ZIZO: A Novel Zoom-In-Zoom-Out Search Algorithm for the Global Parameters of Echo-State Networks

Guodong Wang, *Student Member, IEEE,* Mohamed Amin Ben Sassi, Radu Grosu, *Member, IEEE*

***Abstract*—Echo-state networks (ESNs) are a distinct architecture for recurrent neural networks (RNN). The great advantage of ESN is that they offer an easy way to train the RNN. To make full use of ESN, one needs to first identify their global (hyper) parameters. These are input scaling, leaking rate (for leaky ESN), spectral radius and the size of the ESN. The most recommended way to get their optimal (or sub-optimal) values is by trial-and-error. However, in practice, this method has a very low efficiency. In order to tackle this problem, we propose a novel "Zoom-In-Zoom-Out" (ZIZO) algorithm for generating the global parameters automatically. The proposed technique consists of two major parts. First, we generate random ranges for the parameters of ESNs. Then, based on bootstrap sampling, we search the optimal solution within the fixed specific ranges. To evaluate the proposed method, we use two different data sets which are collected from literature. The obtained results demonstrate the efficiency and accuracy of ZIZO.**

***Index Terms*—Machine intelligence, neural networks, Echo-state networks, sampling, global parameters, machine learning.**

I. Introduction

Echo-state networks (ESNs) provide a distinct architecture and a supervised-learning principle for recurrent neural networks (RNN) [1]. ESNs were first proposed by Herbert Jaeger [2] and are very similar to the Liquid State Machine (LSM) developed in [3]. The main idea of ESNs is feeding an input signal to a large, randomly generated, recurrent, dynamic hidden layer, called a $reservoir$, whose outputs are linearly combined by a memory-less layer called $readout$ [4]. The connections among the input, reservoir and output layers are randomly generated and fixed at the initial stage. As a consequence, for training the ESN only linear regression algorithms are required. Those algorithms only update the weights between reservoir and output layer while the traditional RNN models need a long period of hard training through back-propagation optimization algorithms, as in RNN the learning procedure needs to adjust all the network weights, include input, output and recurrent layers.

During past years, ESNs got more and more attention within the research community and have been adopted in various practical applications such as automatic speech recognition [5], chaotic time series prediction [6], feed-forward control [7], motion identification [8], prediction of telephone-calls load [9]. Despite the wide spectrum of applications, the most intensive research is the prediction of time-series signals. The benchmark test for the Mackey-Glass system [2] and and chaotic time-series signal prediction [6] demonstrated the ability of ESN to model dynamical non-linear systems in a very effective and accurate fashion.

The main bottleneck and crucial problem of ESNs is the lack of an efficient way for searching for the global parameters such as input scaling, leaking rate, spectral radius, and size of the reservoir. The most common used methods so far were restricted to brute-force search and manual adjustment. Mantas suggested in [10] that the best practical way to identify the global parameters is by manual adjustment. The most popular simulation software for reservoir computing - $Oger$ uses brute-force grid-search for exploring the parameters. This leads to optimal or sub-optimal parameter values. But the computing performance is still a serious issue. Skowronski [5] on the other side, manually set up the parameters through many experimental trails in order to make the global parameters more robust to the input signals. The disadvantage of these methods is that they need thorough experimental trails, and prior experience with ESNs. David [11] proposed to use the dynamic profile of the Jacobian of the reservoir instead of static and demonstrated that this approach gives a more accurate description of the reservoir dynamics, and can serve as predictor for the performance. Bianchi [12] presented a valuable tool for exploring the global parameters. They analyzed time-series of neuron activations with Recurrence Plots and Recurrence Quantification Analysis, which permit to visualize and characterize high-dimensional dynamical systems. From the point of automation, these two methods need to much prior information which is very different to be acquire.

In this paper we propose an alternative method based on bootstrap-sampling for automatically computing the optimal values. We use an uniform distribution, from which we draw independent and identically distributed samples in order to generate the global parameters matrix. The proposed algorithm automatically scans the matrix row by row until reaching the predefined termination-condition. The main contribution of this work is to provide an efficient algorithm for automatically computing the optimal or sub-optimal global parameters.

The remainder of the paper is organized as follows. In Section 2, we review the basic structure of ESN. In Section 3, we present our ZIZO search algorithm. To demonstrate the utility of our method, in Section 4 we present our simulation

Guodong Wang, Mohamed Amin Ben Sassi, Radu Grosu are all with the Institute of Computer Engineering, Vienna University of Technology, Treitlstrae 3, 1040 Vienna, Austria e-mails are: guodong.wang@tuwien.ac.at, mohamed.sassi@tuwien.ac.at, radu.grosu@tuwien.ac.at, respectively.

