Trading by estimating the forward distribution using quantization and volatility information

Attila Ceffer, Janos Levendovszky

Abstract

In this paper, novel algorithms are developed for electronic trading on financial time series by using quantization and volatility information for achieving High Frequency Trading (HFT). The proposed methods are estimation based and trading actions are carried out after estimating the Forward Conditional Probability Distribution (FCPD) on the quantized return values. This is motivated by the fact, that the forward distribution can give a more reliable prediction on the future values of the corresponding time series than simply using mean-square-error prediction. From the past samples it is easy to learn the Conditional Expected Value (CEV) if the possible past return values are small, which needs quantization. From the CEV one needs to obtain the FCPD, which can be achieved by introducing a special encoding scheme on the observed prices.

For estimating FCPD, a FeedForward Neural Network (FFNN) will be used, which can provide a good estimation if the return levels are quantized properly and a special encoding scheme is applied. Based on this estimation, a trading signal is launched if the probability of price change becomes significant. This is measured by a quadratic criterion. Due to the encoding scheme and quantization, the complexity of learning and estimation have been reduced, which paves the way towards HFT. The efficiency of trading is further increased by using volatility information also estimated from the FCPD.

The performance analysis of the new method has proven to be profitable according to the different performance measure (achieved profit, average profit, maximum drawdown …etc.) on all historical time series taken from FOREX. However, positive gain has usually been materialized on mid-prices. In order to beat the secondary effects, such as bid-ask spread and transaction costs, we needed to further optimize the method by optimizing meta-parameters with Simulated Annealing. This positive effect was demonstrated in the case of EUR/USD exchange rates.

Keywords: algorithmic trading, neural networks, conditional probability distribution, quantization

JEL Classification: C45

1. Introduction

The selection of portfolios which are optimal in terms of risk-adjusted returns has been an intensive area of research in the recent decades [1]. Furthermore, the main focus of portfolio optimization tends to move towards the application of High Frequency Trading (HFT) when a huge amount of financial data is taken into account within a very short time interval and trading with the optimized portfolio is also to be performed at high frequency within these intervals [2]. HFT presents a challenge to both algorithmic and architectural development, because of the need for developing algorithms running fast on specific architectures (e.g. GPGPU, FPGA chipsets) where speed is the most important attribute. On the other hand, profitable portfolio optimization and trading needs the evaluation of rather complex goal functions with different constraints which sometimes cast the problem in the NP hard domain [3] [4] [5]. As a result, the computational paradigms emerging from the field of neural computing, which support fast parallel implementation, are often used in the field algorithmic trading [6] [7] [8].

In this paper the, trading is done based on the estimated Forward Distribution (FD) based on the article [9]. Since FD takes its values on the possible asset prices (or returns), the number of probabilities to be estimated explodes exponentially with respect to the to the length of the memory. As a result, for the sake of accurate estimations, we need a very large training set, which prevents HFT due to the low speed of learning. In order to speed up data collection and learning (using a small number of samples), we need to quantize the asset prices. Instead of quantizing the price itself, we quantize the change of the prices (returns), which varies in a smaller interval. In the paper, we use the Lloyd-Max algorithm for quantization to attain a good trading performance.

Since the FCPD are calculated only on the quantization levels, the performance of the method highly depends on the quality of these levels. In this way, the error in the estimation of FCPD can be greatly reduced by using a proper quantization algorithm.

In order to beat the bid-ask spread, we have introduced further adjustments in the algorithm:

- trading only in high volatility periods;
- optimizing the meta-parameters by Simulated Annealing.

With these adjustments, we could secure positive profit in the presence of bid-ask spread on EUR/USD currency exchange rates.

The material summarized above is organized as follows:

- in Section 2, the theoretical background of trading by FFNN is outlined;
- in Section 3, encoding schemes are introduced to obtain FCPD;
- in Section 4, we apply optimal quantization for better FCPD estimation;
- in Section 5, we present the further adjustments in the trading algorithm;
- in Section 6, the computational model of the trading algorithm is mapped out;
- in Section 7, we validate the methodology on generated and real historical financial time series.

2. Theoretical background – trading with FFNN

Let us assume that there is a vector valued random asset price process (e.g. the values of currency cross rates), which is denoted by $\mathbf{s}(t) = \left(s_1(t),...,s_2(t)\right)$. The return series of $\boldsymbol{s(t)}$ is defined as $r_i(t) = \dfrac{s_i(t)}{s_i(t-1)} - 1$.

