Real-Time Forecasting by Bio-Inspired Models

Paulo Cortez
Departamento de Sistemas de Informação
Universidade do Minho
Guimarães, Portugal
email: pcortez@dsi.uminho.pt

Miguel Rocha
Departamento de Informática
Universidade do Minho
Braga, Portugal
email: mrocha@di.uminho.pt

Fernando Sollari Allegro
Instituto de Ciências Biomédicas
Abel Salazar
Porto, Portugal
email: SollariAllegro.admn@hgsa.min-saude.pt

José Neves
Departamento de Informática
Universidade do Minho
Braga, Portugal
email: jneves@di.uminho.pt

Abstract

In recent years, bio-inspired methods for problem solving, such as *Artificial Neural Networks (ANNs)* or *Genetic and Evolutionary Algorithms (GEAs)*, have gained an increasing acceptance as alternative approaches for forecasting, due to advantages such as nonlinear learning and adaptive search. The present work reports the use of these techniques for *Real-Time Forecasting (RTF)*, where there is a need for an autonomous system capable of fast replies. Comparisons among bio-inspired and conventional approaches (e.g., *Exponential Smoothing*), revealed better forecasting performances for the evolutionary and connectionist models.

Keywords: Artificial Neural Networks, Exponential Smoothing, Genetic and Evolutionary Algorithms, Real-Time Forecasting, Time Series.

1. Introduction

In many industrial systems, the use of reliable forecasts, based solely on previous data, leads to strategic advantages, which may be the key to success in management and control. *Time Series Forecasting (TSF)*, the forecast of a chronological ordered variable, allows the modeling of complex systems, where the aim is to predict the system's behavior and not how the system works. Contributions from the arenas of *Operational Research*, *Statistics*, and *Computer Science* have lead to solid *TSF* methods. Yet, although these methods give accurate forecasts on linear *Time Series (TS)*, they carry an handicap with noisy or nonlinear components, which are common in real world situations (e.g., in financial daily data) [13].

Alternative approaches for *TSF* arise from the *Artificial Intelligence* field, where there has been a trend to look at *Nature* for inspiration. In particular, studies on the *nervous system* and *biological evolution* influenced the loom of powerful tools, such as *Artificial Neural Networks (ANNs)* and *Genetic and Evolutionary Algorithms (GEAs)*, widely used in scientific and engineering problems, such as the ones of *Combinatorial and Numerical Optimization*, *Pattern Recognition* or *Computer Vision*.

ANNs are connectionist models that mimic the central nervous system, being innate candidates for *TSF* due to capabilities such as nonlinear learning, input-output mapping and noise tolerance [9]. On the other hand, *GEAs* are suited for optimization problems, where the exhaustion of all possible solutions requires huge computation [11]. *GEAs* perform a global multi-point search, being able to escape from undesired local minima. Indeed, comparative studies have shown that both bio-inspired techniques can forecast as well or even better than conventional methods [14, 6]. However, these approaches also present the handicap of requiring more computation, due to the difficulty of optimal model parameters' estimation (e.g., *ANN* topology selection and training).

Real-Time Forecasting (RTF) is a special case of forecasting, where short sampling periods are adopted (e.g., in minutes or seconds), being widely used in control and management of critical systems. Therefore, when designing bio-inspired models for *RTF*, one has to deal with two crucial aspects: the forecasting system needs to work autonomously, without human intervention; and the model parameters need to be estimated within the sampling period.

The present work aims at presenting a *RTF* architecture that is capable of using bio-inspired approaches. The paper is organized as follows: firstly, the basic concepts for *RTF* analysis are defined; then, the forecasting models are presented, Next, a description of the different experiments performed is given, being the results analyzed.