Manuscript received 21 December 2016.

results and experimental analysis. Finally, in Section 5 we present our conclusions and discuss further research plans.

II. Echo-state Networks

A. State equations

The basic structure of an ESN is depicted in Figure 1 where n is the discrete-time steps for the time-series procedure and $u(n)$ is the input at time-step n.

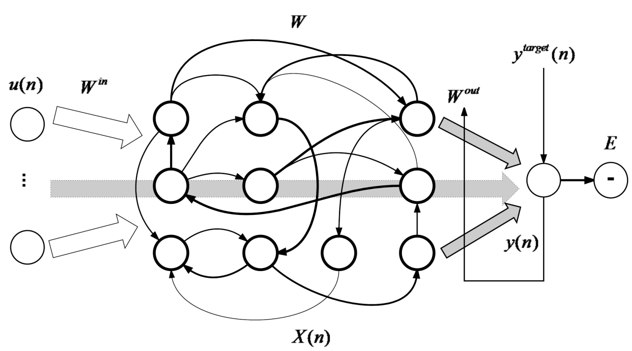


Fig. 1: Basic structure of ESN.

$X(n)$ are the generated states of the reservoir and used to train the ESN with feedback outputs, W^{in} are the input-to-reservoir connection weights which are initialized randomly and W are the internal weights of the reservoir. W and W^{in} are both randomly generated and kept fixed during the whole training process after initial phase. $y(n)$ are the outputs of the readout layer and $y_{target}(n)$ are the reference outputs. E is the unit used for computing the deviation between $y(n)$ and $y_{target}(n)$.

In the reservoir layer, for every neuron one can use different activation functions e.g $sigmoid$ function, $tanh$ function, or other non-linear functions. In this work, we use the $tanh$ function. The state equations of ESN are as follow:

$$X'(n) = f(W^{in}[\beta \cdot u(n)] + WX(n-1)) \quad (1)$$

$$X(n) = (1-\alpha)X(n-1) + \alpha \cdot X'(n) \quad (2)$$

where f is the activation function and α is the leaking rate which is used as memory loss factor of ESN for the historical states. The output layer is defined in Equation 3:

$$y(n) = g(W^{out}[u(n); X(n)]) \quad (3)$$

where $[:;:]$ is the concatenation of column vector (or matrix). For the output active function, one can choose different functions based on practical application.

B. Global control parameters

Given the equations (1), (2) and (3), an ESN model can be defined by a 6-tuple $(W^{in}, W, W^{out}, \alpha, \beta, m)$, m is the size of the $reservoir$. In analogy to RNN, W^{in}, W, W^{out} are called the $local\ parameters$ whereas α, β, m are called the $meta-parameters$ as they working together effect the distribution of W^{in}, W, W^{out} [10]. In this work, we use the name $global\ parameters$ which is introduced by Mantas [10] to reflect the nature of all ESN parameters. There are also other important global parameters, for example, the teacher scaling, feedback scaling, sparsity of the connections in the reservoir, noise in the state-update and etc. In this paper, we focus only on the ones listed as follows.

Size of Reservoir m. The first global parameter is obviously the size of reservoir m who has a great impact on the performance of ESN. Intuitively, the larger m, the better performance is obtained, namely the larger reservoir space will lead to a better chance to find appropriate internal states to produce accurate output. Like the traditional Machine Learning problem, we also need to consider the $overfitting\ issue$, which means that a larger number of reservoir neurons does not always mean better results. Another problem is that a too large reservoir may require extensive data for proper training, and this data may be lacking.

Input Scaling β is used for shifting input values to the activation range of the neurons in the reservoir. It determines the operation point of the nonlinear reservoir behavior. Finding this point is in general very difficult, and it is application dependent. One should choose appropriate input scaling based on the degree of nonlinearity of the tasks, while the amount of nonlinearity that the task needs is difficult to define quantitatively. Therefore, manually trying different input scaling is a tough work for using ESN.

