Comparing Neural Networks and ARMA Models in Artificial Stock Market [*]

Jiří Krtek

Academy of Sciences of the Czech Republic, Institute of Information Theory and Automation.

e-mail: krtek@karlin.mff.cuni.cz

Miloslav Vošvrda [†]

Academy of Sciences of the Czech Republic, Institute of Information Theory and Automation.

e-mail: vosvrda@utia.cas.cz

Abstract. We create a new way of comparing models for forecasting stock prices. Our idea was to create a simple game in which the individual models, which we want to compare, would compete against each other. Therefore, we were inspired by the heterogeneous agent models and we created an artificial market. Models, which we want to compare, act in our artificial market as a forecasting strategies of each agent who trades on the market. There are traded one risky asset paying a dividend and one risk-free asset in our artificial market. Individual agents decide whether to buy or sell the risky asset on the basis of their expectations about future excess return of the risky asset. Their expectations are driven by the models we want to compare. Each agent uses his own model for predicting future prices of risky assets and dividends. Delayed prices of risky asset and dividends provided the basis for predictions. The way how agents trade (bought or sold risky asset) affects the price of risky asset, which in turn influences their expectations and therefore their decisions whether to buy or sell. Moreover, each agent can recalculate his strategy (the parameters of his forecasting model), if he is not satisfied with its performance. So the forecasting strategies and the artificial market evolve side by side.
The models we confront are neural networks – feed-forward neural networks and Elman's simple recurrent neural networks – and vector ARMA models, namely, VAR and VARMA. It remains only to add that the winning model is the one which earns the most money.

Keywords: Neural networks, vector ARMA, artificial market.

JEL classification: C23, C45, C53, G17
AMS classification: 91G80

[*]The research was supported by grant of GA ČR 402/09/H045 and grant SVV 305–09/261315
[†]Supervisor

1. Introduction

In this paper we compare vector ARMA models as classic delegates of linear models and neural networks as delegates of non-linear models: specifically vector AR and ARMA models with various lags of variables and errors with feed-forward,as well as Elman's simple recurrent neural networks with various inputs – lags of variables, and eventually lagged outputs of neurons from the hidden layer (in Elman's networks).

These forecasting models are not compared using real data as it is customary to do. Accuracy of forecasts or their standard deviations are not calculated. We tried to use an unconventional way of comparing forecasting strategies, in order to create a new method.

The models are compared in an artificial stock market with one risk-free asset and one risky stock. Traders, or agents, in the market use the aforementioned models as forecasting strategies. The best model among them is simply the one that earns the most money.

It is also important to create an artificial stock market that has similar characteristics to the real market. The characteristics of the artificial market are therefore also studied and the artificial market is built to be conformable to the real one.

2. Artificial Stock Market

The model which simulates market environment was inspired mostly by [3], [12], [6] and [5].

Two assets are traded in the market. The first is a risk-free asset which is perfectly elastically supplied and has a constant rate of return r. The second one is a risky asset whose price at time t is denoted by p_t. The risky stock can be divided into endlessly small pieces and pays dividend d_t at time t; the dividend is chosen (as in [6]) to follow the stationary AR process

$$d_{t+1} = \bar{d} + \rho(d_t - \bar{d}) + \varepsilon_{d,t+1}, \tag{1}$$

where $\rho = 0.95$ is chosen this way to provide a persistent process which is still stationary, $\bar{d}$ is a chosen constant and $\varepsilon_{d,t} \sim N(0, \sigma_d)$.

There are four groups of traders. The first one is made up of traders who use a vector AR (=VAR) model to predict the future dividend and price change of the risky asset. The second group consists of traders whose forecasting strategy is vector ARMA (=VARMA) model. The third group is comprised of traders whose forecasting strategy uses a feed-forward neural network (=FNN) with three layers having perceptrons as hidden units. Inputs are lagged values of dividends and price changes. The last group uses Elman's simple recurrent neural network (=SRN) to forecast the dividend and price change. Inputs are again lagged dividends and price changes. Number of lags is chosen randomly from the set $\{1, 2, 3\}$ in all cases.

