FOREX Trading Model Based on Forecast Aggregation and Metaheuristic Optimization

Michail G. Papatsimpas
Department of Computer Science and Engineering
University of Ioannina
GR-45110 Ioannina, Greece
mgpapatsimpas@gmail.com

Ioannis Lykogiorgos
Municipality of Ioannina
Sector of E-governance & Smart City
GR-45110 Ioannina, Greece
ilykogiwrgos@ioannina.gr

Konstantinos E. Parsopoulos
Department of Computer Science and Engineering
University of Ioannina
GR-45110 Ioannina, Greece
kostasp@cse.uoi.gr

ABSTRACT

In the dynamic global economy, accuracy in forecasting the foreign currency exchange rates or at least predicting the trend is of crucial importance for successful investments. The use of computational intelligence methods for forecasting has proved to be extremely successful in recent times. The present work introduces a FOREX trading model based on moving average forecast aggregation and metaheuristic optimization. The model harnesses three moving average forecasters optimized by the particle swarm optimization algorithm. Also, it is equipped with a set of trading rules based on technical analysis. Simulation results using real world data are reported for the EUR/USD currency pair. The results show that the proposed trading model can be highly competitive in terms of trading performance against its constituent moving average models.

CCS CONCEPTS

• **Computing methodologies → Randomized search**; **Discrete space search**; **Continuous space search**; • **Applied computing → Forecasting**.

KEYWORDS

Metaheuristics, FOREX, forecast aggregation, optimization.

ACM Reference Format:
Michail G. Papatsimpas, Ioannis Lykogiorgos, and Konstantinos E. Parsopoulos. 2020. FOREX Trading Model Based on Forecast Aggregation and Metaheuristic Optimization. In *11th Hellenic Conference on Artificial Intelligence (SETN 2020), September 2–4, 2020, Athens, Greece.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3411408.3411415

1 INTRODUCTION

The foreign exchange market (widely denoted as FOREX or FX) is the biggest as well as among the most liquid markets in the world. It has always been a challenging market as far as short-term prediction is concerned. Since the breakdown of the Bretton Woods system of fixed exchange rates in 1971-1973 [3] and the implementation of the floating exchange rate system, researchers have been motivated to explain the movements of exchange rates. Among the available currency pairs, the EUR/USD (i.e., Euro vs US Dollar) is the most actively traded one [4].

In the FOREX market, participants can buy, sell, exchange, and speculate on currencies. It is made up of banks, commercial companies, central banks, investment management firms, hedge funds, retail brokers and investors. Forecasting exchange rates is vital for fund managers, borrowers, corporate treasurers, and specialized traders. Nevertheless, it is often difficult to identify a forecasting model because the underlying laws may not be clearly understood.

According to the triennial central bank survey conducted by the Bank for International Settlements (BIS) [1], trading in FOREX markets reached \$6.6 trillions per day in April 2019. The FOREX time series are often characterized by nonlinearity that cannot be satisfactorily handled through traditional linear forecasting techniques [17]. Consequently, forecasting quality becomes questionable. Such drawbacks are responsible for the small percentage (about one third) of the foreign exchange dealers that make an annual profit. Therefore, there is an increasing necessity for efficient trading models and decision-making tools that can support the traders toward more informative decisions. [6, 10, 14]

The present paper proposes a FOREX trading model based on forecast aggregation and metaheuristic optimization. The main idea is to harness the forecasting capability of various forecasters, while using metaheuristics to optimize the parameters and stochastic decisions of the produced trading model. The model embraces three established moving average technical indicators, optimized by the popular particle swarm optimization metaheuristic. Also, it is equipped with a set of training rules rooted in technical analysis. The proposed model is assessed on real world data for the EUR/USD currency pair. Simulation results reveal that the proposed trading model outperforms its constituent moving average models in terms of overall trading performance.

The rest of the paper is organized as follows: Section 2 offers the necessary background information regarding the moving average forecasting techniques and particle swarm optimization. The proposed approach is presented and analyzed in Section 3, while details regarding the specific application on the EUR/USD currency pair are provided in Section 4. Experimental assessment is reported in Section 5, and the paper concludes in Section 6.

2 BACKGROUND INFORMATION

The following paragraphs offer brief descriptions of the moving average forecasting techniques as well as the particle swarm optimization metaheuristic that will be later used for the composition of the proposed FOREX forecasting model.

2.1 Moving Average Forecasting

Among the various technical indicators based on price information, moving averages lie among the oldest, most popular, and most useful ones. Moving averages have been widely used for the identification of price trend in foreign exchange rates and security analysis [13, 26]. They have been widely used for solving time-series smoothing problems [21, 22], as well as in various problems in economics and statistics [11, 23].

Putting it formally, let Y be a time series of length t:

$$Y = \{y_1, y_2, y_3, \ldots, y_t\}.$$

The three moving averages of our interest predict the next value, y_{t+1}, of the time series as follows:

(1) *Simple moving average (SMA)*: It calculates the arithmetic mean over the past m observations. The value of y_{t+1} is estimated as follows:

$$\hat{y}_{t+1} = \frac{y_t + y_{t-1} + \cdots + y_{t-m+1}}{m}. \tag{1}$$

The random walk model is obtained as a special case for $m = 1$.

(2) *Simple exponential smoothing (SES)*: It aggregates the previous estimation, $\hat{y}_t$, and actual value, y_t, of the time series as follows:

$$\hat{y}_{t+1} = a y_t + (1-a)\hat{y}_t, \tag{2}$$

where $a \in [0,1]$ is the relevant weight parameter. Obviously, the random walk model is obtained for $a = 1$, while a constant-forecast model is obtained for $a = 0$.

