Parsimonious Extreme Learning Machine Using Recursive Orthogonal Least Squares

Ning Wang, *Member, IEEE*, Meng Joo Er, *Senior Member, IEEE*, and Min Han, *Senior Member, IEEE*

***Abstract*— Novel constructive and destructive parsimonious extreme learning machines (CP- and DP-ELM) are proposed in this paper. By virtue of the proposed ELMs, parsimonious structure and excellent generalization of multiinput-multioutput single hidden-layer feedforward networks (SLFNs) are obtained. The proposed ELMs are developed by innovative decomposition of the recursive orthogonal least squares procedure into sequential partial orthogonalization (SPO). The salient features of the proposed approaches are as follows: 1) Initial hidden nodes are randomly generated by the ELM methodology and recursively orthogonalized into an upper triangular matrix with dramatic reduction in matrix size; 2) the constructive SPO in the CP-ELM focuses on the partial matrix with the subcolumn of the selected regressor including nonzeros as the first column while the destructive SPO in the DP-ELM operates on the partial matrix including elements determined by the removed regressor; 3) termination criteria for CP- and DP-ELM are simplified by the additional residual error reduction method; and 4) the output weights of the SLFN need not be solved in the model selection procedure and is derived from the final upper triangular equation by backward substitution. Both single- and multi-output real-world regression data sets are used to verify the effectiveness and superiority of the CP- and DP-ELM in terms of parsimonious architecture and generalization accuracy. Innovative applications to nonlinear time-series modeling demonstrate superior identification results.**

***Index Terms*— Extreme learning machine (ELM), parsimonious model selection, recursive orthogonal least squares (ROLS), sequential partial orthogonalization (SPO), single hidden-layer feedforward network (SLFN).**

I. Introduction

RECENTLY, the extreme learning machine (ELM) for single hidden-layer feedforward networks (SLFNs) has attracted much attention from researchers since Huang *et al.* [1] proposed the seminal work of ELM. Evidently, the ELM is an extremely fast batch learning algorithm and can provide good generalization performance [2]–[4]. Online sequential ELM (OS-ELM) [5] is preferred over the original ELM in real-time applications since it can learn the training data chunk by chunk and it discards the data, which has been used for the training process. In this context, finding an optimal model for randomly generated hidden nodes has become a crucial issue in leveraging on the ELM capabilities. However, the ELM of [1] and OS-ELM of [5] cannot provide any effective solutions for architectural design of SLFNs.

Investigations on the ELM for optimal structure have been proposed in destructive and constructive approaches, which have been effectively implemented in fuzzy neural networks [6]–[8]. Rong *et al.* [9] have presented a pruned ELM (P-ELM), which starts with a large network and removes hidden nodes having low relevance to the output. Miche *et al.* [10] have proposed an optimally pruned ELM (OP-ELM) using the multiresponse sparse regression (MRSR) of [11] and leave-one-out validation for the pruning strategy. Evidently, these destructive paradigms face a common difficulty that the initial large structure could inevitably increase the computational burden. The incremental ELM of [2] and its variants [12], [13] revolve around adding hidden nodes one-by-one to the hidden layer and updating output weights incrementally. However, those algorithms cannot realize an optimal structure automatically and output weights of the SLFN need to be updated recursively. Feng *et al.* [14] have proposed the error minimized ELM, which can add random hidden nodes one-by-one or group-by-group. Unfortunately, the nodes added to the hidden layer are randomly generated and might deteriorate the performance with increased number of hidden nodes since no generalization measure is guaranteed. Furthermore, the resulting network structure is similar to the ELM if high prediction performance is required. Recently, Lan *et al.* [15] and Wang *et al.* [16] have presented CS-ELM and CM-ELM for single- and multioutput regression, respectively, whereby hidden nodes are selected by the MRSR and unbiased risk-estimation-based criterion C_p. However, the hidden node selection is carried out on normalized regressors.

It should be emphasized that all previous constructive and destructive methods for model selection directly work on the hidden matrix $\mathbf{H} \in \mathbb{R}^{K\times L}$, where K and L are the numbers of training data and hidden nodes, respectively. Generally, K is significantly larger than L, and thereby dominating the matrix dimension and computational complexity in model selection. It turns out that direct model selection on $\mathbf{H}$ is likely to cause inefficient selection of regressors due to rank deficiency. Nevertheless, in each selection iteration, output weights are required to be recursively or incrementally

Manuscript received April 10, 2013; revised December 16, 2013; accepted December 18, 2013. Date of publication January 9, 2014; date of current version September 15, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 51009017, Grant 51379002, and Grant 61074096, in part by the Applied Basic Research Funds from the Ministry of Transport of China under Grant 2012-329-225-060, in part by the China Post-Doctoral Science Foundation under Grant 2012M520629, in part by the Program for Liaoning Excellent Talents in University under Grant LJQ2013055, and in part by the Fundamental Research Funds for the Central Universities of China under Grant 2009QN025, Grant 2011JC002, and Grant 3132013025.

N. Wang is with Marine Engineering College, Dalian Maritime University, Dalian 116026, China (e-mail: n.wang.dmu.cn@gmail.com).

M. J. Er is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: emjer@ntu.edu.sg).

M. Han is with the Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China (e-mail: minhan@dlut.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2013.2296048

solved from currently selected hidden nodes. Recently, Castaño *et al.* [17] proposed a robust and pruned ELM approach based on principal component analysis (PCA), termed PCA-ELM, whereby the PCA method is used to select the hidden nodes from input features while corresponding input weights are deterministically defined as principal components rather than random ones. Note that the deterministic approach to hidden node selection inevitably results in poor accuracy especially for noisy data.

The orthogonal decomposition is a robust approach to the least squares (LS) problem and is then applied to compute the output weights of RBF networks. The Gram–Schmidt and Givens rotation algorithms are main techniques for implementing the orthogonal LS (OLS) [18]. Stemming from the Gram–Schmidt-based OLS method, Chen *et al.* [19], [20] proposed the OLS-based forward selection of RBF centers by maximizing the error reduction ratio (ERR) of each selected RBF center at each stage. This mechanism enables selection of significant hidden nodes and avoids ill-conditioning regression problems. With the aim of reducing complexity, the fast OLS is implemented by directly updating scalar inner products instead of vectors [21]. Nevertheless, within these algorithms and other variants [22]–[26], the orthogonal matrix of the same dimension as the RBF hidden output matrix need to be explicitly calculated for ERR checking, and thereby leading to high computational burden if large training data sets are considered. In addition, the Gram–Schmidt-based OLS method cannot be easily extended to online cases.

The Givens-rotation-based OLS method of [27] was proposed for RBF network pruning by sequentially removing the regressor with minimal effects on the network output error. Note that the model selection relies on the transformed upper triangular matrix where the orthogonal matrix need not be explicitly stored. The recursive OLS (ROLS) [28] is developed to update the Cholesky factor of the information matrix derived from Givens transformation without the need of forming it. In this context, higher precision and less sensitivity to ill-conditioning would be achieved in a recursive mode. Luo *et al.* [26] proposed the Givens rotation with the Forward selection and EXponential windowing (GFEX) method. Together with the recursive least absolute shrinkage and selection operator (R-LASSO) regularization approach, Bobrow and Murray [29] proposed a less accurate but much faster algorithm termed ROLS-LASSO for time-critical applications. It features a recursive standardization of regressors and performs parameter estimation with the ROLS. In the GFEX and ROLS-LASSO, recursive implementation of QR decomposition requires several costly matrix operations to preserve the orthogonality of the regressor matrix.

Recently, the ROLS method proposed in [30] is directly applied to update the weight matrix of a RBF network, which is further extended to train RBF networks with a set of initial candidate centers generated by the k-means method [31]. It is then followed by the selection of suitable RBF centers from the information matrix after training. Using a localized forgetting method, an adaptive model of the chemical process rig for model predictive control is developed in [32]. Note that the information matrix relies on the initial centers generated by clustering methods and needs complete reorthogonalization in each step of the model selection. Nevertheless, for reliable termination criteria of model selection, the sample number used in the Akaike's FPE rule [32] need to be estimated. For structure optimization of radial basis probabilistic neural networks (RBPNNs) [33], in addition to the ROLS, a minimum volume covering hyperspheres algorithm is proposed to cluster initial hidden node centers while the PSO is employed to further enhance optimization of the initial structure of the RBPNN. Apparently, this algorithm would be extremely time consuming. Lately, a greedy RLS (GRLS) algorithm with exponential window is proposed in [34]. To save storage memory, the GRLS only allows the exchange of neighbor columns in the regressor matrix by conducting Householder reflectors on vectors. This technique is likely to reduce the training speed of model selection.

In this paper, novel constructive and destructive parsimonious ELMs (CP- and DP-ELM) for multi-input multi-output (MIMO) SLFNs are proposed. The main ideas of the two algorithms are as follows: initial hidden node candidates are randomly generated by the ELM and (recursively) orthogonalized using (sequentially arriving) data samples. Unlike previous ROLS procedures, novel recursive orthogonalization, which is decomposed into sequential partial orthogonalization (SPO) and only performed on partial matrices by Givens rotations in each iteration, is proposed.

It turns out that at most one nonzero element need to be decomposed into the first diagonal of the partial matrix while any orthogonal matrix need not be explicitly given. In this context, the hidden output matrix can be orthogonalized into an upper triangular matrix whose size is dramatically reduced as the initial hidden node number is significantly fewer than the sample number. Starting from the upper triangular matrix and the corresponding initial residual error (IRE), the CP- and DP-ELM are developed for constructive and destructive model selection, respectively, since each column corresponds to one hidden node. In the CP-ELM, regressors with maximal significance are recursively added to the subset model selection till the additional residual error reduction (ARER) satisfies the termination criterion. In the DP-ELM, regressors with the minimal significance are recursively removed from current subset till the ARER meets the termination criterion. To achieve fast retriangularization, constructive and destructive SPO (CSPO and DSPO) are proposed to recursively reorthogonalize the augmented and reduced regressor matrices in the CP- and DP-ELM. Note that the output weight matrix is not involved in the foregoing training procedures. Finally, the output weights are solved from the resulting upper triangular equation.

The rest of this paper is organized as follows. Section II briefly presents preliminaries of related works. The key contributions to the CP- and DP-ELM algorithms including candidate regressor generation and transformation based on SPO, model selection using CSPO and DSPO, and output weight estimation are developed in Section III. Section IV presents the simulation studies on the CP- and DP-ELM for multi- and single-output regression benchmark data sets and innovative applications to nonlinear time-series modeling. In Section V, the conclusion is drawn.

II. Preliminaries

In this section, fundamental preliminaries pertinent to our work, i.e., ELM [1] and ROLS [28], are presented.

A. Extreme Learning Machine

Considering the unified framework of SLFNs using additive or RBF nodes, the output of a SLFN is governed by

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \boldsymbol{\theta}_i h(\mathbf{a}_i, b_i, \mathbf{x}), \quad \mathbf{x}, \mathbf{a}_i \in \mathbb{R}^n,\ b_i \in \mathbb{R} \tag{1}$$

where $\mathbf{a}_i$ and b_i are the learning parameters of hidden nodes, $\boldsymbol{\theta}_i$ is the weight connecting the ith hidden node to the output node, $h(\mathbf{a}_i, b_i, \mathbf{x})$ is the output of the ith hidden node with respect to the input $\mathbf{x}$, and L is the number of hidden nodes.

It was shown in [2] that SLFNs with a wide type of random computational hidden nodes possess the universal approximation capability. For a given set of training examples $\{\mathbf{x}(k), \mathbf{t}(k)\}_{k=1}^{K} \in \mathbb{R}^n \times \mathbb{R}^m$, if the network outputs are equal to the targets, we have the following compact formulation:

$$\mathbf{H}\boldsymbol{\Theta} = \mathbf{T} \tag{2}$$

where

$$\begin{aligned}\mathbf{H}(\mathbf{a}_1, \ldots, \mathbf{a}_L, b_1, \ldots, b_L, \mathbf{x}(1), \ldots, \mathbf{x}(K)) &= \begin{bmatrix} h(\mathbf{a}_1, b_1, \mathbf{x}(1)) & \cdots & h(\mathbf{a}_L, b_L, \mathbf{x}(1)) \\ \vdots & \ddots & \vdots \\ h(\mathbf{a}_1, b_1, \mathbf{x}(K)) & \cdots & h(\mathbf{a}_L, b_L, \mathbf{x}(K)) \end{bmatrix}_{K\times L} \\ &= [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_L] \end{aligned} \tag{3}$$

$$\boldsymbol{\Theta} = \begin{bmatrix} \boldsymbol{\theta}_1^T \\ \vdots \\ \boldsymbol{\theta}_L^T \end{bmatrix}_{L\times m} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}^T(1) \\ \vdots \\ \mathbf{t}^T(K) \end{bmatrix}_{K\times m}. \tag{4}$$

Here, $\mathbf{H}$ is called the hidden output matrix of the network, whereby the ith column is the ith hidden node's output vector with respect to the inputs $\mathbf{x}(1), \ldots, \mathbf{x}(K)$ and the jth row is the output vector of the hidden layer with respect to input $\mathbf{x}(j) = [x_1(j), \ldots, x_n(j)]^T$. The terms $\boldsymbol{\Theta}$ and $\mathbf{T}$ are corresponding matrices of the output weights and targets, respectively.

In this case, training a SLFN simply amounts to obtaining the solution of a linear system (2) of the output weights $\boldsymbol{\Theta}$. Under the constraint of minimum norm LS [4], i.e., $\min \|\boldsymbol{\Theta}\|$ and $\min \|\mathbf{H}\boldsymbol{\Theta} - \mathbf{T}\|$, a simple form of the solution of system (2) is given explicitly as follows:

$$\hat{\boldsymbol{\Theta}} = \mathbf{H}^{\dagger}\mathbf{T}, \quad \mathbf{H}^{\dagger} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T \tag{5}$$

where $\mathbf{H}^{\dagger}$ is the Moore–Penrose generalized inverse of $\mathbf{H}$. Huang and Chen [13] have further shown that if the K training data are distinct, $\mathbf{H}$ is of full-column rank with probability one when $L \leq K$. Note that in real-world applications, the number of hidden nodes is always less than the number of training data, i.e., $L < K$. To facilitate the subsequent discussions, the main results of [1] are restated here.