As in [3] and many other heterogeneous-agent models, it is assumed that all traders maximise their expected utility function. It is also assumed that all traders have the same constant absolute risk aversion (=CARA) utility function U, with the same risk aversion parameter λ. Let $W_{i,t}$ denote the wealth of trader i at time t. Furthermore, let $h_{i,t}$ be the number of agent i's shares at time t, then trader's goal is to maximise the expected utility at time $t+1$ given information up to time t over his number of shares, i.e.,

$$\max_{h_i,t} E[U(W_{i,t+1})|I_t] = E_{i,t}[U(W_{i,t+1})], \tag{2}$$

where I_t denotes the information set available at time t.

Under the assumption of exponential CARA utility function and the Gaussian distribution of forecasts, the optimal number of shares at time $t+1$ is the ratio of the expected excess return with respect to the conditional variance of future price and dividends, i.e.,

$$h^*_{i,t+1} = \frac{E_{i,t}[p_{t+1} + d_{t+1}] - (1+r)p_t}{\lambda\sigma^2_{i,t}}, \tag{3}$$

where $\sigma^2_{i,t}$ is the conditional variance of $p_{t+1} + d_{t+1}$ given I_t. Traders differ only in their forecasting strategies, i.e., the way they form their expectation about $p_{t+1}+d_{t+1}$ at time t, i.e., in $E_{i,t}[p_{t+1} + d_{t+1}]$.

2.1. Price Evolution

The model of price evolution is borrowed from [12] and [3].

According to the previous section, $h^*_{i,t}$ will be used for the desired number of risky shares at time t, while $h_{i,t}$ will denote the actual number of shares held.

Let us introduce the following notation. Let $b_{i,t}$, be the number of shares that trader i would like to buy and let $a_{i,t}$ be the number of shares which he would like to sell, i.e.,

$$b_{i,t} = \begin{cases} h^*_{i,t} - h_{i,t-1}, & h^*_{i,t} \geq h_{i,t-1}, \\ 0, & \text{otherwise}. \end{cases} \tag{4}$$

$$a_{i,t} = \begin{cases} h_{i,t-1} - h^*_{i,t}, & h^*_{i,t} < h_{i,t-1}, \\ 0, & \text{otherwise}. \end{cases} \tag{5}$$

Price change of the risky asset is based on the current bid-ask spread. Let N be the total number of traders. If $B_t = \sum_{i=1}^N b_{i,t}$ denotes the overall demand for the risky asset and $A_t = \sum_{i=1}^N a_{i,t}$ the overall supply of the risky asset, then the number of shares held by trader i at time t is given by the following rule:

$$h_{i,t} = \begin{cases} h_{i,t-1} + b_{i,t} - a_{i,t}, & B_t = A_t, \\ h_{i,t-1} + \frac{A_t}{B_t} b_{i,t} - a_{i,t}, & B_t > A_t, \\ h_{i,t-1} + b_{i,t} - \frac{B_t}{A_t} a_{i,t}, & B_t < A_t. \end{cases} \tag{6}$$

The price adjustment is based on the excess demand $B_t - A_t$ and is given by

$$p_{t+1} = p_t(1 + \beta(B_t - A_t)) + \varepsilon_{r,t}, \tag{7}$$

where $\varepsilon_{r,t}$ is the noise (i.i.d. random variables from $N(0, \sigma_r^2)$) added in order to represent other traders potentially present in the market, and β is some function of excess demand. Following [3], β was set to

$$\beta(B_t - A_t) = \begin{cases} \tanh(\beta_1(B_t - A_t)), & B_t \geq A_t, \\ \tanh(\beta_2(B_t - A_t)), & B_t < A_t. \end{cases} \tag{8}$$

2.2. Forecasting Strategies

Traders want to forecast future change of the price and future dividends. As was previously stated, this work compares neural networks with ARMA models. Traders therefore use neural networks or some vector ARMA model as their forecasting strategy. This is the only thing that differentiates one trader from another.

The first group of agents uses VAR(p) model as their forecasting strategy. Each trader chooses randomly his lag p from set $\{1, 2, 3\}$ at the beginning of the simulation, then he uses conditional maximum likelihood for searching for optimal parameters. The conditional maximum likelihood method was implemented in accordance with [11].

