Journal of the Chinese
Statistical Association
Vol. 63, (2025) 364–394

Boosted hybrid method for non-stationary data analysis

Chih-Hao Chang[1†], Shih-Feng Huang[2] and Yu-Sheng Lo[3]

[1] Department of Statistics, National Chengchi University, Taipei, Taiwan
[2] Graduate Institute of Statistics, National Central University, Taoyuan, Taiwan
[3] Institute of Statistics, National University of Kaohsiung, Kaohsiung, Taiwan

ABSTRACT

This paper introduces a novel boosting method for hybrid models designed to analyze non-stationary time series data. The hybrid models are constructed by decomposing the original time series into a stationary component and a non-stationary component, which are respectively fitted using a linear statistical model, such as the autoregressive model, and a nonlinear machine learning technique, such as the neural network method well developed in the literature. To enhance the predictive accuracy of these hybrid models, we propose a boosting approach to improve the nonlinear models in forecasting the non-stationary component. Furthermore, the exponential weighted moving average (EWMA) method is employed for data smoothing, serving as a competitive alternative to the filtering methods used in existing hybrid models. The performance of the hybrid models, along with their boosted and EWMA variants, is evaluated using several metrics. To identify a robust hybrid model with optimal predictive performance, a model selection procedure based on multiple split cross-validation (MS-CV) techniques is implemented, focusing on the overall ranking of prediction metrics across all candidate models. Three empirical studies are conducted to demonstrate the effectiveness of the MS-CV selection procedure.

Key words and phrases: boosting method, hybrid models, linear models, model selection, neural network method, nonlinear models.

JEL classification: C53, E17.

[†]Corresponding to: Chih-Hao Chang
E-mail: jhow@nccu.edu.tw

1. Introduction

Time series analysis is integral to modern data science, with substantial applications in finance and economics. Traditional statistical methods in time series analysis include linear models such as the autoregressive integrated moving average (ARIMA; Box and Jenkins 1976) model and nonlinear models like the Generalized autoregressive conditional heteroskedasticity (GARCH; Bollerslev 1986) model. In recent years, there has been a notable shift towards integrating machine learning techniques into nonlinear time series modeling. Among these techniques, artificial neural networks (ANNs), including long short-term memory (LSTM) networks (Hochreiter and Schmidhuber 1997), have emerged as flexible and effective tools for capturing complex and dynamic patterns in time-dependent data. Despite their strengths, these neural network methods are often plagued by issues such as overfitting and limited interpretability, which can hinder their practical application and predictive accuracy. Balancing model interpretability with predictive power remains a significant challenge in the field of time series analysis.

To address these challenges, hybrid models have become a prominent approach for leveraging both linear and nonlinear methods to enhance the accuracy of time series forecasting. Zhang (2003) introduced an additive hybrid model that integrates ARIMA and ANN to capitalize on their respective strengths in linear and nonlinear analysis. Building on this idea, Khashei and Bijari (2011) proposed an alternative hybrid approach that utilizes historical data, along with the fitted and residual values from ARIMA, as inputs for ANN, aiming to improve predictive performance. Additionally, Babu and Reddy (2014) extended the hybrid model concept by incorporating a moving-average (MA) filter to decompose the original time series into a smoother component and its residuals. In this approach, ARIMA models the smoother series, while ANN models the residuals. Further advancing this approach, Büyükahin and Ertekin (2019) introduced the Augmented Dickey-Fuller (ADF; Dickey and Fuller 1981) test for unit root testing, which adjusts the order of the MA filter for better smoothing. The smoothed series is then modeled with ARIMA, and the historical data, along with the current fitted values and residuals, are used as inputs for ANN forecasting.

In this article, we explore two enhancement approaches applied to the hybrid models previously discussed to improve predictive performance. The first approach involves the use of the exponential weighted moving average (EWMA; Roberts 1959) as an alternative to the smoothing techniques implemented in Babu and Reddy (2014) and Büyükahin and Ertekin (2019). The EWMA method assigns greater weight to more recent data points, thereby effectively capturing rapid fluctuations within the time series. This approach facilitates a more accurate decomposition of the original time series into a smoothed series and its residuals. The second approach involves incorporating boosting methods into the ANN components of the aforementioned hybrid models. Boosting is an ensemble learning technique that combines multiple weak learners to create a robust model, focusing on difficult cases to reduce bias and variance, thus enhancing overall model performance (see Sarkar 2022 for a comprehensive discussion).

