An Evolutionary Approach for Parameter Selection on Time series Forecasting Hybrid Models

Ruy Brito Barbosa  
Paulo Salgado Gomes de Mattos Neto

Universidade Federal de Pernambuco, Centro de Informática, Recife, Pernambuco, Brazil  
rbb3@cin.ufpe.br  
psgmn@cin.ufpe.br

Abstract. Finding which model can best approach the optimal solution for time series forecasting is still a challenge among researchers. It has been shown that while SVRs (Support Vector Regressors) can be applied to a wide range of problems with linear and non-linear components, ARIMA (Autoregressive Integrated Moving Average) models are satisfactorily used for primarily linear problems. Also, it has been shown that hybrid models, as a combined approach, can reach high degree of accuracy. However, finding the best architecture of the model, which may include the set of input parameters and hyperparameters, can be an expensive task due to the dimensions of the search space. In this paper, we propose an evolutionary approach for approximating these parameters employing genetic algorithm concepts. A number of experiments has been conducted in order to show we can outperform results of ad-hoc selected parameters on previous research.

Keywords: Time series forecasting, ARIMA, SVR, Genetic Algorithms, Parameter selection.

1 Introduction

Time series forecasting is the prediction of future observations of quantifiable phenomena, given past observations through time. When the underlying generating process of the phenomenon cannot be formally described, approximative models are employed and used to extrapolate the time series into the future. Many occurrences in the literature focus on proposing new strategies for creating these models or improving existing ones. Improvements in this area may represent significant impact on several fields such as finance, engineering, social sciences and climate analysis.

One of these models which has been widely used is the Autoregressive Integrated Moving Average (ARIMA). In the domain of linear models, the ARIMA is considered to be flexible since it can capture pure autoregressive (AR), pure moving average (MA) and combined (ARMA) series. Despite being constrained to capture only the linear components of a time series, the simplicity of its statistical properties and the Box-Jenkins methodology in the model building process [1]
have made it a popular choice over the past three decades, with several applications across distinct fields, such as finance [2], engineering [3] and agriculture [4]. In reaction to the ARIMA limitation generated by the strong assumption of a linear relationship with lagged data, the literature contains several attempts to propose more robust models. Among these attempts there are Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs).

Considered to be more robust than ARIMA models, i.e. to embrace a wider class of problems for having less limitations, ANNs have been extensively used for capturing the nonlinear components, being the first attempt reported by Lapedes and Farber [5]. The choice for ANNs is usually due to its capability of approximating any continuous measurable function with arbitrarily desired accuracy [6, 7]. Many further references in the literature describe successful applications and detailed analysis of ANNs usage on forecasting.

Support Vector Machines (SVM) applied to regression as proposed by Drucker, Harris et al. [8] have been also applied on forecasting. Thissen et al. [9] and Gui et al. [10] have employed the first SVMs on the problem of time series forecasting. This approach maps samples from original space to high-dimensional space, through kernel tricks, and then constructs a linear decision function to achieve linear regression. When time series data is very complex, SVR models can reach better experimental results compared to ARIMA models.

Hybrid approaches for time series forecasting combine different strategies to achieve better performance. Zhang [11] introduces the combination of ARIMA and ANNs. Zhang’s Hybrid model assumes there is an addictive correlation between linear and nonlinear components. The forecasting is performed in two steps, as to the neural network is assigned the task of predicting the error of the linear model forecasting. Empirical results suggests hybrid models can improve performance metrics of separate pure models.

Khashei and Bijari [12] have questioned Zhang’s primary assumption of an addictive relationship between linear and nonlinear components. Their hybrid approach, which is also based on ARIMA and ANN models, assigns to the neural network the task of deriving the correlation between these components. Empirical comparative results suggest they can improve the forecasting performance by generalizing this correlation.

The recent improvements in the time series forecasting field are significant. Yet there are challenges regarding the choice of parameters on the hybrid model design process. It has been shown that the choice of SVM hyperparameters has significant effect on performance metrics [13].

On the linear forecasting step of the aforementioned hybrid models, most of the parameters are derived from the statistical properties of the series. However, the choice of nonlinear prediction parameters represents a complex task. Trivial
grid-search with cross-validation is not practical because of the large domain of possible parameter values, which is further exacerbated by the lack of prior knowledge on the data.

