MultiGrid-Based Fuzzy Systems for Time Series Prediction: CATS Competition

L. J. Herrera, H. Pomares, I. Rojas, J. Gonzâlez, M. Awad, A. Herrera
Department of Computer Architecture
and Computer Technology
University of Granada
Granada, Spain
E-mail: jherrera@atc.ugr.es

Abstract—**In this paper, the MultiGrid-Based Fuzzy System (MGFS) approach is applied for the CATS Time Series Prediction Benchmark. The MGFS architecture overcomes the problem inherent to all grid-based fuzzy systems when dealing with high dimensional input data, thus keeping low computational cost and high performance. A greedy algorithm for MGFS structure identification allows to perform the input variable selection for the time series prediction problem, while identifying the pseudo-optimal architecture according to the provided dataset.**

I. Introduction

In this paper we propose the MultiGrid-Based Fuzzy System (MGFS) model, for the treatment of high-dimensional problems for function approximation [2], [3] to solve the time series prediction problem CATS - benchmark for *Competition* on Artificial Time Series-.

MultiGrid-Based Fuzzy Systems open a door for the treatment of high dimensional data using Grid-Based Fuzzy Systems (GBFS) for function approximation problems. The algorithm provided for MGFS architecture selection allows to take into account any number of input variables for the prediction, *selecting the most relevant interrelations among the input* variables to perform the final prediction.

The CATS Artificial Time Series consists on 5,000 data points in which 100 of these values are missing. These missing values are divided in 5 blocks

1) - elements 981 to 1,000;
2) - elements 1,981 to 2,000;
3) - elements 2,981 to 3,000;
4) - elements 3,981 to 4,000;
5) - elements 4,981 to 5,000;

These 100 missing values have to be predicted, computing the Mean Square Error E on the 100 missing values using

$$E = \frac{\sum\limits_{m=0}^{4} \sum\limits_{t=1000*m+981}^{1000*(m+1)} (y_t - \hat{y}_t)^2}{100} \tag{1}$$

The authors are requested to provide their prediction for the 100 missing values. The submitted methods will be ranked using this Mean Square Error E. A complete plot of the original time series is shown in *Fig.* 1.

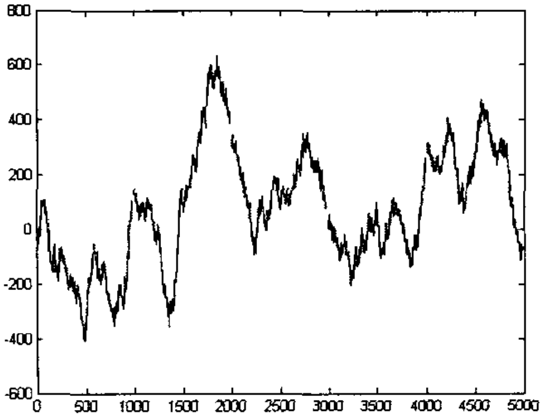

Fig. 1. Original CATS Time Series

II. Data Analysis

For this CATS time series prediction problem, a single series of data points has been provided. No additional information of the underlying data model has been given.

Taking a first look to the original time series we can see that, in principle, there isn't any repetition pattern. The data follows a certain noisy path, but with no identifiable pattern (see *Fig.* 1), in opposite to other well-known time series benchmarks [1], [4], [5].

In general, a predictive model that takes into account previous inputs as input variables has the form

$$\hat{y}(t+h) = F\left(y(t-i_1), y(t-i_2), \ldots, y(t-i_m)\right) \tag{2}$$

where h is the prediction horizon and typically the series i_1, $i_2, \ldots, i_m$ have the form 0, τ, $2*\tau$, ..., $(m-1)*\tau$, where τ is the time delay, according to the Takens' embedding theorem [9] for chaotic time series.

Thus, note that any predictive model that takes into account previous data inputs as input variables as in (2), would have an insufficient amount of training data comparing to the huge resulting input space. There won't be enough similarities among the training data to properly train a predictive model (see *Fig.* 1).