Traders of the second type forecast future values with the aid of VARMA(p,q) model. Lags p and q are again chosen randomly from set $\{1, 2, 3\}$ at the beginning of the simulation. Conditional maximum likelihood is used again. The conditional maximum likelihood method was implemented in accordance with [2], [8] and [7].

There were also two groups whose forecasting strategies were based on neural networks. The first of them used a feed-forward neural network with 3 layers.

Number of lags of price changes and dividends n_l which served as inputs were taken randomly from $\{1, 2, 3\}$. So we had $2n_l$ inputs – n_l for lags of price changes and n_l for lags of dividends – and inputs $\{p_t - p_{t-1}, p_{t-1} - p_{t-2}, \ldots, p_{t-n_l+1} - p_{t-n_l}\}$ and $\{d_t, d_{t-1}, \ldots, d_{t-n_l+1}\}$ were used to generate the output $\{\hat{p}_{t+1} - p_t, \hat{d}_{t+1}\}$ (the estimate of $\{p_{t+1} - p_t, d_{t+1}\}$).

Its hidden layer consists of perceptrons with sigmoid activation function. The number of perceptrons in the hidden layer was chosen randomly from set $\{1, 2, \ldots, \text{number of inputs}\}$. The scheme of topology of the feed-forward network that we used can be seen in Figure 1.

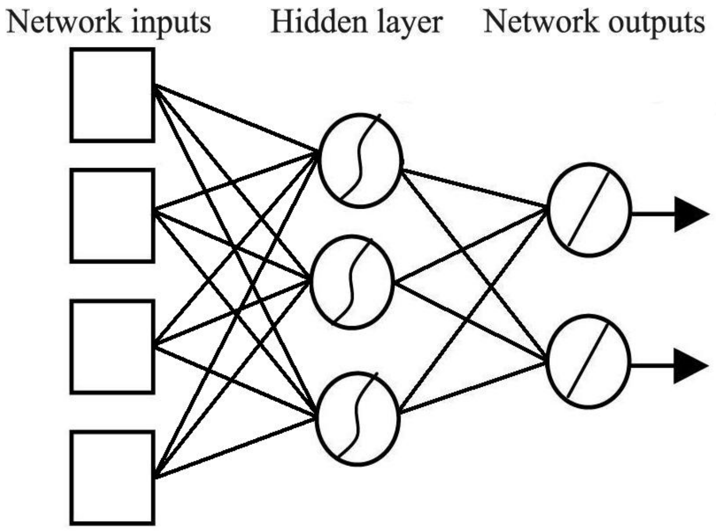

Figure 1: Topology of feed-forward neural network.

The second group employs Elman's simple recurrent neural network with 3 layers. Numbers of lags of price changes and dividends were again taken randomly from

$\{1, 2, 3\}$ and the hidden layer consists again of perceptrons with a sigmoid activation function. The number of hidden neurons in the hidden layer was again chosen randomly from set $\{1, 2, \ldots, 2*\text{number of lags}\}$. So the construction of the topology is analogical to that of a feed-forward neural network except that in this case the topology includes backward loops – the output of each hidden unit is sent back to form a new input vector with the lagged price changes and dividends. The scheme of the Elman's network topology that we used can be seen in Figure 2.

