An Adaptive Resource Allocating Neural Fuzzy Inference System

Minas Pertselakis, Nicolas Tsapatsoulis, Stefanos Kollias and Andreas Stafylopatis

***Abstract*—Adaptivity to non-stationary contexts is a very important property for intelligent systems in general, as well as to a variety of applications of knowledge based systems in the area of Electric Power Systems. In this paper we present an innovative Neural-Fuzzy architecture that exhibits three important properties: online adaptation, knowledge (rule) modeling, and knowledge extraction from numerical data. The ARANFIS (Adaptive Resource Allocating Neural Fuzzy Inference System) has an adaptive structure, which is formed during the training process. We show that the resource allocating methodology enables both online adaptation and rule extraction; the latter differentiates it from the majority of Neurofuzzy systems with fixed structure which perform mainly rule modification/ adaptation rather that rule extraction. The efficiency of the system has been tested on both publicly available data, as well as on a real generated dataset of a 120 MW power plant.**

***Index Terms*-- Knowledge based systems, online adaptation, data-driven knowledge extraction, resource allocation.**

I. Introduction

In the era of ambient intelligence, which is rapidly approaching, if not started, data capture and sensor technologies will be generating enormous amounts of data. Applications and interfaces that will be able to automatically analyze these data, exchange knowledge and make decisions in a given context are strongly desirable. Natural and enjoyable user interactions with such applications will be based on autonomy, avoiding the need for the user to control every action, and adaptivity, so that they are contextualised and personalized, delivering the right information/decision at the right moment.

Information sources contain: (a) raw numerical data obtained through sensors and devices which collect and pre-process data, and (b) knowledge bases (such as rule-based systems, ontologies) for specific tasks; these are in the form of rules, concepts and symbols. This category may also contain databases and databanks, such as data repositories which have been examined and annotated /characterized by experts. What is missing is the appropriate technology for effectively linking these two different types of symbolic and subsymbolic information, in real life situations.

Integrated fuzzy neural models demonstrate the ability to operate and adapt in both numeric and linguistic environments. Many of such hybrid systems have been proposed in the literature [1], including models for embedding a priori knowledge, handling numeric and linguistic features simultaneously, extracting data driven knowledge and interpreting rules [2-8].

When functioning in an environment with non-stationary contexts both online training and adaptation are critical issues. Online adaptation during normal operation is a very complex problem because target outputs are not available. The problem is handled either by using reinforcement learning or semi-supervised techniques [9].

Resource Allocating Network (RAN) architectures [10], were found to be suitable for online modeling of non-stationary processes. In this sequential learning method the network initially contains no hidden nodes. On incoming training examples, based on two criteria, the RAN is either grown or the existing network parameters are adjusted using a least mean square gradient descent. The first criterion is based on the prediction error while the second is the novelty criterion, which states that the distance between the observation and the winning rule should be greater than a threshold. If both the criteria are satisfied, then the data is memorized and a new hidden node is added to the network.

Online adaptation and rule extraction from numerical data is the key properties of ARANFIS. Our model combines a modified RAN structure with a fuzzy neural inference system [11], to exhibit data-driven knowledge extraction and online adaptation. This novel combination addresses adequately the low efficiency presented in fixed neural fuzzy networks.

The basic characteristics of ARANFIS are:

(a) It has a resource allocating architecture through which it exhibits dynamic behavior, necessary in non-stationary contexts

(b) It uses a tunable input fuzzifier that is responsible for fuzzification of numeric data. In other words, numeric inputs are fuzzified using a feature-specific Gaussian spread.

(c) All information that propagates from the input layer is fuzzy. The model therefore uses a composition mechanism that employs a fuzzy mutual subsethood measure to define the activation that propagates to a rule node along a fuzzy connection.

The authors belong to the Department of Electrical and Computer Engineering, NTUA, Athens, Greece.

Corresponding author: Minas Pertselakis (e-mail: mper@cslab.ntua.gr).