Lemma 1 [1]: Given a standard SLFN with K hidden nodes and activation function $h : \mathbb{R}^n \to \mathbb{R}$, which is infinitely differentiable in any interval, for K arbitrary distinct samples $(\mathbf{x}(k), \mathbf{t}(k))$, where $\mathbf{x}(k) \in \mathbb{R}^n$ and $\mathbf{t}(k) \in \mathbb{R}^m$, for any $\mathbf{a}_i$ and b_i randomly chosen from any intervals of $\mathbb{R}^n$ and $\mathbb{R}$, respectively, according to any continuous probability distribution, then, with probability one, the hidden layer output matrix $\mathbf{H}$ of the SLFN is invertible and $\|\mathbf{H}\boldsymbol{\Theta} - \mathbf{T}\| = 0$.

Lemma 2 [1]: Given any small value $\varepsilon > 0$ and activation function $h : \mathbb{R}^n \to \mathbb{R}$, which is infinitely differentiable in any interval, there exists $L \leq K$ such that for K arbitrary distinct samples $(\mathbf{x}(k), \mathbf{t}(k))$, where $\mathbf{x}(k) \in \mathbb{R}^n$ and $\mathbf{t}(k) \in \mathbb{R}^m$, for any $\mathbf{a}_i$ and b_i randomly chosen from any intervals of $\mathbb{R}^n$ and $\mathbb{R}$, respectively, according to any continuous probability distribution, then, with probability one, $\|\mathbf{H}_{K\times L}\boldsymbol{\Theta}_{L\times m} - \mathbf{T}_{K\times m}\| < \varepsilon$.

It follows that these hidden-layer parameters of the ELM can be randomly generated instead of being iteratively tuned. One needs only to calculate the output weights using the LS method in one step.

B. Recursive Orthogonal Least Squares

Considering (2) for a set of sequential samples of input–output training data $\{\mathbf{x}(k), \mathbf{t}(k)\}_{k=1}^{K} \in \mathbf{R}^n \times \mathbf{R}^m$, we have

$$\mathbf{T}^{(k)} = \mathbf{H}\left(\mathbf{X}^{(k)}\right)\boldsymbol{\Theta}(k) + \mathbf{E}^{(k)} \quad k = 1, 2, \ldots, K \tag{6}$$

where

$$\mathbf{X}^{(k)} = [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(k)]^T \tag{7}$$

$$\mathbf{T}^{(k)} = [\mathbf{t}(1), \mathbf{t}(2), \ldots, \mathbf{t}(k)]^T \tag{8}$$

$$\mathbf{E}^{(k)} = [\mathbf{e}(1), \mathbf{e}(2), \ldots, \mathbf{e}(k)]^T \tag{9}$$

and matrices $\mathbf{H}$ and $\boldsymbol{\Theta}$ are defined as (3) and (4), respectively.

The MIMO LS problem is formulated to solve $\boldsymbol{\Theta}(k)$ such that the following error cost function at iteration k is minimized:

$$\min_{\boldsymbol{\Theta}(k)\in\mathbb{R}^{L\times m}} \left\| \begin{bmatrix} \mathbf{T}^{(k-1)} \\ \mathbf{t}^T(k) \end{bmatrix} - \begin{bmatrix} \mathbf{H}^{(k-1)} \\ \mathbf{h}^T(k) \end{bmatrix} \boldsymbol{\Theta}(k) \right\|_F^2 \tag{10}$$

where $\| \bullet \|_F$ is the Frobenius norm defined as $\|\mathbf{E}\|_F^2 = tr(\mathbf{E}^T\mathbf{E})$, and

$$\mathbf{H}^{(k-1)} = [\mathbf{h}(1), \mathbf{h}(2), \ldots, \mathbf{h}(k-1)]^T \tag{11}$$

$$\mathbf{h}^T(i) = [h(\mathbf{a}_1, b_1, \mathbf{x}(i)), \ldots, h(\mathbf{a}_L, b_L, \mathbf{x}(i))]. \tag{12}$$

According to Lemma 1, the matrix $\mathbf{H}$ is of full-column rank with probability one if the corresponding mild constraints are satisfied. Hence, similar to [28], one can obtain the ROLS as follows.

Lemma 3 [28]: If

$$\mathbf{H}^{(k-1)} = \mathbf{Q}(k-1)\begin{bmatrix} \mathbf{R}(k-1) \\ \mathbf{0}_{(k-L-1)\times L} \end{bmatrix} \tag{13}$$

$$\mathbf{Q}^T(k-1)\mathbf{T}^{(k-1)} = \begin{bmatrix} \widehat{\mathbf{T}}^{(k-1)} \\ \widetilde{\mathbf{T}}^{(k-1)} \end{bmatrix} \tag{14}$$

where $\mathbf{Q}(k-1)$ is an $(k-1)\times(k-1)$ orthogonal matrix with orthogonal columns satisfying $\mathbf{Q}^T(k-1)\mathbf{Q}(k-1) = \mathbf{Q}(k-1)\mathbf{Q}^T(k-1) = \mathbf{I}$, and $\mathbf{R}(k-1)$ is an upper triangular matrix with the same rank as $\mathbf{H}^{(k-1)}$, then the minimizer of (10) is equivalent to the following minimizer:

$$\min_{\boldsymbol{\Theta}(k)\in\mathbb{R}^{L\times m}} \left\| \begin{bmatrix} \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{t}^T(k) \end{bmatrix} - \begin{bmatrix} \mathbf{R}(k-1) \\ \mathbf{h}^T(k) \end{bmatrix} \boldsymbol{\Theta}(k) \right\|_F^2. \tag{15}$$

Recursively, one can find the update for $\mathbf{R}(k-1)$ and $\widehat{\mathbf{T}}^{(k-1)}$ by the following orthogonal decomposition:

$$\begin{bmatrix} \mathbf{R}(k-1) \\ \mathbf{h}^T(k) \end{bmatrix} = \mathbf{Q}'(k) \begin{bmatrix} \mathbf{R}(k) \\ \mathbf{0}_{1\times L} \end{bmatrix} \tag{16}$$

$$\mathbf{Q}'^T(k) \begin{bmatrix} \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{t}^T(k) \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{T}}^{(k)} \\ \widetilde{\mathbf{t}}^T(k) \end{bmatrix}. \tag{17}$$

III. Parsimonious Model Selection

A SLFN trained by the ELM can achieve high generalization capability and obtain fast training speed without any iterations. However, these capabilities could only be guaranteed from the premise that a large set of candidate hidden nodes can be randomly generated to not only span the candidate space, i.e., $\text{rank}(\mathbf{H}) = L$, but also sufficiently cover the input training samples for high prediction accuracy. Hence, in addition to inevitably redundant candidate hidden nodes, the ELM faces the dilemma between the random generation of input weights and analytical solutions for output weights.

Accordingly, to circumvent the aforementioned problems, we mainly focus on parsimonious model selection for the ELM scheme using the ROLS method, and thereby optimizing the structure and parameters simultaneously. Hidden nodes of the candidate pool are randomly generated via the original ELM in the initial phase, and then recursively decomposed using the ROLS method as training data samples arrive sequentially. Furthermore, constructive and destructive model selection algorithms based on partial orthogonal decomposition are, respectively, proposed to rank and select significant hidden nodes to achieve parsimonious structure of a SLFN.

A. Generation and Transformation of Hidden Node Candidates

Hidden node candidates of the multioutput ELM (M-ELM) are randomly generated according to the original ELM strategy. Denote the number of candidates by $\bar{L}$, which generally satisfies that $\bar{L} < K$ (even $\bar{L} \ll K$), where K is the number of arbitrary distinct training samples $\{\mathbf{x}(k), \mathbf{t}(k)\}_{k=1}^{K} \in \mathbb{R}^n \times \mathbb{R}^m$. Accordingly, the output of the M-ELM with full candidates ($\bar{L}$ hidden nodes) for sequentially arriving data samples could be represented by

$$\mathbf{f}_{\bar{L}}(\mathbf{x}(k)) = \sum_{l=1}^{\bar{L}} \boldsymbol{\theta}_l h(\mathbf{a}_l, b_l, \mathbf{x}(k)), \mathbf{x}, \mathbf{a}_l \in \mathbb{R}^n, \boldsymbol{\theta}_l \in \mathbb{R}^m \tag{18}$$

where $\mathbf{a}_l$ and b_l are the hidden node parameters, $\boldsymbol{\theta}_l$ is the output weight, and $h(\mathbf{a}_l, b_l, \mathbf{x}(k))$ denotes the output of the lth hidden node with respect to the input $\mathbf{x}(k), k = 1, 2, \ldots, K$. Given the randomly generated hidden node parameters $\mathbf{a}_l$ and b_l, the sequential matrix with respect to the first kth data samples is governed by

$$\begin{bmatrix} \mathbf{T}^{(k-1)} \\ \mathbf{t}^T(k) \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{\bar{L}}^{(k-1)} \\ \mathbf{h}^T(k) \end{bmatrix} \boldsymbol{\Theta}(k) + \begin{bmatrix} \mathbf{E}^{(k-1)} \\ \mathbf{e}^T(k) \end{bmatrix} \tag{19}$$

where $\mathbf{H}_{\bar{L}}^{(k-1)} = [\mathbf{h}_1(\mathbf{X}^{(k-1)}), \ldots, \mathbf{h}_{\bar{L}}(\mathbf{X}^{(k-1)})]_{(k-1)\times\bar{L}}$ is the hidden node output matrix, $\mathbf{T}^{(k-1)}$ is the target matrix, and $\mathbf{E}^{(k-1)}$ is the error matrix with respect to the first $(k-1)$th data samples, respectively. The term $\mathbf{h}^T(k) = [\mathbf{h}_1(\mathbf{x}(k)), \ldots, \mathbf{h}_{\bar{L}}(\mathbf{x}(k))]_{1\times\bar{L}}$ is the hidden node output vector, $\mathbf{t}^T(k)$ is the target vector, $\mathbf{e}^T(k)$ is the error vector with respect to the kth data sample, respectively, and $\boldsymbol{\Theta}(k)$ the corresponding weight matrix at iteration k, which can be determined by the OS-ELM [5] using the RLS method, and thereby involving in matrix singularity and leading to numerical instability as well as inferior robustness since it considers all the candidate hidden nodes, which cannot be guaranteed to be linearly independent. To avoid the foregoing problems and to reduce the computational burden, one does not need to calculate the weight matrix here.

Furthermore, applying the ROLS method of Lemma 3 to (19), we obtain the minimized residual error of the SLFN with full candidate hidden nodes ($\bar{L}$) with respect to the past k training data samples as follows:

$$J(k) = \left\| \begin{bmatrix} \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{t}^T(k) \\ \widetilde{\mathbf{T}}^{(k-1)} \end{bmatrix} - \begin{bmatrix} \mathbf{R}(k-1) \\ \mathbf{h}^T(k) \\ \mathbf{0}_{(k-L-1)\times L} \end{bmatrix} \boldsymbol{\Theta}(k) \right\|_F^2. \tag{20}$$

From (16) and (17), one can obtain the following error cost:

$$J(k) = \left\| \begin{bmatrix} \widehat{\mathbf{T}}^{(k)} - \mathbf{R}(k)\boldsymbol{\Theta}(k) \\ \widetilde{\mathbf{t}}^T(k) \\ \widetilde{\mathbf{T}}^{(k-1)} \end{bmatrix} \right\|_F^2 \tag{21}$$

where the previous residual error $\widetilde{\mathbf{T}}^{(k-1)}$ and incremental error $\widetilde{\mathbf{t}}^T(k)$ can be recursively computed by (14) and (17), respectively. Actually, the orthogonal matrices $\mathbf{Q}(k-1)$ and $\mathbf{Q}'(k)$ need not be explicitly given or stored since the recursive procedure can be directly described as follows:

$$\begin{bmatrix} \mathbf{R}(k-1) & \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{h}^T(k) & \mathbf{t}^T(k) \end{bmatrix} = \mathbf{Q}(k) \begin{bmatrix} \mathbf{R}(k) & \widehat{\mathbf{T}}^{(k)} \\ \mathbf{0}_{1\times\bar{L}} & \widetilde{\mathbf{t}}^T(k) \end{bmatrix} \tag{22}$$

with the initial conditions assigned as follows: $\mathbf{R}(0) = \alpha\mathbf{I}_{\bar{L}\times\bar{L}}$ and $\widehat{\mathbf{T}}^{(0)} = \mathbf{0}_{\bar{L}\times m}$, where α is a small positive value. The term $\mathbf{Q}(k)$ is an orthogonal matrix implemented by the Givens rotating method of [19]. It should also be noted that the matrix $\mathbf{Q}(k)$ is not required to be explicitly recorded.

Furthermore, the foregoing ROLS procedure (22) can be decomposed into a series of SPO using Givens rotations on partial matrices. For the generation and transformation of hidden node candidates, we have the following main results.

Theorem 1: The ROLS procedure (22) can be decomposed into the following SPO using the Givens rotation on partial matrices:

$$\begin{bmatrix} \mathbf{R}(k-1) & \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{h}^T(k) & \mathbf{t}^T(k) \end{bmatrix} = \mathbf{Q}_1(k)\cdots\mathbf{Q}_{\bar{L}}(k) \begin{bmatrix} \mathbf{R}(k) & \widehat{\mathbf{T}}^{(k)} \\ \mathbf{0}_{1\times\bar{L}} & \widetilde{\mathbf{t}}^T(k) \end{bmatrix} \tag{23}$$

where $\mathbf{Q}_l(k), l = 1, \ldots, \bar{L}$ are orthogonal matrices satisfying

$$\mathbf{Q}_l(k)\mathbf{Q}_l^T(k) = \mathbf{Q}_l^T(k)\mathbf{Q}_l(k) = \mathbf{I} \tag{24}$$

$$\mathbf{Q}_l(k) = \begin{bmatrix} \mathbf{I}_{(l-1)\times(l-1)} & \\ & \bar{\mathbf{Q}}_l(k) \end{bmatrix} \tag{25}$$

such that

$$\begin{bmatrix} r_{ll}(k-1) & r_{l,l+1}(k-1) & \cdots & r_{l\bar{L}}(k-1) & \widehat{\mathbf{t}}_l^T(k-1) \\ 0 & * & \cdots & * & \star \\ \vdots & \ddots & \ddots & \vdots & \\ 0 & \cdots & 0 & * & \star \\ \otimes & \otimes & \cdots & \otimes & \otimes \end{bmatrix} = \bar{\mathbf{Q}}_l(k) \begin{bmatrix} r_{ll}(k) & r_{l,l+1}(k) & \cdots & r_{l\bar{L}}(k) & \widehat{\mathbf{t}}_l^T(k) \\ 0 & * & \cdots & * & \star \\ \vdots & \ddots & \ddots & \vdots & \\ 0 & \cdots & 0 & * & \star \\ 0 & * & \cdots & * & \star \end{bmatrix} \tag{26}$$

where $r_{ll}(k)$ and $\widehat{\mathbf{t}}_l^T(k)$ are the lth diagonal of $\mathbf{R}(k)$ and row of $\widehat{\mathbf{T}}^{(k)}$, and $\otimes$ denotes nonzero elements to be rotated.