A portfolio is expressed by a portfolio vector $\mathbf{w}(t) = \left(w_1(t),...,w_2(t)\right)$ yields a linear combination of asset returns (being the portfolio return) as a time series $x(t) := \sum_{i=1}^{N} w_i r_i(t) = \mathbf{w}^T \mathbf{r}(t)$. The possible returns are taken from a discrete set $r_i(t), x(t) \in Q = \{q_1,...,q_M\}$. Let

$$P_i = P\left(x(t+1) = q_i \,\middle|\, x(t) = q_j,...,x\left(t-L+1\right) = q_M\right), i = 1,...,M$$

denote the conditional probabilities having L past observations at hand. With these conditional probabilities, one can devise a Bayesian trading strategy, as follows:

if $P\left(x_i(t+1)>0\right)$ then buy at t and sell at $t+1$;

if $P\left(x_i(t+1)<0\right)$ then sell at t and buy back at $t+1$.

Or more precisely,

$$\begin{aligned} &\text{if } \sum_{i:i>\frac{M}{2}} P_i > \sum_{i:i<\frac{M}{2}} P_i \text{ then buy at } t \text{ and sell at } t+1; \\ &\text{if } \sum_{i:i>\frac{M}{2}} P_i < \sum_{i:i<\frac{M}{2}} P_i \text{ then sell at } t \text{ and buy at } t+1. \end{aligned} \tag{2.1}$$

One can also see that the larger the value $\left(\sum_{i:i>\frac{M}{2}} P_i - \sum_{i:i<\frac{M}{2}} P_i\right)^2$ the more reliable decision we can make on trading. As a result, the risk of the trading can be fine-tuned by choosing a proper $\varepsilon>0$ for which trading can only take place if $\left(\sum_{i:i>\frac{M}{2}} P_i - \sum_{i:i<\frac{M}{2}} P_i\right)^2 > \varepsilon$.

In order to perform trading as detailed above, one needs the estimation of the conditional probabilities $P_i = P\left(x(t+1)=q_i \,\middle|\, x(t)=q_j,...,x\left(t-L+1\right)=q_M\right), i=1,...,M$. The estimation of $P_i, i=1,...,M$ can be given by a FFNN, based on observing the historical part of the time series of a given portfolio *x(t)*. Based on these observations, one can construct a training set containing some samples followed by the observed forward return *x*(*k*+1) given as follows: $\tau^{(K)}=\left\{\left(\mathbf{x}_k, x(k+1)\right), k=1,...,K\right\}$ where $\mathbf{x}_k=\left(x(k),...,x(k-L+1)\right)$. Let us then construct an FFNN based predictor $\tilde{x}(t+1)=Net\left(\mathbf{\Theta},\mathbf{x}_t\right)$, where $\mathbf{x}_t=\left(x(t),...,x(t-L+1)\right)$. After learning we obtain

$$\mathbf{\Theta}_{opt}^{(K)}:\min_{\mathbf{\Theta}}\frac{1}{K}\sum_{k=1}^{K}\left(x(k+1)-Net\left(\mathbf{\Theta},\mathbf{x}_k\right)\right)^2 \tag{2.2}$$

and the FFNN will provide the optimal non-linear prediction

$$Net\left(\mathbf{\Theta},\mathbf{x}_t\right)=E\left(x(t+1)\middle|\mathbf{x}_t\right), \text{ because}$$

$$\lim_{K\to\infty}\frac{1}{K}\sum_{k=1}^{K}\left(x(k+1)-Net\left(\mathbf{\Theta},\mathbf{x}_k\right)\right)^2 = E\left(x(t+1)-Net\left(\mathbf{\Theta},\mathbf{x}_t\right)\right)^2 \text{ and}$$

$\min_{\mathbf{\Theta}} E\left(x(t+1)-Net\left(\mathbf{\Theta},\mathbf{x}_t\right)\right)^2 \to Net\left(\mathbf{\Theta},\mathbf{x}_t\right)=E\left(x(t+1)\middle|\mathbf{x}_t\right)$ (for further details see [10] [11] [12]).

3. Coding scheme to obtain FCPD

To perform the trading algorithm elaborated by formulas (2.1), one needs the conditional probabilities. In order to obtain them, let us encode the possible values of the return of the portfolio into an orthonormal vector set:

$$q_l \to \mathbf{r}^{(l)} : r_i^{(l)} = \delta_{li} = \begin{cases} 1 \text{ if } i = l \\ 0 \text{ otherwise} \end{cases}$$

and rewrite the training set according to the encoding mechanism:

$\tau^{(K)} = \left\{ \left(\mathbf{r}_{k+1}, \mathbf{x}_k \right), k = 1,..., K \right\}$, where $\mathbf{r}_{k+1} = \mathbf{r}^{(l)}$ if $x\left(k+1\right) = q_l$.