Spectral Radius ρ The very nature of an ESN model is the $echo$-$state$ property. The large ρ can lead to reservoirs hosting multiple fixed points, periodic, or even chaotic (when sufficient nonlinearity in the reservoir is reached) spontaneous model modes, violating the echo states property [10], as the state of the reservoir x(n) should be uniquely defined by the fading history of the input u(n). If ρ is too big, the reservoir state $x(n)$ will strongly depends on the initial conditions that were before the input. A strong $u(n)$ will push activations of the neurons away from 0 where their $tanh()$ nonlinearities have a unitary slope to regions where this slope is smaller, thus reducing the gains of the neurons and the effective strength of feedback connections. Therefore, the recommended range of the spectral radius is less than 1 which ensures ESN can produce echo-state in most circumstances. In practical applications, we can obtain the spectral radius as follows:

$$\rho = |eigs(W)| \quad (4)$$

$$W = W \cdot (1.25/\rho) \quad (5)$$

where $eigs$ is the function used to calculate the max eigenvalue of the reservoir weight-matrix W, $|\cdot|$ is the absolute value of one variable.

Leaking Rate α is used to define the speed with which the reservoir states are updated based on the new input and (or) output information. The smaller α means the lower the update speed of the ESN. In practice, it is also hard to define an appropriate updating speed for ESN.

In this paper, we try to develop an algorithm which can automatically generate α, β, m and ρ. As described in equations 4 and 5, ρ is defined by m, so here we just consider the scenery of automatically computing the global parameters: α, β and m.

III. Proposed Method

Our zoom-in-zoom-out (ZIZO) search algorithm was inspired by the *Bootstrap method* (BS) proposed in [13]. BS is a straightforward way to estimate standard errors and confidence intervals. If we find one sample with acceptable MSE (a value below a specified threshold) we are going to pick that sample. Otherwise, we pick the sample with lowest MSE and randomly generate a box around it, by randomly sampling left and right bounds for every dimension of the sample, within the original ranges of the parameters. We then continue the search as before, until the MSE is below the given threshold.

Next we are going to provide the theoretical results and explain the thought behind our proposed algorithm.

Algorithm 1: The Proposed Method - ZIZO Algorithm

Data: Input X, Output Y, sampling rate N, maximum re-sampling times M, desired mean square error E_d, prediction length L, RS is the range of global parameters (α, β, m), update rate γ_α, γ_β, γ_m, randomly generated radius BR.

Result: Optimal or sub-optimal (α_i, β_i, m_i).

1 **begin**
2   **for** $j = 1$ *to* M **do**
3     $PARA$ = rand(N, $[\alpha, \beta, m]) \subset RS$
4   **while** $i = 1$ *to* N && $\epsilon_{ji} > E_d$ **do**
5   ESN={α_i, β_i, m_i } = $PARA_{(i,:)}$
6       $\epsilon_{ji} = E(y, y^{target}) = \frac{1}{L} \sum_{i=1}^{L} \sqrt{(y_i - (y_i)^{target})^2}$
7   **if** $i <= N$ **then**
8   *break*;
9   **else**
10   (α, β, m) = ($min(\epsilon_\alpha)$, $min(\epsilon_\beta)$, $min(\epsilon_m)$)
11   BR = rand(1, [BR_α, BR_β, BR_m])
12   RS_α = [α - $\gamma_\alpha \cdot BR_\alpha$, α + $\gamma_\alpha \cdot BR_\alpha$]
13       $s.t.\ \alpha$ - $\gamma_\alpha \cdot BR_\alpha >= \alpha_{lowerBound}$
14   α + $\gamma_\alpha \cdot BR_\alpha <= \alpha_{uperBound}$
15
16   RS_β = [β - $\gamma_\beta \cdot BR_\beta$, β + $\gamma_\beta \cdot BR_\beta$]
17       $s.t.\ \beta$ - $\gamma_\beta \cdot BR_\beta >= \beta_{lowerBound}$
18   β + $\gamma_\beta \cdot BR_\beta <= \beta_{uperBound}$
19
20   RS_m = [m - $\gamma_m \cdot BR_m$, m + $\gamma_m \cdot BR_m$]
21       $s.t.\ m$ - $\gamma_m \cdot BR_m >= m_{lowerBound}$
22   m + $\gamma_m \cdot BR_m <= m_{uperBound}$
23
24   RS = [RS_α, RS_β, RS_m]
25   **if** j <= M **then**
26   *Get the optimal* $(\alpha_i, \beta_i, m_i) = min(\epsilon)$
27   **else**
28   *Get the sub* $-$ *optimal* $(\alpha_i, \beta_i, m_i) = min(\epsilon)$

