Least Square Parallel Extreme Learning Machine for Modeling NO$_x$ Emission of a 300MW Circulating Fluidized Bed Boiler

XIA LI$^1$, JIANPING LIU$^1$, AND PEIFENG NIU$^2$

$^1$College of Mathematics and Information Technology, Hebei Normal University of Science and Technology, Qinhuangdao 066004, China
$^2$School of Electrical Engineering, Yanshan University, Qinhuangdao 066004, China

Corresponding author: Jianping Liu (liujianping0408@126.com)

This work was supported in part by the Science Research Foundations of the Hebei Normal University of Science and Technology under Grant 2020YB012, in part by the National Natural Science Foundations of China under Grant 61573306, and in part by the Program for the Top Young Talents of Higher Learning Institutions of Hebei under Grant BJ2017033.

ABSTRACT It is very important to establish an accurate combustion characteristics model of a boiler to reduce NOx emission. In this paper, a novel least square parallel extreme learning machine (LSPELM) is firstly proposed, all of whose weights and thresholds are determined by using least square method twice. Then, LSPELM is applied to 11 classical regression problems to test the validity. The experimental results show that, compared with other methods, LSPELM with a few hidden neurons can achieve good generalization and stability. Next, using Moore-Penrose generalized inverse theory and Woodbury formula, an online learning way of LSPELM (OLSPELM) based on sample increment is also proposed. If the samples of the current time are the same as those of the last time, the weights and thresholds of OLSPELM remain unchanged and are not updated. Only when the input samples of two times are different, can the weights and thresholds of OLSPELM be updated adaptively. Finally, LSPELM and OLSPELM are employed to successfully establish offline and online models of NOx emission concentration for a 300WM circulating fluidized bed boiler. The simulation results also show that LSPELM and OLSPELM have better nonlinear generalization ability and stability performance than some other state-of-the-art models. So, the proposed LSPELM and OLSPELM have good application value.

INDEX TERMS Artificial neural network, least square parallel extreme learning machine (LSPELM), online LSPELM (OLSPELM), Woodbury formula, NOx emission model.

I. INTRODUCTION

With increasing energy crisis and awareness of environmental protection, reducing NOx emission from coal-fired boilers is one of the urgent problems to be solved for power plants. In order to reduce NOx emission, an accurate NOx emission concentration model of a boiler firstly needs to be established. However, combustion process of a boiler involves many chemical reactions, such as turbulence, heat and mass transfer. Furthermore, multiple operating parameters of a boiler are highly coupled each other. So, it is difficult to establish an accurate combustion characteristics model of a boiler by using traditional mechanical modeling methods.

At present, artificial neural network (ANN) has become a very important tool of intelligent modeling. ANN is the abstraction, simplification and simulation of human brain, which is also named as neural network. ANN has good predictive ability, self-learning ability and stability. If we use ANN to establish combustion characteristics model of a boiler, we need not learn complex combustion mechanical knowledge of the boiler. Once we have the history combustion data of a boiler, we can establish the model quickly. So, it is very simple, easy and effective to establish mathematical model by using ANN. ANN has been successfully applied in regression approximation [1], [2], data classification [3], [4], signal processing [5], [6], intelligent control [7]–[9], speech recognition [10]–[12] and other fields.

There are many kinds of ANNs, whose core ideas of weight adjustment mostly come from back propagation (BP)
algorithm and its improvements. BP adopts steepest descent method to obtain network parameters based on iterative calculation. However, it takes a lot of training time to establish a model by using BP algorithm. In addition, BP is easy to fall into local optimum, which leads to low generalization ability. For solving the above problems, Huang et al. proposed a new type of neural network-Extreme Learning Machine (ELM) [13] in 2006.

ELM is a single hidden layer feed-forward neural network. The input weights and hidden layer thresholds are randomly generated, and then the output weights are obtained by solving a system of linear equations. Since the output weights are obtained by one-step calculation, compared with BP algorithm, the training time of ELM is greatly reduced. ELM greatly improves the training speed. The output weights of ELM are uniquely determined, so local minimum problem is also avoided [14]. In addition, it also avoids the complicated process of calculating partial derivatives when BP seeks their network weights. More importantly, ELM has better generalization ability than BP neural network. These superiorities of ELM make it widely used in data classification [15], [16], speech recognition [17], [18], wind speed prediction [19]–[21], image processing [22], [23], smart grid [24], [25], and so on. In order to further improve the generalization ability of ELM, many scholars improved it, and then obtained many excellent scientific research results [26]–[32].

In 2015, Tavares et al. proposed an ELM with parallel layer perception (PELM) [32], which is a parallel neural network. The principle of PELM is simple. The training process of PELM does not need iterative calculation, either. For $n$ inputs and $m$ nonlinear neurons, PELM provides $(n + 1)m$ linear parameters, while ELM only has $m$ linear parameters (one for each hidden neuron)[32]. PELM provides more freedom for the proper adjustment than ELM. In [32], PELM was applied to 12 different regression problems and 6 classification problems. The experiment results showed that, compared with ELM, PELM built a well-conditioned hidden layer and kept under control the norm of the linear weight parameters naturally. So, PELM achieved good generalization performance at a fast speed.

However, since PELM randomly selects the input weights and thresholds of hidden layer in lower network, this may cause training target trap into local minimum. The generalization ability and stability of the network may be affected. How to obtain the suitable parameters of hidden layer becomes very important. To solve this problem, one method is to increase the number of neurons of hidden layer. This method could make randomly assigned parameters achieve the similar effect of the optimal parameters as much as possible. However, this method not only complicates the structure of the network, but also may cause over-fitting of the neural network. So, it is urgent to find another way.

In this study, based on Moore-Penrose (MP) generalized inverse theory, by using least square method, a least square PELM (LSPELM) is firstly proposed. LSPELM needs very few neurons of hidden layer. Furthermore, the input weights and thresholds are obtained by least square solution of linear equations, which improves generalization performance. Then, the validity of LSPELM is verified on 11 standard regression data sets from UCI database. The experimental results show that LSPELM with a small number of hidden layer neurons has better generalization ability and stable performance than other networks on most data sets. Next, when data samples arrive one-by-one or chunk-by-chunk (a block of data) with fixed or varying chunk size, based on LSPELM model, using MP generalized inverse theory and Woodbury formula, an online LSPELM (OLSPELM) based on sample increment is also proposed. If the samples of the current time are the same as those of the last time, the weights and thresholds of OLSPELM remain unchanged and are not updated. Only when the input samples of two times are different, can the weights and thresholds of OLSPELM be updated adaptively. The validity of OLSPELM is also verified. Finally, according to the history combustion data of a 300MW circulating fluidized bed boiler (CFBB) in a thermal power plant, the offline and online models of NOx emission concentration are successfully established by using the proposed LSPELM and OLSPELM, respectively.

Now, the novelty and main contributions of this study are highlighted as follows.

- A novel LSPELM based on the least square method is first proposed, and then an OLSPELM based on sample increment is also proposed.
- Generalization and stability of proposed LSPELM and OLSPELM are compared with those of state-of-the-art models.
- LSPELM and OLSPELM are employ to establish offline and online models of NOx emission concentration for a boiler.

The rest of this paper is arranged as follows. PELM is reviewed in Section II. A LSPELM is proposed in Section III. An online learning algorithm OLSPELM based on sample increment is proposed in Section IV. The offline and online models of NOx emission concentration for a CFBB are established in Section V. The study is concluded in Section VI.

II. REVIEW OF PELM

The network structure of PELM is described in Fig.1. Suppose that there is a data set with $N$ arbitrary distinct samples $(x_l, y_l) \in R^{l \times N} \times R^{1 \times N}, l = 1, 2, \ldots, N$ and that PELM has $m$ hidden layer nodes. $U$ is the matrix of input weights and thresholds of upper network. $V$ is the matrix of input weights and hidden thresholds of lower network. $\beta(\cdot), \gamma(\cdot)$ and $\phi(\cdot)$ are activation functions. Therefore, PELM is mathematically modeled as

$$y_l = \beta \left( \sum_{j=1}^{m} \left[ \gamma(a_{ij})\phi(b_{ij}) \right] \right)$$ (1)
where $u_{ji}$ and $v_{ji}$ are the elements of $U$ and $V$, $x_{il}$ is the $i$th input of the $l$th sample and $y_l$ is the output of the $l$th sample. In PELM, the input-output mapping is made by applying the product of functions [32].

