NONLINEAR SYSTEM IDENTIFICATION BASED ON EVOLUTIONARY DYNAMIC NEURAL NETWORKS WITH HYBRID STRUCTURE

Lavinia Ferariu[1] and MihailVoicu[2]

"Gh. Asachi" Technical University of Iaşi
Department of Automatic Control and Industrial Informatics
Bd. D. Mangeron 53 A, RO – 700 050 Iaşi, Romania
Fax: +40-32-230751, e-mail: lferaru@ac.tuiasi.ro[1], mvoicu@ac.tuiasi.ro[2]

Abstract: The paper presents a novel dynamic neural architecture that allows a flexible and compact representation of the nonlinear processes. The suggested neural topology considers local internal recurrence and a heterogeneous structure of the hidden layer. It allows the cooperation between different types of hidden units, such as perceptrons, sigmoidal neurons with functional links, radial basis function structures and/or gaussian neurons with complex weights. An evolutionary multiobjective procedure assists the automatic design of appropriate neural networks. It searches for accurate neural models, characterised by good generalisation capabilities. The experiments reveal that the presented approach is suitable for system identification. *Copyright © 2005 IFAC*

Keywords: neural networks, multiobjective optimisation, genetic algorithms, system identification.

1. INTRODUCTION

The **M**apping **A**rtificial **N**eural **N**etworks (MANNs) have become attractive tools for nonlinear systems identification, due their universal approximation capabilities (Isermann, *et al*., 1997; Haykin, 1999). In order to approximate dynamic nonlinearities, the static neural topologies must be combined with external or internal dynamic blocks, which implement the memory of the model. The later configurations are more difficult to design, but they are able to perform better approximations of the processes' behaviours.

The neural models should optimally satisfy a number of objectives involving accuracy and model parsimony. Their quality highly depends on the training data, the model structure assumptions and the performances of the learning procedure. The selection of optimal neural architectures and optimal parameters represents a complex problem, for which none method can lead to good results for a large variety of practical situations. Though, the evolutionary techniques represent a promising alternative, due the fact that they can efficiently solve complex nonlinear optimisations, being able to cope with ill - behaved problem domains, multimodality, discontinuity, time - variance and noise (Bäck, *et al*., 2000; Fleming and Purshouse, 2002). Based on a stochastic search and on mechanisms similar to biological evolution, the evolutionary algorithms work on a population of possible solutions. At each generation, the best solutions are encouraged to survive and to produce new points from the search space.

The multitude of computational models illustrates that none architecture can be uniformly better than the others. For each type of neural topology, the computational efficiency of the learning procedure and the generalisation capabilities of the model depend on the particular problem to solve.

The paper suggests a novel methodology, which combines the advantages of different neural structures, in order to build flexible and compact models, characterised by higher adaptation capabilities. The new generalised neural architecture contains local internal dynamic blocks and a heterogeneous hidden layer, including gaussian neurons with real or complex parameters, perceptrons and sigmoidal neurons with functional links. An evolutionary method automatically selects convenient neural topologies and parameters. The problem is formulated as a multiobjective optimisation and special mechanisms are used for an efficient exploration of the search space.

The paper is organized as follows. Section 2 describes the static structural elements included in the proposed hybrid neural architecture and section 3 discusses the extension to the dynamic topology. Details regarding the evolutionary design algorithm are given in section 4. Section 5 investigates the applicability of the approach within the framework of system identification problems. Finally, in section 6, several conclusions are presented.

2. STATIC NEURAL NETWORKS WITH HYBRID HIDDEN LAYER

For the sake of simplicity, firstly, the **S**tatic **N**eural **N**etworks with **H**ybrid **H**idden layer (SNNHHs) are introduced. These generalised neural structures offer a higher potential to perform good approximations, for a large variety of applications.

