblems

The estimation $\widehat{\mathbf{\Theta}}$ can be obtained by solving the following subproblem

$$\arg\min_{\mathbf{\Theta}} \frac{1}{2}\|\mathbf{P\Theta} - \mathbf{y}\|_2^2 + \frac{\mu}{2} \parallel \mathbf{\Theta} - v_k - d_k \parallel_2^2 \tag{14}$$

Define the cost function of problem (14) as $J(\mathbf{\Theta})$ which can be rewritten as the following format

$$J(\mathbf{\Theta}) = \frac{1}{2}(\mathbf{P\Theta} - \mathbf{y})^T(\mathbf{P\Theta} - \mathbf{y}) + \frac{\mu}{2}(\mathbf{\Theta} - v_k - d_k)^T(\mathbf{\Theta} - v_k - d_k) \tag{15}$$

The optimal solution of the equation (15) satisfies

$$\nabla J(\mathbf{\Theta}) = \mathbf{P}^T\mathbf{P\Theta} - \mathbf{P}^T\mathbf{y} + \mu(\mathbf{\Theta} - (v_k + d_k)) = 0 \tag{16}$$

Then $\widehat{\mathbf{\Theta}}_{k+1}$ can be calculated by

$$\widehat{\mathbf{\Theta}}_{k+1} = (\mathbf{P}^T\mathbf{P} + \mu\mathbf{I})^{-1}\big(\mathbf{P}^T\mathbf{y} + \mu(v_k + d_k)\big) \tag{17}$$

Given $\widehat{\mathbf{\Theta}}_{k+1}$, v_{k+1} can be obtained from the following suboptimization problem

$$\arg\min_{v} \lambda\|v\|_1 + \frac{\mu}{2} \parallel \widehat{\mathbf{\Theta}}_{k+1} - v - d_k \parallel_2^2 \tag{18}$$

The problem (18) can be exactly solved by introducing the soft thresholding operator $S_{\mu/\lambda}$ [22]

$$S_{\mu/\lambda}(x) = \max(0, x - \mu/\lambda) - \max(0, -x - \mu/\lambda) \tag{19}$$

with λ being the penalty parameter. With the operator $S_{\mu/\lambda}$, the estimation of v_{k+1} can be obtained by

$$v_{k+1} = \max\left(0, (\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\right) - \max\left(0, -(\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\right) \tag{20}$$

Given $\widehat{\mathbf{\Theta}}_{k+1}$ and v_{k+1}, then d_{k+1} can be calculated by

$$d_{k+1} = d_k - (\widehat{\mathbf{\Theta}}_{k+1} - v_{k+1}) \tag{21}$$

3.2. Variable selection

Stability selection was proposed for choosing a proper amount of terms from the dictionary. This is achieved by using subsampling method which yields a new way for variable selection. For better understanding, giving some definitions is necessary.

Define that N_s represents the number of random subsampling and $N_s = N/2$ is recommended according to literature [21]. In addition, suppose $\widetilde{\mathbf{y}}_j$ and $\widetilde{\mathbf{P}}_j$ represent the j_{th} subsampling data of system output and input for $j = 1, \cdots, n$, then we have $\widetilde{\mathbf{y}}_j \in \mathbb{R}^{N_s}$ and $\widetilde{\mathbf{P}}_j \in \mathbb{R}^{N_s \times M}$. It is worth noting that for every subsampling data, SALSA runs the same procedure to produce an intermediate model. With these statements, the specific procedure of variable selection is summarized as follows.

Here, suppose that SALSA is used to build the j_{th} intermediate model with the subsampling data $\widetilde{\mathbf{P}}_j$ and $\widetilde{\mathbf{y}}_j$, $j = 1, \cdots, n$. Specifically, at the first iteration (with $v_0 = \mathbf{0}$ and $d_0 = \mathbf{0}$), $\widehat{\mathbf{\Theta}}_1$ can be calculated

$$\widehat{\mathbf{\Theta}}_1 = (\widetilde{\mathbf{P}}_j^T \widetilde{\mathbf{P}}_j + \mu \mathbf{I})^{-1} \widetilde{\mathbf{P}}_j^T \widetilde{\mathbf{y}}_j \tag{22}$$

Given $\widehat{\mathbf{\Theta}}_1$, then v_1 and d_1 can be alternately estimated

$$\begin{aligned} v_1 &= \max\left(0, \widehat{\mathbf{\Theta}}_1 - \mu/\lambda\right) - \max\left(0, -\widehat{\mathbf{\Theta}}_1 - \mu/\lambda\right) \\ d_1 &= -(\widehat{\mathbf{\Theta}}_1 - v_1) \end{aligned} \tag{23}$$

At the $k+1_{th}$ iteration, $\widehat{\mathbf{\Theta}}_{k+1}$, v_{k+1} and d_{k+1} can be updated

$$\begin{aligned} \widehat{\mathbf{\Theta}}_{k+1} &= (\widetilde{\mathbf{P}}_j^T \widetilde{\mathbf{P}}_j + \mu \mathbf{I})^{-1} \left(\widetilde{\mathbf{P}}_j^T \widetilde{\mathbf{y}}_j + \mu(v_k + d_k)\right) \\ v_{k+1} &= \max\left(0, (\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\right) - \max\left(0, -(\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\right) \\ d_{k+1} &= d_k - (\widehat{\mathbf{\Theta}}_{k+1} - v_k) \end{aligned} \tag{24}$$

The following assumption will be firstly introduced before giving the stopping criterion.

Assumption 1: Suppose the signs and locations of nonzero coefficients of $\widehat{\mathbf{\Theta}}_i (i = 1, 2, \cdots, k)$ obtained during the iterative process are different before the coefficients converge. In addition, assume that at the $k+1_{th}$ step, $\widehat{\mathbf{\Theta}}_{k+1}$ could converge to $\mathbf{\Theta}^*$ as long as with suitable setting of λ and μ.

According to **Assumption 1**, the procedure of SALSA stops when it satisfies

$$sign(\widehat{\mathbf{\Theta}}_{k+1}) = sign(\widehat{\mathbf{\Theta}}_k) \tag{25}$$

$$loc_{k+1} = loc_k \tag{26}$$

Then the term with non-zero coefficient will be selected into the intermediate model defined as $Model_j = [\mathbf{p}_i : \widehat{\Theta}_i \neq 0]$, where $\widehat{\Theta}_i$ is the i_{th} element of $\widehat{\mathbf{\Theta}}_{k+1}$.

