Time Series Forecasting by means of Evolutionary Algorithms

Cristóbal Luque, Jose María Valls Ferrán, Pedro Isasi Viñuela

Universidad Carlos III de Madrid
Departamento de Informática
Av. Universidad 30 - 28911 Spain
{cluque,jvalls,isasi}@inf.uc3m.es

Abstract

Many physical and artificial phenomena can be described by time series. The prediction of such phenomenon could be as complex as interesting. There are many time series forecasting methods, but most of them only look for general rules to predict the whole series. The main problem is that time series usually have local behaviours that don't allow forecasting the time series by general rules. In this paper, a new method for finding local prediction rules is presented. Those local prediction rules can attain a better general prediction accuracy. The method presented in this paper is based on the evolution of a rule system encoded following a Michigan approach. For testing this method, several time series domains have been used: a widely known artificial one, the Mackey-Glass time series, and two real world ones, the Venice Lagon and the sunspot time series.

1 Introduction

A time series consists of an ordered sequence of values of a variable. The goal is to predict future values of the variable, y_i, for $i > D$. In other words, the set $\{y_1, \ldots, y_D\}$ is used to predict $y_{D+\tau}$, where τ is a non negative integer, which receives the name of prediction horizon. In time series related to real phenomena, a good model needs to detect which elements in the data set can generate knowledge refusing those that are noise. In this work a new model has been developed, based on evolutive algorithms to search for rules to detect local behaviours in a time series. That model allows to improve the prediction level in these areas.

Previous works have used linear stochastic models, mainly because they are simple models and their computational burden is low. ARMA (autoregressive moving average) models using data of pressure and level at Venice have been used to forecast the water level at the Venice Lagoon [13]. Following with this domain, in [21] a time series analysis using nonlinear dynamic systems theory and multilayer neural networks models can be found. This strategy is applied to the time sequence of water level data, recorded from Venice Lagoon during the years 1980-1994. In [6],Multilayer Perceptrons are trained using selective learning strategies in order to predict the Venice lagoon time series. In [18], Radial Basis Neural Networks trained with a lazy learning approach, are applied to the same time series and the well-known Mackey-Glass time series. Following the Packard's work to predict dynamical systems [14], [10], [12], and using Evolutionary Algorithms [5] to generate prediction rules on a time series, some advanced Evolutionary Algorithms' techniques to attain better results have been applied to this work.

2 Evolutionary Algorithms for generating prediction rules

For some domains, specially in the case of natural phenomena, the use of techniques of machine learning have to face certain problems. Usually, machine learning techniques, and specially Evolutionary Algorithms, base their learning process on a set of examples. If these examples are fundamentally distributed throughout certain values, the learning process will focus on this range, considering the rest of the values as noise. This fact is positive for some domains, but it becomes a disadvantage in others. For example, in the

1-4244-0910-1/07/$20.00 ©2007 IEEE.

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.

case of stock market prediction or tides prediction, to mention two very different domains, most of the existing measures are over average values. In few occasions, great increases or decreases take place. Nevertheless, those situations are indeed the situations that have more importance from the point of view of the prediction task. Our approach bases on finding rules that represent both, the usual and the atypical behaviours of the time series, in order to be able to predict future values of the series. These rules will be obtained using an Evolutionary Algorithm.

In order to avoid the generalization problem, a Michigan approach has been implemented [2] in the Evolutionary Algorithm, using a Steady-State strategy. In the Michigan's approach, the solution to the problem is the total population instead of the most fitted individual. This way allows the evolution of rules for common behaviours of the problem, but also allows atypical behaviours. Doing it, we pay attention to these unusual behaviours which in other cases would be considered as noise.

Due to the fact that each rule is only evaluated with the examples associated to it, it is only locally or partially applicable. This local characteristic allows the system to use specific rules for particular situations. On the other hand, this method doesn't assure the system to make a prediction for all the time series. A balance between the perfomance of the system and the percentage of prediction must be found.