This work is partially funded by the European IST project "ORESTEIA-Modular Hybrid Systems with Adaptive Functionality", IST- 2000-26091, http://www.image.ntua.gr/oresteia

(d) It aggregates activities at a rule node using a *fuzzy inner product*: a product of mutual subsethoods, which is different from the most common approach to use a fuzzy *min* conjunction operator.

II. The Proposed Architecture

ARANFIS uses the architecture shown in Fig. 1. This architecture has the flexibility to handle both numeric and linguistic inputs simultaneously. Numeric inputs are fuzzified by input nodes, which act as tunable feature fuzzifiers, while connections in the network are represented by Gaussian membership functions specified by a center and a spread. Rule based knowledge is easily translated directly into the network architecture in the form of fuzzy *if-then* rules that are embedded as hidden nodes; rule antecedents as input to hidden connections and rule consequents as hidden to output connections. Knowledge in the form of *if-then* rules can be either derived from clustering numeric data or be embedded directly as a priori knowledge.

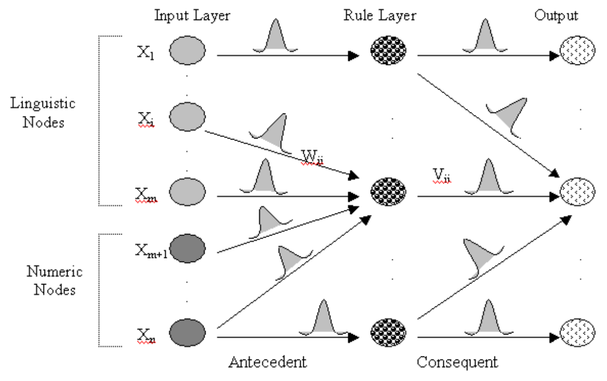

Fig. 1. The ARANFIS architecture

During the sequential learning procedure more hidden nodes can be added in case that the existing ones cannot represent the numerical data. This is clearly the case when functioning in non-stationary contexts. On the other hand proper initialization of the hidden layer is required in order to avoid creating hidden nodes that do not represent meaningful rules. In setting a priori knowledge it is easy to create the appropriate antecedent part of rule by forming connections between the input layer and the hidden nodes. However, when inserting new hidden nodes all connections between inputs and the newly created node should be made and pruned as the learning process evolves.

III. ARANFIS Learning

Learning is incorporated into ARANFIS using the gradient descent method. A squared error criterion is used as a training performance parameter. The squared error $e(t)$ at iteration t is computed in the standard way:

$$e(t) = \frac{1}{2}\sum_{k=1}^{p}(d_k(t) - y_k(t))^2 \qquad (1)$$

where $d_k(t)$ is the desired output and $y_k(t)$ the defuzzified output at node k given by (14). The error is evaluated over all p outputs for a specific pattern input $\underline{x(t)}$.

Fuzzy weights w_{ij} from input nodes i to rule nodes j are modeled by the center w^c_{ij} and spread w^σ_{ij} of a Gaussian fuzzy set and denoted by $w_{ij}=(w^c_{ij}, w^\sigma_{ij})$. In a similar fashion, consequent fuzzy weights from rule nodes j to output nodes k are denoted by $v_{jk} = (v^c_{ij}, v^\sigma_{ij})$. The spread of the i-th fuzzified input element is denoted as x^σ_i while x^c_i is obtained as the crisp value of the i-th input feature element.