Proof: From (23) and (24), we obtain

$$\mathbf{Q}_{\bar{L}}^T(k)\cdots\mathbf{Q}_1^T(k)\begin{bmatrix}\mathbf{R}(k-1) & \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{h}^T(k) & \mathbf{t}^T(k)\end{bmatrix} = \begin{bmatrix}\mathbf{R}(k) & \widehat{\mathbf{T}}^{(k)} \\ \mathbf{0}_{1\times\bar{L}} & \widetilde{\mathbf{t}}^T(k)\end{bmatrix}.$$

Denote $\mathbf{R}_1(k-1) = \begin{bmatrix}\mathbf{R}(k-1) & \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{h}^T(k) & \mathbf{t}^T(k)\end{bmatrix}$. Using (25) and (26), we have

$$\mathbf{Q}_1^T(k)\mathbf{R}_1(k-1) = \begin{bmatrix}\mathbf{R}(k-1) & \widehat{\mathbf{T}}^{(k-1)} \\ \mathbf{h}^T(k) & \mathbf{t}^T(k)\end{bmatrix} = \left[\begin{array}{c|c} r_{11}(k) & r_{12}(k)\cdots r_{1\bar{L}}(k)\ \widehat{\mathbf{t}}_1^T(k) \\ \hline \mathbf{0}_{\bar{L}\times 1} & \mathbf{R}_2(k-1)\end{array}\right].$$

Recursively, we have

$$\mathbf{Q}_l^T(k)\cdots\mathbf{Q}_1^T(k)\mathbf{R}_1(k-1) = \left[\begin{array}{c|c} \mathbf{R}_l(k) & \mathbf{R}_l'(k)\ \widehat{\mathbf{T}}_l(k) \\ \hline \mathbf{0}_{(\bar{L}-l+1)\times l} & \mathbf{R}_{l+1}(k-1)\end{array}\right]$$

where $\mathbf{R}_l(k-1), l = 1, 2, \ldots, \bar{L}$ are as shown on the left side of (26), and

$$\mathbf{R}_l(k) = \begin{bmatrix} r_{11}(k)\cdots r_{1l}(k) \\ \ddots\ \vdots \\ r_{ll}(k)\end{bmatrix}, \mathbf{R}_l'(k) = \begin{bmatrix} r_{1,l+1}(k)\cdots r_{1,\bar{L}}(k) \\ \vdots\quad\ddots\quad\vdots \\ r_{l,l+1}(k)\cdots r_{l,\bar{L}}(k)\end{bmatrix}$$

$$\widehat{\mathbf{T}}_l(k) = \left[\widehat{\mathbf{t}}_1(k)\ \cdots\ \widehat{\mathbf{t}}_l(k)\right]^T.$$

Let $\mathbf{Q}(k) = \mathbf{Q}_1(k)\mathbf{Q}_2(k)\cdots\mathbf{Q}_{\bar{L}}(k)$, we have that (23) is equivalent to (22). This completes the proof. ∎

In this context, from the viewpoint of foregoing insight of the ROLS procedure, one can obtain the IRE regardless of the singularity of matrix $\mathbf{R}$ as follows.

Theorem 2: If the optimal $\boldsymbol{\Theta} = \left[\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{\bar{L}}\right]^T$ is obtained by solving the equations $\mathbf{R}\boldsymbol{\Theta} = \widehat{\mathbf{T}}$ via backward substitution and setting $\boldsymbol{\theta}_l^T = \mathbf{0}$ for $r_{ll} = \alpha$, where $0 < \alpha \ll 1$ is the initial diagonal of $\mathbf{R}$ for avoiding division by zero and r_{ll} is the lth diagonal of $\mathbf{R}$, then the IRE of the M-ELM using the SPO method is determined by

$$\text{IRE} = \left\|\mathbf{T}\right\|_F^2 - \left\|\widehat{\mathbf{T}}\right\|_F^2 \tag{27}$$

where $\mathbf{R} = \mathbf{R}(K)$ and $\widehat{\mathbf{T}} = \widehat{\mathbf{T}}^{(K)}$ are the final upper triangular matrix and transformed output matrix, respectively.

Proof: It follows from Lemmas 1 and 2 that $\mathbf{R}$ is of full rank with probability one. In other words, it is also of certain possibility that $\text{rank}(\mathbf{R}) < \bar{L}$. Hence, we consider both cases for $\mathbf{R}$, i.e., $\text{rank}(\mathbf{R}) = \bar{L}$ and $\text{rank}(\mathbf{R}) < \bar{L}$, as follows.

1) Case 1 ($\text{rank}(\mathbf{R}) = \bar{L}$): Since matrix $\mathbf{R}$ is full rank, the diagonals r_{ll} should be nonzeros. Considering the initial value of the diagonals α, it requires that $r_{ll} > \alpha$. Note that the upper triangular matrix $\mathbf{R}$ can be recursively derived from the SPO according to Theorem 1. It follows that each row of $\mathbf{R}$ should be transformed as follows:

$$\mathbf{Q}_l(k) \neq \mathbf{I}\ \forall\ \ l = 1, 2, \ldots, \bar{L},\ \exists\, k \in \{1, 2, \ldots, K\}$$

where $\mathbf{Q}_l(k)$ is shown in Theorem 1. Correspondingly, the target matrix $\mathbf{T}$ has been orthogonally decomposed into each row of $\widehat{\mathbf{T}}$, and the optimal $\boldsymbol{\Theta} = \left[\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{\bar{L}}\right]^T$ can be solved from the following backward substitutions:

$$\left.\begin{aligned} r_{\bar{L}\bar{L}}\boldsymbol{\theta}_{\bar{L}}^T &= \widehat{\mathbf{t}}_{\bar{L}}^T \\ r_{\bar{L}-1,\bar{L}-1}\boldsymbol{\theta}_{\bar{L}-1}^T + r_{\bar{L}-1,\bar{L}}\boldsymbol{\theta}_{\bar{L}}^T &= \widehat{\mathbf{t}}_{\bar{L}-1}^T \\ &\vdots \\ r_{11}\boldsymbol{\theta}_1^T + \cdots + r_{1,\bar{L}-1}\boldsymbol{\theta}_{\bar{L}-1}^T + r_{1,\bar{L}}\boldsymbol{\theta}_{\bar{L}}^T &= \widehat{\mathbf{t}}_1^T \end{aligned}\right\}. \tag{28}$$

From (21) and (22), we obtain

$$\begin{aligned}\text{IRE} = J(K) &= \left\|\widetilde{\mathbf{T}}^{(K)}\right\|_F^2 = \left\|\widetilde{\mathbf{t}}^T(K)\right\|_F^2 + \left\|\widetilde{\mathbf{T}}^{(K-1)}\right\|_F^2 \\ &= \sum_{k=1}^{K}\left\|\widetilde{\mathbf{t}}^T(k)\right\|_F^2 = \left\|\mathbf{T}\right\|_F^2 - \left\|\widehat{\mathbf{T}}\right\|_F^2.\end{aligned} \tag{29}$$

2) Case 2 ($\text{rank}(\mathbf{R}) < \bar{L}$): It implies that at least one row of the elements are all zeros. Considering the initial diagonals α, it requires that there exists at least one row being not transformable into any decompositions

$$\mathbf{Q}_l(k) = \mathbf{I}\ \ \forall\, k = 1, 2, \ldots, K,\ \exists\, l \in \{1, 2, \ldots, \bar{L}\}.$$

The lth row of $\widehat{\mathbf{T}}$, i.e., $\widehat{\mathbf{t}}_l^T$ are rendered as zeros in addition to $r_{ll} = \alpha \to 0$, and thereby leading to zero divided by zero while calculating $\boldsymbol{\theta}_l^T$ by backward substitution (28). Alternatively, we set $\boldsymbol{\theta}_l^T = \mathbf{0}$, which not only avoids the foregoing problem but also guarantees the IRE is fulfilled since $\widehat{\mathbf{t}}_l^T = \mathbf{0}$. Combining with (29), (27) holds. This concludes the proof. ∎

Accordingly, the original problem (18) can be directly transformed into the following upper triangular formulation:

$$\mathbf{R}\boldsymbol{\Theta} = \widehat{\mathbf{T}} \tag{30}$$

where $\mathbf{R} = \mathbf{R}(K)$ and $\widehat{\mathbf{T}} = \widehat{\mathbf{T}}^{(K)}$ are final upper triangular matrix and transformed output matrix, respectively. It should be pointed out that the weight matrix $\boldsymbol{\Theta}$ is not required here although it can be easily solved according to Theorem 2. It follows that the columns $\mathbf{r}_l$ of the matrix $\mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{\bar{L}}]$ are considered as candidate regressors and the rows $\widehat{\mathbf{t}}_l^T$ of the matrix $\widehat{\mathbf{T}} = [\widehat{\mathbf{t}}_1, \widehat{\mathbf{t}}_2, \ldots, \widehat{\mathbf{t}}_{\bar{L}}]^T$ are the corresponding outputs

$$\widehat{\mathbf{T}} = \sum_{l\in S_L}\mathbf{r}_l\boldsymbol{\theta}_l^T + \widehat{\mathbf{E}}_{S_L} \tag{31}$$

where $S_L = \{l_1, l_2, \ldots, l_L\} \subseteq \{1, 2, \ldots, \bar{L}\}, L \leq \bar{L}$ is the index subset of regressors selected from the full candidates, and $\widehat{\mathbf{E}}_{S_L}$ is the ARE corresponding to the index subset S_L.

Hence, candidate ranking and model selection are ready to be conducted on (30) or (31), i.e., matrices $\mathbf{R}$ and $\widehat{\mathbf{T}}$, based on the significance of corresponding regressors to the output.

B. Constructive Parsimonious Extreme Learning Machine

The CP-ELM scheme begins with no regressors, i.e., the initial regressor matrix $\mathbf{R}_r^{[0]} = \emptyset$. Next, the regressor matrix $\mathbf{R}_r^{[i-1]}$ incrementally recruits the most significant regressor $\mathbf{r}_{l_i}^{[i-1]}$ from the current candidate matrix $\mathbf{R}_c^{[i-1]}$, which correspondingly deletes the selected column

$$\left.\begin{aligned}\mathbf{R}_r^{[i-1]} &\leftarrow \left[\mathbf{R}_r^{[i-1]}, \mathbf{r}_{l_i}^{[i-1]}\right] \\ \mathbf{R}_c^{[i-1]} &\leftarrow \mathbf{R}_c^{[i-1]} \backslash \mathbf{r}_{l_i}^{[i-1]},\ i = 1, 2, \ldots, L\end{aligned}\right\} \tag{32}$$

where $\mathbf{R}_r^{[i-1]} = [\mathbf{r}_{l_1}^{[i-1]}, \mathbf{r}_{l_2}^{[i-1]}, \ldots, \mathbf{r}_{l_i}^{[i-1]}]$, $\mathbf{R}_c^{[i-1]} = [\mathbf{r}_{l_{i+1}}^{[i-1]}, \mathbf{r}_{l_{i+2}}^{[i-1]}, \ldots, \mathbf{r}_{l_{\bar{L}}}^{[i-1]}]$, $\mathbf{r}_{l_i}^{[i-1]} \in \mathbb{R}^{\bar{L}}$, $L \le \bar{L}$, and $l_j \in \{1, \ldots, \bar{L}\}$, $j = 1, \ldots, \bar{L}$. We denote $\mathbf{R}_r^{[0]} = \emptyset$ and $\mathbf{R}_c^{[0]} = \mathbf{R}$.

This is followed by retriangularization of the regressor matrix $\mathbf{R}_r^{[i-1]}$ together with the candidate matrix $\mathbf{R}_c^{[i-1]}$ using Givens orthogonal transformation as follows:

$$\left[\mathbf{R}_r^{[i-1]}\ \mathbf{R}_c^{[i-1]}\right] = \mathbf{Q}^{[i]}\left[\mathbf{R}_r^{[i]}\ \mathbf{R}_c^{[i]}\right] \tag{33a}$$

$$\mathbf{R}_r^{[i]} = \begin{bmatrix}\mathbf{R}^{[i]} \\ \mathbf{0}_{(\bar{L}-i)\times i}\end{bmatrix} \tag{33b}$$

$$\left(\mathbf{Q}^{[i]}\right)^T\begin{bmatrix}\mathbf{T}^{[i-1]} \\ \widehat{\mathbf{T}}^{[i-1]}\end{bmatrix} = \begin{bmatrix}\mathbf{T}^{[i]} \\ \widehat{\mathbf{T}}^{[i]}\end{bmatrix} \tag{33c}$$

where $\mathbf{T}^{[i]} \in \mathbb{R}^{i\times m}$, $\widehat{\mathbf{T}}^{[i]} \in \mathbb{R}^{(\bar{L}-i)\times m}$, and $\mathbf{R}^{[i]} \in \mathbb{R}^{i\times i}$ are an upper triangular matrix, $\mathbf{R}_r^{[0]} = \emptyset$, $\widehat{\mathbf{T}}^{[0]} = \widehat{\mathbf{T}}$, and $\mathbf{Q}^{[i]}$ are an orthogonal matrix derived from Givens rotation and it need not be explicitly recorded. In this case, the iterative regressor selection procedure can be steadily established by the following two important results.