Previous research has shown how evolutionary approaches, such as Genetic Algorithms (GAs), can tackle this parameter selection task. Miller et al [23] presented evidence that GAs are able to optimize appropriately encoded architectures of neural networks. Whitley et al [24] suggest the usage of GAs for optimization of feed-forward network connection weights. On the support vector field, Wu et al [25] introduced a novel Hybrid Genetic Algorithm (HGA) for electricity load forecasting based on the evolutionary optimization of SVR hyperparameters, such as kernel function and kernel parameters.

2 Time series forecasting models

The literature registers several approaches for time series modeling. Although having the clear drawback of only covering linearity, the traditional statistical linear models such as exponential smoothing [14] and ARIMA have been successfully applied to datasets with low complexity. In order to overcome this limitation a new class of nonlinear models has been proposed. Models such as the bilinear model, the threshold autoregressive model and the autoregressive conditional heteroscedastic have been compared to traditional linear models and even though some improvement could be registered, the type of nonlinearity these models could account for was restricted [15]. Hence, it has been proposed another class of nonlinear models including more flexible and robust strategies such as ANNs, SVM and LS-SVR.

More recently, hybrid models have been suggested in order to overcome pure model deficiencies. Since early works from Reid [16] and Bates and Granger [17], a significant potential in combining different strategies was already suggested. Both theoretical and empirical findings have shown how effective and efficient this hybridization can be [18]. In the past years, several strategies have been registered, such as Pai and Lin [19] proposing an hybrid methodology with SVMs and ARIMA for stock prices forecasting and Khashei et al. [20] proposing a solution for financial markets prediction using ARIMA, fuzzy logic and ANNs.

2.1 The ARIMA model

An autoregressive integrated moving average model can be roughly described as a function of the time represented by a linear combination of previous values and random errors. Assuming $\phi_i (i = 1,2,\ldots,p)$ and $\theta_j (j = 1,2,\ldots,q)$ to be the coefficients or previous values ($y_{t-i}$) and random errors ($\epsilon_{t-j}$), respectively, the one-step-ahead can be obtained by
\[ y_t = e_t + \sum_{i=1}^{p} \phi_i y_{t-i} - \sum_{j=1}^{q} \theta_j e_{t-j}, \]  

(1)

where the random errors are assumed to be independently distributed with mean zero and variance \( \sigma^2 \).

Commonly denoted as ARIMA(p, d, q), the model accepts three parameters, namely the order of the previous values (p), the order of differentiating (d) and the order of random errors (q). The Box and Jenkins [1] methodology suggests three iterative steps for model identification, parameter estimation and diagnostic checking. This work proposes the usage of the autocorrelation function and the partial autocorrelation function of the sample data for identifying model parameters. The methodology is iterative and repeated until the identification of a satisfactory model, which is then submitted to forecasting.

### 2.2 Artificial Neural Network models

Artificial neural networks (ANNs) can be considered generalizations of linear models to higher dimensions. The flexibility of the model comes not only from its ability to approximate any continuous measurable function, but also from the parallel processing of data and the independence of prior knowledge of the data. Instead, they are able to adapt to the data and approximate complex underlying generating functions of the series.

Although being a rich class of models, the Multilayer Perceptron (MLP) with a single hidden layer is the most used approach for time series forecasting within the neural networks field. The structure of this network is best described as a stateless weighted directed graph of nodes (processing units) with no cycles. Assuming the network to have a linear activation function on the output layer and \( w_{ij}(i = 0, 1, \ldots, P; j = 1, 2, \ldots, Q) \) to be the connection weights, the one-step-ahead can be obtained by

\[ y_t = w_0 + \sum_{j=1}^{Q} w_{0j}g \left( \sum_{i=1}^{P} w_{ij}y_{t-i} \right) + e_t, \]  

(2)

where, similarly to the ARIMA model, \( P \) represents the order of the previous values. The \( Q \) parameter represents the number of nodes on the single hidden layer and the \( g(\cdot) \) function is a generalization of the hidden layer activation function, which is oftenly a sigmoid function. These parameters are determinant on the model performance and, since there is no systematic rule for selection, their choice is usually experimental. The MLP model used on forecasting can be represented by the \( N^{P-Q+1} \) notation, having an unique number of nodes on the output layer due to the one-step-ahead characteristic of the problem.
2.3 Support Vector Regression models

Support Vector Regression (SVR) consists on applying kernel functions to input data in order to perform linear regression on top of the transformed hyperspace. Some of these kernel functions are able to capture themselves the nonlinearities of the underlying processes generating correlation between input and output data. There are few attempts of adapting classical Support Vector Machines (SVMs) to the regression problem, among which we have $\varepsilon$-SVR, Least Squares SVR (LS-SVR) and Twin SVR (TSVR). In this paper we have proposed the usage of $\varepsilon$-SVR to the forecasting problem.