(3) *Linear exponential smoothing (LES)*: This model assumes that trend appears in the data. Exponential smoothing with a trend works much like simple smoothing except that two components must be updated at each period, namely the level L_t and the trend T_t of the time series. The level is a smoothed estimate of the value at the end of each period. The trend is a smoothed estimate of average growth at the end of each period. When the actual value is observed, the updated estimate of the level is recursively computed by interpolating between y_t and its forecast, using a weight parameter $a \in [0,1]$:

$$L_t = a y_t + (1-a)(L_{t-1} + T_{t-1}). \tag{3}$$

The difference $(L_t - L_{t-1})$ can be interpreted as a noisy measurement of the trend. The updated estimate of the trend is computed by interpolating between this difference and the previous estimate of the trend, T_{t-1}, using a weight parameter $b \in [0,1]$:

$$T_t = b(L_t - L_{t-1}) + (1-b)T_{t-1}. \tag{4}$$

Finally, the desirable forecast is obtained by aggregating the updated level and trend:

$$\hat{y}_{t+1} = L_t + T_t. \tag{5}$$

Small values of b correspond to slowly changing trends over time, while rapidly changing trends are assumed for larger values. The presented method is also called the Holt's linear exponential smoothing method. The simpler Brown's linear exponential smoothing method uses a single smoothing constant for both L_t and T_t.

The presented moving average models have been widely used in time series forecasting, and they are implemented in numerous software packages [24, 25].

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population-based search algorithm inspired by the intrinsic laws that dictate the swarming behavior of particles. It was introduced in the pioneering work of Eberhart and Kennedy in 1995 [18] and, since then, it has gained increasing popularity [2, 8, 15].

Let the n-dimensional bound-constrained optimization problem:

$$\min_{x \in \mathcal{X} \subset \mathbb{R}^n} f(x). \tag{6}$$

The algorithm employs a set of search points:

$$S = \{x_1, x_2, \ldots, x_N\},$$

to probe the search space. This set is called the swarm, and each search point is called a particle and it is defined as:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^T \in \mathcal{X}, \quad i \in I \equiv \{1, 2, \ldots, N\}.$$

Each particle explores the search space by iteratively moving to new positions in $\mathcal{X}$, adjusting its exploratory behavior according to its own findings as well as the findings of the other particles. Its motion is determined by an adaptable position shift, also called the velocity, defined as:

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^T, \quad i \in I,$$

while it retains in memory the best position it has ever visited:

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{in})^T \in \mathcal{X}.$$

The update equations of the particles are given, componentwisely, as follows:

$$v_{ij}(t+1) = \chi \left[v_{ij}(t) + c_1 r_1 \left(p_{ij}(t) - x_{ij}(t) \right) + c_2 r_2 \left(p_{g_i j}(t) - x_{ij}(t) \right) \right],$$
$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1),$$

where $i \in \mathcal{I}$, $j = 1, 2, \ldots, n$, and g_i is the index of the particle that attained the best previous position among all the particles in a predefined neighborhood of the i-th particle. The parameter $\chi > 0$, also called the constriction coefficient, induces convergence properties in the algorithm by restricting the velocities. Also, c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components, respectively. The quantities $r_1, r_2 \sim \mathcal{U}([0,1])$ are real-valued random variables that introduce stochasticity to the algorithm.

After all particles have been updated and evaluated, their best positions are also updated as follows:

$$p_i(t+1) = \begin{cases} x_i(t+1), & \text{if } f\left(x_i(t+1)\right) < f\left(p_i(t)\right), \\ p_i(t), & \text{otherwise.} \end{cases}$$

The reader is referred to [15] for further details on implementations, parameter setting, and applications of the algorithm.

3 PROPOSED MODEL

The proposed FOREX trading model combines a special forecasting procedure and a trading strategy. Both are thoroughly presented in the following paragraphs.

3.1 Forecast Aggregation

The proposed model employs the forecasting techniques described in the previous section, namely simple moving average, simple exponential smoothing, and linear exponential smoothing. The latter is used twice under different parameter configuration. Each one of these techniques produces a forecast for the next day. These forecasts are then aggregated in order to produce a new forecast that is exploited by the trading model.

Let the time series:

$$Y = \{y_1, y_2, \ldots, y_t\},$$

consisting of t observations, and a set of k forecasters:

$$M = \{M_1, M_2, \ldots, M_k\},$$

each one predicting the value y_{t+1}. We assume a weight vector:

$$W = (w_1, w_2, \ldots, w_k),$$

where:

$$w_i \in [-1, 1], \quad i \in K \equiv \{1, 2, \ldots, k\},$$

is the weight associated to forecaster M_i. Note that negative weight values are considered in order to penalize erroneous forecasts.

Then, the aggregate 1-step ahead prediction is calculated as follows:

$$\hat{y}_{t+1} = \sum_{i=1}^{k} w_i \, \hat{y}_{t+1}^{(i)}, \tag{7}$$

where w_i is the weight of forecaster M_i, and $\hat{y}_{t+1}^{(i)}$ is its corresponding prediction at time $t+1$. The produced aggregate forecast is heavily affected by the forecasters' weights, which can result in either poor or high predictive performance. Ideal weights shall produce accurate forecast for the unknown observation y_{t+1}. This requirement implies that:

(a) The weight vector shall be adapted from one prediction to another.
(b) A promising weight vector can be estimated by considering its forecast performance on previous observations of the time series.

The main question in this procedure concerns the number of previous observations (history) that shall be taken into consideration.

Thus, the first step in our proposed model is to find a weight vector that results in forecasts of minimum error for a number of past observations. This is a k-dimensional continuous optimization problem over the set of possible weight vectors. To this end, we set a sliding window of past observations that is used in the optimization procedure. Regarding the underlying objective function, various criteria such as root mean squared error, maximum absolute error, or mean absolute error can be used. The maximum absolute error is adopted because it has the property of minimizing the worst forecast in the whole time window.