Remark 1: Although there might be no exact zero coefficients at each iterative step, comparing with other coefficients, some weights might be very small, e.g., $\|\Theta_i\|_2^2 \ll \|\mathbf{\Theta}\|_2^2$. In this case, to obtain a compact model, those small weights lower than a predetermined threshold will be pruned at each iteration.

Suppose $sn(\mathbf{p}_i)$ represents the number of term $\mathbf{p}_i$ selected by all n intermediate models. Then the term selection frequency $sf(\mathbf{p}_i)$ is defined as

$$sf(\mathbf{p}_i) = \frac{sn(\mathbf{p}_i)}{n} \tag{27}$$

where $sf(\mathbf{p}_i)$ ranges from 0 to 1 . If $sf(\mathbf{p}_i)$ is larger than the predetermined threshold δ_{thr} with $0 \leq \delta_{thr} \leq 1$, which indicates the term $\mathbf{p}_i$ has a high selection frequency and should be selected.

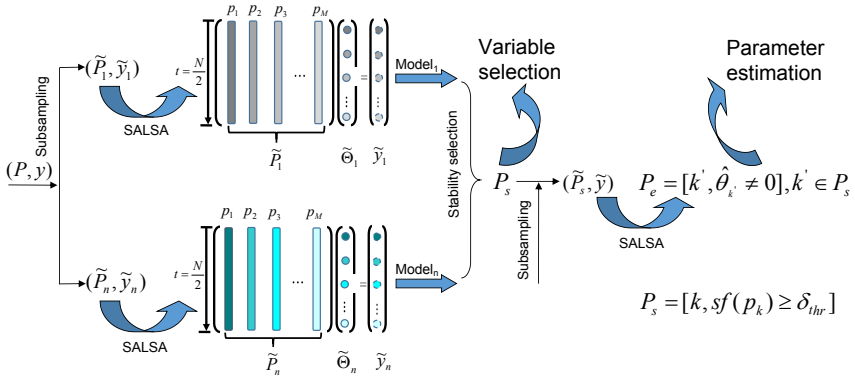

Figure 1: The main idea of SAL algorithm. Here $\mathbf{P}_s$ and $\mathbf{P}_e$ represent model terms sets of variable selection and parameter estimation stage, respectively.

A simple example is used to illustrate how the stability selection works. Suppose the candidate dictionary includes six model terms $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6]$. With subsampling technique, suppose the following five intermediate models are produced by SALSA shown as Table 1.

Table 1: The intermediate models produced by SALSA

Model	Selected Terms
$Model_1$	$[\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_5]$
$Model_2$	$[\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_6]$
$Model_3$	$[\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5]$
$Model_4$	$[\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_5]$
$Model_5$	$[\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_5, \mathbf{p}_6]$

Remark 2: SAL initially builds multiple intermediate models using different training data generated by subsampling, where the amount of those intermediate models is equal to the number of subsamplings. In most cases, 100 subsamplings are sufficient, say $n = 100$. The proposed SAL method only includes highly

frequently selected terms into the final model but excludes those terms with low selecting probabilities. This makes a contribution to build a spare model.

Table 2: The selection frequency of each term

Term	$\mathbf{p}_1$	$\mathbf{p}_2$	$\mathbf{p}_3$	$\mathbf{p}_4$	$\mathbf{p}_5$	$\mathbf{p}_6$
$sn(\mathbf{p}_i)$	5	0	5	1	4	2
$sf(\mathbf{p}_i)$	1	0	1	0.2	0.8	0.4

Following the results in Table 1, we can calculate the selection frequency of each model term shown as Table 2. Generally, the range of δ_{thr} is recommended from 0.6 to 0.9 by the literature [21]. Therefore, given a predetermined threshold, say $\delta_{thr} = 0.6$, then the term $\mathbf{p}_1$,$\mathbf{p}_3$,$\mathbf{p}_5$ will be selected, while the term $\mathbf{p}_2$,$\mathbf{p}_4$,$\mathbf{p}_6$ will be discarded since their low selection frequency. Then the final model obtained at variable selection stage can be described as $\mathbf{P}_s = [\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_5]$.

3.3. Parameter estimation

The coefficients of those selected variables will be further estimated with SALSA. Suppose that $\mathbf{P}_s = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{n_s}]$ is the set of selected terms $\mathbf{p}_i$ with $sf(\mathbf{p}_i) \geq \delta_{thr}$ and $\mathbf{\Theta}_s = [\Theta_1, \Theta_2, \cdots, \Theta_{n_s}]^T$ is the set including related coefficients. Here $\mathbf{P}_s \in \mathbb{R}^{N \times n_s}$, $\mathbf{\Theta}_s \in \mathbb{R}^{n_s}$ and n_s is the number of the selected terms. Since most terms of the dictionary are redundant and the related coefficients should be zero in the final model, then we can rewrite $\mathbf{P}$ and $\mathbf{\Theta}$ as $\mathbf{P} = [\mathbf{P}_s, \mathbf{0}]$ and $\mathbf{\Theta} = \begin{bmatrix} \mathbf{\Theta}_s \\ \mathbf{0} \end{bmatrix}$. Then the equation (3) can be represented as $\mathbf{y} = \mathbf{P}_s\mathbf{\Theta}_s + \xi$, where $\mathbf{y}$ represents system output and ξ represents the residual. With subsampling technique, we have $\widetilde{\mathbf{y}} \in \mathbb{R}^{N_s}$ and $\widetilde{\mathbf{P}}_s \in \mathbb{R}^{N_s \times M}$. For convenience, suppose $\widehat{\mathbf{\Theta}}^e$ is the estimation of $\mathbf{\Theta}_s$. Then $\widehat{\mathbf{\Theta}}^e$ can be iteratively estimated as following

$$\begin{aligned} \widehat{\mathbf{\Theta}}^e_{k+1} &= (\widetilde{\mathbf{P}}_s^T\widetilde{\mathbf{P}}_s + \mu\mathbf{I})^{-1}\big(\widetilde{\mathbf{P}}_s^T\widetilde{\mathbf{y}} + \mu(v_k + d_k)\big) \\ v_{k+1} &= \max\big(0, (\widehat{\mathbf{\Theta}}^e_{k+1} - d_k) - \mu/\lambda\big) - \max\big(0, -(\widehat{\mathbf{\Theta}}^e_{k+1} - d_k) - \mu/\lambda\big) \\ d_{k+1} &= d_k - (\widehat{\mathbf{\Theta}}^e_{k+1} - v_k) \end{aligned} \tag{28}$$

Here $\mathbf{P}_e = [\mathbf{p}_i : \widehat{\Theta}_i \neq 0]$ is the model set obtained at parameter estimation stage and $\widehat{\Theta}_i$ is the i_{th} element of $\widehat{\mathbf{\Theta}}^e$.