The $\varepsilon$-SVR model employs a margin of tolerance $\varepsilon$. Assuming $w$ to be the weight vector representing the coefficients of the transformed input values and $\Phi(\cdot)$ the chosen kernel function, the one-step-ahead can be obtained by

$$y_t = (w \cdot \Phi(y_{t-1}, y_{t-2}, ..., y_{t-p})) + \theta,$$  \hspace{1cm} (3)

where $\theta$ is the bias term. Typical kernel functions are Radial Basis Function (RBF), polynomial and sigmoid. The model training process aims to find the regressor with maximum deviation ($\varepsilon$) from the respective feature sets while being as flat as possible. Thus, this task is usually interpreted as a convex optimization problem.

Along with the kernel function and the margin of tolerance, some hyperparameters are considered to be significant on the model performance. The factor of regularization $C$ trades off between the flatness and the amount up to which deviations larger to $\varepsilon$ are tolerated [21]. Furthermore, the $\gamma$ parameter controls the influence of a single training sample. Hence, it can be interpreted as the inverse of the radius of influence of samples selected by the model as support vectors. Some of these parameters are only relevant to specific types of kernel functions. In the next sections we will refer to $S(\Phi(\cdot), C, \gamma, \varepsilon)$ as the model configured with the described hyperparameters.

2.4 Zhang’s hybrid model

Zhang introduced the combination of the ARIMA and ANNs to time series forecasting. The main assumption of his hybrid model is that the residuals from the linear forecasting must contain nonlinearities, and this residuals could be forecasted with a flexible model such as neural networks. Thus, he defines the one-step-ahead in terms of linear and nonlinear components as

$$y_t = L_t + N_t$$  \hspace{1cm} (4)
The forecasting is performed in two steps. In the first step, an ARIMA model is used to analyse the linear part of the series. The residuals containing nonlinearities are analysed sequentially in a second step. They can be interpreted as a second time series of the form

\[ e_t = y_t - L', \quad (5) \]

where \( L' \) is the linear forecast. The neural network would be applied as the function \( f(\cdot) \) as follows

\[ e_t = f(e_{t-1}, e_{t-2}, \ldots, e_{t-p}) + \epsilon, \quad (6) \]

being \( \epsilon \) the random error term. Zhang’s empirical results suggested that hybrid approaches could improve performance metrics of pure models [11].

### 2.5 Khashei & Bijari’s hybrid model

Khashei & Bijari extended Zhang’s model by generalizing the correlation between the linear and nonlinear components, which on the previous model was assumed to be additive (see Eq. 4). Given no prior assumption about this correlation, it was given to the neural network the task of approximating the underlying functions that generate the correlation itself. Thus, as a generalization of Eq 4, the forecasting can be obtained by

\[ y_t = f(L_t, N_t), \quad (7) \]

In a similar fashion, the forecasting is performed in two steps and the first step is the prediction of the linear components of the series. However, the second step considers the linear forecast as part of the input vector, given by

\[ x = (y_{t-1}, y_{t-2}, \ldots, y_{t-p}, L'_{t-1}, e_{t-1}, e_{t-2}, \ldots, e_{t-q}), \quad (8) \]

where the errors of lagged values are obtained as in Eq 5. Hence, following the structure of Eq 2 and considering a linear activation function on the output layer, the one-step-ahead prediction of the neural network is given by

\[ y_t = w_0 + \sum_{j=1}^{Q} w_{0j} + \sum_{i=1}^{P-p+q+1} w_{ij}x_i + e_t, \quad (9) \]

where \( w_{ij}, (i = 0, 1, \ldots, P; j = 1, 2, \ldots, Q) \) are the connection weights, \( Q \) is the number of nodes on the unique hidden layer, \( P = p + q + 1 \) is the dimension of the input vector (see Eq 8) and \( g(\cdot) \) is the generalization of the hidden layer activation function.
3 Formulation of the proposed model

Hybrid models, as a combined approach for time series forecasting, have their performance deeply associated not only with the chosen strategy but also with the choice of input parameters and hyperparameters. Designing the proposed model, we decided to use an Autoregressive Integrated Moving Average (ARIMA) model along with Support Vector Regressors (SVRs), respectively aiming to predict the linear and non-linear components of the series. The choice of the remaining factors which may influence the performance of the hybrid model has been the task of a Genetic Algorithm (GA).

