ORIENTAL JOURNAL OF
COMPUTER SCIENCE & TECHNOLOGY
An International Open Free Access, Peer Reviewed Research Journal
Published By: **Oriental Scientific Publishing Co., India.**
www.computerscijournal.org

ISSN: 0974-6471
December 2015,
Vol. 8, No. (3):
Pgs. 205-215

An Aproach of Design and Training of Artificial Neural Networks By Applying Stochastic Search Method

KOSTANTIN P. NIKOLIC

Faculty of Management, Department of Informatics, Novi Sad.

(Received: September 01, 2015; Accepted: November 20, 2015)

ABSTRACT

Although vast research works have been paid (throughout some 20 years back) regarding formal synthesis of an ANN it is somehow still open issue. This paper does not consider the mentioned formal synthesis aspects but intends to introduce an original engineering approach offering some advantages whenever training and designing of artificial neurel networks are under consideration. In that sense the author uses powerful combined method of stochastic direct search, the univerzal approximator and simulation method. Author has created specific stochastic serch algorithm wich in some cases having advantages over numerous known methods wich are based on the application of the gradient. The said algorithm incorporates universal approximation and simulation during designing and training neurel networks. The offered approach is applicable for wide range of artificial neural networks types icluding recurrent ones in real time. The presented numerical examples illustrate applicability of the offered advanced approach.

Key words: Formalization, Artificial Neural Network (ANN), Syntesis, Stochastic Search (SS) method, Stochastic Direct Search (SDS), Univerzal Approximation (UA), Simulation, Recurent Neural Network (RNN or RANN).

INTRODUCTION

This paper is motivated to introduce some experiences related to analysis & synthesis of artificial neural networks (ANN) and to offer rather simple but effective approache in solving practical problems with ANN.

Besides, this paper points out implementation of algorithm based on stochastic search (SS) i.e. Stochastic Direct Search (SDS). Basics of SDS were launched the same time (70-th years last century) when other statistical methods were introduced such as stochastic aproximation, min square error and others[1,2,3,4].

Apart of proved advantages of SDS method over others somehow it has not been remarkably used. It is worthwhile to mention that SDS method superior vs others based on the gradient[5]. That the SDS is superior so to the method of stochastic approximation has been shown in[6]. The superiority is more emphasized by increasing of the dimensions of the problem.

During last 20 years the practice has been changed by numerous published papers of well known researchers[7,8,9]. In the same time the author has been focusing an attention onto SDS method supported by appropriate algorithms suitable for identification and optimization of control systems.

Some of results were published on international conferences[10,11,12]. Last 10 years were dedicated to SDS implementation on analysis & synthesis of ANN[13,14,15].

Among pointing out of SDS method implementation, the presented approach offers simplified instruction in choice of an ANN architecture based on known works[16,17,18,19,20] linked to universal approximation. An implementation of combination between theory of universal approximations and simulated methods linked to SDS algorithms create a rather new options for efficient procedures in analysis and synthesis ANN.

In order to prove the SDS efficiency and validity of numerical experiments it is compared against back propagation error method (BPE) as reference[21].

Nowadays computer technology enables ease handling of numerous variables and parameters including optional combination of heuristic algorithms whenever further development of SDS is not concerned. Numerical mathematic and experiments are crucial tools of researchers where quantification of researching results are considered.

METHOD

An artificial neuron, as per Mc Cullock & Pitts has limited ability in neuro-processing[22]. Having in minds the aforesaid complex structures are created in order to exert improved performances [23,24]. Further on a focus will be paid onto ANN multi-layers feed-forward (FANN) and real-time recurrent neurel networks (RTRNN); (Fig. 1 & Fig. 2). Complex structures of FANN type shown in Fig. 1 having such properties which enable successful training under supervising by using BPE[21]. It is worthwhile to point-

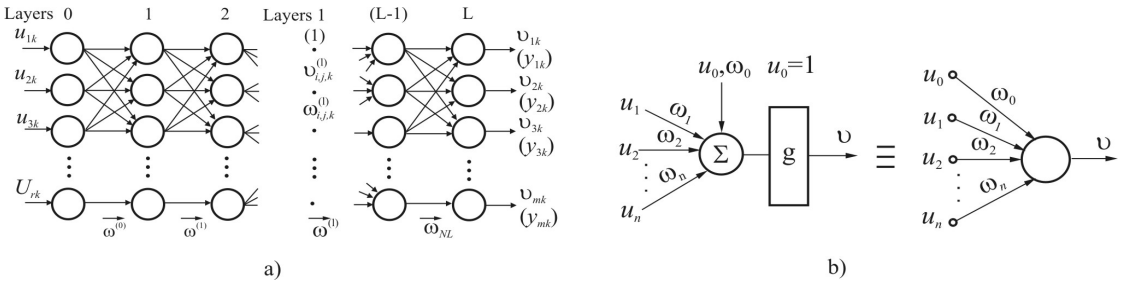

Fig. 1: The FANN general type a); and type of neurons in the same b)

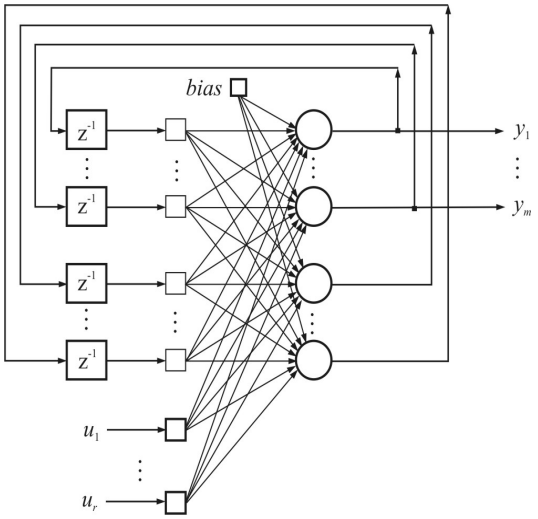

Fig. 2: Rekurent neural network. Jordan's RANN

out that almost all ANN can be transformed into FANN (BPE Throught Time (BPETT))[25,26]. What give enough reasons to start our further presentation with this type of ANN i.e. the FANN.