Remark 3: Both pruning and stability selection can contribute to model sparsity. Pruning can remove the terms little or negative contribution to the model performance. For example, some redundant terms may fit into noise when they enter the model. The small term coefficients indicate their insignificant impact on the model performance. These noisy terms can be excluded with the pruning technique. Stability selection excludes the redundant terms according to the frequency of presence in all the intermediate models. For example, the redundant terms with low selecting probability can be removed by stability selection, no matter their coefficients are large or small. The use of both pruning and stability selection can significantly improve model sparsity, since they are compensatable to each other. For example, in the case of terms with large

coefficients but low selecting probability, stability selection instead of pruning takes effects. In the case of terms with small coefficients but high selecting probability, pruning other than stability selection takes effects. It is worth pointing out that model $\mathbf{P}_s$ generated at the first stage (variable selection) may not be optimal, especially when certain correlate terms with high cross correlation weights. In this case, the redundant terms can be further penalized due to the use of SALSA at the second stage (parameter estimation), leading to a more compact model.

3.4. The main procedure of SAL

With SALSA and stability selection, the proposed SAL method can produce a more compact model. The main procedure of SAL is summarized as **Algorithm 1**.

Algorithm 1 : SAL

Variable selection stage:

Set μ, λ and $v_0 = d_0 = \mathbf{0}$

For j=1:n

  random subsampling $\rightarrow \widetilde{\mathbf{y}}_j$ and $\widetilde{\mathbf{P}}_j$

  Repeat

    $\widehat{\mathbf{\Theta}}_{k+1} = (\widetilde{\mathbf{P}}_j^T \widetilde{\mathbf{P}}_j + \mu \mathbf{I})^{-1} \big(\widetilde{\mathbf{P}}_j^T \widetilde{\mathbf{y}}_j + \mu(v_k + d_k)\big)$

    $v_{k+1} = \max\big(0, (\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\big) -$
    $\max\big(0, -(\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\big)$

    $d_{k+1} = d_k - (\widehat{\mathbf{\Theta}}_{k+1} - v_{k+1})$

    $k \leftarrow k + 1$

  until stopping criterion (25) is satisfied.

end

if $sf(\mathbf{p}_i) \geq \delta_{thr}$
  then term $\mathbf{p}_i$ is selected.

end

Parameter estimation stage:

random subsampling $\rightarrow \widetilde{\mathbf{y}}$ and $\widetilde{\mathbf{P}}_s$

Repeat

  $\widehat{\mathbf{\Theta}}^e_{k+1} = (\widetilde{\mathbf{P}}_s^T \widetilde{\mathbf{P}}_s + \mu \mathbf{I})^{-1} \big(\widetilde{\mathbf{P}}_s^T \widetilde{\mathbf{y}} + \mu(v_k + d_k)\big)$

  $v_{k+1} = \max\big(0, (\widehat{\mathbf{\Theta}}^e_{k+1} - d_k) - \mu/\lambda\big) -$
  $\max\big(0, -(\widehat{\mathbf{\Theta}}^e_{k+1} - d_k) - \mu/\lambda\big)$

  $d_{k+1} = d_k - (\widehat{\mathbf{\Theta}}^e_{k+1} - v_k)$

  $k \leftarrow k + 1$

until stopping criterion (25) is satisfied.

Remark 4: The new SAL method first determines the model terms via the stability selection and then estimates the model parameters via the SALSA. Two steps have no dependence with each other and the proposed method is not restricted to SALSA for parameter estimation. In other words, the proposed method can employ other parameter identification methods, such as bias-eliminated Least Squares [23], iteratively reweighted Least Squares methods [24] or penalized MM-estimators [25]. Although these methods can be used for parameter estimation, they are out of scope of this paper and will be used for future study.

Remark 5: Although the proposed SAL is able to reduce the redundant terms, SAL requires more computations comparing with the original SALSA since SAL applies subsampling technique. If the subsampling times are chosen as 100, namely $n = 100$, there are 100 intermediate models. As each intermediate model only use half data, it requires approximately $1/4$ computations, compared to using the whole data with SALSA method. Therefore, SAL constructing 100 intermediate models, needs about 25 times of SALSA with the whole data. For the standard SALSA, the solution is computed directly, which means one must compute $(\widetilde{\mathbf{P}}_j^T\widetilde{\mathbf{P}}_j + \mu\mathbf{I})^{-1}_{M\times M}$ and other three matrix multiplications. Thus directly solving the solution $\widehat{\mathbf{\Theta}}_{k+1} = (\widetilde{\mathbf{P}}_j^T\widetilde{\mathbf{P}}_j + \mu\mathbf{I})^{-1}\big(\widetilde{\mathbf{P}}_j^T\widetilde{\mathbf{y}}_j + \mu(v_k + d_k)\big)$, the computation cost T measured by the amount of multiplies is $T = \mathcal{O}(M^3 + M^2 + MN_s + 2M)$. The leading cost of each iteration of SALSA will be either $\mathcal{O}(M^3)$ or $\mathcal{O}(MN_s)$. In the experiments, the predetermined value of N_s is smaller than M^2, so each iteration has $\mathcal{O}(M^3)$ cost.

4. Theoretical analysis

4.1. The convergence

The proposed SAL is under the SALSA framework. Therefore, if SALSA is convergent then we can get the same conclusion for SAL.