Theorem 3: The ith regressor selection $\mathbf{r}_{l_i}^{[i-1]}$ is equivalent to the following CSPO implemented by partial Givens row rotations:

$$\left[\mathbf{r}_{l_i,i\sim j_i}^{[i-1]}\ \widehat{\mathbf{T}}_{1\sim j_i-i+1}^{[i-1]}\right] = \mathbf{Q}_{i\sim j_i}^{[i]}\begin{bmatrix} r_{ii}^{[i]} & \left(\mathbf{t}^{[i]}\right)^T \\ \mathbf{0}_{(j_i-i)\times 1} & \widehat{\mathbf{T}}_{1\sim j_i-i}^{[i]}\end{bmatrix} \tag{34}$$

where j_i is the number of nonzero elements of $\mathbf{r}_{l_i}^{[i-1]}$, subscript $i \sim j_i$ denote rows from i to j_i, $j_i \ge i$, $r_{ii}^{[i]}$ and $\left(\mathbf{t}^{[i]}\right)^T$ correspond to the ith diagonal of $\mathbf{R}_r^{[i]}$ and the ith row of $\mathbf{T}^{[i]}$, respectively, and the partial matrix $\mathbf{Q}_{i\sim j_i}^{[i]}$ satisfies $\mathbf{Q}_{i\sim j_i}^{[i]}\left(\mathbf{Q}_{i\sim j_i}^{[i]}\right)^T = \left(\mathbf{Q}_{i\sim j_i}^{[i]}\right)^T\mathbf{Q}_{i\sim j_i}^{[i]} = \mathbf{I}$ and is given by

$$\mathbf{Q}^{[i]} = \begin{bmatrix}\mathbf{I}_{(i-1)\times(i-1)} & & \\ & \mathbf{Q}_{i\sim j_i}^{[i]} & \\ & & \mathbf{I}_{(\bar{L}-j_i)\times(\bar{L}-j_i)}\end{bmatrix}. \tag{35}$$

Proof: From (33), the retriangularization for the ith regressor matrix need only to zero the nonzero elements below the ith row in the ith column of matrix $\left[\mathbf{R}_r^{[i-1]}, \mathbf{R}_c^{[i-1]}\right]$ since the previous columns have been triangularized as

$$\left[\mathbf{r}_{l_1}^{[i-1]}, \mathbf{r}_{l_2}^{[i-1]}, \ldots, \mathbf{r}_{l_{i-1}}^{[i-1]}\right] = \begin{bmatrix}\mathbf{R}^{[i-1]} \\ \mathbf{0}_{(\bar{L}-i+1)\times(i-1)}\end{bmatrix}.$$

That is, only row rotations between the ith element and $i+1, \ldots, j_i$th elements need to be conducted since the regressor $\mathbf{r}_{l_i}^{[i-1]}$ just has j_i nonzero elements denoted as $\mathbf{r}_{l_i,i\sim j_i}^{[i-1]}$

$$\mathbf{r}_{l_i,i\sim j_i}^{[i-1]} = \mathbf{Q}_{i\sim j_i}^{[i]}\begin{bmatrix} r_{ii}^{[i]} \\ \mathbf{0}_{(j_i-i)\times 1}\end{bmatrix}$$

where $\mathbf{Q}_{i\sim j_i}^{[i]}$ is an orthogonal matrix. Using (35), we obtain

$$\begin{aligned}&\left(\mathbf{Q}^{[i]}\right)^T\left[\mathbf{R}_r^{[i-1]}\ \mathbf{R}_c^{[i-1]}\right] \\ &\quad= \begin{bmatrix}\mathbf{R}^{[i-1]} & \mathbf{r}_{l_i,1\sim i-1}^{[i-1]} & \mathbf{R}_{c,1\sim i-1}^{[i-1]} \\ \mathbf{0}_{1\times(i-1)} & r_{ii}^{[i]} & \mathbf{R}_{c,i}^{[i]} \\ \mathbf{0}_{(j_i-i)\times(i-1)} & \mathbf{0}_{(j_i-i)\times 1} & \mathbf{R}_{c,i+1\sim j_i}^{[i]} \\ \mathbf{0}_{(\bar{L}-j_i)\times(i-1)} & \mathbf{0}_{(\bar{L}-j_i)\times 1} & \mathbf{R}_{c,j_i+1\sim\bar{L}}^{[i-1]}\end{bmatrix}\end{aligned}$$

and $\left(\mathbf{Q}^{[i]}\right)^T\begin{bmatrix}\mathbf{T}^{[i-1]} \\ \widehat{\mathbf{T}}^{[i-1]}\end{bmatrix} = \begin{bmatrix}\mathbf{T}_{1\sim i-1}^{[i-1]} \\ \left(\mathbf{t}^{[i]}\right)^T \\ \widehat{\mathbf{T}}_{1\sim j_i-i}^{[i]} \\ \widehat{\mathbf{T}}_{j_i-i+1\sim\bar{L}}^{[i-1]}\end{bmatrix}$.

Let

$$\mathbf{R}^{[i]} = \begin{bmatrix}\mathbf{R}^{[i-1]} & \mathbf{r}_{l_i,1\sim i-1}^{[i-1]} \\ \mathbf{0}_{1\times(i-1)} & r_{ii}^{[i]}\end{bmatrix},\ \mathbf{R}_c^{[i]} = \begin{bmatrix}\mathbf{R}_{c,1\sim i-1}^{[i-1]} \\ \mathbf{R}_{c,i}^{[i]} \\ \mathbf{R}_{c,i+1\sim j_i}^{[i]} \\ \mathbf{R}_{c,j_i+1\sim\bar{L}}^{[i-1]}\end{bmatrix}$$

$$\mathbf{T}^{[i]} = \begin{bmatrix}\mathbf{T}_{1\sim i-1}^{[i-1]} \\ \left(\mathbf{t}^{[i]}\right)^T\end{bmatrix},\ \widehat{\mathbf{T}}^{[i]} = \begin{bmatrix}\widehat{\mathbf{T}}_{1\sim j_i-i}^{[i]} \\ \widehat{\mathbf{T}}_{j_i-i+1\sim\bar{L}}^{[i-1]}\end{bmatrix}.$$

It is not difficult to observe that Givens orthogonal transformation (33) can be realized by (34) and (35). In addition, it should be noted that the selection of $\mathbf{r}_{l_i}^{[i-1]}$ is meaningless if $j_i < i$. When regressor $\mathbf{r}_{l_i}^{[i-1]}$ is selected as the ith column of $\mathbf{R}_r^{[i]}$ for retriangularization, there is no need to do any row rotations since there are only $j_i < i$ nonzero elements in the ith column. In this case, the ith row of $\mathbf{R}_r^{[i]}$ would become zero, which implies that the last row $\left(\mathbf{t}^{[i]}\right)^T$ of $\mathbf{T}^{[i]}$ cannot be canceled irrespective of the values assumed by the weight $\boldsymbol{\Theta}$. Hence, the selection of regressors is meaningful only if $j_i \ge i$. Specifically, the partial matrix $\mathbf{Q}_{i\sim j_i}^{[i]}$ would become 1 when $j_i = i$, and thereby the matrix $\mathbf{Q}^{[i]}$ being $\mathbf{I}$ which implies that no transformations are needed. This concludes the proof. ■

Theorem 4: The ARER ratio (ARERR) due to the ith regressor selection $\mathbf{r}_{l_i}^{[i-1]}$ using CSPO is determined by

$$\begin{aligned}\mathrm{ARERR}^{[i]}\left(\mathbf{r}_{l_i}^{[i-1]}\right) &\triangleq \left(\left\|\widehat{\mathbf{E}}_{S_{i-1}}\right\|_F^2 - \left\|\widehat{\mathbf{E}}_{S_i}\right\|_F^2\right)\Big/\left\|\widehat{\mathbf{T}}\right\|_F^2 \\ &= \left\|\left(\mathbf{t}^{[i]}\right)^T\right\|_F^2\Big/\left\|\widehat{\mathbf{T}}\right\|_F^2\end{aligned} \tag{36}$$

where $L \le \bar{L}$, $\widehat{\mathbf{E}}_{S_i}$ is the ARE corresponding to the index subset S_i, $\widehat{\mathbf{E}}_{S_0} = \widehat{\mathbf{T}}$, $\left(\mathbf{t}^{[i]}\right)^T$ is recursively obtained by (34), and $\widehat{\mathbf{T}}$ is the initial transformed target given by (31). Accordingly, the ARER is given by

$$\mathrm{ARER}(S_i) \triangleq \left\|\widehat{\mathbf{E}}_{S_i}\right\|_F^2\Big/\left\|\widehat{\mathbf{T}}\right\|_F^2 = 1 - \sum_{j=1}^{i}\mathrm{ARERR}^{[j]}. \tag{37}$$

Proof: From (31) and (33), once the ith regressor has been selected, the regression problem can be described as follows:

$$\mathbf{Q}^{[i]}\begin{bmatrix}\mathbf{T}^{[i]} \\ \widehat{\mathbf{T}}^{[i]}\end{bmatrix} = \mathbf{Q}^{[i]}\begin{bmatrix}\mathbf{R}^{[i]} \\ \mathbf{0}_{(\bar{L}-i)\times i}\end{bmatrix}\boldsymbol{\Theta}^{[i]} + \mathbf{E}^{[i]} \tag{38}$$

where $\mathbf{E}^{[i]} = \widehat{\mathbf{E}}_{S_i}$. Accordingly, the optimal weight can be solved by minimizing the following residual error:

$$J^{[i]} = \left\|\mathbf{E}^{[i]}\right\|_F^2 = \left\|\begin{bmatrix}\mathbf{T}^{[i]} - \mathbf{R}^{[i]}\boldsymbol{\Theta}^{[i]} \\ \widehat{\mathbf{T}}^{[i]}\end{bmatrix}\right\|_F^2. \tag{39}$$

One can obtain the optimal $\boldsymbol{\Theta}^{[i]}$ by solving the equation $\mathbf{R}^{[i]}\boldsymbol{\Theta}^{[i]} = \mathbf{T}^{[i]}$. This leads to the minimized residual error as $J_{\min}^{[i]} = \|\widehat{\mathbf{T}}^{[i]}\|_F^2$.

Similarly, the previous residual error for the $i-1$ selected regressors can be rewritten as $J_{\min}^{[i-1]} = \|\widehat{\mathbf{T}}^{[i-1]}\|_F^2$. In this case, by (34) in Theorem 3, the ARE reduction due to the ith newly recruited regressor can be computed as follows:

$$\begin{aligned}\left\|\mathbf{E}^{[i-1]}\right\|_F^2 - \left\|\mathbf{E}^{[i]}\right\|_F^2 &= \left\|\widehat{\mathbf{T}}^{[i-1]}\right\|_F^2 - \left\|\widehat{\mathbf{T}}^{[i]}\right\|_F^2 \\ &= \left\|\left(\mathbf{t}^{[i]}\right)^T\right\|_F^2. \end{aligned} \tag{40}$$

Hence, (36) holds. In addition, since

$$\sum_{j=1}^{i} \mathrm{ARERR}^{[j]}\left(\mathbf{r}_{l_j}^{[i-1]}\right) = \left(\left\|\widehat{\mathbf{E}}_{S_0}\right\|_F^2 - \left\|\widehat{\mathbf{E}}_{S_i}\right\|_F^2\right) \Big/ \left\|\widehat{\mathbf{T}}\right\|_F^2$$

and $\widehat{\mathbf{E}}_{S_0} = \widehat{\mathbf{T}}$, (37) holds. This concludes the proof. ■

Based on the foregoing results, we now propose the selection criterion under which the most significant regressors are recommended to be selected incrementally, and the termination criterion for determining the total number of hidden nodes.

For the ith regressor selection, it is required to select the regressor $\mathbf{r}_{l_i^\star}$ such that

$$l_i^\star = \arg \max_{i \le l_i \le \bar{L}} \mathrm{ARERR}^{[i]}\left(\mathbf{r}_{l_i}^{[i-1]}\right) \tag{41}$$

where the $\mathrm{ARERR}^{[i]}$ can be derived from (34) and (36).

Accordingly, the ARER can be calculated as follows:

$$\mathrm{ARER}^{[i]}\left(S_i^\star\right) = 1 - \sum_{j=1}^{i} \mathrm{ARERR}^{[j]}\left(\mathbf{r}_{l_j^\star}^{[i-1]}\right) \tag{42}$$

where $S_i^\star = \{l_1^\star, \ldots, l_i^\star\}$ is the optimal regressor index subset.

To realize the intuitive termination criterion of the constructive selection procedure, a predefined threshold $\epsilon \ll 1$ can be chosen such that the model selection procedure would stop once the following criterion is satisfied:

$$\mathrm{ARER}^{[i]} < \epsilon \ll 1. \tag{43}$$

It follows that the regressor indices in $S_i^\star$ corresponding to the original hidden nodes $\{\mathbf{h}_{l_1^\star}, \mathbf{h}_{l_2^\star}, \ldots, \mathbf{h}_{l_i^\star}\}$ can be readily obtained. We can directly solve the optimal weight $\boldsymbol{\Theta}$ from the following upper triangular matrix equation:

$$\mathbf{R}^{[i]}\boldsymbol{\Theta} = \mathbf{T}^{[i]} \tag{44}$$

where matrices $\mathbf{R}^{[i]}$ and $\mathbf{T}^{[i]}$ are derived from (33). It should be noted that the weight $\boldsymbol{\Theta}$ can be easily solved by backward substitution since $\mathbf{R}^{[i]}$ is an upper triangular matrix with nonzero diagonal entries. Accordingly, the complete CP-ELM algorithm can be summarized as follows.

Step 1: Given sequential training data $\{\mathbf{x}(k), \mathbf{t}(k)\}_{k=1}^K \in \mathbb{R}^n \times \mathbb{R}^m$ and randomly generated initial hidden nodes $h(\mathbf{a}_l, b_l, \mathbf{x}(k))$, $l = 1, 2, \ldots, \bar{L}$, the SPO method in (23) is used to orthogonally transform the hidden node output matrix $\mathbf{H}$ and target matrix $\mathbf{T}$ into $\mathbf{R}$ and $\widehat{\mathbf{T}}$ as (30), respectively.