The proposed neural topology results as an aggregation between neurons characterised by global and local responses. Both simple and complex structural elements are considered, allowing an efficient tuning of the neural topology. The hidden neurons with a global response extrapolate the region beyond the interval where the training data were acquired. In contrast, the output of a localised neuron is nonzero if the inputs belong to a small region of the input space. Beyond this region, the response of the neuron is zero (Haykin, 1999). The mixed structure can take advantage from the generalisation capabilities of the neurons with global response, recommended for interpolation problems, and from the computational efficiency of the localised neurons, useful for extrapolation problems.

The neural structures accepted for the hidden layer are indicated in sequel. The approach can be easily extended in order to accept other neural structures. As neurons with global response, the **S**tandard **P**erceptron (SP) and the **S**igmoidal neuron with trigonometric **F**unctional **L**inks (SFL) are considered. The later configuration results by functionally expanding the initial inputs of a perceptron, using a sub - set of orthogonal trigonometric functions (Patra, *et al.*, 1999). The trigonometric functions guarantee a compact representation (Patra, *et al.*, 1999). For a predefined order of functional expansion P, the input of the sigmoidal activation function results as a weighted sum computed between the terms x_i, $\cos(p\pi x_i)$, $\sin(p\pi x_i)$, with $p=1,..,P$, $i=1,..,R$, where x_i denotes the initial, unexpanded ith input of the neuron. In fact, the functional expansions increase the dimensionality of the input pattern and thus, the identification of complex nonlinear functions becomes simpler. For several study cases, Patra *et al.* (1999) indicate better overall performances for the SFL structures than for the SP neurons.

The localised structures included into the hybrid architecture are the **S**tandard **G**aussian neuron (SG) with real parameters (centres and spreads) and the structures with **R**adial **B**asis function and **C**omplex weights (RBC), described by the following input - output mapping (Igelnik, *et al.*, 2001):

$$y_n = e^{-(w_n)^2\left[\left(\sum_{i=1}^{R}\cos\theta_{ni}\cdot z_{ni}\right)^2+\left(\sum_{i=1}^{R}\sin\theta_{ni}\cdot z_{ni}\right)^2\right]}, \quad (1)$$

with $z_{ni}=x_i-c_{ni}$. Here y_n represents the output of the neuron n, x_i ($i=1,..,R$) denote the inputs of the neuron, c_{ni}, $w_n e^{j\theta_{ni}}$ (with $j=\sqrt{-1}$) indicate the centre and the complex weight associated to the ith input connection of the neuron n, respectively. For certain applications, recent studies reveal some advantages of the neural structures with complex weights: a decreasing of the computational time and better approximation capabilities than the nets characterised by real weights.

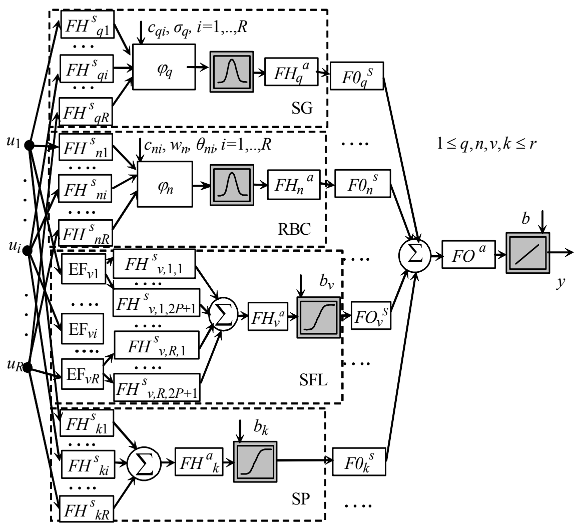


Fig. 1: The topology of DNNHHs (R inputs, maximum r hidden neurons, one output). Here, EF denote the functional extension blocks (SFL neurons), φ and c indicate the input operator and the centres of the localised neurons, respectively, b represents the bias (SP, SFL and output linear neuron), σ indicates the spread (SG neuron) and θ, w denote the phase and the magnitude of the complex weights (RBC neurons).