2. Time Series Forecasting

A *Time Series (TS)* is a collection of time ordered observations x_t, each one recorded at a specific time t (period). A *TS* model ($\widehat{x}_t$) assumes that past patterns will occur in the future. The *error* of a forecast is given by the difference between actual values and those predicted by the model:

$$e_t = x_t - \widehat{x}_t \quad (1)$$

The overall performance of a model is evaluated by a forecasting accuracy measure, namely the *Root Mean*

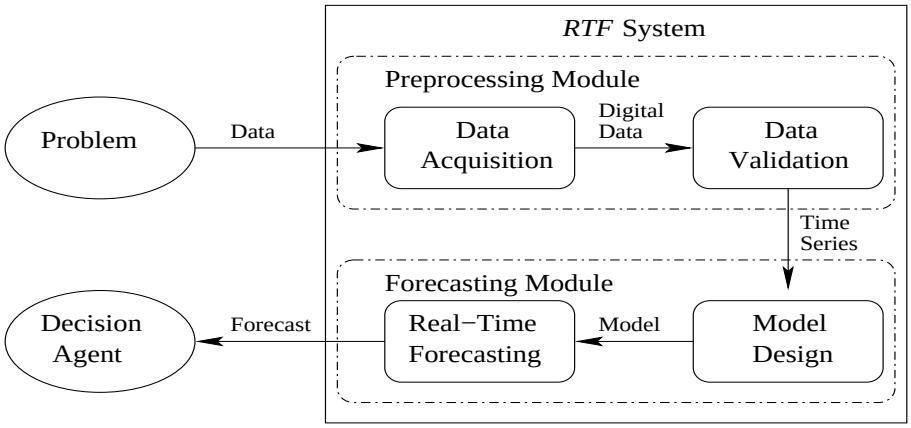

Figure 1. The *RTF* architecture.

Squared (RMSE) and *Normalized Mean Square Error (NMSE)*, given in the form:

$$\begin{aligned} RMSE &= \sqrt{\frac{\sum_{i=t+1}^{t+L} e_i^2}{L}} \\ NMSE &= \frac{\sum_{i=t+1}^{t+L} e_i^2}{\sum_{i=t+1}^{t+L} (x_i - \overline{x})^2} \end{aligned} \tag{2}$$

where L denotes the number of forecasts and $\overline{x}$ the mean of the *TS*.

One popular *TSF* method is *Exponential Smoothing (ES)*, which is based on some underlying patterns (e.g., trend and seasonal ones) that are distinguished from random noise by averaging the historical values. This popularity is due to advantages such as the simplicity of use, the reduced computational demand, and the accuracy on short-term forecasts, specially with seasonal series. The general model, also known as *Holt-Winters*, is defined by a set of equations [8], as stated below:

$$\begin{aligned} F_t &= \alpha \frac{x_t}{S_{t-K}} + (1-\alpha)(F_{t-1} + T_{t-1}) \\ T_t &= \beta(F_t - F_{t-1}) + (1-\beta)T_{t-1} \\ S_t &= \gamma \frac{x_t}{F_t} + (1-\gamma)S_{t-K} \\ \widehat{x}_t &= (F_{t-1} + T_{t-1}) \times S_{t-K} \end{aligned} \tag{3}$$

where F_t, T_t and S_t stand for the smoothing, trend and seasonal estimates, K for the seasonal period, and α, β and γ for the model parameters.

Another important *TSF* model is the *AutoRegressive Moving Average (ARMA)* one, which is based on a linear combination of past values (AR components) and errors (MA components), given in the form [2]:

$$\widehat{x}_t = \mu + \sum_{i=1}^{P} A_i x_{t-i} + \sum_{j=1}^{Q} M_j e_{t-j} \tag{4}$$

where P and Q denote the AR and MA orders, A_i and M_j the AR and MA coefficients, being μ a constant value. Both the constant and the coefficients of the model are usually estimated using statistical approaches (e.g., least squares methods).

3. Real-Time Forecasting

Real-Time Forecasting (RTF) is a special case of forecasting, where forecasts need to be automatically issued, with a fast timing (e.g., every minute or second), being a crucial element in many control systems, such as the ones of urban water pollution [7] or the power load forecasting [4].

Since forecasting techniques operate on data generated by past events, an automated system will need to go through four operations [8]: *data collection*, *data validation*, *model building* and *model extrapolation*. Based in this cycle, a *RTF* architecture is presented (Figure 1), which envelopes two main components: the *preprocessing* and *forecasting* modules.

The first module transforms the raw data given by the problem, through the use of a data acquisition system (e.g., analogic digital converter). Then, the validation block (e.g., data filter) condensates the resulting information into a time series with a regular sampling period. In this work, this module will not be considered, since it is problem dependent (e.g., requiring the use hardware components such as *temperature sensors*), being assumed that the data is correctly sampled and validated.

