Journal of Machine Learning Research 1 (2000) 1-48 Submitted 4/00; Published 10/00

Virtual Stock Exchange. The Software

Jonas Mockus JMOCKUS@GMAIL.COM
Institute of Mathematics and Informatics
Vilnius University
Vilnius, LT-08663, Lithuania

Igor Katin IGOR@GETWEB.LT
Institute of Mathematics and Informatics
Vilnius University
Vilnius, LT-08663, Lithuania

Joana Katina JOANA@GETWEB.LT
Institute of Mathematics and Informatics
Vilnius University
Vilnius, LT-08663, Lithuania

Editor: to be appointed by editors

Abstract

Considerable scientific efforts were, and still are aimed at the optimization of prediction models and methods. More complex models provide better fitting to historical data, as usual. However, the recent results show that under some unfavorable conditions, the correlation is negative.

This work was designed to investigate both the influence of over-fitting and the relation of profits to the prediction errors. The second problem is probably even more important, since this is almost an open area so far, in the field of financial optimization.

The new, and, in our opinion, the most important observation of this work was that minimal prediction errors do not necessarily provide maximal portfolio profits. A possible reason of this seemingly counter-intuitive result is that these profits are not necessarily monotonous functions of prediction errors in the standard statistical prediction models.

If this observation will be confirmed by independent sources, then the results may influence the future research in the investment optimization. The emphasis should be switched to direct search of robust trading rules, less sensitive to unpredictable market changes. Therefore, the software is platform independent, open, and designed as a tool for scientific cooperation.

Keywords: Stock-Exchange, Computer Simulation, Java Software

1. Introduction

The software is in Java and implements the PORTFOLIO model including the basic algorithm and the data base for running the experiments and analyzing the results. The mathematical description is in Mockus et al. (2014). However, to grasp the basic ideas, a simple singe-stock version will be described using the notation of the prototype model in Mockus (2011).

©2000 Jonas Mockus and Igor Katin and Joana Katina.

The objective of the PORTFOLIO model is not forecasting, but simulation of stock exchange processes that are affected by predictions of the participants. The multi-stock extension and a number of different trading rules represent both the heuristics of potential investors and the well-known theoretical investment strategies.

This makes the model more realistic and allows the portfolio optimization in the space of investment strategies, in both the historical and virtual environments. This is an essential improvement comparing with traditional single-stock models with direct interaction of investment agents.

The "virtual" stock exchange can help in testing the assumption of rational investor behavior vs. the recent theories that explain financial markets by irrational responses of major market participants.

The model is designed as a tool to represent behavior of individual investor, which wants to predict how the expected profit depends on different investment rules using different forecasting methods of real and virtual stocks. It is assumed that only available information is the historic data of real stocks.

Optimization in the space of investment strategies and implementation of both the real and virtual stock market in the single model are the new properties of the PORTFOLIO model. The unexpected result was that the minimal stock price prediction errors do not necessarily provide the maximal profits. Therefore, the complete information is presented for the independent testing and verification of this important new result. This result can be tested and verified independently without special skills and equipment, all the experimental conditions are defined and reproducible.

2. PORTFOLIO Model

In this section, the software of the recent stock exchange model PORTFOLIO is described. The algorithmic diagram and the process logic are presented.

The profits of both investors and banks are calculated. The PORTFOLIO model simulates behavior of group of investors, who trade stocks in real and virtual environments. The optimization is performed on a set of investment strategies. The investor strategies include prediction models and trading rules. This is the main specific feature of the PORTFOLIO model.

Using this software, investors can choose one of 190 investment strategies, including ten trading rules and nineteen forecasting models. Three of these trading rules model known theoretical results, the others are new and simulates heuristics of different investors with different approaches to risk.

Investigating the real environment, historical stock prices of popular international companies can be used. In the virtual environment, prices are generated by simulation of behavior of up to eight different major investors. The random noise simulates the influence of small investors.

The structure of software presents possibility to extend model: to add new prediction methods and trading rules. For large scale automatic experiments, the MySQL technology was applied using the NetBeans and XAMPP tools. So, the software can be used, modified, tested and verified independently.

3. Virtual Stock Exchange, a Simple Version

Here the mathematical formulas of single-stock prototype model are presented as an introduction to the recent multi-stock PORTFOLIO model.