Step 2: Begin with no regressor, i.e., $\mathbf{R}_r^{[0]} = \emptyset$, $\mathbf{R}_c^{[0]} = \mathbf{R}$ and $\widehat{\mathbf{T}}^{[0]} = \widehat{\mathbf{T}}$. Next, at the ith iteration, the regressor matrix $\mathbf{R}_r^{[i-1]}$ incrementally recruits the most significant regressor $\mathbf{r}_{l_i^\star}^{[i-1]}$ from the current candidate matrix $\mathbf{R}_c^{[i-1]}$ which correspondingly deletes the selected column according to (32). Using (34), retriangularization on matrices $\mathbf{R}_r^{[i-1]}$ and $\widehat{\mathbf{T}}^{[i-1]}$ is realised by CSPO to obtain $\mathbf{R}_r^{[i]}$ and $\widehat{\mathbf{T}}^{[i]}$. From (41) and (42), calculate the $\mathrm{ARER}^{[i]}\left(S_i^\star\right)$, where $S_i^\star = \{l_1^\star, l_2^\star, \ldots, l_i^\star\}$ is the index set of selected regressors. If $\mathrm{ARER}^{[i]}\left(S_i^\star\right) < \epsilon$, set $i = i+1$. Otherwise, go to Step 3.

Step 3: Stop the regressor selection procedure and solve the weight matrix $\boldsymbol{\Theta}$ from $\mathbf{R}^{[i]}\boldsymbol{\Theta} = \mathbf{T}^{[i]}$. Finally, the selected hidden nodes are stored as $\mathbf{H} = [\cdots, \mathbf{h}_l, \cdots]$, where $l \in S_i^\star$.

C. Destructive Parsimonious Extreme Learning Machine

The DP-ELM scheme begins with full regressors as a candidate pool, i.e., the initial regressor matrix $\mathbf{R}^{[0]} = \mathbf{R}$ and $\widehat{\mathbf{T}}^{[0]} = \widehat{\mathbf{T}}$. Next, at the ith iteration, the candidate regressor matrix $\mathbf{R}_c^{[i]}$ is obtained by removing the least significant regressor from the previous regressor matrix $\mathbf{R}^{[i-1]}$

$$\mathbf{R}_c^{[i]} \leftarrow \mathbf{R}^{[i-1]} \backslash \mathbf{r}_{l_i}^{[i-1]} \quad i = 1, 2, \ldots, L \tag{45}$$

where $L < \bar{L}$, $l_i \in \{1, 2, \ldots, \bar{L}\}$ denotes the original index of the removed regressor, and $\mathbf{r}_{l_i}^{[i-1]}$ denotes the removed column regressor in the previous regressor matrix $\mathbf{R}^{[i-1]}$, which is recursively retriangularized by Givens rotation as follows:

$$\mathbf{R}_c^{[i]} = \mathbf{Q}^{[i]} \begin{bmatrix} \mathbf{R}^{[i]} \\ \mathbf{0}_{1\times(\bar{L}-i)} \end{bmatrix} \tag{46}$$

$$\left(\mathbf{Q}^{[i]}\right)^T \widehat{\mathbf{T}}^{[i-1]} = \begin{bmatrix} \widehat{\mathbf{T}}^{[i]} \\ \left(\mathbf{t}^{[i]}\right)^T \end{bmatrix} \tag{47}$$

where $\mathbf{Q}^{[i]}$ is the orthogonal matrix satisfying $\mathbf{Q}^{[i]}(\mathbf{Q}^{[i]})^T = (\mathbf{Q}^{[i]})^T\mathbf{Q}^{[i]} = \mathbf{I}_{(\bar{L}-i+1)\times(\bar{L}-i+1)}$, $\mathbf{R}^{[i]}$ is an upper triangular matrix, and $(\mathbf{t}^{[i]})^T$ is the last row of the transformed output. Actually, the aforementioned procedure would become much simpler since it needs only one Givens rotation to be performed on each column of the partial matrix including the columns after the (l_i-1)th column in $\mathbf{R}_c^{[i]}$.

Corollary 1: The ith regressor- removal $\mathbf{r}_{l_i}^{[i-1]}$ is equivalent to the following DSPO using partial Givens row rotations:

$$\begin{bmatrix} \mathbf{R}_{c,j(i)}^{[i]} & \widehat{\mathbf{T}}_{j(i)}^{[i-1]} \end{bmatrix} = \mathbf{Q}_{j(i)}^{[i]} \begin{bmatrix} \mathbf{R}_{j(i)}^{[i]} & \widehat{\mathbf{T}}_{j(i)}^{[i]} \\ \mathbf{0}_{1\times(\bar{L}-i-j(i)+1)} & \left(\mathbf{t}^{[i]}\right)^T \end{bmatrix} \tag{48}$$

where $j(i)$ denotes the column index of the removed regressor from the previous regressor matrix $\mathbf{R}^{[i-1]}$, $\mathbf{R}_{c,j(i)}^{[i]}$, and $\mathbf{R}_{j(i)}^{[i]}$ are the submatrices including elements from $j(i)$th row and $j(i)$th column to the end of $\mathbf{R}_c^{[i]}$ and $\mathbf{R}^{[i]}$, respectively, $\widehat{\mathbf{T}}_{j(i)}^{[i-1]}$ and $\widehat{\mathbf{T}}_{j(i)}^{[i]}$ are the submatrices including rows from $j(i)$th to the end of $\widehat{\mathbf{T}}^{[i-1]}$ and $\widehat{\mathbf{T}}^{[i]}$, respectively. The orthogonal matrix $\mathbf{Q}_{j(i)}^{[i]}$ is governed by

$$\mathbf{Q}^{[i]} = \begin{bmatrix} \mathbf{I}_{(j(i)-1)\times(j(i)-1)} & \\ & \mathbf{Q}_{j(i)}^{[i]} \end{bmatrix}. \tag{49}$$

Proof: This proof is similar to that of Theorem 3. ■

It should be noted that $\mathbf{R}^{[0]} = \mathbf{R}$, $\widehat{\mathbf{T}}^{[0]} = \widehat{\mathbf{T}}$ and $\mathbf{Q}_{j(i)}^{[i]}$ is an orthogonal matrix derived from the DSPO and need not be explicitly recorded. In this case, the ARE growth (AREG) due to the removed regressor $\mathbf{r}_{l_i}^{[i-1]}$ can be obtained using the following corollary.

Corollary 2: In the ith iteration, the AREG ratio (AREGR) due to the l_ith regressor $\mathbf{r}_{l_i}^{[i-1]}$ removal by DSPO is given by

$$\begin{aligned}\mathrm{AREGR}^{[i]}\left(\mathbf{r}_{l_i}^{[i-1]}\right) &\triangleq \left(\left\|\widehat{\mathbf{E}}_{S_i}\right\|_F^2 - \left\|\widehat{\mathbf{E}}_{S_{i-1}}\right\|_F^2\right) \Big/ \left\|\widehat{\mathbf{T}}\right\|_F^2 \\ &= \left\|\left(\mathbf{t}^{[i]}\right)^T\right\|_F^2 \Big/ \left\|\widehat{\mathbf{T}}\right\|_F^2 \end{aligned} \tag{50}$$

where $L < \bar{L}$, $\widehat{\mathbf{E}}_{S_i}$ is the ARE corresponding to the index subset S_i, $\widehat{\mathbf{E}}_{S_0} = \mathbf{0}$, $S_0 = \emptyset$, $\left(\mathbf{t}^{[i]}\right)^T$ is recursively obtained by (48), and $\widehat{\mathbf{T}}$ is the initial transformed target given by (31). Accordingly, the ARER is given by

$$\mathrm{ARER}(S_i) \triangleq \left\|\widehat{\mathbf{E}}_{S_i}\right\|_F^2 \Big/ \left\|\widehat{\mathbf{T}}\right\|_F^2 = \sum_{j=1}^{i} \mathrm{AREGR}^{[j]}. \tag{51}$$

Proof: In the ith iteration, from (48) in Corollary 1, the last row $\left(\mathbf{t}^{[i]}\right)^T$ of the target matrix would remain regardless of how the weight matrix $\boldsymbol{\Theta}$ is optimized. The following proof is similar to that of Theorem 4. One can easily prove that (50) and (51) hold. This concludes the proof. ■

Given that we have established the foregoing results, we now propose the removal criterion under which the most insignificant regressors are selected to be removed iteratively, as well as the termination criterion, which determines the final structure of the DP-ELM.

For the removal of the ith regressor, it is required to delete the regressor $\mathbf{r}_{l_i^\circ}^{[i-1]}$ such that

$$l_i^\circ = \arg\min_{l_i} \mathrm{AREGR}^{[i]}\left(\mathbf{r}_{l_i}^{[i-1]}\right) \tag{52}$$

where the $\mathrm{AREGR}^{[i]}$ is derived from (50). Accordingly, the ARER can be calculated as follows:

$$\mathrm{ARER}^{[i]}\left(S_i^\circ\right) = \sum_{j=1}^{i} \mathrm{AREGR}^{[j]}\left(\mathbf{r}_{l_j^\circ}^{[j-1]}\right) \tag{53}$$

where $S_i^\circ = \{l_1^\circ, \ldots, l_i^\circ\}$ is the index subset of removals.

Similar to the CP-ELM termination criterion, the same threshold $\epsilon \ll 1$ can be used in DP-ELM such that (43) holds. Once the removal of the l_{i+1}°th regressor fulfills this criterion, the DP-ELM model selection procedure would stop and the regressor indices $\{l_1^\circ, \ldots, l_i^\circ\}$ corresponding to the original hidden nodes $\{\mathbf{h}_{l_1^\circ}, \ldots, \mathbf{h}_{l_i^\circ}\}$ can be removed. Then, we can directly solve the optimal weight $\boldsymbol{\Theta}$ from the final upper triangular equation $\mathbf{R}^{[i]}\boldsymbol{\Theta} = \widehat{\mathbf{T}}^{[i]}$, where matrices $\mathbf{R}^{[i]}$ and $\widehat{\mathbf{T}}^{[i]}$ are derived from (48) via iterative partial transformation. The complete algorithm structure of the DP-ELM is as follows.

Step 1: Given sequential training data $\{\mathbf{x}(k), \mathbf{t}(k)\}_{k=1}^K \in \mathbb{R}^n \times \mathbb{R}^m$ and randomly generated initial hidden nodes $h\left(\mathbf{a}_l, b_l, \mathbf{x}(k)\right), l = 1, 2, \ldots, \bar{L}$, the ROLS method given by (23) is used to orthogonally transform the hidden node output matrix $\mathbf{H}$ and target matrix $\mathbf{T}$ into $\mathbf{R}$ and $\widehat{\mathbf{T}}$ given by (30), respectively.

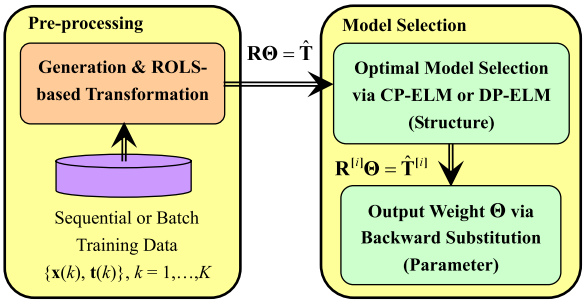

Fig. 1. Entire scheme for the CP- and DP-ELM.

Step 2: Begin with full regressors, i.e., $\mathbf{R}^{[0]} = \mathbf{R}$ and $\widehat{\mathbf{T}}^{[0]} = \widehat{\mathbf{T}}$. Next, at the ith iteration, the candidate regressor matrix $\mathbf{R}_c^{[i]}$ is obtained by removing the least significant regressor $\mathbf{r}_{l_i^\circ}^{[i-1]}$ from the previous regressor matrix $\mathbf{R}^{[i-1]}$ given by (45), which is recursively derived from the DSPO given by (48), and thereby the transformed output matrix $\widehat{\mathbf{T}}^{[i]}$. According to (51) and (53), calculate the $\mathrm{ARER}^{[i]}\left(S_i^\circ\right)$, where $S_i^\circ = \{l_1^\circ, l_2^\circ, \ldots, l_i^\circ\}$ is the index set of removed regressors. If $\mathrm{ARER}^{[i]}\left(S_i^\circ\right) < \epsilon$, set $i = i + 1$. Otherwise, go to Step 3.

Step 3: Stop the regressor removal procedure and solve the weight matrix $\boldsymbol{\Theta}$ from $\mathbf{R}^{[i]}\boldsymbol{\Theta} = \widehat{\mathbf{T}}^{[i]}$. The finally selected hidden nodes are stored as $\mathbf{H} = [\cdots, \mathbf{h}_l, \cdots]$, where $l \notin S_i^\circ$.

In summary, the entire scheme for the CP- and DP-ELM is depicted in Fig. 1, and the final residual error (FRE) can be computed as follows:

$$\mathrm{FRE} \triangleq \|\mathbf{E}\|_F^2 = \|\mathbf{T}\|_F^2 - (1 - \mathrm{ARER})\left\|\widehat{\mathbf{T}}\right\|_F^2 \tag{54}$$

where the ARER can be computed using (42) and (53) for CP- and DP-ELM, respectively. Since $\mathrm{ARER} < \epsilon \ll 1$, the FRE would be explicitly bounded as $\mathrm{FRE} < \mathrm{IRE} + \varepsilon$, where $\varepsilon = \epsilon\left\|\widehat{\mathbf{T}}\right\|_F^2$.

IV. Simulation Studies and Applications

In this section, performance evaluation and innovative applications of the proposed CP- and DP-ELM algorithms are carried out. First, effective performance and superiority of the proposed approaches are demonstrated by simulation studies on regression benchmark data sets [35], [36]. Next, innovative applications of CP- and DP-ELM to nonlinear time-series identification are presented. All the simulation studies are carried out using MATLAB R2011b running on a PC with a CPU of 2.0 and 2-GB RAM. For all the regression applications, the input data have been normalized to $[-1, 1]$ while the output data have been normalized to $[0, 1]$. The common parameter ϵ used in the CP- and DP-ELM is chosen as $\epsilon = 0.001$. To obtain the average performance rather than the best one, 30 trials are conducted for each data case.