Theorem 1 [26]: Consider problem (7), where f_1 and f_2 are closed, proper convex functions. Suppose arbitrary $\mu > 0$ and $v_0, d_0 \in \mathbb{R}^M$. Let $\{\eta_k \geq 0, k = 0, 1, \cdots \infty\}$ and $\{\nu_k \geq 0, k = 0, 1, \cdots \infty\}$ be two sequences that satisfy

$$\sum_{k=0}^{\infty} \eta_k < \infty \quad and \quad \sum_{k=0}^{\infty} \nu_k < \infty \tag{29}$$

Consider three sequences $\{\widehat{\mathbf{\Theta}}_k \in \mathbb{R}^M, k = 0, 1, \cdots\}$, $\{v_k \in \mathbb{R}^M, k = 0, 1, \cdots\}$ and $\{d_k \in \mathbb{R}^M, k = 0, 1, \cdots\}$ that satisfy

$$\eta_k \geq \left\| \widehat{\mathbf{\Theta}}_{k+1} - \arg\min_{\mathbf{\Theta}} f_1(\mathbf{\Theta}) + \frac{\mu}{2}\|\mathbf{\Theta} - v_k - d_k\|_2^2 \right\| \tag{30}$$

$$\nu_k \geq \left\| v_{k+1} - \arg\min_{v} f_2(v) + \frac{\mu}{2}\|\widehat{\mathbf{\Theta}}_{k+1} - v - d_k\|_2^2 \right\| \tag{31}$$

$$d_{k+1} = d_k + \widehat{\mathbf{\Theta}}_{k+1} - v_{k+1} \tag{32}$$

If problem (7) has a solution, the sequence $\{\widehat{\mathbf{\Theta}}_k\}$ converges, namely, $\widehat{\mathbf{\Theta}}_k \to \mathbf{\Theta}^*$, where $\mathbf{\Theta}^*$ is a solution of (7). However, if there is no solution for (7), then at least one of $\{v_k\}$ or $\{d_k\}$ diverges.

Here we give the convergence proof of SALSA based on **Theorem 1**. According to the proof of the literature [16], SALSA is convergent if the following two subproblems

$$\arg\min_{\mathbf{\Theta}} f_1(\mathbf{\Theta}) + \frac{\mu}{2}\|\mathbf{\Theta} - v_k - d_k\|_2^2 \tag{33}$$

and

$$\arg\min_{v} f_2(v) + \frac{\mu}{2} \parallel \widehat{\mathbf{\Theta}}_{k+1} - v - d_k \parallel_2^2 \tag{34}$$

can be solved exactly. Since $f_1(\mathbf{\Theta}) = \frac{1}{2}\|\mathbf{P\Theta} - \mathbf{y}\|_2^2$ and $f_2(v) = \lambda\|v\|_1$, the solutions of subproblems above can be solved exactly as mentioned in section 2.2.2, namely

$$\widehat{\mathbf{\Theta}}_{k+1} = (\mathbf{P}^T\mathbf{P} + \mu\mathbf{I})^{-1}\big(\mathbf{P}^T\mathbf{y} + \mu(v_k + d_k)\big) \tag{35}$$

and

$$v_{k+1} = \max\big(0, (\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\big) - \max\big(0, -(\widehat{\mathbf{\Theta}}_{k+1} - d_k) - \mu/\lambda\big) \tag{36}$$

Then the convergence of SALSA is guaranteed since subproblems (9),(10) can be solved exactly [16]. Therefore, the proposed SAL algorithm is also convergent as aforementioned.

4.2. The number of falsely selected terms

Although stability selection only select the highly selected terms of the intermediate models, redundant terms may also be included into the final model. Generally, the redundant terms selected into the final model are considered as the falsely selected terms and the amount can be bounded. For analyzing the amount of falsely selected terms, some basic concepts should be introduced in advance. Suppose S and Z are set of important and redundant terms, respectively, where $S = \{k : \Theta_k \neq 0\}$ and $Z = \{k : \Theta_k = 0\}$ with $k = 1, 2, \cdots, M$. Meanwhile, $\widehat{S} = \{k : \widehat{\Theta}_k \neq 0\}$ is considered as the estimation of the set S.

Remark 6: For the variable selection procedure of SAL algorithm, there are two tuning parameters $\lambda \in \Lambda \subseteq \mathbb{R}^+$ and $\mu \in \Lambda \subseteq \mathbb{R}^+$, respectively. Here Λ is the set which contains the value of tuning parameters. Even if the value of μ is fixed, for every $\lambda \in \Lambda$, we obtain an estimation of the set $\widehat{S}$ which is marked as $\widehat{S}^\lambda$ and vice versa. Therefore, tuning two parameters simultaneously makes the variable selection and the related theoretical analysis more intractable. For analyzing the amount of falsely selected terms, we set $\mu = \lambda$, such that a certain bound can be analyzed theoretically under certain assumptions.

For a given $\lambda \in \Lambda$ and a random subsample $\mathbf{I}_s$ with size N_s, we can get the selected set $\widehat{S}^\lambda(\mathbf{I}_s)$ and let $\widehat{S}^\lambda$=$\widehat{S}^\lambda(\mathbf{I}_s)$. The following definitions and assumption are also necessary for analyzing the amount of the falsely selected variables at the first stage.

Definition 1 (Selection probabilities): For every set $K \subseteq \{1, 2, \cdots, M\}$, $\widehat{\Pi}_K^\lambda$ represents the probability of K being in the structure estimate set $\widehat{S}^\lambda(\mathbf{I}_s)$, which is defined as

$$\widehat{\Pi}_K^\lambda = P(K \subseteq \widehat{S}^\lambda(\mathbf{I}_s)) \tag{37}$$

where P is with respect to random subsampling. Meanwhile, $\widehat{\Pi}_k^\lambda$ represents the selection probability of every variable being in the set $\widehat{S}^\lambda(\mathbf{I}_s)$ for $k = 1, \cdots, M$.

Definition 2 (Stable variables): For given $\lambda \in \Lambda$, the set $\mathbf{P}_s$ including stable variables is defined as

$$\mathbf{P}_s = \{k : \max_{\lambda \in \Lambda} \widehat{\Pi}_k^\lambda \geq \delta_{thr}\} = \{k : sf(\mathbf{p}_i) \geq \delta_{thr}\} \tag{38}$$

Definition 3 (False selections): Define $\widehat{S}^\Lambda = \cup_{\lambda \in \Lambda} \widehat{S}^\lambda$ if tuning the value of λ and q is the average amount of the selected variables with $q = E(|\widehat{S}^\Lambda(\mathbf{I}_s)|)$. Then the amount of falsely selected variables can be defined

$$V = |\mathbf{P}_s \cap Z| \tag{39}$$

Assumption 2: For $\lambda \in \Lambda$, suppose that

$$\frac{E(|S \cap \widehat{S}^\lambda|)}{|S|} \geq \frac{E(|Z \cap \widehat{S}^\lambda|)}{|Z|} \tag{40}$$

which means the random guessing is not better than the original procedure.