3.1 Buying and Selling Strategies

We consider a virtual market of I major players $i = 1, ..., I$ and $j = 1, ..., J$ stocks. The following notation is used:
$z(t, j) = z(t, i, j)$ is the price of stock j at time t, predicted by the player i ,
$Z(t, j)$ is the actual[1] price at time t,
$U(t, j) = U(t, i, j)$ is the actual profit accumulated at time t by the player i buying-selling stock j,
$\delta(t, j)$ is the dividend of stock j at time t,
$\alpha(t)$ is the yield at time t,
$\gamma(t)$ is the bank interest at time t,
$\beta(t, i) = \beta(t, i, j)$ is the relative stock j price change at time t as predicted by the player i

$$\beta(t, i, j) = (z(t + 1, i, j) - Z(t, j))/Z(t, j), \tag{1}$$

Expected profitability [2] (relative profit) $p(t, i, j)$ of an investment at time t depends on the predicted change of stock prices $\beta_i(t, j)$, dividends $\delta_i(t, j)$, the yield $\alpha(t)$, and the interest $\gamma(t)$

$$p(t, i) = \begin{cases} \beta(t) + \delta(t) - \gamma(t). & \text{investing borrowed money} \\ \beta(t) + \delta(t) - \alpha(t). & \text{investing own money} \end{cases} \tag{2}$$

The aim is profit, thus a customer i will buy some amount $n_b(t, i, j) \geq n(t, j)$ of stocks j, if profitability is greater comparing with the relative transaction cost $\tau(t, n)$; $p(t, i, j) > \tau(t, n)$, and will sell stocks, if the relative loss (negative profitability $-p(t, i, j)$) is greater as compared with the transaction cost $p(t, i, j) < -\tau(t, n)$, or will do nothing, if $-\tau(t, n) \leq p(t, i, j) \leq \tau(t, n)$. Here the relative transaction cost is defined as the relation

$$\tau(t, n) = \frac{\tau_0}{n(t, j)Z(t, j)}, \tag{3}$$

where τ_0 is the actual transaction cost and $n = n(t, j)$ is the number of transaction stocks. It follows from the equality $\tau(t, n) = p(t, i, j)$ that the minimal number of stocks to cover transaction expenses is

$$n(t, j) = \frac{\tau_0}{p(t, i, j)Z(t, j)}. \tag{4}$$

Therefore, the buying-selling strategy $S(t, i, j)$ of stock j by the customer i at time t in

1. The term 'actual' means simulated
2. The term "profit" can define losses if negative terms prevail.

terms of profitability levels is as follows

$$S(t,i,j) = \begin{cases} \text{buy } n_b(t,i) \geq n(t,j) \text{ stocks}, & \text{if } p(t,i,j) \geq \tau(t,n) \text{ and } n \leq n_b^{max} \\ \text{sell } n_s(t,i,j) \geq n(t,j) \text{ stocks}, & \text{if } p(t,i,j) \leq -\tau(t,n) \text{ and } n \leq n_s^{max}, \\ \text{wait}, & \text{if } |p(t,i,j)| \leq \tau(t,n^{max}). \end{cases} \quad (5)$$

Here $n^{max} = \max(n_b^{max}, n_s^{max})$, where n_b^{max} is the maximal number of stocks to buy, and n_s^{max} is the maximal number of stocks to sell.
If

$$n_b(t,i,j) = n_b^{max} \; and \; n_s(t,i,j) = n_s^{max}, \quad (6)$$

then this buying/selling strategy reflects the behavior of risk-neutral stockholders which invest all available resources if the expected profitability is higher than the transaction cost. If the expected losses are greater, then all the stocks are sold. It means that stockholders may tolerate a considerable probability of losses, if the expected profits are positive. In this way, the maximal expected profit is provided. However, the probability to get losses instead of profits could be near to 0.5.

From expressions (1) and (2), the buying-selling strategy $S(t,i,j)$ in terms of stock price levels is

$$S(t,i,j) = \begin{cases} \text{buy } n_b(t,i,j) \geq n(t,j) \text{ stocks}, & \text{if } Z(t,j) \leq z_b(t,n,i,j) \text{ and } n \leq n_b^{max}, \\ \text{sell } n_s(t,i,j) \geq n(t,j) \text{ stocks}, & \text{if } Z(t,j) \geq z_s(t,n,i,j) \text{ and } n \leq n_s^{max}, \\ \text{wait}, & \text{otherwise.} \end{cases} \quad (7)$$

Here the price level of the player i to buy at least $n = n(t,j)$ stocks at time t is

$$z_b(t,n,i,j) = z(t+1,i,j)/(1 - \delta(t) + \alpha(t) + h(t) + \tau(t,n)). \quad (8)$$

The price level of the player i to sell at least $n = n(t,j)$ stocks at time t is

$$z_s(t,n,i,j) = z(t+1,i,j)/(1 - \delta(t) + \alpha(t) + h(t) - \tau(t,n)), \quad (9)$$

where $z(t+1,i,j)$ is the stock j price predicted by the investor i at time $t+1$.
The market buying price at time t is the largest buying price of players $i = 1, ..., I$
$z_b(t,n) = z_b(t,n,i,j^{max})$,
where
$i^{max} = arg\max_i z_b(t,n,i,j)$.
The market selling price at time t is the lowest selling price of players $i = 1, ..., I$
$z_s(t,n) = z_b(t,n,i,j^{min})$,
$i^{min} = arg\min_i z_s(t,n,i,j)$.