A. Performance Evaluation on Benchmark Data Sets

Simulation studies are conducted on both multi- and single-output regression problems in Table I, whereby the training

TABLE I
SPECIFICATION OF REAL-WORLD BENCHMARK MULTIPLE- AND SINGLE-OUTPUT REGRESSIONS

Datasets	# Attributes	# Outputs	# Training	# Testing
Concrete slump	7	3	50	53
Abalone†	6	3	2000	2177
Wine (red)†	10	2	800	799
Wine (white)†	10	2	2400	2498
Auto MPG	7	1	196	200
Bank	8	1	4500	3692
Boston	13	1	250	256
California	8	1	8000	12640
Delta ailerons	5	1	3000	4129
Delta elevators	6	1	4000	5517
Servo	4	1	80	87

†The last 2 or 3 variables in original datasets are considered as outputs.

and testing samples are not overlapped and randomly selected from the original data sets. Furthermore, comprehensive comparisons of the CP- and DP-ELM with ELM [1], OP-ELM [10] and PCA-ELM [17] are presented using both Sigmoid and RBF hidden nodes. Note that only Sigmoid hidden nodes are used in the PCA-ELM since no other biases can be obtained except input weights using principal components. In this case, the number of initial hidden candidates in the PCA-ELM is equal to the input dimension. To have fair comparisons, the ELM employs the near optimal number of hidden nodes from the set $\{10, 20, \ldots, 100\}$ via cross-validation and are used as the initial candidates for OP-, CP-, and DP-ELM. Simulation results for real-world benchmark multi- and single-output regression cases are shown in Tables II and III, respectively, where the final numbers of hidden nodes of the OP-ELM take the resulting individual models with maximum hidden nodes since the original OP-ELM works on single-output regressions, i.e., separate single-output models with different hidden nodes estimating individual outputs, respectively. In comparison with the ELM and OP-ELM, the CP- and DP-ELM always select the optimal subset of hidden nodes from the set of candidate hidden nodes since the CP- and DP-ELM generate a SLFN based on the significance of the hidden nodes defined by ARERR and AREGR, and can achieve a parsimonious network structure with a predefined residual error. To make a fair comparison with the PCA-ELM, ELM, and OP-ELM, the Parsimony Ratio, which is defined as $L/\bar{L}$ is introduced to evaluate the final structure compactness with respect to the initial one, where L and $\bar{L}$ are the final and initial number of hidden nodes, respectively.

For the multioutput regression results shown in Table II, the CP- and DP-ELM using both Sigmoid and RBF hidden nodes uniformly achieve the smallest Parsimony Ratio, i.e., the most parsimonious structure. Simulation results also demonstrate that the CP- and DP-ELM with significantly parsimonious model selection from Sigmoid and RBF hidden candidates are able to achieve remarkably superior generalization in comparison with the OP- and PCA-ELM. Compared with the ELM, the CP- and DP-ELM using compact RBF nodes always perform better generalization in addition that the ones using significantly fewer Sigmoid nodes are capable of achieving superior performance except for the data set Wine(white), whereby the accuracy of CP- and DP-ELM would be comparable with that of the ELM. Due to the randomness of initial candidate hidden nodes in each simulation trial, the numbers of hidden nodes of the CP- and DP-ELM would slightly fluctuate around the mean values. However, the fluctuations are far less than those of OP-ELM. It can be observed from Table II that the DP-ELM is likely to obtain much more compact model since insignificant hidden nodes are recursively removed from previous candidate nodes till the ARER is not less than the threshold ϵ, while the CP-ELM recursively adds the most significant hidden nodes to the previously selected ones till the ARER is less than the same threshold ϵ. Note that the ARERR and AREGR are evaluated on the increasing and decreasing ARER, respectively. Compared with the CP-ELM, the DP-ELM would require more parsimonious hidden nodes for the same ϵ. Note that the DP-ELM starts from full candidates and inevitably have large initial computation burden, and thereby taking longer training time. On the contrary, the CP-ELM begins with no hidden nodes and recursively recruits significant hidden nodes, and thereby taking shorter training time. In comparison with ELM and PCA-ELM, the CP-, DP-, and OP-ELM would spend more training time due to optimal model selection. We can observe that the CP- and DP-ELM are much faster than the OP-ELM if small numbers of hidden nodes are initialized, and it will take a relatively long training time for large numbers of initial hidden nodes. Actually, building up a parsimonious SLFN from a relatively compact set of random hidden nodes rather than a large set is more meaningful for optimal model selection. In this case, the learning speeds of CP- and DP-ELM are acceptable in real-world applications where large numbers of initial hidden nodes are undesirable.

For the single-output data sets shown in Table III, it can be observed that the CP- and DP-ELM using both Sigmoid and RBF hidden nodes automatically create compact SLFNs with remarkable generalization. Compared with the OP-ELM, the CP- and DP-ELM achieve much more parsimonious structure for optimal model selection except for data sets Auto MPG and Servo, where the OP-ELM leads to excessive parsimony and deterioration in generalization performance. In comparison with PCA-ELM, the CP- and DP-ELM have much smaller Parsimony Ratio and significantly superior generalization can be achieved. From Table III, we can also observe that the CP- and DP-ELM preserve reliably stable model selection with minor deviations which reflect insensitivities to the randomness of hidden nodes in the initial candidate pool. However, the model structure from the OP-ELM varies critically with the initial candidate reservoir of random hidden nodes.

Note that the output weights of CP- and DP-ELM are solved from the final model selection. The resulting accuracy is much higher than those of the OP- and PCA-ELM since insignificant hidden nodes are removed without update of output weight in the OP-ELM and only a subset of principal components are employed as the final model selection in the PCA-ELM, respectively. In comparison with the ELM, the CP- and DP-ELM with significantly more compact structure still perform remarkably well, even though the ELM

TABLE II
PERFORMANCE COMPARISONS ON BENCHMARK MULTIPLE-OUTPUT REGRESSIONS

Datasets	Hidden node type	Algorithms	Testing RMSE		# Hidden nodes			Parsimony ratio $L/\bar{L}$		Training time (sec.)	Testing time (sec.)
			Mean	Dev.	Initial node $\bar{L}$	Final node L Mean	Dev.	Mean	Dev.		
Concrete slump	Sigmoid	ELM [1]	0.2325	0.0105	10	10	–	1.0000	–	0.0023	0.0010
		OP-ELM [10]	0.2649	0.0183	10	7.7000	2.5173	0.7700	0.2517	0.0215	0.0009
		PCA-ELM [17]	0.2454	–	7	5	–	0.7143	–	0.0041	**0.0003**
		CP-ELM	0.2324	0.0108	10	**5.8400**	1.2014	**0.5840**	0.1201	0.0115	0.0004
		DP-ELM	**0.2310**	0.0085	10	6.3400	0.8947	0.6340	0.0895	0.0359	0.0004
	RBF	ELM [1]	0.2567	0.0241	20	20	–	1.0000	–	0.0025	0.0019
		OP-ELM [10]	0.2635	0.0215	20	12.8000	3.3746	0.6400	0.1687	0.0293	0.0012
		CP-ELM	**0.2400**	0.0214	20	10.7800	1.6199	0.5390	0.0810	0.0718	0.0009
		DP-ELM	0.2437	0.0224	20	**10.7000**	1.2164	**0.5350**	0.0608	0.1747	**0.0008**
Abalone†	Sigmoid	ELM [1]	0.0555	0.0009	20	20	–	1.0000	–	0.0125	0.0100
		OP-ELM [10]	0.1103	0.0107	20	19.7500	1.1180	0.9875	0.0559	0.4196	0.0159
		PCA-ELM [17]	0.0577	–	6	5	–	0.8333	–	0.0045	**0.0028**
		CP-ELM	0.0555	0.0005	20	**9.2500**	0.9105	**0.4625**	0.0455	0.0725	0.0050
		DP-ELM	**0.0552**	0.0008	20	9.5500	1.3169	0.4775	0.0658	0.2145	0.0034
	RBF	ELM [1]	0.0575	0.0036	20	20	–	1.0000	–	0.0156	0.0094
		OP-ELM [10]	0.1126	0.0078	20	19.2500	1.8317	0.9625	0.0916	0.3947	0.0119
		CP-ELM	0.0573	0.0017	20	13.9500	1.1459	0.6975	0.0573	0.0811	0.0084
		DP-ELM	**0.0572**	0.0031	20	**13.1000**	1.1653	**0.6550**	0.0583	0.1786	**0.0037**
Wine (red)†	Sigmoid	ELM [1]	0.1141	0.0018	40	40	–	1.0000	–	0.0081	0.0072
		OP-ELM [10]	0.1563	0.0115	40	29.8000	4.6247	0.7450	0.1156	0.1872	0.0069
		PCA-ELM [17]	0.1419	–	10	7	–	0.7000	–	0.0032	**0.0034**
		CP-ELM	**0.1138**	0.0017	40	27.8000	1.6903	0.6950	0.0423	0.8586	0.0056
		DP-ELM	0.1141	0.0017	40	**25.2200**	1.6817	**0.6305**	0.0420	1.6792	0.0048
	RBF	ELM [1]	0.1195	0.0045	50	50	–	1.0000	–	0.0087	0.0115
		OP-ELM [10]	0.1590	0.0117	50	34.9000	4.4596	0.6980	0.1115	0.2365	0.0081
		CP-ELM	**0.1181**	0.0039	50	36.3200	2.1038	0.7264	0.0421	2.0530	0.0087
		DP-ELM	0.1193	0.0050	50	**33.7800**	1.7530	**0.6756**	0.0351	3.2851	**0.0072**
Wine (white)†	Sigmoid	ELM [1]	**0.1023**	0.0052	70	70	–	1.0000	–	0.0300	0.0499
		OP-ELM [10]	0.1823	0.0099	70	62.5000	6.1478	0.8929	0.0878	1.9993	0.0396
		PCA-ELM [17]	0.1479	–	10	7	–	0.7000	–	0.0046	**0.0037**
		CP-ELM	0.1028	0.0035	70	40.4000	2.5714	0.5771	0.0367	8.7348	0.0262
		DP-ELM	0.1032	0.0045	70	**36.0600**	1.9630	**0.4264**	0.0280	13.3562	0.0275
	RBF	ELM [1]	0.1551	0.1044	40	40	–	1.0000	–	0.0315	0.0296
		OP-ELM [10]	0.1893	0.0159	40	37.9000	2.6897	0.9475	0.0672	0.9476	0.0243
		CP-ELM	**0.1343**	0.0604	40	28.4600	2.6280	0.7115	0.0657	0.8839	0.0200
		DP-ELM	0.1467	0.0973	40	**26.9600**	2.0199	**0.6740**	0.0505	1.5784	**0.0190**

employs near optimal numbers of hidden nodes. In addition, the CP- and DP-ELM start from the upper triangular matrix **R** rather than the original hidden node output matrix **H**. Hence, the computational complexity relies on the number of hidden nodes $\bar{L}$ which is dramatically smaller than the observation number K. However, the OP- and PCA-ELM operate on **H** and **X**, and are sensitive to data size. From Tables II and III, it can also be observed that the CP- and DP-ELM have longer training time than the ELM, OP-ELM and PCA-ELM but achieving remarkable generalization and parsimonious structure simultaneously. Actually, if the number of initial hidden nodes is less than 50, the training time of the CP- and DP-ELM would be dramatically reduced. Since the motivation of this paper is to achieve a parsimonious SLFN using the CP- and DP-ELM with high generalization performance, the advantages of the proposed approaches outweigh the cost of longer training time. Furthermore, in terms of execution speed (i.e., testing time in Tables II and III), the CP- and DP-ELM uniformly achieve much faster responses in comparison with the ELM and OP-ELM.

All the aforementioned performance comparisons with ELM, OP-ELM and PCA-ELM have been validated by statistical ranking results shown in Table IV. Considering both Sigmoid and RBF hidden nodes, each algorithm runs 30 trials for individual average results of each data set. Next, the statistical ranking of each algorithm is determined by averaging the performance rank over all the data sets. In this context, the DP- and CP-ELM achieve the best performance in terms of parsimony, generalization and execution speed. In summary, the CP- and DP-ELM are able to strike an excellent balance between the high accuracy and parsimonious structure, thereby possessing superior generalization capability.