The principle of ZIZO is presented in Fig 2. In the initial stage, according to the ranges (α_i, β_i, m_i) of the global parameters p_i, we obtain the box in dotted line (we assume that the parameters are in two dimensions). Then, we randomly generate N random samples inside the box, pick each of the samples as the parameters of ESN and run the ESN model. In the next step we check the MSE of the output. If it is less than θ, then we stop. After exhausting all the N sample, if we still did not find the desired global parameters, then the one with the minimum ϵ which MSE achieved is chosen as a reference point for generating the new searching box. For each box we choose randomly ranges (α_i, β_i, m_i) based on the reference points. Specifically, for each global parameter p with value v_p, we choose a random range $[a_p, b_p]$ where a_p lies in the interval $[u_{p^m}, v_p]$ and b_p lies in the interval $[v_p, u_{p^M}]$, where $[u_{p^m}, u_{p^M}]$ is the range of the parameter p within the dashed box. As shown in Fig 2, after the first N trials, we get the reference point C_1 and not the optimal parameters. Then, a new box around C_1 is randomly generated. The way of how to generate the random box is shown in line 11, 16 and 20. γ_α, γ_β and γ_m are used for adjusting the boundary of the generated random box. Next, the previous steps are repeated. Finally, we may have reference points C_2, C_3, C_4, ..., C_i, C_n, C_{star}. By fixing the size of the box in advance, it is always possible to miss the chance of getting the optimal solution. For instance, in tree searching, one has both width-searching and depth-searching. If one misses the optimal solution in the searching process, one has no chance to get back to it later on, without backtracking. In the ZIZO search algorithm, we propose a random-box-size search based on previous search results. This Zoom-in-Zoom-Out strategy gives us the chance to get back to possibly-missed previous parameter-search areas (akin to backtracking).

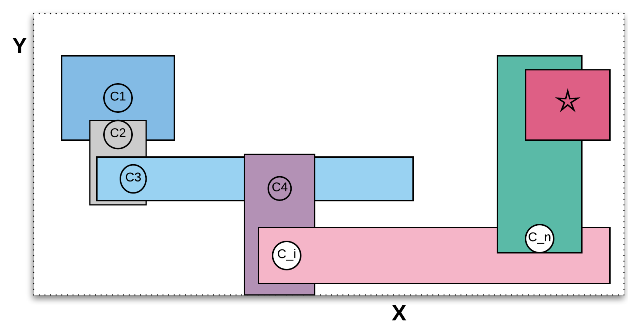


Fig. 2: Principle of ZIZO.

According the bootstrap theory, if we randomly select q different samples (data points) in one specific box with uniform distribution, then every sample is chosen with probability $1/q$. Then, if we sample infinitely many times, we obtain the optimal parameters (e.g., standard deviation) while the percentage of un-selected samples in the data-set is $\lim_{q\to+\infty} (1 - \frac{1}{q})^q \approx e^{-1} = 0.368$. Comparing with brute-force grid search method, the proposed method will obtain the optimal parameters using only 63.2% data in the worst case.

The time complexity of the proposed algorithm is $O(M \cdot N)$, and once we setup the values for M and N, the computing cost of our algorithm is constant, namely $O(constant)$. This is the greatest advantage compared to other existing algorithms, for

TABLE I: Selected Parameters for Mackey-Glass Model

Case	Iterations	α	β	m
1	101	0.47666	0.80618	444
2	38	0.81052	0.93495	379
3	112	0.32762	0.89783	364
4	59	0.6172	0.97062	393
5	101	0.46795	0.87622	407
6	5	0.64911	0.80069	447
7	245	0.54739	0.53543	351
8	266	0.68985	0.9884	216
9	43	0.50805	0.78367	427
10	106	0.50058	0.90311	304

TABLE II: Selected Parameters for Mackey-Glass Model

Case	MSE	Computing Time	Atuomation
case 1	7.5562e-07		
case 2	8.3911e-06		
case 3	9.2214e-06		
case 4	3.2251e-06		
case 5	9.9136e-07	$\leq 30s$	automation
case 6	7.7338e-06		
case 7	1.5869e-06		
case 8	3.9363e-07		
case 9	6.6814e-07		
case 10	7.3933e-07		
reference[10]	1.0e-6	$\geq 30s$	semi-automation

instance, brute force searching, grid searching, evolutionary algorithms and so forth. To demonstrate the effectiveness and accuracy of the proposed method, we apply it in two case studies. One is the well known benchmark test for Mackey-Glass model , and the other is the collected motor-current signals from a machining process.

IV. Experiments

We re-implemented the ESN model originally supplied by [10] in Matlab. To evaluate the proposed method, we run our algorithm on two different data sets: (a) The data generated from the Mackey-Glass model and (b) Data generated from the current signals of a machining process.