The free parameters of the system, meaning both the centers and spreads of antecedent and consequent connections as well as the spreads of the input features, are modified on the basis of update equations taking the following forms:

$$w^c_{ij}(t+1) = w^c_{ij}(t) - \eta(t)\cdot a_j \frac{\partial e(t)}{\partial w^c_{ij}(t)} \qquad (2)$$

$$v^c_{jk}(t+1) = v^c_{jk}(t) - \eta(t)\cdot a_j \frac{\partial e(t)}{\partial v^c_{jk}(t)} \qquad (3)$$

$$w^\sigma_{ij}(t+1) = w^\sigma_{ij}(t) - \eta(t)\cdot a_j \frac{\partial e(t)}{\partial w^\sigma_{ij}(t)} \qquad (4)$$

$$v^\sigma_{jk}(t+1) = v^\sigma_{jk}(t) - \eta(t)\cdot a_j \frac{\partial e(t)}{\partial v^\sigma_{jk}(t)} \qquad (5)$$

$$x^\sigma_{i}(t+1) = x^\sigma_{i}(t) - \eta(t)\cdot a_j \frac{\partial e(t)}{\partial x^\sigma_{i}(t)} \qquad (6)$$

where $\eta(t)$ is the online computed learning rate (see Section II.B.2) and a_j is a parameter related with j-th hidden node, accounting for soft competitive learning (see Section II.B.3).

1) Evaluation of partial derivatives

In the following we consider that the network consists of n inputs, p outputs, and $q(t)$ hidden nodes at iteration t (since ARANFIS has a resource allocating structure).

Computing the partial derivatives required in the above update equations, we get the following results:

For the error derivative with respect to consequent centers:

$$\frac{\partial e(t)}{\partial v^c_{jk}(t)} = -(d_k - y_k)\frac{z_j v^\sigma_{jk}}{\sum_{j=1}^{q(t)} z_j v^\sigma_{jk}} \qquad (7)$$

where $z_j = \prod_{i=1}^{n} E_{ij}$ and E_{ij} is given by (15),

and the error derivative with respect to the consequent spreads:

(1)

$$\frac{\partial e(t)}{\partial v_{jk}^{\sigma}(t)} = -(d_k - y_k)\frac{z_j v_{jk}^{c}\sum_{j=1}^{q(t)} z_j v_{jk}^{\sigma} - z_j \sum_{j=1}^{q(t)} z_j v_{jk}^{c} v_{jk}^{\sigma}}{\left(\sum_{j=1}^{q(t)} z_j v_{jk}^{\sigma}\right)^2} \quad (8)$$

The error derivatives with respect to antecedent centers and spreads involve subsethood derivatives in the chain and are somewhat more complex to evaluate. Specifically, the error derivative chains with respect to antecedent centers and spreads are, respectively:

$$\frac{\partial e}{\partial w_{ij}^{c}} = \sum_{k=1}^{p} -(d_k - y_k)\frac{\partial y_k}{\partial z_j}\frac{\partial z_j}{\partial \mathrm{E}_{ij}}\frac{\partial \mathrm{E}_{ij}}{\partial w_{ij}^{c}} \quad (9)$$

$$\frac{\partial e}{\partial w_{ij}^{\sigma}} = \sum_{k=1}^{p} -(d_k - y_k)\frac{\partial y_k}{\partial z_j}\frac{\partial z_j}{\partial \mathrm{E}_{ij}}\frac{\partial \mathrm{E}_{ij}}{\partial w_{ij}^{\sigma}} \quad (10)$$

With respect to input feature spreads, the error derivative chains are:

$$\frac{\partial e(t)}{\partial x_i^{\sigma}(t)} = \sum_{j=1}^{q(t)}\sum_{k=1}^{p} -(d_k - y_k)\frac{\partial y_k}{\partial z_j}\frac{\partial z_j}{\partial \mathrm{E}_{ij}}\frac{\partial \mathrm{E}_{ij}}{\partial x_i^{\sigma}} \quad (11)$$

where

$$\frac{\partial y_k(t)}{\partial z_j(t)} = \frac{v_{jk}^{\sigma}(v_{jk}^{c} - y_k)}{\sum_{j=1}^{q(t)} z_j v_{jk}^{\sigma}} \quad (12)$$

and

$$\frac{\partial z_j}{\partial E_{ij}} = \frac{\prod_{i=1}^{n} E_{ij}}{E_{ij}} \quad (13)$$

$$y_k(t) = \frac{\prod_{j=1}^{q(t)} z_j v_{jk}^{c} v_{jk}^{s}}{\prod_{j=1}^{q(t)} z_j v_{jk}^{s}} \quad (14)$$

