IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 23, Issue 6, Ser. I (Nov. –Dec. 2021), PP 01-12
www.iosrjournals.org

An Improved Extreme Learning Machine for Process Modeling

Hend Liouane[1], Okba Taouali[2]
[1]*National School of Engineering Monastir,Tunisia*
[2]*Department of Computer Engineering, Faculty of Computers and Information Technology, University of Tabuk, Tabuk, Saudi Arabia*

Abstract: *This paper presents an interesting methodology for a class of nonlinear system identification via the recurrent Extreme Learning Machine (ELM). The recurrent ELM has been used for training single hidden layer feed-forward neural networks. Vis-à-vis to various feed-forward NN the ELM is remarkably efficient and tends to reach global optimum with high convergence speed. For many real applications, in control system of view point can be represented by nonlinear autoregressive with external input (NARX) models. The recurrent ELM presents efficient tools for working with dynamical recurrent data. Recurrent Neural Network (RNN) is an adequate structure for NARX models and can be trained in online context. The proposed ELM structure algorithm offers the advantages of using RNN structure and ELM learning algorithm. Then, it overcomes the insufficiency derived from conventional gradient type based algorithms. A highly accurate, low time convergence can be obtained with recurrent ELM algorithm. The proposed method has been tested to model a Wiener-Hammerstein benchmark and the tank process and the results are satisfactory*

Key Word*: adaptive identification, extreme learning machine, recurrent neural network, process modeling.*

Date of Submission: 04-11-2021 Date of Acceptance: 18-11-2021

I. Introduction

Practically, the machine learning concepts play a preponderant role in many computer science technologies and especially in online identification process domain [25-27]. In fact, identification process via, neural network paradigm is represented by a set of processing units, organized in different modes to form the network structure that model's the dynamics of system identification. Every processing unit compute a local performance based on inputs from its neighbors and delivers a resulting output. The global results of the NN are derived from the interaction between the processing units called neurons. Many neural network structure have been proposed in the field of learning computing and the widely used models is the Single-Layer Feed Forward Networks (SLFN) which has good learning ability and remarkable simplicity of implementation. Furthermore, the SLFN has been most used in many fields of research and application and the conventional training algorithms such as back-propagation algorithm (BP) and the Levenberg-Marquardt scheme (LM) have been used successfully in training neural networks but this algorithm is relatively slow in learning process and includes some inherent imperfections and imprecision. In fact, several iterations are needed in the gradient descent technique in order to converge by adjusting the weights and bias. Then, the training process takes a computational effort. Moreover, by the uses of the gradient descent algorithm, it is simple to fall into local minima and the performance of the NN is very sensitive to the optimized learning parameters.

Recently, simple in its principle, Random Projection concept has gained a lot of popularity, especially in the area of neural networks learning machine. The best known method is the Extreme Learning Machine (ELM) [28,29] for feed forward neural networks [9]. In fact, thanks to random projection concepts and the Moore-Penrose generalized inverse matrix, Extreme Learning Machine process becomes a popular training algorithm and gaining significant interests during recent years due to its simple structure and computational process, single hidden layer and single linear output layer. Initially proposed in by Wang [8], the main idea of the ELM [30,31] is presented by the exploitation of the SLFNs, firstly the input weights and hidden biases are randomly generated, secondly, the output weights are analytically determined using the least-square method. The output weights obtained using the Moore-Penrose generalize inverse matrix. Indeed, the ELM facilitates the SLFNs implementation and allows a significant training time reduction. Moreover, there has been a lot of interest in extreme learning machine domain and many variant of the ELM concepts have been proposed to improve the efficiency given by the learning algorithm. Initially, the ELM neural network exploits the similar activation function for hidden nodes, but the ELMs have been generalized to involve additional activation functions that are not neuron alike [11]. The Incremental Extreme Learning Machine algorithms (I-ELM [5], EI-

ELM [6] and EM-ELM [7]) can randomly increment the number of the hidden neurons and successively optimize the hidden-layer characteristics such as the number of neurons and the activation functions. Including the various extensions, the online sequential learning algorithm based on ELM (OS-ELM) gained popularity, according to which learning is performed by treating the data one-by-one or chunk-by-chunk with the regularization of the chunk size. The output weights are analytically updated based on the sequentially arriving data and the recursive least square algorithm. [20].

Moreover, based on the artificial intelligence and the global optimization tools, many approaches have been applied to improve the ELM robustness and efficiency. Suresh et al [17], propose the exploitation of the genetic algorithm mataheuristics called RCGA-ELM for characterizing the optimal number of hidden nodes, input weights and bias values. Han et al [15, 16], exploit the PSO technique for the same problem in SLFNs neural network, Cao et al, [10] propose a dynamic evolutionary Extreme Learning Machine algorithm called self-adaptive ELM which learning process is performed by the Differential Evolution algorithm for global optimization of the hidden network nodes. However, in control system engineering domain the ELM can be applied for system identification in discrete time by using a series-parallel formulation model [3]. Indeed, a generic nonlinear identification using the nonlinear auto regressive model with exogenous input (NARX) is considered. From the viewpoint of the industry and real-world applications, there is a need to identify a non-linear system to be controlled. Nonlinear system with an exogenous input control signal can be represented by NARX models. Recurrent neural network is an adequate structure for NARX models [12].

In this paper, we use the ELM based on recursive least square algorithm to learn a recurrent neural network in online context. The recurrent learning ELM in online mode, the present RNN model are used in order to identify the behavior of a nonlinear system described by the Input Output Data description.

The paper is organized as follows: section II presents the output error or parallel structural identification principle. Section III describes the proposed adaptive online extreme learning machine algorithm. Training results with the considered examples and performances comparison are presented in section IV. Section V, a conclusion concludes is the paper.