As a particular case of Eq. (1), if the activation functions $\beta(\cdot)$ and $\gamma(\cdot)$ are linear functions, then the network output $y_l$ is calculated by using Eq. (4).

$$y_l = \sum_{j=1}^{m} [a_{jl} \phi(b_{jl})]$$

Replacing Eqs. (2) and (3) in Eq. (4), we can obtain Eq. (5)

$$y_l = \sum_{j=1}^{m} \left( \sum_{i=0}^{n} u_{ji} x_{il} \right) \phi \left( \sum_{i=0}^{n} v_{ji} x_{il} \right)$$

The rearranged elements of $U$ construct a row vector $c$ as follows:

$$c = [u_{10}, \ldots, u_{1n}, u_{20}, \ldots, u_{2n}, u_{m0}, \ldots, u_{mn}]$$

According to MP generalized inverse theory, the minimum norm least square solution of Eq. (8) is

$$c = yH^+$$

where $H^+$ is MP generalized inverse of $H$. After the row vector $c$ is calculated, the output of PELM can be calculated by using Eq. (5).

The learning process of PELM is described as follows:

1. Randomly generate $v_{ji} \in [-1, 1], i = 0, 1, \ldots, n, j = 1, 2, \ldots, m$.
2. Calculate the matrix $H$ according to Eq. (7).
3. Calculate the row vector $c$ according to Eq. (9). So, $u_{ji}$ of Eq. (6) can be obtained, $i = 0, 1, \ldots, n, j = 1, 2, \ldots, m$.

III. A LEAST SQUARE PELM

Through the learning process of PELM, it can be known that the input weights and thresholds of lower layer network are randomly generated. However, randomly generated parameters are not necessarily the suitable parameters. Therefore, in order to obtain the better network parameters of PELM, the input weights and thresholds related to input samples are given by using least square method, and a least square PELM (LSPELM) is proposed. A detailed derivation process of LSPELM is shown below.

A. DETERMINE THE INPUT WEIGHTS AND THRESHOLDS OF LOWER LAYER NETWORK

Suppose that the input weight and threshold matrix $V$ of lower layer network and the output layer threshold $d$ are both...
known. Then, Eq. (10) can be obtained by using Eq. (8).
\[ \beta(cH + dl) = y \]

(10)

Suppose that the activation function \( \beta(\cdot) \) of the output layer is a reversible function. Using Eq. (10), Eqs. (11) and (12) are obtained.
\[
cH + dl = \beta^{-1}(y) \\
cH = \beta^{-1}(y) - dl
\]

(11)

(12)

Then, the output matrix \( H \) of hidden layer can be estimated by using Eq. (13).
\[
\| c\hat{H} - [\beta^{-1}(y) - dl] \|_2 = \| c\hat{H} - \beta^{-1}(y) + dl \|_2 \\
= \min_{\hat{H}} \| c\hat{H} - \beta^{-1}(y) + dl \|_2
\]

(13)

According to MP generalized inverse theory, the minimum norm least square solution of Eq. (13) is
\[
\hat{H} = c^+[\beta^{-1}(y) - dl]
\]

(14)

where \( c^+ \) is MP generalized inverse of vector \( c \).

On the other hand, combining with Eq. (7), the following derivation can be made.
\[
\hat{H} = \begin{bmatrix}
    x_{01}\phi(b_{11}) & x_{02}\phi(b_{12}) & \cdots & x_{0N}\phi(b_{1N}) \\
    \vdots & \ddots & \cdots & \vdots \\
    x_{n1}\phi(b_{m1}) & x_{n2}\phi(b_{m2}) & \cdots & x_{nN}\phi(b_{mN})
\end{bmatrix}
\]

\[
= \begin{bmatrix}
    x_{01} & x_{02} & \cdots & x_{0N} \\
    \vdots & \ddots & \cdots & \vdots \\
    x_{n1} & x_{n2} & \cdots & x_{nN}
\end{bmatrix}
\]

(15)

The “\( \circ \)” operation represents Hadamard product. So, \( H_2 \) can be calculated by using Eq. (16) as follows:
\[
H_2 = \hat{H} \circ H_3
\]

(16)

where
\[
H_3 = \begin{bmatrix}
    1 & 1 & \cdots & 1 \\
    x_{01} & x_{02} & \cdots & x_{0N} \\
    \vdots & \ddots & \cdots & \vdots \\
    x_{n1} & x_{n2} & \cdots & x_{nN}
\end{bmatrix}
\]

(7)
We construct a new matrix $G$ by taking the row 1, the row $n + 2$, the row $2n + 3$, \cdots, and the row $(m - 1)n + m$ of the matrix $H_2$ as follows:

$$
G = \begin{bmatrix}
\phi(b_{11}) & \phi(b_{12}) & \cdots & \phi(b_{1N}) \\
\phi(b_{21}) & \phi(b_{22}) & \cdots & \phi(b_{2N}) \\
\vdots & \vdots & \ddots & \vdots \\
\phi(b_{m1}) & \phi(b_{m2}) & \cdots & \phi(b_{mN})
\end{bmatrix}
$$

$$
= \phi\left(
\begin{bmatrix}
b_{11} & b_{12} & \cdots & b_{1N} \\
b_{21} & b_{22} & \cdots & b_{2N} \\
\vdots & \vdots & \ddots & \vdots \\
b_{m1} & b_{m2} & \cdots & b_{mN}
\end{bmatrix}
\begin{bmatrix}
v_1 \cdot x_1 & v_1 \cdot x_2 & \cdots & v_1 \cdot x_N \\
v_2 \cdot x_1 & v_2 \cdot x_2 & \cdots & v_2 \cdot x_N \\
\vdots & \vdots & \ddots & \vdots \\
v_m \cdot x_1 & v_m \cdot x_2 & \cdots & v_m \cdot x_N
\end{bmatrix}
\right)
$$

$$
= \phi\left(
\begin{bmatrix}
v_1 \\
v_2 \\
v_m
\end{bmatrix}
\begin{bmatrix}
x_1 & x_2 & \cdots & x_N
\end{bmatrix}
\right)
$$

where the definitions of $V$ and $X$ are as follows:

$$
V = \begin{bmatrix}
v_1 \\
v_2 \\
v_m
\end{bmatrix}
= \begin{bmatrix}
v_{10} & v_{11} & \cdots & v_{1n} \\
v_{20} & v_{21} & \cdots & v_{2n} \\
v_{m0} & v_{m1} & \cdots & v_{mn}
\end{bmatrix}
$$

$$
X = \begin{bmatrix}
x_1 & x_2 & \cdots & x_N
\end{bmatrix}
= \begin{bmatrix}
x_{01} & x_{02} & \cdots & x_{0N} \\
x_{11} & x_{12} & \cdots & x_{1N} \\
x_{m1} & x_{m2} & \cdots & x_{mN}
\end{bmatrix}
= \begin{bmatrix}
1 & 1 & \cdots & 1 \\
x_{11} & x_{12} & \cdots & x_{1N} \\
x_{m1} & x_{m2} & \cdots & x_{mN}
\end{bmatrix}
$$

If $\phi$ is a reversible function, then

$$
VX = \phi^{-1}(G)
$$

where $\phi^{-1}$ is the inverse function of $\phi$. The input weight and threshold matrix $V$ of lower network can be estimated by using Eq. (21).

$$
\| \hat{V}X - \phi^{-1}(G) \|_2 = \min_{V} \| VX - \phi^{-1}(G) \|_2
$$

(21)

According to MP generalized inverse theory, the minimum norm least square solution of Eq. (21) is as follows

$$
\hat{V} = \phi^{-1}(G)X^+
$$

(22)

where $X^+$ is MP generalized inverse of $X$.

Suppose that there is a matrix $Q_{m \times 1}$, which can make $\phi^{-1}(G) = Q_{y1 \times N}$. Therefore, Eq. (22) can be rewritten as

$$
\hat{V} = QyX^+
$$

(23)

Since $\phi^{-1}(G)$ is unknown before the training, the elements of matrix $Q$ can be randomly generated between 0 and 1.