Then by minimizing the error function

$$\frac{1}{K} \sum_{k=1}^{K} \left\| \mathbf{r}_{k+1} - Net\left(\mathbf{\Theta}, \mathbf{x}_k \right) \right\|^2 \to E \left\| \mathbf{r} - Net\left(\mathbf{\Theta}, \mathbf{x} \right) \right\|^2,$$

one will obtain $Net\left(\mathbf{\Theta}_{opt}^{(K)}, \mathbf{x} \right) = E\left(\mathbf{r} \middle| \mathbf{x} \right)$, where due to the encoding, component l of the conditional expected value will yield the corresponding conditional probability as

$$E_l\left(\mathbf{r} \middle| \mathbf{x} \right) = \sum_{i=1}^{M} r_l^{(i)} p\left(\mathbf{r}^{(i)} \middle| \mathbf{x} \right) = \sum_{i=1}^{M} \delta_{li} p\left(\mathbf{r}^{(i)} \middle| \mathbf{x} \right) = p\left(\mathbf{r}^{(l)} \middle| \mathbf{x} \right) = p\left(x(t+1) = q_l \middle| \mathbf{x} \right) = P_l$$

Having the conditional distribution at hand, one can then implement the trading strategy discussed above, in the form of

$$\begin{aligned} &\text{if } \left(\sum_{i:i>\frac{M}{2}} P_i - \sum_{i:i<\frac{M}{2}} P_i \right)^2 \geq \varepsilon \text{ then buy at } t \text{ and sell at } t+1; \\ &\text{if } \left(\sum_{i:i>\frac{M}{2}} P_i - \sum_{i:i<\frac{M}{2}} P_i \right)^2 \leq -\varepsilon \text{ then sell at } t \text{ and buy at } t+1. \end{aligned} \tag{3.1}$$

Unfortunately, the method outlined above, requires high complexity neural network as the dimension of the output $\mathbf{y} = Net\left(\mathbf{\Theta}, \mathbf{x} \right)$ is $\dim(\mathbf{y}) = M$ which is the number of possible returns. The architecture is shown on *Figure 1*.

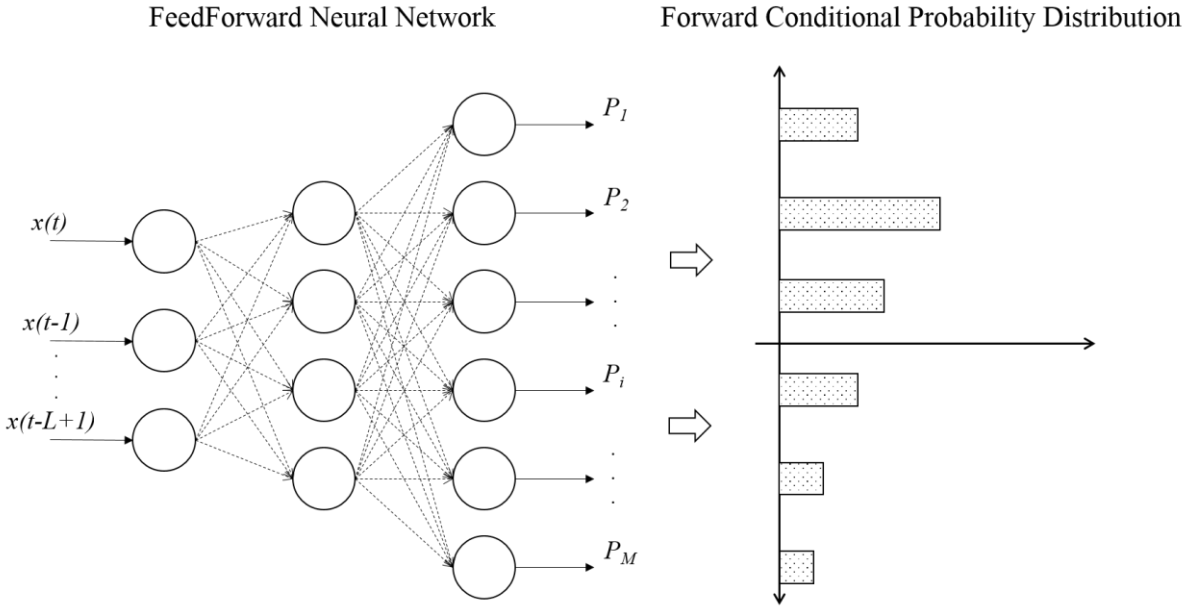

Figure 1: The neural architecture to estimate FCPD

High complexity FFNN contains a large number of free parameters which, in turn, requires a large learning set to train. This will prevent fast execution of the strategy and, as a result, hinders HFT. Thus, the present effort in this section is focused on decreasing the number of outputs, which will also reduce the number of free parameters to optimized. In order to achieve this, we quantize the time series.