In this study, we refer to the hybrid models proposed by Zhang (2003), Khashei and Bijari (2011), Babu and Reddy (2014), and Büyükahin and Ertekin (2019) as the additive hybrid (AH) model, the hybrid neural network (HNN) model, the additive hybrid moving average (AHMA) model, and the smoothed hybrid neural network (SHNN) model, respectively. The models adjusted using the EWMA method are denoted as EWAH, EWHNN, EWAHMA, and EWSHNN. Additionally, all these models can be enhanced using the boosting method, resulting in the following names: B-AH, B-HNN, B-AHMA, B-SHNN, B-EWAH, B-EWHNN, B-EWAHMA, and B-EWSHNN. We will compare the performance of these models through empirical data analysis to evaluate the effectiveness of the EWMA and boosting enhancements. The accuracy and stability of both the original and enhanced hybrid models will be assessed using various metrics, and a model selection mechanism will be established to identify and validate the most competitive model; further details are provided in Section 4.

This paper is organized as follows. Section 2 provides an overview of the hybrid methods discussed in the literature. Section 3 details the data preprocessing techniques proposed in this study, including the integration of the EWMA method and the boosting techniques applied to the hybrid models. In Section 4, we conduct three empirical studies to perform a comprehensive comparison among the hybrid models introduced in this article. Finally, Section 5 offers a summary and discussion of the findings.

2. Notations and literature review

In this section, we present the ARIMA, ANN, and hybrid models as discussed in the literature, detailing the prediction processes for these hybrid models. We begin by defining the notations and their physical meanings for each model. For each time index $t \in \{1, \ldots, T\}$, these notations include the observation y_t, its linear component L_t, and its nonlinear component N_t. Additionally, we denote the low- and high-volatility components of y_t, generated by smoothing filters, as ℓ_t and r_t, respectively. The residuals of the linear model fitted using ARIMA are denoted by e_t. Furthermore, we represent the ARIMA-based linear function by g and the ANN-based nonlinear function by f throughout this article.

2.1 Autoregressive integrated moving average (ARIMA) model

The ARIMA model is a linear forecasting model characterized by the parameters $p, q, d \in \mathbb{N}$. It employs differencing d times to remove the non-stationarity from the original time series, allowing for the fitting of an autoregressive moving average (ARMA) model with p autoregressive terms and q moving average terms. When the series does not require differencing (i.e., $d = 0$), the ARIMA model can be expressed as follows:

$$y_t = a_0 + \sum_{i=1}^{p} \alpha_i y_{t-i} + \epsilon_t + \sum_{j=1}^{q} \beta_j \epsilon_{t-j}, \tag{1}$$

where a_0 denotes the intercept term, $\alpha_1, \ldots, \alpha_p$ are the p autoregressive coefficients, $\beta_1, \ldots, \beta_q$ are the q moving average coefficients, and ϵ_t represents the white noise error term, which follows a normal distribution with mean zero and variance $\sigma^2 > 0$.

2.2 Artificial neural network (ANN) model

The ANN model is a highly flexible and widely used nonlinear model in the field of machine learning. Its structure allows for variation in the number of layers and neurons within the model. The ANN model is typically divided into an input layer, hidden layer, and output layer. To model time series data using this three-layer network, we construct a nonlinear function h of $y_{t-1}, \ldots, y_{t-N}$ as follows:

$$y_t = \omega_0 + \sum_{j=1}^{J} \omega_j h\left(\omega_{0j} + \sum_{i=1}^{I} \omega_{ij} y_{t-i}\right) + \zeta_t, \tag{2}$$

where for each $t = 1, \ldots, T$, J denotes the number of nodes in the hidden layer, I denotes the number of nodes in the input layer, w_j and w_{ij} are the weights of the ANN model, and ζ_t represents the noise of the model. The function h applied to the hidden layers is commonly chosen to be a sigmoid, ReLU, or tanh function in practice.

2.3 Additive hybrid (AH) model

Zhang (2003) introduces a hybrid model that assumes

$$y_t = L_t + N_t, \quad t = 1, \dots, T \tag{3}$$

where L_t and N_t denote the linear and nonlinear components of y_t, respectively. The linear component is modeled using ARIMA, providing the predictor $\hat{L}_t$. The residuals from this linear fit, which are assumed to contain only nonlinear relationships, are then modeled using an ANN to obtain the predictor $\hat{N}_t$. This hybrid approach, which combines both linear and nonlinear components, aims to enhance overall forecasting performance. The algorithm proposed by Zhang (2003), referred to as AH in this paper, for fitting these two components is detailed in Algorithm AH.