B. Nonlinear Time-Series Modeling

Consider a 2-D nonlinear time series [22]

$$\begin{aligned} y(t) = &\left[0.8 - 0.5\exp\left(-y^2(t-1)\right)\right] y(t-1) \\ &- \left[0.3 + 0.9\exp\left(-y^2(t-1)\right)\right] y(t-2) \\ &+ 0.1\sin\left(\pi\, y(t-1)\right) + e(t) \end{aligned} \tag{55}$$

where $e(t)$ is a zero mean white noise sequence with variance σ, i.e., $e(t) \sim N(0, \sigma)$. With initial conditions $y(0) = 0.1$ and $y(-1) = 0$, 1000 noisy samples are generated for training the SLFN and the noise-free limit cycle governed by (54) with $e(t) = 0$ is required to be modeled. According to

TABLE III
PERFORMANCE COMPARISONS ON BENCHMARK SINGLE-OUTPUT REGRESSIONS

Datasets	Hidden node type	Algorithms	Testing RMSE		# Hidden nodes			Parsimony ratio $L/\bar{L}$		Training time (sec.)	Testing time (sec.)
					Initial	Final node L					
			Mean	Dev.	node $\bar{L}$	Mean	Dev.	Mean	Dev.		
Auto MPG	Sigmoid	ELM [1]	0.0737	0.0037	30	30	–	1.0000	–	0.0016	0.0037
		OP-ELM [10]	0.1126	0.0131	30	16.5000	4.0066	0.5500	0.1336	0.0546	0.0028
		PCA-ELM [17]	0.0890	–	7	7	–	1.0000	–	0.0026	**0.0001**
		CP-ELM	**0.0735**	0.0030	30	19.4500	2.5644	0.6483	0.0855	0.2941	0.0029
		DP-ELM	0.0743	0.0040	30	**16.2000**	2.5874	**0.5400**	0.0862	0.6777	0.0022
	RBF	ELM [1]	0.0714	0.0052	40	40	–	1.0000	–	0.0100	0.0031
		OP-ELM [10]	0.1097	0.0153	40	**19.7500**	5.9549	**0.4938**	0.1489	0.0702	0.0028
		CP-ELM	**0.0711**	0.0041	40	26.5500	2.9105	0.6638	0.0728	0.8915	0.0031
		DP-ELM	0.0716	0.0047	40	22.8500	2.0333	0.5713	0.0508	1.8517	**0.0006**
Bank	Sigmoid	ELM [1]	**0.0472**	0.0004	100	100	–	1.0000	–	0.0796	0.0852
		OP-ELM [10]	0.1567	0.0016	100	81.5000	5.2967	0.8150	0.0530	9.1697	0.0693
		PCA-ELM [17]	0.1553	–	8	7	–	0.8750	–	0.0073	**0.0103**
		CP-ELM	*0.0473*	0.0005	100	66.2000	5.4324	0.6620	0.0543	44.8815	0.0534
		DP-ELM	*0.0473*	0.0004	100	**58.6000**	4.0332	**0.5860**	0.0403	51.3025	0.0568
	RBF	ELM [1]	**0.0491**	0.0024	100	100	–	1.0000	–	0.1326	0.0861
		OP-ELM [10]	0.1571	0.0020	100	77.5000	6.3465	0.7750	0.0635	9.0886	0.0705
		CP-ELM	*0.0493*	0.0024	100	72.5000	5.8737	0.7250	0.0587	47.1841	0.0677
		DP-ELM	*0.0493*	0.0024	100	**62.9000**	6.3675	**0.6290**	0.0637	48.9765	**0.0518**
Boston	Sigmoid	ELM [1]	0.1084	0.0077	50	50	–	1.0000	–	0.0039	0.0062
		OP-ELM [10]	0.1568	0.0110	50	28.2500	4.6665	0.5650	0.0933	0.0967	0.0031
		PCA-ELM [17]	0.1274	–	13	9	–	0.6923	–	0.0034	**0.0006**
		CP-ELM	**0.1074**	0.0047	50	30.1500	2.9069	0.6030	0.0581	2.0358	0.0019
		DP-ELM	0.1076	0.0059	50	**25.3000**	2.6378	**0.5060**	0.0528	4.1481	0.0022
	RBF	ELM [1]	0.1162	0.0120	80	80	–	1.0000	–	0.0172	0.0150
		OP-ELM [10]	0.1661	0.0157	80	41.0000	8.5224	0.5125	0.1065	0.1810	0.0053
		CP-ELM	**0.1108**	0.0116	80	47.2000	3.3811	0.5900	0.0423	15.3505	**0.0037**
		DP-ELM	0.1121	0.0101	80	**39.2000**	2.6675	**0.4900**	0.0333	22.8042	**0.0037**
California	Sigmoid	ELM [1]	0.1332	0.0012	80	80	–	1.0000	–	0.1014	0.2864
		OP-ELM [10]	0.1984	0.0105	80	59.0000	5.6765	0.7375	0.0710	7.6955	0.2278
		PCA-ELM [17]	0.1578	–	8	5	–	0.6250	–	0.0106	**0.0240**
		CP-ELM	**0.1331**	0.0007	80	44.9000	3.9567	0.5613	0.0495	15.9917	0.1794
		DP-ELM	0.1336	0.0007	80	**37.3000**	2.4518	**0.4662**	0.0306	22.3690	0.1554
	RBF	ELM [1]	**0.1301**	0.0012	100	100	–	1.0000	–	0.2808	0.3541
		OP-ELM [10]	0.2035	0.0092	100	74.0000	5.6765	0.7400	0.0568	11.0074	0.2730
		CP-ELM	**0.1301**	0.0011	100	61.4000	5.6608	0.6140	0.0566	47.4664	0.2072
		DP-ELM	0.1303	0.0008	100	**49.7000**	3.6530	**0.4970**	0.0365	57.4723	**0.1919**
Delta ailerons	Sigmoid	ELM [1]	0.0395	0.0002	70	70	–	1.0000	–	0.0343	0.0649
		OP-ELM [10]	0.0585	0.0023	70	41.5000	7.8351	0.5929	0.1119	2.5881	0.0343
		PCA-ELM [17]	0.0624	–	5	5	–	1.0000	–	0.0068	**0.0037**
		CP-ELM	**0.0393**	0.0001	70	37.3000	4.3729	0.5329	0.0625	8.6487	0.0293
		DP-ELM	0.0394	0.0001	70	**32.5000**	3.8658	**0.4643**	0.0552	0.1938	0.0278
	RBF	ELM [1]	0.0391	0.0002	90	90	–	1.0000	–	0.0796	0.0743
		OP-ELM [10]	0.0612	0.0037	90	52.5000	6.3465	0.5833	0.0705	3.8267	0.0421
		CP-ELM	**0.0390**	0.0001	90	56.1000	5.4863	0.6233	0.0610	29.1145	0.0480
		DP-ELM	0.0392	0.0002	90	**46.9000**	4.3321	**0.5211**	0.0481	35.1377	**0.0384**
Delta elevators	Sigmoid	ELM [1]	**0.0524**	0.0001	70	70	–	1.0000	–	0.0468	0.0774
		OP-ELM [10]	0.0650	0.0029	70	44.5000	3.6893	0.6357	0.0527	4.6005	0.0484
		PCA-ELM [17]	0.0572	–	6	6	–	1.0000	–	0.0074	**0.0059**
		CP-ELM	*0.0525*	0.0001	70	36.0000	4.8990	0.5143	0.0700	9.0184	0.0471
		DP-ELM	*0.0525*	0.0001	70	**30.6000**	3.0258	**0.4371**	0.0432	14.6547	0.0399
	RBF	ELM [1]	**0.0524**	0.0001	70	70	–	1.0000	–	0.0780	0.0792
		OP-ELM [10]	0.0650	0.0017	70	39.5000	7.2457	0.5643	0.1035	4.2463	0.0427
		CP-ELM	**0.0524**	8e-05	70	35.3000	4.6200	0.5043	0.0660	7.7454	0.0409
		DP-ELM	**0.0524**	9e-05	70	**29.5000**	2.6352	**0.4214**	0.0376	14.1961	**0.0324**
Servo	Sigmoid	ELM [1]	0.1339	0.0185	20	20	–	1.0000	–	0.0028	0.0005
		OP-ELM [10]	0.1641	0.0141	20	**10.6000**	3.4464	**0.5300**	0.1723	0.0250	0.0016
		PCA-ELM [17]	0.2278	–	4	4	–	1.0000	–	0.0011	**0.0001**
		CP-ELM	**0.1328**	0.0169	20	16.4800	1.2493	0.8240	0.0625	0.0827	0.0006
		DP-ELM	0.1338	0.0160	20	15.1600	1.2835	0.7580	0.0642	0.1154	0.0004
	RBF	ELM [1]	0.1236	0.0227	30	30	–	1.0000	–	0.0028	0.0037
		OP-ELM [10]	0.1663	0.0182	30	**12.9000**	4.5277	**0.4300**	0.1509	0.0440	**0.0020**
		CP-ELM	**0.1199**	0.0218	30	24.6400	1.7815	0.8213	0.0594	0.3413	0.0028
		DP-ELM	0.1232	0.0199	30	21.9400	1.5039	0.7313	0.0501	0.6047	0.0034

the ELM methodology, a SLFN using near optimal number of randomly generated hidden nodes is developed as the initial model for identification of the output $\hat{y}(t)$ with input $x(t) = [y(t-1), y(t-2)]^T$. To comprehensively evaluate the applications of CP- and DP-ELM to nonlinear time-series modeling, we consider three typical cases with different

TABLE IV

STATISTICAL RANKING COMPARISONS ON REAL-WORLD BENCHMARK REGRESSIONS

Node type	Algorithms	Testing RMSE		# Hidden node		Parsimony ratio		Training time		Testing time	
		Mean	Dev.	Mean	Dev.	Mean	Dev.	Mean	Dev.	Mean	Dev.
Sigmoid	ELM [1]	2.0909	0.8312	5.0000	0.0000	4.6364	0.5045	1.8182	0.4045	4.8182	0.4045
	OP-ELM [10]	4.8182	0.4045	3.6364	0.6742	3.0909	1.0445	3.2727	0.6467	4.0909	0.5394
	PCA-ELM [17]	4.1818	0.4045	**1.0000**	0.0000	3.5455	0.5222	**1.1818**	0.4045	**1.0000**	0.0000
	CP-ELM	**1.4545**	0.5222	3.0909	0.7006	2.0909	0.7006	3.8182	0.4045	*2.6364*	0.6742
	DP-ELM	2.0909	0.7006	2.2727	0.4671	**1.2727**	0.4671	4.9091	0.3015	*2.3636*	0.5045
RBF	ELM [1]	2.2727	0.9045	4.0000	0.0000	4.0000	0.0000	**1.0000**	0.0000	3.8192	0.4045
	OP-ELM [10]	4.0000	0.0000	2.3636	0.8090	2.3636	0.8090	2.2727	0.6467	2.6364	0.8090
	CP-ELM	**1.1818**	0.4045	2.4545	0.5222	2.4545	0.5222	2.8182	0.4045	2.1818	0.6030
	DP-ELM	2.0909	0.7006	**1.1818**	0.4045	**1.1818**	0.4045	3.9091	0.3015	**1.1818**	0.6030

TABLE V

PERFORMANCE COMPARISONS ON NONLINEAR TIME-SERIES IDENTIFICATION

Variances	Hidden node type	Algorithms	Testing RMSE		# Hidden nodes L			Training time (sec.)	Testing time (sec.)
					Initial	Final			
			Mean	Dev.	node	Mean	Dev.		
$\sigma = 0.01$	Sigmoid	ELM [1]	**0.0436**	0.0008	20	20	–	0.0037	0.0087
		OP-ELM [10]	0.0707	0.0181	20	19.4000	1.6413	0.1076	0.0044
		CP-ELM	0.0441	0.0009	20	16.0000	1.3248	0.0886	**0.0022**
		DP-ELM	0.0438	0.0009	20	**15.0400**	1.1599	0.2265	0.0025
	RBF	ELM [1]	0.0440	0.0004	30	30	–	0.0053	0.0109
		OP-ELM [10]	0.0559	0.0059	30	26.9000	3.3335	0.1494	0.0053
		CP-ELM	**0.0439**	0.0007	30	25.0600	1.6340	0.3176	**0.0031**
		DP-ELM	0.0440	0.0006	30	**22.3200**	1.6218	0.8006	**0.0031**
$\sigma = 0.05$	Sigmoid	ELM [1]	**0.0526**	0.0017	30	30	–	0.0019	0.0066
		OP-ELM [10]	0.0845	0.0185	30	23.9000	4.2003	0.1435	0.0066
		CP-ELM	0.0529	0.0020	30	25.9800	1.7201	0.2911	**0.0041**
		DP-ELM	0.0528	0.0017	30	**23.8200**	1.7343	0.7663	0.0053
	RBF	ELM [1]	0.0528	0.0019	30	30	–	0.0078	0.0087
		OP-ELM [10]	0.0864	0.0151	30	23.4000	3.9693	0.1432	0.0055
		CP-ELM	**0.0527**	0.0018	30	25.4000	2.1093	0.3092	**0.0034**
		DP-ELM	**0.0527**	0.0020	30	**23.1200**	1.4234	0.7778	0.0053
$\sigma = 0.1$	Sigmoid	ELM [1]	**0.0691**	0.0040	30	30	–	0.0062	0.0075
		OP-ELM [10]	0.0993	0.0159	30	**19.3000**	4.5187	0.1460	**0.0012**
		CP-ELM	0.0696	0.0039	30	26.2400	1.7907	0.3157	0.0037
		DP-ELM	0.0697	0.0042	30	23.9200	1.9042	0.7575	0.0018
	RBF	ELM [1]	0.0746	0.0052	20	20	–	0.0041	0.0047
		OP-ELM [10]	0.1150	0.0286	20	**15.9000**	3.4538	0.0955	0.0062
		CP-ELM	**0.0743**	0.0049	20	17.5000	1.1824	0.0817	0.0019
		DP-ELM	0.0744	0.0053	20	16.2400	1.2216	0.2181	**0.0012**

variances, i.e., σ=0.01, 0.05, and 0.1, for which the parameter ϵ is simply chosen as 0.001, 0.002, and 0.003, respectively.

In each case, both Sigmoid and RBF hidden nodes are considered. Compared with the ELM and OP-ELM, the results including identification accuracy, numbers of hidden nodes, training time, and testing time are listed in Table V, from which we can observe that the CP- and DP-ELM are remarkably superior to the ELM and OP-ELM in terms of parsimonious structure, excellent generalization accuracy and extremely fast execution speed. It should be noted that the CP- and DP-ELM with much more parsimonious structures achieve even better identification accuracy than the ELM using near optimal hidden nodes predetermined a priori. However, the OP-ELM leads to inferior identification even though more hidden nodes are recruited. For the highly noisy data (i.e., $\sigma = 0.1$), the CP- and DP-ELM still preserve optimal model selection with high accuracy while the OP-ELM tends to excessive pruning.

From the viewpoint of hidden node selection and removal in CP- and DP-ELM, an insight into model selection using Sigmoid hidden nodes for the case $\sigma = 0.1$ is shown in Fig. 2. The ARERs shown in Fig. 2(a) monotonously decrease to zero while the number of hidden nodes selected approaches to full candidate number $\bar{L}$. Once the termination criteria defined by ϵ are satisfied, the final network structures are determined

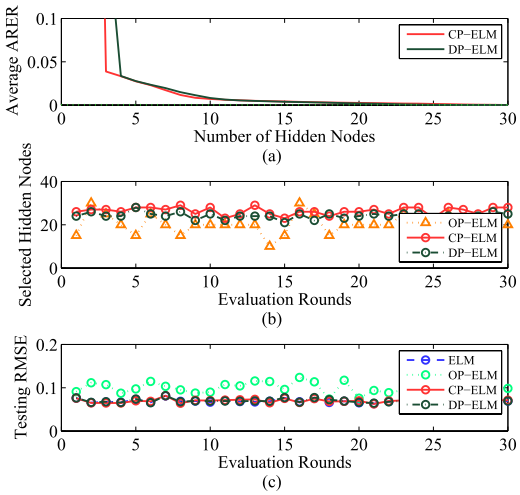

Fig. 2. Model selection of CP- and DP-ELM using Sigmoid hidden nodes for nonlinear time series ($\sigma = 0.1$). (a) Average ARER. (b) Selected hidden node numbers versus evaluation trials. (c) Testing RMSE versus evaluation trials.