0-7803-8359-1/04/$20.00 ©2004 IEEE

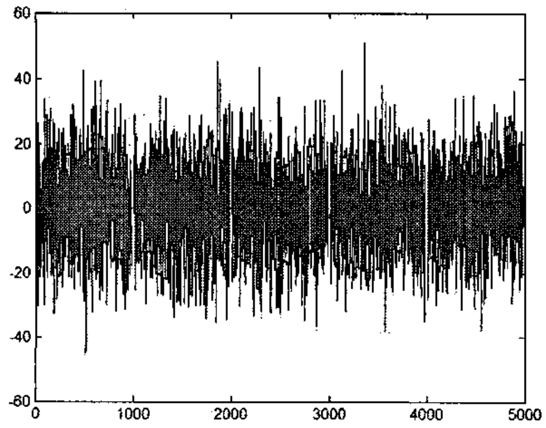

Fig. 2. Differentiated Time Series

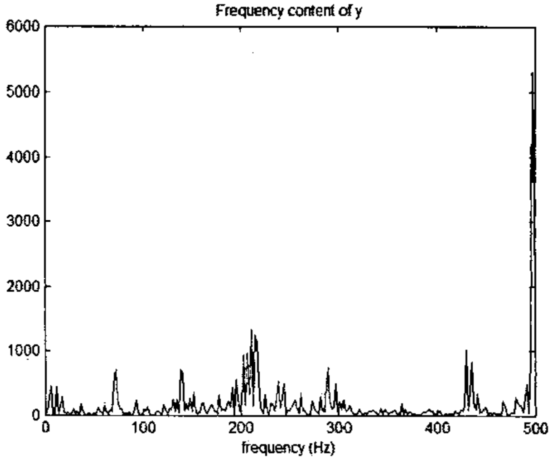

Fig. 4. FFT of the original Time Series

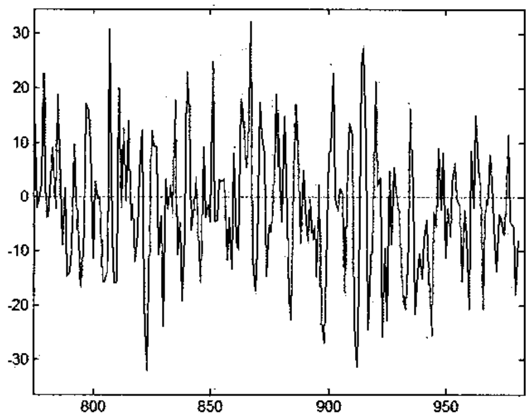

Fig. 3. Partial view of the Differentiated Time Series

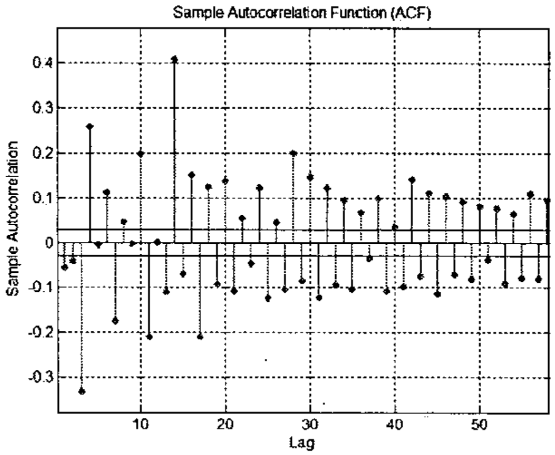

Fig. 5. Sample Auto-Correlation function of the Differentiated Time Series

A. *Differentiated Time Series*

As noted in [5], instead of considering the original time series, we can work with the 'differentiated' time series. New information is taken into account, considering the differences from one data point with respect to the previous one. The whole 'differentiated' time series is shown in *Fig.* 2, and a partial detail is shown in *Fig.* 3.