3.1 Linear forecasting and series preprocessing

The chosen ARIMA model has been successfully used by Zhang [11] for predicting linear components. We decided not to include the optimisation of this model hyperparameters as a task for the GA, since their optimal values for the experimented datasets were already known. Hence, the linear forecasting step has been removed from the evolutionary process and included in the time series preprocessing routine.

The main goal of this preprocessing is to build a dataset for which for each entry on the original time series, we would have not only the linear forecast, but also previous values and errors, according to previous linear forecasts. Since we are using the optimal number of time lags $p$, picked upon analysis of the correlation between current term and previous ones, the preprocessed dataset should have then, $2p + 1$ columns. Assuming $L'_t$ as the linear forecast at time $t$, the function describing the process can be expressed as

$$P(p, t) = [y_{r-1}, y_{r-2}, ..., y_{r-p}, L'_t, e_{r-1}, e_{r-2}, ..., e_{r-p}]$$  \hspace{1cm} (10)

3.2 Nonlinear forecasting

The SVR model has been applied to a subset of the vector resulting from the previous preprocessing. Even though the number of time lags $p$ is optimal for the linear forecast, they may not be relevant on the non-linear step. Therefore, the GA aims to optimise this subset.

The remaining factors which may influence the performance of the model are the SVR hyperparameters. Namely, the kernel function ($\phi(\cdot)$), the factor of regularization ($C$), the kernel coefficient for non-linear hyperplanes ($\gamma$) and the threshold of distance between prediction and real value for which no penalty is charged ($\epsilon$). Optimising these parameters is also part of the GA objective.
3.3 Chromosome definition

The genetic composition of the individual must include both input parameters and hyperparameters, since both types of parameters must be optimised by the GA. The input parameters are submitted to a binary function, \( g(\cdot) \), indicating whether the parameter is selected or not.

![Fig 1. Chromosome definition](image)

Analysing the genetic structure (Fig. 1) it is possible to classify the gene according to the type of entity it represents. These types determine the crossover process, which must maintain the nature of the entity. Thus, we can define three crossover strategies:

- Any gene arising from the crossing of binary genes inherits the value from one of the parents randomly.
- Any gene arising from the crossing of categorical genes inherits the value from one of the parents randomly.
- Any gene arising from the crossing of numerical genes inherits the mean of the parents’ values weighted randomly.

Moreover, it is possible to classify the gene according to the parameter type as these types determine the initialization process. It can be defined as follows:

- Any input parameter with up to \( N \) genes will be initialized with only the first \( k \) \((0 \leq k \leq N)\) genes set, being the choice of \( k \) random.
- Any hyperparameter will be initialized as a random choice of preset start values (see Table 1).

The process of genetic mutation, to the which are exposed only the hyperparameters, is defined as a local and random search with limited iterations. Thus, during the mutation, the individual will only update its parameters in case it finds a better suited set of parameters.

The hyperparameter neighborhood grid (Table 2) is intended to provide similar solutions to the current one. For numeric parameters, the grid stores the current
parameters multiplied by log-separated ratios. The local search iterates randomly on the hyperparameter neighborhood grid and applies k-fold cross-validation. The choice for the randomized search instead of the exhaustive one is due to performance, as the former performed only approximately 15% of the later iterations.

3.4 Evolutionary process

The implemented GA had an initial population with a fixed number of individuals, which evolved over a limited number of generations. The selection has been applied according to the following process:

- The best fitted individual is selected as a parent for the next generation;
- From the remaining individuals, 20% are selected as parents for the next generation. The probability of being picked is proportional to the squared fitness value;
- The set of selected parents for the next generation doubles its size with random new individuals;
- The selected individuals are coupled and crossed randomly until the new generation reaches the generation size;
- 5% of the new generation individuals are submitted to mutation.