3. DYNAMIC NEURAL NETWORKS WITH HYBRID HIDDEN LAYER

For providing a better approximation of the dynamic nonlinearities, extension to **D**ynamic **N**eural **N**etworks with **H**ybrid **H**idden layer (DNNHHs) is considered. It results by including **A**uto - **R**egressive **M**oving **A**verage (ARMA) filters into the static topology. The ARMA filters placed on the input connections of a neuron implement the local synaptic feedback and the ARMA filters placed before/after the activation functions of the neurons implement a local activation feedback (Isermann, *et al.*, 1997). The internal dynamic blocks are denoted as follows (Fig. 1): FH^s_{ni}, FO^s_n, n=1,…,r, i=1,…,R represent the synaptic filters corresponding to the ith input connection of the nth hidden neuron and to the nth input connection of the output neuron, respectively; FH^a_n, FO^a denote the activation filters for the nth hidden neuron and for the output neuron, respectively.

The ARMA filters can reduce the level of noise that affects the neural inputs and memorize the necessary past states of the neural network. As consequence, an important reduction of input space dimensionality can be obtained, because it is not necessary to extend the neural input with lagged measurements. The neural input vector has to include the currents inputs of the process and the plant outputs measured at the previous sample time. Moreover, no *a priori* information about the process dynamic orders and the process dead time is required.

The dynamic architecture has to include only stable ARMA filters, characterised by zeros and poles which can be correctly sampled subject to the considered training data set, in order to prevent the achievement of unexpected behaviours, in the case of unlearned input data. The suggested design procedure eliminates all inconvenient zeros and poles, without changing the static gain of the filter.

DNNHHs can provide a compact representation of the dynamic processes. Taking into account the maximum time - delays implemented according to the dynamic architecture indicated in Fig. 1, DNNHHs have to be compared with the SNNHHs having $8R$ inputs. In some situations (e. g. $R > 2$ and $r > 1$), the dynamic topology requires a smaller number of parameters for any configuration of the hidden layer. Moreover, the design algorithm described in the following guarantees a significant reduction of the total number of parameters, encouraging the selection of simpler topologies.

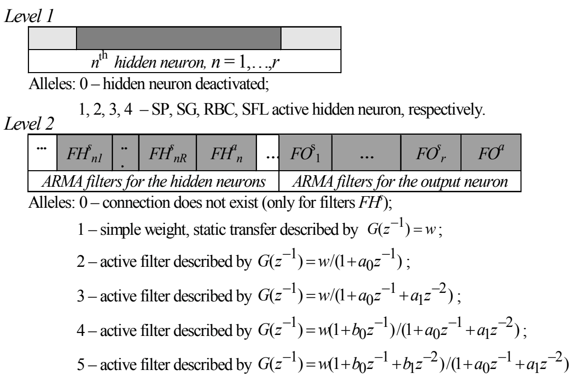
3. EVOLUTIONARY DESIGN OF DNNHHs

The present approach considers an evolutionary design procedure, in order to search for the optimal DNNHHs topologies and parameters. It allows a flexible configuration of the heterogeneous neural architecture. One considers neural topologies for which the hidden layer can include any combination of SG, RBC, SP, SFL neural structures, the input layer is not fully connected with the hidden neurons, not all permitted dynamic structures are compulsory and different complexity orders for the internal ARMA filters are permitted. The static neural topologies and the homogeneous structures of the hidden layer are also accepted.

A difficult task to solve is the selection of an appropriate neural architecture. It requires large computational resources, due the fact that, for each analysed topology, a convenient set of parameters has to be computed. Thus, the suggested evolutionary design procedure is divided into two stages: the first stage uses evolutionary mechanisms for the selection of optimal neural topology and a fast backpropagation algorithm for a preliminary computation of the neural parameters; at the second stage, an improved learning procedure is applied on a reduced set of selected architectures.