Let m be the selected window size, and let:

$$A = \{y_{t-m+1}, y_{t-m+2}, \ldots, y_t\} \subseteq Y,$$

be the sliding window (past observations) that is used to predict y_{t+1}. Then, for each observation $y_i \in A$, we compute a forecast value $\hat{y}_i^{(j)}$, $j \in K$, with each one of the available forecasters. Assuming a weight vector $W = (w_1, w_2, \ldots, w_k)$, the forecast values are aggregated according to Eq. (7), producing the aggregate forecast:

$$\hat{y}_i = \sum_{j=1}^{k} w_j \, \hat{y}_i^{(j)}.$$

Thus, for the whole set A, we receive a set of aggregate forecasts:

$$F = \{\hat{y}_{t-m+1}, \hat{y}_{t-m+2}, \ldots, \hat{y}_t\},$$

each one having absolute error:

$$\varepsilon_i = |\hat{y}_i - y_i|, \quad i = t-m+1, \ldots, t.$$

Obviously, these errors are tightly related to the employed weight vector W. Fixing the size m of the window, the weight vector can be optimized in terms of the selected optimization criterion. Assuming that this criterion is the minimization of the maximum absolute error, the weight vector W can be computed as the solution of the following min-max problem:

$$\min_{W} \; E(W) \triangleq \max\{\varepsilon_{t-m+1}, \varepsilon_{t-m+2}, \ldots, \varepsilon_t\}. \tag{8}$$

In simple words, the optimization algorithm tries to find the specific weight vector that minimizes the selected optimization criterion (maximum absolute error) for the selected window A of past observations. We should also mention that the selection of maximum absolute error as the optimization criterion was made retrospectively, and it was based on experimental basis. The pseudocode of computing $E(W)$ for a given weight vector W is provided in Algorithm 1.

Algorithm 1: Pseudocode for the evaluation of $E(W)$ for a given weight vector W.

```
Data: Weight vector, $W = (w_1, \ldots, w_k)$; time series,
      $Y = \{y_1, y_2, \ldots, y_t\}$; set of forecasters,
      $M = \{M_1, M_2, \ldots, M_k\}$; window size, $m$.
Result: Objective value, $E(W)$.
/* individual forecaster predictions */
for $i = t-m+1 \ldots t$ do
    for $j = 1 \ldots k$ do
        $\hat{y}_i^{(j)} \leftarrow M_j(y_1, y_2, \ldots, y_{i-1})$
    end
end
/* aggregate predictions and errors */
for $i = t-m+1 \ldots t$ do
    $\hat{y}_i \leftarrow 0$
    for $j = 1 \ldots k$ do
        $\hat{y}_i \leftarrow \hat{y}_i + w_j \, \hat{y}_i^{(j)}$
    end
    $\varepsilon_i \leftarrow |\hat{y}_i - y_i|$
end
/* objective value */
$E(W) \leftarrow \max\{\varepsilon_{t-m+1}, \varepsilon_{t-m+2}, \ldots, \varepsilon_t\}$
Return $E(W)$
```

Algorithm 2: Pseudocode of weight vector optimization using particle swarm optimization.

```
Data: Swarm size, $N$; PSO parameters; maximum iterations, $\tau_{\max}$.
Result: Optimal weight vector, $W^*$.
/* initialize swarm of weight vectors */
$\tau \leftarrow 0$
for $i = 1 \ldots N$ do
    $W_i^{(\tau)} \leftarrow \mathcal{U}\left([-1,1]^k\right)$
    $f_i^{(\tau)} \leftarrow E\left(W_i^{(\tau)}\right)$    [ use Algorithm 1 ]
end
$S^{(\tau)} \leftarrow \left\{W_1^{(\tau)}, \ldots, W_N^{(\tau)}\right\}$    // swarm
$P^{(\tau)} \leftarrow S^{(\tau)}$    // best positions
$g^* \leftarrow \arg\min_{i=1\ldots N} \left\{f\left(P_i^{(\tau)}\right)\right\}$
/* evolve swarm */
while $\tau < \tau_{\max}$ do
    $\tau \leftarrow \tau + 1$
    $S^{(\tau)} \leftarrow$ update-swarm $\left(S^{(\tau-1)}\right)$
    for $i = 1 \ldots N$ do
        $f_i^{(\tau)} \leftarrow E\left(W_i^{(\tau)}\right)$    [ use Algorithm 1 ]
    end
    $P^{(\tau)} \leftarrow$ update-best-positions $\left(S^{(\tau)}, P^{(\tau-1)}\right)$
    $g^* \leftarrow \arg\min_{i=1\ldots N} \left\{f\left(P_i^{(\tau)}\right)\right\}$
end
$W^* \leftarrow P_{g^*}^{(\tau)}$
Return $W^*$
```

Let the obtained optimal weight vector be:

$$W^* = \left(w_1^*, \ldots, w_k^*\right).$$

Then, the aggregate forecast for y_{t+1} is eventually computed as:

$$\hat{y}_{t+1} = \sum_{j=1}^{k} w_j^* \hat{y}_{t+1}^{(j)}, \tag{9}$$

where $\hat{y}_{t+1}^{(j)}$ is the forecast value of y_{t+1} offered by the j-th forecaster M_j. Thus, our model aggregates the predictions of the k forecasters using weights that provide the best possible aggregate predictions in the past m moves.

When the actual value y_{t+1} is available, the model proceeds to the next predicted value $\hat{y}_{t+2}$ by computing a new optimal weight vector, following the same procedure as above. Therefore, our forecasting model essentially constitutes a dynamic optimization scheme.