SDS Method in ANN training

Throughout of FANN supervised training process an intention is to minimize an error at the network output excited by training pair:

$$e_{sk}^{(L)} = y_{sk}^{(t)} - y_{sk}^{(L)} \quad ...(1)$$

according to cost function:

$$Q_k = 1/2[\Sigma(e_{sk}^{(L)}]^2 \quad ...(2)$$

by iteration procedure

$$\omega_{v+1} = \omega_v + \Delta\omega_v$$

where are: v -step of iteration; s=1,2,3,… .,m; k= 1,2,3…..,K; K is number of training pairs

$p_k = (u_{sk}, y_{sk}^{(L)})$ in a set P; $p_k \in P$, L is the layer on the end of FANN.

Back Propagation Error (BPE) uses iterative procedure:

$$\Delta\omega_{ij}^{(l)} = - \alpha \, \partial Q_k / \partial \omega_{ij}^{(l)}; \; 0<\alpha\leq 1 \qquad ...(3)$$

where are:

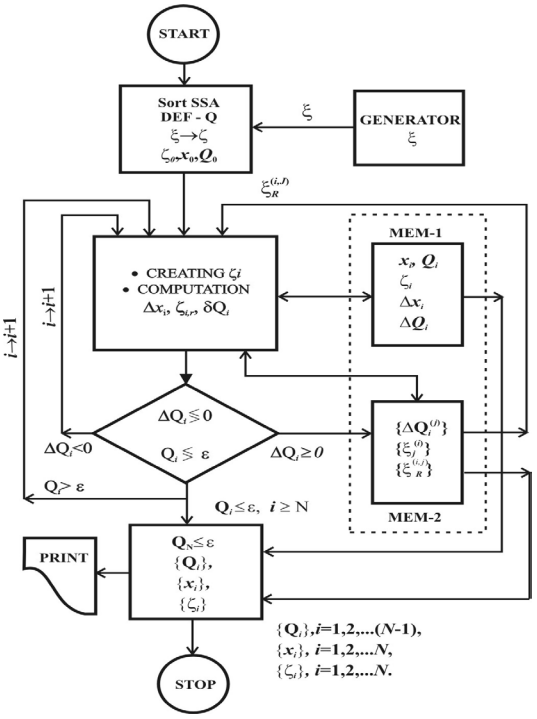

Fig. 3: Diagram of procedure for computer processing of MN-SSA

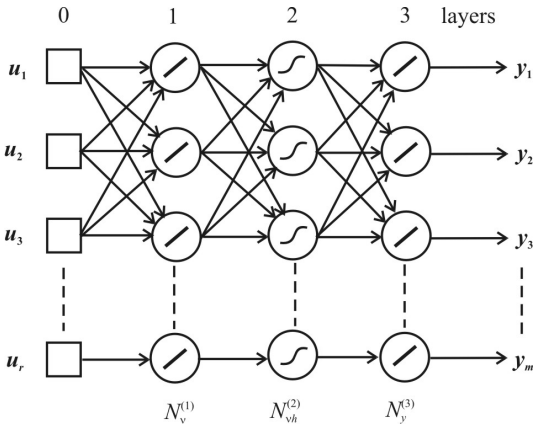

Fig. 4: The general structure of ANN to the universal approximator

$$\partial Q_k/\partial \omega_{ij}(l) = [\partial Q_k/\partial net_j^{(l)}][\partial net_j^{(l)}/\partial \omega_{ij}^{(l)}];$$
$$\partial net_j^{(l)}/\partial \omega ij(l) = u_j^{(l-1)}; \; \partial Q_k/\partial net_j^{(l)} = \delta_j^{(l)}; \qquad ...(4)$$

i, j are indexes at two neurons at neighbouring layers l-1 and l. The motion $\delta_j^{(l)}$ is important variable in optimisation process of BPE for FANN in back propagation stage after final completed forward step where succesive feeding of input out of set of training pairs[21]

SDS iterative procedures are performed in steps of random vectors varying for intire FANN network is:

$$\Delta\omega_v = \alpha \; ort \; \xi_v , \quad \Delta Q_v<0 \qquad ...(5)$$

i.e. as per layers *l* for ζ successful step of iteration[27];

$$\Delta\omega_{ij,v}^{(l)} = \alpha \; \zeta_{ij,v}^{(l)} \qquad ...(6)$$

where are: l = 1,2,3,...,L ; i = 1,2,3,... $N_i^{(l-1)}$; j = 1,2,3,...,$N_j^{(l)}$; *l* –layers FANN, *i* and *j* are neurons indexes of two adjacent layers, with total neuron number $N_i^{(l-1)}$ and $N_j^{(l)}$; ζ is random vector the same dimension as vector ω; $0< \alpha \leq 1$; v=1,2,3,...,Nε, where Nε is the final step of search; $\zeta_{ij,v}^{(l)}$ are the components of $ort\xi_v=\zeta_v$, $|\zeta_v|=1$.