The latter module, builds the forecasts that will be sent to a decision agent (e.g., control system). In principle, any forecasting method can be adopted for *RTF*, if the temporal requirement is fulfilled. On the other hand, real-time series can have hundreds or even thousands of observations, forming patterns that can change dynamically in time. To overcome these hurdles, two strategies will be adopted:

- the forecasting models will have fixed structures (i.e., with a predefined number of parameters) and;
- only recent events will be considered, when estimating a forecasting model.

The first approach simplifies the forecasting process, reducing the computational requirement to the optimization of the model's parameters. The second approach makes use of a *sliding time window*, which de-

fines the available forecasting data, at present time t. The sliding window has the advantage of reducing the number of learning examples, while keeping a medium-term memory.

The whole dynamic forecasting process will be defined in terms of the *Sampling Period (SP)*, the number of *Ahead Forecasts (AF)*, the sliding time *Window Displacement (WD)* (applied each period) and the *Window Length (WL)* (Figure 2).

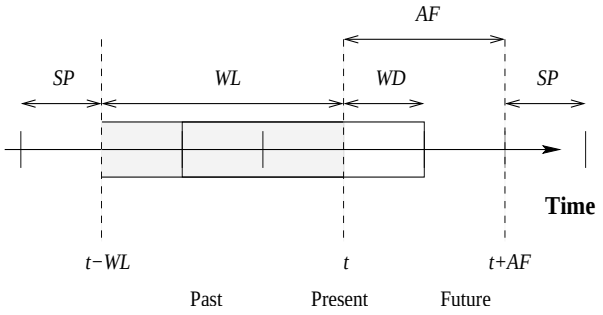

Figure 2. The *RTF* dynamic process.

Since the first three factors are defined by the problem, the tunning of the process will depend on the *WL* value. A large value will difficult the model's estimation step, while a small one will provide insufficient information.

4. Artificial Neural Networks

An *Artificial Neural Network (ANN)* is made up by simple processing units, the *neurons*, which are connected in a network by synaptic strengths, where the acquired knowledge is stored. One can find a kaleidoscope of different *ANNs*, that diverge on several features, such as the learning paradigm or the internal architecture [9]. In a *Feedforward Neural Network (FNN)*, neurons are grouped in layers and only forward connections exist. This provides a powerful architecture, capable of learning any kind of continuous nonlinear mapping, with successful applications ranging from *Computer Vision*, *Data Analysis* or *Expert Systems*, just to name a few. *FFNs* are usually trained by gradient descent algorithms, such as the popular *Backpropagation*, or fast variants like *RPROP* [12].

The use of *ANNs* for *TSF* began in the late eighties, with encouraging results, namely when applied to financial markets, and the field has been consistently growing since [13].

FFNs can perform one step ahead forecasts by feeding its inputs with n past values. In previous work [5], *FFNs* have been successfully applied to forecast regular time series, using topologies with one hidden layer, one output node, *bias* and shortcut connections (Figure 3). To enhance nonlinearity, the *logistic* activation function was applied on the hidden nodes, while in the output node, the *linear* function was used instead, to scale the range of the outputs (the logistic function has a [0,1] codomain). This solution avoids the need of filtering pro-

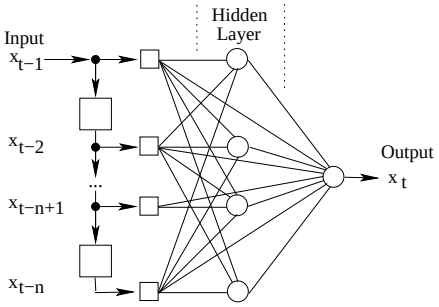

Figure 3. The *FNN* structure.

cedures, which may give rise to loose information (e.g., *rescaling*). The general model provided by the *FNN* is given in the form:

$$\widehat{x}_t = w_{o,0} + \sum_{i=1}^{n} x_{t-i} w_{o,i} + \sum_{i=n+1}^{o-1} f\left(\sum_{j=1}^{n} x_{t-j} w_{i,j} + w_{i,0}\right) w_{o,i} \quad (5)$$

where $w_{i,j}$ denotes the weight of the connection from node j to i (if $j = 0$ then it is a *bias* connection), o the output node, f the logistic function ($f(x) = \frac{1}{1+e^{-x}}$), and n the number of input nodes.