4. Optimal quantization for improving the estimation of FCPD

The returns of a financial time series can be approximated by Gaussian random variables. However, equidistant quantization is only optimal for samples following uniform distribution. In order to overcome this shortcoming, we have quantized the expected value of the squared quantization error (i.e. the squared difference between original and quantized signals) can be reduced by applying non-equidistant quantization, due to the fact that quantizing the components which occur with smaller probability with larger error than the components which occur with higher probability, the overall error can be made smaller [13] [14]. In this way, one can obtain a more accurate estimation of the FCPD, which may yield better trading decisions. To determine the optimal quantization levels, we used the Lloyd-Max algorithm.

The Lloyd-Max algorithm:

1. use an initial set of representative levels: $q_i \quad i = 1, 2, ..., M$
2. assign each sample $x(t)$ in training set $\tau^{(K)}$ to closest representative q_i:
$$C_i = \left\{ x \in \tau^{(K)} : Q(x) = i \right\} \quad i = 2, 3, ..., M$$
3. calculate new representative levels: $q_i = \frac{1}{\|C_i\|} \sum_{x \in B_i} x \quad i = 1, 2, ..., M$
4. repeat 2. and 3. until no further distortion reduction (or applying a stopping criterion).

Our simulations have proven that by running the Lloyd Max algorithm, the quantization error drops to 10 times lower than using equidistant quantization due to minimizing the objective function $\sum_{i=1}^{M-1} \int_{C_i}^{C_{i+1}} \left(x - q_i \right)^2 p(x) dx$.

5. Further adjustments in the trading algorithm

The results have shown, that profitable trading is feasible using neural networks by predicting the forward conditional distribution on mid prices. However, to achieve profits in the presence of the bid ask spread and transaction costs, further refinements are needed.

5.1. Trading in high volatility periods

Based on the predicted standard deviation we can improve the trading efficiency. Each time the neural network gives an entry signal (either ‘long’ or ‘short’), we calculate the standard deviation from the forward conditional distribution as

$$\sigma(t) = \sqrt{\sum_{i=1}^{M} P_i \left(q_i - \sum_{i=1}^{M} P_i q_i \right)^2} .$$

If the standard deviation reaches a given threshold (high volatility), we enter into the trade, otherwise (low volatility), we stay away from the market.

$$\text{if } \left(\sum_{i:i>\frac{M}{2}} P_i - \sum_{i:i<\frac{M}{2}} P_i \right)^2 \geq \varepsilon \text{ and } \sigma(t) > \eta \text{ then buy at } t \text{ and sell at } t+1;$$
$$\text{if } \left(\sum_{i:i>\frac{M}{2}} P_i - \sum_{i:i<\frac{M}{2}} P_i \right)^2 \leq -\varepsilon \text{ and } \sigma(t) > \eta \text{ then sell at } t \text{ and buy at } t+1. \tag{5.1}$$

5.2. Optimizing the meta-parameters

We introduced three methods to improve the performance:

1. applying Lloyd-Max quantization algorithm to obtain accurate conditional probability estimation (see Section 4.);
2. applying a filter which let us entering the trade only if the probability of upside (or downside) movement reaches a given threshold ε (see formula 3.1);
3. applying a filter which let us entering the trade only if the standard deviation of the predicted conditional forward distribution reaches a given threshold η (see Section 5.1.).

In order to get the best results we should not only test these methods separately, but the parameters of the model must be optimized together leading to meta-model optimization performed by Simulated Annealing [15].
We optimize the following parameters:

- *M*, the number of quantization levels;
- *L*, the memory of the process (number of inputs of FFNN);
- ε, threshold for probabilities;
- η, threshold for standard deviation;
- *N*, the number of neurons in the hidden layer;
- T_{train}, the length of training data expressed in hours;
- $T_{retrain}$, the retraining time window expressed in hours.

These parameters are optimized by Adaptive Simulated Annealing algorithm, which is available e.g. in MATLAB.

6. Computational approach

We wrote a comprehensive optimization framework to optimize the metaparameters on a parallel manner. The model is shown on *Figure 2*.