3 Description of the method

3.1 Encoding

The approach suggested in this paper is based on the generation of rules for making predictions. The first step consists of fixing a value for the constant D, that represents the number of consecutive time instants used to make the prediction. For instance, if $D = 5$ a rule could be a condition like "if the value of the variable at time unit 1 is smaller than 100 and bigger than 50, at time unit 2 is smaller than 90 and bigger than 40, at time unit 3 is smaller than 5 and bigger than -10, at time unit 5 is smaller than 100 and bigger than 1, then the measure at time unit $5+\tau$ will be 33 with an expected error of 5". This rule could be expressed as:

$$\text{IF } (50 < y_1 < 100)\ AND\ (40 < y_2 < 90)$$

$$AND\ (-10 < y_3 < 5) AND\ (1 < y_5 < 100)$$

$$\text{THEN prediction} = 33 \pm 3$$

In figure 1 a graphical representation of a rule is shown. For a rule (R), a conditional part (C_R) and a predicting part (P_R) are defined. The conditional part consists of a set of pairs of intervals $C_R = \{I_1^R, I_2^R, ..., I_i^R, ..., I_D^R\}$; where each I_i^R is an interval $I_i^R = \{LL_i^R, UL_i^R\}$, being LL_i^R the lower limit, and UL_i^R the upper limit for the i-th input data. The predicting part is composed by two values: the prediction value and the expected error, $P_R = \{p_R, e_R\}$.

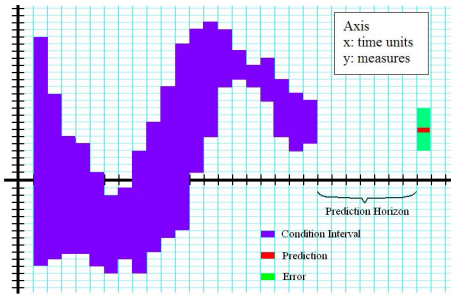

Figure 1. Graphical representation of a rule

The conditional part could include void values ($*$) that means that the value for this interval is irrelevant, that is, we don't care which value it has.

The previous rule is encoded as:

$$(50, 100, 40, 90, -10, 5, *, *, 1, 100, 33, 5)$$

Genetic operators can be applied to generate new rules from an initial population. In order to do that we need to define a crossover process. We select two rules in the population and they produce a new offspring rule. This offspring will inherit his genes from his parents, being each gene an interval I_j. For each $i < D$ the offspring can inherit two genes (one from each parent) with the same probability. This type of crossover is known as uniform crossover. This offspring will not inherit the values for 'prediction' and 'error', as we can see in the following example:

Parent A:

$$(50, 100, 40, 90, -10, 5, *, *, 1, 100, 33, 5)$$

Parent B:

$$(\mathbf{60}, \mathbf{90}, \mathbf{10}, \mathbf{20}, \mathbf{15}, \mathbf{30}, \mathbf{40}, \mathbf{45}, *, *, \mathbf{60}, \mathbf{8})$$

Offspring:

$$(50, 100, \mathbf{10}, \mathbf{20}, -10, 5, \mathbf{40}, \mathbf{45}, *, *, p, e)$$

Once generated, an offspring may suffer mutation of some gene. This mutation process consists of enlargement, shrink or moving up or down the interval encoded by the gene. Let R be an individual, the process to obtain the prediction and the error values for R is the following:

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.

- Calculate the set of points of the time series $S = \{x_1, x_2, ..., x_k, ..., x_m\}$ such that fits the conditional part of the rule R. This subset will be called $C_R(S)$:

$$C_R(S) = \{\vec{X}_i | \vec{X}_i \ fits \ C_R\}$$

where $\vec{X}_i = (x_i, x_{i+1}, ..., x_{i+D-1})$ and the vector $\vec{X}_i$ fits C_R if:

$$LL_1^R \leq x_i \leq UL_1^R, LL_2^R \leq x_{i+1} \leq UL_2^R, ...$$
$$..., LL_D^R \leq x_{i+D-1} \leq UL_D^R$$

- Once calculated $C_R(S)$, the next step consists of determining the output for the prediction horizon τ for each point. This value is: $v_i = x_{i+D-1+\tau}$