It has been previously mentioned that MN-SDS algorithm was created and suggested by the author[27]. Its basic version is defined by:

$$\Delta\omega_{v+1} = \begin{cases} \alpha \, \zeta_v , \quad \Delta Q_v < 0 \; ; \; \Delta Q = Q_v - Q_{v-1} \\ \alpha \zeta_\lambda^{(v)} , \; \Delta Q_\lambda^{(v)} \geq 0 , \\ - \alpha \zeta_R^{(v,\ddot{E})} + \alpha \, \zeta_{v+1} \; ; \; \Delta Q_{v+1} < 0 , \end{cases} \qquad ...(7)$$

where: $\zeta_v = ort \, \xi_v$; ζ_v - is refered to *v* step of iterration, ΔQ_v – increment of cost function Q in v effective iteration step; $\zeta_R^{(v,\ddot{E})}$ – is resultant of all failed steps $\zeta_\lambda^{(v)}$, $\Delta Q_\lambda^{(v)}$ - increment of Q for failed steps λ, λ=1,2,3,...,ζ; ζ_{v+1} is first step with $\Delta Q_{v+1}<0$ after λ=Ë ; for ξ and ξ to apply the relation $|\xi|\leq 1$, $|\zeta|=1$.

The using $ort\xi=\zeta$ is nessesary for the improvement stability of process training. Minimum of a cost function Q is obtained in forward stage for particular choice of SDS algorithm. SDS

does not require both nither differentiability and continuity of the activation function. SDS is rather more convinient when it is implemented jointly with simulation models during optimisation i.e. training process. For more complex ANN processes (number of dimension more the ten) almost all SDS algorithms exert better convergence performance compared to BPE methods[5]. The Fig. 3 shows diagram of MN-SDS numerical procedure.

Design Problems-ANN Synthesis

The synthesis problem of an ANN is rather complex one. On the other hand the said problem is not always formalised. Apart of intentions of various researchers to reach an answer mostly partial solutions have been offered for some specific ANN architetures. Somehow the experience of designers has crucial role in an ANN architeture choice related to neurel networks. In this chapter a simplified instruction for FANN choice are presented having under certain anticipations sourcing from theory of universal aproximator[16-20]. In Fig. 4 one form of universal approximator (UA) is presented.

If we, for a network with r inputs and m outputs assume number of neurons layerwise, than there are N_ω interlinks between neurons in that network and N_θ bias. A simplified assessment of required thraining samples based on theoretic results for UA[18,20] is as follows:

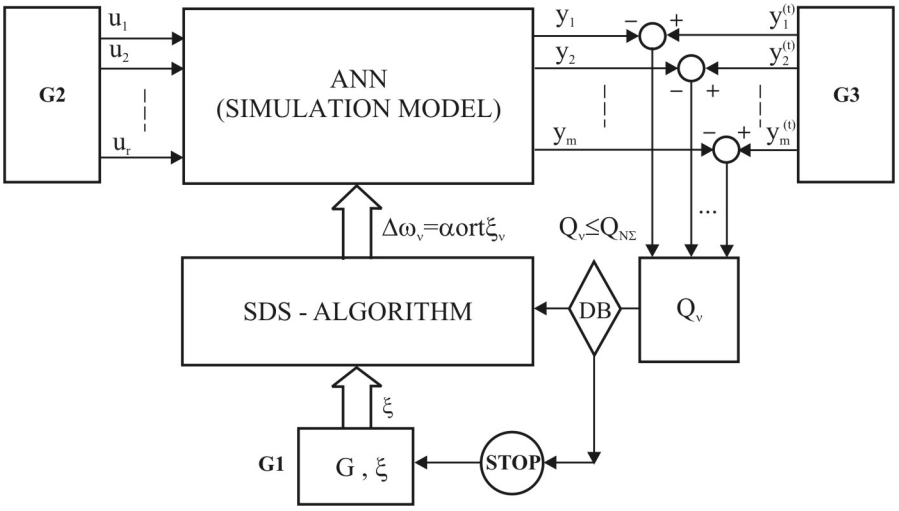

Fig. 5: Training ANN by aplying SDS algorithm and simulation

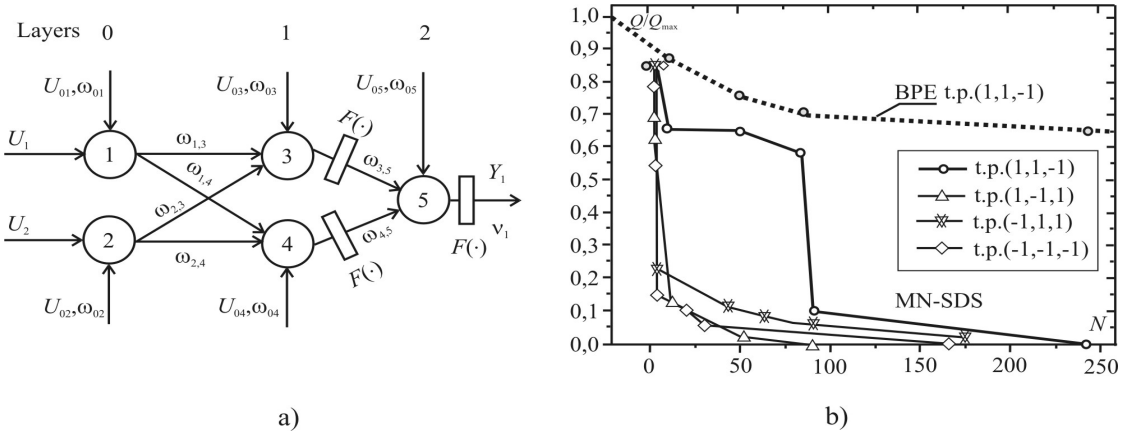

Fig. 6: Multy layers FANN a) Cost function for g1(x)- step function achieved by MN-SDS; BPE training for training par tp(1,1,-1) is finished after 600 iteration b)

$$N_p = 10\,(N_\omega + N_\theta) \quad \ldots(8)$$

where is: N_p presents number of training samples in a set P required to have network generalization yield more then 90%, under constrains:

$$(r + N_{vh}) \gg m \quad \text{and} \quad C = N_\omega/m, \quad \ldots(9)$$

N_{vh} is a number of hidden neurons, C is a network capacity. It is not ease to satisfy condition for set of training samples. Mostly number of samples Np is not always enough. If there is a limit for samples number than is decrease i.e. unless relations given in (8) & (9) are settled. It is not expected to have neither unique & optimal solution since there is no limit for distribution of hidden neurons. Relation (9) indicate that there is no unique solution. It is not to be assumed as defficiency since there is enough space for optional solution in FANN architecture. For some other type of networks there is no an appropriate guideline as previously indicated.

An Application of Simulation Models in ANN Training

An application of simulation methods with iterative algoritms enables avoiding of mass calculation specifically when a network architecture

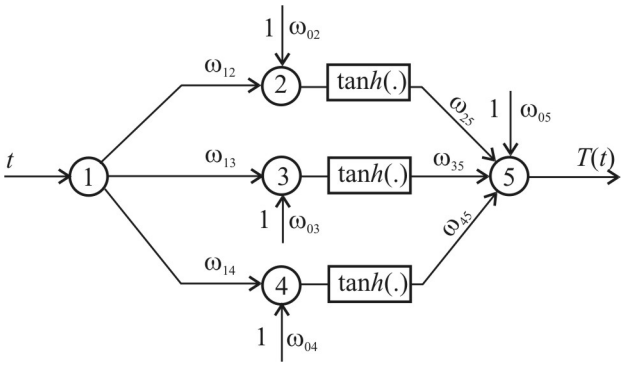

Fig. 7: Oriented graph of NN3' = (1 – 3 – 1)

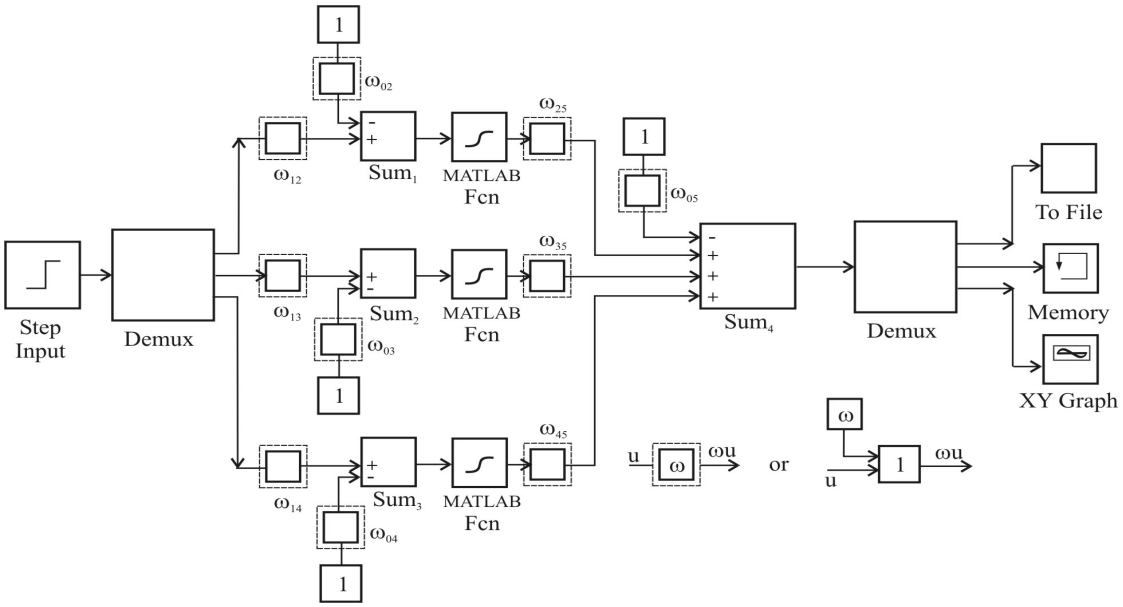

Fig. 8: SIMULINK simulation model neural networks of the FANN on Fig. 6

is slightly complex. Simulation models are more convinient whenever real-time processes are in question. One of most complex ANN dynamic training methods incorporate so called simulated annealing process[28]. The simulated annealing process is rather combinatorial complex method, but also rather useful in ANN synthesis. However, presented simulation approach here is conceptualy connected with the theory of processes control. Presentation ANN over graf oriented form is the most appropriate to translate in the simulation model.