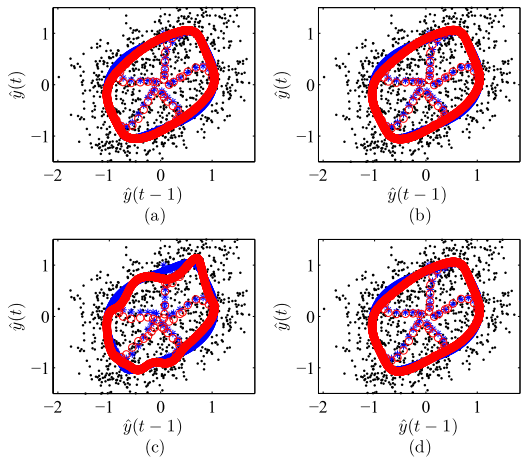

Fig. 3. Identification results of (a) CP-, (b) DP-, (c) OP-, and (d) ELM using Sigmoid hidden nodes for nonlinear time series with $\sigma = 0.1$ ($'\cdot'$, $'*'$, and $'\circ'$ denote noisy, noise-free, and identification points, respectively).

and shown in Fig. 2(b), whereby the CP- and DP-ELM are less sensitive to the initial hidden node randomness than the OP-ELM. Accordingly, the threshold ϵ helps to strike an excellent balance between the accuracy and compactness, and thereby contributing to remarkable generalization shown in Fig. 2(c), which illustrates uniformly superior identification performance. It also reveals that the CP- and DP-ELM using parsimonious hidden nodes are capable of identifying the essential nonlinearity of the time series (54), which suffers from highly noisy measurements shown in Fig. 3. The CP-ELM, DP-ELM, and ELM are able to reproduce the underlying nonlinearity of the time series while the OP-ELM has lost identification capability to some extent. Note that the CP-ELM can automatically select significant hidden nodes from the initial candidates, which contribute to the design of ELM-based SLFNs while the DP-ELM is also able to recursively remove insignificant ones, and thereby leaving the hidden nodes with high significance in the resulting SLFNs. With a compact network consisting of significant hidden nodes, the CP- and DP-ELM therefore contribute to superior identification and generalization by excluding redundant hidden nodes, which lead to overfitting noisy samples in the ELM. In this case, the CP- and DP-ELM achieve convincing performance in applications to time-series modeling and are remarkably superior to other methods like ELM and OP-ELM in terms of identification accuracy and optimal model selection.

V. Conclusion

In this paper, novel CP- and DP-ELM algorithms for training multiple-output SLFNs using ROLS have been proposed. In the initial step, $\bar{L}$ candidate hidden nodes are randomly generated by the ELM technique, and recursively orthogonally transformed to an upper triangular matrix $\mathbf{R}$, whereby the dimension is significantly less than that of the hidden output matrix $\mathbf{H}$. It should be noted that the foregoing ROLS method is efficiently implemented by our proposed SPO, which can dramatically reduce computational burden. Starting from the matrix $\mathbf{R}$, the CP- and DP-ELM have been proposed to realize regressor selection and removal according to regressor significance defined by the ARERR and AREGR, respectively. Furthermore, as the approximation criterion, the ARER is used to determine the SLFN structure complexity. For recursive selection or removal of regressors, retriangularizations based on CSPO and DSPO are proposed to recursively carry out on the matrix of selected regressors together with candidates. As the final step of CP- and DP-ELM, the output weight matrix $\mathbf{\Theta}$ is quickly solved from the upper triangular equation $\mathbf{R\Theta} = \mathbf{T}$ by backward substitution. To validate the effectiveness and superiority of the proposed CP- and DP-ELM, simulation studies and comprehensive comparisons with ELM, OP-ELM, and PCA-ELM methods are carried out on not only multiple- and single-output benchmark regression data sets but also applications to nonlinear time-series modeling. Simulation results demonstrate that the CP- and DP-ELM are able to generate more parsimonious SLFNs with high generalization and identification accuracy from real-world and noisy data samples.

Acknowledgment

The authors would like to thank the Editor-in-Chief, Associate Editor and anonymous referees for their invaluable comments in enhancing the quality of this paper.

References

[1] 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.

[2] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 879–892, Jul. 2006.

[3] G.-B. Huang, D.-H. Wang, and Y. Lan, "Extreme learning machines: A survey," *Int. J. Mach. Learn. Cybern.*, vol. 2, no. 2, pp. 107–122, Jun. 2011.

[4] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.

[5] 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.

[6] N. Wang, M. J. Er, and X.-Y. Meng, "A fast and accurate online self-organizing scheme for parsimonious fuzzy neural networks," *Neurocomputing*, vol. 72, nos. 16–18, pp. 3818–3829, Oct. 2009.

[7] N. Wang, M. J. Er, X.-Y. Meng, and X. Li, "An online self-organizing scheme for parsimonious and accurate fuzzy neural networks," *Int. J. Neural Syst.*, vol. 20, no. 5, pp. 389–405, Oct. 2010.

[8] N. Wang, "A generalized ellipsoidal basis function based online self-constructing fuzzy neural network," *Neural Process. Lett.*, vol. 34, no. 1, pp. 13–37, Aug. 2011.

[9] H.-J. Rong, Y.-S. Ong, A.-H. Tan, and Z. Zhu, "A fast pruned-extreme learning machine for classification problem," *Neurocomputing*, vol. 72, nos. 1–3, pp. 359–366, Dec. 2008.

[10] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, "OP-ELM: Optimally pruned extreme learning machine," *IEEE Trans. Neural Netw.*, vol. 21, no. 1, pp. 158–162, Jan. 2010.

[11] T. Similä and J. Tikka, "Multiresponse sparse regression with application to multidimensional scaling," in *Proc. 15th ICANN*, Warsaw, Poland, Sep. 2005, pp. 97–102.

[12] 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.

[13] G.-B. Huang and L. Chen, "Convex incremental extreme learning machine," *Neurocomputing*, vol. 70, nos. 16–18, pp. 3056–3062, Oct. 2007.
[14] G. Feng, G.-B. Huang, Q. Lin, and R. Gay, "Error minimized extreme learning machine with growth of hidden nodes and incremental learning," *IEEE Trans. Neural Netw.*, vol. 20, no. 8, pp. 1352–1357, Aug. 2009.
[15] Y. Lan, Y. C. Soh, and G.-B. Huang, "Constructive hidden hodes selection of extreme learning machine for regression," *Neurocomputing*, vol. 73, nos. 16–18, pp. 3191–3199, Oct. 2010.
[16] N. Wang, M. Han, N. Dong, and M. J. Er. Constructive multi-output extreme learning machine with application to large tanker motion dynamics identification. *Neurocomputing* [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2013.01.062
[17] A. Castaño, F. Fernández-Navarro, and C. Hervás-Martínez, "PCA-ELM: A robust and pruned extreme learning machine approach based on principal component analysis," *Neural Process. Lett.*, vol. 37, no. 3, pp. 377–392, Jun. 2013.
[18] J. Stark, "Adaptive model selection using orthogonal least squares methods," *Proc. R. Soc. London A, Math., Phys. Eng. Sci.*, vol. 453, no. 1956, pp. 21–42, Jan. 1997.
[19] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification," *Int. J. Control*, vol. 50, no. 5, pp. 1873–1896, 1989.
[20] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.
[21] S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset model selection," *IEEE Trans. Signal Process.*, vol. 43, no. 7, pp. 1713–1715, Jul. 1995.
[22] S. Chen, P. M. Grant, and C. F. N. Cowan, "Orthogonal least-squares algorithm for training radial basis function networks," *IEE Proc., F*, vol. 139, no. 6, pp. 378–384, 1992.
[23] K. Z. Mao, "Fast orthogonal forward selection algorithm for feature subset selection," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1218–1224, Sep. 2002.
[24] S. Chen, "Local regularization assisted orthogonal least squares regression," *Neurocomputing*, vol. 69, nos. 4–6, pp. 559–585, Jan. 2006.
[25] S. Chen, X. Hong, B. L. Luk, and C. J. Harris, "Orthogonal-least-squares regression: A unified approach for data modelling," *Neurocomputing*, vol. 72, nos. 10–12, pp. 2670–2681, Jun. 2009.
[26] W. Luo, S. A. Billings, and K. M. Tsang, "On-line structure detection and parameter estimation with exponential windowing for nonlinear systems," *Eur. J. Control*, vol. 2, no. 4, pp. 291–304, 1996.
[27] X. Hong and S. A. Billings, "Givens rotation based fast backward elimination algorithm for RBF neural network pruning," *IEE Proc. Control Theory Appl.*, vol. 144, no. 5, pp. 381–384, Sep. 1997.
[28] J. E. Bobrow and W. Murray, "An algorithm for RLS identification of parameters that vary quickly with time," *IEEE Trans. Autom. Control*, vol. 38, no. 2, pp. 351–354, Feb. 1993.
[29] J. E. Bobrow and W. Murray, "Adaptive model selection for polynomial NARX models," *IET Control Theory Appl.*, vol. 4, no. 12, pp. 2693–2706, Dec. 2010.
[30] D. L. Yu, J. B. Gomm, and D. Williams, "A recursive orthogonal least squares algorithm for training RBF networks," *Neural Process. Lett.*, vol. 5, no. 3, pp. 167–176, Jun. 1997.
[31] J. B. Gomm and D. L. Yu, "Selecting radial basis function network centers with recursive orthogonal least squares training," *IEEE Trans. Neural Netw.*, vol. 11, no. 2, pp. 306–314, Mar. 2000.
[32] D. L. Yu, D. W. Yu, and J. B. Gomm, "Neural model adaptation and predictive control of a chemical process rig," *IEEE Trans. Control Syst. Technol.*, vol. 14, no. 5, pp. 828–840, Sep. 2006.
[33] D. S. Huang and J. X. Du, "A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2099–2115, Dec. 2008.
[34] B. Dumitrescu, A. Onose, P. Helin, and I. Tăbus, "Greedy sparse RLS," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2194–2207, May 2012.
[35] C. L. Blake and C. J. Merz, "UCI repository of machine learning databases," Ph.D. dissertation, Dept. Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 1998.
[36] M. Mike, "Statistical datasets," Ph.D. dissertation, Dept. Statist., Univ. Carnegie Mellon, Pittsburgh, PA, USA, 1989.

Ning Wang (S'08–M'12) received the B.Eng. degree in marine engineering and the Ph.D. degree in control theory and engineering from Dalian Maritime University (DMU), Dalian, China, in 2004 and 2009, respectively.

He was a Joint Training Ph.D. Student with Nanyang Technological University, Singapore, from 2008 to 2009, which was financially supported by the China Scholarship Council. He is currently an Associate Professor with the Marine Engineering College, DMU. His current research interests include artificial neural networks, fuzzy systems, machine learning, ship intelligent control, and dynamic ship navigational safety assessment.

Dr. Wang received the Nomination Award of Liaoning Province Excellent Doctoral Dissertation, the DMU Excellent Doctoral Dissertation Award, the DMU Outstanding Ph.D. Student Award in 2010, the Liaoning Province Award for Technological Invention and the Honour of Liaoning BaiQianWan Talents, the Liaoning Excellent Talents and Dalian Leading Talents, and the Excellent Government-Funded Scholars and Students Award in 2009.

Meng Joo Er (S'82–M'87–SM'07) is currently a Full Professor of electrical and electronic engineering with Nanyang Technological University, Singapore. He has authored five books, 16 book chapters, and more than 500 refereed journal and conference papers. His current research interests include intelligent control theory and applications, computational intelligence, robotics and automation, sensor networks, biomedical engineering, and cognitive science.

He received the Institution of Engineers, Singapore (IES) Prestigious Engineering Achievement Award for his research project entitled "Development of Intelligent Techniques for Modeling, Controlling and Optimizing Complex Manufacturing Systems" in 2011. He is the only dual winner of the Singapore IES Prestigious Publication Award in Application in 1996 and the IES Prestigious Publication Award in Theory in 2001. He received the Teacher of the Year Award from the School of Electrical and Electronic Engineering in 1999, the School of Electrical and Electronic Engineering Year Two Teaching Excellence Award in 2008, and the Most Zealous Professor of the Year Award in 2009. He received the Best Session Presentation Award at the World Congress on Computational Intelligence in 2006 and the Best Presentation Award at the International Symposium on Extreme Learning Machine in 2012. He was a Chairman of the IEEE Computational Intelligence Society (CIS) Singapore Chapter from 2009 to 2011 and the Singapore Chapter won the CIS Outstanding Chapter Award in 2012. He was bestowed the IEEE Outstanding Volunteer Award (Singapore Section) and the IES Silver Medal in 2011. He has more than 40 awards at international and local competitions. Currently, he serves as an Editor-in-Chief of the *International Journal of Electrical and Electronic Engineering and Telecommunications*, an Area Editor of the *International Journal of Intelligent Systems Science*, an Associate Editor of 11 refereed international journals, including the IEEE TRANSACTION ON FUZZY SYSTEMS, and an Editorial Board Member of the *Electronic Engineering Times*. He was invited to deliver more than 60 keynote speeches and invited talks overseas. Due to his outstanding achievements in education, research, administration and professional services, he is listed in Who's Who in Engineering, Singapore Edition 2013.

Min Han (M'95–A'03–SM'06) received the B.S. and M.S. degrees from the Department of Electrical Engineering, Dalian University of Technology, Dalian, China, and the M.S. and Ph.D. degrees from Kyushu University, Fukuoka, Japan, in 1982, 1993, 1996, and 1999, respectively.

She is a Professor with the Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology. Her current research interests include neural networks and chaos and their applications to control and identification.