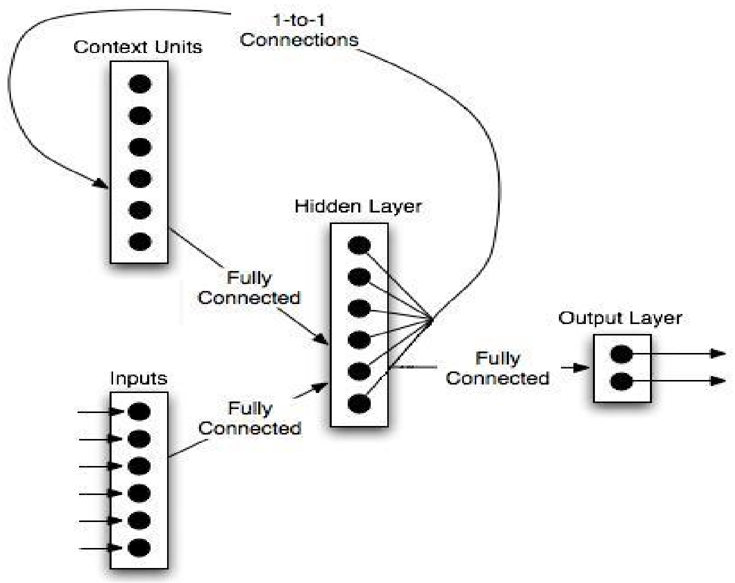

Figure 2: Topology of Elman's simple recurrent neural network.

Backpropagation, respective backpropagation through time, and genetic algorithms were used for learning FNN and respective SRN. Genetic algorithms were used only for optimisation of the weight matrix of the network. All included learning algorithms had access to the correct inputs, so it was an instance of "learning with the teacher" as can be seen in Section 2.3.

Networks were not learned each time a new input–output pair was discovered. Rather, they were learned after a bigger set of input–output pairs was known, as can be seen in Section 2.3.

The maximum number of iterations, respective generations in the case of genetic algorithms, was set to 1000. We also stop the learning process when the decrease of the error function was less than 0.001.

The individuals in the genetic algorithms case were represented as vectors of real numbers where each number represented one weight of the network. The population size was set to 50, probability of mutation to 0.04, new generation always contained the best found individual and the rest of it was formed by crossover and mutation of the best 5 individuals from the previous generation.

For more information about feed-forward neural networks and Elman's networks see [10], [1] or [9].

In the formula (3), with which traders form their desired holding of shares, the conditional variance of $p_{t+1} + d_{t+1}$, i.e., the term $\sigma^2_{i,t}$ is also used. Following [3] we let all traders estimate this conditional variance by

$$\sigma^2_{i,t} = (1-\theta)\sigma^2_{t-1|n} + \theta(p_t + d_t - E_{i,t-1}[p_t + d_t])^2, \tag{9}$$

where

$$\sigma^2_{t|n} = \frac{\sum_{j=0}^{n-1}[P_{t-j} - \bar{P}_{t|n}]^2}{n-1} \tag{10}$$

with

$$\bar{P}_{t|n} = \frac{\sum_{j=0}^{n-1} P_{t-j}}{n}. \tag{11}$$

2.3. Learning of Traders

The learning was taken from [3],or more accurately a part of trader learning from [3] was adopted and part was left out. No business school is in our artificial market, so there is no social interaction, which means that every agent has his own forecasting strategy and his own learning algorithm and he does not tell any other agent about them. He keeps his forecasting strategy secret. The big companies in the real market also keeps their forecasting models secret, so this feature of the artificial market is not in contrast with reality.

Each trader was assigned an amount of money and number of risky shares at the beginning of the simulation. This forms his initial wealth. Every trader knows his wealth as well as the wealth of all other agents in this artificial market. After some period of time, say k, he counts the difference between his present wealth and wealth at time $t-k$, i.e., $W_{i,t} - W_{i,t-k}$. This indicates how much money he has earned or lost. He can also count how much money other agents have earned or lost; thereby, he knows how good he is in comparison to other agents. Everyone consequently gets a rank $R_{i,t}$. This makes some kind of social pressure on trader i. Number

$$r_{i,t} = \frac{R_{i,t}}{N} \tag{12}$$

is then the probability that agent i recounts parameters of his forecasting strategy due to *pressure of society.* The smaller his rank the smaller the probability that he learns new parameters of his strategy.

Each trader can also recounts his strategy on the basis of growth rate of his wealth over the previous period.

Therefore, even if trader i does not recount his strategy because of *pressure of society*, there is still chance that he will do it for a different reason. Let

$$\chi^k_{i,t} = \frac{W_{i,t} - W_{i,t-k}}{|W_{i,t-k}|} \tag{13}$$

be trader i's growth rate of wealth over the previous period. It in fact measures how effective trader i's strategy was. Then

$$s_{i,t} = \frac{1}{1 + \exp\{\chi^k_{i,t}\}} \tag{14}$$

is the probability that trader i will recount his strategy because of its low efficiency.