From the conceptual point of view an application of simulation models with SDS algorithms is shown in the Fig. 5. In the Fig. 5 G1 is random numbers generator, G2 is input signals generator, G3 is output of training pairs signal generator, Q is a block of cost function. The decision block DB for an iteration step–stop is not necessary to be positioned as indicated in the Fig. 5. An iteration steps whenever the condition $Q_v \leq Q_{N\varepsilon}$ at $v=N\varepsilon$ i.e. min Q has been achieved.

In the Fig. 5 all elements of FANN network are replaced by appropriate components of simulation model. The reason to choose of MATLAB SIMULINK is facilitation of creation of simulation models both linear or non-linear functioning in real time. In addition to the aforesaid strong graphic environment and programming options for complex attempts[29].

EXAMPLES

Example 1:

Multy-layers FANN capable to reproduce the true table XOR circit.

An artificial neuron of a basic definition i.e. perceptron can not „recognize" XOR logic circuit. The previously said has a historical meaning[30]. If someone had used SDS algorithms than the neuro-science would not have stagnated almost 20 years.

The two layers perseptrons as per the Fig 6.a) with the two neurons in hidden layer and the output neuron gives FANN capable to reproduce the true table XOR. In order to point out some advantages of MN-SDS let us the hidden neurons have the activation function as

$$g_1(x)= \begin{cases} 1 \ , & x \geq 0 \\ 0 \ , & x < 0 \end{cases} \quad \ldots(10)$$

A direct implementation of BPE is not possible since $g_1(x)$ is not differenciable. The BPE implementation is possible if the function $g_1(x)$ is aproximated by the logistic function:

$$g_2(x)= 1/(1+\exp(-cx)); \quad c \geq 10 \quad \ldots(11)$$

Available training pairs to be used in training process give the true table of XOR:

$$\{P_k(u_{1,k}, u_{2,k}, y^{(t)}_{1,k}\} \rightarrow [P'_1(0,0,0), P_2(0,1,1), P_3(1,0,1), P_4(1,1,0)] \quad \ldots(12)$$

$$\{P'_k(u'_{1,k}, u'_{2,k}, y^{(t)}_{k}{}')\} \rightarrow [P'_1(-1,-1,-1), P'_2(-1,1;1), P'_3(1,-1;1), P'_4(1,1;-1)] \quad \ldots(13)$$

where is k=1,2,3,4.

Table 1:

t mjn	T0C
0	20
10	20
20	50
30	50
40	100
50	100
70	150
80	250
90	250
100	350
105	350
110	450
115	450
120	550
125	600
130	600
140	600
160	600
170	600
175	425
185	325
205	150
215	110
225	75
235	60
240	50

The stohastic search algorithm MN-SDS is aplayed to training ANN on Fig.6 a) over the series training pairs (13). For some variables in the Fig 6.a) a fixed values will be adopted.

$$U_{01}=U_{02}=0;\ U_{03}=U_{04}=U_{05}=-1;\ \omega_{01}=\omega_{02}=0\ .$$

Further on number of variable parameters (n) for ANN in the Fig 6.a) is n=9. The random numbers vector ξ would have the same dimension. Its components would have indexes of certain weights as per the Fig 6.a).

So, the random vector $\xi=(\xi_1, \xi_2,\ldots \ldots \xi_9)^T$; $|\xi q| \leq 1$; q=1,2,…n; n=9.

The searching starts with an initial random choice $\xi_0 = \omega_0$:

ω_0 = (0.873, 0.784, 0.953, 0.598, 1.013, 0.550, 0.740, 0.191, 0.573)^T.

Final values of components random vector after $N\varepsilon \geq 350$ iteration with SDS algorithm i.e. MN-SDS for epoch i.e.for complete training set P are:

$\omega_{N\varepsilon}$ =(0.999, 0.997, 0.995, 0.998, 1.503, 0.990, 0.511, -1.982, 0.982)^T,

So, final values weights for ANN from the Fig 6.a) after arrounding are:

$\omega_{N\varepsilon}$ = (1, 1, 1, 1, 1.50, 1, 0.50, - 2, 1)^T, i.e.
ω_{13} =1, ω_{14} =1 ω_{23} =1, ω_{24} =1, ω_{03} =1.50,
ω_{04} =1, ω_{05} =0.50, ω_{35} = -2, ω_{45} =1 …(14)

The achieved relative error is less than 2%.

By using sequence of training pairs $\{P'_k(u'_{1,k}, u'_{2,k})\}$ a better convergence is achieved during training process.

The trend of the cost functions behavior during the training process for each sample individually is shown in the Fig 6. b). In the same figure the cost function behavior is shown by use of BPE method for one single sample tp(1,1,-1). The trend of cost function for wholl epoch is similer. The optimization process i.e. training in that case is completed after $N'\varepsilon \geq 600$ iteration.

Example 2

An intelligent monitoring of technological processes

This example considers typical process related to numerous metallurgical & chemical technologies. Namely, it considers initial stages of tempering of agregates into rated production state and monitoring of situation when the agregate is down of regular operation as well. This case is related to tempering process of fluo reactors (FR) and smelters of metal raw materials. The tempering is usually recorded and measured data presented in the Table-1[27]. Data from the Table-1 can be used to create regresion model.