- This new value is added as a new component to the vector $\vec{X}_i$, so we get:

$$C'_R(S) = \{\vec{X}'_i | \vec{X}_i \ fits \ C_R\}$$

where:

$$\vec{X}'_i = (x_i, x_{i+1}, ..., x_{i+(D-1)}, v_i) = (\vec{X}_i, v_i)$$

- The prediction p_R for the rule R is calculated by mean of a linear regression with all the vectors in the set $C'_R(S)$. In order to do that it's necessary to calculate the coefficients of the regression $\vec{A} = (a_0, a_1, ..., a_D)$ which define the hyperplane that better approximates the set of points $C'_R(S)$. Let $\tilde{v}_i$ be the estimated value obtained by the regression at the point $\vec{X}_i$, it is defined as:

$$\tilde{v}_i = a_0 x_i + a_1 x_{i+1} + \ldots + a_{D-1} x_{i+D-1} + a_D$$

- Thus, the estimated error value for the rule, e_R, is:

$$e_R = Max_i\{|v_i - \tilde{v}_i| / \vec{X}_i \in C_R(S)\}$$

Therefore, each individual represents a rule able to predict the series partially. The set of all the individuals (all the rules) defines the prediction system. Nevertheless, zones of the series that do not have any associated rule could be found. In this case, the system cannot make a decision in this region. It is desirable, and it is an objective of this work, to make the unpredicted zone as small as possible. Our system, therefore, must look for individuals that, on the training set, predict the maximum number of points with the minimum error. Therefore, the $fitness$ function for an individual R is defined as:

```
IF ((NR>1) AND (eR < EMAX)) THEN
      fitness = (NR*EMAX) - eR
ELSE
   fitness = f_min
```

where NR is the number of points of the training set that fit the condition C_R (i.e. $NR = cardinal(C_R(S))$). EMAX is a parameter of the algorithm that punishes the individuals with a maximum absolute error greater than EMAX. f_min is a minimum value assigned to the individuals whose rule is not fitted at any point in the training set.

The goal of this fitness function is to establish a balance between individuals whose prediction error is the lowest possible, and at the same time, the number of points of the series in which it makes prediction (the number of points of $C_R(S)$) is the highest possible.

3.2 Initialization

The method designed tends to maintain the diversity of the individuals in the population as well as its capacity to predict different zones at the prediction space. But the diversity must exist previously. In order to do that, a specific procedure of population initialization has been devised. The main idea of this procedure is to make an uniform distribution throughout the range of possible output data. For example, in the case of Venice lagoon tide prediction , the output ranges from -50 cm to 150 cm. If the population has 100 individuals, the algorithm creates 100 intervals of 2 cm width, and in this way all the possible values of the output are included. The initialization procedure creates a rule for each interval previously mentioned, so this rule's prediction is included in the interval. The initialization procedure for an interval I has the following steps:

1. Select all the training patterns $\vec{X}_i$ whose output belongs to I.

2. Determine a maximum and a minimum value for each input variable of all the pattern selected in the previous step.

3. Those maximum and minimum values define the values assigned to the rule for each input variable. This rule's prediction is set as the mean of the output value of the patterns selected in the step 1.

This procedure will produce very general rules, so they cover all the prediction space. The Evolutionary Algorithm will improve the rules in order to make them more specific.

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.

3.3 Evolution of rules

Basically, the method described in this paper bases on using a Michigan approach with a steady state strategy. That means that in each generation two individuals are selected proportionally to the fitness function. This selection is made by means of three rounds trials. Those parents produce only one offspring by crossover. Then the algorithm replaces the nearest individual to the offspring in phenotypic distance, i.e. looks for the individual in the population that make predictions on similar zones in the prediction space. The offspring replaces the individual selected in the population, if and only if its fitness is better than the individual selected. Else the population doesn't change. This replacing method is mainly used in crowding methods [3], in which them try to maintain a diverse population in order to find several solutions to the problem. In the case of study of this paper, this approach is widely justified by the fact that we are looking for several solutions to cover the space of prediction as much as possible, so that generated rules could predict the highest number of situations. Moreover, the diversity of the solutions allows the generation of rules for specific highly special situations.

3.4 Prediction

This statistical method obtains different solutions in different executions. After each execution the solutions obtained at the end of the process are added to the obtained in previous executions. The number of executions is determined by the percentage of the search space covered by the rules. The set of all the rules obtained in the different executions is the final solution of the system. Once the solution is obtained, it is used to produce outputs to unknown inputs patterns. This is done following the next steps:

- For each input pattern, we look for the rules that this pattern fits.
- Each rule produce an output for this pattern.
- The final system output (i.e., the prediction of the system) is the mean of the output for each pattern.