So the final probability that trader i recounts his strategy is

$$u_{i,t} = r_{i,t} + (1 - r_{i,t})s_{i,t} = \frac{R_{i,t}}{N} + \frac{N - R_{i,t}}{N} \frac{1}{1 + \exp\{\chi_{i,t}^k\}}. \tag{15}$$

If trader i recounts parameters of his strategy at the end of the tth day, he will learn on the data of length m, where m is a constant common to all traders.

3. Simulation

The simulation had 10000 time steps and its main goal was to show whether some group of traders outperforms others. But it was also studied whether the artificial market behaves as the real one, i.e., whether stylised facts – returns have a non-Gaussian distribution, prices follow random walk – are present in our artificial market.

Several simulations with different setups were performed. Traders always learn at the beginning of the simulation on simulated data of length 100. The dividend process was generated according to (1). Values of price were taken from Gaussian distribution $N(\mu_p, \sigma_p^2)$. Moreover, the last values of dividend and price served as initial values to whole simulation. Setup of the simulation can be seen in Table 1.

We were inspired by [5] and, in order to make our artificial market more realistic, we distributed the initial money and shares in some simulations according to Zipf's law in the following way: 20 % of the traders possessed 80 % of the initial total wealth, the remaining 80 % of the traders then possessed only 20 % of initial total wealth.

3.1. Verifying Stylised Facts

As we can see in the tables below, almost all stylised facts holds in the artificial market. The price series are depicted in Figures 3 and 4.

The first stylised fact is that distribution of returns $r_t = \log(p_t) - \log(p_{t-1})$ is non-Gaussian. The Jeraque–Bera test was used to test it, see Table 2.

As you can see, the null hypothesis about Gaussian distribution of returns cannot be rejected in most periods, so this stylised fact does not hold.

The Dickey–Fuller test was used to test the null about price series having unit root. It rejected null only in the first period in the case without Zipf's law (see Table 3). This may simply be due to the fact that it was the initial period and the market did not perform well yet. If we tested the period 400–2000, DF–test will not reject the null (p–value = 0.0786).

In the case with Zipf's law, DF–test rejected the null only in the second period (see Table 3). This cannot be justified by any logic, but we came to the conclusion that this stylised fact held.

If we confirm that returns are i.i.d. random variables, the last stylised fact will also be verified. The Brock-Dechert-Scheinkman test was chosen for this. The results of the test were not sensitive to the distance parameter. The distance parameter

Market parameters	
Initial amount of money	100
Total amount of shares	30
interest rate r	0.001
price adjustment function	tanh
price adjustment $\beta 1$	2×10^{-6}
price adjustment $\beta 2$	2×10^{-6}
Price process	
Initial mean μ_p	150
Initial variance σ_p^2	0.16
noise parameter σ_r^2	0.04
Dividend process	
ρ	0.95
$\bar{d}$	0.2
σ_d	0.02
Learning parameters	
k	150
m	180
Traders	
Number of traders in each group	15
Total number of traders	60
Risk aversion parameter λ	3
parameter θ	0.01333
parameter n	10

Table 1: Setup of the modified simulation.

period	Jeraque–Bera	p–value	Jeraque–Bera, Zipf	p-value, Zipf
1–2000	0.3355	0.846	1.4796	0.477
2001–4000	1.3545	0.508	8.8353	0.012
4001–6000	8.896	0.012	2.7963	0.247
6001–8000	1.8851	0.390	0.9549	0.620
8001–9999	0.2053	0.902	3.2191	0.200

Table 2: Gaussian distribution testing of return series. The "Zipf" attribute means that the initial money and shares were distributed according to Zipf's law.

was chosen to be $\epsilon = \text{st.dev.}$. The parameter of embedding dimensions was chosen to be $m = 5$. The results can be seen in Table 4.

As we can see there is only one rejection, so we can again say, that this stylised fact holds in our artificial market.