Algorithm AH

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function AH($\mathcal{D}$)
2:     Obtain the linear predictor $\hat{L}_t$ based on $\mathcal{D}$ fitted by the ARIMA model.
3:     Obtain the residuals $e_t = y_t - \hat{L}_t$.
4:     Obtain the nonlinear predictor $\hat{N}_t$ based on the ANN method with input variables $e_{t-1}, e_{t-2}, \dots, e_{t-n}$, $n \in \mathbb{N}$:
               $\hat{N}_t = f(e_{t-1}, \dots, e_{t-n})$
5:     return $\hat{y}_t = \hat{L}_t + \hat{N}_t$, for $t = 1, \dots, T$.
6: end function
```

2.4 Hybrid neural network (HNN) model

In contrast to the additive hybrid model proposed by Zhang (2003), Khashei and Bijari (2011) develops a model that captures the time series data as a nonlinear function of both the linear component and its residuals, which are constructed using the ARIMA model and historical data. The nonlinear function is estimated using an ANN. Unlike the approach of Zhang (2003), which assumes an additive relationship between the linear and nonlinear components, the model proposed by Khashei and Bijari (2011) does not impose such an assumption. Instead, it assumes that

$$y_t = f(L_t, N_t), \quad t = 1, \dots, T \tag{4}$$

where L_t and N_t represent the linear and nonlinear components of y_t, respectively. The linear component is modeled using ARIMA to obtain the predictor $\hat{L}_t$. Residuals from the linear component, along with $\hat{L}_t$ and the original data, are then used as inputs to the ANN to derive the final predictor $\hat{y}_t$ of y_t. The algorithm proposed by Khashei and Bijari (2011), referred to as HNN in this article, for fitting the nonlinear function of these components is detailed in Algorithm HNN.

Algorithm HNN

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function HNN($\mathcal{D}$)
2:     Obtain the linear predictor $\hat{L}_t$ based on $\mathcal{D}$ fitted by the ARIMA model.
3:     Obtain the residuals $e_t = y_t - \hat{L}_t$.
4:     return the nonlinear predictor $\hat{y}_t$ by ANN with input variables including $\hat{L}_t$, $e_{t-1}, \dots, e_{t-n}$ and $y_{t-1}, \dots, y_{t-m}$:
           $\hat{y}_t = f(e_{t-1}, \dots, e_{t-n}, \hat{L}_t, y_{t-1}, \dots, y_{t-m}), \quad 1 < n < t, \quad 1 < m < t.$
5: end function
```

2.5 Additive hybrid moving average (AHMA) model

Babu and Reddy (2014) introduce a modification to the additive hybrid model by incorporating a moving average (MA) filter. Unlike the linear and nonlinear decomposition approach of Zhang (2003), Babu and Reddy (2014) propose to decompose the original time series into a low-volatility component ℓ_t and a high-volatility component r_t using a fixed MA filter. Specifically, the decomposition is defined as:

$$\begin{aligned} \ell_t &= \frac{1}{s} \sum_{i=t-s+1}^{t} y_i, \\ r_t &= y_t - \ell_t, \quad t = s, \dots, T \end{aligned} \tag{5}$$

where s represents the length of the fixed MA filter and is chosen to approximate the kurtosis of ℓ_t to 3. The kurtosis of ℓ_t is defined as:

$$k_\ell \equiv \frac{\mathrm{E}\big((\ell_t - \mathrm{E}(\ell_t))^4\big)}{\big(\mathrm{E}(\ell_t - \mathrm{E}(\ell_t))^2\big)^2}, \tag{6}$$

and is estimated using the method of moments on ℓ_t. Kurtosis quantifies the thickness or heaviness of the tails of a distribution. The kurtosis of a standard normal distribution is 3. Therefore, the filtering method aims to produce a low-volatility series, ℓ_t, that approximates a normal distribution. The residue, considered the high-volatility component, r_t, is assumed to be nonlinear. These two components, ℓ_t and r_t, are then modeled using ARIMA and ANN, respectively, to obtain the predictors $\hat{\ell}_t$ and $\hat{r}_t$. The algorithm proposed by Babu and Reddy (2014), referred to as AHMA in this paper, which fits this additive hybrid model of the two components, is detailed in Algorithm AHMA.