5. Genetic and Evolutionary Algorithms

The term *Genetic and Evolutionary Algorithms (GEAs)* is used to name a family of computational procedures where a number of potential solutions to a problem makes the way to an evolving population. Each individual codes a solution in a string (*chromosome*) of symbols (*genes*), being assigned a numerical value (*fitness*), that stands for a solution's quality measure. New solutions are created through the application of genetic operators (typically *crossover* or *mutation*). The whole process evolves via a process of stochastic selection biased to favor individuals with higher fitnesses.

When one is faced with problems where the parameters are given by real values, the best strategy is to represent them directly into the chromosome, using a *Real-Valued Representation (RVR)*, which allows the definition of richer genetic operators [11].

It is surprising to realize that the work in applying *GEAs* to forecasting is so scarce. In fact, although there are some publications in this area, these are not numerous nor noticeable. The existent work focuses mainly in some kind of parameter optimization, under conventional models such as *Holt-Winters* [1] or *ARIMA* [3].

In past work [6], the authors used *GEAs* with *RVRs* to optimize the coefficients of *ARMA* models, where the genes in the chromosome code for the weights by which previous values and errors are multiplied, being the model given by the equation:

$$\widehat{x}_t = g_0 + \sum_{i \in \{1,\ldots,P\}} g_i x_{(t-i)} + \sum_{i \in \{1,\ldots,Q\}} g_{(i+P)} e_{(t-i)} \quad (6)$$

where g_i stands for the i-th gene of the individuals' chromosome.

6. Some Experimental Results

To the experiments carried out in this work, two real-time series, from the *geology* and *health* sciences, were selected (Figure 4):

kobe a non-linear series, with a total of 1200 observations, taken each second from a seismograph of the Kobe earthquake [10]; and

heart a trended series, referring to 2400 recordings (with a 0.5 second sampling), of the heart rate of a Boston hospital patient [15].

Both these series present important feature variations through time, due to the effects of an earthquake (**kobe**) and sleeping apnea states (**heart**).

The *RTF* system values were set to $SP = 1s$, $AF = 1$ and $WD = 1$, for the **kobe** series, and $SP = 0.5s$, $AF = 2$ and $WD = 2$, for the **heart** one.

The two bio-inspired models (based on *ANNs* and *GEAs*) were compared to the *Holt-Winters (HW)* one. A fixed *ANN* topology with ten input and three hidden nodes was used, being the training performed by the *RPROP* algorithm. In the case of the *GEA*, n was also set to ten, being used an $ARMA(10, 10)$ model. The population size was set to 50, being the *arithmetical* crossover [11] responsible for breeding $\frac{2}{3}$ of the offspring and a *gaussian mutation* operator accountable for the remaining ones. For the *HW* method, non-seasonal models were considered, being the smoothing and trend coefficients set by a grid search. The *WL* was set to 100, a configuration that allows a fast learning with a sufficient number of training samples (e.g., in a 450MHz Pentium III machine, 90 iterations of the *FNN* training and 35 of the *GEA* can be executed in one second). Finally, the forecasting values were only accounted after 200 seconds, a start-up time set to permit the tunning of each model. Thus, the evaluation will occur throughout the rest 1000 seconds, in a total of 1000 (2000) forecasts for the **kobe** (**heart**) series.

All experiments reported in this work were conducted using programming environments developed in *C++*, under the *Linux* operating system. For the bio-inspired models, thirty independent runs were performed in every case to insure statistical significance. Since results may depend on the computational power available, all simulations were tested in three different machines (Table 1). For the *HW* method, a grid search with a 0.1 step in machine **A**, 0.05 in machine **B** and 0.01 in machine **C** was applied, to estimate the α and β coefficients.

The overall results are condensed in Table 2, in terms of the $RMSE$ and $MNQE$ (in brackets) forecasting errors, being the bio-inspired results (columns **GEA** and **FNN**) presented as the mean of the thirty runs.

The results show that for the *HW* method, there is no significative improvement when enlarging the search space (defined by the grid search). In contrast, the computational power affects the bio-inspired techniques.