Also this last partial detail of the differentiated series, shows that there isn't recognizable pattern in the series, unlike several well-known benchmark time series where a specific behavior is present in the data. But for the differentiated time series, the model (see *Fig.* 1) might be used, since all the data is inside a certain short range.

B. *FFT of the CATS Time Series*

The fast fourier transform of the original CATS time series (see *Fig.* 4) showns the high level of "noise" present in the CATS time series, i.e. there is a very important component of high frequency in the Time Series. There is not a strong influence of any other frequency component in the FFT of the series.

C. *Sample auto-correlation function of the Differentiated Time Series*

Computing the sample auto-correlation function of the differentiated time series (see *Fig.* 5), it can be noted the positive auto-correlation of the even lags while for the odd lags, the sample auto-correlation is always negative.

This leads to the fact that the differentiated time series could be separated in two different time series with different behaviors. Thus two different 'recomposed' time series with different trends could be reconstructed. See *Fig.* 6 to check for the first 980 data points, the original Time Series and the two artificially reconstructed time series for even and odd sequences of differences.

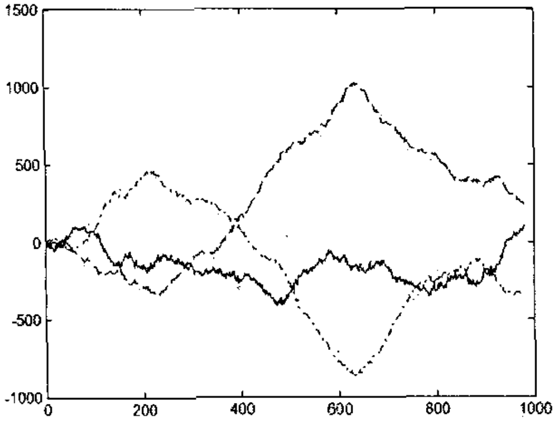

Fig. 6. "Even" and "Odd" series compared to the original CATS Time Series (solid line) for the first 980 points

III. MultiGrid Fuzzy Systems

A wide range of paradigms have been applied to the well-known problem of Time Series Prediction. In particular Fuzzy Systems have been successfully applied in several cases [10], [8]. In general, Fuzzy Systems present a number of advantages comparing to other paradigms, such as: (a) the models obtained with this paradigm are understandable, avoiding the black-box curse of other paradigms; (b) the underlying model is simple, but can explain complex nonlinear relations among variables; (c) it can be expressed in terms of linguistic rules, thus providing full interpretability to the model; (d) it doesn't need an accurate mathematical model and (e) it has high generalization capability [2].

In this work we propose to use a new modified additive fuzzy-based approach based on Grid-Based Fuzzy Systems [6], [7], which we call a MultiGrid-Based Fuzzy Systems model [2], that keeps its philosophy and advantages while avoiding its main drawbacks, the curse of dimensionality. MGFS are able to detect the most relevant input variables in a function approximation problem while detecting unneeded interrelations among them.

Considering in general grid-based fuzzy systems for function approximation problems, when dealing with a high number of input variables, an n-dimensional grid might seem useless for our aim of obtaining a useful approximation of the given data points, since having too many rules as well as too many antecedents on each rule, results in an incomprehensible huge model. Besides, the management of so many parameters may reach an efficiency bottleneck, resulting in a problem practically impossible to optimize.

Fig. 7 shows the proposed MGFS architecture [2] to deal with high dimensional input spaces.

