2 NEURO-ADAPTIVE MODEL FOR FINANCIAL FORECASTING

Iulian NĂSTAC*
Emilian DOBRESCU**
Elena PELINESCU***

Abstract

The paper advances an original artificial intelligence-based mechanism for specific economic predictions. The aim is to forecast the exchange rate of euro versus the Romanian currency using a large set of financial data. The possible influence of specific forecasting indicators (such as Sibiu Futures Stock Exchange market) on the evolution of the exchange rate in Romania is also analyzed. The time series under discussion are inherently non-stationary. This aspect implies that the distribution of the time series changes over time. The recent data points could provide more important information than the far distant data points. Therefore, we propose a new adaptive retraining mechanism to take this characteristic into account. The algorithm establishes how a viable structure of an artificial neural network (ANN) at a previous moment of time could be retrained in an efficient manner, in order to support modifications in a complex input-output function of a financial forecasting system. In this system, all the inputs and outputs vary dynamically, and different time delays might occur. A "remembering process" for the former knowledge achieved in the previous learning phase is used to enhance the accuracy of the predictions.

The results show that the first training (which includes the searching phase for the optimal architecture) always takes a relatively long time, but then the system can be very easily retrained, since there are no changes in the structure. The advantage of the retraining procedure is that some relevant aspects are preserved ("remembered") not only from the immediate previous training phase, but also from the previous but one phase, and so on. A kind of "slow forgetting process" also occurs; thus for the ANN it is much easier to remember specific aspects of the previous training instead of the first training.

The experiments reveal the high importance of the retraining phase as an upgrading/updating process and the effect of ignoring it, as well. There has been a

* Senior Assistant Professor (Lecturer), Polytechnic University of Bucharest, nastac@ieee.org.
** Professor, Senior Researcher, National Institute of Economic Research, Romanian Academy, emiliand@clicknet.ro.
*** Senior Researcher, Institute of Economic Forecasting, Romanian Academy, Elena.Pelinescu @itcnet.ro.

decrease in the test error when successive retraining phases were performed and the neural system accumulated experience.

Keywords: Neural networks, exchange rate, adaptive retraining, delay vectors, iterative simulation

JEL Classification: C45, C53, F47

1. Introduction

The forecasting of the exchange rate in Romania as in many other countries also recently integrated into the European Union is very important for any operator on the monetary and forex market. Generally, the exchange rate is modeled using explanatory variables such as:

- the monetary indicators (Dornbush 1994, Mishkin 2001, Fair 1994, Adams and Dixon 1989, Krugman and Obstfeld 2000, de Bondt *et al.*, 1997, Weyerstrass 2000, Matthews 1985),
- its previous levels (Jahnke *et al.*, 2000),
- the domestic inflation and the foreign capital inflows (Bergstrom *et al.*, 1994, Anderson 1990, Neu 1990, Abel and Bernanke 2001).

In the last version of the Romanian macromodel (Dobrescu 2006) that refers to the annual indicators, beside the actual sluggishness, two factors are particularly involved: the domestic inflation and the foreign capital inflows. The present attempt is dedicated to shorter predictions and involves a large set of available information. The behavior of the operators is based on current information offered by many institutions, some of the data are statistical and others are forecasted. Synthetically, the data base used in simulations is described by the following parameters (Table 1):

Table 1

Database indicators

	Indicators	Symbol	Frequency
I.	**Statistical information**		
A.	***General information***		
1	1. Real Gross Domestic Product growth	GDP	Quarterly
2	2. Current Account deficit	CA	Monthly
3	3. Consolidated general budget deficit as percentage on GDP	CGD	Quarterly
4	4. Net foreign direct investment	FDI	Monthly
5	5. Medium and long term external dept	ExD	Monthly
6	6. NBR Foreign exchange reserve	ER	Monthly
7	7. Export of good and services	X	Monthly
8	8. Import of good and services	M	Monthly
9	9. Net monthly average wage on the economy	Nw	Monthly

	Indicators	Symbol	Frequency
B.	***Specifics information***		
10	1. Exchange rate Dollar/ROL	E$	Daily
11	2. Exchange rate EUR/ROL	Eeur	Daily
12	3. Consumer goods index	CPI_R	Monthly
13	4. Monetary base M0	M0	Monthly
14	5. Reference rate of BNR	r_d	Monthly
15	6. Speed between lending and deposit average interest rata of banks for non-government, non-banks clients	Δr	Monthly
16	7. Total domestic credit	DC	Monthly
17	8. Portfolio investment, sold	PI	Monthly
18	9. Current transfers and incomes	CTI	Monthly
19	10. Turnover	T	Monthly
20	11. BET Index	BET	Daily
C.	***External Information***		
21	1. Ratio EUR/Dollar	Ra	Daily
22	2. Exchange rate Euro/ROL	$Reur_{EU}$	Daily
23	3. Refinancing ECB interest rate	R_{ecb}	Monthly
24	4. Brent oil price	op	Monthly
25	5. HIPC (EU 27)	HIPC	Monthly
II:	**Prospective information**		
A.	***A. General information***		
26	1. Real GDP growth	GDP_f	Annual
27	2. Export of goods and services, FOB, growth rate	X_f	Annual
28	3. Import of goods and services, FOB, growth rate	M_f	Annual
29	4. Commercial trade deficit, mill Euro	Ct_f	Annual
30	5. Growth of consumer price, annual average	CPI_f	Annual
31	6. Growth of consumer price, December/December	$CPId_f$	Annual
B.	***Specific forecasting information***		
32	1. Inflation target	IT_f	Annual
33.	2. Future exchange rate Dollar/ROL, 1 month	Fe$	Daily
34	3. Future exchange rate Euro/ROL 1 month	FeEur	Daily
35	4. Ratio EUR/Dollar, 1 month	Fra	Daily

The diversity of the frequency and period of these variables is difficult to introduce into a classical model, so we decided to use an artificial intelligence technique, which could be a useful instrument in macroeconomic analysis and prediction of the exchange rate in Romania. We consider that the advantages of artificial neural network in that application are: the capability to process a large metadata base that contains both statistical and forecasting information; the elimination of the restrictions regarding stationarity of the data series; the ability to extract significant information from its training data; and the possibility to introduce a new adaptive retraining mechanism, which takes to into account the fact that the recent data points could provide more important information as compared to the distant past ones. This way, one may analyze, for instance, if the forecasting information like the ones provided by the Sibiu Futures Stock Exchange market influences the behavior of the agents.

Artificial neural networks (ANNs) have been widely applied to forecasting problems (Nastac 2004, Huang and Lewis 2003, Zhang *et al* 1998). There is considerable interest in the development of reliable forecasting models for financial applications. Models based on the ANN have been found to be suitable for certain applications where other techniques failed. The idea that the ANN can be used for a better understanding of the economic complex mechanisms is found in the literature (Salzano 1999, Shadbolt 2002, Zhang 2003).

The goal of our research was to find a practical mathematical model that describes the relationship between a set of input variables and one output variable that models the EUR/ROL exchange rate. All inputs and the output vary dynamically, and different time delays might occur. Changing an input variable may result in an output change that starts only a day later and goes on for up to several days.

The entire amount of data consists of more than 2500 rows (time steps) – one data row every day over 7 years (January 2000 – December 2006). The time series under discussion are non-stationary. The non-stationary characteristic implies that the distribution of the time series changes over time. The recent data points could provide more important information than the distant data points. Therefore, we propose a new adaptive retraining mechanism to take this characteristic into account.

This paper is organized as follows. Section 2 presents the issue that concerns the model structure and data preprocessing. Into the next section, we introduce the adaptive retraining technique and explain our approach. The main features of the experimental results are given in Section 4. Finally, Section 5 concludes the paper.

2. The Model Architecture

The time delay or dead time is frequently encountered in financial systems. It is well known that feedback control in the presence of time delay leads to particular difficulties, since a delay places a limit on the time interval.