Instead of statistic methods it is possible to achieve over neuro models by using of so called universal approximator. Namely, it consider creation of three layer network with one input and one output. Since it contains complex interrelation it is advisable to start with the following architecture: 10 neurons (with linear activation function), 10 neurons in hidden layer (with non-linear activation function) and a neuron at output (with linear activation function). The aforesaid can be presented in the scheme NN10=(1-10-10-1). The first neuron at input operates as a demultiplexor. Since universal aproximator has the structure of FANN network that it is rather ease to get overall number of weights $N\theta$ =120 and overall number of thresholds $N\theta$= 21. If we refer to relations (2-9) & (2-10) for generalization more then 90% it is necessary Np= 1410 training samples what is more that contained in the Table-1 That is the reason to reduce number of neurons as per scheme NN5= (1-5-5-1). For the previous architecture the number of training samples is 460. Consequently next reduction bring to NN2=(1-2-2-1) claiming to have some 130 training samples not available in the Table-1. In order to overcome this problem and realize the ANN model additional training samples at each 2min (with supposing the tempereture in FR until the next measurement does not change or with linear incrising) are necessary what would provide some 125; we can say almost 130 samples. Having in minds conditions from (2-8) and (2-9) the structure NN'=(1-3-1) could fit the case

as shown in the Fig. 7. If for a nonlinearity activation function is used tanh(x), the final structure of ANN can be discribe with expression:

$$T(t)=\omega_{25}\tanh(\omega_{12}t+\omega_{02})+\omega_{35}\tanh(\omega_{13}t+\omega_{03})+\omega_{45}\tanh(\omega_{14}+\omega_{04})+\omega_{05} \quad ...(15)$$

By applying MN-SDS algorithm unknown parameters can be determined-weights ANN; in this case ther are ten:

$\xi = (\xi_1, \xi_2, \xi_3, \ldots\ldots\ldots, \xi_q)^T$,
$|\xi_q| \le 1$, q=1,2,…n, n=10.

By random selection is determined the start values of weights:

$$\xi_0 = \omega_0 = [0.1576,\ 0.9572,\ 0.4218,\ 0.8003,\ 0.7922,\ 0.9706,\ 0.9706,\ 0.4854,\ 0.1419,\ 0.9157]^T \quad ...(16)$$

After copmletition of the process optimization i.e. training for a whole epoch, the vector weights is:

$$\omega_{N\varepsilon} = [\ 1.705,\ 3.847,\ 11.61,\ 3.159,\ -3.597,\ 0.731,\ -1.670,\ -7.021,\ -13.23,\ 02939] \quad \ldots(17)$$

In the FIG. 8 it is presented a simulation model of the FANN on the Fig. 7. In conjunction with MN- SDS algorithm it can perform the training process. For discreet input values (independent of time) from the set of training sessions pairs mathematical procedure is relatively simple.

Example 3
Training recurrent neural networks with time-varying

The aim of this example is that on conceptual level through the fully connected RNN in the Fig.2 (Jordan's RNN), show that the use the coupling simulation model of network and SDS algorithms (according to the scheme on the Fig.5), formally is identical in procedures as in the case of training FANN. This approach to training is numerically simpler then numerical-analytical methods that use gradient of a cost function.

If inputs in RNN are time variable with duration T, network will posses dynamic behavior. This RNN is colled Real-Time RNN i.e. RTRNN[31]. Preform of input signals in discret form gives a series samples with ô repeat period; ô of sampling should be in accordance with the sampling theorem.

For discretized inputs may be indicated signals in RTRNN by:
- $u_i\ (n)$ - extern inputs,
- $u_j(n)$ - inputs of feedback link in network,
- $x_j(n)$ - interaction potencial for j neuron,
- $y_j(n+1)=g_j(x_j(n))$ - extern actual outputs of certain neurons ;with out stimulus in time interval [n,n+1].

The error between the current value $y_j(n)$ and the desired value $y_j^{(d)}(n)$ of actual output is

$$e_j(n)= y_j(n) - y_j^{(d)}(n) \quad \ldots(18)$$

The cost function Q is defined for one sample and all samples in the time interval T:

$$Q(n)= 1/2\ [e_j^2(n)] \quad \ldots(19)$$

and

$$Q_{tot} = \Sigma\ Q(n) \quad \ldots(20)$$

Analitical-numerical methods often for a step in procedure of optimization i.e. training use gradient of Q:

$$\Delta\omega = -\eta\ \nabla_w Q_{tot}\ ; \quad \ldots(21)$$

where $\nabla_w Q_{tot}$ is the gradient of Q_{tot}, $0 < \eta \le 1$,

where: W - is a matrix of parameters i.e weights; η -is coefficient of a promptness learnings.
In SDS i.e. MN-SDS algorithm an iteration step is defined by (2-5) i.e.:

$$\Delta\omega = \alpha\ \mathrm{ort}\xi;\ \mathrm{ort}\ \xi=\zeta,\ |\xi| \le 1,\ |\zeta| = 1$$

$0 < \alpha \le 1$, α - is coefficient of a promptness learning's.

Training of RTRNN based on the application of the MN-SDS and the simulation model of the same network (according to the Fig.5) is far simple and less subject to the phenomenom of unstable states than the case in analytical-numerical methods[32].

The RTRNN type given on Fig.2 has significant performances. Their implementation is

effective when it comes to projects prediction at non-stationary processes and phenomena[33]. In such cases training procedures are performed in the parameter space dimension of over 100.