Table 1. The computer machines.

Machine	Processor	Frequency	Cache
A	Pentium II	350MHz	512kB
B	Pentium III	450MHz	512kB
C	Pentium III	933MHz	256kB

However, for the *FNNs*, the forecasting performance improvement does not follow the computational growth.

For instance, for the **kobe** series, there is a 0.2% decrease in the forecasting error, when moving from machine **A** to **B**, and from machine **B** to **C**. Yet, in the first case there is an 30% increment in the computational power, while in the second case the magnification is about 100%. This fact can be explained by the nature of the *FNN* learning; i.e., as time goes by, convergence is much slower. This behavior does not occur in the same manner with the *GEAs*, which present higher error variations (1.3% and 2.1%), which can be explained by a slower learning (one *GEA* iteration takes approximately the double computation time, when compared to a *FNN*).

As an example, Figure 5 plots the last 50 real-time forecasting errors for the **kobe** series, as computed by machine **B**. For the bio-inspired models, it was decided to use the average of the thirty runs. The errors obtained by the *HW* method present a high amplitude, far away from the zero axis (which denotes a perfect forecast). In contrast, the bio-inspired techniques present better performances.

The example is consistent with the results of Table 2, where the *HW* presents an excessive forecasting error, failing to predict the nonlinear behavior of the **kobe** series. Although the *HW* performance improves for the second series (**heart**), it is still outperformed by the bio-inspired outcomes. The *FNNs* lay out even better forecasts than the *GEAs*, albeit the difference shrinks when the computational power increases (machine **C**).

7. Conclusions and Future Work

The surge of new bio-inspired optimization techniques, such as *ANNs* and *GEAs*, has created new exciting possibilities for the field of forecasting. In this work, a autonomous real-time forecasting system for bio-inspired models is applied, assuming no prior knowledge over each series.

Comparative experiments, among conventional and bio-inspired approaches, have shown better performances for the latter ones, specially when considering the connectionist models.

In future research it is intend to explore different *ANNs* topologies, such as *Recurrent Neural Networks* or *Radial Basis Functions*. Another area of interest my rely in the enrichment of the *GEA* forecasting models with the integration of nonlinear functions (e.g., logarithmic or trigonometric). Finally, one promising field is the application of similar approaches to real-world applications (e.g., *bioengineering* or *intensive care unit* control systems).