The expressions for antecedent connection mutual subsethood partial derivatives with respect to antecedent centers and spreads, as also to input feature spreads, are obtained by differentiating the expression of the mutual subsethood definition:

$$E_{ij} = E(s_i, w_{ij}) = \frac{C(s_i \cap w_{ij})}{C(s_i) + C(w_{ij}) - C(s_i \cap w_{ij})} \quad (15)$$

where s_i is the fuzzy input signal $S(x_i) = (x_i^{c}, x_i^{\sigma})$, and $C(A) = \int_{-\infty}^{+\infty} a(x)dx$ is the cardinality of a fuzzy set A described by the membership function *a(x)* and thus $C(A \cap B)$ is the intersection surface between A and B membership functions .

2) Online learning rate

Selection of a value for the learning rate, η, has a significant effect on the network performance since it is related to the rate of convergence. It was discovered that the appropriate manipulation of η during the training process can lead to very good results and, hence a large number of different methods for its adaptation have been proposed in the literature [12],[13].

In ARANFIS learning rate is computed based on the assumption that the training data set can be divided into subsets with similar patterns. Since the proposed network has clusters at the hidden layer, the learning rate can be set as:

$$\eta(t) = \frac{\beta}{\sqrt{\mathrm{N}_1^2(t) + \mathrm{N}_2^2(t) + ... + \mathrm{N}_{q(t)}^2(t) + 1}} \quad (16)$$

where β is an empirically selected constant and $N_j(t)$ is the number of input patterns for which the *j*-th hidden node was the winning node, up to iteration *t*.

3) Soft competitive learning parameter

In sequential learning, updating the weights of the antecedent and consequent connections for all rules may lead to inefficient weight updating for the low activated rules.

The parameter a_j in (2)-(7) indicates the similarity between the *j*-th hidden node and the input pattern $\underline{x}(t)$. Let us define $\underline{\mu}_j$ as the vector of the centers of fuzzy weights of the antecedent part of *j*-th rule denoted as $\underline{\mu}_j = [w_{1j}^{c} w_{2j}^{c} ... w_{nj}^{c}]^T$

Due to the smaller steps taken for the adaptation process, soft competitive learning (winner-take-most) will lead quickly to a nearby optimum, while hard competitive learning (winner-take-all) will have more possibilities to get stuck into well-separated local optima [14]. So in addition to the winner, the proposed network also updates the antecedent and consequent connections of some other hidden nodes depending on their similarity with $\underline{x}(t)$:

$$a_j = 1 - \frac{\left\|\underline{x}(t) - \underline{\mu}_j\right\| - \left\|\underline{x}(t) - \underline{\mu}_{nearest}\right\|}{\left\|\underline{x}(t) - \underline{\mu}_{farthest}\right\| - \left\|\underline{x}(t) - \underline{\mu}_{nearest}\right\|} \quad (17)$$

where $\underline{\mu}_{farthest}$ and $\underline{\mu}_{nearest}$ are the farthest and nearest hidden nodes from $\underline{x}(t)$ respectively.

4) Creating a hidden node

Training data are supplied to ARANFIS in the form of pairs ($\underline{x}(t)$, $\underline{d}(t)$) of input and target vectors. If a new input $\underline{x}(t)$ does not significantly activate any rules and the prediction error is significantly large, a new rule is created, having weight centers of the antecedent part the crisp values of $\underline{x}(t)$, by allocating a new hidden node and the number of rules is increased. The weight spreads of the antecedent part are set proportionally to the distance from the existing nearest hidden node to the new node. In this way new inputs are more likely to match the newly created hidden node.