II. Identification problem by ELM

System identification is qualified by the most important field in applied science and technology, especially in control systems, the identification problems is to determine a mathematical model that can rigorously approximates the output of an unknown system described by the inputs and outputs data. In fact, most proposed methods exploit a collection of inputs outputs data to tune parameters of the model according the desired accuracy, followed by a validation procedure of model system. Many of this model structure are based on time invariant model. In practice, most systems exhibit certain nonlinear characteristics and cannot be described as linear systems. In such situations, nonlinear system identification tools are needed in order to yield more accurate modeling capabilities. In variety domain, the identification process is implemented as a neural network and the identification process is the training of the network. In order to identify a nonlinear system, presented in black box form, the output error identification approach, which is well-known in the identification field, can be applied [18].

The output error identification approach is represented in Figure (1).

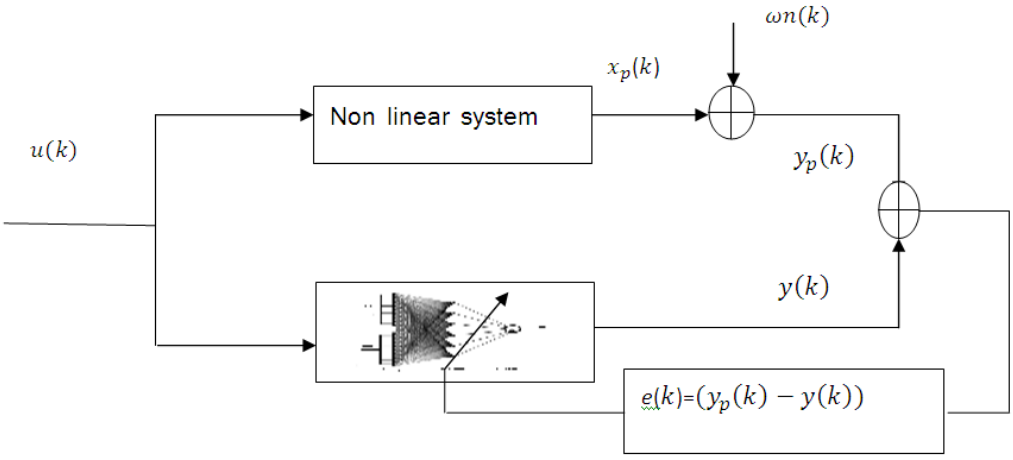

Figure 1: Output error identification approach based NN

The nonlinear system can be described by the following state representation

$$\begin{cases} x_p(k+1) = \Phi\left[X_p(k)U(k)\right] \\ y_p(k) = x_p(k) + \omega \mathrm{n}(k) \end{cases} \quad (1)$$

where

$$\begin{cases} X_p(k) = [x_p(k) \quad \dots \quad x_p(k-d_y+1)] \\ U(k) = [u(k) \quad \dots \quad u(k-d_u+1)] \end{cases} \quad (2)$$

Φ : is a nonlinear function
ωn : is a sequence of additive white noise.

The RNN or NARX model can be represented by the following equation:

$$y(k+1) = f\left[y(k) \dots y\left(k-d_y\right), u(k) \dots u(k-d_u)\right] \quad (3)$$

The associated structure neural network with $d_u$and d_y are the delayed input output order respectively, is recurrent of d_y order. $k$is the time index sample and f is an unknown nonlinear function.
The predictor neural network with $y(k)$ as the output has to imitate the process behavior such that:

$$y_p(k) - y(k) = e(k) \quad (4)$$

$e(k)$ is the output error which should tends to zero when the process is well-identified.
A set of measurements can be carried out on a nonlinear system $(\mathrm{u(k.T)}, \mathrm{y_p(k.T)})$ with T is the sampling rate. From this data, a predictor model would be derived, whose dynamical behavior should be as close to the process as possible.

II.1 Basic ELM algorithm

ELM can construct different learning algorithms through choosing different types and numbers of activation functions with a feed-forward structure as shown in Figure 2. After the input weights and the hidden neurons biases are randomly projected, the ELM can be simply considered as a linear system and the output weights of ELM can be analytically determined through a simple generalized inverse operation of the hidden layer outputs matrices.

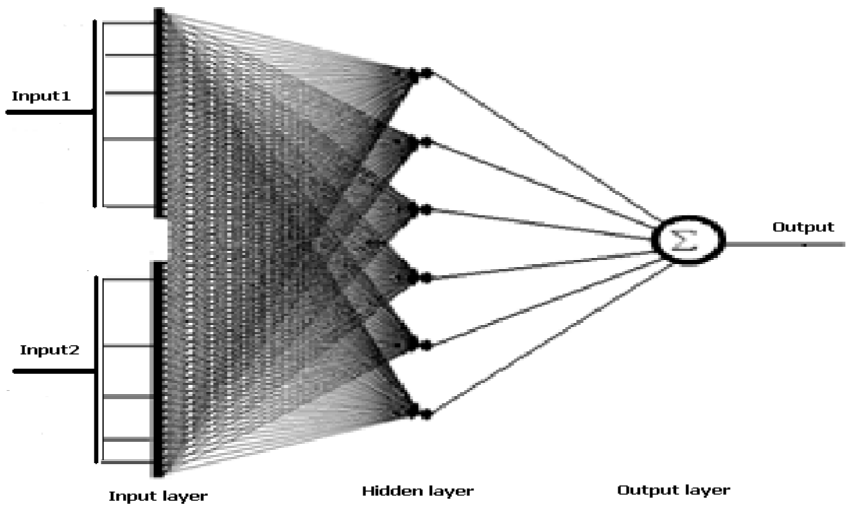

Figure 2: Feed forward network structure

The output of ELM with m hidden neurons, N inputs and L outputs are mathematically modeled as

$$y_l = \sum_{j=1}^{m} \beta_{j,l}\, g\left(\sum_{i=1}^{N} w_{i,j}\, x_i + b_j\right) \text{ with } l = 1 \dots . L$$