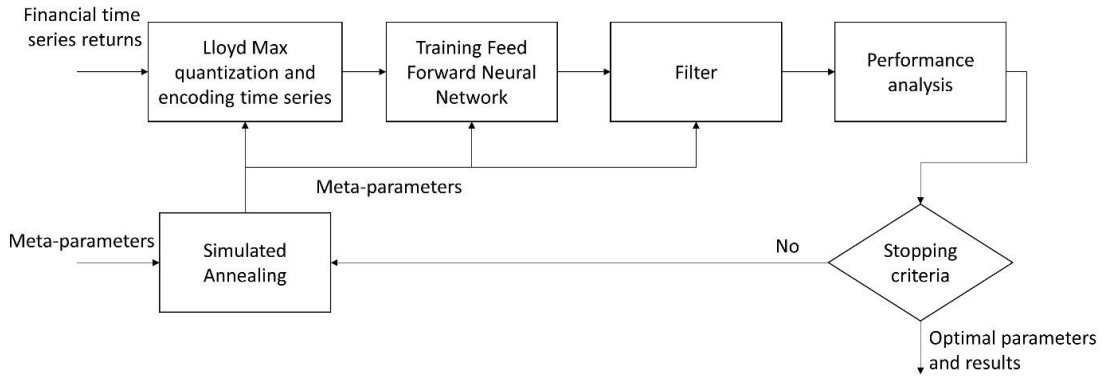

Figure 2: Optimization procedure

The FFNN model parameter identification and the evaluation of prediction by calculating the conditional probabilities is made on multiple portfolios, which helps us select an optimal portfolio. Then, the trading signal for taking the appropriate trading actions is made on the basis of the identified portfolio and the cost function.

Our computational framework is shown by *Figure 2* and detailed as follows:

- select a single asset portfolio vector $\mathbf{w}^T = \{w_1, ..., w_N\}$ and the corresponding time series;
- quantize the time series;
- fit a Feed Forward Neural Network to the time series of the selected portfolio by using the Levenberg-Marquardt learning algorithm;
- evaluate the objective function by predicting the future conditional probability distribution according to the identified model given the portfolio;
- continue the optimization process until the optimal parameters are obtained. The objective function we use is maximizing the average profit per trade;
- form a trading signal based on the price behaviour of the optimal portfolio to decide on which trading action is to be launched;

Finally, one can carry out a performance analysis by testing and evaluating various numerical indicators for the sake of comparing the profitability of the different methods.

Since Lloyd-Max quantization algorithm works on an iterative manner, it significantly increases the computing time. However, we have shown, that using this optimized quantization, better profit can be achieved on each time series we tested on.

6. Performance analysis

An extensive back-testing framework has been created to handle trading actions on various input data and provide numerical results for the sake of comparing different methods on different time series (either FOREX or artificially generated data).

At first, we investigate the estimation performance of the proposed model on generated data. The results showed, that FFNN can successfully predict the forward distribution, furthermore in some case it is more accurate, than the standard histogram method (simply calculating the relative frequencies). *Table 11* in the Appendix shows the corresponding results.

For a detailed comparative analysis, the following performance measures were calculated for each experiments on the corresponding time series:

- Profit gained - the money realized by the agent;
- Maximum drawdown - the maximum loss from a peak to a trough of the balance;
- Number of trades – the number of trades the agent made on the on the evaluation period;
- Winning rate - ratio of the total number of winning trades to the number of all trades;
- Average trade duration – the average holding period of an asset or portfolio in seconds.
- Average profit per trade (in points, which is the smallest possible price change) - the money realized by the agent divided by the number of trades.

In this section, we show the numerical results obtained on the following foreign exchange data sets:

- EUR/USD;

- GBP/USD and
- NZD/USD

tick data on the day of 2015. 05. 27. We only tested on a single asset time series, however the procedures can be used for multiasset portfolios.

In each case the length of memory of neural network was *3* and we used the last 30 minutes of tick data observations from the past to fit FFNNs, while we retrained the network after 3 minutes of tick data. The number of quantization levels was $M=5$ in each simulation.
For the sake of comparison, we used several FFNNs, each with different number of neurons in the hidden layer. The results are shown on the following tables.

6.1. Numerical results obtained in the case of using equidistant quantization

For the sake of comparison, we ran the trading algorithm in the case of using equidistant quantization.