First stage of the design procedure. At each iteration, the evolutionary algorithm acts on a population of *Nind* individuals, each individual encoding a possible DNNHH architecture. An efficient exploration of the search space is performed using two – level hierarchical chromosomes (Fig. 2). The highest priority level of the chromosome (level 1) specifies the type of the hidden neurons included in the encoded neural topology. The second priority level indicates the active dynamic filters and their complexity. The control genes included in level 1 can activate (when their value is nonzero) or deactivate (when their value is 0) the corresponding parametric genes contained in level 2.

If an individual encodes an incorrect topology, remedy actions are applied. For the considered encoding, the repair procedure has to verify that the hidden layer includes at least one hidden neuron, each input is connected to at least one hidden neuron and each hidden neuron is connected to at least one input of the network.



Level 1
n^{th} hidden neuron, $n = 1,\ldots,r$
Alleles: 0 – hidden neuron deactivated;
1, 2, 3, 4 – SP, SG, RBC, SFL active hidden neuron, respectively.
Level 2
…
FH^{s}_{n1}
FH^{s}_{nR}
FH^{a}_{n}
…
FO^{s}_{1}
…
FO^{s}_{r}
FO^{a}
ARMA filters for the hidden neurons
ARMA filters for the output neuron
Alleles: 0 – connection does not exist (only for filters FH^{s});
1 – simple weight, static transfer described by $G(z^{-1}) = w$;
2 – active filter described by $G(z^{-1}) = w/(1+a_0z^{-1})$;
3 – active filter described by $G(z^{-1}) = w/(1+a_0z^{-1}+a_1z^{-2})$;
4 – active filter described by $G(z^{-1}) = w(1+b_0z^{-1})/(1+a_0z^{-1}+a_1z^{-2})$;
5 – active filter described by $G(z^{-1}) = w(1+b_0z^{-1}+b_1z^{-2})/(1+a_0z^{-1}+a_1z^{-2})$

Fig. 2: Hierarchical encoding of DNNHHs topology.

For preventing the competing conventions, the offspring are generated using mutation operator (Bäck, *et al.*, 2000).

The DNNHHs design is formulated as a multiobjective optimisation. Six objective functions are considered, assigned with different priorities. The highest priority objective function f_1, namely the total output squared errors computed for the normalised training data set, indicates the neural model accuracy. Its values are obtained after applying, in sequel, the extended real - time recurrent backpropagation algorithm (for a small number of epochs) and the ARMA filters' repair procedure (for eliminating the inconvenient zeros and poles, as indicated in Section 2). The other objective functions, having the same low - level priority, describe the complexity order of the neural architecture and allow the selection of simple neural models, with expected good generalisation capabilities. They are configured taking into account the functionality and the complexity of each permitted structural block: f_2 - the number of active hidden neurons; f_3 - the number of active synaptic dynamic blocks; f_4, f_5 - the number of parameters required by all synaptic and activation blocks (static or dynamic), respectively; f_6 - the number of active input connections corresponding to the hidden layer.

The multiobjective optimisation algorithm considers a progressive articulation between the search procedure and the decision mechanism (Fonseca, and Fleming, 1998; Marcu, *et al.*, 1999; Deb, 2001). A goal is associated to each objective function. The goals, defining the desired area for the objective values, are adapted according to the mean performances of the current population. Special Pareto - ranking techniques are used, in order to encourage the survival of accurate models, characterised by simple architectures (Ferariu and Marcu, 2002). If a neural topology satisfies all imposed goals, its rank is computed according to f_1, otherwise a Pareto - ranking procedure is considered.