So far we considered fixed size m of the sliding window. In the trivial case, the user may simply use all the past observations, i.e., $m = t$. However, this choice may be inferior to smaller window sizes because the time series may have radically changed (in level, trend, etc) throughout the whole set of observations. Thus, smaller windows that can locally capture the dynamic of the time series may be preferable.

Alternatively, the sliding window can have adaptable size. One way to achieve it is by incorporating m as an additional variable in the weight vector and let it be determined by the search procedure that is used to solve the optimization problem of Eq. (8). Note that this is a mixed integer problem and may require special algorithms for its solution.

Taking into consideration all the above, particle swarm optimization is the selected solver in our model. This choice is driven by its efficiency, minor implementation effort, and its straightforward applicability in mixed integer problems [15]. More specifically, the algorithm initializes a swarm of N weight vectors and evolve them in order to minimize the maximum absolute error in the selected time window (or determines also the window size itself). The outcome is the optimal weight vector that offers the final prediction according to Eq. (9). Pseudocode of the application of PSO for the detection of optimal weight vector is given in Algorithm 2.

3.2 Trading Strategy

Although a reliable forecasting method is essential, it is not adequate to build a successful trading system. In fact, trading profit requires also good trading strategies that can take full advantage of forecasting methods.

Technical analysis offers estimation of stock price values using historical price patterns and trading volume [9]. The study in [12] justified for the first time the importance of technical analysis in decision-making in FOREX markets. Today, it is almost universally used by practitioners in formulating short-term exchange rate expectations [16].

For this reason, we equipped our FOREX trading model with a trading strategy based on technical analysis indices. The indices have been tailored for the specific EUR/USD pair of our interest. In a currency pair, the numerator and the denominator currencies are called the base and the quote currency, respectively. Specifically, an exchange rate represents how much of the quote currency is needed in order to get one unit of the base currency. Trading signals are generated by buying (resp. selling) and selling (resp. buying) the base and quote currency.

Thus, at each step, our model produces one of the following three trading signals:

(a) *Buy signal*: In this case, prediction suggests that, at the next step, the base currency will increase in value against the quote currency. Thus, we "open" a buy position by buying the base and selling the quote currency at time t.
(b) *Sell signal*: The sell signal is the exact opposite of a buy signal. In this case, the model predicted that, in the next step, the base currency will decrease in value against the quote currency. Therefore, we "open" a sell position by selling the base and buying the quote currency at time t.
(c) *Neutral signal*: The neutral signal rarely occurs because it expects that in the next step the price will remain constant.

These signals determine the trader's movements, i.e., positions in the market, which are defined as follows:

(a) *Buy position*: We open a buy position when our model produces a buy signal.

(b) *Sell position*: We open a sell position when our model produces a sell signal.
(c) *Hold position*: We have a hold position when the same signal as the one in the previous step appears. For instance, if we get a buy signal at both times $t-1$ and t, then we keep open the already opened buy position and do not open a second one.
(d) *Neutral position*: When a neutral signal appears, we stay neutral and we do not open a position.

Taking into account the above trading signals, we derived three trading rules. If the actual value of the exchange rate at time t is lower than the obtained forecast value at time $t+1$, we have a buy signal. On the other hand, if the actual value of the exchange rate at time t is greater than the forecast value at $t+1$, we have a sell signal. Finally, if the actual value of the exchange rate at time t is equal to the forecast value at $t+1$, we have a neutral signal. The trading rules are summarized as follows:

(R1) **IF** $Actual_t < Forecast_{t+1}$ **THEN** $Signal_t \longleftarrow$ *Buy.*
(R2) **IF** $Actual_t > Forecast_{t+1}$ **THEN** $Signal_t \longleftarrow$ *Sell.*
(R3) **IF** $Actual_t = Forecast_{t+1}$ **THEN** $Signal_t \longleftarrow$ *Neutral.*

However, before a position is opened we must check if it is a hold position:

(R4) **IF** $Signal_{t-1} = Signal_t$ **THEN** $Position_t \longleftarrow$ *Hold.*

Finally, our model selects one of the above rules and, if there is no hold position, it opens the corresponding position. The aforementioned trading strategy is applied to all the implemented trading models.

3.3 Profit Calculation

The profit is measured in pips. Pip expresses the smallest change in value between two currencies. In most currencies, it is the fourth decimal point (1 pip = 0.0001 of a cent), thus the result is multiplied by 10000. As we previously mentioned in Section 3.2, a position opens when a buy (resp. sell) signal is encountered, and closes when a sell (resp. buy) signal occurs. In the case of a hold position, the profit remains constant.

Thus, the total profit of the model is calculated as follows:

(a) If a buy position was opened at time t_{open} and the current signal at time t_{now} is sell, the profit is:
$$\text{profit}_{\text{now}} = 10000\,(Actual_{\text{now}} - Actual_{\text{open}}).$$
(b) If a buy position was opened at time t_{open} and the current signal at time t_{now} is buy, the profit is:
$$\text{profit}_{\text{now}} = \text{profit}_{\text{open}}.$$
(c) If a sell position was opened at time t_{open} and the current signal at time t_{now} is buy, the profit is:
$$\text{profit}_{\text{now}} = 10000\,(Actual_{\text{open}} - Actual_{\text{now}}).$$
(d) If a sell position was opened at time t_{open} and the current signal at time t_{now} is sell, the profit is:
$$\text{profit}_{\text{now}} = \text{profit}_{\text{open}}.$$

These profit calculation rules were used in our experimental analysis in order to assess the trading quality of the proposed model.