Figure 1 shows our idea of training a feed-forward ANN such that the latter becomes a predictor. We use delayed rows of more than 30 input data (see the final part of this section) to simulate the current states of the EUR/ROL exchange rate. For learning purposes, the network inputs involve many blocks with several time-delayed values of financial system inputs, and fewer blocks with system delayed output. The ANN target-output consists of the current value of the corresponding EUR/ROL exchange rate. Therefore, the system tries to match the current values of the output, by properly adjusting a function of the past values of the inputs and output (Figure 1).

At the current moment, t, the output (see Figure 1) is affected by the **P** inputs at different previous time steps ($t - i\_d_1, \ldots, t - i\_d_n$), and also by the outputs at other previous time steps ($t - o\_d_1, \ldots, t - o\_d_m$), respectively. We denote by *In_Del* and *Out_Del* two *delay vectors* that include the delays that we take into account:

$$In\_Del = \left[i\_d_1, i\_d_2, \ldots, i\_d_n \right] \qquad (1)$$

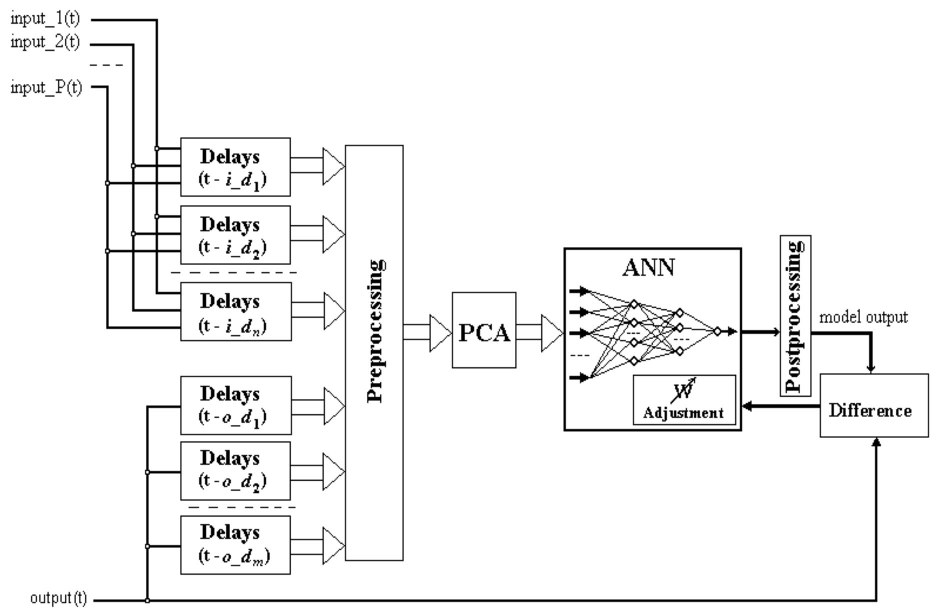
and

$$Out\_Del = [o\_d_1, o\_d_2, ..., o\_d_m] \quad (2)$$

where n > m.

Figure 1

The forecasting architecture. The training process



For *In_Del*, we use various delay vectors with *n* = 7, 8 or 9 elements, whose values are within a range of twenty days. Regarding *Out_Del*, we employ different combinations, with *m* = 3, 4 or 5 elements, covering about one week. The distribution of the vector elements is preferably (but not compulsory) chosen similarly to the Gamma distribution. The elements of each vector are in ascending order. Consequently, the maximum values of any delay vector are i_d_n or o_d_m, respectively. The recurrent relation performed by the model is as follows:

$$y(t+1) = F(X(t+1-In\_Del(i)), y(t-Out\_Del(j))) \quad (3)$$

where X is the input vector; $i = 1,...,n$ and $j = 1,...,m$.

We use feed-forward ANNs with two hidden layers in order to achieve a good approximation function, based on our preliminary research, where we have obtained better results in the case of two hidden layers than in the case of one hidden layer, however maintaining a similar ratio (approx. 5/1) between the number of training samples and the total number of weights. The ANN models, depicted in Figure 1, use training sets of $V\text{-}i\_d_n$ input-output pairs for model adaptation (see the next section), where *V* = 2240 is the initial time steps interval employed for the training purpose.

Once we have established all the influences on the output at the moment *t*, we apply Principal Component Analysis (PCA) (Jackson 1991) to reduce the dimensionality of the input space and to un-correlate the inputs. Before applying PCA, we preprocessed the inputs and outputs, by replacing the missing data using the previously available values and, then, by applying the normalization. Data preprocessing prepares the raw data for the forecasting model and turns it into a format that will be easier and more effectively processed. Finally, we have applied the reverse process of normalization, in order to de-normalize the simulated outputs. Data preprocessing and data post-processing are essential steps of the knowledge discovery process in the real world applications, and they greatly improve the network's ability to capture valuable information, if they are correctly carried out (Hagan *et al*., 1996, Basheer *et al*., 2000).

Our attempt involves a number of **P** variables (more than 30). Statistical data have different frequencies, such as:

- daily frequency (forex exchange rate, future exchange rate for one month and the BET index);
- quarterly frequency (GDP, the share of consolidated budget in GDP);
- monthly frequency (CPI, interest rate, exports and imports of goods and services, etc).

In order to use all these data with different frequencies, we decided to transform them into such data of higher frequency, on the basis of a natural formation mechanism of the market operators' behavior that implied to keep the information unchanged during the period between two updating time steps. For example, if we have only annual data, we keep it unchanged during 365 or 366 days. For the days without transactions, we decided to keep the previous transaction figure, in order to have a complete data series.

In our model we used two different kinds of data:

- *Statistical data* that were classified as:
  - *General data* that characterizes the macroeconomic development of Romania (9 indicators);
  - *Specific* data that are directly linked to the exchange rate evolution (11 indicators);
  - *External data* that refer to significant indicators of external market evolution, focused on the European Union market and the US market (6 indicators).
- *Forecasting data* that were also classified as:
  - *General data* that characterize the macroeconomic development of Romania (10 indicators);
  - S*pecific* data that are directly related to the exchange rate evolution (4 indicators);
  - There is a possibility to use E*xternal data* also, but we decided to employ them in a further investigation that is not the subject of this paper.

All the previous indicators used in the model are presented in Table 1.

- Additionally, we introduced a Month indicator L (the days of January are denoted by 1, the days of February by 2 and so on).
- We also tested the influence of the other three supplementary inputs that represent the "Sibiu Futures exchange rate of one month" for EUR/ROL, USD/ROL and EUR/USD exchange rates.

The period covered by analysis is from the beginning of 2000 to the end of 2006. The possible connection between the exchange rate and other mentioned variables has been checked using the Granger causality tests (Granger 1969, Granger 1988), which were computed for different number of lags (starting from 26). As expected for a daily analysis, such interdependence is clearly revealed in the case of 1-2 lags, so that an artificial neural network based on the previously mentioned variables can be considered as economically consistent.

3. The Adaptive Retraining Procedure

The feature of universal functional approximator (Hornik *et al*., 1989) adds the power and flexibility of the neural networks to the process of learning complex patterns and relationships. However, the potential risk of using the universal approximator is the over fitting problem, since it is often easy to train a large network model to learn the peculiarities, as well as the underlying relationship. Therefore, the balance between the learning capability and the generalization power is very important in neural network forecasting applications.

As basic training algorithm, we use the Scale Conjugate Gradient (SCG) algorithm (Moller 1993). In order to avoid the over fitting phenomenon, we apply the early stopping method (*validation stop*) (Hagan *et al.*, 1996) during the training process.

Next, the adaptivity of the result is performed (and improved), by using the *retraining technique* (Năstac, 2004, Năstac and Matei, 2003), in a special way. This technique is a mechanism for extracting practical information directly from the weights of a reference ANN that was already trained in a preliminary phase. The retraining procedure reduces the reference network weights (and biases) by a *scaling factor* γ, $0<\gamma<1$. The reduced weights are further used as the initial weights of a new training sequence, with the expectation of a better accuracy.