In particular, a new hidden node is set according to the following equations:

$q(t) = q(t-1) + 1$, increase the number of rules

$N_{q(t)} = 1$, set the number of clustered by the new rule input patterns to one

$$\underline{\mu}_{q(t)} = [w^c_{1q(t)} w^c_{2q(t)} \dots w^c_{nq(t)}]^T = [x_1(t) x_2(t) \dots x_n(t)]^T$$

$$w^s_{iq(t)} = k\sqrt{(x_i(t) - \mu_{i,nearest})^2}, \quad i = 1,\dots,n$$

where k is a constant (overlap factor) and $\mu_{i,nearest}$ is the center of the antecedent connection weight, from the i-th input to existing nearest hidden node from $\underline{x}(t)$.

$$v^c_{q(t)k} = d_k(t) - y_k(t), \quad k = 1,\dots,p$$

(set the weight centers of the consequent connections of the new rule to the difference between the desired and the defuzzified output).

The weight spreads $v^s_{q(t)k}, k = 1,\dots,p$ of the consequent part are set randomly to values lying in the range $[\min_{j,k}\{v^s_{jk}\}, \max_{j,k}\{v^s_{jk}\}]$.

IV. Experimental Results

ARANFIS is applicable to a variety of domains ranging from inference systems to knowledge extraction schemes and from classification problems to time series prediction. Four experiments were performed in order to illustrate its efficiency in the above cases. The first one deals with knowledge extraction from numerical data, the second is a time series prediction problem based on publicly available data (Mackey-Glass), while the third is a pure classification problem based on the iris data. The fourth one involves a real dataset of a power plant in order to demonstrate ARANFIS applicability in the field of power systems.

A. Rule Extraction from Numerical Data

In order to validate the rule extraction capability of ARANFIS we created a 2-D synthetic dataset consisting of four classes as shown in Fig.2. The basic aims of the experiment were:

(a) To identify the input partitions of the data

(b) To explore ARANFIS ability to infer *if-then* rules given the input partitions

(c) To check the classification accuracy of ARANFIS w.r.t to classic classification methods like LVQ, and *k*-NN.

Starting from the third aim, which actually wasn't the most important for the particular experiment, we observe from Table I that the classification accuracy of ARANFIS is significantly higher than the other methods while at the same time it creates less clusters (with the exception of the LVQ method).

ARANFIS creates ten partitions in for the x_1 dimension (horizontal axis in Fig.2) and eleven partitions for the x_2 dimension (vertical axis in Fig.2). It is clear that such kind of partition is not typical for fuzzy sets where the linguistic terms that are used rarely exceed five (for example *very low*, *low*, *medium*, *high*, *very high*).

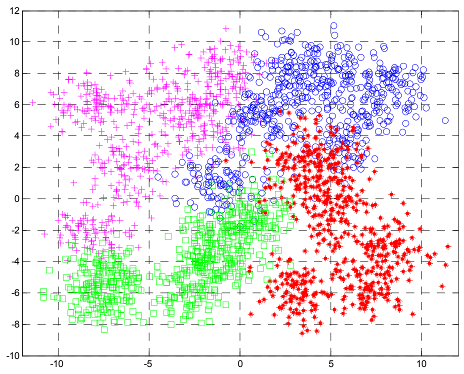

Fig. 2. Synthetically generated 2-Dimensional data for rule extraction validation.

In order to reduce the partitions that are created we should make the conditions that govern hidden rule creation more rigorous. The results shown in Table I were obtained by using the following conditions for creating a new hidden node:

$$|d_k(t) - y_k(t)| > \varepsilon = 0.5 \text{ ,(prediction error)} \qquad (18)$$

and

$$\max_j\{z_j\} < \delta = 0.5 \text{, (rule activation)} \qquad (19)$$

TABLE I
COMPARISON OF ARANFIS WITH OTHER CLASSIFICATION METHODS FOR THE 2-D SYNTHETICALLY GENERATED DATA

Method	Created Clusters	Accuracy (%)
LVQ	24	83.3
1-NN	36	85.7
k-NN, *k*=5	36	88.2
ARANFIS	29	90.4

Setting more rigorous conditions implies higher ε and lower δ. In Table II it is shown that the reduction of the number of hidden nodes does not significantly depreciate the classification performance of the network. In the case of 16 hidden nodes five partitions per dimension were created (it should be noted here that the number of clusters is less than the product of the number of partitions per dimension since the created rules do no cover all combinations; for example the case of x_1=*low* and x_2=*high* may not be encountered).