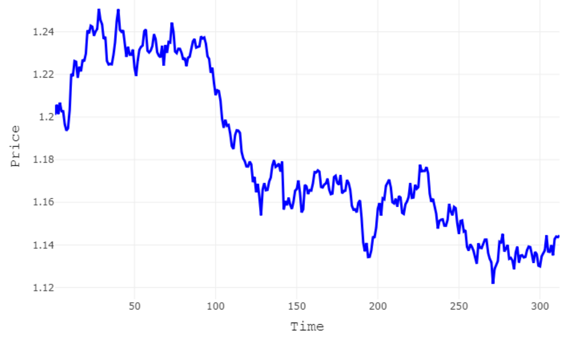

Figure 1: EUR/USD closing price from December 31, 2017, to December 30, 2018.

4 APPLICATION DETAILS ON EUR/USD CURRENCY PAIR

Our main experimental goal was to investigate whether the proposed model is profitable or not. For this purpose, we considered the daily closing price of the EUR/USD time series (obtained by the FXCM broker [5]), which is depicted in Fig. 1. The selected time period is from December 31, 2017, to December 30, 2018, consisting of 312 observations (the market is closed on weekends). We started our research in the FOREX market in 2019, thus we selected the previous year as the investigating time series. In order to assess our model's performance, we set a specific window length of known observations of the time series and then applied our trading model for the rest of the time series. At the end, we evaluated the overall profitability of the model, assuming that no transaction fees incur per trade.

For presentation purpose, let the window size be equal to half of the dataset, i.e., 156 observations (this is one of the studied cases in our experimental setting). Then, we created all subsequent subsets of observations of length 156. For example, subset sub_1 starts from day_1, and ends at day_{156}, subset sub_2 starts from day_2, and ends at day_{157}, etc. This is dictated by the different optimal weight vector generated by the PSO algorithm in every run, which results in new forecasts and, eventually, new trading signals.

The subsets have the following form:

$$\begin{array}{lcl} sub_1 & = & \{cp_1, cp_2, \ldots, cp_{156}\}, \\ sub_2 & = & \{cp_2, cp_3, \ldots, cp_{157}\}, \\ sub_3 & = & \{cp_3, cp_4, \ldots, cp_{158}\}, \\ \vdots & & \vdots \\ sub_{156} & = & \{cp_{156}, cp_{157}, \ldots, cp_{311}\}, \end{array}$$

where cp is the daily closing price. Applying these subsets as inputs to our model, we receive a vector of forecasts regarding the second half of the original dataset. For instance, sub_1 produces a forecast for the closing price of day_{157}, sub_2 produces a forecast for the closing price of day_{158}, etc. The complete forecast vector has the following form:

$$F = \left(\hat{cp}_{157}, \hat{cp}_{158}, \ldots, \hat{cp}_{312}\right).$$

The trading performance of all the investigated models (the proposed one as well as each moving average technique, individually) is measured by the following factors:

(a) *Trades*: The total number of trades 'opened' during the simulation period.
(b) *Winning trades*: This is the number of trades in which, the trader made the right decision. If divided with the total number of trades, it gives the winning rate. The higher the winning rate the better is a trading system.
(c) *Profit/Loss ratio*: The profit/loss ratio (P/L Ratio) measures how well a trading system is performing. Obviously, the higher the ratio the better the system is. For example, if a system had a winning average of $900 per trade and an average loss over the same time of $300 per trade, then the profit/loss ratio would be 3:1.
(d) *Profit*: Traders often use pips in order to refer to gains or losses. The pips are then converted to the base currency's value. Suppose a trader made a profit from a buy position equal to 30 pips. Let the trader's account balance be 350000 EUR. The conversion of profit from pips to EUR is as follows:
Value of pip in USD: $350000 * 0.0001 = 35$ USD
Value of pip in EUR: $35/1.1130 = 31.45$ EUR
Trade Profit: $30 * 31.45 = 943.4$ EUR
(f) *Maximum DrowDown*: Maximum drowdown (MDD) is a specific measure that looks for the greatest movement from a high point to a low point, before a new peak is achieved. It is measured in pips and it takes only negative values. It shows the maximum loss the trade has reached while it was open.

Besides overall profitability, the predictive performance of the investigated models were measured according to the following metrics:

(a) Root mean squared error:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

(b) Mean absolute error:

$$MAE = \sum_{i=1}^{n} \frac{|\hat{y}_i - y_i|}{n}$$

(c) Mean absolute percentage error:

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\hat{y}_i - y_i}{y_i} \right|$$

All these criteria were considered in our experimental analysis.

5 EXPERIMENTAL RESULTS

The experimental analysis of our proposed model included comparisons with the three types of moving averages discussed in Section 2.1. The main goal was to investigate the trading performance of the proposed model against the rest, as well as to study the forecasting accuracy of the models.

The simple moving average (denoted as SMA) is conceptually very easy to understand. The simple and linear exponential smoothing (denoted as SES and LES, respectively) are easy to implement and they emphasize on the most recent data. The forecasters' parameters were configured following suggestions in relevant literature [7, 19, 20]:

(a) SMA: number of past observations, $m = 3$.
(b) SES: weight parameter, $a = 0.5$.
(c) LES0: weight parameters, $a = 0.5$, $b = 0.1$.
(d) LES1: weight parameters, $a = 0.8$, $b = 0.01$.