Theorem 2: With the stronger **Assumption 2**, the average amount of the falsely selected terms in model set $\mathbf{P}_s$ could be bounded by the following inequality

$$E(V) \leq \frac{1}{2\delta_{thr} - 1} \frac{q^2}{M} \tag{41}$$

Proof of Theorem 2: For simplifying the proof, define $Z_\Lambda = Z \cap \widehat{S}^\Lambda$ is the set of redundant variables in set Z selected into $\widehat{S}^\Lambda$, similarly, $U_\Lambda = S \cap \widehat{S}^\Lambda$ is the set of variables in set S selected into $\widehat{S}^\Lambda$. Then the average amount of falsely selected terms can be calculated by $E(|Z_\Lambda|) = E(|\widehat{S}^\Lambda|) - E(|U_\Lambda|) = q - E(|U_\Lambda|)$. According to **Assumption 2**, $E(|U_\Lambda|) \geq E(|Z_\Lambda|)|S|/|Z|$, then we have $(1 + |S|/|Z|)E(|Z_\Lambda|) \leq q$ by eliminating variable $E(|U_\Lambda|)$. With simple operations, then we have $|Z|^{-1}E(|Z_\Lambda|) \leq q/M$. Here assume that for $\lambda \in \Lambda$, the distribution of $\{1_{k \in \widehat{S}^\lambda}, k \in Z\}$ is exchangeable, then we further obtain

$$(k \in \widehat{S}^\Lambda) = E(|Z_\Lambda|)/|Z| \leq q/M \tag{42}$$

for $k \in Z$. With the inequality (42), the sample splitting will be further introduced in the following part for proving the inequality (41).

Suppose $\mathbf{I}_1$ and $\mathbf{I}_2$ are two random subset obtained by random splitting the data into two non-overlapping part with size N_s and $\mathbf{I}_1 \cap \mathbf{I}_2 = \emptyset$. Similar with the definition of selection probability, define the simultaneously selection probability

$$\widehat{\Pi}_K^{sim,\lambda} = P(K \subseteq \widehat{S}^{sim,\lambda}) \tag{43}$$

where P is with respect to random sample splitting and $\widehat{S}^{sim,\lambda} = \widehat{S}^\lambda(\mathbf{I}_1) \cap \widehat{S}^\lambda(\mathbf{I}_2)$ represents the simultaneously selected set.

If $P(K \subseteq \cup_{\lambda \in \Lambda} \widehat{S}^\lambda) \leq \varepsilon$ for some $\Lambda \subseteq \mathbb{R}^+$, then $P(\max_{\lambda \in \Lambda} \widehat{\Pi}_K^{sim,\lambda} \geq \xi) \leq \varepsilon^2/\xi$ for $0 < \xi < 1$ [21]. Therefore, we have $P(\max_{\lambda \in \Lambda} \widehat{\Pi}_k^{sim,\lambda} \geq \xi) \leq (q/M)^2/\xi$ for $k \in Z$. Since the inequality $\widehat{\Pi}_K^{sim,\lambda} \geq 2\widehat{\Pi}_K^\lambda - 1$

holds, it follows $P(\max_{\lambda\in\Lambda}\widehat{\Pi}_k^{\lambda}\geq\delta_{thr})\leq P((\max_{\lambda\in\Lambda}\widehat{\Pi}_k^{sim,\lambda}+1)/2\geq\delta_{thr})\leq(q/M)^2/(2\delta_{thr}-1)$. Finally, the inequality $E(V)=\sum_{k\in Z}P(\max_{\lambda\in\Lambda}\widehat{\Pi}_k^{\lambda}\geq\delta_{thr})\leq\frac{1}{2\delta_{thr}-1}\frac{q^2}{M}$ is approved.

Remark 7: The **Theorem 2** shows that the number of falsely selected variables at the first stage can be bounded under certain assumption. Meanwhile, the second stage may further discard falsely selected terms in certain cases as aforementioned. Therefore, the average amount of redundant terms selected into the final model set $\mathbf{P}_e$ is no more than $\frac{1}{2\delta_{thr}-1}\frac{q^2}{M}$.

5. Simulation

In this section, two nonlinear examples and one simulation data set from the process industry are used to test the effectiveness of the proposed method. SAL is also compared with several popular methods including OFR, Lasso and SALSA in terms of their performance. For fair comparison, each algorithm is repeated many times and the best result is selected as the final model.

5.1. Example 1

First, consider the nonlinear benchmark example [27]:

$$\begin{aligned}z(t)&=0.2z^3(t-1)+0.7z(t-1)u(t-1)\\&\quad+0.6u^2(t-2)-0.5z(t-2)\\&\quad-0.7z(t-2)u^2(t-2)\\y(t)&=z(t)+e(t).\end{aligned}\tag{44}$$

where $u(t)$ represents the system input and $z(t)$ represents the output at interval t. A uniformly distributed white noise u(t) is used to excite the system with u(t) $\in$ [-1,1]. The system output $z(t)$ is disturbed by a Gaussian noise sequence $e(t)$. The signal-to-noise ratio is chosen as 15dB. The model input of the unknown nonlinear system includes the delayed input and output, namely $\{z(t-1),z(t-2),z(t-3),z(t-4),u(t-1),u(t-2),u(t-3)\}$. Total 3000 input and output samples are generated for training and testing. The Mean Absolute Error (MAE) is used to test the model performance [28]

$$MAE=\frac{1}{M}\sum_{i=1}^{M}\left|\boldsymbol{\Theta}_i-\widehat{\boldsymbol{\Theta}}_i\right|\tag{45}$$

where $\widehat{\boldsymbol{\Theta}}$ is the estimation of the true coefficients $\boldsymbol{\Theta}$.