Briefly, the entire technique can be summarized by the following phases:

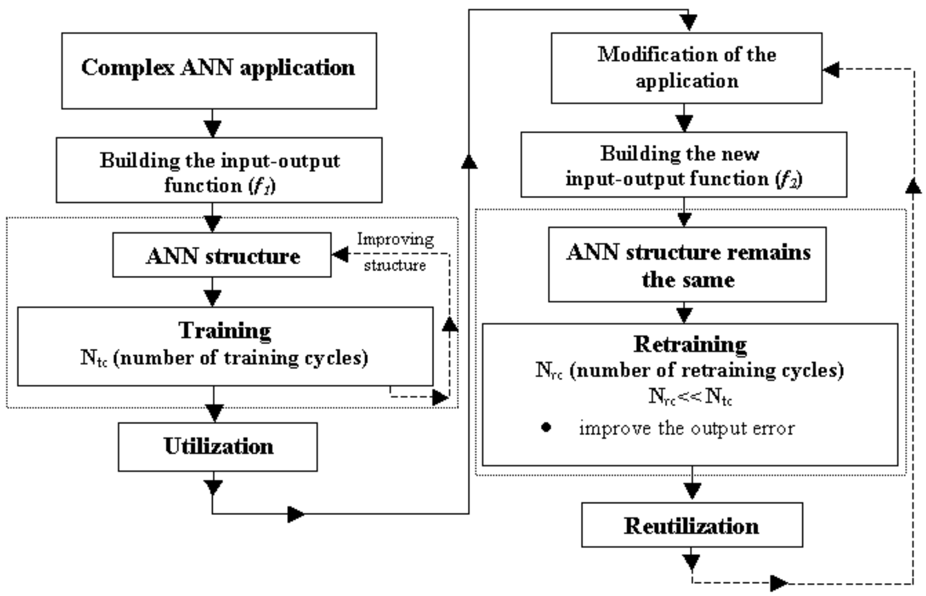
- Training an Artificial Neural Network in a natural way with *validation stop,* and with the weights initialized to small, uniformly distributed values;
- Reducing the first network weights and biases by a *scaling factor* γ ($0 < \gamma < 1$);
- Retraining the network with the new initial weights;
- Comparing the *validation error* (or training error) in both cases.

An advantage of this technique is a significant decrease in the number of training cycles, as compared to the classical training methods (Năstac and Matei, 2003).

The retraining technique allows us to improve continuously the model, at times, by using new (shifted) databases (see Figure 2).

Figure 2

The retraining technique (step by step)



The data that we have used in our model consist of V-i_d_n input-output pairs during each training (or retraining) phase, where V = 2200 (January 1, 2000 – January 8, 2006) is the initial time steps interval employed for the training purpose. As splitting criterion, we randomly choose approximately 85% of the data (V-i_d_n) for the training set, and the rest for validation. Furthermore, we imposed the supplementary condition:

$$E_{val} \leq \frac{6}{5} \cdot E_{tr} \tag{4}$$

to avoid a large difference (more than 20%, see Figure 3) between the error of the training set (E_{tr}) and the error of the validation set (E_{val}). In this way, the over fitting phenomenon on the test set will be considerably reduced. In our approach, the validation set acts at the same time as a kind of test set, although there is a real and separate test set of T = 20 different and successive time steps (where T<<V).

Next, we describe the *steps* that we have taken to adapt our model:

1. Firstly, we set the proper number of hidden neurons for each hidden layer (N_{h1} and N_{h2}). Each of the training sessions started with the weights initialized to small uniformly distributed values (Hagan *et al*., 1996, Năstac and Matei, 2003). We tested several pyramidal ANN architectures, with N_{h1} and N_{h2} taking values in the vicinity of the geometric mean (Basheer *et al*., 2000) of the neighboring layers, and observing the following rules:

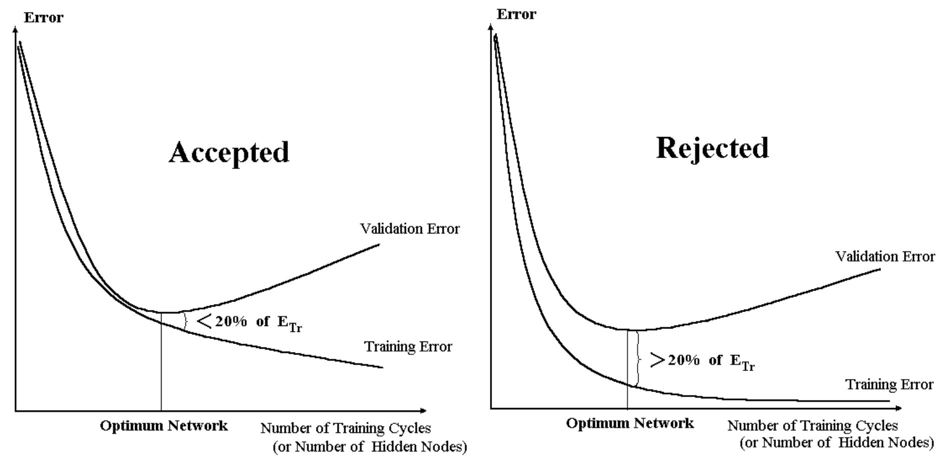
$$N_i \geq N_{h1} \geq N_{h2} \geq N_o \tag{5}$$

$$\sqrt{N_i * N_{h2}} + 5 \geq N_{h1} \geq \sqrt{N_i * N_{h2}} - 5 \quad (6)$$

$$\sqrt{N_{h1} \cdot N_o} + 5 \geq N_{h2} \geq \sqrt{N_{h1} \cdot N_o} - 5 \quad (7)$$

Figure 3

A supplementary condition over the validation set



Above, N_i is the number of inputs after the PCA block and N_o is the number of the outputs (N_o=1). Each architecture was tested five times, with random initial settings of the weights and different training/validation sets. We chose the best model with respect to the smallest error between the desired and simulated outputs. This error (E_{tot}) was calculated for V-i_d_n data that included both training and validation sets.

2. Secondly, we predicted the T values of the outputs (during the interval (V+1) - (V+T)), in a sequential mode. Let us call this step the **Iterative Simulation** (IS) of the output. Therefore, in order to produce one output at time step *t*, the neural network used as input the estimated outputs (besides the real inputs) that were calculated at the previous steps, by using other simulated outputs, and so on. Applying this iterative process, a forecast may be extended as many steps as required, nevertheless taking the risk that each step increases the forecasting error.

   Then, we computed the error *ERR* (Nastac 2004) that represents the accuracy of the approximation of the output data, within the forecasting horizon of *T* time steps:

$$ERR = \frac{100}{T} \sum_{p=1}^{T} \frac{|O_{Rp} - O_{Fp}|}{|O_{Rp}|} \cdot f(p) \quad (8)$$

where T = number of time steps (days)

O_{Rp} = real output at time step p

O_{Fp} = forecasted output at time step p

and $f(p) = \dfrac{T}{T + p}$ a weight function that decreases with the number of time steps *p*.

3. Thirdly, we applied the retraining technique for a shifted interval of time steps (*Shift*+1) – (*Shift*+*V*), where *Shift* $\leq$ *T*. Here we used the ANN architecture that resulted at the end of the previous step. We applied this technique for each value of γ (γ = 0.1, 0.2, …, 0.9), keeping the neural network (weight distribution) that achieved the minimum error as the reference network. We repeated this step five times, and we randomly reconstructed the training and validation sets each time.
4. Fourthly, we predicted the *T* values of the outputs (during the interval of time steps (*Shift*+*V*+1) - (*Shift*+*V*+*T*)) in the same sequential mode as in step 2 (Iterative Simulation).
5. We repeated *L* times the steps 3 and 4 at successive shifted intervals of *V* time steps for the retraining processes and *T* time steps for the sequential forecasting. Each time the intervals were ascendingly repositioned with *Shift* time steps (days).