Each group of variables is used to define a Grid-Based Fuzzy System (GBFS) from which a set of rules is obtained in the form [7]

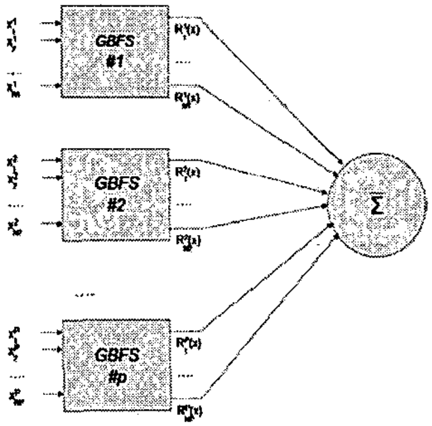

Fig. 7. MultiGrid-Based Fuzzy System (MGFS)

$$\begin{array}{l}\text{IF } x_1 \text{ is } X_1^{i_1} \text{ AND } \ldots \text{ AND } x_N \text{ is } X_N^{i_N} \\ \text{THEN } R_i^p = R_{i_1 i_2 \ldots i_N}\end{array} \tag{3}$$

being R_i^p the $i-th$ rule of the $p-th$ GBFS. Thus, all the rules from all the GBFS form the whole MGFS, whose output is obtained by weighted average defuzzification. Therefore the final output of the system for any input value $\vec{x} = (x_1, x_2, , x_N)$, can be expressed as

$$F(\vec{x}, MF, R, C) = \frac{\sum\limits_{p=1}^{P} \sum\limits_{j=1}^{R_p} R_j^p \prod\limits_{m=1}^{N_p} \mu_m^{j_p}(x_m)}{\sum\limits_{p=1}^{P} \sum\limits_{j=1}^{R_p} \prod\limits_{m=1}^{N_p} \mu_m^{j_p}(x_m)} \tag{4}$$

where explicit statement is made on the dependency of the output function with the structure of membership functions (MF) of the system, with the consequents of the whole set of rules R, and with the hard structure of the system $C = \left\{\left\{x_1^1, x_2^1, x_{N1}^1\right\}, \left\{x_1^2, x_2^2, ...x_{N2}^2\right\}, \ldots, \left\{x_1^P, x_2^P, x_{Np}^P\right\}\right\}$, i.e, the input variables entering each individual GBFS.

IV. MultiGrid Hard-Structure Identification

We present a very effective and automatic algorithm to determine the groups of variables that will form each sub-grid. All the sub-grids together will comprise the system hard-structure, as shown in *Fig.* 7.

A Top-Down algorithm was presented in [2] that, from a complete GBFS (a MGFS with one single sub-grid which has all the input variables) discards more complex structures in favor of simpler ones while keeping a certain error limit and keeping the number of MFs per variable. This approach, though very powerful to discover intrinsic relations in the input variables, has several disadvantages that makes it unsuitable for time series forecasting. That is to say, the starting point is a whole GBFS system, which can be computationally too expensive when having a high number of input variables, thus this starting point is unfeasible for middle-complexity problems.

A different bottom-up approach, first taken in [3], starts with the simplest configuration possible, having one sub-grid per input variable, and performing a search in groups-complexity increase. A different idea surrounds this second alternative: instead of looking for keeping a certain error tolerance while reducing the whole-system complexity, we will keep a certain computational complexity limit (number of parameters to be optimized = number of rules) while searching for the system that performs best, given this number of rules.

The limit in the number of rules for which we will search the best MGFS structure, can be selected taking into account the number of training points that we have (considering that usually it is not worth having a higher number of parameters than data), the number of input variables and the computational cost of the algorithm used.

Given a limit in the number of parameters, in order to equitably compare different MGFS structures throughout the proposed algorithm, the rules must be equally distributed among all the sub-grids forming the MGFS and their variables inside. The heuristic technique that we will use is to homogeneously split the rules among all the sub-grids and, inside each sub-grid, to homogeneously distribute the number of MFs per variable in order to approximate the number of rules assigned. We will use triangular-partitioned MFs [6] so that the optimal rule consequents can then be obtained by linear methods [2], [6].

As seen before, the algorithm works in a greedy manner, starting from a MGFS hard-structure having n sub-grids with one input variable each, where n is the initial number of input variables. Then, it will sequentially search for additions of one variable to one sub-grids, but without searching for component sub-grids of order m until all the possible combinations of order $m-1$ have been explored. Always keeping fixed the number of rules. The MGFS hard-structure that provides the lowest error given the training dataset D and the fixed number of rules will be chosen as the candidate.