Here, the parameters of OFR, Lasso, SALSA and SAL are defined and determined as $\rho=0.031$, $\lambda_L=0.0042$, $\lambda_S=0.215$ and $\lambda_{SA}=0.3$. All the results are listed in the Table 3, it can be seen that although the coefficients of important terms are close to the true value, OFR selects a redundant term $z(t-4)u(t-2)^2$ into the final model. The reason is that according to the error reduction ration (ERR) criterion, the unimportant term $z(t-4)u(t-2)^2$ is firstly selected into the final model due to its largest ERR value. For SALSA, it can not produce a sparse model, although most coefficients of other 115 variables are small. In addition, Lasso

Table 3: The simulation results for example 1

Algorithm	Selected Terms	Coefficient	MAE
OFR	$z(t-4)u(t-2)^2$	-0.0057	0.0002
	$\mathbf{u(t-2)u(t-2)}$	0.6008	
	$\mathbf{z(t-2)}$	-0.4981	
	$\mathbf{u(t-1)z(t-1)}$	0.6951	
	$\mathbf{z(t-2)u(t-2)^2}$	-0.6989	
	$\mathbf{z(t-1)^3}$	0.1940	
Lasso	$z(t-4)u(t-2)^2$	0.0023	0.0011
	$\mathbf{u(t-2)u(t-2)}$	0.5800	
	$\mathbf{z(t-2)}$	-0.4956	
	$\mathbf{u(t-1)z(t-1)}$	0.6700	
	$\mathbf{z(t-2)u(t-2)^2}$	-0.6527	
	$\mathbf{z(t-1)^3}$	0.1723	
SALSA	$\mathbf{z(t-2)}$	-0.5121	0.0290
	$\mathbf{u(t-1)z(t-1)}$	0.7493	
	$\mathbf{u(t-2)u(t-2)}$	0.5970	
	$\mathbf{z(t-2)u(t-2)^2}$	-0.6376	
	$\mathbf{z(t-1)^3}$	0.1686	
	other 115 terms	$\vdots$	
SAL	$\mathbf{z(t-2)}$	-0.4979	0.0001
	$\mathbf{u(t-1)z(t-1)}$	0.6971	
	$\mathbf{u(t-2)u(t-2)}$	0.6007	
	$\mathbf{z(t-2)u(t-2)^2}$	-0.6970	
	$\mathbf{z(t-1)^3}$	0.1995	

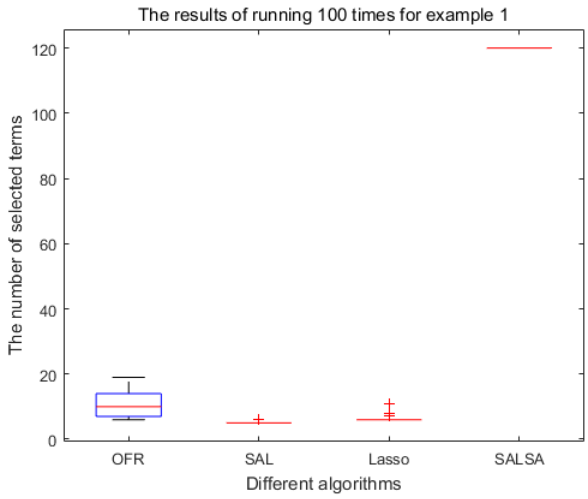

Figure 2: Box plots of the number of model terms produced by four algorithms for example 1.

also includes a redundant term into the final model, while SAL is able to build an optimal model without redundant terms and the parameter estimation of SAL is accurate with MAE being 0.0001.

To test the sensitivity of algorithm to noise, Monte Carlo simulation with 100 repetitions are carried out and the Box plots of the number of model terms are shown in Fig.2. From this figure, one can get the same conclusion with that of Table 3. One can see that in most cases, both SAL and Lasso can produce a more compact model comparing with other algorithms. In addition, OFR often select redundant terms into the model and SALSA can not produce a sparse model.

5.2. Example 2

Consider another sparse nonlinear system [29]:

$$\begin{aligned} z(t) = & -0.3u(t-2) + 0.8z(t-1) + u(t-1) \\ & - 0.4u(t-3) + 0.25u(t-1)u(t-2) \\ & - 0.3u^3(t-1) + 0.24u^3(t-2) \\ & - 0.2u(t-2)u(t-3) \\ y(t) = & \, z(t) + e(t). \end{aligned} \tag{46}$$

where $u(t)$ represents the system input and $z(t)$ represents the output at interval t. A white noise $u(t) \in [-1, 1]$ is used as the system input with uniform distribution and SNR value of 15dB.

There are total 3500 samples generated and the delayed input and output $\{z(t-1), z(t-2), u(t-1), u(t-2), u(t-3)\}$ are used for model input. These delayed inputs and outputs can form a model term candidate pool with 56 polynomial terms. The MAE is also used to test the model performance. The parameters are determined as $\rho = 0.031$, $\lambda_L = 0.0012$, $\lambda_S = 0.019$, and $\lambda_{SA} = 0.01$, respectively and all results are listed in Table 4. One can see that SAL has the smallest MAE with value being 0.0015, which means the parameter estimation of SAL is more accurate than that of other algorithms. In addition, SAL builds a more compact model without redundant terms comparing with other methods. In this numerical example, SAL implements variable selection and parameter estimation simultaneously at the second stage, leading to a more parsimonious model.

Again, Monte-carlo simulation repeated 100 times is carried out and all the results are shown in Fig.3. From the figure, one can also get the conclusion that in most cases, SAL can build a more compact model comparing with OFR, Lasso and SALSA method.

5.3. Process industry data set

The real data of the pH neutralization process taken from the DaISy library [30] is used to illustrate the performance of the proposed method. The input variables of this process industry system include $u_1(t)$ and $u_2(t)$ which represent acid and base solution flow in liters, respectively. The system output $y(t)$ is the pH value of the solution in the tank. Since the slow reaction in chemical process, the long delayed input and output $\{u_1(t-1), u_1(t-2), \cdots, u_1(t-15), u_2(t-1), u_2(t-2), \cdots, u_2(t-15), y_1(t-1), y_1(t-2), \cdots, y_1(t-15)\}$ are