Also, a convenient migration strategy is implemented (Ferariu and Marcu, 2002). The design procedure considers a supplementary auxiliary population, which evolves cvasi - independently, subject to the highest priority objective. Once at *No_migr* generations, an exchange of information is permitted between the two populations. Thus, the genetic material of the main population is enriched with more accurate models and a significant decreasing of the highest priority goal is achieved. The algorithm encourages the survival and the duplication of accurate models, whilst maintaining an adequate complexity order of the encoded topology.

Second stage of the design procedure. At the end of the evolutionary loop, the neural architectures of the main population are supplementary trained using a hybrid supervised learning procedure, denoted COMT. It switches for *N_com* times from an extended backpropagation procedure to a standard genetic search. The robustness of the evolutionary search offers greater chances to prevent the locking into local optimum points and the gradient - based method improves the convergence speed of COMT. Each individual of the evolutionary algorithm encodes a set of neural parameters, corresponding to the analysed topology. The offspring are generated by arithmetic recombination and mutation, allowing only small variations of the parents' genetic material.

A schematic description of the design algorithm is presented in the following:

```
1. Create the initial main and auxiliary population with Nind individuals (random uniform distribution). Correct the architectures. Initialise the goals.
2. Train the neural networks encoded into the population using an extended backpropagation algorithm (for $\alpha$ epochs). Correct the ARMA filters.
3. Evaluate the chromosomes according to all considered objectives and compute the fitness values.
4. Loop over a number of Max_gen generations:
  4.1 For the main and the auxiliary population:
    4.1.1. Select parents for the reproduction pool.
    4.1.2. Apply mutation operator. Correct the offspring (if necessary).
    4.1.3. Train the neural networks encoded by the offspring using backpropagation procedure ($\alpha$ epochs). Correct the ARMA filters.
    4.1.4. Evaluate the offspring and compute the fitness values.
    4.1.5. Insert the offspring into the population, according to the Pareto reservation strategy.
    4.1.6. Once at No_migr generations, exchange individuals with the other subpopulation (migration stage).
    4.1.7. Adapt the goals. Compute the fitness values.
5. Train all individuals of the main population with COMT procedure. Correct the ARMA filters.
6. Select the best individual(s).
```

4. APPLICATION

The applicability of the suggested method is studied with respect to the neural identification of two systems, characterised by different levels of complexity: the laboratory set - up "Three Tank system" (Amira DTS 200) and an industrial system, namely the first section of an **E**vaporation **S**tation (ES) from the Sugar factory of Lublin, Poland.

The experimental set - up Amira DTS 200 (Amira, 1993) consists of three cylindrical tanks with identical cross sections, being filled with water (Fig. 3). The tanks are interconnected with circular pipes. All three tanks are equipped with piezo - resistive pressure transducers for measuring the level of the liquid. The volume flows of lateral tanks ($q_1(t)$ and $q_2(t)$) represent the two inputs of the system. Three system outputs are considered, namely the liquid levels in the tanks. Here, t stands for the time variable. A nonlinear analytical model of this plant is available, but it offers a limited approximation.

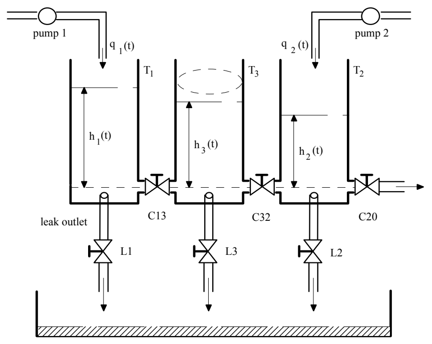


Fig. 3. The "Three Tank System" AMIRA DTS 200.

For the experiments, the reference values of the liquid levels were changed pulse - wise, using different magnitudes and periods of rectangular pulses for each controlled tank. The input - output data of the process were sampled at every T_S = 5s, during a test period of 400s. The sample time, selected by trial, permits the acquisition of representative measurements.