4 Experiments

The method have been applied to three different domains: an artificial domain widely use in the bibliography (Mackey-Glass series) and two time series corresponding to a natural phenomenons, the water level in Venice Lagoon and the sunspot time series.

4.1 Venice Lagoon time series

This real world time series represents the behavior of the water level at Venice lagoon. Unusual high tides result from a combination of chaotic climatic elements with the more normal, periodic, tidal systems associated with a particular area. The prediction of high tides has always been the subject of intense interest, not only from a human point of view, but also from an economic one, and the water level of Venice Lagoon is a clear example of these events [13], [11]. The most famous example of flooding in the Venice lagoon occurred in November 1966 when the Venice Lagoon rose by nearly 2 meters above the normal water level. That phenomenon is known as "high water" and many efforts have been made in Italy to develop systems for predicting sea level in Venice, mainly for the prediction of the high water phenomenon [17]. Different approaches have been developed for the purpose of predicting the behavior of sea level at the Venice lagoon [17, 19]. Multilayer feedforward neural networks have also been used to predict the water level [21] obtaining same advantages over linear and traditional models. Standard methods produce very good mean predictions, but they are not so successful for those unusual values. For the experiments explained above , the following values for the variables τ and D have been used: $D = 24$, and $\tau = 1, 4, 12, 24, 28, 48, 72, 96$. That means that the measures of the 24 consecutive hours of the water level have been used to predict the water level, measured in cm, 1,4,12... etc hours later.

The results for such experiments are shown in table 1, towards the results of other work for the same domain [21], using Neural Networks. Those experiments use a training set of 45.000 measures, and a validation set of 10.000. The populations were evolved along 75.000 generations. The rules used as input 24 consecutive hours of the water level. The value of the "Percentage of prediction" is the percentage of points in the validation set such that there is a prediction for it by, at least, a rule. No rule made a prediction for the rest of the validation set. The column 'Error RS' in the table 1 shows the error for the Rule System described in this paper, and 'Error NN' shows the error for Neural Networks obtained in [21]. The error measures used in those experiments is root mean squared error (RMSE). If we define $e = \frac{1}{2}(x - \bar{x})^2$, RMSE can be expressed as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} e^2}{n}}$$

The results show a similar improvement of the prediction in horizons starting on 4 hours, and similar re-

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.

Table 1. Comparative of results for the Venice Lagoon Time series

Horizont	Percentage of prediction	Error RS	Error NN
1	91,3%	3,37	3,30
4	99,1%	8,26	9,55
12	98,0%	8,46	11,38
24	99,3%	8,70	11,64
28	98,8%	11,62	15,74
48	97,8%	11,28	-
72	99,7%	14,45	-
96	99,5%	16,04	-

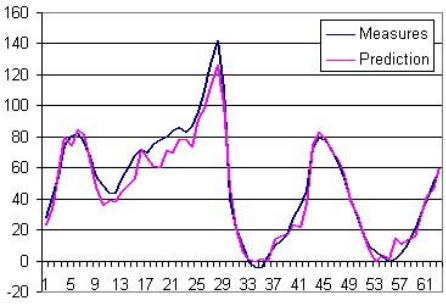

Figure 2. Prediction for an inusual tide with horizon 1

sults for 1 hour horizon. In all the cases the objective was to maximize the percentage of predicted data of validation set avoiding a high mean error. It is interesting to observe that, when the prediction horizon increases, the percentage of prediction does not diminish. Thus, the system seems to be stable to variations of the prediction horizon. This property seems very interesting, because it shows that the rules are adapted to the special and local characteristics of the series. In addition, it can be observed that if the prediction horizon is increased, less rules are necessary to predict a percentage even higher of the series. All this almost doesn't increase the error value.

It is important to observe that the prediction accuracy of the rule system outperforms the accuracy of the neural network system for horizons bigger that 1 hour, being the percentage of prediction data very close to 100%.

A comparison between real tide and prediction for a case of inusual high tide can be seen at figure 2. It can be seen how good the predicted value to the real time series is, even for unusual behaviours.