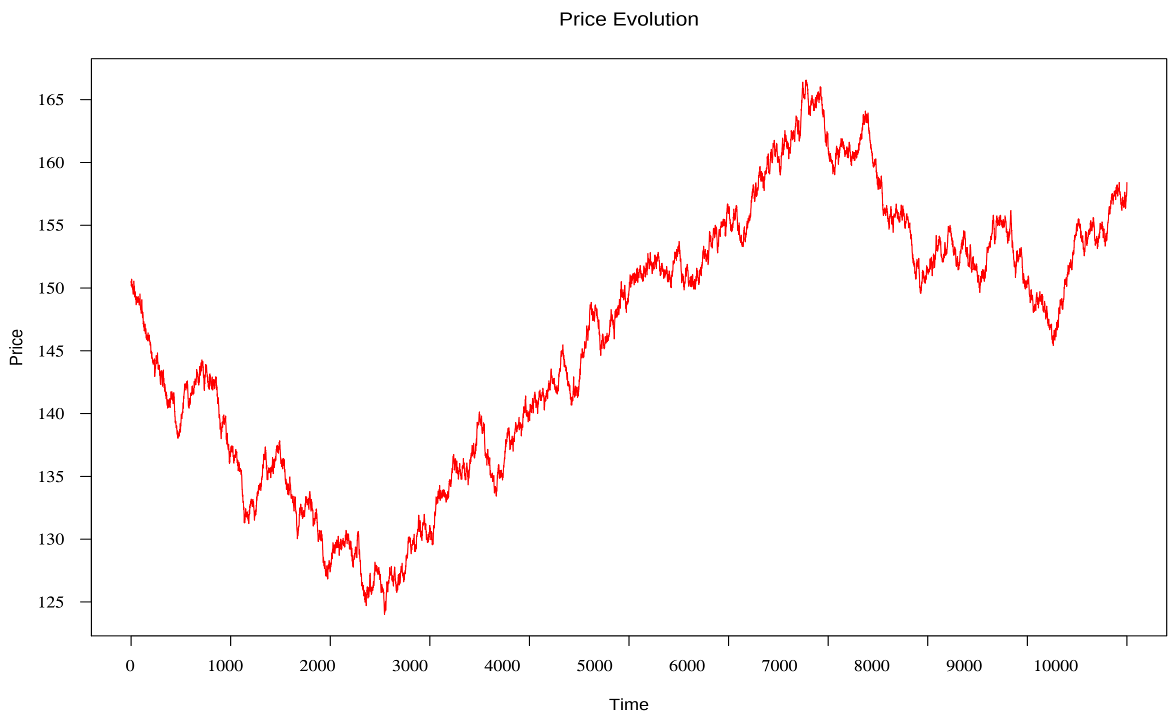

Figure 3: Price evolution in artificial market without Zipf's law.

period	p–value	p–value, Zipf
1–2000	0.0136	0.684
2001–4000	0.193	0.0233
4001–6000	0.062	0.102
6001–8000	0.557	0.113
8001–9999	0.436	0.19

Table 3: Dickey–Fuller test of null that price series has unit root. The "Zipf" attribute means that the initial money and shares were distributed according to Zipf's law.

period	p–value $(m = (2, 3, 4, 5))$	p–value $(m = (2, 3, 4, 5))$, Zipf
1–2000	$(0.35, 0.24, 0.24, 0.36)$	$(0.77, 0.44, 0.45, 0.46)$
2001–4000	$(0.63, 0.99, 0.96, 0.88)$	$(0.74, 0.93, 0.91, 0.87)$
4001–6000	$(0.0271, 0.10, 0.20, 0.16)$	$(0.56, 0.98, 0.69, 0.39)$
6001–8000	$(0.85, 0.89, 0.66, 0.68)$	$(0.91, 0.84, 0.81, 0.63)$
8001–9999	$(0.92, 1.00, 0.95, 0.55)$	$(0.67, 0.90, 0.89, 0.57)$

Table 4: BDS–test of null, i.e., whether returns are i.i.d. The "Zipf" attribute means that the initial money and shares were distributed according to Zipf's law.