3.2 Price Simulation

3.2.1 Buying-selling price

The market buying price of stock j at time t is the largest buying price of players $i = 1, ..., I$
$z_b(t,n,j) = z_b(t,n,i,j^{max})$,

where
$i^{max} = arg\max_i z_b(t,n,i,j)$.
The market selling price at time t is the lowest selling price of players $i = 1,...,I$
$z_s(t,n,j) = z_s(t,n,i,j^{min})$,
$i^{min} = arg\min_i z_s(t,n,i,j)$.
The number of stocks j owned by the player i at time $t+1$ is

$$N(t+1,i,j) = \begin{cases} N(t,i,j) + n_b(t,n,i,j), & \text{if } Z(t,j) < z_b(t,n,j), \\ N(t,i,j) - n_s(t,n,i,j), & \text{if } Z(t,j) > z_s(t,n,j), \\ N(t,i,j), & \text{if no deal.} \end{cases} \tag{10}$$

Here $n_b(t,n,i,j)$ and $n_s(t,i,j)$ are the numbers of stocks j for buying and selling operations by the player i at time t.

3.3 Investors Profit

The product $N(0,i,j)\ Z(0,j)$ is the initial investment to buy $N(0,i,j)$ shares j using an investors' own capital at the initial price $Z(0,j)$. The initial funds to invest are $C_0(0,i)$ and the initial credit limit is $L(0,i)$.
$L(t,i),\ t = 1,...,T$ is the credit available for a customer i at time t . The investors' own funds in cash $C_0(t,i)$ available for investing at time t are defined by the recurrent expression

$$C_0(t,i) = C_0(t-1,i) - \sum_j (N(t,i,j) - N(t-1,i,j))\ Z(t,j), \tag{11}$$

where $t = 1,...,T$. Here the product $(N(t,i,j) - N(t-1,i,j))\ Z(t,j)$ defines the money involved in buying-selling stocks.
Stocks are obtained using both investor's own money $C_0(t,i)$ and the funds $b(t,i)$ borrowed at the moment t. The borrowed sum of the stockholder i accumulated at time t is

$$B(t,i) = \sum_{s=1}^{t} b(s,i)), \tag{12}$$

The symbol $b(t,i)$ shows what the user i borrows at the moment t

The general borrowing expenses are

$$B_{sum}(t,i) = B(t,i) + \sum_{s=1}^{t} B(s,i))\ \gamma(s,i)), \tag{13}$$

where the first term denotes the loan accumulated at time t and the second term shows the interest.
An investor i gets a profit as the difference between the income from selling and buying stocks $D(t,i)$ and expenses for the borrowing funds $B_{sum}(t,i)$

$$U(t,i) = C_0(t,i) + D(t,i) - B_{sum}(t,i), \tag{14}$$

where

$$D(t,i) = \sum_j N(t,i,j)\ Z(t,j) - N(0,i,j)\ Z(0,j), \tag{15}$$

The investors i profit at the end of investment period is denoted by

$$U_i = U(T, i). \tag{16}$$

The funds available for the investor i at time t are

$$C(t, i) = C_0(t, i) + L(t, i) - B_{sum}(t, i). \tag{17}$$

An investor is trying to maximize gains by borrowing money to invest in shares that appreciate more than what it costs him by way of interest. It means leveraging shares for an investment.
The number $n_b(t.j)$ of stocks j to buy at the time t is restricted by the following inequalities

$$\sum_j n(t, j)Z(t, j) \leq C(t, i). \tag{18}$$

and

$$n(t, j) \leq n_b(t, i, j). \tag{19}$$

where the last inequality restricts the transactions costs. The stockholder will be insolvent at the time $t = t_i^*$ if the loan exceeds the assets

$$B_{sum}(t_i, j) > C_0(t_i, j) + L(t_i, j) + \sum_j N(t, i, j)\ Z(t, j), \tag{20}$$

since there will not be enough money to pay back all the borrowing expenses $B_{sum}(t_i^*, i)$. This can happen without buying additional stocks, because the interest $B_{sum}(t, i)$ accumulates automatically.