A. Mackey-Glass model

A sample of the Mackey-Glass model generated data is depicted in Figure 3, where

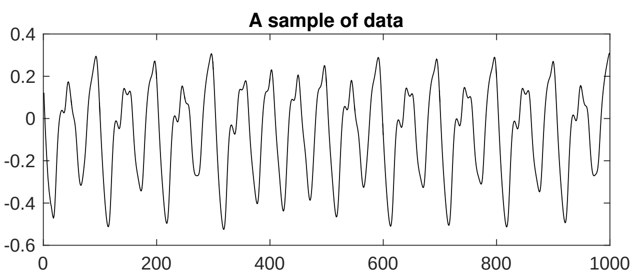


Fig. 3: A sample of Mackey-Glass Model.

10000 data points are used. The first 2000 data points are used for training ESN and the remaining data for prediction. For comparing the simulation results with benchmark test, the first 500 predicted points are used for calculating the MSE. By using the proposed method, we can get different parameters for ESN which are listed in Table I. The relevant simulation setup is: $M = 500$, $N = 100$ and the threshold $MSE = 1.0e-6$. In [10], an MSE with the order $1.0e-6$ is obtained. The comparison between the best result and our prosed approach is shown in Table II

In Table I and Table II, we show that a high predictive accuracy can be automatically generated. In fact, we were able to get 10 times higher accuracy. Table I shows the simulation results. In the first row, one can see that 101 iterations are needed to obtain a $MSE = 7.5562e-07$. In the 5th row, after same times of iterations, one can obtain different MSE which is $9.9136e-07$. Comparing this two cases, the deviation between them is tiny while the root of the difference is from the random-search box. To visualize what happens inside the ESN, we plot the states of $Weight\ Distribution$ (Fig 4) and $Reservoir\ States$ (Fig 5). For clearly showing what happened inside of the reservoir in running time, we only plotted part of the states of reservoir in the first 200 ms (Same for Fig 9). We also plot the deviation between the target and the prediction signals (Fig 6) showing the accuracy of the obtained results. Except the prediction accuracy, the time complexity of ZIZO is constant $O(M \cdot N)$, M times N is the maximum iterations ZIZO needs to search the optimal solutions. All the executed simulations are finished within 30s while the existing methods need much longer time. According to the theory of ESN,

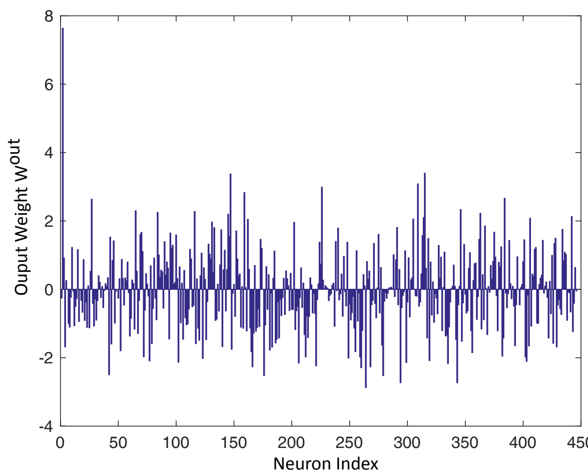


Fig. 4: Weight distribution.

the weight distribution and reservoir states work together and control the prediction accuracy. Fig 4 shows that the range of weight values of ESN is between $(-2.9, 7.8)$ and Fig 5 shows that the obtained ESN has very rich reservoir-states. Those states should be independent with each other as much as possible since the output is a linear combination of reservoir-states. Figure 6 shows the prediction results generated by the trained ESN. For the first 1000 data points, one can see that the prediction is close to zero.

One can see that in all the cases, the generated global parameters lead to good prediction accuracy. For practical application, one can select an appropriate one according to specific selecting-rules [10]. To acquire a better ESN, there are

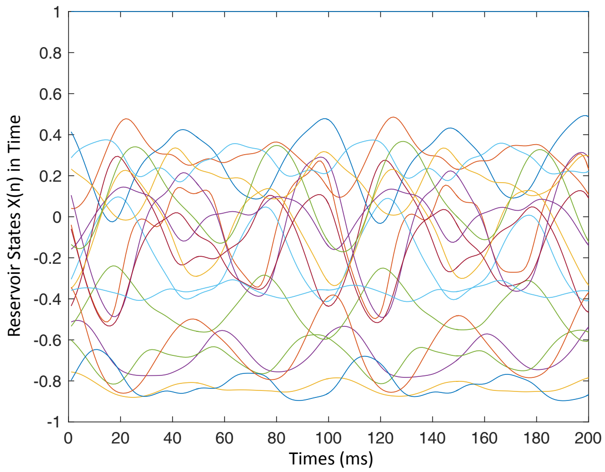


Fig. 5: Reservoir states.