The employed PSO parameters were set based on preliminary experimentation results:

(a) Swarm size: 50
(b) Maximum iterations: 50000
(c) Neighborhood radius r: 10 (lbest PSO), 50 (gbest PSO)
(d) $\chi = 0.729$, $c_1 = c_2 = 2.05$

Moreover, the following parameters were used in our forecast aggregation model:

(a) Sliding window size m: 10, 50, 156
(b) Criterion of prediction quality: (1) MAE, (2) RMSE, (3) MAPE

The different parameter combinations lead to the following variants of the proposed model:

(a) PM1: $r = 10, m = 10$, crit=1
(b) PM2: $r = 10, m = 156$, crit=1
(c) PM3: $r = 50, m = 50$, crit=1
(d) PM4: $r = 50, m = 156$, crit=1
(e) PM5: $r = 10, m = 10$, crit=2
(f) PM6: $r = 10, m = 156$, crit=2
(g) PM7: $r = 50, m = 50$, crit=2
(h) PM8: $r = 50, m = 156$, crit=2
(i) PM9: $r = 10, m = 10$, crit=3
(j) PM10: $r = 10, m = 156$, crit=3
(k) PM11: $r = 50, m = 50$, crit=3
(l) PM12: $r = 50, m = 156$, crit=3

Table 1 reports the trading performance of the proposed model and the competing moving averages, following the notation above. Table 2 reports the forecasting accuracy of the models. In addition, Figs. 2–6 illustrate the actual EUR/USD time series as well as the predicted one for the different forecasting models, and Figs. 7–11 illustrate their absolute price differences, respectively.

The results offer some interesting observations. Firstly, Table 2 reveals that the proposed model does not always achieve the optimal forecasting performance. In fact, some of the standard moving average methods achieve better forecasting values in different metrics. Nevertheless, PM2 achieved the best predictive performance among all the considered models and for all performance metrics.

Although forecasting quality is an essential part of a trading system, profitability is of utmost importance. In terms of trading performance, Table 1 reveals that the moving averages are habitually outperformed by the proposed model. Among the different variants, PM12 is distinguished for its performance. A close inspection reveals that only the SMA model yields slightly higher profit than PM12. However, taking into account the crucial P/L ratio, the PM12 variant outperforms all other models, including PM2 that had the best predictive performance and SMA that returned the highest profit. P/L ratio is important because it gives the average profit over the average loss. The P/L ratio for SMA is approximately 0.56, which means that the average profit is 0.56 times the average loss.

Model	Trades	Winning Trades	Winning Rate (%)	P/L Ratio	Profit	MDD
SMA	62	46	74.19	0.5587832	790.1	-273.5
SES	52	38	73	0.5861233	574.9	-273.5
LES0	53	37	67	0.5833109	493.6	-283.2
LES1	79	37	46	1.0374730	-116.3	-277.8
PM1	51	23	41	0.9656548	-242.3	-176.9
PM2	71	43	60	0.7395230	250.8	-285.1
PM3	51	28	50.9	0.6372584	-251.9	-333.6
PM4	73	43	58.9	0.7562141	186.8	-285.1
PM5	43	21	48.84	0.6657352	-382.8	-241.4
PM6	59	37	62.71	0.6284347	131.2	-356.9
PM7	45	24	53.3	0.4274727	-655.4	-315.2
PM8	59	37	62.71	0.6665664	206.8	-356.9
PM9	50	28	56	0.7687286	32.8	-252
PM10	23	14	60.87	1.2082440	451.9	-362.6
PM11	47	24	51.1	0.8333815	-110	-230.3
PM12	**22**	**16**	**72.8**	**2.469614**	**737.3**	**-183.5**

Table 1: Trading performance of the proposed and competing models.

Model	RMSE	MAE	MAPE
SMA	0.005181080	0.004144637	0.003601229
SES	0.004835647	0.003849293	0.003345160
LES0	0.005014031	0.003986668	0.003464697
LES1	0.004797997	0.003897429	0.003385538
PM1	0.005019186	0.004039309	0.003509919
PM2	0.004474617	0.003437553	0.002988580
PM3	0.004515067	0.003556731	0.003091672
PM4	0.004487645	0.003449729	0.002999111
PM5	0.004877512	0.003829868	0.003328356
PM6	0.004479707	0.003486007	0.003029919
PM7	0.004555266	0.003592481	0.003122309
PM8	0.005121730	0.003730936	0.003239469
PM9	0.005525165	0.004502554	0.003911257
PM10	0.006330544	0.005089500	0.004414105
PM11	0.005669328	0.004456507	0.003865104
PM12	**0.006248751**	**0.005031460**	**0.004362498**

Table 2: Forecasting accuracy of the forecasting models under different metrics.

On the other hand, in our model the P/L ratio is 2.46, which indicates that the average profit is 2.46 times the average loss, implying its clear superiority.

In addition, the total number of trades and profit play a significant role in the selection of the best model. PM12 has almost $\frac{1}{3}$ of the trades that SMA has, but it yields slightly less profit than the SMA model. However, PM12 earns in average 46,08 pips per trade, whereas SMA earns 17,18 pips per trade. Moreover, PM12 assists traders in the avoidance of their capital loss due to HFT (High Frequency Trading). As a result, our model with a few trades achieves to risk less and earn a lot.

Furthermore, the maximum drawdown (MDD) of each model is significant, because it constitutes an index of the model's risk. Table 1 reveals that the distinguished PM12 model exhibits the smallest risk among all other models. Taking all these factors into account, we can safely conclude that PM12 is a very competitive approach that leaves fertile ground for further improvements.

6 CONCLUSIONS

We introduced a FOREX trading model based on moving average forecast aggregation and trading rules. The aggregate forecast was optimized using the particle swarm optimization algorithm. Preliminary assessment of the proposed model was conducted on the EUR/USD time series under various parameter settings. Comparisons with the constituent moving average models were conducted.