Part of Profit by Stock j In longer term investment strategies using the Sharpe ratio, the general profit should be divided between different stocks. Denote by $C_0(0, i, j) \leq L(0, i, j)$ the initial funds to be invested in the stock $j = 1, ..., J$, where $L(0, i, j)$ is the initial credit limit for stock j and

$$\sum_j L(0, i, j) = L(0, i) \tag{21}$$

For example, the initial funds may be divided into equal parts

$$\sum_j C_0(0, i, j) = C_0(0, i)/J \tag{22}$$

The investors' own funds in cash $C_0(t, i, j)$, accumulated buying-selling stocks j and available for investing at time t in the stock j, are defined by the recurrent expression below

$$C_0(t, i, j) = C_0(t - 1, i, j) - (N(t, i, j) - N(t - 1, i, j))\ Z(t, j), \tag{23}$$

where $t = 1, ..., T$. Here the product $(N(t, i, j) - N(t - 1, i, j))\ Z(t, j)$ defines the money involved in buying-selling stocks.j

Stocks are obtained using both investors' own money $C_0(t,i,j)$ and the funds $b(t,i,j)$ borrowed at the moment t. The borrowed sum of the stockholder i for the stock j accumulated at time t is

$$B(t,i,j) = \sum_{s=1}^{t} b(s,i,j)), \tag{24}$$

where

$$\sum_{j} B(t,i,j) = B(t,i) \tag{25}$$

The symbol $b(t,i,j)$ shows what the user i borrows at the moment t for the stock j.

The general borrowing expenses for stock j are

$$B_{sum}(t,i,j) = B(t,i,j) + \sum_{s=1}^{t} B(s,i,j))\ \gamma(s,i)), \tag{26}$$

where the first term denotes the loan accumulated at time t, the second term shows the interest, and

$$\sum_{j} B_{sum}(t,i,j) = B_{sum}(t,i). \tag{27}$$

The investor i gets a profit as the difference between the income from selling and buying stocks $D(t,i,j)$ and expenses for the borrowing funds $B_{sum}(t,i,j)$

$$U(t,i,j) = C_0(t,i,j) + D(t,i,j) - B_{sum}(t,i,j), \tag{28}$$

where

$$D(t,i,j) = N(t,i,j)\ Z(t,j) - N(0,i,j)\ Z(0,j), \tag{29}$$

The investor's i profit from the stock j at the end of investment period is denoted as

$$U_{i.j} = U(T,i.j), \tag{30}$$

where

$$\sum_{j} U_{i.j} = U_i, \tag{31}$$

If for some reason equalities (25), (27), and (31) are violated, then the normalization of components may be applied to restore them..

3.4 Basic Software Scheme

In this section, the basic algorithmic scheme is presented , see Figure 1. It reflects model's work-flow and shows the main blocks.

—

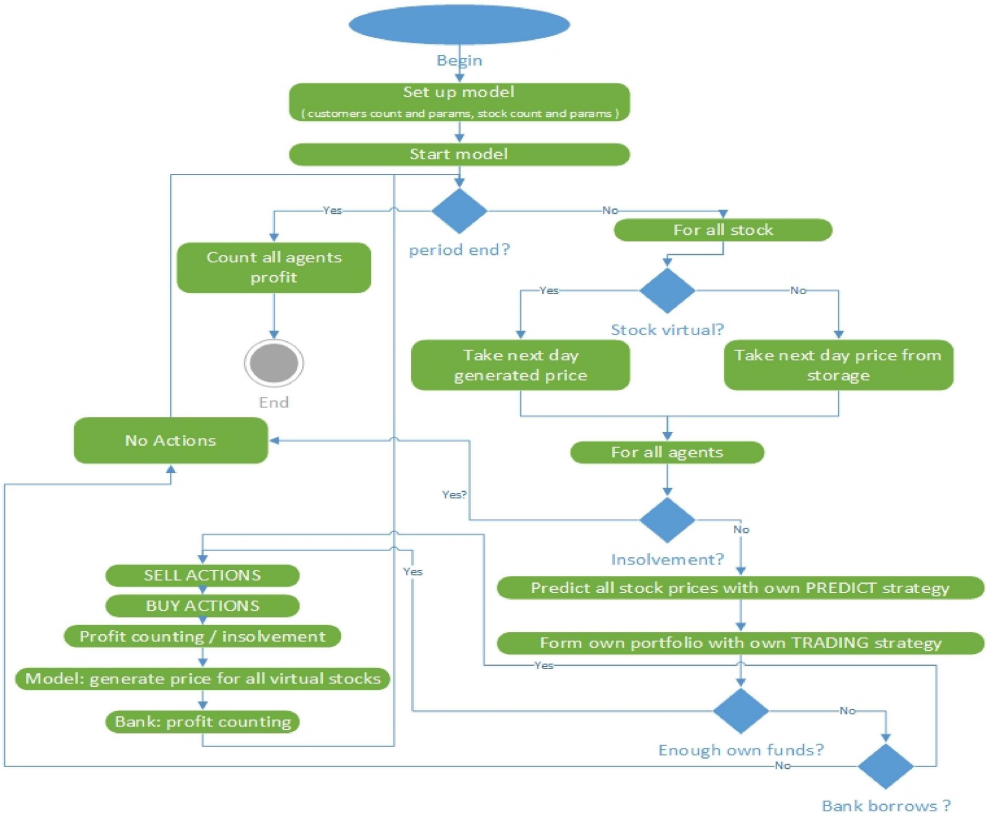


Figure 1: *Basic PORTFOLIO scheme*

4. User Guide

4.1 Creating the Database "experiment"

The guide is written for Windows environment. The procedure is similar in Mac and Linux