Firstly, a decisive role in choosing the best model is played by the mean square error of the differences between the real and the simulated outputs of V-i_d_n data rows, which included both the training and the validation sets. Afterwards, the retraining technique adapts the ANN system, in order to learn continuously the latest evolution of the financial process. The retraining process can be viewed as a “remembering process” of the former knowledge achieved in the previous learning phases. Figure 4 illustrates the evolution of the trainings. The retraining technique allows us to improve continuously the model, at times, by using new (shifted) databases (see Figure 4). For a single combination of the delay vectors, we obtained (and used) a model with its associated adaptive behavior. The above-mentioned steps were applied to different delay vectors.

Figure 4

The training and retraining phases

4. Experimental Results

We performed the steps described in Section 3 for various combinations of delay vectors, such as:

Case I: *In_Del* = [1 2 3 4 5 6 8 12] and *Out_Del* = [0 1 2 4].

Case II: *In_Del* = [2 3 4 5 6 7 9 13] and *Out_Del* = [0 1 2 4].

Case III: *In_Del* = [3 4 5 6 7 8 10 14] and *Out_Del* = [0 1 2 4].

Case IV: *In_Del* = [4 5 6 7 8 9 11 15] and *Out_Del* = [0 1 2 4].

Case V: *In_Del* = [5 6 7 8 9 10 12 16] and *Out_Del* = [0 1 2 4].

Case VI: *In_Del* = [6 7 8 9 10 12 16] and *Out_Del* = [0 1 2 4].

Case VII: *In_Del* = [7 8 9 10 11 13 16] and *Out_Del* = [0 1 2 4].

Each time, the first step, which determines the optimum architecture, required a somewhat longer time (approx. one day). The optimum architecture highly depends on the delay vectors. Then, for each retraining phase the program worked for about one hour. There was a clear difference between the first training process, which needed a long time to search for the best architecture, and the retraining on the other hand. It can be quite easy to retrain a good ANN architecture several times, by using a shifted training set.

Next, we shall concentrate especially on Case I, since it is very useful to predict the exchange rate for the next day having all the information up to the present day (see formula 3 when using the delay vectors of the first Case). In this case, Table 2 shows the values of the test error (*ERR* according to (8)) for iterative simulations of the output, computed at the end of the first training and, then, after each successive retraining phase (L = 40), when using different values of the prediction horizon (T).

Table 2

Evolution of Test Error (ERR for the Iterative Simulations of Output) of Case I during the training and retraining phases for different values of T

	Training / retraining interval	Time period for training	ERR (Case I)					
			33 inputs					36 inputs
			T=1	T=3	T=7	T=15	T=30	T=30
First training	1 – 2200	01.01.2000 - 01.08.2006	2.1387	2.8856	3.7954	4.2723	4.3185	6.8036
Retraining 1	2 – 2201	01.02.2000 - 01.09.2006	0.25772	0.3848	0.63175	0.53869	0.5592	1.6479
Retraining 2	3 – 2202	01.03.2000 - 01.10.2006	0.45772	0.49306	0.44378	0.37196	0.73979	1.7582
Retraining 3	4 – 2203	01.04.2000 - 01.11.2006	0.21918	0.76882	1.0928	1.2134	0.9866	3.6997
Retraining 4	5 – 2204	01.05.2000 - 01.12.2006	0.80736	1.0125	1.2086	1.2213	0.95671	0.99947
Retraining 5	6 – 2205	01.06.2000 - 01.13.2006	0.049294	0.085675	0.22419	0.2126	0.44314	0.65897

	Training / retraining interval	Time period for training	ERR (Case I)					
			33 inputs					36 inputs
			T=1	T=3	T=7	T=15	T=30	T=30
Retraining 6	7 – 2206	01.07.2000 - 01.14.2006	0.015575	0.12337	0.22866	0.18648	0.40513	0.73477
Retraining 7	8 – 2207	01.08.2000 - 01.15.2006	0.11574	0.31291	0.37031	0.27792	0.5245	0.66889
Retraining 8	9 – 2208	01.09.2000 - 01.16.2006	0.36433	0.46133	0.46925	0.36557	0.36487	1.328
Retraining 9	10 – 2209	01.10.2000 - 01.17.2006	0.18493	0.19075	0.15858	0.21023	0.75129	1.0524
Retraining 10	11 – 2210	01.11.2000 - 01.18.2006	0.14805	0.1671	0.13561	0.23375	0.97112	0.95238
Retraining 11	12 – 2211	01.12.2000 - 01.19.2006	0.074407	0.090585	0.1441	0.2901	0.73957	0.88631
Retraining 12	13 – 2212	01.13.2000 - 01.20.2006	0.076778	0.089306	0.22326	0.38945	0.34145	0.79346
Retraining 13	14 – 2213	01.14.2000 - 01.21.2006	0.044689	0.12977	0.25736	0.39816	0.38538	0.37487
Retraining 14	15 – 2214	01.15.2000 - 01.22.2006	0.2056	0.37461	0.58472	0.70546	0.49651	0.44108
Retraining 15	16 – 2215	01.16.2000 - 01.23.2006	0.36572	0.52136	0.67204	0.81206	0.70054	0.35835
Retraining 16	17 – 2216	01.17.2000 - 01.24.2006	0.23738	0.42856	0.5438	0.60861	0.56222	0.49794
Retraining 17	18 – 2217	01.18.2000 - 01.25.2006	0.38928	0.41171	0.48634	0.39433	0.36041	0.2812
Retraining 18	19 – 2218	01.19.2000 - 01.26.2006	0.2667	0.34083	0.45804	0.33101	0.24459	0.17784
Retraining 19	20 – 2219	01.20.2000 - 01.27.2006	0.063721	0.12594	0.17982	0.83789	1.4244	0.55356
Retraining 20	21 – 2220	01.21.2000 - 01.28.2006	0.016919	0.1221	0.22721	0.71081	0.99439	0.24403
Retraining 21	22 – 2221	01.22.2000 - 01.29.2006	0.25454	0.27707	0.35196	0.56387	0.61286	0.95913
Retraining 22	23 – 2222	01.23.2000 - 01.30.2006	0.19422	0.23567	0.2527	0.26508	0.4137	1.1581
Retraining 23	24 – 2223	01.24.2000 - 01.31.2006	0.093558	0.20829	0.73109	1.0858	1.1786	1.2231
Retraining 24	25 – 2224	01.25.2000 - 02.01.2006	0.031368	0.3043	0.65436	0.83866	0.83426	1.0103
Retraining 25	26 – 2225	01.26.2000 - 02.02.2006	0.3603	0.65889	0.90449	1.0292	0.97643	1.2173
Retraining 26	27 – 2226	01.27.2000 - 02.03.2006	0.42913	0.67081	0.7131	0.74739	0.64082	1.0412
Retraining 27	28 – 2227	01.28.2000 - 02.04.2006	0.10476	0.0897	0.23975	0.34532	0.61015	0.89879
Retraining 28	29 – 2228	01.29.2000 - 02.05.2006	0.0005274	0.074813	0.21271	0.36104	0.37248	0.95359
Retraining 29	30 – 2229	01.30.2000 - 02.06.2006	0.032522	0.15216	0.24535	0.27055	0.26649	1.1093