TABLE II
TESTING ACCURACY AS A FUNCTION OF CREATED CLUSTERS

Methods	Created Clusters	Accuracy (%)
ARANFIS	16	87.9
ARANFIS	29	90.4

In order to explore ARANFIS ability of creating *if-then* rules we examine the case in which the partitions (per dimension) of the input pattern are a priori known. In the data shown in Fig.

2 we consider five partitions per dimension corresponding to linguistic terms *very low (VL)*, *low (L)*, *medium (M)*, *high (H)*, *very high (VH)*. The 15 rules that were created are shown in Table III. Rule based classification through the fuzzy rules shown in Table III (using Gaussian fuzzy sets as membership functions) reach a performance rate of 83.5% (similar to that of LVQ).

TABLE III
FUZZY RULES GENERATED WITH FIVE PARTITIONS PER DIMENSION

Fuzzy Rule Number	IF		THEN
	x_1	x_2	$y \in Class\#$
1	VL	L	4
2	VL	M	1
3	VL	H	1
4	L	VL	4
5	L	H	1
6	L	VH	1
7	M	L	4
8	M	M	2
9	H	VL	3
10	H	L	3
11	H	M	3
12	H	VH	2
13	VH	L	3
14	VH	H	2
15	VH	VH	2

B. Time Series Prediction

ARANFIS was tested in time prediction by considering a benchmark chaotic time series first investigated by Mackey and Glass [15]. The results when using several numbers of rules are shown in Table IV for both the training and testing sets. It should be noted that the resource allocation procedure was not activated during this experiment, due to the very small prediction error, thus, the number of rules remain the same by which ARANFIS was initialized.

TABLE IV
RMSE OF ARANFIS FOR MACKEY-GLASS TIME SERIES PREDICTION AFTER 50 EPOCHS FOR DIFFERENT NUMBER OF RULES

Number of Fuzzy Rules	RMSE (training set)	RMSE (testing set)
3	0.0121	0.0267
5	0.0098	0.0225
7	0.0087	0.0198
10	0.0085	0.0191

TABLE V
COMPARISON OF ARANFIS WITH OTHER TECHNIQUES FOR TIME SERIES PREDICTION USING NRMSE (NORMALIZED MEAN SQUARE ERROR [16])

Method	NRMSE
GEFREX	0.0061
ANFIS	0.0074
EfuNN	0.056
Auto Regressive Model	0.19
Back-Propagation MLP model	0.02
6th order polynomial	0.04
Linear Predictive Model	0.55
ARANFIS (7 rules)	0.017

Comparison with other methods is given in Table V using the results of [17]. Two of the methods presented in Table V outperform ARANFIS (GEFREX, ANFIS). However, ARANFIS performs very well given that it is not optimized for time series prediction.

C. Classification Performance

For measuring the classification performance of ARANFIS we used the iris data. Iris data involves classification of three subspecies of the Iris flower, namely *Iris sestos*a, *Iris versicolo*r, and *Iris virginica* on the basis of four feature measurements of the Iris flower—sepal length, sepal width, petal length, and petal width [18]. The input pattern set comprises of 150 four-dimensional patterns.
The obtained results (resubstitution accuracy) as well as comparisons with other methods (with the help of [19]) are given in Table VI. We observe that ARANFIS outperforms all listed soft computing techniques, with the exception of FuGeNeSys, creating as few as three rules.