Step 1, see Figure 2. Install XAMPP. To do this, download the free XAMPP software, and, after some "Next" steps, select Apache, MySQL, PHP, phpMyAdmin components:

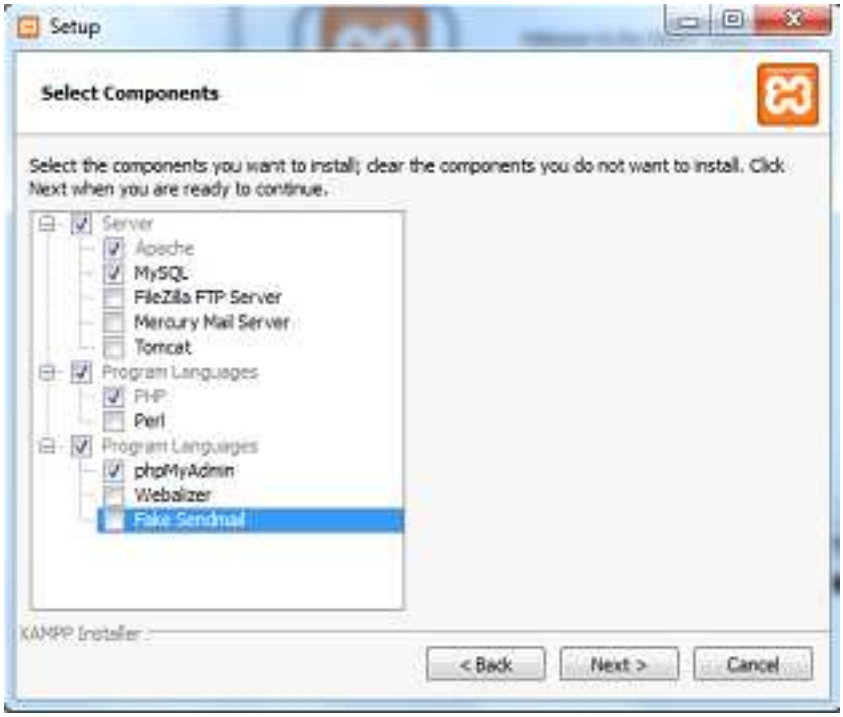


Fig. 2 *Selection of XAMPP components*

After some additional "Next" steps check "Finish" to open the XAMPP Control Panel, see Figure 3:



Fig. 3 *Starting Apache and MySQL*

By checking “Netstat” button, provide that ports 80 and 3306 would be free. Otherwise, change the ports in the Apache and MySQL settings by checking the corresponding “Config” button and editing “httpd.config” file, see Figure 4:

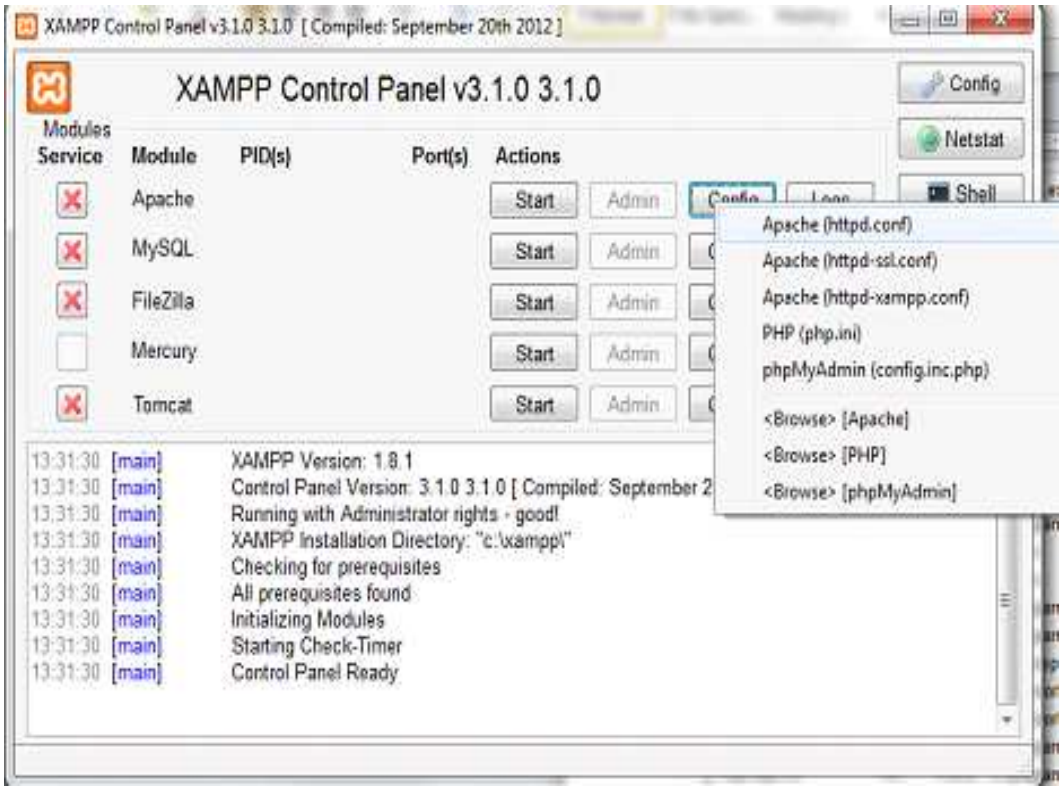


Fig. 4 *Configuring Apache and MySQL ports*

If port 80 is not free, then in the line “Listen 80” change the port number to the free one, for example 90.

The SSL port can be changed by editing the line “Listen 443” in the “httpd-ssl.conf” file.

MySQL port can be changed by editing the line “Port = 3306”.

Step 2. Start phpMyAdmin. If the ports were not changed start http://localhost/phpmyadmin/. If, for example, the port 80 was changed to 90, then start http://localhost:90/phpmyadmin/. This operation opens the window, see Figure 5:

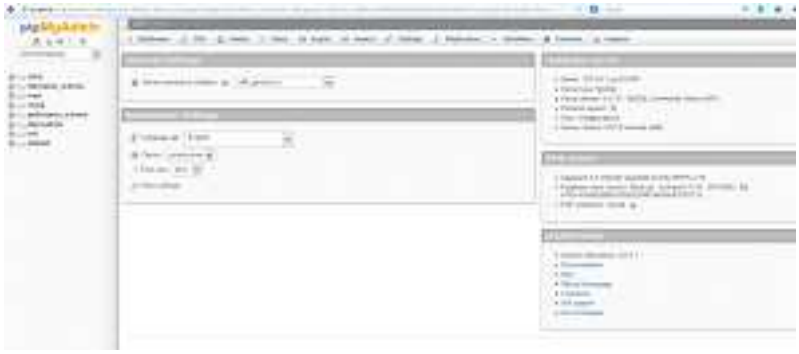


Fig. 5 *Opening phpmyadmin window*

Step 3. Create the data base, for example “experiment”. In Figure 6, check “Database”, write the name “experiment”, and select corresponding code:



Fig. 6 *Creating the database “experiment”*

Check “Create”, and see the new data base in the left side. If no data base is seen, check for the errors and repeat the process.

4.2 Running the PORTFOLIO Model

Step 4. Extract the Java archive ’PORTFOLIO/stock/stock.zip’ and run ’dist/stock.jar’

Step 5. Select the number of stocks and other initial data. In Figure 7, four stocks are selected, all four of virtual market (generated data).

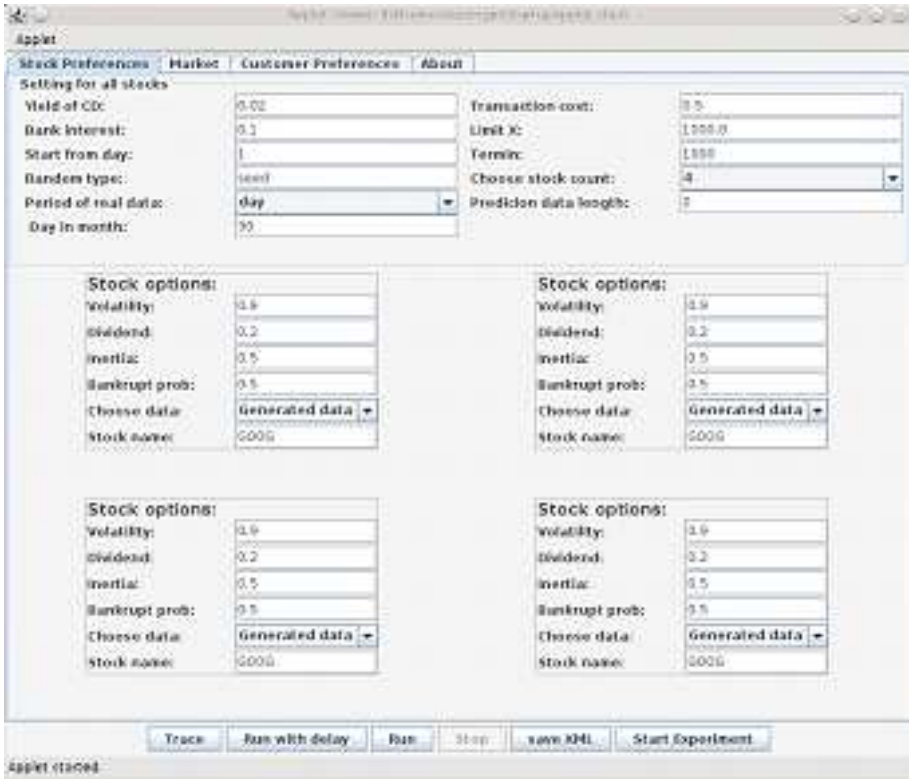


Fig. 7 *Setting the number and parameters of stocks*

Select the investors and trading strategies. In Figure 8 two investors were selected:

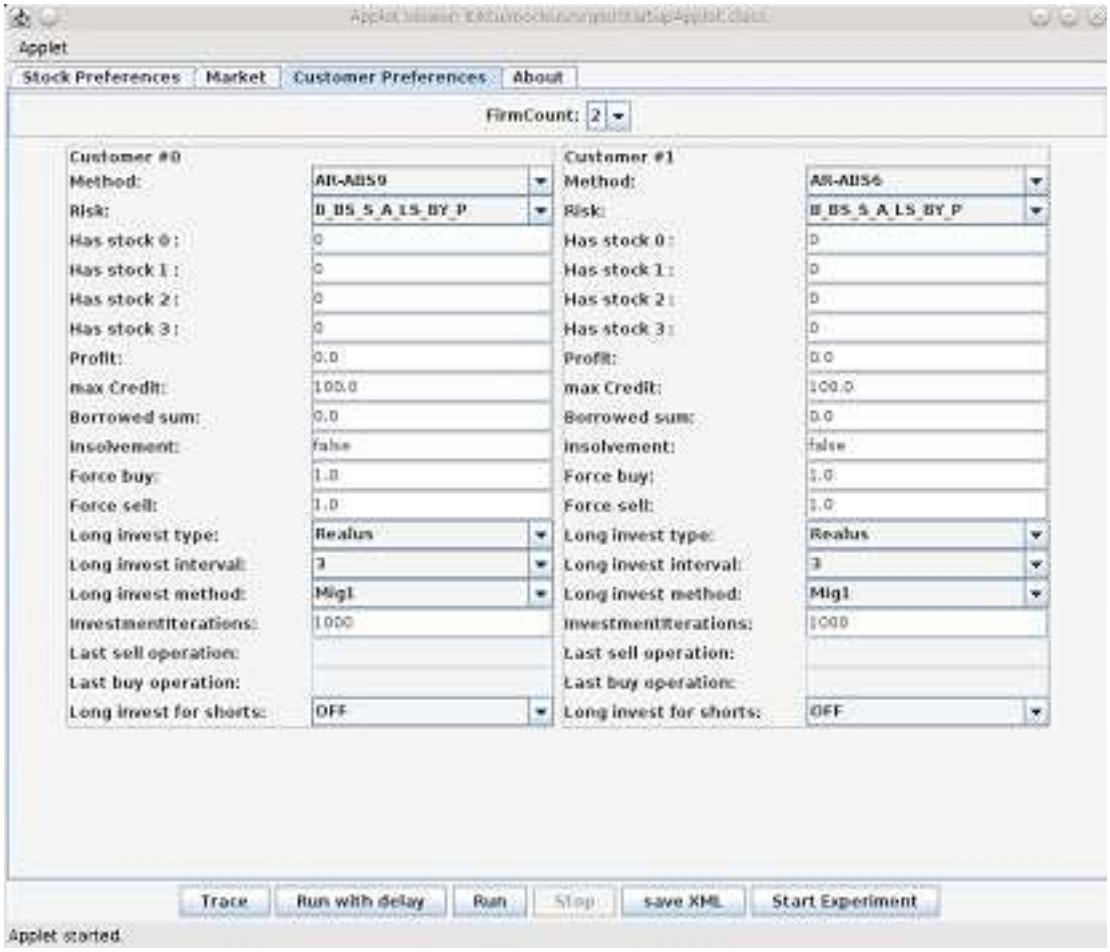


Fig. 8 *Setting the number of investors and their and parameters*

Step 6. Start experiment by checking the corresponding button in the lower-right of Figure 8. If no mistakes, the Figure 9 will appear:

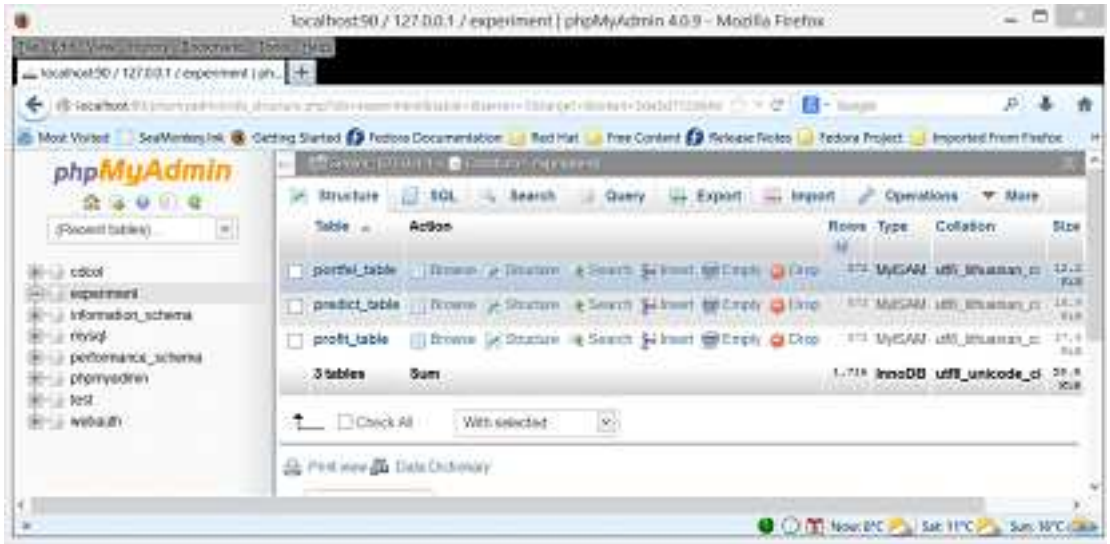


Fig. 9 *The experiment window*

After the experiment is finished, the data can be extracted by standard means of SQL.

4.3 The Java Code

Java code is in the extracted archive 'stock.zip' in the folder 'src'. The Figure 10 shows a fragment of Java class "StockTradeThread" in the src folder 'lt.ktu.mockus.srgm' opened as the NetBeans project 'stock'.

Additional trade rules can also be included. Recompile, if needed, by "clean and build" and run 'dist/stock.jar'.

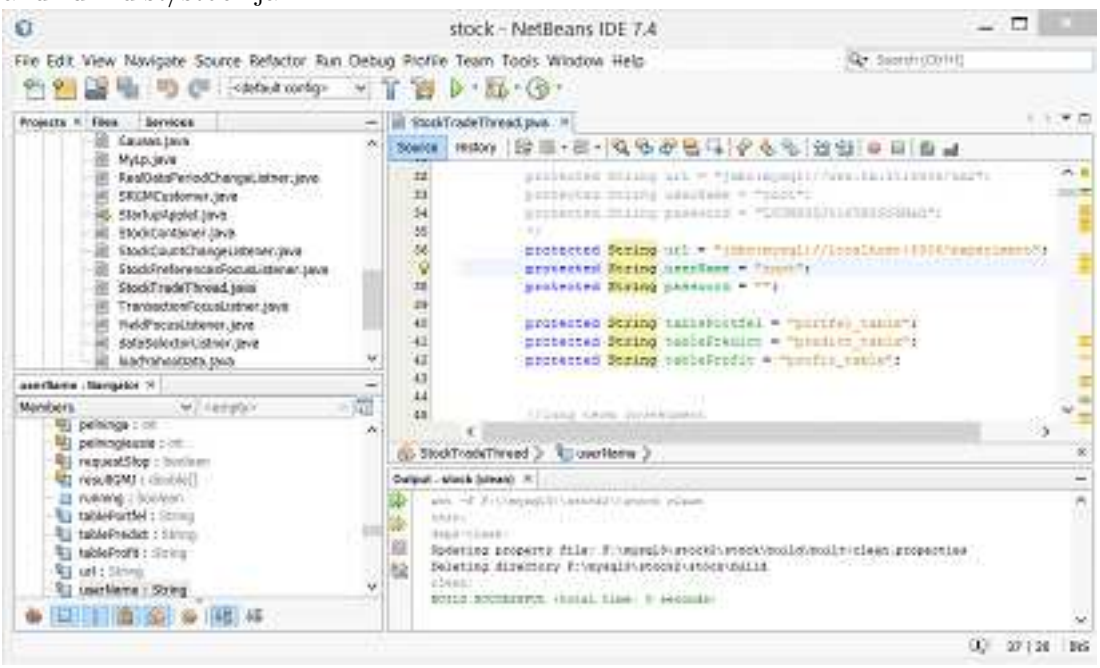


Fig. 10 *A fragment of Java code in the NetBeans project*

The software of the complete model is not easy for understanding. However, the means for the calculation of profits and prediction errors are very simple. The profit is calculated as a difference between selling and buying prices minus transaction costs and bank charges. The prediction errors, for example MAE, is the absolute value of difference between the predicted and observed stock price values.

References

J. Mockus. On simulation of Nash equilibrium in the stock exchange context. *Informatica*, 23:77–104, 2011.

J. Mockus, I. Katin, and J. Katina. On the optimization of investment strate- gies in the context of virtual financial market by the individual ap- proach to risk. *Informatica*, 25: 241–264, 2014.