**B. DETERMINE THE INPUT WEIGHT AND THRESHOLD MATRIX U OF UPPER LAYER NETWORK AND THE OUTPUT LAYER THRESHOLD d**

After the input weight and threshold matrix $V$ of lower layer network is determined, according to Eq. (17), the output matrix $G$ of hidden layer of lower layer network can be obtained. Then, according to Eq. (15), the matrix $\hat{H}$ is obtained. Therefore, LSPELM can be considered a linear system. Eq. (10) can be rewritten into Eq. (24) as follows:

$$
\beta\left([c \ d]^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix}\right) = y
$$

(24)

Let $\varsigma = [c \ d]$, and then Eq. (24) changes into

$$
\beta\left(\varsigma^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix}\right) = y
$$

(25)

Then, $\varsigma$ can be obtained by using Eq. (26).

$$
\| \beta\left(\varsigma^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix}\right) - y \|_2 = \min_{\varsigma} \| \beta\left(\varsigma^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix}\right) - y \|_2
$$

(26)

Because $\beta(\cdot)$ is a reversible function, the matrix $\hat{\varsigma}$ can be further determined by using Eq. (27).

$$
\| \hat{\varsigma}^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix} - \beta^{-1}(y) \|_2 = \min_{\varsigma} \| \hat{\varsigma}^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix} - \beta^{-1}(y) \|_2
$$

(27)

According to MP generalized inverse theory, the minimum norm least square solution of Eq. (27) is given as follows.

$$
\hat{\varsigma} = \beta^{-1}(y)^{\top}\begin{bmatrix}\hat{H} \\ I\end{bmatrix}
$$

(28)

Therefore, the input weight $c$ of upper network and output layer threshold $d$ can be obtained by using Eq. (29).

$$
\begin{cases}
c = \hat{\varsigma}(1 : mn + m) \\
d = \hat{\varsigma}(mn + m + 1)
\end{cases}
$$

(29)

Finally, the input weight $c$ can be transformed into the matrix $U$ by using Eq. (6).

Through the above derivation, it can be seen that compared with original PELM network, LSPELM network has three following advantages:

1. Although the matrix $Q$ is also randomly generated, the numbers of random numbers is reduced from original $m \times (n + 1)$ to $m$. The reduction of the number of random numbers ensures that LSPELM network will own better stability than original PELM, which is also verified in the following simulation experiments.
(2) The input weight and threshold matrix \( \hat{V} \) of lower network of LSPELM is the minimum norm least square solution of Eq. (20). It can be seen from [33] that the network with smaller weights often has better generalization performance than original network.

(3) The input weight and threshold matrix \( \hat{V} \) of LSPELM is obtained by Eq. (23). The matrix \( \hat{V} \) is related to the input and output variables of samples. Compared with original PELM network, LSPELM network is more closely related to samples. Therefore, LSPELM may have better generalization performance than PELM.

C. Training Process of LSPELM

Suppose that there is a training data \( \{(x_i, y_i) | x_i \in \mathbb{R}^n, y_i \in \mathbb{R}, i = 1, 2, \ldots, N \} \) and that the number of neurons in each hidden layer is \( m \). The detailed training process of LSPELM is described as follows:

Step 1: Randomly generate the matrix \( Q_{m \times 1} \).

Step 2: Analytically calculate the matrix \( \hat{V} \) according to Eq. (23).

Step 3: Calculate the matrix \( G \) according to Eq. (17), and then obtain the matrix \( H_2 \).

Step 4: Obtain the output matrix \( \hat{H} \) by using Eq. (15).

Step 5: Calculate \( \xi \) by using Eq. (28).

Step 6: Obtain the matrix \( U \) of upper layer network and the output layer threshold \( d \) by using Eq. (29).

After all of the weights and thresholds are determined, LSPELM can be used to predict unknown samples. For LSPELM, the training process shows that the input weights and thresholds and the output threshold are obtained by solving linear equations in one step. They are not obtained by stepwise iteration like traditional BP neural network. Therefore, compared with BP neural network, the training speed of LSPELM is very fast. According to [32], PELM has been proved a universal approximator. PELM randomly selects the input weights and thresholds of hidden layer, while our proposed LSPELM obtains the input weights and thresholds by using the least square method. LSPELM does not change the model structure of conventional PELM, but presents an initialization algorithm for PELM. Since PELM is convergent, LSPELM is also convergent.

D. Performance Testing of LSPELM

To verify the performance of LSPELM, we implement the testing on 11 benchmark regression data sets from well-known UCI database. These data sets have been used by many scholars to verify generalization performance of proposed model [13], [34]. Each data set is randomly divided into training set, testing set and verification set, which are shown in Table 1. In order to prevent the under-fitting and over-fitting of the network, the number of hidden layer neurons needs to be selected during the testing. As in [35], the validation set for each data set is used to determine the optimal number of neurons. When the Root Mean Square Error (RMSE) on validation set is minimal, the corresponding number of neurons is then used as the optimal number for prediction.

All the experiments are run in MATLAB 2012a with Windows 7, 2.5 GHZ and 4.00GB RAM. In order to compare LSPELM with other methods, PELM, two-hidden-layer ELM (TELM) [36] and linear regressing (LR) method are selected. TELM also has two hidden layers which are connected in series, but the two hidden layers of PELM and LSPELM are connected in parallel. According to [36], for regression problems, the activation functions of two hidden layers of TELM are recommended to use \( g(x) = (1 - e^{-x})/(1 + e^{-x}) \), and the regression effect is better. For PELM and LSPELM, the activation function \( \phi(x) \) is set as sigmoid function, i.e. \( \phi(x) = 1/(1 + e^{-x}) \). The function \( \beta(x) \) is set as linear function. In [32], the generalization ability of PELM has been compared with ELM in detail. A large number of regression experiments show that PELM has better performance than ELM on most data sets. Therefore, repeated comparison between PELM and ELM is not made in this section.

To prevent different magnitudes of the attributes of data set from affecting regression effect, the input and output attributes are both normalized to \([0.01,1]\). Each experiment is repeated 20 and 50 times in order to prevent the effect of random numbers. As in [35], the maximum neuron number in each hidden layer is set as 50. The optimal number of neurons in each network is shown in Table 2.

| Data set     | Attribute | Number of data | Data division |
|--------------|-----------|----------------|--------------|
|              |           |                | Training | Testing | Validation |
| Abalone      | 8         | 4177           | 2177      | 1000    | 1000       |
| Auto MPG     | 7         | 392            | 200       | 96      | 96         |
| Bank domains | 8         | 8190           | 4500      | 1845    | 1845       |
| Triazines    | 60        | 186            | 100       | 43      | 43         |
| Servo        | 4         | 167            | 80        | 44      | 43         |
| Stocky domain| 9         | 950            | 450       | 250     | 250        |
| Delta-aileron| 5         | 7129           | 4000      | 2065    | 2064       |
| Machine CPU  | 60        | 209            | 100       | 55      | 54         |
| Boston housing| 13       | 506            | 200       | 153     | 153        |
| Cancer       | 32        | 194            | 100       | 47      | 47         |
| Price        | 15        | 159            | 80        | 40      | 39         |

Note: Since TELM, PELM and LSPELM all have two hidden layers, the optimal neuron number for each network is double the number above.

| Data set     | TELM | PELM | LSPELM |
|--------------|------|------|--------|
| Abalone      | 39   | 5    | 2      |
| Auto MPG     | 12   | 4    | 3      |
| Bank domains | 42   | 10   | 4      |
| Triazines    | 15   | 1    | 1      |
| Servo        | 29   | 5    | 3      |
| Stocky domain| 50   | 12   | 16     |
| Delta-aileron| 21   | 1    | 1      |
| Machine CPU  | 2    | 1    | 1      |
| Boston housing| 40  | 2    | 2      |
| Cancer       | 13   | 1    | 1      |
| Price        | 15   | 1    | 1      |
Although these three networks are all double hidden layer feed-forward neural networks, the connection way and training methods are very different. Therefore, the number of neurons for the same data is also very different. The results in Table 2 show that TELM connected in series requires more hidden layer neurons than PELM and LSPELM connected in parallel. The optimal number of neurons for LSPELM is fewer than or equal to that of PELM on almost all data sets, but still much fewer than TELM.

In order to compare the generalization performance of different models, the mean and standard deviation (S.D.) (except LR method) of RMSE, the mean of Mean Absolute Percentage Error (MAPE) and the mean of determination coefficient $R^2$ are recorded in Tables 3-6. The computational formulas of RMSE, MAPE and $R^2$ are shown as Eq. (30)-Eq. (32). The smaller the first three indicators and the larger the fourth indicator, the better the performance of the model. In addition, we also compare the training and testing time of four models. The comparing results are shown in Table 7.

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (y_i - \hat{y}_i)^2}$$  \hspace{1cm} (30)$$

$$\text{MAPE} = \frac{1}{N} \sum_{n=1}^{N} \frac{|y_i - \hat{y}_i|}{y_i} \times 100$$  \hspace{1cm} (31)$$

$$R^2 = \frac{\sum_{n=1}^{N} (\hat{y}_i - \bar{y})^2}{\sum_{n=1}^{N} (y_i - \bar{y})^2}$$  \hspace{1cm} (32)$$

where $N$ represents the number of samples, $y_i$ represents the actual value of output variable, $\hat{y}_i$ represents the predicted value of output variable, and $\bar{y}$ represents the mean of the actual values of output variable.