	Training / retraining interval	Time period for training	ERR (Case I)					
			33 inputs					36 inputs
			T=1	T=3	T=7	T=15	T=30	T=30
Retraining 30	31 – 2230	01.31.2000 - 02.07.2006	0.10169	0.25625	0.2563	0.20624	0.46375	1.3185
Retraining 31	32 – 2231	02.01.2000 - 02.08.2006	0.24534	0.37058	0.28295	0.22135	0.52646	0.59664
Retraining 32	33 – 2232	02.02.2000 - 02.09.2006	0.27922	0.28109	0.25888	0.18676	0.29903	0.53754
Retraining 33	34 – 2233	02.03.2000 - 02.10.2006	0.16562	0.17574	0.24491	0.22742	0.32142	0.2161
Retraining 34	35 – 2234	02.04.2000 - 02.11.2006	0.040922	0.32488	0.39714	0.34703	0.46982	1.3069
Retraining 35	36 – 2235	02.05.2000 - 02.12.2006	0.447	0.63564	0.59412	0.53827	0.61246	0.81452
Retraining 36	37 – 2236	02.06.2000 - 02.13.2006	0.3318	0.3983	0.26585	0.21885	0.68876	0.47123
Retraining 37	38 – 2237	02.07.2000 - 02.14.2006	0.29619	0.26533	0.18963	0.21749	0.37173	0.27451
Retraining 38	39 – 2238	02.08.2000 - 02.15.2006	0.22654	0.14381	0.12891	0.19279	0.19405	0.39522
Retraining 39	40 – 2239	02.09.2000 - 02.16.2006	0.063032	0.045212	0.06817	0.15099	0.20866	0.53497
Retraining 40	41 – 2240	02.10.2000 - 02.17.2006	0.035246	0.038618	0.067873	0.12967	0.38001	0.59025

We carried out the simulations under the following assumptions:

- *V* = 2200 days are enough for the first training phase and then for each retraining phase;
- *T* = 1 (or 3, 7, 15, 30) days represents the prediction horizon;
- *Shift* = 1 day is the shifting time for the next retraining.

It is worth mentioning that the values of the previous parameters can be easily changed. Choosing the number of samples for training is an open issue: not too small to have enough data (more than five times the number of samples versus the number of weights), but not too large, especially in a non-stationary environment.

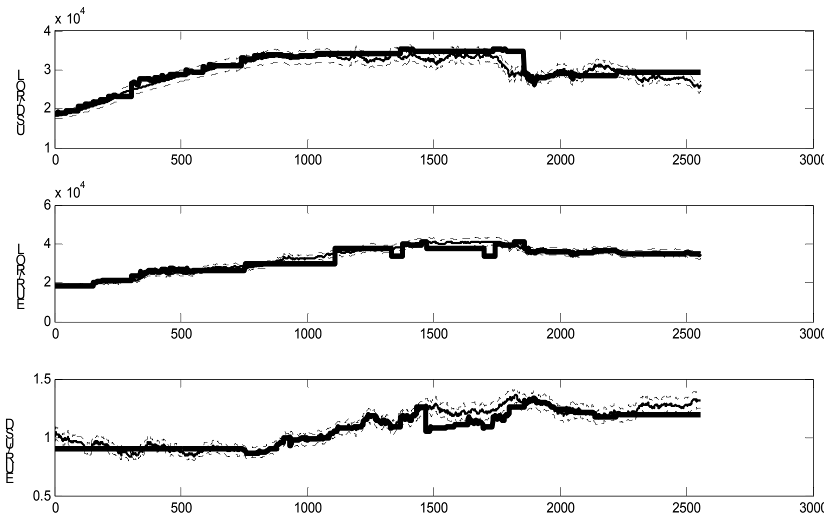
In Table 2, for the column that corresponds to T=1 we have the evolution of the ERR when the prediction horizon is only one day. This could be enough if we intend to maintain the shifting time for the next retraining at the value *Shift* = 1. In order to have an idea what happens for a while if the model is frozen (not updated) from the current retraining we enlarged the prediction horizon (*T*) to different numbers of time steps. This will be used in the estimation of the robustness of the model. We remark a decreasing trend of the ERR irrespective of the value of the prediction horizon (see Table 2, Table 3, Figure 6 and Figure 7). This is a very important fact that shows the clear advantages of the retraining technique.

The last column of Table 2 represents the results of using an extended number of inputs (36), which include three supplementary parameters that represent the "Sibiu

Futures exchange rate of one month" for EUR/ROL, USD/ROL and EUR/USD exchange rates. The error ERR was computed for the case of T=30 in order to have a comparison with the previous column that contains the results of the same model when using only 33 inputs. It is very easy to observe that the supplementary parameters did not improve the values of ERR as we initially expected (see also Figures 6 and 7). The reason for this outcome is the quality of these three supplementary parameters (see Figure 5).

Figure 5

The current exchange rate (thin line) and the "Sibiu Futures exchange rate of one month" rate (thick line)



In Figure 5, the current exchange rates are depicted by the thin line and the "Sibiu Futures exchange rate of one month" (those three supplementary inputs) are showed in thick line. We can remark three tubes (dotted lines) that border the maximum variation of 5% around the current exchange rates. The shapes of the supplementary inputs can explain now the weakness of the model that includes them.

In Table 2, it seems that the delay vectors have been properly chosen, since, finally, there has been a decrease in the test error when successive retraining phases were performed. The next table (Table 3) includes two kinds of columns for the ERR: the left one shows the evolution of the ERR when **Iterative Simulation** (IS) is employed (see the second step from the previous section), and the right one when the system "**Always uses the Real Inputs"** (ARI), which included the real previous outputs and not the estimated ones. It is quite remarkable that the Iterative Simulation provides pretty close values (usually a bit higher, but the difference is not significant) of the ERR as compared to the situation when the system is always fed with real inputs. Practically, a long-term forecasting can be implemented using only the Iterative Simulation and the second approach remains a utopia.

Table 3

Evolution of Test Error (ERR) for the Iterative Simulations of Output (IS) and Always Real Inputs (ARI) of Case I during the training and retraining phases when T=30