Algorithm AHMA

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function AHMA($\mathcal{D}$)
2:     Use the fix MA filter to generate the low-volatility component $\ell_t$ and high-volatility component $r_t$ defined in (5) with $s$ being chosen such that $k_\ell$ defined in (6) approximating to 3.
3:     Obtain the low-volatility predictor of $\ell_t$:
           $\hat{\ell}_t = g(\ell_{t-1}, \dots, \ell_{t-m}, e_{t-1}, \dots, e_{t-n}), \quad s < n < t, \quad s < m < t$
       where the linear function $g$ and residuals $e_{t-1}, \dots, e_{t-n}$ are obtained from fitting the ARIMA model to $\ell_t$.
4:     Obtain the high-volatility predictor of $r_t$:
           $\hat{r}_t = f(r_{t-1}, \dots, r_{t-p}), \quad s < p < t,$
       where the nonlinear function $g$ is obtained from fitting the ANN model to $r_{t-1}, \dots, r_{t-p}$.
5:     return $\hat{y}_t = \hat{\ell}_t + \hat{r}_t$, for $t = s, \dots, T$.
6: end function
```

2.6 Smoothed hybrid neural network (SHNN) model

Büyükahin and Ertekin (2019) apply the ADF test, a unit root test, to assess whether differencing is necessary before fitting the hybrid model proposed by Babu and Reddy (2014). Once a stationary series is obtained, the fixed MA filter is utilized once more to decompose the series into a low-volatility component, ℓ_t, and a high-volatility component, r_t. In contrast to Babu and Reddy (2014), who fit only the high-volatility component to an artificial neural network (ANN) model, Büyükahin and Ertekin (2019) also incorporate the predicted values from the ARIMA model fitted to the low-volatility component, together with historical data, as inputs for constructing the ANN model.

The algorithm proposed by Büyükahin and Ertekin (2019), referred to as SHNN in this article, is detailed in Algorithm SHNN.

Algorithm SHNN

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function SHNN($\mathcal{D}$)
2:     Calculate the $p$-value of the ADF test on the given series, and apply the differencing method to the series as the updated series until the $p$-value is larger than 0.05.
3:     Use the fix MA filter to decompose the generated series for the low-volatility component $\ell_t$ and high-volatility component $r_t$ defined in (5) with $s$ being chosen such that $k_\ell$ defined in (6) approximating to 3.
4:     Obtain the low-volatility predictor of $\ell_t$:
           $$\hat{\ell}_t = g(\ell_{t-1}, \dots, \ell_{t-m}, e_{t-1}, \dots, e_{t-n}), \quad s < n < t, \quad s < m < t$$
       where the linear function $g$ and residuals $e_{t-1}, \dots, e_{t-n}$ are obtained from fitting the ARIMA model to $\ell_t$.
5:     return the predictor of $y_t$, $t = 1, \dots, T$:
           $$\hat{y}_t = f(r_{t-1}, \dots, r_{t-p}, \hat{L}_t, y_{t-1}, \dots, y_{t-q}), \quad s < p < t, \quad s < q < t,$$
       where the nonlinear function $g$ is obtained from fitting the ANN model to $r_t$, $\hat{L}_t$ and $y_t$.
6: end function
```

3. The proposed method

In this section, we present two approaches—the EWMA smoothing method and the boosting method—applied to the hybrid models discussed in Section 2 to enhance predictive performance.

3.1 Exponentially weighted moving average (EWMA)

This section details the application of the EWMA method to enhance hybrid models discussed in the literature. EWMA is a widely-used technique in time series analysis that assigns exponentially decreasing weights to past observations, making it particularly effective in capturing recent trends. Unlike fix MA filters, EWMA is highly sensitive to recent changes and thus well-suited for short-term forecasting, especially when recent data exhibit significant trends. Hence in this article, EWMA functions is considered as an alternative to the fixed MA filter for smoothing the original time series data. The trend extracted by EWMA, denoted as S_t, is defined as follows:

$$S_t = \alpha y_t + (1-\alpha)S_{t-1}, \quad t = 2, \dots, T, \tag{7}$$

where $\alpha \in (0,1)$ and $S_1 = y_1$. While α is often set to a default value of 0.2, it can be estimated using the `HoltWinters` function from the STATS package in R. EWMA's primary advantages include its simplicity and effectiveness in smoothing time series data and making short-term forecasts. The one-step-ahead forecast is given by the current smoothed value, i.e., $\hat{y}_{t+1} = S_t$, calculated using all past observations.

We now discuss the integration of EWMA with the ARIMA model. We first extract the trend component S_t defined in (7) and compute the residuals $e_t = y_t - S_t$. The ARIMA model is then fitted to these residuals to obtain the predictor $\hat{e}_t$. Consequently, the overall prediction is given by $\hat{y}_t = S_t + \hat{e}_t$. We refer to this combined approach as EWMA+ARIMA, with the algorithm detailed in Algorithm EWMA+ARIMA.