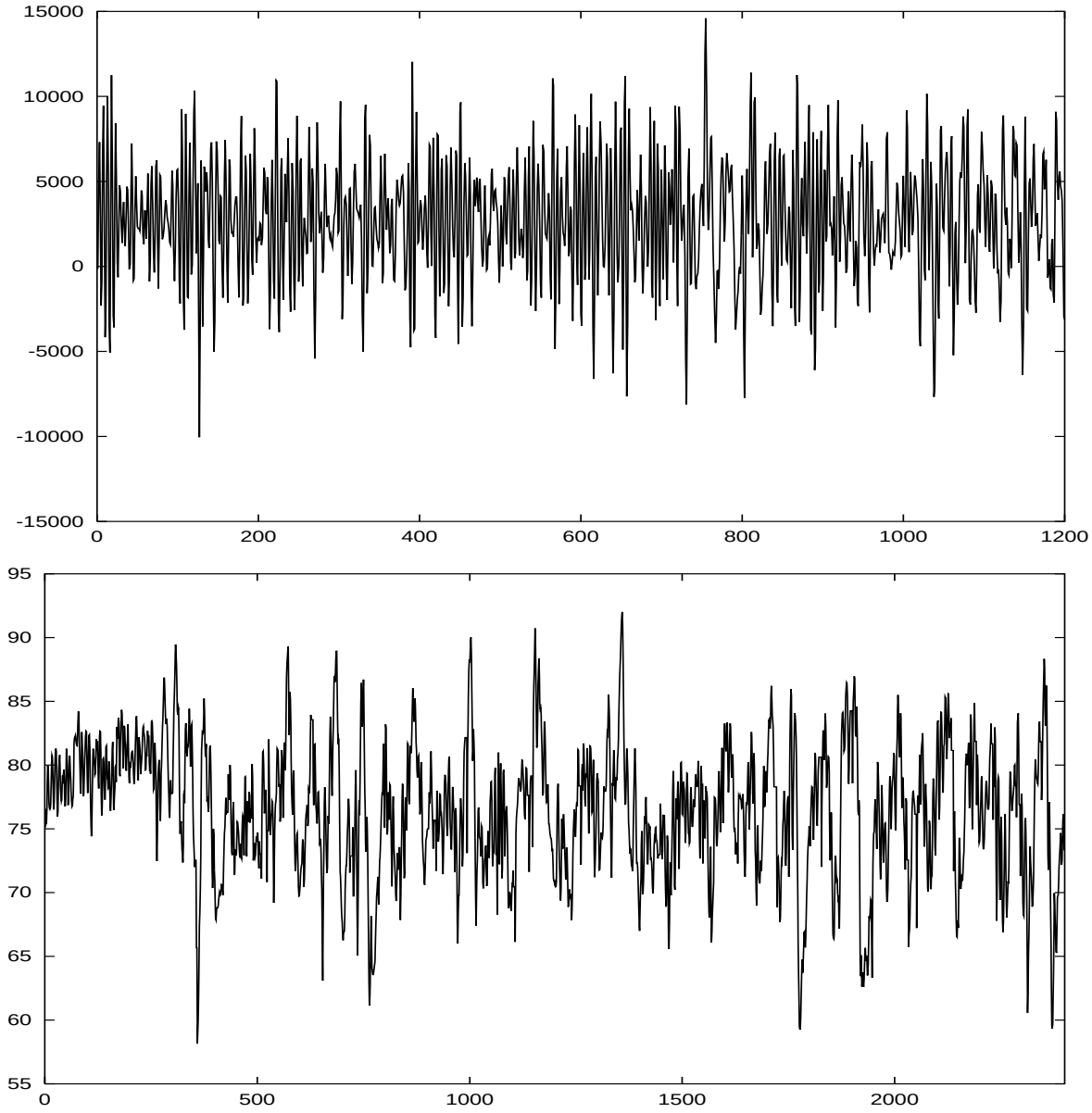

Figure 4. The **kobe** and *heart* series.

Acknowledgements

The authors would like to acknowledge the support from FCT, given under the project POSI/EEI/13096/2000.

References

[1] A. Agapie and A. Agapie. Forecasting the Economic Cycles Based on an Extension of the Holt-Winters Model. A Genetic Algorithms Approach. In *Proc. of the IEEE Comp. Intelligence for Financial Forecasting Engineering*, pages 96–99, 1997.

[2] G. Box and G. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden Day, San Francisco, USA, 1976.

[3] C. Chai, C. Chuek, M. DP, and T. Huat. Time Series Modelling and Forecasting using Genetic Algorithms. In *Proc. of the First International Conference on Knowledge-Based Intelligent Electronic Systems*, volume 1, pages 260–268, 1997.

[4] W. Charytoniuk and M. Chen. Very Short-Term Load Forecasting Using Artificial Neural Networks. *IEEE Trans. on Power Systems*, 15(1):263–268, February 2000.

[5] P. Cortez, M. Rocha, and J. Neves. Evolving Time Series Forecasting Neural Network Models. In *Proc. of Int. Symposium on Adaptive Systems: Evolutionary Computation and Probabilistic Graphical Models (ISAS 2001)*, March 2001.

[6] P. Cortez, M. Rocha, and J. Neves. Genetic and Evolutionary Algorithms for Time Series Forecast-

Table 2. Comparison among the different forecasting methods.

Series	Machine	HW	GEA	FFN
	A	3269 (93%)	736 (5.0%)	590 (3.0%)
kobe	**B**	3278 (93%)	696 (4.3%)	572 (2.8%)
	C	3286 (93%)	**652** (3.7%)	**547** (2.6%)
	A	1.93 (13.7%)	1.91 (13.1%)	1.68 (10.4%)
heart	**B**	1.93 (13.7%)	1.87 (13.0%)	1.68 (10.3%)
	C	1.93 (13.7%)	**1.73** (11.0%)	**1.66** (10.1%)