Table 3 shows the mean and S.D. of RMSE for running 20 times on testing samples. LSPELM obtains the best predicted means on six data sets of Abalone, Auto.MPG, Bank domains, Machine CPU, Boston housing and Cancer. TELM only obtains the best predicted mean on Delta-ailerons. PELM obtains the best predicted means on Servo and Stocks domain. LR method only obtains the best predicted means on Triazines and Price. This simulation results also show that LSPELM has the smallest mean of RMSE on most data sets. So, the generalization ability is optimal, which shows a big advantage of the network. In addition, the S.D. of the RMSE of LSPELM is very close to zero on almost all data sets. Compared with TELM and PELM, LSPELM obtains the smallest S.D. on all benchmark data sets. Therefore, LSPELM is least affected by random numbers and has the best stability. The computational result is consistent with the
TABLE 5. Comparisons of mean and S.D. for running 50 times on testing samples.

| Data set       | TELM Mean | PELM Mean | LSPELM Mean | LR Mean |
|----------------|-----------|-----------|-------------|--------|
| Abalone        | 0.0895    | 0.0294    | 0.0894      | 0.0161 |
| Auto MPG       | 0.0879    | 0.0053    | 0.1015      | 0.0069 |
| Bank domains   | 0.0428    | 0.0006    | 0.0425      | 0.0003 |
| Triazines      | 0.2507    | 0.0225    | 0.3065      | 0.0669 |
| Servo          | 0.1544    | 0.0793    | 0.1633      | 0.0145 |
| Stocks domain  | 0.0404    | 0.0014    | 0.0361      | 0.0020 |
| Delta-aileron  | 0.0385    | 0.0002    | 0.0397      | 0.0011 |
| Machine CPU    | 0.0882    | 0.0252    | 0.0419      | 0.0024 |
| Boston housing | 0.1062    | 0.0560    | 0.1058      | 0.0120 |
| Cancer         | 0.2844    | 0.0166    | 0.3048      | 0.0376 |
| Price          | 0.1064    | 0.0162    | 0.1126      | 0.0141 |

TABLE 6. Comparisons of MAPE and $R^2$ for running 50 times on testing samples.

| Data set       | TELM MAPE | PELM MAPE | LSPELM MAPE | LR MAPE |
|----------------|-----------|-----------|-------------|--------|
| Abalone        | 17.1171   | 17.0743   | 16.7149     | 17.8163|
| Auto MPG       | 15.5840   | 17.7819   | 14.5891     | 21.1305|
| Bank domains   | 50.7847   | 50.0643   | 24.5874     | 45.7725|
| Triazines      | 212.1385  | 201.7776  | 197.2755    | 197.9402|
| Servo          | 89.3908   | 117.6021  | 80.5546     | 146.1619|
| Stocks domain  | 10.4204   | 9.1370    | 11.2341     | 22.2939 |
| Delta-aileron  | 6.1124    | 6.0504    | 6.5368      | 6.3429 |
| Machine CPU    | 112.4557  | 46.6315   | 42.7825     | 44.7754 |
| Boston housing | 27.3450   | 31.7598   | 28.5296     | 31.6490 |
| Cancer         | 189.3816  | 142.3649  | 123.9730    | 130.7970|
| Price          | 38.3033   | 51.5854   | 40.9705     | 44.6432 |

Theoretical analysis in Section III(B), which shows another advantage of LSPELM.

Table 4 shows the means of MAPE and $R^2$ for running 20 experiments on testing samples. As can be seen from the MAPE, LSPELM shows the best predicted results on five data sets of Abalone, Auto.MPG, Bankdomains, Machine CPU and Cancer. TELM obtains the best results on three data sets of Servo, Boston housing and Price. PELM obtains the best results on three data sets of Triazines, Stocks domain and Delta-aileron. LR method does not obtain the best predicted MAPE on any data set. Table 6 shows the means of MAPE and $R^2$ for running 50 experiments. LSPELM shows the best predicted results on seven data sets of Abalone, Auto.MPG, Bank domains, Servo, Triazines, Machine CPU and Cancer. Other methods obtain the best predicted results on a smaller number of data sets than LSPELM. The experiment result of $R^2$ shows that the fitting effect of LSPELM is best on most data sets. The good prediction ability and generalization performance of LSPELM are verified once again.

Taking account of running time in Table 7, since LR method is very simple, the computational time is shortest on most data sets and the testing result is worse. For ELM-based improved models, TELM takes the least amount of time on Triazines. PELM only obtains the best fitting result on Stocks domain. Our proposed LSPELM takes the least amount of time on five data sets of Stock domain, Machine CPU, Boston housing, Cancer and Price. This is attributed to the fact that LSPELM uses a smaller number of neurons on these data sets.

In summary, through the above analysis of testing results, we can know that LSPELM obtains good prediction ability and generalization performance for different data sets with a few hidden layer neurons.
A. Learning Process of OLSPELM

The derivation process of LSPELM in Section III(A) shows that the input weight and threshold matrix $\hat{V}$ of lower network can be calculated by Eq. (23), i.e. $\hat{V} = QyX^T$. Here $Q$ is a $m \times 1$ matrix, $y$ is the output variable of training data, and $X^T$ is generalized inverse of $X$. Suppose that the rank of $X$ is $n + 1$, where $n$ is the attribute dimension of input variables. According to MP generalized inverse theory, $X^+$ can be expressed as

$$X^+ = X^T(XX^T)^{-1}$$  \hspace{1cm} (33)

Therefore, for initial training subset $L_0 = \{(x_i, y_i)\}_{i=1}^{N_0}$, the input weight and threshold matrix $\hat{V}_0$ of lower network can be calculated by using Eq. (34)

$$\hat{V}_0 = Qy_0X_0^T = Qy_0X_0^T(X_0X_0^T)^{-1}$$ \hspace{1cm} (34)

where $X_0 = [x_1, x_2, \ldots, x_{N_0}] = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N_0} \\ \vdots & \vdots & & \vdots \\ x_{N_01} & x_{N_02} & \cdots & x_{N_0N_0} \end{bmatrix}$, and $y_0 = [y_1, y_2, \ldots, y_{N_0}]$. Let $M_0 = X_0X_0^T$. So, Eq. (34) is transformed as

$$\hat{V}_0M_0 = Qy_0X_0^+$$ \hspace{1cm} (35)

According to Eq. (17), the output matrix of hidden layer $G_0$ of lower network is obtained, and then the matrix $\hat{H}_0$ is obtained according to Eq. (15).

$$\hat{H}_0 = \begin{bmatrix} x_{01}\phi(b_{11}) & x_{02}\phi(b_{12}) & \cdots & x_{0N_0}\phi(b_{1N_0}) \\ \vdots & \vdots & & \vdots \\ x_{n1}\phi(b_{11}) & x_{n2}\phi(b_{12}) & \cdots & x_{nN_0}\phi(b_{1N_0}) \end{bmatrix}$$

According to Eq. (28), we can obtain Eq. (36).

$$\tilde{\xi}_0 = \beta^{-1}(y_0)K_0^T(K_0K_0^T)^{-1}$$ \hspace{1cm} (36)

Let $K_0 = \begin{bmatrix} \hat{H}_0 \\ I_0 \end{bmatrix}$. So, Eq. (36) is transformed as

$$\tilde{\xi}_0 = \beta^{-1}(y_0)K_0^T(K_0K_0^T)^{-1}$$ \hspace{1cm} (37)

Therefore, the input weight $c_0$ and output layer threshold $d_0$ are

$$\begin{bmatrix} c_0 \\ d_0 \end{bmatrix} = \tilde{\xi}_0(1 : mn + m)$$ \hspace{1cm} (38)

Let $P_0 = K_0K_0^T$, and then Eq. (37) is transformed as

$$\tilde{\xi}_0P_0 = \beta^{-1}(y_0)K_0^T$$ \hspace{1cm} (39)

When training subset $L_1 = \{(x_i, y_i)\}_{i=N_0+1}^{N_0+N_1}$ with $N_1$ new observation samples arrives, according to Eq. (23), the input weight and threshold matrix $\hat{V}_1$ of lower layer network can be calculated by using Eq. (40).