Where y_l is the l th output of the network , x_i is the ith input, $\beta_{j,l}$ is the output weight between the j hidden node and the l output, g(.) is the activation function of the hidden neurons,$w_{i,j}$ is the input weight and b_j is the bias of the neuron j in the hidden layer. If the activation function is infinitely differentiable, then the input weights and biases could be assigned randomly. Thus, the input weights and biases do not need to be adjusted in

the training process. The output of the hidden neurons can be computed after these parameters are randomly fixed. The ELM learning process consists in obtaining the least square solution $\hat{\beta}$ of linear system $Y = H\beta$ where Y is the output neural network matrix, β is the weights of the connections between the hidden layer and the output layer and H is the output hidden layer matrix, which can be expressed as follows:

$$H(w_{i,j}, b_j, x_i) = \begin{bmatrix} g(w_{1,1}x_1 + b_1) & \dots & g(w_{1,m}x_1 + b_m) \\ \vdots & \ddots & \vdots \\ g(w_{N,1}x_N + b_1) & \dots & g(w_{N,m}x_N + b_m) \end{bmatrix}$$

If the number m of hidden neurons is equal to the number N of distinct training samples (m=N), H is a square matrix, then invertible and ELM can approximate these training samples. However, in most cases, the number of hidden neurons is far less than the number of distinct training samples (m<N), H is non square matrix and the smallest norm least square solution can be solved by$\hat{\beta} = H^{+}HY$ where H^{+} is the Moore-Penrose generalized inverse matrix. As it is shown, ELM is formulated as a linear in parameter yielding to solve a linear system. Compared to conventional FFNN learning methods, ELM which is based on least squares, is efficient and tends to reach a global optimum with high speed convergence [8-9].

II.2 Adaptive extreme learning machine(A-ELM)

To achieve a desirable set of synaptic weights to predefined network architecture, a training process is needed. The proposed A-ELM can be used as a training algorithm for a recurrent neural network in the same frame-work as feed-forward network.

A-ELM consists in training the network permanently during operation thanks to recursive ELM procedure and the RLS algorithm. The output of the feed-forward network is delayed and feedback to the input of the neural network as shown in Figure (3).

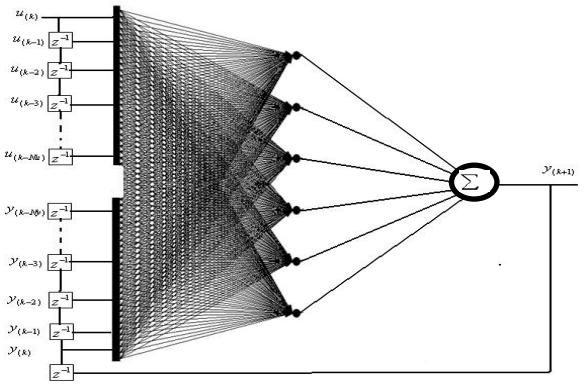

Figure 3. Recurrent neural networks architectural

with $U(k) = [u(k) \dots\dots u(k - d_u + 1)]$; $Y(k) = [y(k) \dots\dots\dots y(k - d_y + 1)]$

The output at time $t = kT$ is defined as$y(t = kT) = y(k)$. Therefore the network output can be represented as follows:

$$y(k+1) = f\left[y(k), y(k-1) \dots\dots, y\left(k - d_y + 1\right), u(k), \dots\dots\dots u(k - d_u + 1)\right]$$

$$y(k+1) = \sum_{j=1}^{\mathrm{n_h}} \beta_j(k+1)\, g\left[\left(\sum_{i=0}^{d_u - 1} w_{i,j}u(k-i)\right) + \left(\sum_{i=0}^{d_y - 1} w_{i,j}y(k-i)\right) + b_j\right] \quad (5)$$

The activation function g represents a feed-forward network non-linear function and the external variable $u(k)$ represents the network input at a defined time t = kT. $w_{i,j}$and b_j are the learning parameters of the hidden nodes.

A-ELM consists in continuously calculating the different coefficients of the recurrent network, by using ELM. The latter involves the data pertaining to a time window of a finite length which shifts in time (sliding window) and thus, the coefficients are updated with the following equations:

$$H(k)\beta(k) = y_p(k) \quad (6)$$

where

$$H(k) = \begin{bmatrix} g(\sum_{i=1}^{d_u} w_{i,1}\, u(k-i) + \sum_{i=1}^{d_y} w_{i,1}\, y(k-i) + b_1) \\ \vdots \\ \vdots \\ g(\sum_{i=1}^{d_u} w_{i,n_h}\, u(k-i) + \sum_{i=1}^{d_y} w_{i,n_h}\, y(k-i) + b_{n_h}) \end{bmatrix}^T \quad (7)$$

$$= \begin{bmatrix} H_1(k) \\ \vdots \\ \vdots \\ H_{n_h}(k) \end{bmatrix}^T$$

$H(k)$is the hidden layer output vector, with$n_h > (d_u + d_y)$ hidden nodes number, at time k. T
The output weight vector $\beta(k) = [\beta_1 \ldots \ldots \beta_{n_h}]^T$ is then obtained by the least squares solution:

$$\hat{\beta}(k) = [H(k)^T H(k)]^{-1} H(k)^T Y(k) \quad (8)$$

Generally, the matrix inversion lemma is very important tools to convert least square algorithm into recursive least square algorithm. Indeed, a generalized recursive algorithm for updating the least-squares solution can be computed as follows.