Hidden layer size	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
1	-0.43 %	0.49 %	2	50.0 %	1945	-21.50
3	-2.60 %	7.45 %	37	46.0 %	1262	-7.02
5	-0.46 %	0.54 %	4	50.0 %	976	-11.50
8	2.57 %	7.20 %	216	81.5 %	338	1.19
10	2.61 %	8.02 %	284	81.0 %	245	0.92
20	3.53 %	7.32 %	299	78.9 %	239	1.18
30	2.39 %	7.76 %	236	72.5 %	307	1.01
50	4.95 %	6.42 %	355	78.3 %	202	1.39

Table 1: Trading on the EUR/USD

Hidden layer size	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
1	-1.11 %	1.11 %	2	0.00 %	2655	-55.50
3	-3.16 %	6.37 %	57	78.95 %	1456	-5.54
5	0.27 %	5.37 %	149	81.88 %	518	0.18
8	1.80 %	4.64 %	182	80.77 %	415	0.98
10	-2.43 %	6.42 %	157	88.54 %	530	-1.54
20	2.31 %	4.76 %	246	73.58 %	307	0.94
30	2.72 %	5.76 %	253	68.77 %	299	1.07
50	2.97 %	5.97 %	368	64.95 %	204	0.80

Table 2: Trading on the GBP/USD

Hidden layer size	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
1	0.98 %	1.52 %	10	40 %	2018	9.80
3	0.55 %	3.03 %	94	55 %	724	0.58
5	1.23 %	3.29 %	278	69 %	215	0.44

8	0.75 %	4.98 %	400	70 %	156	0.18
10	3.16 %	4.58 %	552	72 %	110	0.57
20	5.63 %	2.90 %	504	75 %	113	1.11
30	6.15 %	3.23 %	607	74 %	91	1.01
50	7.59 %	3.40 %	747	72 %	75	1.02

Table 3: Trading on the NZD/USD

One must note, that the winning ratio (on mid prices) was very high (75 % or more), which means the FFNN can predict the future prices.

To calculate profitability of the trading algorithm in real circumstances, one must taking into account the spread. We measured the average spread of the 3 assets on the testing period:

- EUR/USD – 5.49 points;
- GBP/USD – 9.55 points;
- NZD/USD – 12.29 points.

Unfortunately, the price movement is not enough to cover the bid-ask spread (the average spread is higher than the average profit per trade).

It can be also seen that by increasing the neurons in the hidden layer, the average trade duration is decreasing and the number of trades is increasing.

6.2. Numerical results obtained in the case of using Lloyd-Max quantization

Next, we applied Lloyd-Max quantization on the return series. The results are shown below.

Hidden layer size	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
1	0.96 %	0.00 %	1	100.0 %	1859	96.00
3	-2.20 %	2.67 %	2	50.0 %	16011	-110.00
5	0.95 %	0.04 %	4	75.0 %	3625	23.75
8	1.00 %	0.72 %	17	58.8 %	767	5.88
10	0.16 %	3.47 %	232	85.3 %	318	0.07
20	3.78 %	4.34 %	214	66.8 %	292	1.76
30	10.09 %	5.04 %	546	76.0 %	103	1.85
50	21.85 %	3.15 %	740	70.5 %	67	2.95

Table 4: Trading on the EUR/USD

Hidden layer size	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
1	0.03 %	0.00 %	2	100.0 %	0.11	1.50
3	1.50 %	0.14 %	42	81.0 %	159	3.57
5	-7.25 %	9.07 %	193	66.8 %	288	-3.75
8	-0.52 %	3.55 %	255	83.1 %	268	-0.20
10	-1.05 %	4.31 %	304	84.5 %	225	-0.34
20	0.93 %	7.83 %	1017	70.2 %	50	0.09
30	8.27 %	5.05 %	1248	72.8 %	33	0.66
50	6.72 %	7.16 %	1472	69.2 %	31	0.45

Table 5: Trading on the GBP/USD

Hidden layer size	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
1	0.00 %	0.00 %	0	0.0 %	0	0.00
3	-0.72 %	1.72 %	19	63.2 %	139	-3.79
5	4.57 %	2.18 %	500	81.0 %	111	0.91
8	12.05 %	1.75 %	695	87.1 %	70	1.73
10	16.43 %	1.96 %	1166	78.1 %	31	1.41
20	17.03 %	1.60 %	1462	73.3 %	25	1.16
30	17.41 %	3.08 %	1781	77.5 %	22	0.97
50	17.32 %	3.20 %	1869	72.4 %	22	0.93

Table 6: Trading on the NZD/USD

One can see, that the profit generated by Lloyd-Max optimizer trading algorithm on each time series is increased compared to the profits achieved by equidistant quantization. On the other hand, the maximal drawdown is decreased, which is also favourable. Unfortunately, the average profit per trade did not increased, which is essential for beating the bid-ask spread and transaction costs.

7. Further optimizing the trading performance

The next numerical results are obtained by improving the trading with the methods listed in Section 5.

First, we examined the effect of the filter specified in Section 5.1. on EUR/USD return series. We set η from 0.5E-6 to 4E-6 to discover the best parameter setting. The results are shown below.