$$\hat{V}_1 = Q[y_0, y_1][X_0, X_1]^T([X_0, X_1][X_0, X_1]^T)^{-1}$$

$$= Q[y_0, y_1]\begin{bmatrix} X_0^T \\ X_1^T \end{bmatrix}([X_0, X_1]X_0^T)^{-1}$$

$$= (Qy_0X_0^T + Qy_1X_1^T)(X_0X_0^T + X_1X_1^T)^{-1}$$ \hspace{1cm} (40)

where $X_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_{1,N_0+1} & x_{1,N_0+2} & \cdots & x_{1,N_0+N_1} \\ \vdots & \vdots & & \vdots \\ x_{N_0+1,N_0+1} & x_{N_0+1,N_0+2} & \cdots & x_{N_0+N_1,N_0+N_1} \end{bmatrix}$ and $y_1 = [y_{N_0+1}, y_{N_0+2}, \ldots, y_{N_0+N_1}]$. Suppose that the matrix $M_1$ is constructed as Eq. (41).

$$M_1 = X_0X_0^T + X_1X_1^T = M_0 + X_1X_1^T$$ \hspace{1cm} (41)
Substituting Eqs. (35) and (41) into Eq. (40), and we can obtain
\[
\hat{V}_1 = (\hat{V}_0M_0 + Qy_1X_1^T)M_1^{-1} = [\hat{V}_0(M_1 - X_1^TX_1) + Qy_1X_1^T]M_1^{-1} = \hat{V}_0 + (Qy_1 - \hat{V}_0X_1)X_1^TM_1^{-1} (42)
\]
When two learning samples at adjacent times are the same, i.e. \(X_1 = X_0\) and \(y_1 = y_0\), Eq. (42) can be rewritten as Eq. (43).
\[
\hat{V}_1 = \hat{V}_0 + (Qy_0 - \hat{V}_0X_0)X_0^TM_1^{-1} = \hat{V}_0 + (Qy_0X_0^T - \hat{V}_0X_0)X_0^TM_1^{-1} = \hat{V}_0 + (Qy_0X_0^T - \hat{V}_0M_0)X_0^TM_1^{-1} (43)
\]
According to Eq. (35), we can obtain \(Qy_0X_0^T - \hat{V}_0M_0 = 0\). So, \(\hat{V}_1 = \hat{V}_0\). In this situation, the input weights aren’t adjusted. Therefore, only when the learning samples at the adjacent time are different, can the input weights and the thresholds be adjusted adaptively according to the arrival of new samples. This updated way can improve efficiency and save time.

Next, the updated formulas of the input weight and threshold matrix \(c_1\) of upper layer network and output layer threshold \(d_1\) will be derived below. Let \(K_1 = \begin{bmatrix} \hat{H}_1 \\ I_1 \end{bmatrix}\), where \(\hat{H}_1\) is shown in Eq. (44), as shown at the bottom of this page.

According to Eq. (28), and then we can obtain
\[
\hat{\varsigma}_1 = \beta^{-1}(y_0, y_1)(K_0, K_1)^T[(K_0, K_1)(K_0, K_1)^T]^{-1} = [\beta^{-1}(y_0), \beta^{-1}(y_1)](K_0, K_1)^T(K_0K_0^T + K_1K_1^T)^{-1} = [\beta^{-1}(y_0)K_0^T + \beta^{-1}(y_1)K_1^T](K_0K_0^T + K_1K_1^T)^{-1} (45)
\]
Let \(P_1 = K_0K_0^T + K_1K_1^T = P_0 + K_1K_1^T\), according to Eq. (39), and then Eq. (45) is transformed as
\[
\hat{\varsigma}_1 = [\hat{\varsigma}_0P_0 + \beta^{-1}(y_1)K_1^T]P_1^{-1} = [\hat{\varsigma}_0(P_1 - K_1K_1^T) + \beta^{-1}(y_1)K_1^T]P_1^{-1} = \hat{\varsigma}_0 + [\beta^{-1}(y_1) - \hat{\varsigma}_0K_1]K_1^TP_1^{-1} (46)
\]
Then, according to Eq. (29), the updated formulas of \(c_1\) and \(d_1\) are obtained.
\[
\begin{cases}
  c_1 = \hat{\varsigma}_1(1: mn + m) \\
  d_1 = \hat{\varsigma}_1(mn + m + 1) 
\end{cases} (47)
\]
In generally, if the \((k + 1)\)th training subset \(L_{k+1} = \{x_{i,k+1}, y_{i,k+1}\}_{i=1}^{N_{k+1}}\) with \(N_{k+1}\) observed samples arrives, according to Eqs. (41) and (42), the \((k + 1)\)th updated formula of \(\hat{V}_{k+1}\) is
\[
\hat{V}_{k+1} = \hat{V}_k + (Qy_{k+1} - \hat{V}_kX_{k+1})X_{k+1}^TM_{k+1}^{-1} (48)
\]
where
\[
M_{k+1} = M_k + X_{k+1}X_{k+1}^T (49)
\]
Let \(S = M^{-1}\), and then
\[
S_{k+1} = M_{k+1}^{-1} = (M_k + X_{k+1}X_{k+1}^T)^{-1} (50)
\]
According to Woodbury formula, Eq. (50) can be transformed into
\[
S_{k+1} = (M_k + X_{k+1}X_{k+1}^T)^{-1} = M_k^{-1} - M_k^{-1}X_{k+1}(I + X_{k+1}M_k^{-1}X_{k+1})^{-1}X_{k+1}M_k^{-1} = S_k - S_kX_{k+1}(I + X_{k+1}S_kX_{k+1})^{-1}X_{k+1}S_k (51)
\]
Therefore, Eq. (48) can be rewritten as
\[
\hat{V}_{k+1} = \hat{V}_k + (Qy_{k+1} - \hat{V}_kX_{k+1})X_{k+1}^TS_{k+1} (52)
\]
Similar to deducing that when the learning samples at adjacent times are the same, there is \(\hat{V}_1 = \hat{V}_0\), it can also deduce that when the learning samples at adjacent times are the same, there is also \(\hat{V}_{k+1} = \hat{V}_k\). Therefore, the updated formula of \(\hat{V}_{k+1}\) is
\[
\begin{cases}
  \hat{V}_{k+1} = \hat{V}_k & \text{if } y_{k+1} = y_{k+1} \text{ and } X_{k+1} = X_{k+1} \\
  \hat{V}_{k+1} = \hat{V}_k + (Qy_{k+1} - \hat{V}_kX_{k+1})X_{k+1}^TS_{k+1} & \text{Others} 
\end{cases} (53)
\]
In addition, according to Eq. (46), the updated formula \(\hat{\varsigma}_{k+1}\) of the \((k + 1)\)th is
\[
\hat{\varsigma}_{k+1} = \hat{\varsigma}_k + [\beta^{-1}(y_{k+1}) - \hat{\varsigma}_kK_1]K_1^TP_1^{-1} (54)
\]
Let \(R = P^{-1}\), and then
\[
R_{k+1} = P_{k+1}^{-1} = (P_k + K_{k+1}K_{k+1}^T)^{-1} (55)
\]
According to Woodbury formula, Eq. (55) can be transformed into
\[
R_{k+1} = P_k^{-1} - P_k^{-1}K_{k+1}(I + K_{k+1}P_k^{-1}K_{k+1})^{-1}K_{k+1}P_k^{-1} - R_{k+1}K_{k+1}(I + K_{k+1}R_{k+1}K_{k+1})^{-1}K_{k+1}R_{k+1} (56)
\]
Furthermore, Eq. (54) can be written as

\[ \hat{\xi}_{k+1} = \hat{\xi}_k + [\beta^{-1}(y_{k+1}) - \hat{\xi}_k K_{k+1}] K_{k+1}^T R_{k+1} \]  

(57)

Furthermore, \( c_{k+1} \) and \( d_{k+1} \) are obtained as follows.

\[
\begin{align*}
  c_{k+1} &= \hat{\xi}_{k+1} (1 : mn + m) \\
  d_{k+1} &= \hat{\xi}_{k+1} (mn + m + 1)
\end{align*}
\]

(58)

So far, the updated formulas of all the weights and thresholds of OLSPELM have been deduced with the arrival of new samples. From the above derivation process, it can be analyzed that OLSPELM can adaptively update all the weights and thresholds of the network based on sample increment. OLSPELM is an online form of LSPELM. Since LSPELM is convergent, OLSPELM is also a universal approximator.

B. TRAINING STEPS OF OLSPELM

According to the learning process of OLSPELM described in Section IV(A), the training steps of OLSPELM can be mainly summarized two processes, which are shown in detail as follows.