The identification task is done for the normal system behaviour (outlets L_1, L_2, L_3 closed and valves C_{13}, C_{32}, C_{20} opened). For the model validation, one considers 34 testing data sets acquired in different days of plant exploitation, in order to completely illustrate the influence of the system environment.

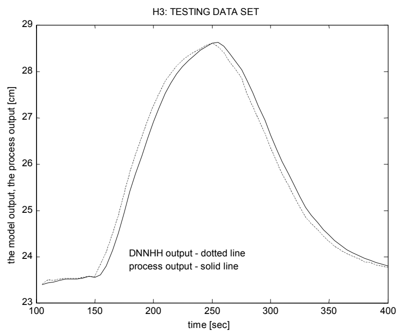
To estimate each process output, DNNHHs with 5 inputs and one output has to be designed. In all investigations a reduced number of hidden neurons was sufficient, i.e. $r = 3$. Several experiments were necessary for tuning the parameters of the evolutionary design procedure. Early migration between the main and the auxiliary population can allow a premature exchange of information, with negative effect on the exploration capabilities of the algorithm. Also, if the evolutionary process works on insufficiently trained networks, the results are unsatisfactory. Some of the resulted topologies are analysed in the following. They were obtained considering for the first stage $Nind = 32$,

$Max\_gen = 30$, $No\_migr = 10$, $\alpha = 200$, and for the second stage $N\_com = 4$ switches between the genetic search and the backpropagation procedure.

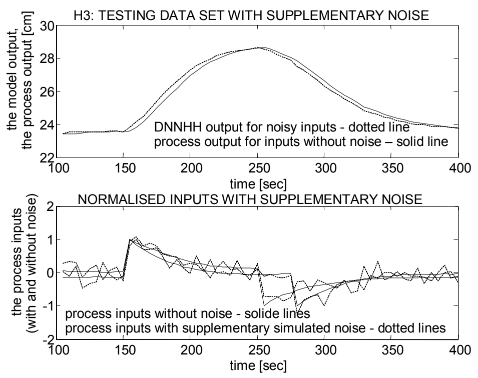
For the DNNHH estimating h_3, some details are given. The selected topology includes 2 hidden neurons (1SP, 1 SG) with 6 dynamic synaptic filters and 1 output neuron with 1 dynamic synaptic filter ($f_2 = 2$, $f_3 = 7$, $f_4 = 22$, $f_5 = 2$, $f_6 = 8$). The performances of the resulted neural model are presented in Table 1 (the first line). Here *PN* represents the total number of parameters, *SSEr* indicates the total output squared error computed for the data sets without normalisation and $RE_{\max}$ denotes the maximum value of the relative error, computed as follows:

$$RE = \frac{y - \hat{y}}{y} \cdot 100\%, \qquad (2)$$

where $\hat{y}$ represents the estimation of the process output y. The testing data set considered for this analysis is characterised by the worst approximation. The generalisation capabilities are also illustrated in Fig. 4a, with respect to the same testing data set. As indicated in Fig. 4b, the selected model guarantees a good rejection of the supplementary simulated white noise, acting at the input of the process.



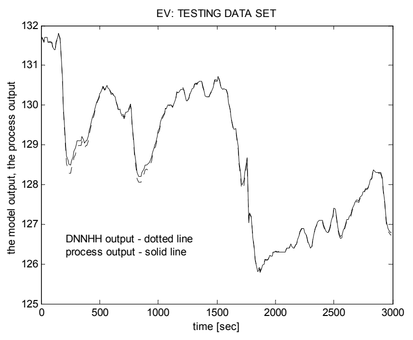
a)



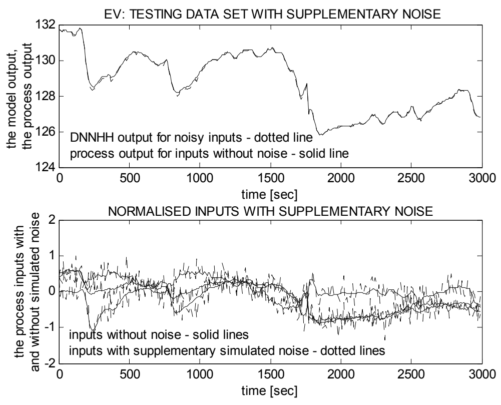
b)

Fig. 4. Model validation (h_3) - testing data set with the worst approximation: a) the measured data set; b) the data set with supplementary simulated additive white noise, acting on the inputs of the process (mean 0, standard deviation 0.2 for inputs scaled in [-1,1]).