TABLE VI
COMPARISON OF ARANFIS WITH OTHER TECHNIUES FOR IRIS DATA CLASSIFICATION

Method	Rules	Resubstitution Accuracy (%)
FuGeNeSys	5	100
NEFCLASS	7	96.7
ReFuNN	9	95.3
EFuNN	17	95.3
FuNe-I	7	96.0
ARANFIS	3	98.0

It should be noted that typically ARANFIS creates three rules for the *iris* classification problem. In order to create more rules the values of ε and δ in (18)-(19) were increased and reduced respectively, deviating from 0.5, which is their default value. In Table VII, though, it is shown that the classification accuracy does not improve as new rules are created.

TABLE VII
RESUBSTITUTION ERROR FOR ARANFIS AS A FUNCTION OF RULES (IRIS DATA)

Rules	3	5	6	7	9	10
Errors	3	3	3	2	2	2

D. Power System Regression Task

This experiment utilizes real data from a power plant (Pont-sur-Sambre (France)) of 120 MW, which can be obtained from the DaISy database [20]. In this dataset there are 5 inputs, defined as the gas flow, the turbines valves opening, the super heater spray flow, the gas dampers and the air flow, while the two outputs, in our case, are the steam pressure and the main stem temperature. Using 200 samples with a total sampling time of 1228.8 sec, we had the following results after 150 epochs, keeping ε and δ at 0.5. The two figures below (Fig. 3 and 4) show how well the system performs in order to approximate the given function.

TABLE VIII
RESUBSTITUTION ROOTMEANSQUARE ERROR FOR ARANFIS IN A REGRESSION TASK (POWER PLANT DATA)

Rules	7	8	10
RMSE	0.1061	0.1003	0.0989

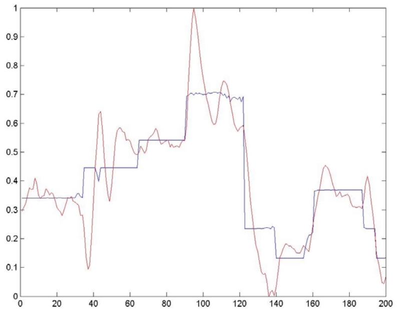

Fig. 3. The steam pressure output. (the real output is colored red, while the approximation function is the blue one).

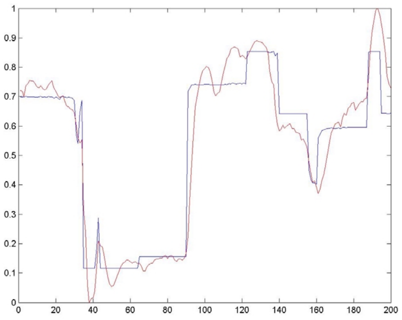

Fig. 4. The main stem temperature output. (the real output is colored red, while the approximation function is the blue one).

V. CONCLUSIONS

In this paper we propose an innovative neural fuzzy architecture, which combines resource allocating procedures and fuzzy sets in order to incorporate (a) inferencing capabilities, (b) expert knowledge modeling, (c) knowledge extraction from numerical data. ARANFIS employs tunable feature fuzzifiers that convert numeric inputs to Gaussian fuzzy sets; mutual subsethood based activation spread; and a fuzzy inner product conjunction operator. ARANFIS embeds rule-based knowledge directly into its architecture while its resource allocating structure enables new rules to be created. The latter is very important for two reasons: (a) there are several domains in which no estimation about the number of rules that are required to solve a particular problem is available, (b) rules can be created to model a changing of a context. Thus, the dynamically formed architecture of ARANFIS is capable of providing the means to model non-stationary phenomena.

ARANFIS knowledge extraction ability was tested on a variety of applications. A synthetically generated dataset was used to show that the performance of knowledge extraction via *if-then* rules is promising, especially in the case where the partition of the input space (per dimension) is a priori known. In contrary, in cases where the partition of the input space is unknown ARANFIS tends to over-partition the input dimensions. The authors are working towards improving this limitation.