II.3 Recursive Least-Squares Techniques in ELM

Suppose now that we are given another data at k+1 instant then $H(k+1)$

$$\hat{\beta}(k+1) = [H(k+1)^T H(k+1)]^{-1} H(k+1)^T Y(k+1)$$

Let:

$$\hat{\beta}(k+1) = \hat{\beta}(k) + K(k+1) * \mathcal{E}(k+1)$$

where $\hat{\beta}(k)$ is a vector of model parameters to estimate
$\mathcal{E}(k)$ is the difference between the measured output and the estimated output at time k
$K(k)$ is the scaling factor - sometimes known as the Kalman Gain

Suppose we have all the data collected up to time k. Then define the formation of the matrix at time k as

$$\Psi_k^{-1} = H_k^T H_k$$

$$H_{k+1}^T H_{k+1} = \begin{bmatrix} H_k & h_{k+1} \end{bmatrix}^T \times \begin{bmatrix} H_k \\ h_{k+1} \end{bmatrix}$$

$$= H_k^T H_k + h_{(k+1)}^T h_{(k+1)}$$

$$\Psi_{k+1}^{-1} = \Psi_k^{-1} + h_{(k+1)}^T h_{(k+1)}$$

$$H_{k+1}^T Y_{k+1} = \begin{bmatrix} H_k & h_{k+1} \end{bmatrix}^T \times \begin{bmatrix} y_k \\ y_{k+1} \end{bmatrix} = H_k^T y_k + h_{k+1} y_{k+1}$$

$$\beta_{k+1} = \left(H_{k+1}^T H_{k+1}\right)^{-1} H_{k+1} Y_{k+1} = \Psi_{k+1}\left(H_k^T y_k + h_{k+1} y_{k+1}\right)$$

Because $\beta_k = \Psi_k\left(H_k^T Y_k\right)$, $\Psi_k^{-1} \beta_k = \left(H_k^T y_k\right)$

OR

$$\Psi_{k+1}^{-1} = \Psi_k^{-1} + h_{k+1} h_{k+1}^T \,,\ \Psi_k^{-1} = \Psi_{k+1}^{-1} - h_{k+1} h_{k+1}^T$$

then

$$\beta_{k+1} = \Psi_{k+1}\left(\Psi_{k+1}^{-1} \beta_k - h_{k+1} h_{k+1}^T \beta_k + h_{k+1} y_{k+1}\right)$$

$$\beta_{k+1} = \beta_k + \Psi_{k+1} h_{k+1} (y_{k+1} - h_{k+1}^T \beta_k)$$

$$\varepsilon_n = y_{k+1} - h_{k+1}^T \beta_k$$

$$\beta_{k+1} = \beta_k + K_{k+1} \varepsilon_{k+1}$$

$$K_{k+1} = \Psi_{k+1} h_{k+1}$$

$$\Psi_{k+1} = \Psi_k - \frac{\Psi_k h_{k+1} h_{k+1}^T \Psi_k}{1 + h_{k+1}^T \Psi_k h_{k+1}}$$

Ψ is proportional to the covariance matrix of the estimate, and is thus called the covariance matrix.
The algorithm has to be initialized with β0 =0 and Ψ0. Generally, Ψ(0) is initialized as α*Id where Id is the identity matrix and α is a large positive number. The larger α, the less confidence is put in the initial estimate β0
By introducing of the forgetting factor and if we would like to modify the recursive least squares ELM algorithm so that older data has less effect on the coefficient estimation of the ELM output. This could be done by biasing the objective of the squared error function that we are trying to minimise (i.e. the squared error)

$$J_{\beta(k+1)} = \sum_{i=1}^{k+1} \lambda^{k-i} \left[y_{k+1} - h_{k+1}^T \beta_k \right]^2$$

This same weighting function when used on an ARX model can be used to bias the calculation of the covariance $\Psi_k^{-1} = H_k^T H_k$ matrix giving more recent values greater prominence, as follows.

$$\Psi_k^{-1} = H_k^T H_k$$

$$\Psi_k^{-1} = \sum_{i=1}^{k} \lambda^{k-i} h_i^T h_i$$

Where λ is chosen to be between 0 and 1. Typically λ=[0.95….0.99].
RLS Algorithm with forgetting factor

$$\varepsilon_n = y_{k+1} - h_{k+1}^T \beta_k$$

$$\beta_{k+1} = \beta_k + K_{k+1} \varepsilon_{k+1}$$

$$K_{k+1} = \Psi_{k+1} h_{k+1}$$

$$\Psi_{k+1} = \frac{1}{\lambda} \left(\Psi_k - \frac{\Psi_k h_{k+1} h_{k+1}^T \Psi_k}{\lambda + h_{k+1}^T \Psi_k h_{k+1}} \right)$$

II.4 The proposed Algorithm
The AO-ELM contains two phases
Phase 1: Initialization

- Set the RNN structure
- Set the number of nodes$n_h$in the hidden layer.
- Generate random weights and bias

Phase 2: Training
For each sample u(k), y(k)

- Set up input vector
- Compute $H(k+1)$: output hidden layer
- Compute $\beta(k+1)$ by :

$$\varepsilon_n = y_{k+1} - h_{k+1}^T \beta_k$$

$$P_{k+1} = P_k - \frac{P_k h_{k+1} h_{k+1}^T P_k}{1 + h_{k+1}^T P_k h_{k+1}}$$

$$K_{k+1} = P_{k+1} h_{k+1}$$

$$\beta_{k+1} = \beta_k + K_{k+1} \varepsilon_{k+1}$$

End For

III. Simulation Results

In this section, the performance of the adaptive ELM algorithm is evaluated vis-à-vis the nonlinear system identification algorithm based on sliding-window kernel RLS algorithm "SWKRLS ",and the kernel recursive least squares algorithm "RKLS " [3, 21, 23].

The proposed algorithm has been tested for modeling a Wiener Hammerstein benchmark and a tank process. Simulation results are given by only the input u(k) and the output y(k) are observable and the input output system are the y(k),y(k-1),y(k-2), y(k-3) and the u(k). For the simulation process, initially the empty database is used with y(k-1)=y(k-2)= y(k-3)=0. Indeed, the sigmoid activation function in the hidden layer neurons is used. The number of hidden neurons is equal to number of input of the NN, nh=5.