The MGFS approach, using this convenient greedy approach for structure selection, provides complexity reduction while searching for the variables that influence most the output, while discarding unneeded interrelations among the input variables. It is a fast and powerful tool, suitable for this time series problem, in which no additional information is given and the input space is in principle undefined for the problem of predicting the 100 points.

V. Modeling the CATS Time Series Problem

In the Data Analysis section, we already presented the sample auto-correlation function of the differentiated time series. If we get the sample partial autocorrelation function (partial ACF) of the differentiated time series (see *Fig.* 8), we can see that only the first 20-25 previous data might influence the prediction of each data $\hat{y}(t)$.

Several tests were performed considering even up to 100 previous values to predict the $\hat{y}(t+1)$, but it didn't bring better results. Also as mentioned in the Data Analysis Section, subsection C, some tests where performed considering the two different time series, even and odd, but no better results where obtained. Then the first 25 variables will thus finally be selected to launch the MGFS structure selection algorithm with the initial objective of predicting $\hat{y}(t+1)$,

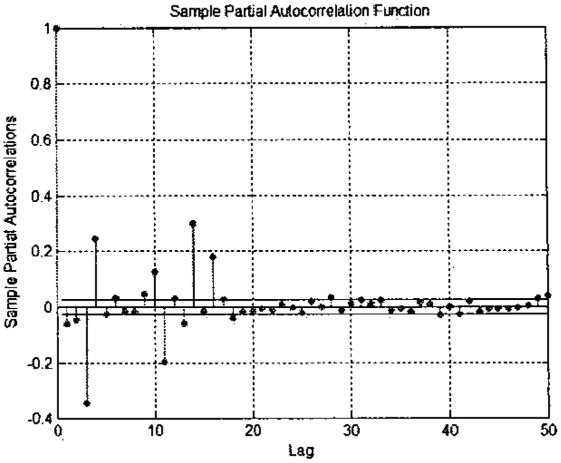

Fig. 8. Sample Partial Auto-Correlation function of the differentiated Time Series

$$\hat{y}(t+1) = F\left(y(t-0), y(t-1), \ldots, y(t-24)\right) \quad (5)$$

The MGFS structure selection algorithm selects the following MGFS configuration: $C = \{\{y(t-2)\}, \{y(t-3)\}, \{y(t-15), y(t-1)\}, \{y(t-13), y(t-12)\}\}$. The test error obtained with this configuration (taking approximately 4000 data points as training and the rest 1000 as test) is 0.815 NRMSE. Thus:

$$\begin{aligned}\hat{y}(t+1) &= F\left(y(t-2)\right) + F\left(y(t-3)\right) + \\ &+ F\left(y(t-15), y(t-1)\right) + \\ &+ F\left(y(t-13), y(t-12)\right)\end{aligned} \quad (6)$$

Note the agreement of the model obtained with the sample auto-correlation function in *Fig.* 8.

The last thing to do is to obtain models to predict the values $\hat{y}(t+2)$, $\hat{y}(t+3)$, …, $\hat{y}(t+20)$. The same pseudo-optimal model obtained could be used to predict the whole series, but the prediction error (that is very high) would be dragged from the predicted value $\hat{y}(981)$ until $\hat{y}(1000)$ -for the first part of the unknown data-. Note that the pseudo-optimal model obtained to predict $\hat{y}(t+1)$, can also be used to predict $\hat{y}(t+1)$, $\hat{y}(t+2)$ and $\hat{y}(t+3)$ since neither $y(t)$ or $y(t-1)$ are input variables for the model.