Table 2. Comparative of results for the Mackey-Glass time series

Pred. Hor.	Perc. pred.	Error	Error MRAN	Error RAN
50	78,9 %	0,025	0,040	-
85	78,2 %	0,046	-	0,050

4.2 Mackey-Glass time series

The Mackey-Glass time series is an artificial series widely used in the domain of the time series forecasting, [15][20], because it has specially interesting characteristics. It's a chaotic series that needs to be defined with great detail. It is defined by the following differential equation:

$$\frac{ds(t)}{dt} = -bs(t) + a\frac{s(t-\lambda)}{1+s(t-\lambda)^{10}}$$

As the papers referred, the values $a = 0.2$, $b = 0.1$ and $\lambda = 17$ were used to generate the time series.

5000 values of the time series are generated using the above equation. The initial 3500 samples are discarded in order to avoid the initialization transients. 1000 data points, corresponding to the sample time between 3500 and 4499, have been chosen for the training set. The test set is composed by the points corresponding to the time interval $[4500, 5000]$. All data points are normalized in the interval $[0, 1]$.

A comparative of the results for the algorithm with the results of [15] for a horizon of 85 (column "Error RAN"), and with the result of [20] for a horizon of 50 (column "Error MRAN"), can be seen in table 2. The error used for the comparison is NMSE (Normalized mean squared error). In both cases, an improvement of the result were attained, that, although they are not so significant, they suggest us that we have a better level of prediction for the difficult regions of the time series.

The percentage of prediction for the test set (near 80%) induces us to think that the discarded elements were certainly inductive of high errors, since its discarding allows better results than the obtained in the bibliography.

4.3 Sunspot Time Series

This time series contains the average number of sunspots per month measured from January of 1749 to March of 1977. These data are available at **http://sidc.oma.be** ('RWC Belgium World Data Center for the Sunspot'). That chaotic time series has local behaviours, noise and even unpredictable zones

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.

Table 3. Comparative of results for the sunspot time series.

Pred. Horiz.	Perc. of pred.	Rule System error	Feedfw NN error	Recurr. NN error
1	100%	0,00228	0,00511	0,00511
4	97,6%	0,00351	0,00965	0,00838
8	95,2%	0,00377	0,01177	0,00781
12	100%	0,00642	0,01587	0,01080
18	99,8%	0,01021	0,02570	0,01464

using the archived knowledge. A comparative of the results of the experiments compared to the results in [7] can be seen at table 3. The error measure used is:

$$e = \frac{1}{2(N+\tau)} \sum_{i=0}^{N} (x(i) - \tilde{x}(i))^2$$

In all cases the experiments were done using the same data set: from January of 1749 to December of 1919 for training, and from January of 1929 to March of 1977 for validation, standardized in the $[0, 1]$ interval; in all the cases, 24 inputs were used. The predictions in [7] were done by multilayer feedforward networks. In all the cases the algorithm explained in this paper improves the results in [7]. A deeper study of the results confirms the ability of this system for recognize, in a local way, the peculiarities of the series, as in the previous domains.

5 Conclusions

This article presents a new method based on prediction rules for time series forecasting, although it can be generalized for any problem that requires a learning process based on examples. One of the problems in the time series field is the generalization ability of the artificial intelligence learning systems. On one hand, general systems produce very good predictions over all the standard behaviours of the time series, but those predictions usually fail over extreme behaviours. For some domains this fact is critical, because those extreme behaviours are the most interesting.

In order to solve this problem a rules based system has been designed, using a Michigan approach, using selection by trials and replacing new individuals by a Steady-State strategy. This method includes a specific initialization procedure, shown at section 3.2, and a process designed to maintain the diversity of the solutions. This method presents the characteristic of not being able to predict the whole time series, but on the other hand, it has a better accuracy, even for unusual behaviours. The algorithm can also be tuned in order to attain a higher prediction percentage at the cost of worse prediction results.

The results show that for special situations, mainly for unusual behaviours (high tides, function peaks, etc.) the system is able to obtain better results than the previous works, although the mean quality of the predictions over the whole series is not significantly better. Therefore, the system can find, if it is possible, good rules for unusual situations, but it cannot find better rules for standard behaviours of the time series than the previous works, where standard behaviours means the behaviours that more often are repeated along the time series.

Another interesting characteristic of the system is its ability to find regions in the series whose behaviour is not able to be generalizable. When the series contains regions with special particularities, the system cannot only localize them, but it can build rules for a best prediction of those ones. The proposed method has been devised to solve time series problem, but it also can be applied to other machine learning domains.