Minimal standard deviation - η (E-6)	Profit	Maximum Drawdown	Number of trades	Winning rate	Average trade duration	Average profit in points
0.5	8.19 %	2.08 %	842	63 %	48.43	0.972
1	6.66 %	2.12 %	570	68.4 %	47.32	1.168
1.25	7.78 %	1.69 %	465	74.6 %	56.44	1.673
1.5	8.98 %	0.91 %	390	70.8 %	24.5	2.302
1.75	7.02 %	0.91 %	257	73.9 %	43.6	2.731
2	5.90 %	0.41 %	224	75.4 %	42.48	2.633
2.25	**4.71 %**	**0.22 %**	**164**	**72.6 %**	**55.17**	**2.872**
2.5	1.00 %	0.22 %	122	63.9 %	2.98	0.819
3	0.11 %	0.22 %	31	71.0 %	6.84	0.354
4	0.11 %	0.13 %	7	71.4 %	6.52	1.571

Table 7: Trading on the EUR/USD

One can see, that the predicted standard deviation has a high impact on the trading results. To ensure profitability, one must beat the bid-ask spread and transaction cost – e.g. the average profits on the trades must be greater, than the average spread. By setting the filter to 2.25E-6 on EUR/USD, we can secure almost 3 points per trade.

7.1. Optimizing the meta-parameters

To achieve more profit, we optimized the meta-parameters of the model. The optimization were performed on a 2nd generation Intel Core i7 machine, using 4 parallel threads at a time. During 2 hours, the optimizer run the strategy for approximately 100 different states. The following tables show the 10 best results for the three FOREX time series.

Case	Profit	Max drawdown	Average profit in points	Number of trades	Hit ratio
1	5.35 %	1.33 %	44.6	12	66.67 %
2	5.31 %	1.34 %	35.4	15	66.67 %
3	5.26 %	1.34 %	35.1	15	60.00 %
4	6.55 %	0.22 %	29.7	22	63.64 %
5	6.78 %	0.18 %	27.1	25	72.00 %
6	1.97 %	1.77 %	19.7	10	60.00 %
7	6.31 %	0.31 %	14.2	45	46.67 %
8	3.48 %	1.50 %	13.2	25	64.00 %
9	7.11 %	0.20 %	12.5	59	69.49 %
10	4.86 %	1.65 %	10.4	47	42.55 %

Table 8: Trading on the EUR/USD

Case	Profit	Max drawdown	Average profit in points	Number of trades	Hit ratio
1	1.93 %	0.24 %	9.2	21	71.43 %
2	0.84 %	0.18 %	7.0	12	75.00 %
3	1.29 %	0.33 %	2.8	45	77.78 %
4	1.54 %	0.36 %	2.8	55	80.00 %
5	0.33 %	0.05 %	2.7	12	75.00 %
6	2.21 %	0.35 %	2.6	83	81.93 %
7	0.34 %	1.79 %	2.4	14	42.86 %
8	1.60 %	0.78 %	2.3	68	61.76 %
9	1.56 %	0.35 %	2.2	70	65.71 %
10	6.08 %	0.37 %	2.1	282	67.73 %

Table 9: Trading on the GBP/USD

Case	Profit	Max drawdown	Average profit in points	Number of trades	Hit ratio
1	1.01 %	4.98 %	5.61	18	44.44 %
2	1.64 %	0.16 %	4.56	36	66.67 %
3	5.67 %	1.96 %	4.2	135	57.78 %
4	4.31 %	1.76 %	3.88	111	81.08 %
5	4.91 %	1.10 %	2.7	182	56.04 %
6	4.86 %	2.36 %	1.79	271	81.18 %
7	9.78 %	1.83 %	1.77	553	75.95 %
8	3.89 %	0.76 %	1.66	234	53.85 %

9	14.46 %	1.80 %	1.54	939	74.33 %
10	11.81 %	1.58 %	1.51	782	72.25 %

Table 10: Trading on the NZD/USD

Note that in each of these cases we could secure positive profit on the mid prices. The largest increase of average profit was attained on the EUR/USD rate, which may, in this case, will pave the way towards profitable trading on bid-ask price series also. In the Appendix, the parameters of the ten cases are given. While, on the other two assets, the strategy does not have enough prediction power to yield profitable trading.

Conclusions

As the numerical results demonstrated, the new methods were able to yield profit on the mid-prices of the underlying rates (EUR/USD, GBP/USD, NZD/USD). This profit was even achieving 21.85 % in the best case. However, when the bid-ask spread was also taken into account in the numerical analysis, then positive profit tended to remain only in the case of EUR/USD. As a result, further adjustments needed to beat the bid-ask spread. These further adjustments included:

- applying optimal quantization by Lloyd-Max algorithm;
- trading only in high volatility periods;
- optimizing the meta-parameters by Simulated Annealing.