**Step 1:** Initialization phase. Suppose that initial training sample set is \( L_0 = \{(x_i, y_i)\}_{i=1}^{N_0} \).

1. Randomly generate a \( m \times 1 \) matrix \( Q \).
2. Calculate the input weight and threshold matrix \( \hat{V}_0 \) by using Eq. (34).
3. Obtain the output matrix \( G_0 \) of lower network according to Eq. (17), and then obtain \( \hat{H}_0 \) according to Eq. (15).
4. Combine \( \hat{H}_0 \) and \( I_0 \) into the matrix \( K_0 \).
5. Calculate the input weight \( c_0 \) and output layer threshold \( d_0 \) according to Eqs. (37) and (38).
6. Set the parameter \( k = 0 \).

**Step 2:** Updating phase of the weights and thresholds.

Suppose that the training sample set of the \((k+1)\)th is \( L_{k+1} = \{(x_i, y_i)\}_{i=1}^{N_{k+1}} \) and \( N_{k+1} = \sum_{j=0}^{k+1} N_j \).

1. Update the input weight and threshold matrix \( \hat{V}_{k+1} \) according to Eqs. (51) and (53).
2. Calculate the hidden output matrix \( G_{k+1} \) according to Eq. (17), and then obtain \( \hat{H}_{k+1} \). Combine \( \hat{H}_{k+1} \) and \( I_{k+1} \) into the matrix \( K_{k+1} \).
3. Update the matrix \( \hat{\xi}_{k+1} \) according to Eqs. (56) and (57).
4. Split the matrix \( \hat{\xi}_{k+1} \) into \( c_{k+1} \) and \( d_{k+1} \) according to Eq. (58).
5. Set \( k = k + 1 \), and then return to Step 2 until the end of the learning.

C. PERFORMANCE TESTING OF OLSPELM

To verify the performance of OLSPELM, we present some simulation studies on two large datasets of Condition Based Maintenance of Naval Propulsion Plants (CBM) [37] and Elevators [38]. The data sets with their corresponding partitions and the optimal number of neurons are shown in Table 8.

LSPELM directly learns all the training samples once, while OLSPELM divides the training set into two parts. One part is initial training samples, which are learned once. The other part is sequence learning samples, which are mainly used to simulate the generation of new data. OLSPELM can learn data one-by-one or chunk-by-chunk (a block of data) with fixed or varying chunk size. Through a large number of simulation experiments, it is found that different online learning modes have less influence on the model, so hundred-by-hundred learning mode is selected in this section. In addition, in order to explain the influence of the number of initial samples on OLSPELM, the number of initial training samples is set as 500 and 1000, respectively. In order to fairly verify the performance of online models, OLSPELM is compared with famous OSELM [39] and online PELM (OPELM). Every model runs 20 times. The computational results are given in Tables 9-10.

From the value of every performance index in Table 9, regardless of the size of initial samples, OLSPELM shows the best performance on the training and testing data sets. From Table 10, since OSELM is very simple, the computational time is shortest in three online models. However, the computational results of training and testing are worst. When the number of initial training data is 500, OLSPELM shows the best performance on training and testing data sets except the running time. When the number of initial training data is 1000, OLSPELM and OPELM present similar predicted results on training data. However, for testing data, compared with OPELM, OLSPELM presents much better predicted results except the running time. Therefore, OLSPELM is superior to OSELM and OPELM.

V. MODEL NOx EMISSION CONCENTRATION FOR A CFBB USING LSPELM AND OLSPELM

NOx is an important environmental pollutant from combustion emissions of a CFBB. With the deterioration of global environment and people’s increasing emphasis on energy conservation and environmental protection, reducing NOx emission from power station boilers becomes an urgent matter. Combustion optimization technology is one of the effective measures to reduce NOx emission. In order to achieve...
combustion optimization of a boiler, it is firstly necessary to establish a combustion characteristic model of NOx emission concentration.

The modeling methods of NOx emission concentration are mainly divided into two categories. The first category is mechanistic modeling method based on detailed combustion processes. However, combustion process of a boiler is a very complicated physical and chemical process. The parameters such as coal feeder rate, air velocity, air temperature, and oxygen content of the flue gas in the combustion process are characterized by strong coupling and nonlinearity. So, it is difficult to establish an accurate combustion characteristics model. The second category is artificial intelligence technology, which is based on historical combustion data of a CFBB collected from Distributed Control System (DCS). We need not learn the complex combustion mechanism knowledge of a CFBB to model the combustion characteristics. Once we have history data of boiler combustion, we can establish the combustion characteristics model quickly. So, this method is simple and efficient. In this study, LSPELM and OLSPELM are used to establish the offline and online models of NOx emission for a CFBB.

**A. EXPERIMENTAL DATA AND THE SELECTION OF OPERATIONAL PARAMETERS**

Partial combustion history data are collected from a 300 MW CFBB of a power plant under normal operating conditions. Due to the limitations of field equipment and sampling techniques, 600 samples with 25 attribute parameters related to NOx emission concentration are collected, which are shown in Table 11. The sampling interval between two samples is 30 seconds. In order to more easily see the distribution characteristics and fluctuations of operating parameters, whose boxplots are given in Fig. 2.
TABLE 11. Operating parameters of a 300MW CFBB.

| Data | Load (%) | CFR (th⁻¹) | PATB (°C) | SAV (kNm⁻³h⁻¹) | OC (%) | PAV (kNm⁻³h⁻¹) | EGT (°C) |
|------|----------|------------|-----------|----------------|--------|----------------|---------|
| A    |          |            |            |                |        |                |         |
| B    | 64.524   | 60.521     | 36.58      | 37.129         | 36.233 | 252.499        | 250.19  |
| C    |          |            |            |                |        |                |         |
| D    |          |            |            |                |        |                |         |

These 25 attribute parameters conclude one boiler load (%), four coal feeder rates (CFR, th⁻¹), two primary air temperatures of burner inlet (PATB, °C), two secondary air velocities (SAV, kNm⁻³h⁻¹), one flue gas oxygen content (OC, %), two primary air velocities (PAV, kNm⁻³h⁻¹), one exhaust gas temperature (EGT, °C), two unburned carbons in flue dust (UCFD, %), two primary air temperatures of fan inlet (PATF, °C), and four outlet temperatures of slag cooler (OTSC, °C), two currents of powder feeding machine (CPFM, A), and two secondary air temperatures of air preheater outlet (SAT, °C).

To verify that the 25 attribute parameters are statistically related to the NOx emission concentration, Spearman test is performed in the SPSS 19.0 software. The significance level is set as 0.05. Since the overall distribution of the data is unknown, Pearson test is not used. The detailed testing result is shown in Table 12 below.

The P value in Table 12 indicates that there is a statistically linear correlation between NOx emission concentration and the 24 attribute parameters except PA V(B). Although there is no linear correlation between NOx emission concentration and PA V(B), it is known from the actual situation that the PA V is nonlinearly related to NOx emissions. In addition, although the 25 attribute parameters may be related and coupled each other, LSPELM can directly deal with multivariate related problems, and automatically reflect the correlation degree of each parameter to the weight of the network through the training process. Therefore, the 25 attribute parameters are used as input parameters of LSPELM, and the NOx emission concentration are used as output parameter to establish combustion characteristic model of the CFBB. In order to eliminate the influence of the magnitude between different attributes, input and output attributes are normalized to [0.01,1]. After the results are calculated, these attributes are anti-normalized.

B. OFFLINE ESTABLISH THE MODEL OF NOx EMISSION CONCENTRATION BY USING LSPELM

In this section, the proposed LSPELM are used to establish offline model of NOx emission concentration of the boiler. The collected 600 samples are randomly divided into three parts. 300 samples are used as training set to establish the model, 150 samples are used as verification set to select the suitable number of neurons, and the remaining 150 samples are used as testing sets to verify generalization performance. As in Section III(D), in order to prevent under-fitting and over-fitting of neural network, appropriate number of neurons needs to be set by the principle of minimizing the RMSE on verification set. For ELM and TELM that require relatively many neurons, the maximum neuron number in each layer is set as 50 according to experience. According to Section III(D), PELM and LSPELM require fewer neurons, so the maximum neuron number in each hidden layer is set as 5. The activation functions of the hidden layers are the same as those of Section III(D). Similarly, to prevent the effects of random initialization on the networks, each experiment is repeated 20 times. Fig. 3 shows the curves of predicted precision with increasing number of neurons in hidden layer on validation set.