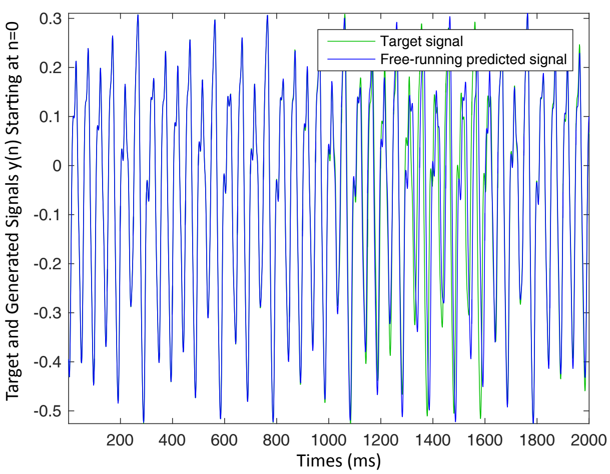


Fig. 6: Target and prediction signals.

several rules should be applied. From the point of stability, the longer "zero-error" time (pints) in the Figure of 6, the better stability of ESN. The weight distribution and the correlation among the lines of reservoir states are two important signs for selecting appropriate global parameters: 1) The weights should very close to each other, and if the range is smaller, it is better to stability of ESN. 2) Reservoir sates should be independent with each other as much as possible. According to the simulation results of Mackey-Glass benchmark test and the selecting rules, we can conclude that when the $Weight$ in the range $[-10,\ 10]$ and $Reservoir\ States$ fall into $[-1,\ 1]$, the ESN is stable.

B. Motor-current signal prediction

Motor-current is one of the most important parameters for monitoring the health states of machining tools [14]. In this section, we use the milling process generated data in [15]. The motor-current of DC spindle is selected for evaluating the performance of our proposed method. A sample of the current data is depicted in Figure 7. In the industrial engineering, the signal in the first 3000 time steps can be dropped, because during this period the system is very unstable. However, even after a long running time the collected signals contained oscillatory states (see the down part of Figure 7). Therefore, we use the data points in the range $[3000, 9000]$. The simulation set up is: $M = 500$, $N = 100$, the threshold $MSE = 1.0e-2$ and the predicted first 500 data points are used for computing MSE.

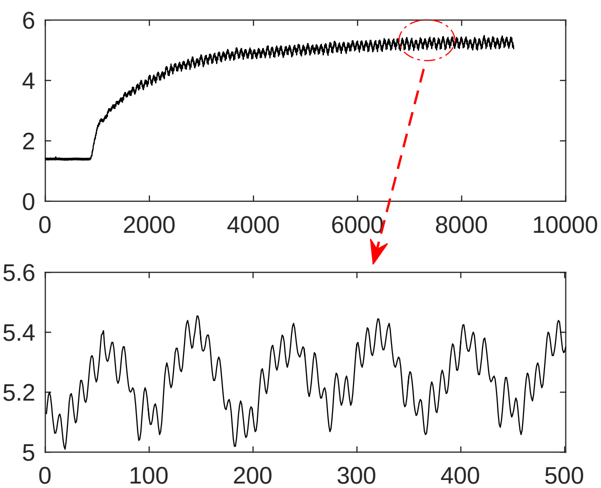


Fig. 7: A sample of DC motor generated current signal (up) and points from 7500 to 8000 which is marked by the red circle (down).

As the sensory data from the machining contains a considerable amount of noise, the model is not very accurate. From the industrial point of view, a good prediction accuracy is around $1.0e-3$. Similarly to the previous benchmark test, we provide 10 different runs of our algorithm (Table III). One can see that a pretty good result is obtained in case 5 since the ESN generates very robust reservoir-states. As the data is noisy, the range of $weight\ distribution$ and $reservoir\ state$ are $[-80,\ 80]$ and $[-1,\ 8]$. This significantly affects the prediction accuracy. It indicates that reducing the nose is an essential step for using ESN. In spite of this oscillatory behaviors, one can see that the ZIZO algorithm can quickly find sub-optimal global parameters.