The developed trading system exhibited competitive overall performance, with a specific variant achieving highest profitability with minimum loss risk. This comes despite the fact that traditional moving average models were able to outperform our proposed model in terms of forecasting accuracy. Nevertheless, in terms of

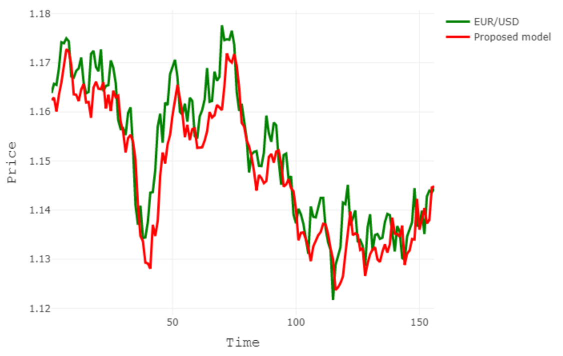

Figure 2: Actual vs predicted EUR/USD time series for the PM12 model.

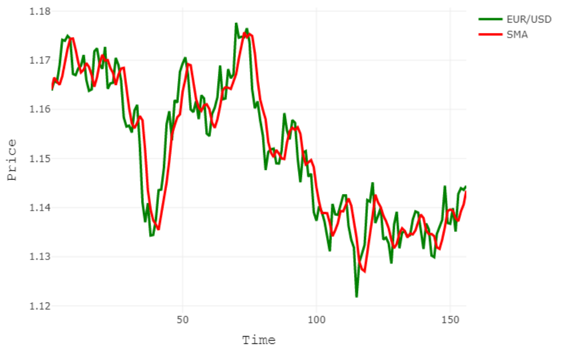

Figure 3: Actual vs predicted EUR/USD time series for the SMA model.

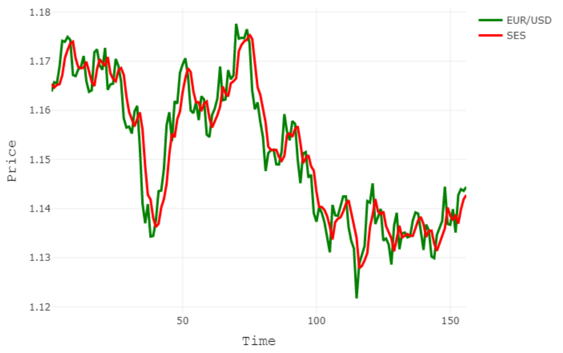

Figure 4: Actual vs predicted EUR/USD time series for the SES model.

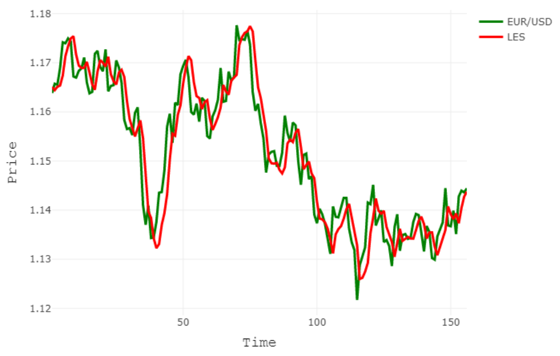

Figure 5: Actual vs predicted EUR/USD time series for the LES0 model.

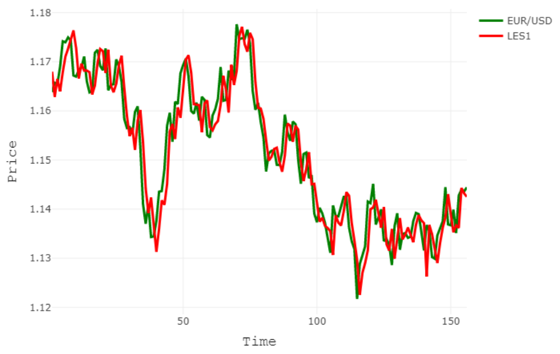

Figure 6: Actual vs predicted EUR/USD time series for the LES1 model.

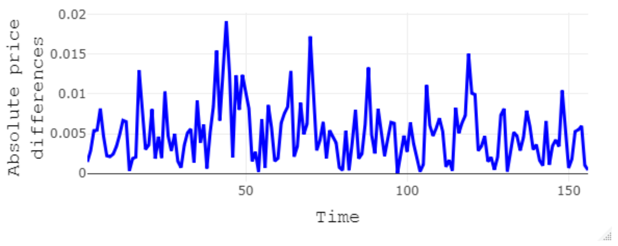

Figure 7: Absolute price differences between actual and predicted EUR/USD time series for the PM12 model.

trading performance the proposed model outperformed all the rest. In future work we intend to measure our model's performance in a shorter and a longer period, perform statistical testing in order to verify the statistical significance of the numerical results, as well as consider the transaction fees that incur per trade.

In general, we outlined the difficulty of FOREX market forecasting and introduced a profitable trading system. This indicates that traders should experiment beyond the boundaries of traditional models. Their trading decisions shall be based on forward-looking expectations from models and strategies that are optimized within a hybrid trading approach.

Nonetheless, there are still many directions to elaborate in search of efficient calibration of computational intelligent models for financial and economic forecasting tasks.

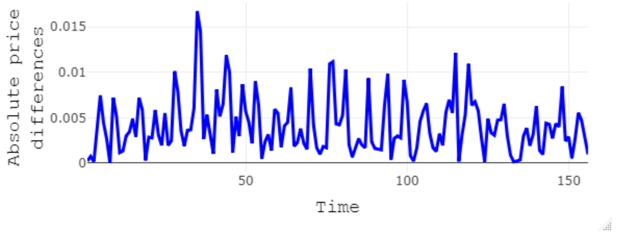

Figure 8: Absolute price differences between actual and predicted EUR/USD time series for the SMA model.