Several approaches can be taken now to obtain the rest of the models. We could simply predict (always working with the differentiated time series), for the P-th unknown data point

$$\hat{y}(t+P) = F\left(y(t-0), y(t-1), \ldots, y(t-24)\right) \quad (7)$$

TABLE I

PREDICTION ERROR FOR $\hat{y}(t+p)$ FOR $p = 1:20$, AND LAG VALUES IN THE MGFS STRUCTURES

To predict	Trn Err	Tst Err	MGFS Struct. ($\{\{y(t-lag),\dots\}\dots\}$
$\hat{y}(t+1)$	0.797	0.815	$\{\{2\},\{3\},\{15,1\},\{13,12\}\}$
$\hat{y}(t+2)$	0.797	0.815	$\{\{1\},\{2\},\{14,0\},\{12,11\}\}$
$\hat{y}(t+3)$	0.808	0.822	$\{\{0\},\{1\},\{13\},\{11,10\}\}$
$\hat{y}(t+4)$	0.838	0.840	$\{\{0\},\{12\},\{10,9\}\}$
$\hat{y}(t+5)$	0.851	0.863	$\{\{11\},\{9,8\},\{9,1\}\}$
$\hat{y}(t+6)$	0.852	0.863	$\{\{10\},\{8,7\},\{8,0\}\}$
$\hat{y}(t+7)$	0.817	0.879	$\{\{9,10\},\{7,6,2,0\},\{7,6,2,15\}\}$
$\hat{y}(t+8)$	0.839	0.863	$\{\{8,9\},\{6,5,1\}\}$
$\hat{y}(t+9)$	0.839	0.863	$\{\{7,8\},\{5,4,0\}\}$
$\hat{y}(t+10)$	0.840	0.855	$\{\{6,7\},\{4,3,1\}\}$
$\hat{y}(t+11)$	0.830	0.850	$\{\{3,2,0\},\{6,5,1\}\}$
$\hat{y}(t+12)$	0.842	0.857	$\{\{2,1\},\{4,5,0\}\}$
$\hat{y}(t+13)$	0.851	0.862	$\{\{1,0\},\{3,4\}\}$
$\hat{y}(t+14)$	0.866	0.862	$\{\{2\},\{0,3\}\}$
$\hat{y}(t+15)$	0.934	0.940	$\{\{2\},\{13\},\{15\}\}$
$\hat{y}(t+16)$	0.934	0.940	$\{\{1\},\{12\},\{14\}\}$
$\hat{y}(t+17)$	0.934	0.940	$\{\{0\},\{11\},\{13\}\}$
$\hat{y}(t+18)$	0.951	0.944	$\{\{10\},\{12\}\}$
$\hat{y}(t+19)$	0.951	0.944	$\{\{9\},\{11\}\}$
$\hat{y}(t+20)$	0.950	0.945	$\{\{8\},\{10\}\}$

or

$$\sum_{p=1}^{P} \hat{y}(t+p) = F\left(y(t-0), y(t-1), \dots, y(t-24)\right) \quad (8)$$

i.e. the difference among the original time series data point $y(980+P)$ - $y(980)$.

The models obtained for the first possibility by applying the MGFS hard-structure selection algorithm as well as the training and test error obtained are shown in Table I. Note how the error increases as we lose information, i.e. as we try to predict further data points. Note also how the algorithm obtains sometimes similar pseudo-optimal models for close values of P in $\hat{y}(t+P)$.

For the second possibility, i.e. equation 8, the training and test error obtained were a bit worse than for the first alternative, i.e. equation 7.

An example of the approximation of a complete section of twenty data points using the models in Table I is shown in *Fig.* 9 (although the overall estimated MSE is 1000).

For the last 20 data points to be predicted, i.e. from 4980-5000, this model will be used.

Nevertheless, since for the first 80 points to be predicted, we do know the continuation of the Time Series, the same methodology can be taken, but reversing the Time Series to predict the four sections of 20 data points (thus taking into account the following data).