	Training / retraining interval	Time period for training	ERR (Case I)			
			33 inputs		36 inputs	
			IS	ARI	IS	ARI
First training	1 – 2200	01.01.2000 - 01.08.2006	4.3185	4.2391	6.8036	6.4361
Retraining 1	2 – 2201	01.02.2000 - 01.09.2006	0.5592	0.54745	1.6479	1.6208
Retraining 2	3 – 2202	01.03.2000 - 01.10.2006	0.73979	0.7293	1.7582	1.7145
Retraining 3	4 – 2203	01.04.2000 - 01.11.2006	0.9866	0.95797	3.6997	3.4594
Retraining 4	5 – 2204	01.05.2000 - 01.12.2006	0.95671	0.92525	0.99947	0.96055
Retraining 5	6 – 2205	01.06.2000 - 01.13.2006	0.44314	0.4334	0.65897	0.63753
Retraining 6	7 – 2206	01.07.2000 - 01.14.2006	0.40513	0.39207	0.73477	0.71173
Retraining 7	8 – 2207	01.08.2000 - 01.15.2006	0.5245	0.49901	0.66889	0.65108
Retraining 8	9 – 2208	01.09.2000 - 01.16.2006	0.36487	0.344	1.328	1.3033
Retraining 9	10 – 2209	01.10.2000 - 01.17.2006	0.75129	0.73434	1.0524	1.0255
Retraining 10	11 – 2210	01.11.2000 - 01.18.2006	0.97112	0.93854	0.95238	0.92397
Retraining 11	12 – 2211	01.12.2000 - 01.19.2006	0.73957	0.70687	0.88631	0.86028
Retraining 12	13 – 2212	01.13.2000 - 01.20.2006	0.34145	0.32716	0.79346	0.76972
Retraining 13	14 – 2213	01.14.2000 - 01.21.2006	0.38538	0.36437	0.37487	0.36298
Retraining 14	15 – 2214	01.15.2000 - 01.22.2006	0.49651	0.47724	0.44108	0.42954
Retraining 15	16 – 2215	01.16.2000 - 01.23.2006	0.70054	0.67179	0.35835	0.35233
Retraining 16	17 – 2216	01.17.2000 - 01.24.2006	0.56222	0.53796	0.49794	0.4807
Retraining 17	18 – 2217	01.18.2000 - 01.25.2006	0.36041	0.34827	0.2812	0.27538
Retraining 18	19 – 2218	01.19.2000 - 01.26.2006	0.24459	0.2384	0.17784	0.17644
Retraining 19	20 – 2219	01.20.2000 - 01.27.2006	1.4244	1.3659	0.55356	0.53951
Retraining 20	21 – 2220	01.21.2000 - 01.28.2006	0.99439	0.95142	0.24403	0.23829
Retraining 21	22 – 2221	01.22.2000 - 01.29.2006	0.61286	0.58908	0.95913	0.9312
Retraining 22	23 – 2222	01.23.2000 - 01.30.2006	0.4137	0.40612	1.1581	1.1155
Retraining 23	24 – 2223	01.24.2000 - 01.31.2006	1.1786	1.1329	1.2231	1.1825
Retraining 24	25 – 2224	01.25.2000 - 02.01.2006	0.83426	0.80241	1.0103	0.97381
Retraining 25	26 – 2225	01.26.2000 - 02.02.2006	0.97643	0.93282	1.2173	1.1793
Retraining 26	27 – 2226	01.27.2000 - 02.03.2006	0.64082	0.61318	1.0412	1.0118
Retraining 27	28 – 2227	01.28.2000 - 02.04.2006	0.61015	0.59145	0.89879	0.88054
Retraining 28	29 – 2228	01.29.2000 - 02.05.2006	0.37248	0.35458	0.95359	0.93104
Retraining 29	30 – 2229	01.30.2000 - 02.06.2006	0.26649	0.25757	1.1093	1.0919
Retraining 30	31 – 2230	01.31.2000 - 02.07.2006	0.46375	0.44561	1.3185	1.2903
Retraining 31	32 – 2231	02.01.2000 - 02.08.2006	0.52646	0.50453	0.59664	0.57321
Retraining 32	33 – 2232	02.02.2000 - 02.09.2006	0.29903	0.28836	0.53754	0.52085
Retraining 33	34 – 2233	02.03.2000 - 02.10.2006	0.32142	0.30645	0.2161	0.2147
Retraining 34	35 – 2234	02.04.2000 - 02.11.2006	0.46982	0.45186	1.3069	1.2289
Retraining 35	36 – 2235	02.05.2000 - 02.12.2006	0.61246	0.58427	0.81452	0.76826
Retraining 36	37 – 2236	02.06.2000 - 02.13.2006	0.68876	0.66023	0.47123	0.45008
Retraining 37	38 – 2237	02.07.2000 - 02.14.2006	0.37173	0.34761	0.27451	0.26621
Retraining 38	39 – 2238	02.08.2000 - 02.15.2006	0.19405	0.18702	0.39522	0.38304
Retraining 39	40 – 2239	02.09.2000 - 02.16.2006	0.20866	0.20311	0.53497	0.51477
Retraining 40	41 – 2240	02.10.2000 - 02.17.2006	0.38001	0.35892	0.59025	0.56768

An example of the error trends for Case I (33 inputs) is showed in Figure 6, which includes two graphs of the ERR: the left one shows the evolution of the ERR when **Iterative Simulations** (IS) are performed, and the right one when the system "**Always uses the Real Inputs"** (ARI). One may note that the abscissa represents the numbers of the successive retraining phases and the first value 0 is associated with the first training.

Figure 6

ERR trend (Case I – 33 inputs) of test sets for the first training and *L* = 40 successive retraining phases

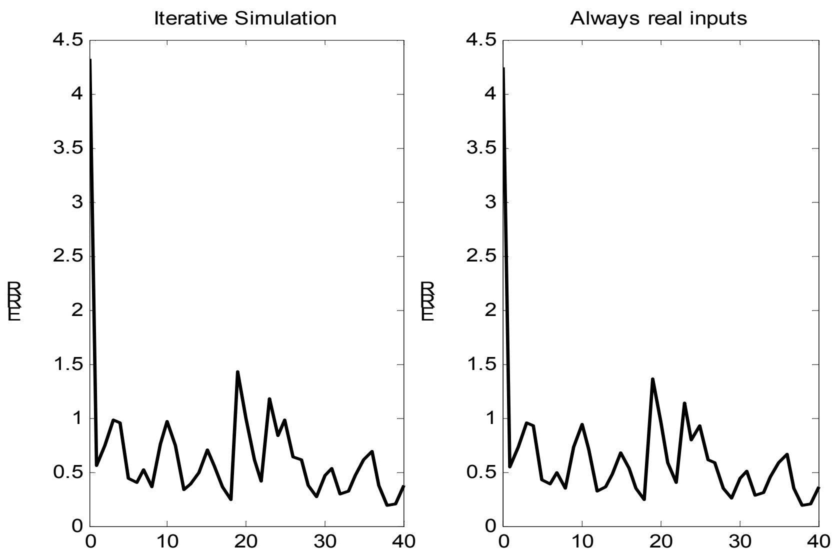


Figure 7 is somewhat similar with the previous one but it results from the model with 36 inputs.

The quality of the predictions can also be graphically analyzed, by enforcing a tube around the real outputs, given by a function like the one below:

$$f(n) = A + n \cdot q \tag{9}$$

Here, *A* is an acceptable prediction error, *q* is an increasing factor and *n* is the number of predicted time steps. The predicted output values should fall within the interval output(*n*) +/- *f*(*n*), represented by the dotted lines in Figures 8 – 10.

Figure 8 shows the graphs of the EUR/ROL exchange rate (Case II) for the last test interval (of retraining 20). The real data are represented with thin (blue) lines and the neural network output values with thick (red) lines. There are two graphs in the same figure: the first shows the evolution of the output when Iterative Simulation (IS) is employed and the second one when the system *Always* uses *Real Inputs* (ARI). There is a "tube" (dotted lines) around the real data, given by the function *f*(*n*)=300+0.05·*n* (where *n* = 1 … 40). The abscissa shows the number of days in the years when predictions are performed.

Figure 7

ERR trend (Case I – 36 inputs) of test sets for the first training and *L* = 40 successive retraining phases

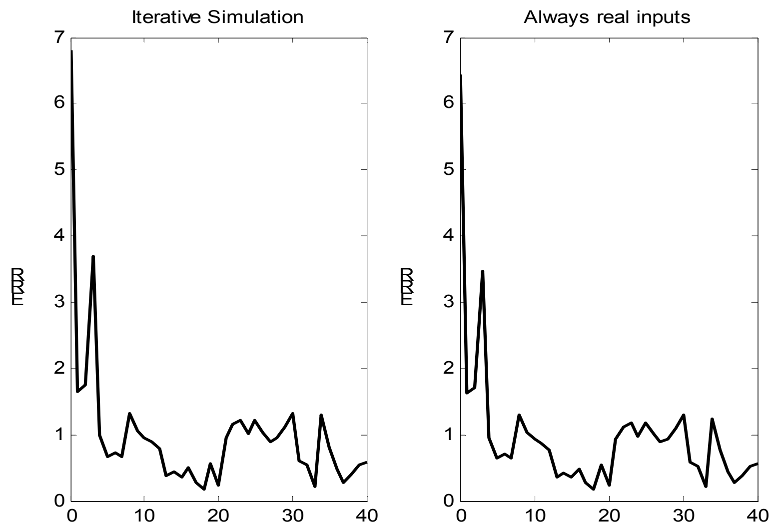
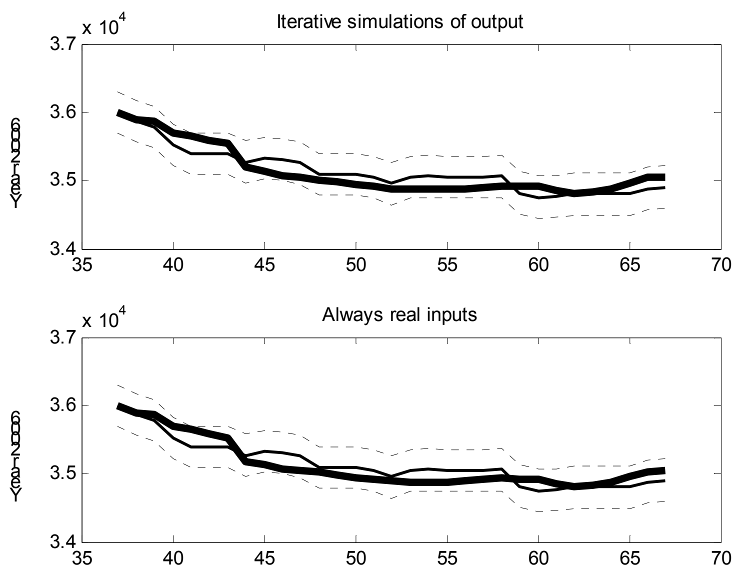