The algorithm runs until either the maximum number of epochs is reached or the best individual, for a specific amount of epochs, present no evolution on its fitness value. A pseudocode of the implementation is displayed below.

```plaintext
Main class:

maximumEpoch integer
currentEpoch integer
bestIndividuals []Individual

func Run(dataset Dataset)
    input := PreProcess(dataset)
    population := GenerateInitialPopulation()
    while !HasReachedMaximumEpoch() and !HasConverged(population):
        population.Select()
        population.Mutate()
        bestIndividuals.Append(population.BestIndividual())

func PreProcess(dataset Dataset) Input
    // Apply ARIMA and build input as a new dataset
    // with offset of lag P

func HasReachedMaximumEpoch() bool
    return currentEpoch == maximumEpoch

func HasConverged(population Population) bool
    // Check if best individuals from last K generations
    // have the same fitness
```
Population class:

```go
individuals []Individual

func Select() void
    individuals.SortAsc(individual => individual.Fitness())
    bestIndividuals := individuals.TakeBest()
    worstIndividuals := individuals.TakeWorst()
    parents := worstIndividuals.RouletteSelectBest()
    parents.AddRandomIndividuals()
    newIndividuals := parents.ApplyCrossover()
    individuals = newIndividuals.Concat(bestIndividuals)

func Mutate() void
    foreach individual in individuals:
        if RandomShouldMutate():
            individual.Mutate()

func BestIndividual() Individual
    individuals.SortDesc(individual => individual.Fitness())
    return individuals[0]
```

Individual class:

```go
input Input
params Parameters
model Model
fitness float

func Fitness() float
    if fitness == null:
        fitness = 1 / (TestModel() ^ 2)
    return fitness

func Mutate() void
    // Perform local search iterating randomly on the hyperparameter
    // neighborhood grid and applying k-fold cross-validation

func TestModel() float
    return model.Test(input, params)

func TrainModel() void
    model.Train(input, params)
```

4 Application of the proposed model

Aiming to test our proposed model against other approaches, we performed some experiments keeping consistency with previous research. In this section, we describe how the model has been configured and detail how these experiments have been conducted.

4.1 Datasets

Three well-known datasets have been used in the experiments. Namely, the Wolf’s sunspot data, the Canadian lynx data and the GBP to USD exchange rate. All of them have been exhaustively studied in the statistical as well as the machine learning literature. Hybrid models have improved linear model performances suggesting that nonlinear components can be found in each of them.
The sunspot data we considered in the study contains annual occurrences of sunspots from 1700 to 1987, which represent 288 observations. The analysis of sunspot activity has practical importance to geophysicists, environment scientists, and climatologists [26]. The series is considered as nonlinear and non-Gaussian, and have been used in previous research [27]. The plot of the series (Fig. 3) suggests an approximately decennial periodicity.

Considering our study applies SVR models and understanding their performance dependency upon feature scaling, we applied standardization (Z-score normalization), as given by

$$
\tau(x) = \frac{x - \mu}{\sigma}.
$$

The operation has been used solely for optimisation purposes and the inverse transformation, $\tau^{-1}(x)$, has been applied prior to performance measurement in order to adopt consistency with previous research.

The canadian lynx data we consider in the study represents the yearly amount of lynx trapped in the Mackenzie River district of Northern Canada from 1821 to 1934, corresponding to 114 observations. As the sunspot dataset, it has been widely studied and both linear and nonlinear components have been spotted. Furthermore, the plot of the series (Fig. 4) also suggests an approximately decennial periodicity. Following previous research, the logarithm (to the base 10) has been used both on modeling and performance measurement.

The GBP to USD exchange rate considered in the study represent weekly averages from 1980 to 1993, corresponding to 731 observations (Fig. 5). Yet an important field to international finance, the analysis of exchange rate represents a very difficult task since most of the studies applied so far have not proved to be better than a naive random walk model. According to previous research, the natural logarithm has also been used both on modeling and performance measurement.
Fig. 3. Wolf’s sunspot series.

Fig. 4. Canadian lynx data.

Fig. 5. GBP to USD exchange rate data.
4.2 Model configuration

The proposed model was been experimented considering some preset constants. Among which we can mention the hyperparameter initial values grid (Table 1), used to initialize the hyperparameters for every individual with approximately log-separated values and the hyperparameter neighborhood grid (Table 2), used to identify similar configurations for the current individual.