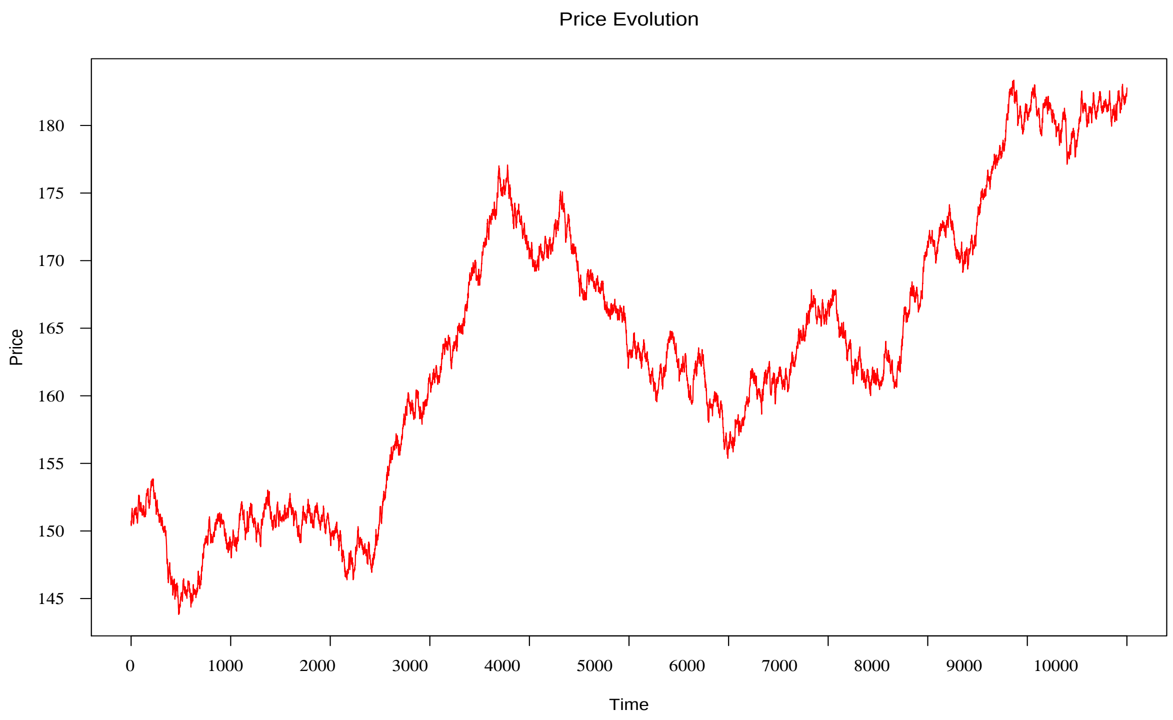

Figure 4: Price evolution in artificial market with Zipf's law.

3.2. Comparing Strategies

When comparing agents, the criteria of their success was the amount of money they earned – the final wealth they achieved. We also present how many times agents recounted the parameters of their forecasting strategies, but it is only an extrinsic indicator.

In the second type of simulation, the simulation with Zipf's law, the amount of money criterion was changed to the ratio of the final and initial wealth values. The results are shown in Table 5.

strategy	final wealth	recounting	fraction of wealths	recounting, Zipf
VAR	2171119	43.6	41899.5	47.2
VARMA	2226319	47.3	40942.4	48.3
FNN	2064486	50.7	39989.8	48.7
SRN	2047640	54.3	43640.5	52.7

Table 5: The average final wealth and number of recounting parameters of the forecasting strategy in the case of simulation without Zipf's law. The average ratio of the final and initial wealth values, and again the number of recounting parameters of the forecasting strategy in the case of simulation with Zipf's law.

We can see that the best type of forecasting strategy in terms of the highest average final wealth is VARMA. If we take a look at Table 6, we will see again that VARMA outperforms other models. Neural networks do not have better results, probably due to their complexity. If we want to use some neural network, the main problem is to choose a suitable topology. We chose the topologies of our networks randomly and that is probably why they did not perform better.

rank	strategy	final wealth	recounting
1	VARMA(3,2)	3385824	36
2	VARMA(2,2)	2623202	38
3	SRN(2,2,G)	2234082	40
4	FNN(1,1,G)	2225980	27
5	VARMA(1,2)	2223562	34
6	VARMA(3,1)	2220488	37
7	FNN(1,1,G)	2219796	36
8	VAR(3)	2201358	36

Table 6: The chart of 8 best strategies in term of highest final wealth – simulation without Zipf's law. Neural networks are stated with number of lags of variables they used (first number in brackets), number of hidden neurons (second number in brackets) and their learning algorithm (the letter – B for Backpropagation, G for Genetic algorithms).