Figure 8

Data forecasting for the test interval of retraining 29 (Case I – 33 inputs)



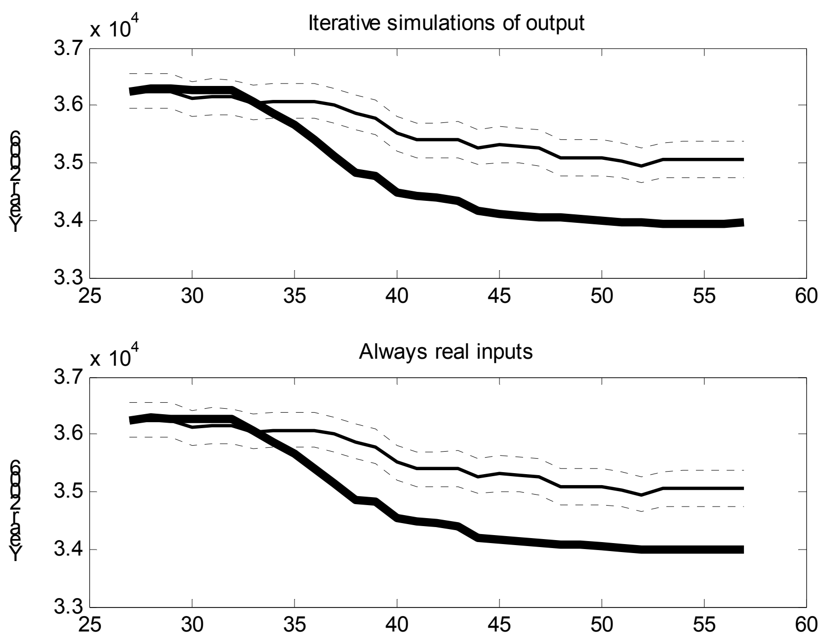
ERR_{IS} = 0.26649 and ERR_{ARI} = 0.25757.

One may note that the graphs are extended to the left with one more value that corresponds to the last value of the training/validation interval. In this way, both lines (thin and thick) start from approximately the same point. The difference between **ERR_{IS} = 0.26649 and ERR_{ARI} = 0.25757** is not significant. These values were computed by using the relation 8 for all the data represented in Figure 8. It is noticeable that there are not visible differences of these two graphs. It seems that the model is robust and works very well with the iterative simulations, as well as on the ARI way. We obtained similar results in all situations without any exception.

A natural question is: "***What is the matter without retraining***?" In order to have a clear answer to this question we have to study Table 2, more specifically the evolutions of the ERR for different values of the prediction horizon T: one day, three days, one week, half a month and a month. All these evolutions show a relative similar decreasing trend. It is obviously that for T=30 we obtained the highest values for the evolution of ERR during all 40 successive retraining phases. However, if we use the associated graphs (like the ones in Figure 8) for each value of the ERR, then we can easily remark when the forecasted values turn on the wrong way (see Figure 9). In this way, we can estimate the right shifting time (*Shift*) for the next retraining. For the case in Figure 9, the highest acceptable value is Shift=5. When the retraining is performed the ERR comes back to the normal range since the model is accommodated to the new changing input-output function.

Figure 9

Data forecasting for the test interval of retraining 19 (Case I – 33 inputs)



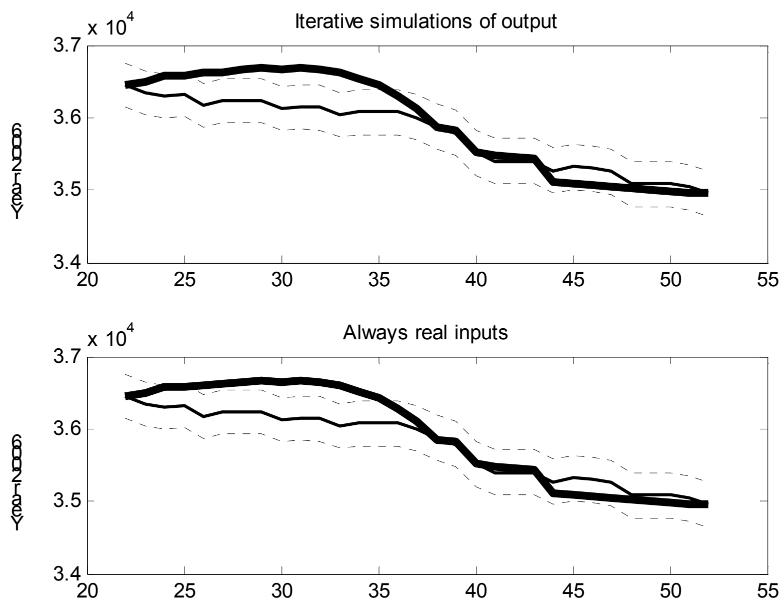
ERR_{IS} = 1.4244 and ERR_{ARI} = 1.3659

Table 2 and associated figures of ERR reveal the high importance of the retraining phase as an upgrading/updating process and the effect of ignoring it, as well.

We noticed that, except for a few cases, in almost all the graph representations of the predictions the trends were well captured (even outside the "tube") by using our approach. Moreover, we showed both graphs (with Iterative Simulation and when the system always uses the real inputs) in order to demonstrate the robustness of the model. The Iterative Simulation does not increase the error as much as one could expect at the first sight. The long-term prediction is not very accurate as long as after a while the simulated outputs evidently exceed the limits of the "tube" around the real outputs. Nevertheless, we remark sometime interesting evolutions of the predictions (Figure 10) that return, after a while, to the desired range.

Figure 10

Data forecasting for the test interval of retraining 14 (Case I – 33 inputs)



ERR_{IS} = 0.49651 and ERR_{ARI} = 0.47724.

5. Conclusions

The ANNs ability to extract significant information from its training data provides a valuable framework for the representation of relationships that are present in the structure of the data. This allows for both the interpolation among the *a priori* defined points and the extrapolation outside the range bordered by the extreme points of the training set.

The evaluation of the test error shows that the adaptive retraining technique can gradually improve, on the average, the achieved results. Our practical experience reveals that the first training (which includes the searching phase for the optimal architecture) always takes a relatively long time, but then the system can be very easily retrained, as there are no changes in the structure. The great advantage of the retraining technique is that some relevant aspects are preserved (*remembered*) not only from the immediate previous training phase, but also from the last but one phase, and so on. A kind of *slow forgetting process* also occurs; thus it is much easier for the ANN to remember specific aspects of the previous training instead of the first training. It means that the former information accumulated during the previous trainings will be slowly forgotten and the learning process will be adapted to the newest evolutions of the financial process.

In the presented applications, the optimum shifting time for the next retraining is one day. This way the model can be quickly updated using the retraining procedure. Nevertheless, the graphs of the predictions show that the system can still provide correct values without retraining for several days but there is a major risk of loosing the unexpected changes in the financial environment.

We remark that some supplementary parameters (like Sibiu Futures exchange rate of one month) did not improve the values of the ERR as we initially expected. The reason for this outcome is very probably the inconsistent quality of these three supplementary indicators that exceed sometimes the limits of an acceptable precision, and the real market did not take into account their influence. Our next approach will involve the one-month prediction in order to have a direct comparison with the Sibiu Futures Stock Exchange market. The results of forecasting the exchange rate for one day suggest that this technique could be extended to a large period of forecasting (a week, a month, 3 months, 6 months or more) without difficulties, and we intend to present these simulations in another paper.