<table>
<thead>
<tr>
<th>φ(·)</th>
<th>linear</th>
<th>rbf</th>
<th>poly</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>0.1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>γ</td>
<td>1e-7</td>
<td>1e-4</td>
<td>1e-2</td>
</tr>
<tr>
<td>ε</td>
<td>0</td>
<td>0.03</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 1. Hyperparameter start values grid

<table>
<thead>
<tr>
<th>φ(·)</th>
<th>linear</th>
<th>rbf</th>
<th>poly</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>0.01</td>
<td>0.1</td>
<td>10</td>
</tr>
<tr>
<td>γ</td>
<td>0.01</td>
<td>0.1</td>
<td>10</td>
</tr>
<tr>
<td>ε</td>
<td>0.01</td>
<td>0.1</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 2. Hyperparameter neighborhood grid

Some other properties of the evolutionary process have been considered to be sensitive to the dataset. The maximum number of epochs for each evolutionary process has varied according on the dataset, but the number of epochs with no improvement for the convergence to be assumed has been set to 10% of the maximum number of epochs.

5 Results and comparison with previous research

In this section, we test how the proposed algorithm is able to outperform previous research on linear, nonlinear and hybrid approaches. We compare against the ARIMA linear prediction, the SVR prediction, the Zhang’s hybrid model and the Khashei and Bijari hybrid model. To align with previous research, we adopted the two error measures as performance indicators: Mean Absolute Error (MAE) and Mean Squared Error (MSE), defined as

\[ MAE = \frac{1}{N} \sum_{i=1}^{N} e_i, \]  
\[ (12) \]
\[ MSE = \frac{1}{N} \sum_{i=1}^{N} (e_i)^2 , \]  

(13)

Also, in the following sections, we present the comparison between the error measures relative to previous research. In this context, negative values represent improvement from our model and positive values the opposite.

5.1 Wolf’s sunspot series

For the Wolf’s sunspot series, two test horizons of 35 and 67 years have been considered on previous research. A subset autoregressive model of order nine has been found to be adequate judging by residual analysis and the most parsimonious among all ARIMA models. As the coefficients for the 3 to 8 have been constrained to 0, the subset of the lags correspond to \{1, 2, 9\}. This model has been employed by many researchers [11, 26, 27].

Exhaustive search with k-fold cross-validation analysis on the hyperparameter start values grid (see Table 1) has shown that the best SVR model was the \( S(rbf; 10^3; 10^{-4}; 3.10^{-1}) \) of lag 2. This model has shown better results for both horizons and both performance indicators.

The neural network based hybrid solution employed by Khashei & Bijari is a \( N^{7-4-1} \), with a sigmoid and a linear activation function, respectively on the hidden and output layers. The input layer correspond to the vector of seven nodes: \([y_{t-1}, y_{t-2}, y_{t-3}, y_{t-4}, L_t', e_{t-1}, e_{t-2}]\).

For the horizon of 67 years, the proposed algorithm was initialized with 200 individuals and has evolved through 116 epochs. The best-fitted individual suggests the \( S(rbf; 5.89 . 10^2; 1.7 . 10^{-2}; 4,14 . 1) \) model with the input parameters vector \([y_{t-1}, y_{t-2}, L_t', e_{t-1}, e_{t-2}]\).

Fig. 6. Wolf sunspot series prediction plot.
<table>
<thead>
<tr>
<th>Model</th>
<th>Horizon 35</th>
<th></th>
<th>Horizon 67</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
<td>MSE</td>
</tr>
<tr>
<td>Auto-regressive Integrated Moving Average (ARIMA)</td>
<td>11.319</td>
<td>216.965</td>
<td>13.033</td>
<td>306.082</td>
</tr>
<tr>
<td>Support Vector Regressor (SVR)</td>
<td>18.606</td>
<td>730.062</td>
<td>14.543</td>
<td>381.407</td>
</tr>
<tr>
<td>Zhang’s Hybrid Model</td>
<td>10.831</td>
<td>186.827</td>
<td>12.780</td>
<td>280.159</td>
</tr>
<tr>
<td>Khashei &amp; Bijari NN-based Hybrid Model</td>
<td>8.847</td>
<td>129.425</td>
<td>11.447</td>
<td>218.642</td>
</tr>
<tr>
<td>Our GA-optimised SVR-based model</td>
<td>11.781</td>
<td>239.925</td>
<td>11.085</td>
<td>231.260</td>
</tr>
</tbody>
</table>

Table 3. Comparative results for the sunspot series.