The experiments are repeated for the identification of the industrial system. The ES has to increase the concentration of the sucrose juice (Bartys and Wasiewicz, 1998). The thin juice passes, in sequence, through all five sections of the ES, each one reducing the water content. Due to its complexity, the process is decomposed in several sub - processes. One of them, namely the **ev**aporator [EV] is identified using DNNHHs. It has three inputs (the steam flow to the input of ES, the steam temperature at the input of ES and the juice temperature after heater) and one output (the juice temperature after section 1 of ES). The model is designed using real data collected from the sugar factory during one month of plant exploitation, using the sample period *Ts* =10 sec. The selected learning data set contains 3000 rows and corresponds to a production shift. It illustrates the maximum possible excitation of the process and it includes a reduced number of missing or uncertain values. The isolated missing and uncertain values have been replaced by means of polynomial interpolation. In order to reduce the noise, a low - pass filtering, based on 4th order Butterworth filters, has been performed. This also allows the reduction of the amount of data used during the learning stage. The data have been decimated using each 10th sampled value. The validation of the neural model is done with respect to another testing data set, which includes measurements acquired from the previous month of plant exploitation.



a)



b)

Fig. 5. Model validation (EV) - testing data set: a) the acquired testing data set; b) the testing data set with supplementary additive simulated white noise acting on the inputs of the process (mean 0, standard deviation 0.2 for inputs scaled in [-1,1]).

Table 1. The performances of the DNNHH models estimating the output h_3 (AMIRA DTS 200) and the output of the evaporator system

		training data set		testing data set	
	PN	*SSEr*	RE_{max}	*SSEr*	RE_{max}
h_3	27	0.0444	0.26%	2.718	1.65%
EV	14	0.3412	0.009%	1.1087	1.7%

A set of preliminary experiments was carried out, in order to find appropriate values for all parameters of the design procedure ($Nind = 64$, $Max\_gen = 60$, $No\_migr = 20$, $\alpha = 200$, $N\_com = 4$). Table 1 (second line) indicates that the DNNHHs models are characterized by good accuracy and have good generalization capabilities. The architecture of the selected neural model includes 1 SP hidden neuron with 1 active synaptic filter and 1 output neuron with 1 active synaptic filter ($f_2 = 1$, $f_3 = 2$, $f_4 = 10$, $f_5 = 2$, $f_6 = 4$). Fig. 5 illustrates that the DNNHH can perform a good approximation of the testing data set, even if supplementary additive simulated white noise is considered on the process inputs.

Table 2 compares the DNNHHs with other dynamic neural models based on local internal feedbacks. Here f_1 denotes the squared output error computed for the normalised training data set and *PN* indicates the number of neural parameters. The DMLPs contain SP hidden neurons and accept supplementary lateral connections between the hidden neurons (Marcu *et al.*, 1999); the DGNNs include both SP and SG hidden neurons (Ferariu and Marcu, 2002); the DCWNNs include RBC hidden neurons (Ferariu, 2003). The last two architectures contain output ARMA filters, placed on recurrent connections provided from the output of the neurons to the input of their activation function. The design methodology based on DNNHHs selects the convenient models with respect to objectives involving accuracy and parsimony. Though, this general design approach can be overtaken by domain specific methods, for certain particular cases.

Table 2. Comparison with other dynamic neural models.