This article has been financed by the Spanish founded research MCyT project TRACER, Ref: TIC2002-04498-C05-04M.

References

[1] T. Bäck, H.P. Schwefel.: Evolutionary Algorithms: Some Very Old Strategies for Optimization and Adaptation. In Perret-Gallix (1992), pp. 247-254.

[2] L.B. Booker, D.E. Goldberg, J.H. Holland: Classifier Systems and Genetic Algorithms. Artificial Intelligence No 40 (1989), pp. 235-282.

[3] K.A. De Jong: Analysis of the Behavior of a Class of Genetic Adaptive Systems. PhD thesis, University of Michigan, August, (1975).

[4] K.A. De Jong, W.M. Spears, F.D. Gordon: Usign Genetic Algorithms for Concept Learning. Machine Learning 13 (1993), pp. 198-228.

[5] D.B. Fogel: An introduction to simulated evolutionary optimization. IEEE transactions on neural networks, vol 5, n 1, jan 1994.

[6] I.M. Galván, P. Isasi, R. Aler, J.M. Valls: A selective learning method to improve the generalization of multilayer feedforward neural networks. International Journal of Neural Systems, Vol 11, No 2 (2001), pp. 167-177.

[7] I.M. Galván, P. Isasi: Multi-step Learning Rule for Recurrent Neural Models: An Application to Time

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.

Series Forecasting. Neural Processing 13 (2001), pp.115-133.

[8] J.H. Holland: Adaptation in Natural and Artificial Systems. University of Michigan Press (1975).

[9] C.Z Janikow: A Knowledge Intensive Genetic Algorithm for Supervised Learning. Machine Learning 13 (1993), pp. 189-228.

[10] T.P. Meyer, N. H. Packard: Local Forecasting of High-Dimensional Chaotic Dynamics, Nonlinear modeling and forecasting. 1990; editors, Martin Casdagli, Stephen Eubank, pp. 249-263.

[11] Michelato, A., Mosetti, R., and Viezzoli, D. (1983). Statistical forecasting of strong surges and aplication to the lagoon of Venice. *Boll. Ocean. Teor. Appl.*, 1:67–83.

[12] M. Mitchell: An introduction to Genetic Algorithms, Cambridge, MA: MIT Press (1996), pp. 55-65.

[13] E. Moretti, A. Tomasin: Un contributo matematico all'elaborazione previsionale dei dati di marea a Venecia. Boll. Ocean. Teor. Appl. 2 (1984), pp. 45-61.

[14] N. H. Packard: A genetic learning algorithm for the analysis of complex data. complex systems 4, no 5 (1990), pp. 543-572.

[15] J. Platt: A Resource-Allocating Network for Function Interpolation. Neural Computation, 3 (1991), pp. 213-225.

[16] S.F. Smith: A Learning System Based on Genetic Adaptative Algorithms. Ph.D. Thesis, University of Pittsburgh (1980).

[17] Tomasin, A. (1973). A computer simulation of the Adriatic Sea for the study of its dynamics and for the forecasting of floods in the town of Venice. *Comp. Phys. Comm.*, 5:51.

[18] Valls, J. M., Galván, I. M., and Isasi, P. (2004). Lazy learning in radial basis neural networks: a way of achieving more accurate models. *Neural Processing Letters*, 20:105–124.

[19] Vittori, G. (1992). On the chaotic features of tide elevation in the lagoon Venice. *Proc. of the ICCE-92, 23rd International Coference on Coastal Engineering*, pages 4–9.

[20] L. Yingwei, N. Sundararajan, P. Saratchandran. A Sequential Learning Scheme for Function Aproximation using Minimal Radial Basis Function Neural Networks. Neural Computation, 9 (1997), pp. 461-478.

[21] J.M. Zaldívar, E. Gutiérrez, I.M. Galván, F. Strozzi, A. Tomasin: Forecasting high waters an Venice Lagoon using chaotic time series analysis and nonlinear neural network. Journal of Hydroinformatics 02.1 (2000), pp. 61-84.

Authorized licensed use limited to: Univ Carlos III. Downloaded on April 2, 2009 at 09:46 from IEEE Xplore. Restrictions apply.