With these adjustments, we could secure positive profit in the presence of bid-ask spread. In the case of EUR/USD, the profit was more than 5.3 % for the tested one-day period. Nevertheless, these adjustments needed heuristic considerations (such as trying out empirically the size of memory and identifying high volatility periods). There is also a slowdown of the execution of algorithm due to running Simulated Annealing.

References

[1] K. Anagnostopoulos and G. Mamanis, “The mean-variance cardinality constrained portfolio optimization problem: An experimental evaluation of five multiobjective evolutionary algorithms,” *Expert Systems with Applications,* pp. 14208-14217, 2011.

[2] E. P. Chan, Quantitative Trading, Wiley, 2008.

[3] A. D'Aspremont, “Identifying small mean-reverting portfolios,” *Quantitative Finance,* pp. 351-364, 2011.

[4] I. R. Sipos and J. Levendovszky, “Optimizing sparse mean reverting portfolios,” *Algorithmic Finance,* pp. 127-139, 2013.

[5] N. Fogarasi and J. Levendovszky, “Sparse, mean reverting portfolio selection using simulated annealing,” *Algorithmic Finance,* pp. 197-211, 2013.

[6] I. Kaastra and M. Boyd, “Designing a Neural Network for Forecasting Financial and Economic Time Series,” *Neurocomputing,* vol. 10, no. 3, pp. 215-236, 1996.

[7] E. W. Saad, D. V. Prokhorov and D. C. Wunsch, “Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks,” *IEEE Transactions on Neural Networks,* vol. 9, no. 6, pp. 1456-1470, 1998.

[8] J. Levendovszky és F. Kia, „Prediction based – high frequency trading on financial time series,” *Periodica Polytechnica,* %1. kötet1, pp. 29-34, 2012.

[9] J. Levendovszky, I. Reguly, A. Olah and A. Ceffer, "Low complexity algorithmic trading by Feedforward Neural Networks," 2016.

[10] K. Hornik, M. Stinchcombe and H. White, “Multilayer Feedforward Networks are Universal Approximators,” *Neural Networks, VOL. 2,* pp. 359-366, 1989.

[11] S. Haykin, Neural Network Theory: a Comprehensive Foundation, Prentice Hall, 1999.

[12] K. Funahashi, “On the Approximate Realization of Continuous Mappings by Neural Networks,” *Neural Networks, VOL. 2,* pp. 183-192, 1989.

[13] S. Lloyd, “Least squares quantization in PCM,” *IEEE Transactions on Information Theory,* vol. 28, no. 2, pp. 129-137, 1982.

[14] J. Max, “Quantizing for minimum distortion,” *IRE Transactions on Information Theory,* vol. 6, no. 1, pp. 7-12, 1960.

[15] L. Ingber, „Adaptive simulated annealing (ASA): Lessons learned,” *Control and Cybernetics,* %1. kötet25, %1. szám1, pp. 33-54, 1996.

Appendix

In the tables below, we give the results in details. *Table 11* shows the performance of predicting the FCPD by FFNN and standard histogram method.

MSE	*L=1*	*L=2*	***L=3***
N=1	0.00296	0.0089	0.00153
N=3	0.00741	0.00867	0.00162
N=5	0.00624	0.00697	0.00157
N=8	0.00624	0.00613	0.00144
N=10	0.00624	0.00486	0.00174
N=20	0.00624	0.00412	0.00178
N=30	0.00624	0.00412	0.00189
N=50	0.00624	0.00412	0.00224
Histogram	0.00624	0.00412	0.00224

Table 11: Validating the prediction model

The next table shows the metaparameters of the ten cases of EUR/USD.

Case	**ε**	**M**	**L**	**N**	**η**	**T_{train}**	**$T_{retrain}$**
1	0.2723	6	1	20	3.50E-06	3.00	3.00
2	0.0710	4	1	30	4.50E-06	3.00	3.00
3	0.3461	4	1	50	3.50E-06	3.00	3.00
4	0.5866	8	2	10	4.00E-06	1.00	3.00
5	0.5866	6	2	10	4.00E-06	1.00	3.00
6	0.4789	8	2	3	2.75E-06	4.00	0.17
7	0.4952	8	3	1	3.50E-06	3.00	0.33
8	0.2624	6	1	50	2.50E-06	4.00	1.00
9	0.5866	8	3	10	4.00E-06	1.00	3.00
10	0.0710	4	1	30	2.75E-06	3.00	3.00

Table 12: The metaparameters of the ten cases on EUR/USD