Table 4: The simulation results for example 2

Algorithm	Selected Terms	Coefficient	MAE
OFR	$\mathbf{u(t-1)}$	1.0092	0.0047
	$\mathbf{u(t-2)}$	-0.3832	
	$\mathbf{u(t-1)u(t-2)}$	0.2475	
	$\mathbf{u(t-1)^3}$	-0.3164	
	$\mathbf{u(t-2)u(t-3)}$	-0.2067	
	$u(t-3)^3$	-0.0111	
	$\mathbf{z(t-1)}$	0.8722	
	$\mathbf{u(t-3)}$	-0.4337	
	$\mathbf{u(t-2)^3}$	0.2799	
Lasso	$\mathbf{u(t-1)}$	0.9803	0.0448
	$\mathbf{u(t-2)}$	0.4262	
	$\mathbf{u(t-3)}$	-0.0121	
	$\mathbf{z(t-1)}$	0.0501	
	$\mathbf{u(t-1)u(t-2)}$	0.2473	
	$\mathbf{u(t-1)^3}$	-0.2760	
	other 12 terms	⋮	
SALSA	$\mathbf{u(t-1)}$	1.0057	0.0540
	$\mathbf{u(t-2)}$	-0.3810	
	$\mathbf{u(t-3)}$	-0.4429	
	$\mathbf{z(t-1)}$	0.8673	
	$\mathbf{u(t-1)u(t-2)}$	0.2849	
	$\mathbf{u(t-2)u(t-3)}$	-0.2030	
	$\mathbf{u(t-1)^3}$	-0.3151	
	$\mathbf{u(t-2)^3}$	0.2077	
	other 48 terms	⋮	
SAL	$\mathbf{u(t-1)}$	0.9984	0.0015
	$\mathbf{u(t-2)}$	-0.3192	
	$\mathbf{u(t-3)}$	-0.4005	
	$\mathbf{z(t-1)}$	0.8210	
	$\mathbf{u(t-1)u(t-2)}$	0.2470	
	$\mathbf{u(t-2)u(t-3)}$	-0.1934	
	$\mathbf{u(t-1)^3}$	-0.2967	
	$\mathbf{u(t-2)^3}$	0.2693	

Figure 3: Box plots of the number of model terms produced by four algorithms for example 2.

used for model input and the polynomial NARX model with order up to 2 is used. These chosen variables can produce 1081 polynomial terms. The following mean squared error (MSE) is used to test the model performance

$$MSE = \frac{1}{N}\sum_{t=1}^{N}\left(y(t) - \widehat{y}(t)\right)^2 \tag{47}$$

where $\widehat{y}(t)$ is the prediction of the unknown system.

The simulation data set has 2001 samples, which is divided into two subdata sets. The first 1200 samples are for training data and the other samples are used for testing data. The Lasso and SALSA method select too many redundant terms into the final model, leading to the overfitting phenomenon. Therefore, the obtained models have poor generalization performance and those simulation results are not listed. However, both OFR and SAL can build a sparse model with few terms. The model obtained by OFR is described as

$$\begin{aligned} y(t) = &-0.0840u_2(t-1)y(t-2) + 1.2216u_2(t-1) \\ &+ 0.9663y(t-1) \end{aligned} \tag{48}$$

and the model produced by SAL is

$$y(t) = 1.1138y(t-1) + 0.0781u_2(t-5) - 0.1259y(t-4) \tag{49}$$

According to the results shown in Fig.4-5, it can be seen that SAL builds a linear model to describe the nonlinear industry process system, while the model performs better than that of OFR in terms of MSE on the testing data. A parsimonious model with good generalization performance is always desirable especially for the process industry system, therefore the effectiveness of SAL has been demonstrated by these comparisons.

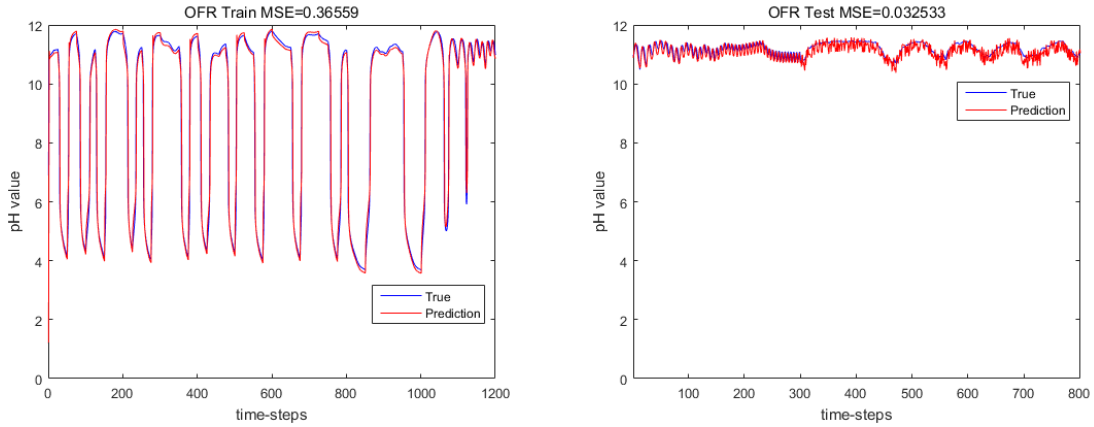

Figure 4: The training and testing performance of OFR.

6. Conclusion

In this paper, Sparse Augmented Lagrangian (SAL) algorithm is proposed to solve the l_1 minimization problem, leading to a sparse solution by implementing variable selection and parameter estimation. Mean-

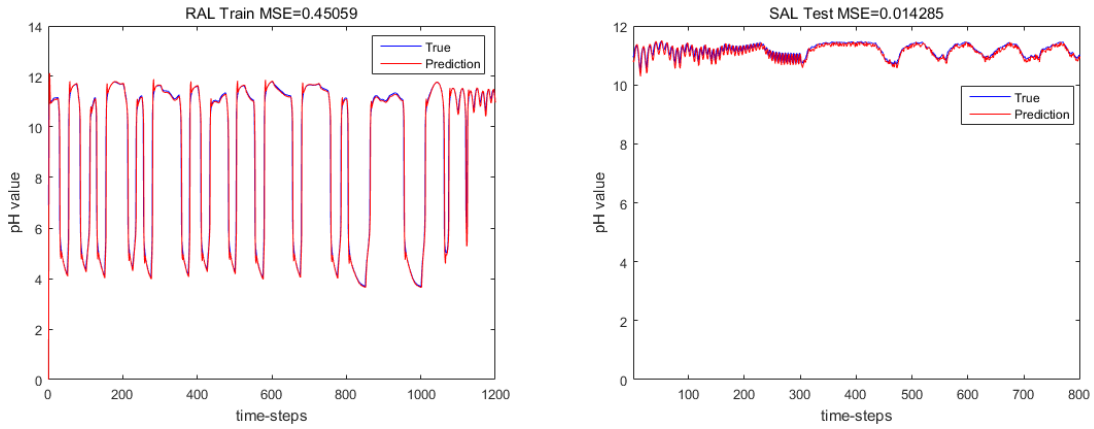

Figure 5: The training and testing performance of SAL.

while, the convergence of SAL is guaranteed. More specifically, at variable selection stage, stability selection is used to choose the highly selected terms of those intermediate models produced by SALSA, where SALSA converts the original optimization problem into several subproblems solved in a computationally efficient manner. At the second stage, SALSA is further used for parameters estimation. In certain cases, variable selection and parameter estimation will be simultaneously carried out for building a more compact model at the second stage. The effectiveness of SAL has been proved in comparison to several popular algorithms. In the near future, instead of directly computing the inverse of matrix like standard SALSA, the authors will adopt other strategies to solve the subproblems so that each iteration could be computed more cheaply.