Let us mention some numerous referring to the case when RTRNN on the Fig.2 reduced structure according to the following: the number of processing neurons is N=4, the number of line inputs is R=2 and the number of outputs is M=2. After the previous reducing the RNN on Fig.2, the new networks has:

$N^2+ NR = 4^2 + 4x2 = 24$ interface link,
$N^2+NR+N=4^2+4x2+4=28$ parameters (weights) to be determined in the process of training.

So, the random vector for applaying SDS is the same dimensions. With this level of dimensions SDS methods are more effective than BPE method ANN training.

The simulation model of reducing RTRNN is somewhat more complex than the model on Fig. 8. For RTRNN simulation it is need the SIMULINK with part for real-time processing.

In the case of optimization i.e. training a RTRNN shoud be taken appropriate scaling times T and $_T$ due to the relationship, to the time required for computing the optimization cycles.

DISCUSION

Synthesis of an ANN is simply linked to its designing. Somehow the problem of synthesis formalization is a still open issue. Generally speaking the synthesis is well facilitated based on designer experiences and numerous available computer tools. This paper suggests an approach facilitating at least initial estimation of ANN structures. The said approach is based on a stochastic direct search method, implementation of the theory of universal approximation and application of simulation models.

An implementation of SDS algorithms in an ANN designing process deserves to mention that there are certain facilitation in optimization i.e. training of an FANN. A FANN optimization by implementation of SDS algorithm is performed in forward stage. That means quite reduced job to be expected compared to BPE method. The BPE method require multilayer FANN structure but if not, than it should be transformed into FANN. For SDS method it is easeness but transformation is not ever necessary, having in minds the presented approach in this paper. Attempts in other types of ANN into FANN transformation are useful since it enables use of results based on theory of universal approximation.

Strict explanation of an universal approximation is related to binary mapping of network input-output[16]. It could be said that it is intuitively adopted that the said theory is valid whenever mapping of continuous signals of network input-output. The author experiences indicate that the theory of universal approximator is rather correct. Simplification of some theoretical results facilitates to understand the paradigm of network ability to learn i.e. exert some inteligent behavior with more-or-less success. It is worthwhile to mention that training through optimization does not mean that network possess required level of ability to learn. Certain level of generalization provides conditions that network can recognize a samples not used in the training process(of the same population). In this approach to post-inspection training FANN through set of pairs of test not necessery. The theory of universal approximatlonr supports some relations in the FANN structure, mostly when hidden neurons with non-linear activation function are in question.

The choice of SIMULINK simulation port of MATLAB package is governed by practical reasons having in minds that it supports consideration both linear and non-linear objects including its vast graphical environment. The simulation is useful when recurrent neurel networks are considered specifically in real-time operation. By working with a RTRNN condition instability in networks may appear. combination of SDS & simulation methods is quite something new in an ANN designing process what enrich the designer experience.

The examples in this paper are two aims: a) comparison SDS and BPE (the first example),

b) the others (two example) are chosen to illustrate the essence of this paper.

CONCLUSION

An application of stoshastic direct search method in design & training of artificial neuron networks offers a quite new approach having in minds numerical procedures efficiency vs known algorthms based on gradient features. It is well confirmed that for complex systems having unknown parameters more than ten, stochastic direct search shows remarkable advantages over gradient methods. In addition to the previous said SDS method is rather imune to presence of noice, ease iteration handling, adaptable against the problem nature and applicability to either systems with determined or stochastic mathematical description. This paper presents one new application of the MN-SDS algorithm, created by the author. The aforesaid integrates the properties of both non-linear algorithm and algorithm with accumulated information enabling to obtain self-learning ability.

The presented approach of design & training of neural networks is related so to the theory of universal aproximation. The said theory provides an insight into method of design and training of FANN, since it set up a conditions for hidden neurons having non-linear activation function in networks. Besides, the theory of univerzal approximation gives ration between number of interlinks vs the range of sinapses and scope of training samples. The simplicity presented in this paper provides rather accurate ratio and could be good orientation regarding starting assessment in a artifisial neural networks design.

The next feature of the presented approach is creation of straight interlink of stochastic direct search and simulation methods. The paper suggests implementation of SIMULINK method as software component of MATLAB package. The said package enables creation of linear and non-linear systems with lamp parameters with rich graphic environment. The SIMULINK provides options for presentation of neural networks in the form of oriented graph. The simulation model for the case of this form is easily acheieved regardless of the type of ANN. By carrying out the simulation process and MN-SDS for RTRNN unstable situation in network may take place. This last remark is a challenge for future researching. The obtained results out of numerical experiments related to the examples has been compared against BPE method and BPE through times. The comparison has shown the validity of results as well advantages of stochastic direct search. Numerical procedures has been supported by use of MATLAB R2014b, The Math Works Inc.

ACKNOWLEDGMENTS

"For software Matlab in numerical experiments I am grateful to Mr. Milorad Pascas Research Assistant at the ICEF of Electrical Faculty University of Belgrade".