Model	h_3		EV	
	PN	f_1	*PN*	f_1
DNNHH	27	0.0065	14	0.015
DMLP	21	0.0168	24	0.0348
DGNN	31	0.0044	26	0.0312
DCWNN	-	-	21	0.0316

Even the methodology is based on computationally intensive evolutionary mechanisms, because it provides the automatic selection of the neural architectures, the required design time results much smaller than in the case of a manual configuration of the neural topologies (several hours instead of several days, for the considered study cases).

5. CONCLUSIONS

The paper presents a novel dynamic neural architecture, characterised by a hybrid structure of the hidden layer. It combines the advantages of all component neural structures, offering improved approximation capabilities. The heterogeneous neural topology and the local internal dynamic blocks are adapted according to the dynamic characteristics of the system that has to be identified. No model structure assumptions are required.

The experimental results indicate that the proposed methodology is able to produce neural models with good performances of approximation and generalisation. Though, the approach needs large computational resources, so it is recommended for nonlinear identifications, if poor *a priori* information about the models is available and/or high performances of the neural models are requested.

Further research will investigate the efficiency of the fault diagnosis systems based on DNNHH observer schemes.

REFERENCES

Amira (1993). *Laboratory Experiment Three-Tank System*. Amira GmbH, Duisburg, Germany.

Bäck, T., D. Fogel and Z Michalewicz. (2000). *Evolutionary Computation 2. Advanced Algorithms and Operators*. Institute of Physics Publishing, USA.

Bartys, M. and P. Wasiewicz (1998). Description of Sugar Technology Process and Artificial Fault Generation. *Prep. EC INCO - Copernicus Workshop IQ2FD*, Kazimierz, Poland, pp.17-32.

Deb, K. (2001). *Multi - Objective Optimization using Evolutionary Algorithms*. Wiley, USA.

Ferariu, L. and T. Marcu. (2002). Evolutionary Design of Dynamic Neural Networks Applied to System Identification. *Proc. of IFAC Congress b'02*, Barcelona, Spain, CDROM-2110.

Ferariu, L. (2003). Nonlinear System Identification Based on Evolutionary Dynamic Neural Networks with Complex Weights. In: *Proc. of European Control Conference, ECC'03*, Cambridge, UK, 2003, CDROM - 239.

Fleming, P. J. and R. C. Purshouse (2002). Evolutionary algorithms in control systems engineering: a survey. *Control Engineering Practice*, **10**, 1223 - 1241.

Fonseca, C.M. and P. J. Fleming (1998). Multiobjective Optimisation and Multiple Constraint Handling with Evolutionary Algorithms – Part I: A Unified Formulation. *IEEE Transactions on Systems, Man, and Cybernetics – Part A*, **28** (1), 26-37.

Haykin S. (1999). *Neural Networks - A Comprehensive Foundation.* McMillan College Publishing Company, New York, 2nd Edition.

Igelnik B, M. Tabib-Azar and S. R. LeClair (2001). A Net with Complex Weights. *IEEE Transactions on Neural Networks*, **12** (2), 236 – 249.

Isermann, R., S. Ernst and O. Nelles (1997). Identification with Dynamic Neural Networks: Architectures, Comparisons, Applications. In: *Preprints of IFAC Symposium on System Identification,* Fukuoka, Japan, Vol.3, pp. 997-1022.

Marcu, T., L. Ferariu and P. M. Frank (1999). Genetic Evolving of Dynamic Neural Networks with Application to Process Fault Diagnosis. In: *Proceedings of European Control Conference*, Karlsruhe, Germany, CD-ROM - F1046-1.

Patra, J. C., R. N Pal., B. N. Chatterji, G. Panda (1999). Identification of Nonlinear Dynamic Systems Using Functional Link Arfificial Neural Networks. *IEEE Transactions on System, Man and Cybernetics – Part B*, **29**, 2, 254-262.