The final remark refers to the basic training algorithm. Even if the SCG algorithm is not the fastest algorithm, the great advantage is that this technique works very efficient for networks with a large number of weights. The SCG is something of a compromise; it does not require large computational memory, and yet it still has a good convergence and is very robust. Furthermore, we always apply the early stopping method (validation stop) during the training process, in order to avoid the over-fitting phenomenon. In addition, it is well known that for the early stopping one must be careful not to use an algorithm that converges too rapidly. The SCG is properly suited for the validation stop method. Nevertheless, it is quite easy to replace the SCG algorithm with another one, since the adaptive retraining technique is flexible and independent of the basic training algorithm.

Acknowledgements

The CEEX Research Project No 76/26.07.2006 has partially supported this work. We gratefully acknowledge the assistance of the Institute of Economic Forecasting, led by Prof. Lucian-Liviu Albu, and the financial support from the Romanian Commercial Bank. The authors would also like to thank Prof. Dan Popescu and Radu Balbaie for their help to obtain the Sibiu Futures data. And finally, we really appreciate the young Ph.D. Students Petre Caraiani and Andrei Dospinescu for their effort to build the initial data base.

References

Abel, A.B., and Bernanke B.S., (2001), *Macroeconomics*, Fourth Edition, Addison Wesley.

Adams, P.D., and Dixon P.D., (1989), "Forecasts for the Australian Economy in 1989-90 and 1990-91", *The Australian Economic Review*, 4th Quarter, pp. 6-7 (http://www.unibw-hamburg.de/WWEB/math/uebe/zuhause.engl.html, Model Australia 87/1).

Anderson, J.B., (1990), *Economic Policy Alternatives for the Latin American Crisis*, Taylor&Francis, New York, pp. 46-47 (http://www.hsu-hh.de/uebe/index_pxYttPvomemOS8pr.html, Model Ecuador 90/1).

Basheer, I.A., and Hajmeer, M., (2000), "Artificial neural networks: fundamentals, computing, design, and application", *Journal of Microbiological Methods*, Elsevier Science, Vol. 43, pp. 3-31.

Bergstrom, A.R., Nowman K.B., and Wandasiewicz S., (1994), "Monetary and Fiscal Policy in a Second-Order Continuous Time Macroeconometric Model of the United Kingdom", *Journal of Economic Dynamics and Control* 18, 3-4, pp.731-761 (http://econpapers.repec.org/article/eeedyncon/default18.htm).

Bordo M. D., (2003), *Exchange Rate Regime Choice in Historical Perspective*, IMF, Working Paper, 03/160, International Monetary Fund, Washington DC.

de Bondt, G.J., van Els P.J.A., and Stokman A.C.J., (1997), *EUROMON, A Macreoeconometric Multi-Country Model for the EU*, De Nederlandsche Bank NV, 40-51 (http://www.hsu-hh.de/uebe/index_biygHK6MujxyExXh.html, Model 96/13 of the Netherlands and of the EU countries).

Dickey, D. and Pantula, S., (1987), "Determining the Order of Differencing in Autoregressive Process", *Journal of Business and Economic Statistics* 5, pp. 455 - 461.

Dickey, D., Hasza D.P., and Fuller W.A., (1984), "Testing for unit roots in seasonal time series", *Journal of the American Statistical Association* 79, pp. 355-367.

Dobrescu, E., (2006), *Macromodels of the Romanian Market Economy*, Editura Economică, Bucureşti.

Dornbush, R., (1994), *Exchange Rate and Inflation*, The MIT Press, Cambridge, Massachusetts.

Fair, R.C., (1994), "Testing Macroeconometric Models", Cambridge, Mass., London, pp. 356-376, 396-401 (http://www.unibw-hamburg.de/WWEB/math/uebe/zuhause.engl.html, Model USA 75/5, Model World 84/20).

Granger, C.W.J., (1969), "Investigating Causal Relations by Econometric Models", *Econometrica*, 37, pp. 424-438.

Granger C.W.J., (1988), "Some Recent Developments in a Concept of Causality", *Journal of Econometrics*, 39, pp. 199-211.

Hagan, M.T., Demuth, H.B., and Beale, M., (1996), *Neural Networks Design*, MA: PWS Publishing, Boston.

Hornik, K., Stinchcombe, M., and White, H., (1989), “Multilayer feedforward networks are universal approximators,” *Neural Networks*, vol.2, pp. 359-366.

Huang, J. Q., and Lewis, F. L., (2003) “Neural-network predictive control for nonlinear dynamic systems with time-delay”, *IEEE Trans. Neural Networks*, vol. 14, pp. 377-389.

Jackson, J.E., (1991), *A user guide to principal components*, John Wiley, New York.

Jahnke, W. *et al*., (2000), *Macro-Econometric Multi-Country Model: MEMMOD*, Deutsche Bundesbank, Frankfurt am Main, Germany, June, pp.146-273.

(http://www.hsu-hh.de/uebe/index_9B2IAO2icwCqr3Vs.html, Model Japan 00/1).

Krugman, P.R. and Obstfeld M., (2000), *International Economics – Theory and Policy*, Fifth Edition, Addison-Wesley Longman.

Matthews, K., (1985), “Forecasting With a Rational Expectations Model of the UK”, *Oxford Bulletin of Economics and Statistics*, Vol. 47, No. 4, 11/85, [http://www.unibw-hamburg.de/WWEB/math/uebe/zuhause.engl.html, Model United Kingdom 80-9].

Mishkin, F.S., (2001), *The Economics of Money, Banking, and Financial Markets*, Sixth Edition, Addison Wesley Longman.

Moller, M.F., “A scaled conjugate gradient algorithm for fast supervised learning,” *Neural Networks*, vol. 6, pp. 525-533, 1993.

Mussa M., (1986), “Nominal Exchange Rate Regimes and the Behavior of Real Exchange Rates, Evidence and Implications”, *Carnegie-Rochester Conference Series on Public Policy*, Vol. 25, pp. 117-213.

Năstac, I., (2004) “An Adaptive Retraining Technique to Predict the Critical Process Variables”, *TUCS Technical Report*, No. 616, June 2004, Turku, Finland

(http://www.tucs.fi/research/series/serie.php?type=techreport&year=2004).

Năstac, I., and Matei, R., (2003) “Fast retraining of artificial neural networks,” in *Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, Wang *et al*. (Eds.), Springer-Verlag in the series of *Lecture Notes in Artificial Intelligence* (LNAI 2639), pp. 458-462.

Neu, C.R., (1990), “Medium-Term Prospects for the Mexican Economy: Some Modeling Results”, RAND, N-3035-USDP, July

(http://www.hsu-hh.de/uebe/index_CvbvYnW41M0w0Seo.html, Model structure of MEXMOD 90/1).

Pelinescu, E., (2003), “FEER: Relevant theoretical and empirical literature“, *Romanian Journal of Economic Forecasting*, Spec. Edition 1., The Expert Publishing House, Bucharest, pp. 111-118.

Salzano, M., (1999), “Neural Networks as tools for increasing the forecast and control of complex economic systems”, *Economics & Complexity*, Vol. 2, No. 2.

Shadbolt, J. (Ed.), and Taylor, J.G., (Eds.) (2002), *Neural Networks and the Financial Markets*, Springer-Verlag Telos.

Weyerstrass, K., (2000), *SLOPOL1 – A Macroeconomic Model for Slovenia*, University of Klagenfurt, Austria, Department of Economics

(http://www.unibw-hamburg.de/WWEB/math/uebe/zuhause.engl.html, Model Slovenia 2000/1).

Zhang, G., Patuwo, B.E., and Hu, M.Y., (1998), "Forecasting with artificial neural networks: The state of the art," *Int. J. Forecasting*, vol. 14, pp. 35-62.

Zhang, G.P., (2003), "Neural Networks in Business Forecasting", *Information Science Publishing*.