REFERENCES

1. Rastrigin L. A., Stochastics Search Methods. In: Science Publishing, Moscow, Russia (1969).
2. Dvorezky A., On Stochastic Approximation, In: *Proc. III Berklly Symposium Math. Stat. and Probability,* Vol.1, University of California Press, 1956; Berkeley, Calif, USA,
3. Widrow B. and Hoff M.E., Adaptive switching circuits, *IRE WESCON Convention Record,* pp 96-104 (1960).
4. Maquardt D.W., An algorithm for least squares estimation of non-linear parameters, *J. Soc. Ind. Appl. Math*, **2**, pp. 431-441 (1963).
5. Rastrigin, L.A. Comparison of methods of gradient and stochastics search methods. In: *Stochastics Search Methods*, pp. 102-198. Sciece Publishing, 1968; Moscow, Russia
6. Rastrigin L.A. and Rubinshtain L.S., Comparison of stochastic search and stochastic approximation method. In: *The Theory and Application Stochastic Search Method*, Zinatne, Latvia, pp. 149-156 (1968).
7. Zhiglavsky A.A., Theory of Global Random Search. *Kluwer Academic,* Boston (1991).

8. Baba N., Shomen T. and Sovaragi Y., A modified Convergence Theorem for a Random Optimization Method. *Information Science*, **13**, pp.159 - 166 (1997).
9. Spall J. C., Introduction to Stochastic Search and Optimization: Estimation, *Simulation and Control Automation and Remote Control,* **26**, pp. 224- 251 (2003).
10. Nikolic K.P., An approach of random variables generation for an adaptive stochastic search. In: *Proceeding ETRAN 96,* Zlatibor, Serbia pp. 358-361.
11. Nikolic K.P., An identification of complex industrial systems by stochastic search method. In: *Proceeding ETAN 79*; Maribor, Vol III pp 179-186.
12. Nikoliæ K.P., An identification of non-linear objects of complex industrial systems. In: *Proceeding ETRAN 98*, Vrnjacka banja, Serbia, pp. 359-362.
13. Nikolic K.P., Neural networks in complex control systems and stochastic search algorithms. In*: Proceeding ETRAN 2009,* Bukovicka banja, Serbia vol.3, pp. 170-173
14. Nikolic K.P., Abramovic B., Neural networks synthesis by using of stochastic search methods. In: *Proceeding ETRAN* 2004, Cacak, 2004, Serbia pp. 115- .
15. Nikolic K.P., Abramovic B. and Scepanovic I.: An approach to Synthesis and Analisys of Complex Neural Network. In: *Proceeding of International Symposium NEUREL* Belgrade, 2006
16. Baum E.B., On the Capabilities of Multilayer Perceptrons, *Journal o*f *Complexity*, **4**., pp.193-215 (1988).
17. Baum E.E. and Haussler D.: What Size Net Gives Valid Generalization?, *Neural Computation*, **1**., pp.151-160 (1989).
18. Hornik K., Stinchomba M. and White H.: Universal Approximation of an Unknow Mapping and Its Devaritives Using Multilayer Feedforward Networks. *Neural Networks*, **3**, pp.551-560 (1990).
19. Hornik K., Approximation Capabilities of Multilayer Feedforward Networks*. Neural Networks*, **4**. pp.251-257 (1991).
20. Leshno M., Lin V.Y., Pinkus A. and Schockan S.: Multilayer Feedforward Networks with a Nonpolonomical Activation Function can Approximat any Function. *Neural Networks*, **6**, pp. 861-867 (1993).
21. Rumelhart D.E., Hinton G.E. and Williams R.J.: Learning Representation by Back-propagation Errors. *Nature*, **232**, pp 533-536 (1986).
22. McCulloch W.S. and Pitts W.: A logical Collums at Ideas Immanent in Neurons Activity. *Bull. Mathematical Biophisics*, **5**, pp.115-133 (1943).
23. Grossberg S.: Nonlinear Neural Networks: Principles, Mechanisms and Architectures. *Neural Networks*, **1**, pp 17-61 (1988).
24. Nelson N.N. and Illingworth W.T.: A Practical Guide to Neural Nets. Addison-Wesley Publishing Company, Inc., (1991).
25. P.J. Webros: Back-Propagation Time: What is does and how do it. *Proc. of IEEE 78*, pp.1950 (1960)
26. Haykin S.: Back-Propagation Throught Time. In: Ne*ural* Networks (A Comprehensive Foundation), Macmillan College Publishing Company, 1994. New York, pp. 520-521
27. Nikolic K.P.: Stochastic Search Algorithms for Identification, Optimization and Training of Artificial Neural Networks. *International. Journal AANS,Hindawi*,2015;http://www.hindawi.com/journals/aans/2015/931379/
28. Kirkpatrick S., Gellat C.D. and Vecchi M.P.: Optimization by Simulated Annealing. *Science*, **220** No. 4598, pp. 671-680 (1983)
29. MATLAB 2014b, The MATH WORKS Inc, 2015
30. Minsky M. and Pappert S.: Perceptrons. In: *An* Introduction *to Computational Geometry*, MIT Press, 1969; Cembridge, Mass.
31. Haykin S.: Real-Time Recurrent Networks. In: *Neural Networks* (A Comprehensive Foundation), Macmillan College Publishing Company, 1994; New York, pp. 521-526
32. Williams R.J. and Zipser D.: A learning algorithm for continually running fully recurrent neural networks. *Neural Computation 1*, pp. 270-280 (1989).
33. Goh S. L., Popovic D., Tanaka T. and Mandic D.: Complex-valued neurel Network schemes for online processing of wind signal. In*: Proceeding of International Symposium NEUREL*, 2004; Belgrade, pp. 249-253