III.1 Winner Hammerstein Benchmark

The system to be modeled is sketched by Figure 4. It consists of an electronic nonlinear system with a Wiener Hammerstein structure that was built by Gerd Vandersteen [24]. This process was adopted as a nonlinear system benchmark in SYSID 2009 and it represents a challenge to identify using the kernel methods.

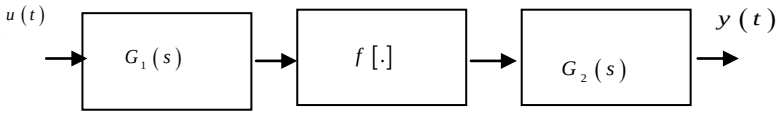

Figure 4. Wiener Hammerstein benchmark

For the Wiener Hammerstein model we perform the identification processes for 167709 time steps. In Figures 5, 6, 7 and 8, the Adaptive A-ELM outputs as well as the system output, the evolution of the Mean Square Error (MSE) and the weight beta evolution and its convergence are represented. A great similarity between the model output and the system output is detected and the Normalized Mean Square Error is equal to 0.0079 with 0.007sec a computation time for each sample.

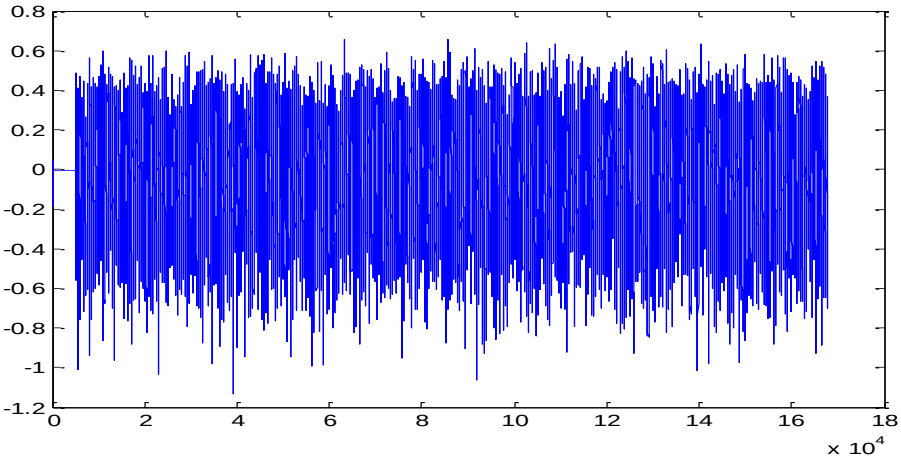

Figure 5. System and model A-ELM outputs during the online identification.

The performance of our algorithm is picked in the training sample window [60000 60500]. The main remark is that the model output and the system output have an important similarity. This shows the good performances of the proposed online identification method.

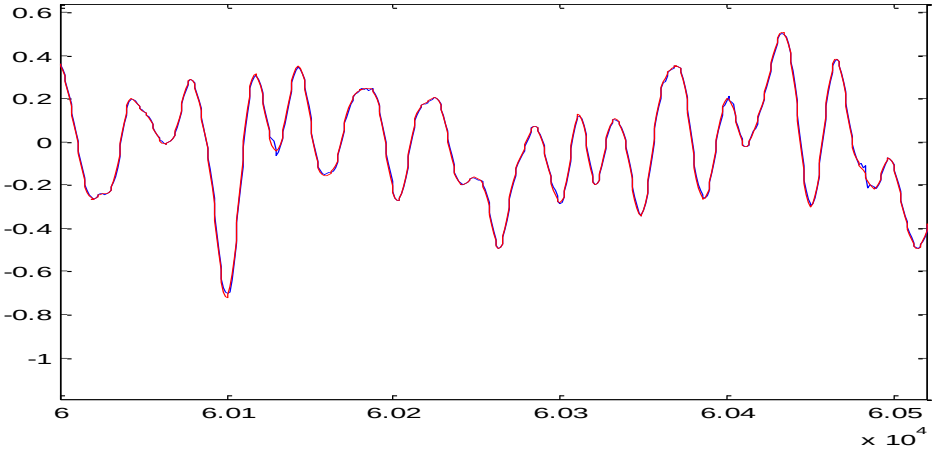

Figure 6: System and model A-ELM outputs during the online identification.

The identification error is given by figure 7. Figure 8 shows the weight beta evolution and its convergence.

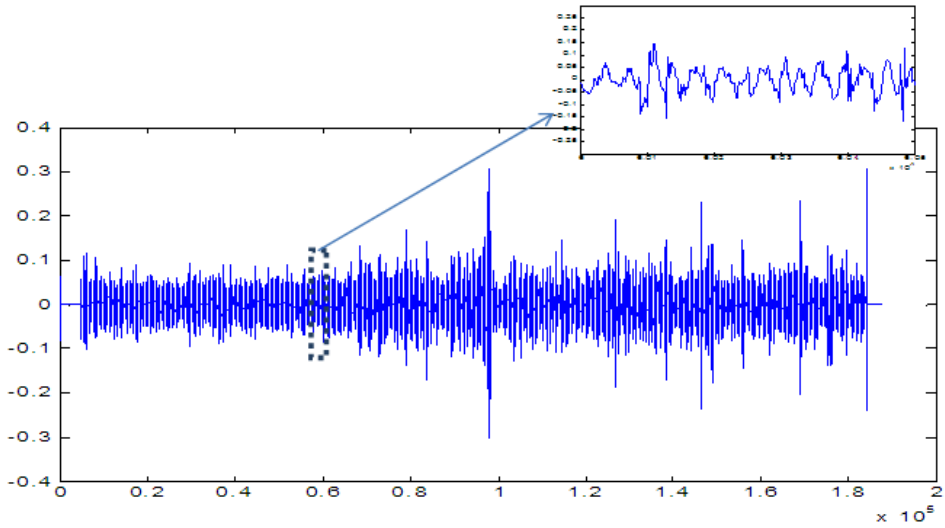

Figure 7: Evolution of MSE.