TABLE III: Selected Parameters for the prediction of Motor-current

Case	Iterations	α	β	m	MSE
1	132	0.29062	0.33634	369	2.2221e-03
2	16	0.56803	0.023929	453	1.0332e-03
3	12	0.36892	0.44524	168	2.5729e-03
4	93	0.57715	0.17624	213	6.8746e-03
5	182	0.39731	0.039942	379	2.0152e-03
6	193	0.6857	0.1667	444	2.2767e-02
7	452	0.9722	0.088102	437	1.6422e-03
8	141	0.29936	0.4698	155	2.0604e-03
9	99	0.87626	0.045372	371	3.0254e-03
10	57	0.78899	0.10019	351	2.1324e-03

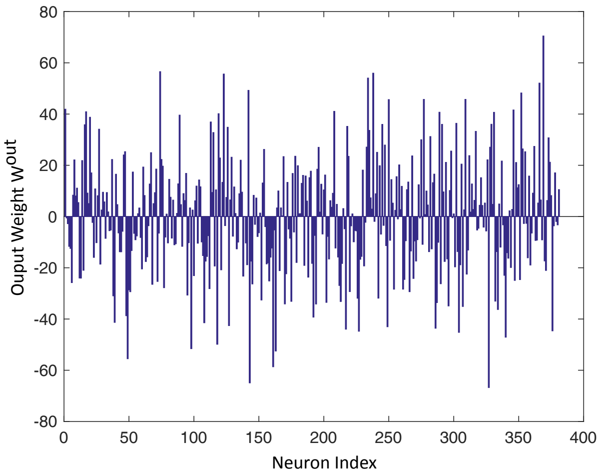


Fig. 8: Weight distribution.

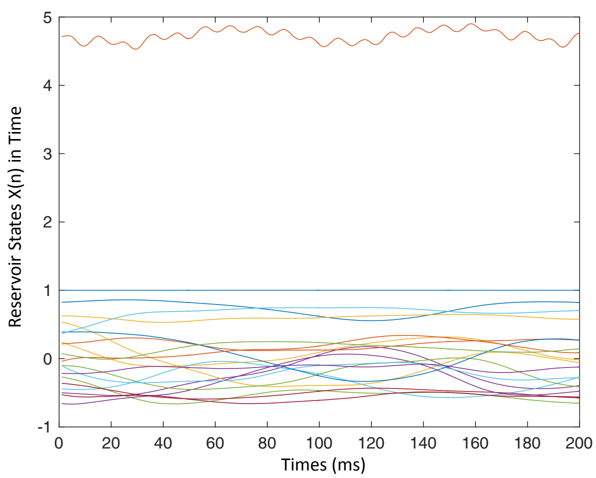


Fig. 9: Reservoir states.

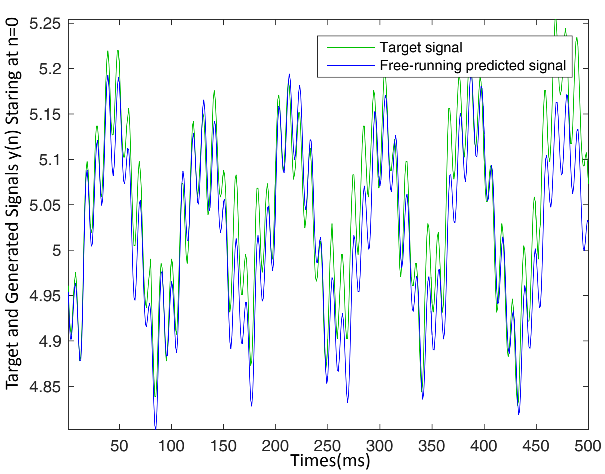


Fig. 10: 5th prediction results

V. Conclusion

In this paper we propose a simple but very effective bootstrap-inspired zoom-in-zoom-out parameter-search method for identifying the optimal global parameters of echo-state networks (ESN). The great advantages of ZIZIO are its simplicity and performance in efficiently generating a batch of meaningful and accurate parameters (α, β, m) for practical application of ESNs.

As future work, we plan to extend our algorithm as follows:

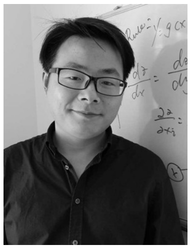
1) Further improve our ZIZO search algorithm by accelerating the searching process as many iterations are needed in order to reach good accuracy.
2) Apply this work to monitor the healthy states of Cyber-Physical Production Systems especially the machining tools. More precisely, since the sharp degree of machining tools gets damaged after specific working time, one should be able to predict the healthy states of machines. Therefore, we will try to use ESN to model the states of machining tools and predict the emergent behaviors of manufacturing systems.

Acknowledgment

This work is funded by the Doctoral College - Cyber-Physical Production Systems Project, Vienna University of Technology, Austria.