<table>
<thead>
<tr>
<th>Model</th>
<th>Horizon 35</th>
<th></th>
<th>Horizon 67</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
<td>MSE</td>
</tr>
<tr>
<td>Auto-regressive Integrated Moving Average (ARIMA)</td>
<td>+3.40%</td>
<td>+10.5%</td>
<td>-14.94%</td>
<td>-24.44%</td>
</tr>
<tr>
<td>Support Vector Regressor (SVR)</td>
<td>-36.6%</td>
<td>-67.13%</td>
<td>-23.77%</td>
<td>-39.36%</td>
</tr>
<tr>
<td>Zhang’s Hybrid Model</td>
<td>+8.06%</td>
<td>+22.1%</td>
<td>-13.26%</td>
<td>-17.45%</td>
</tr>
<tr>
<td>Khashei &amp; Bijari NN-based Hybrid Model</td>
<td>+24.9%</td>
<td>+46.0%</td>
<td>-3.16%</td>
<td>+5.45%</td>
</tr>
</tbody>
</table>

Table 4. Percentage of improvement for the sunspot series.

5.2 Canadian lynx series

For the Canadian lynx series, a single test horizon of 14 years has been used on previous research. In a similar fashion, the best-fitted autoregressive model was an AR(12). This model has been employed by many researchers [11, 27].

On a similar fashion, a complete search with k-fold cross-validation analysis on the hyperparameter start values grid (see Table 1) has indicated the $S(rbf; 10^3; 10^{-1}; 3.10^{-1})$ as the best model. However, while MAE could be optimised for lag 3, MSE was optimised for lag 2.

The neural network based hybrid solution employed by Khashei & Bijari is a $N^{8 \times 3 \times 1}$, with a $tanh$ and a linear activation function, respectively on the hidden and output layers. The input layers correspond to the vector of eight nodes: $[y_{r-1}, y_{r-2}, y_{r-3}, y_{r-4}, y_{r-5}, L', e_{r-1}, e_{r-2}]$.

Our proposed algorithm was initialized with 400 individuals and has evolved through 179 epochs. The initial size of the population for this experiment has been intentionally increased to avoid early convergence. The best-fitted individual suggests the $S(rbf; 1; 6.47 \cdot 10^2; 2; 98 \cdot 10^2)$ model with the input parameters vector $[y_{r-1}, y_{r-2}, y_{r-5}, L', e_{r-1}, e_{r-4}]$. 
Fig. 7. Canadian lynx series prediction plot.

<table>
<thead>
<tr>
<th>Model</th>
<th>Horizon 14</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
</tr>
<tr>
<td>Auto-regressive Integrated Moving Average (ARIMA)</td>
<td>0.112255</td>
</tr>
<tr>
<td>Support Vector Regressor (SVR)</td>
<td>0.077828</td>
</tr>
<tr>
<td>Zhang’s Hybrid Model</td>
<td>0.103972</td>
</tr>
<tr>
<td>Khashei &amp; Bijari NN-based Hybrid Model</td>
<td>0.085055</td>
</tr>
<tr>
<td><strong>Our GA-optimised SVR-based model</strong></td>
<td><strong>0.053734</strong></td>
</tr>
</tbody>
</table>

Table 5. Comparative results for the canadian lynx series.

<table>
<thead>
<tr>
<th>Model</th>
<th>Horizon 14</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
</tr>
<tr>
<td>Auto-regressive Integrated Moving Average (ARIMA)</td>
<td>-52.13%</td>
</tr>
<tr>
<td>Support Vector Regressor (SVR)</td>
<td>-30.95%</td>
</tr>
<tr>
<td>Zhang’s Hybrid Model</td>
<td>-48.31%</td>
</tr>
<tr>
<td>Khashei &amp; Bijari NN-based Hybrid Model</td>
<td>-36.82%</td>
</tr>
</tbody>
</table>

Table 6. Percentage of improvement for the canadian lynx series.

5.3 GBP to USD exchange rate series

For the exchange rate series, three test horizon of 1, 6 and 12 months have been used on previous research. According to previous research, many exchange rate
series follow the theory of efficient market hypothesis (EMH) [28]. Hence, the best prediction of tomorrow’s exchange can be considered the current exchange rate, which follows a simple random walk [29]. For that reason, we have considered the ARIMA model of order (0, 1, 0), which is a general form of a random walk.