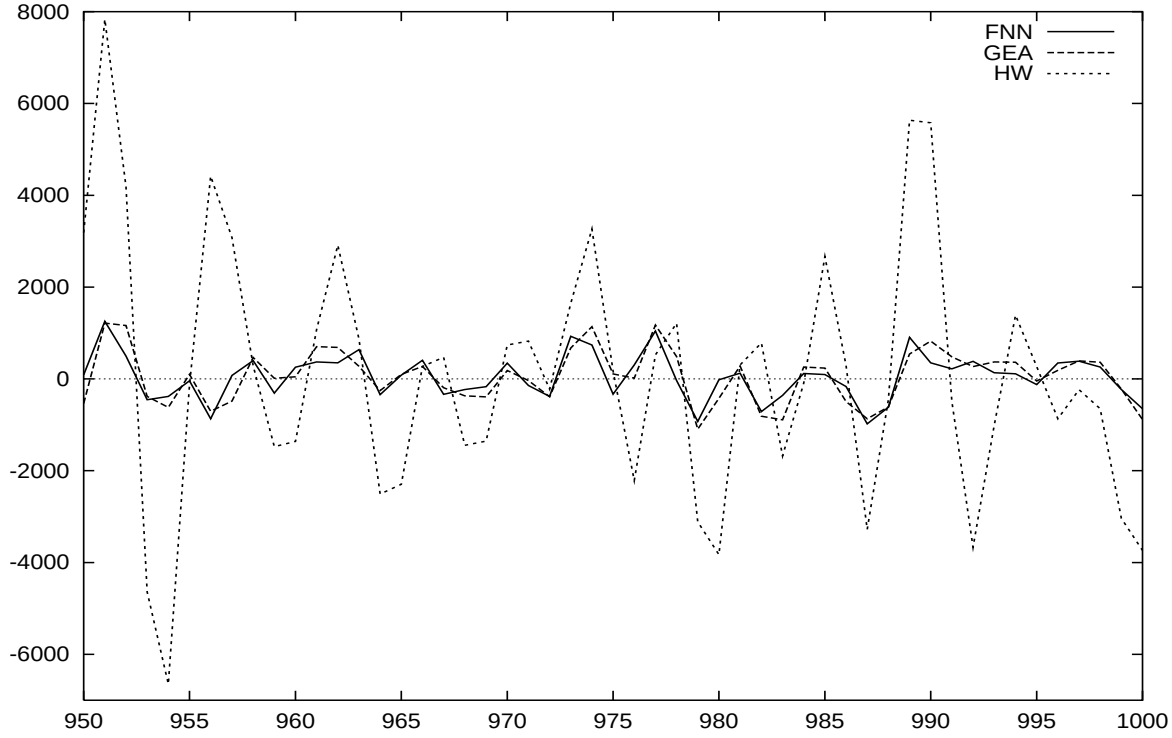

Figure 5. The last 50 forecasting errors for the **kobe** series.

ing. In L. Monostori, J. Váncza, and M. Alis, editors, *Engineering of Intelligent Systems: Proc. of IEA/AIE 2001, LNAI 2070*, pages 393–402. Springer-Verlag, June 2001.

[7] N. Gong and T. Denceux. Urban storm water pollution forecasting using recurrent neural networks. *Journal of Systems Engineering*, 6:137–147, 1996.

[8] J. Hanke and A. Reitsch. *Business Forecasting*. Allyn and Bancon Publishing Company Inc., Massachussetts, USA, 1989.

[9] S. Haykin. *Neural Networks - A Compreensive Foundation*. Prentice-Hall, New Jersey, 2nd edition, 1999.

[10] R. Hyndman. *Time Series Data Library*. Available from http://www-personal.buseco.monash.edu.au/-˜hyndman/TSDL/, 2001.

[11] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, USA, third edition, 1996.

[12] M. Riedmiller. Supervised Learning in Multilayer Perceptrons - from Backpropagation to Adaptive Learning Techniques. *Computer Standards and Interfaces*, 16, 1994.

[13] E. Shoneburg. Price Prediction using Neural Networks: A Project Report. *Neurocomputing*, 2:17–27, 1990.

[14] Z. Tang and F. Fishwick. Feed-forward Neural Nets as Models for Time Series Forecasting. *ORSA Journal of Computing*, 5(4):374–386, 1993.

[15] A. Weigend and N. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley, USA, 1994.