References

[1] L. Ljung, System Identification: Theory for the user, 2nd Edition, Prentice Hall, 1999.

[2] F. He, H.L. Wei, S.A. Billings, Identification and frequency domain analysis of non-stationary and nonlinear systems using time-varying NARMAX models, *International Journal of Systems Science*, 46(11): 2087–2100, 2015.

[3] F. He, M. Brown, H. Yue, Maximin and Bayesian robust experimental design for measurement set selection in modelling biochemical regulatory systems, *International Journal of Robust & Nonlinear Control*, 20(9): 1059–1078, 2010.

[4] L. Zhang, K. Li, Forward and backward least angle regression for nonlinear system identification, *Automatica*, 53(6): 94-102, 2015.

[5] L. Zhang, K. Li, E.W. Bai, G.W. Irwin, Two-Stage Orthogonal Least Squares Methods for Neural Network Construction, *IEEE Transactions on Neural Networks & Learning Systems*, 26(8): 1608, 2015.

[6] Y.C. Pati, R. Rezaiifar, P.S. Krishnaprasad, Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, *Conference on Signals, Systems & Computers*, 1: 40-44, 2002.

[7] S. Chen, S. A. Billings, W. Luo, Orthogonal least squares methods and their application to non-linear system identification, *International Journal of Control*, 50: 1873–1896, 1989.

[8] A. Sherstinsky, R.W. Picard, On the efficiency of the orthogonal least squares training method for radial basis function networks, *IEEE Transactions on Neural Networks*, 7(1): 195-200, 1996.

[9] L. Rebolloneira, D. Lowe, Optimized orthogonal matching pursuit approach, *IEEE Signal Processing Letters*, 9(4): 134-140, 2002.

[10] Y. Guo, L.Z. Guo, S.A. Billings, H.L. Wei, An Iterative Orthogonal Forward Regression Algorithm, *Internation Journal of Systems Science*, 46(5): 776–789, 2015.

[11] S. Boyd, L. Vandenberghe,Convex Optimization, *Cambridge University Press*, 2004.

[12] D. Wipf, S. Nagarajan, Iterative Reweighted l_1 and l_2 Methods for Finding Sparse Solutions, *IEEE Journal of Selected Topics in Signal Processing*, 4(2): 317-329, 2010.

[13] E. Candes, M. Wakin, S. Boyd, Enhancing sparsity by reweighted l_1 minimization, *Journal of Fourier Analysis & Applications*, 14(5-6): 877-905, 2008.

[14] W. Pan, Y. Yuan, J. Goncalves, G.B. Stan, A Sparse Bayesian Approach to the Identification of Nonlinear State-Space Systems, *IEEE Trasactions on Automatic Control*, 61(1): 182-187, 2015.

[15] W.R. Jacobs, T. Baldacchino, S.R. Anderson, Sparse Bayesian Identification of Polynomial NARX Models, *IFAC-PapersOnline*, 48(28): 172-177, 2015.

[16] M.V. Afonso, J.M. Bioucas-Dias, M.A.T. Figueiredo, Fast image recovery using variable splitting and constrained optimization, *IEEE Transactions on Image Processing*, 19(9): 2345-2356, 2010.

[17] X. Li, B. Zhao, X. Lu, Key Frame Extraction in the Summary Space, *IEEE Transactions on Cybernetics*, 99: 1–12, 2017.

[18] Y. Yuan, X. Zheng, X. Lu, Discovering Diverse Subset for Unsupervised Hyperspectral Band Selection, *IEEE Transactions on Image Processing*, 26(1): 51–64, 2017.

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers, *Foundations & Trends in Machine Learning*, 3(1): 1-122, 2011.

[20] T. Baldacchino, S. R. Anderson, V. Kadirkamanathan, Computational system identification for Bayesian NARMAX modelling, *Automatica*, 49 (9): 2641–2651, 2013.

[21] N. Meinshausen, P. Buhlmann, Stability selection, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4): 417–473, 2010.

[22] W. Pan, A. Sootla, G.B. Stan, Distributed Reconstruction of Nonlinear Networks: An ADMM Approach, *IFAC Proceedings Volumes*, 47(3): 3208-3213, 2014.

[23] W.X. Zheng, On least-squares identification of ARMAX models, *15th IFAC Triennial World Congress*, 391-396, 2002.

[24] K. Chen, Q. Lv, Y. Lu, Y. Dou, Robust regularized extreme learning machine for regression using iteratively reweighted least squares, *Neurocomputing*, 230, 2016.

[25] E. Smucler, V.J. Yohai, Robust and sparse estimators for linear regression models, *Computational Statistics & Data Analysis*, 111: 116-130, 2017.

[26] J. Eckstein, D. Bertsekas, On the DouglasCRachford splitting method and the proximal point algorithm for maximal monotone operators, *Mathematical Programming*, 5: 293–318, 1992.

[27] K.Z. Mao, S.A. Billings, Algorithms for minimal model structure detection in nonlinear dynamic system identification, *International Journal of Control*, 68(2): 311–330, 2003.

[28] C. Willmott, K. Matsuura, Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in assessing average model performance, *Climate Research*, 30: 79-82, 2005.

[29] L. Piroddi, W. Spinelli, An identification algorithm for polynomial NARX models based on simulation error minimization, *International Journal of Control*, 76(17): 1767–1781, 2015.

[30] B.L.R. De Moor, DaISy: Database for the Identification of Systems, Department of Electrical Engineering, ESAT/STADIUS, KU Leuven, Belgium, URL: `http://homes.esat.kuleuven.be/~smc/daisy/daisydata.html`