On the other hand, Elman's networks overruled the second type of simulation, where the Zipf's law was used – see Table 7. But it was caused by the fact that in this group of traders number of the wealthy ones was the smallest – those who got more money and shares at the beginning of the simulation. The best traders, traders who increased their wealth most, are mostly the poorer ones.

From both Tables 6 and 7 and from Table 8 we can also see that neural networks with genetic algorithms for their training outperform those with backpropagation.

rank	strategy	fraction of wealths	recounting	initial wealth
1	VAR(2)	100090	66	0.576012
2	SRN(2,1,G)	64867	65	3.050353
3	FNN(2,3,G)	63902	64	3.787920
4	SRN(5,4,G)	59898	62	4.283111
5	FNN(1,1,G)	59548	63	3.993537
6	SRN(1,1,B)	58456	67	0.282847
7	FNN(3,1,G)	56663	64	6.684457
8	SRN(5,4,G)	51787	62	4.524875

Table 7: Chart showing the 8 best strategies in terms of the highest ratio of final and initial wealth – simulation with Zipf's law. Neural networks are swown with the number of lags of variables they used (the first number in brackets), the number of hidden neurons (the second number in brackets) and their learning algorithm (the letter – B for Backpropagation, G for Genetic algorithms).

strategy	training alg.	average final wealth
FNN	Backpropagation	2017850
FNN	Genetic algorithm	2117785
SRN	Backpropagation	1994354
SRN	Genetic algorithm	2108539

Table 8: Average final wealth in simulation without Zipf's law.

4. Conclusions

We created an artificial market without social interactions of agents in order to compare forecasting strategies in it. We tried to propose a new way of comparing strategies.

We succeeded in creating an artificial stock market which was close to the real one in terms of stylised facts. It seems that VARMA models outperformed VAR models and neural networks, which is probably due to the randomly chosen topology. Backpropagation also seems to be less efficient than genetic algorithms in this setup, which is probably due to the fact that genetic algorithms can seek through a larger space.

References

[1] Šíma, J., and Neruda, R.: *Teoretické Otázky Neuronových Sítí.* Matfyzpress, Praha, 1996.

[2] Lütkepohl, H.: *New Introduction to Multiple Time Series Analysis.* Springer, 2005.

[3] Chen, S., and Yeh, C.: *Genetic Programming in Agent-Based Artificial Markets: Simulations and Analysis.*

[4] Reinsel, G. C.: *Elements of Multivariate Time Series Analysis.* Springer, 2003.

[5] Matassini, L., and Franci, F.: On Financial Markets Trading, *Physica A: Statistical Mechanics and its Applications* **3–4** (2001), 526–542.

[6] LeBaron, B.: *Building the Santa Fe Artificial Stock Market.* Brandeis University, 2002.

[7] Saracoglu, R.: *The Maximum Likelihood Estimation of Parameters in Mixed Autoregressive Moving-Average Multivariate Models.* Federal Reserve Bank of Minneapolis, 1977.

[8] Berndt, E. K., and Hall, B. H., and Hall, R. E., and Hausman, J. A.: Estimation and Inference in Nonlinear Structural Models, *Annals of Economic and Social Measurement* (1974), 653–665.

[9] Dorffner, G.: Neural Networks for Time Series Processing, *Neural Network World* **6** (1996), 447–468.

[10] Boden, M.: *A Guide to Recurrent Neural Networks and Backpropagation.* In the Dallas project, 2002.

[11] Hamilton, J. D: *Time Series Analysis.* Princeton University Press, 1994.

[12] Palmer, R. G., and Arthur, W. B., and Holland, J. H., and LeBaron, B., and Taylor, P.: Artificial Economic Life: A Simple Model of a Stockmarket, *Physica D* **75** (1994), 264–274.