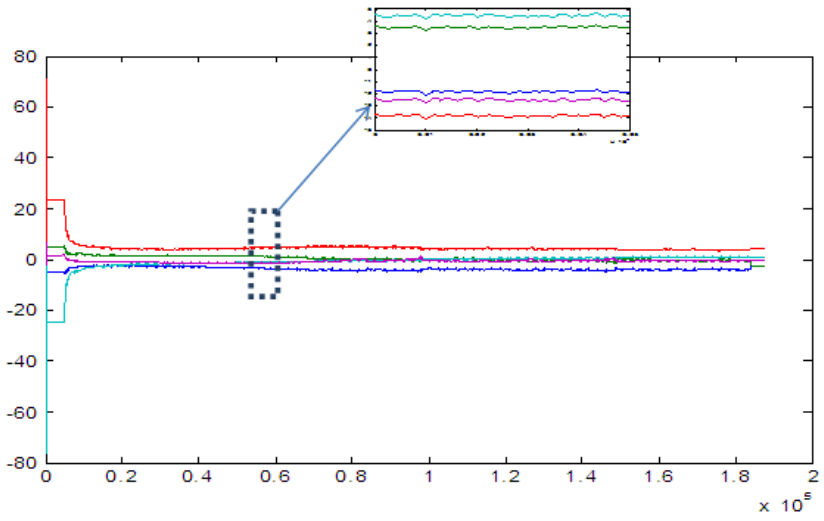

Figure 8: the weight beta evolution

III.2 Tank process

The process is a fluid level control system consisting of two cascaded tanks with free outlets fed by a pump. The water is transported by the pump to the upper of the two tanks. The process is depicted in Figure 9. The input signal to the process is the voltage applied to the pump and the two output signals consist of measurements of the water level of the tanks. Since the outlets are open, the result is a dynamic that varies nonlinearly with the level of water. The process is controlled from a PC equipped with MATLAB interfaces to the A/D and D/A converters.

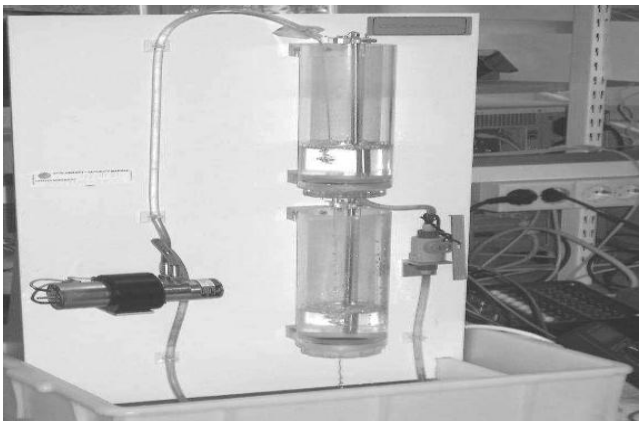

Figure 9. The cascaded tanks.

The laboratory process is suitable for physical modelling. Application of Bernoulli's principle and conservation of mass results in

$$\begin{cases} \begin{pmatrix} \frac{dh_1}{dt} \\ \frac{dh_2}{dt} \end{pmatrix} = \begin{pmatrix} -\frac{a_1\sqrt{2g}}{A_1}\sqrt{h_1} + \frac{1}{A_1}k\,u(t) \\ -\frac{a_2\sqrt{2g}}{A_2}\sqrt{h_2} + \frac{a_1\sqrt{2g}}{A_2}\sqrt{h_1} \end{pmatrix} + \begin{pmatrix} w_1(t) \\ w_2(t) \end{pmatrix} \\ \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} h_1(t) \\ h_2(t) \end{pmatrix} + \begin{pmatrix} e_1(t) \\ e_2(t) \end{pmatrix} \end{cases}$$

Where h1 and h2 denote the levels of the upper and the lower tank, respectively. w1(t) and w2(t) are system noises. The outputs are given by y1(t) and y2(t) , these are corrupted by the measurement disturbances e1(t) and e2(t) . The areas of the tanks are A1 and A2 while the effluent areas are denoted flow conversion constant by k
All data was collected in open loop experiments using zero zero-order hold (ZoH) sampling. The data that was recorded from the cascaded tanks also used an input signal that was generated as the uniformly distributed input signal above. The data are collected each period of 4.0 s and provide 2400 samples of input-output.

Only the input u(k) and the output y(k) are observable and the input output system are the y(k),y(k-1),y(k-2), y(k-3) and the u(k). We perform the identification processes for 1500 time steps. For the simulation process, initially the empty database is used with x1(0)=x2(0)=0. Indeed, the sigmoid activation function in the hidden layer neurons is used. The number of hidden neurons is equal to number of input of the NN, nh=5.
In Figures 10, 11, 12 and 13, the online A-ELM outputs as well as the system output, the evolution of the Mean Square Error (MSE) and the weight beta evolution and its convergence are represented. A great similarity between the model output and the system output is detected and the Mean Square Error is equal to 1.4810^{-5}

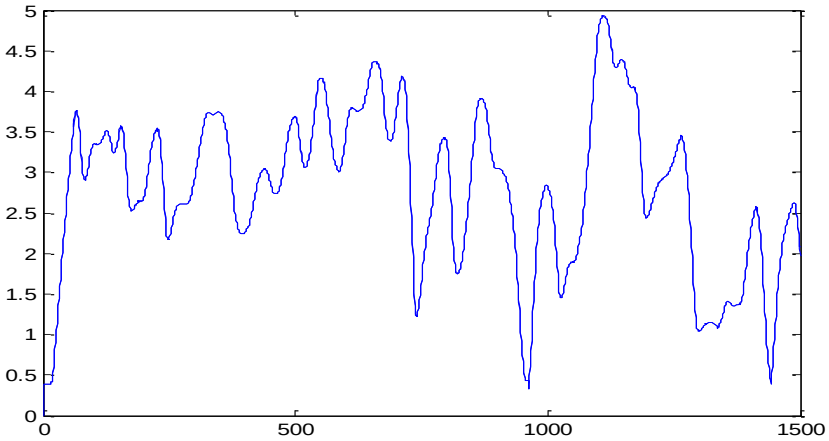

Figure 10: System and model A-ELM outputs during the online identification.