The classification performance of ARANFIS turns out to be excellent. By allocating as few as three rules outperforms the majority of the soft-computing schemes that were tested on the iris classification problem. On the other hand, Mackey-Glass time-series prediction is not the favorite field for ARANFIS since it does not require dynamic resource allocation. However, in a real data environment of a power plant system ARANFIS achieves satisfactory performance, in addition to low normalized root mean squared error.

VI. REFERENCES

[1] S. Mitra, and Y. Hayashi, "Neuro-fuzzy rule generation: Survey in soft computing framework," *IEEE Trans. Neural Networks*, vol. 11, pp. 748-768, May 2000.

[2] C. T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput*, vol. 40, pp. 1320–1336, Dec. 1991.

[3] H. Ishibuchi, "Neural network that learn from fuzzy if-then rules," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 85-97, May 1993.

[4] L. M. Fu, "Learning capacity and sample complexity on expert networks," *IEEE Trans. Neural Networks*, vol. 7, pp. 1517-1520, 1996.

[5] D. Nauck and R. Kruse, "A neuro-fuzzy method to learn fuzzy classification rules from data," *Fuzzy Sets and Systems*, vol.89, pp. 277-288, 1997.

[6] S. Mitra, R. K. De, and S. K. Pal, "Knowledge-based fuzzy MLP for classification and rule generation," *IEEE Trans. Neural Networks*, vol. 8, pp. 1338-1350, Nov. 1997.

[7] Y. Lin, G. A. Cunningham, III, and S. V. Coggeshall, "Using fuzzy partition to create fuzzy systems from input–output data and set the initial weights in a fuzzy neural network," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 614–621, Nov. 1997.

[8] Y. Jin, "Fuzzy modeling of high dimensional systems: Complexity reduction and interpretability improvement," *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 212-221, Apr. 2000.

[9] V. Vapnik, *Statistical Learning Theory*, John Wiley and Sons, 1998.

[10] J. Platt, "A resource-allocating network for function interpolation," *Neural Computing*, vol. 3, no. 2, pp. 213-225, 1991.

[11] S. Paul, S. Kumar, "Subsethood-Product Fuzzy Neural Inference System (SuPFuNIS)" *IEEE Trans. On Neural Networks*, vol. 13, No. 3, pp. 578-599, 2002

[12] M. Moreira and E. Fiesler, "Neural networks with adaptive learning rate and momentum terms," IDIAP, Martigny, Switzerland, Tech. Rep. 95-04, 1995.

[13] M. V. Solodov and B. F. Svaiter. "A comparison of rates of convergence of two inexact proximal point algorithms," in *Nonlinear optimisation and related topics*, G. D. Pillo and F. Giannesi,Ed. *Applied Optimization 36*, , Kluwer Academic Publishers, 2000, pp. 415-427.

[14] P. Scheunders, "A comparison of clustering algorithms applied to color image quantization," *Pattern Recognition Letters*, vol. 18, pp. 1379-1384, 1997.

[15] M. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, pp. 287–289, 1977.

[16] J. Kim and N. Kasabov, "HyFIS: Adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems," *Neural Networks*, vol. 12, no. 9, pp. 1301–1321, 1999.

[17] D. Kim and C. Kim, "Forecasting time series with genetic fuzzy predictor ensemble," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 523–535, Nov. 1997.

[18] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, no. 2, pp. 179–188, 1936. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[19] L. I. Kuncheva and J. C. Bezdek, "Nearest prototype classification: Clustering, genetic algorithms, or random search?" *IEEE Trans. Syst., Man Cybern.* C, vol. 28, pp. 160–164, Feb. 1998.

[20] B.L.R. De Moor (ed.), DaISy: Database for the Identification of Systems, ESAT/SISTA, K.U. Leuven, Belgium [Online]. Available: http://www.esat.kuleuven.ac.be/sista/daisy/