References

[1] Herbert. Jaeger, *Echo state network*. Scholarpedia, 2(9):2330, 2007.
[2] Jaeger, Herbert, *The Echo state approach to analysing and training recurrent neural networks-with an erratum note*. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148 (2001): 34.
[3] Maass, Wolfgang, Thomas Natschlger, and Henry Markram, *Real-time computing without stable states: A new framework for neural computation based on perturbations*. Neural computation 14.11 (2002): 2531-2560.
[4] Lkse, Sigurd, Filippo Maria Bianchi, and Robert Jenssen, *Training Echo state Networks with Regularization through Dimensionality Reduction*. arXiv preprint arXiv:1608.04622 (2016).
[5] Skowronski, Mark D., and John G. Harris, *Automatic speech recognition using a predictive Echo state network classifier*. Neural networks 20.3 (2007): 414-423.
[6] Li, Decai, Min Han, and Jun Wang, *Chaotic time series prediction based on a novel robust Echo state network*. IEEE Transactions on Neural Networks and Learning Systems 23.5 (2012): 787-799.
[7] Xu, Dongming, Jing Lan, and Jose C. Principe, *Direct adaptive control: an Echo state network and genetic algorithm approach*. Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.. Vol. 3. IEEE, 2005.
[8] Ishu K, van Der Zant T, Becanovic V, Ploger P, *Identification of motion with Echo state network*. OCEANS'04. MTTS/IEEE TECHNO-OCEAN'04. Vol. 3. IEEE, 2004.
[9] Bianchi, Filippo Maria, et al, *Prediction of telephone calls load using Echo state Network with exogenous variables*. Neural Networks 71 (2015): 204-213.
[10] Lukosevicius, Mantas, *A practical guide to applying Echo state networks*. Neural networks: Tricks of the trade. Springer Berlin Heidelberg, 2012. 659-686.
[11] Verstraeten, David, and Benjamin Schrauwen, *On the quantification of dynamics in reservoir computing*. International Conference on Artificial Neural Networks. Springer Berlin Heidelberg, 2009.
[12] Bianchi, Filippo Maria, Lorenzo Livi, and Cesare Alippi, *Investigating Echo-State Networks Dynamics by Means of Recurrence Analysis*. IEEE Transactions on Neural Networks and Learning Systems (2016).
[13] Efron, Bradley, *Bootstrap methods: another look at the jackknife*. Breakthroughs in Statistics. Springer New York, 1992. 569-593.
[14] Salgado, D. R., and F. J. Alonso, *An approach based on current and sound signals for in-process tool wear monitoring*. International Journal of Machine Tools and Manufacture 47.14 (2007): 2140-2152.
[15] NASA, *http://ti.arc.nasa.gov/project/prognostic-data-repository*. NASA Ames Research Center, Moffett Field, CA.



Guodong Wang received the M.Sc. degree in systems science from Nanjing university of information science and technology, Nanjing, China, in 2014. He is currently pursuing the Ph.D. degree in computer science with Professor Radu Grosu from Vienna University of Technology, Vienna, Austria.

He is working as a research assistant at the institute of computer engineering of the Vienna University of Technology. His research interests include cyber-physical systems, smart manufacturing, machine learning, data mining, deep neural networks, statistic analysis, multi-agent computing.

Mohamed Amin Ben Sassi received the Ph.D degree in applied Mathematics from Grenoble university, France, dealing with analysis and control of polynomial dynamical systems. He did a first two years postdoc in the University of Colorado Boulder under supervision of Associate Professor Sriram Sankarananarayanan. Now he is a post-doctoral researcher in Vienna University of Technology since April 2016 under supervision of Assistant Professor Ezio Bartocci and Professor Radu Grosu. His current work is on optimization, verification and control of Hybrid systems.

Radu Grosu received the Ph.D. degree in computer science from Technical University of Munich, Munich, Germany, in 1994. He is a Professor and Head of the Cyber-Physical Group at the Faculty of Informatics of the Vienna University of Technology, and a Research Professor at the Computer Science Department of the State University of New York at Stony Brook. His research interests include modeling, analysis and control of cyber-physical and biological systems and his application focus includes green operating systems, mobile ad-hoc networks, automotive systems, the Mars rover, cardiac-cell networks and genetic regulatory networks.

He is the recipient of the National Science Foundation Career Award, the State University of New York Research Foundation Promising Inventor Award, the ACM Service Award, and a member of the International Federation of Information Processing WG 2.2. Before receiving his appointment at the Vienna University of Technology, he was an Associate Professor in the Computer Science Department of the State University of New York at Stony Brook, where he co-directed the Concurrent-Systems laboratory and co-founded the Systems-Biology laboratory. He was a Research Associate in the Computer Science Department of the University of Pennsylvania.