The performance of our algorithm is picked in the training sample window. The main remark is that the model output and the system output have an important similarity. This shows the good performances of the proposed online identification method.

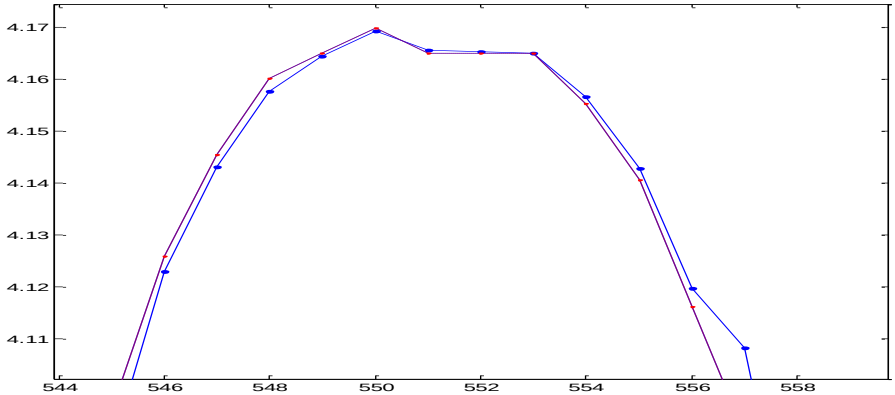

Figure 11: System and model A-ELM outputs during the online identification.

The evolution of error in identification phase is given by figure 12. Figure 13 shows the weight beta evolution and its convergence.

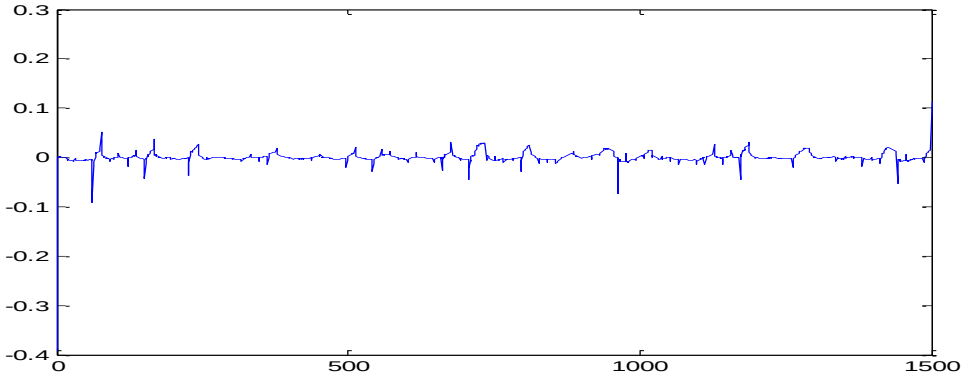

Figure 12: Evolution of MSE.

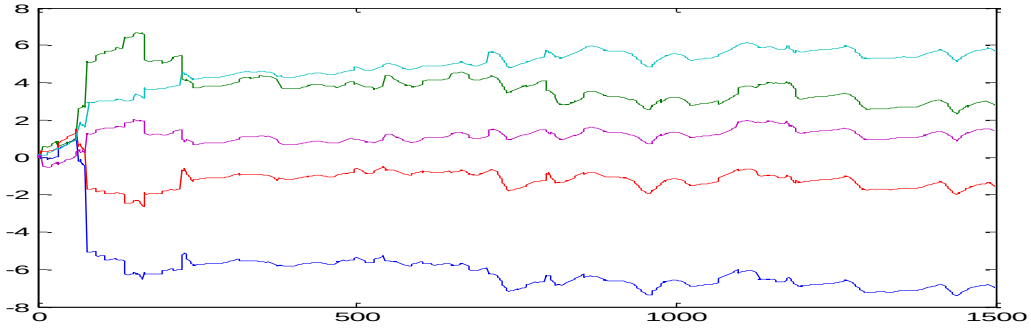

Figure 13 the weight beta evolution

For the two examples in the Table 1 we summarized the performance of the online identification algorithms in terms of Average computation time and Normalized Means Square Error (NMSE). We notice that the online A-ELM method has an average computation time and NMSE less than the SWKRLS and KRLS methods.

Table no 1 Performances of the A-ELM identification methods

Example	Methods	Av. Computation time (sec)	*NMSE*
Wiener Hammerstein benchmark	SWKRLS	0.0112	0.0172
	KRLS	0.114	0.004
	A-ELM	0.0077	0.0079
process tanks	SWKRLS	0.0012	0.00024
	KRLS	0.00132	0.000084
	A-ELM	0.000279	0.000014

IV. Conclusions

In this paper, non-linear system identification is considered. An adaptive extreme learning machine method is proposed for training a recurrent neural network. Using Akaike information criterion, the structure of a recurrent neural network can be self-defined. The proposed online training algorithm can be used without introducing any specific parameter such as the number of hidden nodes. It is self-tuned. For many real applications, presented as black boxes, and for its control purpose, online identification is needed. Through several experiments, the accuracy and the good scaling properties of the proposed A-ELM method are illustrated. This algorithm has been tested to identify a Wiener-Hammerstein benchmark and a tank process and the results are highly appreciated.