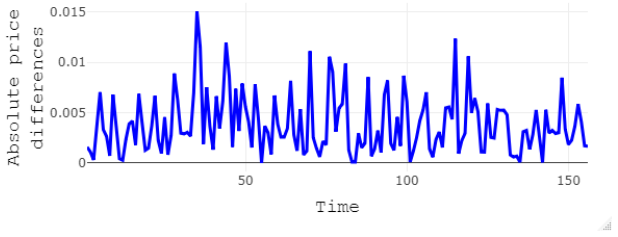

Figure 9: Absolute price differences between actual and predicted EUR/USD time series for the SES model.

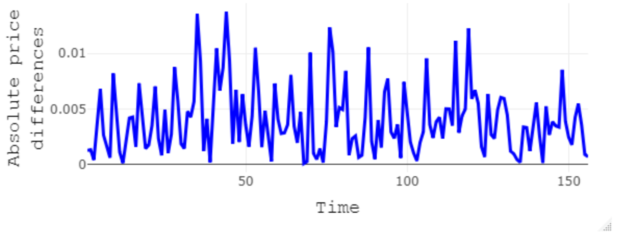

Figure 10: Absolute price differences between actual and predicted EUR/USD time series for the LES model.

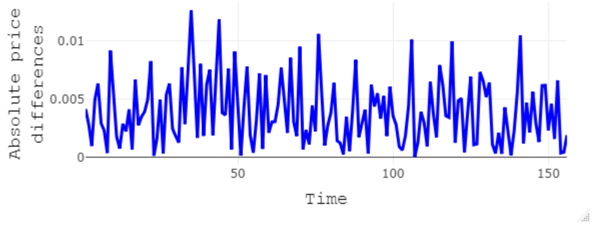

Figure 11: Absolute price differences between actual and predicted EUR/USD time series for the LES1 model.

REFERENCES

[1] BIS. 2019. *Foreign exchange turnover in April 2019.* Bank for International Settlements. https://www.bis.org/statistics/rpfx19_fx.htm
[2] Chen C. and Ye F. 2012. Particle swarm optimization algorithm and its application to clustering analysis. In *2012 Proceedings of 17th Conference on Electrical Power Distribution.* 789–794.
[3] James C. 2020. *Bretton Woods Agreement.* https://www.investopedia.com/terms/b/brettonwoodsagreement.asp
[4] ECB. 2018. *Euro money market study 2018.* European Central Bank. https://www.ecb.europa.eu/pub/euromoneymarket/pdf/ecb.euromoneymarket201909_study.en.pdf
[5] FXCM. [n.d.]. . https://www.fxcm.com/gr/
[6] Sermpinis G., Dunis C., Laws J., and Stasinakis C. 2012. Forecasting and trading the EUR/USD exchange rate with stochastic Neural Network combination and time-varying leverage. *Decision Support Systems* 54, 1 (2012), 316 – 329.
[7] Hameed H.H. 2015. *Smoothing Techniques for Time Series Forecasting.* Master of Science.
[8] Robinson J. and Rahmat-Samii Y. 2004. Particle swarm optimization in electromagnetics. *IEEE Transactions on Antennas and Propagation* 52, 2 (2004), 397–407.
[9] Schwager J.D. 1999. *Getting Started in Technical Analysis.*
[10] K.K. L., Yu, Lai, and Wang S. 2005. Designing a hybrid AI system as a forex trading decision support tool. In *17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05).* 5 pp.–93.
[11] Evans M.K. 2002. *Practical business forecasting.*
[12] Taylor M.P. and Allen H. 1990. Charts, Noise and Fundamentals in the London Foreign Exchange Market. *Economic Journal* 100, 400 (1990), 49–59.
[13] Taylor M.P. and Allen H. 1992. The use of technical analysis in the foreign exchange market. *Journal of International Money and Finance* 11, 3 (1992), 304 – 314.
[14] Czekalski P., Niezabitowski M., and Styblinski R. 2015. ANN for FOREX Forecasting and Trading. In *2015 20th International Conference on Control Systems and Computer Science.* 322–328.
[15] K. E. Parsopoulos and M. N. Vrahatis. 2010. *Particle Swarm Optimization and Intelligence: Advances and Applications.* Information Science Publishing (IGI Global).
[16] Chang P.H. and Osler C.L. 1999. Methodical Madness: Technical Analysis and the Irrationality of Exchange- Rate Forecasts. *The Economic Journal* 109, 458 (1999), 636–661.
[17] Tsaih R., Hsu Y., and Lai C.C. 1998. Forecasting S&P 500 stock index futures with a hybrid AI system. *Decision Support Systems* 23, 2 (1998), 161 – 174.
[18] Eberhart R.C. and Kennedy J. 1995. A New Optimizer Using Particles Swarm Theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science* (1995). IEEE.
[19] Hyndman R.J. 2013. *Forecasting: principles and practice.*
[20] Wasendorf R.R. 2009. *SFO Personal Investor Series: Forex Trading.*
[21] Yager R.R. 2008. Time Series Smoothing and OWA Aggregation. *IEEE Transactions on Fuzzy Systems* 16, 4 (2008), 994–1007.
[22] Yager R.R. 2013. Exponential smoothing with credibility weighted observations. *Information Sciences* 252 (2013), 96 – 105.
[23] Alna S.E. and Ahiakpor F. 2011. ARIMA (Autoregressive Integrated Moving Average) Approach to Predicting Inflation in Ghana (March 28, 2011). *Journal of Economics and International Finance* 5 (2011), 328–336.
[24] Dayal V. 2015. *An Introduction to R for Quantitative Economics: Graphing, Simulating and Computing.* Springer. http://www.springer.com/978-81-322-2340-5
[25] McKinney W. 2017. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython.*
[26] Lui Y.H. and Mole D. 1998. The use of fundamental and technical analyses by foreign exchange dealers: Hong Kong evidence. *Journal of International Money and Finance* 17, 3 (1998), 535 – 545.