A new set of 20 different models have been obtained using this reversed prediction model.

$$\hat{y}(t-P) = F\left(y(t+0), y(t+1), \dots, y(t+24)\right) \quad (9)$$

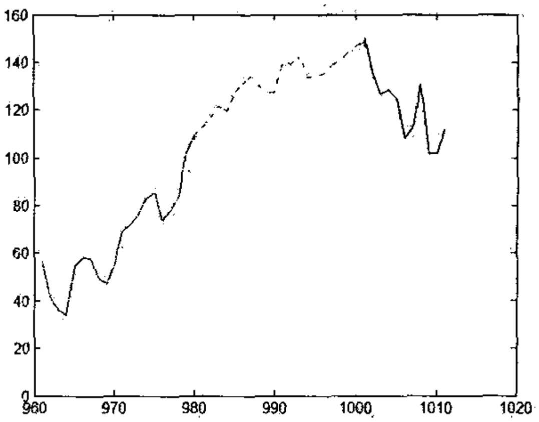

Fig. 9. First approximation using a complete model in Table I

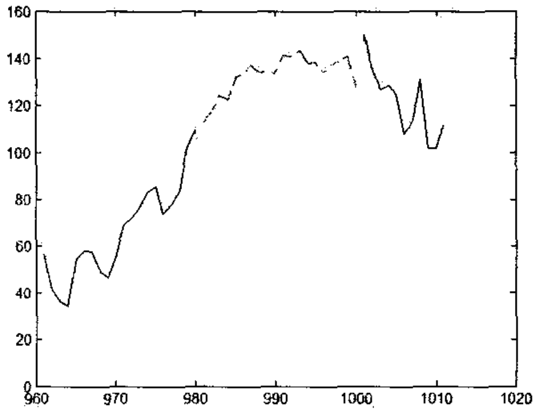

Fig. 10. Final prediction for the first section

Now since we have two complete multi-models to predict the first four sections of 20 unknown data points, a weighted average strategy has been applied to combine the output of the two models. The weights of the two model outputs have been obtaining by considering their error performance, both for $\hat{y}(t+P)$ and $\hat{y}(t-P')$.

The final expected MSE for this combined methodology taking into account transverse and reversed prediction for the unknown data points is 300.

The final predictions for the 100 data are shown in figures *Figs.* 10 to 14.

VI. CONCLUSIONS

In this paper, we have proposed a model to solve the CATS Time Series Benchmark using MultiGrid-Based Fuzzy Systems. This type of additive fuzzy system allows to keep a low computational cost while being able to consider a wide range of input variables. This approach is therefore very