The different test horizons have presented heterogeneous hyperparameter sets during the complete search analysis. For the 1-month test horizon, the best SVR model for both MSE and MAE was a $S(rbf; 10^{3}; 10^{-2}; 10^{-2})$ with lag 9. The 6-month and the 12-month test horizons, on the other hand, could be optimally predicted for both performance metrics with an $S(rbf; 10^{2}; 10^{-2}; 0)$ of lag 1. The nature of this series combined with the results achieved for the shortest test horizon may suggest that this model has presented an highly overfitted solution.

Khashei & Bijari’s neural network hybrid solution is described as a $N^{12-4-1}$, with a sigmoid and a linear activation function, respectively on the hidden and output layers. The input layer, consisting of twelve nodes, is represented as $[y_{t-1}, y_{t-2}, y_{t-3}, y_{t-4}, y_{t-5}, L', e_{t-1}, e_{t-2}, e_{t-3}, e_{t-4}, e_{t-5}, e_{t-6}]$.

Our proposed algorithm was initialized with 200 individuals and has evolved through 119 epochs. The best-fitted individual suggests the $S(rbf; 3.939 \cdot 10^{1}; 6.08 \cdot 10^{-2}; 1,18 \cdot 10^{-2})$ model with the input parameters vector $[y_{t-1}, e_{t-1}, e_{t-2}, e_{t-3}, e_{t-4}, e_{t-5}]$. Note that for this series, the algorithm has proposed ignoring the linear forecast.

![Fig. 8. GBP to USD exchange rate series prediction plot.](image)

<table>
<thead>
<tr>
<th>Model</th>
<th>1 month horizon</th>
<th>6 months horizon</th>
<th>12 months horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
</tr>
<tr>
<td>Auto-regressive Integrated Moving Average (ARIMA)</td>
<td>5.016</td>
<td>36.8493</td>
<td>6.0447</td>
</tr>
<tr>
<td>Support Vector Regressor (SVR)</td>
<td>4.214</td>
<td>24.108</td>
<td>5.4688</td>
</tr>
</tbody>
</table>
Zhang’s Hybrid Model 4.146 26.7259 5.8823 56.5507 5.1212 43.5907
Khashei & Bijari NN-based Hybrid Model 3.972 23.9915 5.3361 42.7822 4.9691 36.4774
Our GA-optimised SVR-based model 2.879 8.88162 3.8661 24.2081 4.6900 32.9742

Table 7. Comparative results for the GBP to USD exchange rate series.

<table>
<thead>
<tr>
<th>Model</th>
<th>1 month horizon</th>
<th></th>
<th></th>
<th>6 months horizon</th>
<th></th>
<th></th>
<th>12 months horizon</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
<td>MSE</td>
</tr>
<tr>
<td>Auto-regressive Integrated Moving Average (ARIMA)</td>
<td>-42.60%</td>
<td>-75.89%</td>
<td>-36.04%</td>
<td>-57.21%</td>
<td>-12.46%</td>
<td>-27.20%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support Vector Regressor (SVR)</td>
<td>-31.68%</td>
<td>-63.15%</td>
<td>-29.30%</td>
<td>-43.19%</td>
<td>-21.94%</td>
<td>-39.47%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zhang’s Hybrid Model</td>
<td>-30.56%</td>
<td>-66.76%</td>
<td>-34.27%</td>
<td>-57.19%</td>
<td>-8.41%</td>
<td>-24.35%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Khashei &amp; Bijari NN-based Hybrid Model</td>
<td>-27.51%</td>
<td>-62.98%</td>
<td>-27.54%</td>
<td>-43.41%</td>
<td>-5.61%</td>
<td>-9.60%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8. Percentage of improvement for the GBP to USD exchange rate series.

6 Conclusion and future research

In this work, we have proposed a novel approach for optimising parameters of an hybrid model for time series forecasting. The main goal is to propose a method to automate the parameter selection within the design process of an hybrid model. The cost of the optimisation task grows with the search space, which is primarily defined by the aspects of the design process we consider in the problem. Not covering all aspects of this process is, then, the main limitation of our work.

Not every time series can be optimally forecast with ARIMA or ε-SVR models, therefore other models should be considered. Perhaps deciding which model to use should also be part of the Evolutionary Algorithm task, which may not necessarily be a Genetic Algorithm. The less assumptions are taken, the more generalist will be the method. Future work should, then, extrapolate our assumptions in order to achieve better results.

7 References