According to the principle of error minimization, the numbers of neurons of ELM and TELM are set as 27 and 39, respectively. In addition, it can be clearly seen from Fig. 3 that after the first neuron, the RMSEs of PELM and LSPELM on verification data do not decrease, but keep rising. So, the over-fitting phenomenon of neural networks occurs. In order to prevent over-fitting, a fewer neurons can be used, which is also reduce the complexity of the network. So, for PELM and LSPELM, the neuron number in each hidden layer is selected as 1. After the number of neurons in hidden layer is selected,
the model of NOx emission concentration can be established according to Section III(C).

Fig. 4 shows predicted NOx emission concentration by using LSPELM model. Table 13 gives the comparison of predicted precision of NOx emission concentration. As seen from Table 13, although LSPELM uses a few neurons, the mean and S.D. of RMSE, and the mean of MAPE are smallest in four models. The maximum MAPE is 1.0523% by calculation, so the maximum value of MAPE is smaller than the mean of MAPE of other models. From the three
testing indicators, we know that the generalization ability of LSPELM is much better than original PELM and other models. It can be seen that the predicted effect of LSPELM is best in four models. Fig. 4 and Table 13 show that LSPELM not only has higher prediction precision and generalization ability, but also has better stability.

C. ONLINE ESTABLISH THE MODEL OF NOx EMISSION CONCENTRATION BY USING OLSPELM

The established model in Section V(B) is a static description of combustion characteristics of a boiler, so LSPELM is an offline modeling method. However, the combustion process of a boiler is very complex and will be affected by many factors, so the operating conditions of a boiler will constantly

---

**FIGURE 3.** Variation curves of predicted precision for four models.

**FIGURE 4.** Predicted NOx emission by using LSPELM model.

**FIGURE 5.** Box comparison of RMSE for predicted NOx emission.

**FIGURE 6.** Flowchart of OLSPELM for online modeling NOx emission.
change. If new conditions have new features during dynamic process, the offline established model needs to abandon the results of the last learning. This leads to wasting a lot of time and space resources. Therefore, in this situation, online modeling can track the combustion dynamics of a boiler. So, the proposed OLSPELM is used to establish an online model of NOx emission concentration.

The data classification is the same as the offline modeling. In order to explain the influence of the number of initial samples, the number of initial training samples is set as 100, 200 and 250, respectively. Since the number of sequence learning samples is small, one-by-one learning mode is selected in this section. In addition, all model parameters are set the same as Sections V(B). For a more vivid observation, the modeling flowchart of OLSPELM is given in Fig. 6. The simulation results are shown in Table 14 and Figs. 7-9.

Table 14 shows that no matter how many initial samples, every performance index of OLSPELM is smallest on the training and testing data sets. Therefore, OLSPELM is significantly superior to OSELM and OPELM. In particular, OLSPELM has more obvious advantage on the S.D.. The S.D. of OLSPELM is two orders of magnitude smaller than that of OPELM and one order of magnitude smaller than that of OSELM. This indicates that the stability of OLSPELM is much better than that of OPELM and OSELM. In addition, when the number of initial training samples increases from 100 to 200, all three indexes of OLSPELM are better on the testing set. When the number of initial samples increases from 200 to 250, except the slight increasing of S.D., the other two indicators are also getting better and better.

Table 14. Comparison of predicted precision of online models.

| Initial training sample | Data       | Performance index | OSELM    | OPELM    | OLSPELM   |
|-------------------------|------------|-------------------|----------|----------|-----------|
|                         |            | RMSE Mean         | 1.3876   | 1.5252   | 1.3393    |
|                         |            | S.D.              | 0.0738   | 0.1673   | 0.0069    |
|                         |            | MAPE(%) Mean      | 0.7063   | 0.7725   | 0.6662    |
|                         |            | RMSE Mean         | 2.5735   | 2.6102   | 1.9206    |
|                         |            | S.D.              | 0.6491   | 1.0111   | 0.0758    |
|                         |            | MAPE(%) Mean      | 1.3617   | 1.3855   | 0.9611    |
| 100                     | Training   | RMSE Mean         | 1.3744   | 1.6874   | 1.3424    |
|                         |            | S.D.              | 0.0626   | 0.2534   | 0.0049    |
|                         |            | MAPE(%) Mean      | 0.7004   | 0.8469   | 0.6690    |
|                         | Testing    | RMSE Mean         | 2.4634   | 2.9942   | 1.8856    |
|                         |            | S.D.              | 0.6217   | 1.6087   | 0.0494    |
|                         |            | MAPE(%) Mean      | 1.3032   | 1.5904   | 0.9380    |
| 200                     | Training   | RMSE Mean         | 1.3876   | 1.6931   | 1.3406    |
|                         |            | S.D.              | 0.0738   | 0.4584   | 0.0059    |
|                         |            | MAPE(%) Mean      | 0.7063   | 0.8279   | 0.6674    |
|                         | Testing    | RMSE Mean         | 2.5735   | 4.8426   | 1.8608    |
|                         |            | S.D.              | 0.6491   | 4.8339   | 0.0813    |
|                         |            | MAPE(%) Mean      | 1.3617   | 2.4625   | 0.9282    |

FIGURE 7. Predicted NOx emission concentration on initial training samples.

FIGURE 8. Predicted NOx emission concentration on sequence learning samples.
learning samples. Fig. 9 gives the predicted result on 300 testing samples. The maximum value of relative errors is 1.5978% by calculation on testing samples. These three figures show that OLSPELM has small prediction error and good prediction effect.

In a word, OLSPELM not only has better identification performance than OSELM and OPELM, but also has stronger generalization ability and better stability than OSELM and OPELM. Therefore, OLSPELM is very suitable to describe the dynamic combustion characteristics of the CFBB.

VI. CONCLUSION

In this paper, a novel LSPELM is firstly given by the minimum norm least square solution of linear equations. Then, 11 well-known regression data sets from UCI are used to verify the performance of LSPELM. The experimental results show that LSPELM with a fewer hidden layer neurons has better generalization ability and stability than PELM and other methods. Furthermore, using MP generalized inverse theory and Woodbury formula, OLSPELM based on sample increment is also given. Next, according to the combustion history data of a 300MW CFBB from a thermal power plant, an offline model of NOx emission concentration is established by using LSPELM. The simulation results show that LSPELM can better represent the functional relationship between NOx emission concentration and boiler-related operating parameters than PELM, ELM and other model. Finally, an online model of NOx emission concentration by using OLSPELM is also established. The simulation results also show that OLSPELM has better nonlinear identification ability, generalization ability and stability performance than OSELM and OPELM. OLSPELM can effectively identify the complex dynamic combustion process of the boiler. In short, the proposed LSPELM and OLSPELM are effective and useful modeling tools. The proposed methods can be expanded to solve the classification problems, the image processing problems and speech recognition problems.

The input weight and threshold matrix of LSPELM is related to the samples. So, compared with randomly initialized PELM network, LSPELM have better generalization performance than original PELM. on the other hand, the reduction of the number of random numbers ensures that LSPELM owns better stability than PELM. However, due to the limitations of field equipment and sampling technology, the size of history combustion data is small. How to collect a larger amount of samples that can represent the combustion operation of the boiler needs further research. In addition, we more focus on incremental learning of OLSPELM, and decremental learning is not considered in the model. In order to further improve the performance of the model, more attention needs to be paid to consider some mechanisms to lower the importance of the old samples. This can be considered in the near future work.