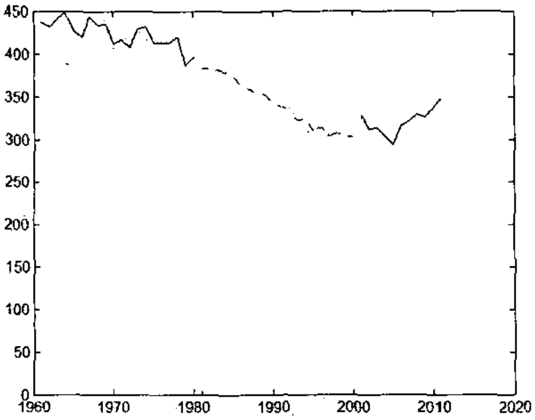

Fig. 11. Final prediction for the second section

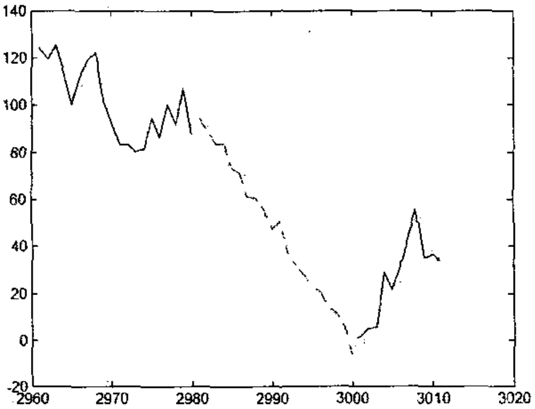

Fig. 12. Final prediction for the third section

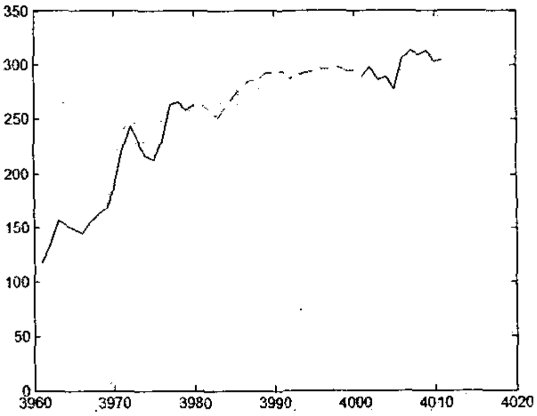

Fig. 13. Final prediction for the fourth section

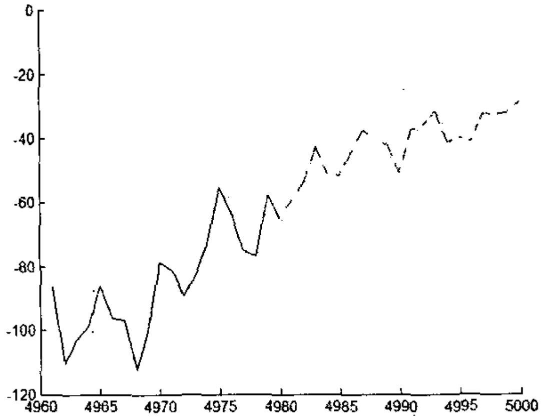

Fig. 14. Final prediction for the fifth section

suitable for Time Series Forecasting problems where the input variables involved are not determined, and an effective solution is needed.

ACKNOWLEDGEMENTS

a This work has been partially supported by the Spanish CICYT Project DPI2001-3219.

REFERENCES

[1] Chang, M.-W., Chen, B.-J., Lin, C.-J.: EUNITE Network Competition: Electricity Load Forecasting , November 2001. Winner of EUNITE world wide competition on electricity load prediction.

[2] Herrera, L.J., Pomares, H., Rojas, I., Valenzuela, O., Awad, M.: MultiGrid-Based Fuzzy Systems for Function Approximation. LNCS/LNAI MICAI'2004, to appear

[3] Herrera, L.J., Pomares, H., Rojas, I., Valenzuela, O., Gonzalez, J., Awad, M.: MultiGrid-Based Fuzzy Systems for Time Series Forecasting: Overcoming the curse of dimensionality. ESANN2004, to appear

[4] Mackey, M.C., Glass,L.: Oscillation and chaos in physcological control systems. Science, 197, July (1977) 287-289

[5] McNames, J., Suykens J. A. K. , Vandewalle, J.: Winning Entry of the K. U. Leuven Time-Series Prediction Competition, International Journal of Bifurcation and Chaos, Vol. 9, No. 8, August 1999

[6] Pomares, H., Rojas, I., Ortega, J., Prieto, A.: A systematic approach to a self-generating fuzzy rule-table for function approximation. IEEE Trans. Syst., Man, Cybern. vol.30. (2000) 431–447

[7] Rojas, I., Pomares, H., Ortega, J., Prieto, A.: Self-Organized Fuzzy System Generation from Training Examples. IEEE Trans. Fuzzy Systems, vol.8, no.1. February (2000) 23–36

[8] Rojas, I., Pomares, H., Bernier, J.L., Ortega, J., Pino, B., Pelayo, F.J., Prieto, A.: Time Series analysis using normalized PG-RBF network with regression weights. NeuroComputing, 42, (2002) 267-285

[9] Takens, F.: Detecting strange attractors in turbulence. In D.A. Rand and L.S. Young (Eds.), Dynamical Systems and Turbulence, Volume 898 of Lecture Notes in MAthematics, pp. 336-381, (1981)

[10] Wang, L.X., Mendel, J.M.: Generating fuzzy rules by learning from examples. IEEE Trans. Syst. Man and Cyber. November/December(1992) 1414–1427