References

[1]. Horvath G. Neural networks in systems identification. In: Ablameyko S, Goras L, Gori M, Piuri (szerk.) V, editors.Neural Networks in Measurement Systems. Amsterdam: IOS Press, 2003. pp. 43-78.
[2]. Narendra KS, Parthasaraty K.Neural networks and dynamical systems. J Approx Reason 1992;6:109-131.
[3]. O. Taouali, I. Elaissi, and H. Messaoud 2012 Design and comparative study of online method identification of nonlinear system in RKHS space. Artificial Intelligence Review 2012; 4-37:289-300
[4]. Wang L, Zeng Y, Chen T. Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. J Expert syst Appl 2015; 42: 855-863.
[5]. G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," IEEE Transactions on Neural Networks, 2006; 17- 4 : 879–892.
[6]. G. Huang and L. Chen, "Enhanced random search based incremental extreme learning machine," Neurocomputing, 2008; 71-16: 3460– 3468.
[7]. G. Feng, G.-B. Huang, Q. Lin, and R. Gay, "Error Minimized Extreme Learning Machine With Growth of Hidden Nodes and Incremental Learning," IEEE Transactions on Neural Networks, 2009; 20-8 :1352 –1357.
[8]. Huang GB, Zhu QY, Siew CK. Extreme learning machine: theory and applications. J Neurocomputing 2006; 70: 489-501.
[9]. Huang.G, Huang GB, Song S. Trends in extreme learning machines: A review. J Neural Networks, 2015, 61, 32-48.
[10]. Cao J, Lin Z, Huang G-B (2012) Self-adaptive evolutionary extreme learning machine. Neural Process Lett 36(3):285–305. doi:10.1007/s11063-012-9236-y
[11]. G.-B. Huang, L. Chen, Convex incremental extreme learning machine, Neurocomputing 2007; 70-16: 3056–3062.
[12]. Feng P, Zhao HB. Online sequential extreme learning machine based multilayer perception with output self feedback for time series prediction. J ShangaiJiaotongUniv(Sci) 2013; 18: 366-375.
[13]. Liang NY, Huang GB, Saratchandran P, Sundarajan N. A fast and accurate online sequential learning algorithm for feed-forward networks. IEEE Trans Neural Networks 2006; 17: 1411-1423.
[14]. Diaconescu E. The use of NARX neural networks to predict chaotic time series. WSEAS Trans Compt Res 2008; 3: 182-191.
[15]. Zhao X . A perturbed particle swarm algorithm for numerical optimization. Appl Soft Compt. 2009; 10-1:119–124
[16]. Zhao X, Liu Z, Yang X . A multi-swarm cooperative multistage perturbation guiding particle swarm optimizer. Appl Soft Compt. 2014; 22:77–93
[17]. S. Suresh, S. Saraswathi, and N. Sundararajan, "Performance enhancement of extreme learning machine for multi-category sparse data classification problems," Engineering Applications of Artificial Intelligence. 2010; 23- 7 : 1149–1157.
[18]. Ljung L. System identification- theory for the user. 2nd ed. NJ: Prentice-Hall, 1999.

[19]. Bidyadhar S, Debashisha J. An improved differential evolution trained neural network scheme for nonlinear system identification. J AutomComput 2009; 6: 137-144.
[20]. N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks," IEEE Transactions on Neural Networks, 2006; 17-6: 1411–1423.
[21]. Engel Y, Mannor S, Meir R (2003) The kernel recursive least squares algorithm (RKLS). Rapport technique 2003.
[22]. T. Wigren. Input-output data sets for development and benchmarking in nonlinear identification, Technical Reports from the department of Information Technology. 2010, Uppsala University, Uppsala, Sweden.
[23]. [Van Vaerenbergh S, Vıa J, Santamarıa I. Nonlinear system identification using a new sliding-window kernel RLS algorithm. J. Commun. 2007; 2-3
[24]. Vandersteen G. (1997). Identification of linear and nonlinear systems in an errors-in-variables least squares and total least squares framework. Phd-thesis, Vrije Universiteit Brussel. 1997
[25]. Jaffel, I., Taouali, O., Harkat, M. F., & Messaoud, H. (2016). Moving window KPCA with reduced complexity for nonlinear dynamic process monitoring. ISA transactions, 64, 184-192.
[26]. Taouali O, Elaissi I, Messaoud H, Online identification of nonlinear system using reduced kernel principal component analysis. Neural Comput & Applic 21(1):161–169,2012.
[27]. Taouali O, Elaissi I, Messaoud H, Dimensionality reduction of RKHS model parameters, ISA transactions, Vol57, pp.205-210,2015.
[28]. Wang, J., Lu, S., Wang, SH. et al. A review on extreme learning machine. Multimed Tools Appl, 2021
[29]. Albadr, M., Tiuna, S.: Extreme Learning Machine: A Review. International Journal of Applied Engineering Research 12 ,4610–4623, 2017
[30]. Iago Richard Rodrigues, Sebastião Rogério da Silva Neto, Judith Kelner, Djamel Sadok and Patricia Takako Endo, Convolutional Extreme Learning Machines: A Systematic Review, Informatics , 8, 33,2021
[31]. Zhong-kai Feng , Wen-jing Niu ,Zheng-yang Tang, Yang Xu, Hai- rong Zhang, Evolutionary artificial intelligence model via cooperation search algorithm and extreme learning machine for multiple scales nonstationary hydrological time series prediction,Journal of Hydrology ,595, 2021,

Hend Liouane. "An Improved Extreme Learning Machine for Process Modeling." *IOSR Journal of Computer Engineering (IOSR-JCE)*, 23(6), 2021, pp. 01-12.