REFERENCES

[1] Z. He, Y. Chen, Z. Shang, C. Li, L. Li, and M. Xu, “A novel wind speed forecasting model based on moving window and multi-objective particle swarm optimization algorithm,” Appl. Math. Model., vol. 76, pp. 717–740, Dec. 2019.
[2] J. Szoplik, “Forecasting of natural gas consumption with artificial neural networks,” Energy, vol. 85, pp. 208–220, Jun. 2015.
[3] B. A. Garro, R. Vázquez, and R. A. García, “Classification of DNA microarays using artificial neural networks and ABC algorithm,” Appl. Soft Comput., vol. 38, pp. 548–560, Jan. 2016.
[4] N. Stahl, G. Mathiaison, G. Falkman, and A. Karlsson, “Using recurrent neural networks with attention for detecting problematic slab shapes in steel rolling,” Appl. Math. Model., vol. 70, pp. 365–377, Jun. 2019.
[5] A. K. Azad, L. Wang, N. Guo, H.-Y. Tam, and C. Lu, “Signal processing using artificial neural network for BOTDA sensor system,” Opt. Express, vol. 24, no. 6, pp. 67–69, May 2016.
[6] S. Xu, K. Liu, and X. Li, “A fuzzy process neural network model and its application in process signal classification,” Neurocomputing, vol. 335, pp. 1–8, Mar. 2019.
[7] K. Bouhoun, K. Yazid, M. S. Boucheriti, and A. Chérifi, “Hybrid control of the three phase induction machine using artificial neural networks and fuzzy logic,” Appl. Soft Comput., vol. 55, pp. 289–301, Jun. 2017.
[8] K. Venkatesan and U. Ramachandraiah, “Climate responsive cooling control using artificial neural networks,” J. Building Eng., vol. 19, pp. 191–204, Sep. 2018.
[9] L. Nie, J. Guan, C. Lu, H. Zheng, and Z. Yin, “Longitudinal speed control of autonomous vehicle based on a self-adaptive PID of radial basis function neural network,” IET Intell. Transp. Syst., vol. 12, no. 6, pp. 485–494, Aug. 2018.
[10] G. Dede and M. H. Sazli, “Speech recognition with artificial neural networks,” Digit. Signal Process., vol. 20, no. 3, pp. 763–768, May 2010.
[11] S. R. Shahamiri and S. B. Salim, “Real-time frequency-based noise-robust automatic speech recognition using multi-nets artificial neural networks: A multi-views multi-learners approach,” Neurocomputing, vol. 129, pp. 199–207, Apr. 2014.
[12] S. M. Siniscalchi, T. Svendsen, and C.-H. Lee, “An artificial neural network approach to automatic speech processing,” Neurocomputing, vol. 140, pp. 326–338, Sep. 2014.
[13] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications,” Neurocomputing, vol. 70, nos. 1–3, pp. 489–501, Dec. 2006.
[14] G.-B. Huang and L. Chen, “Enhanced random search based incremental extreme learning machine,” Neurocomputing, vol. 71, nos. 16–18, pp. 3460–3468, Oct. 2008.
[15] A. Bequè and S. Lessmann, “Extreme learning machines for credit scoring: An empirical evaluation,” Expert Syst. Appl., vol. 86, pp. 42–53, Nov. 2017.
[16] P. Mohapatra, S. Chakravarty, and P. D. Dash, “An improved cuckoo search based extreme learning machine for medical data classification,” Swarm Evol. Comput., vol. 24, pp. 25–40, Oct. 2015.
[17] Z.-T. Liu, M. Wu, W.-H. Cao, J.-W. Mao, J.-P. Xu, and G.-Z. Tan, “Speech emotion recognition based on feature selection and extreme learning machine decision tree,” Neurocomputing, vol. 273, pp. 271–280, Jan. 2018.

[18] Y. Lan, Z. Hu, Y. C. Soh, and G.-B. Huang, “An extreme learning machine approach for speaker recognition,” Neural Comput. Appl., vol. 22, nos. 3–4, pp. 417–425, Mar. 2013.

[19] Z. Tian, Y. Ren, and G. Wang, “Short-term wind speed prediction based on improved PSO algorithm optimized EM-ELM,” Energy Sour. A, Recovery, Utilization, Environ. Effects, vol. 41, no. 1, pp. 26–46, Jan. 2019.

[20] Z. D. Tian, S. J. Li, and Y. H. Wang, “A approach using ensemble empirical mode decomposition-permutation entropy and regularized extreme learning machine for short-term wind power,” Wind Energy, vol. 23, no. 2, pp. 164–171, Feb. 2020.

[21] Z. Tian, G. Wang, S. Li, Y. Wang, and X. Wang, “Artificial bee colony algorithm—optimized error minimized extreme learning machine and its application in short-term wind speed prediction,” Wind Eng., vol. 43, no. 3, pp. 263–276, Jun. 2019.

[22] S. Suresh, R. Venkatesh Babu, and H. J. Kim, “No-reference image quality assessment using modified extreme learning machine classifier,” Appl. Soft Comput., vol. 9, no. 2, pp. 541–552, Mar. 2009.

[23] S. Wang, C. Deng, W. Lin, G.-B. Huang, and B. Zhao, “NMF-based image quality assessment using extreme learning machine,” IEEE Trans. Cybern., vol. 47, no. 1, pp. 232–243, Jan. 2017.

[24] S. Hassan, A. Khosravi, J. Jaafar, and M. A. Khanesar, “A systematic design of interval type-2 fuzzy logic system using extreme learning machine for electricity load demand forecasting,” Int. J. Electr. Power Energy Syst., vol. 82, no. 1, pp. 1–10, Nov. 2016.

[25] Y. Xu, Z. Y. Dong, Z. Xu, K. Meng, and K. P. Wong, “An intelligent dynamic security assessment framework for power systems with wind power,” IEEE Trans. Ind. Informat., vol. 8, no. 4, pp. 995–1003, Nov. 2012.

[26] J. Cao, Z. Lin, and G.-B. Huang, “Self-adaptive evolutionary extreme learning machine,” Neural Process. Lett., vol. 36, no. 3, pp. 285–305, Dec. 2012.

[27] F. Han, H.-F. Yao, and Q.-H. Ling, “An improved evolutionary extreme learning machine based on particle swarm optimization,” Neurocomputing, vol. 116, pp. 87–93, Sep. 2013.

[28] Y. Xu, M. Zhang, L. Ye, Q. Zhu, Z. Geng, Y.-L. He, and Y. Han, “A novel prediction intervals method integrating an error & self-feedback extreme learning machine with particle swarm optimization for energy consumption robust prediction,” Energy, vol. 164, pp. 137–146, Dec. 2018.

[29] Z. Tian, G. Wang, Y. Ren, S. Li, and Y. Wang, “An adaptive online sequential extreme learning machine for short-term wind speed prediction based on improved artificial bee colony algorithm,” Neural Netw. World, vol. 28, no. 3, pp. 191–212, 2018.

[30] Z. D. Tian, Y. Ren, and G. Wang, “Short-term wind power prediction based on empirical mode decomposition and improved extreme learning machine,” J. Electr. Eng. Technol., vol. 13, no. 5, pp. 1841–1851, Sep. 2018.

[31] Z. Tian, S. Li, Y. Wang, and X. Wang, “Network traffic prediction method based on improved ABC algorithm optimized EM-ELM,” J. China Universities Posts Telecommun., vol. 25, no. 3, pp. 33–44, Jun. 2018.

[32] L. D. Tavares, R. R. Saldanha, and D. A. G. Vieira, “Extreme learning machine with parallel layer perceptrons,” Neurocomputing, vol. 166, pp. 164–171, Oct. 2015.

[33] P. L. Bartlett, “The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network,” IEEE Trans. Inf. Theory, vol. 44, no. 2, pp. 525–536, Mar. 1998.

[34] G. Li, P. Niu, X. Duan, and X. Zhang, “Fast learning network: A novel artificial neural network with a fast learning speed,” Neural Comput. Appl., vol. 24, nos. 7–8, pp. 1683–1695, Jun. 2014.

[35] Y. Ma, P. Niu, X. Zhang, and G. Li, “Research and application of quantum-inspired double parallel feed-forward neural network,” Knowl.-Based Syst., vol. 136, pp. 140–149, Nov. 2017.

[36] B. Y. Qu, B. F. Lang, J. J. Liang, A. K. Qin, and O. D. Crisalle, “Two-hidden-layer extreme learning machine for regression and classification,” Neurocomputing, vol. 175, pp. 826–834, Jan. 2016.

[37] A. Coraddu, L. Oneto, A. Ghio, S. Savio, D. Anguita, and M. Figari, “Machine learning approaches for improving condition-based maintenance of naval propulsion plants,” Proc. Inst. Mech. Eng. M: J. Eng. Maritime Environ., vol. 230, no. 1, pp. 136–153, Feb. 2016.

[38] L. Torgo, Regression Datasets. Accessed: Jun. 1999. [Online]. Available: http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html

[39] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, “A fast and accurate online sequential learning algorithm for feedforward networks,” IEEE Trans. Neural Netw., vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

XIA LI received the M.S. degree in computational mathematics and the Ph.D. degree in control science and engineering from Yanshan University, China, in 2005 and 2019, respectively. She is a Lecturer with the Hebei Normal University of Science and Technology, China. Her research interests include neural networks, evolutionary algorithms, and combustion optimizing.

JIANPING LIU received the M.S. degree in computational mathematics and the Ph.D. degree in computer science and technology from Yanshan University, China, in 2009 and 2019, respectively. He is an Associate Professor with the Hebei Normal University of Science and Technology, China. His research interests include neural networks and intelligent algorithms.

PEIFENG NIU received the Ph.D. degree from Northeast University, China. He is a Professor with the School of Electrical Engineering, Yanshan University, China. His main research interests include complex industrial systems, intelligent modeling, and intelligent control.