Algorithm EWMA+ARIMA

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function EWMA+ARIMA($\mathcal{D}$)
2:     Obtain the smoothed series $S_t$ be EWMA as given in (7), and its residue $e_t = y_t - S_t$.
3:     Obtain the linear predictor of $e_t$ by fitting the ARIMA model:
           $\hat{e}_t = g(e_{t-1}, \dots, e_{t-a}, \eta_{t-1}, \dots, \eta_{t-b}), \quad 1 < a < t, \quad 1 < b < t,$
       where $g$ is obtained from fitting the ARIMA model to $e_t$, and $\eta_t$ are the corresponding residuals.
4:     return the predictor $\hat{y}_t = S_t + \hat{e}_t$, $t = 1, \dots, T$.
5: end function
```

Recall that both the Additive Hybrid (AH) model proposed by Zhang (2003) and the Hybrid Neural Network (HNN) model introduced by Khashei and Bijari (2011) leverage the residuals from the ARIMA model to construct the nonlinear component of their predictions. In this study, we replace the residuals e_t in Algorithm AH and Algorithm HNN with those obtained from the EWMA+ARIMA method outlined in Algorithm EWMA+ARIMA. The modified algorithms are referred to as EWAH and EWHNN, detailed in Algorithm EWAH and Algorithm EWHNN.

Algorithm EWAH

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function EWAH($\mathcal{D}$)
2:     Obtain the predictor from Algorithm EWMA+ARIMA that is denoted by $\hat{L}_t$.
3:     Obtain the residuals $e_t = y_t - \hat{L}_t$.
4:     Obtain the nonlinear predictor $\hat{N}_t$ based on the ANN method with input variables $e_{t-1}, e_{t-1}, \dots, e_{t-n}$, $n \in \mathbb{N}$:
           $$\hat{N}_t = f(e_{t-1}, \dots, e_{t-n})$$
5:     return $\hat{y}_t = \hat{L}_t + \hat{N}_t$, for $t = 1, \dots, T$.
6: end function
```

Algorithm EWHNN

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function EWHNN($\mathcal{D}$)
2:     Obtain the predictor from Algorithm EWMA+ARIMA that is denoted by $\hat{L}_t$.
3:     Obtain the residuals $e_t = y_t - \hat{L}_t$.
4:     return the nonlinear predictor $\hat{y}_t$ by ANN with input variables including $\hat{L}_t$, $e_{t-1}, \dots, e_{t-n}$ and $y_{t-1}, \dots, y_{t-m}$:
           $$\hat{y}_t = f(e_{t-1}, \dots, e_{t-n}, \hat{L}_t, y_{t-1}, \dots, y_{t-m}), \quad 1 < n < t, \quad 1 < m < t.$$
5: end function
```

Finally, we replace the fix Moving Average (MA) filter used in Algorithm AHMA by Babu and Reddy (2014) and Algorithm SHNN by Büyükahin and Ertekin (2019) with the EWMA smoothing method as described in (7) for extracting the low-volatility series. The modified algorithms are designated as EWAHMA and EWSHNN, as detailed in Algorithm EWAHMA and Algorithm EWSHNN, respectively.

Algorithm EWAHMA

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function EWAHMA($\mathcal{D}$)
2:     Use the EWMA filter given in (7) to generate the low-volatility component that
       is defined by $\ell_t$, and high-volatility component $r_t = y_t - \ell_t$.
3:     Obtain the low-volatility predictor of $\ell_t$:
           $$\hat{\ell}_t = g(\ell_{t-1}, \dots, \ell_{t-m}, e_{t-1}, \dots, e_{t-n}), \quad s < n < t, \quad s < m < t$$
       where the linear function $g$ and residuals $e_{t-1}, \dots, e_{t-n}$ are obtained from fitting
       the ARIMA model to $\ell_t$.
4:     Obtain the high-volatility predictor of $r_t$:
           $$\hat{r}_t = f(r_{t-1}, \dots, r_{t-p}), \quad s < p < t,$$
       where the nonlinear function $g$ is obtained from fitting the ANN model to
       $r_{t-1}, \dots, r_{t-p}$.
5:     return $\hat{y}_t = \hat{\ell}_t + \hat{r}_t$, for $t = s, \dots, T$.
6: end function
```

Algorithm EWSHNN

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\dots,T}$.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function EWSHNN($\mathcal{D}$)
2:     Calculate the $p$-value of the ADF test on the given series, and apply the differ-
       encing method to the series as the updated series until the $p$-value is larger than
       0.05.
3:     Use the EWMA filter given in (7) to generate the low-volatility component that
       is defined by $\ell_t$, and high-volatility component $r_t = y_t - \ell_t$.
4:     Obtain the low-volatility predictor of $\ell_t$:
           $$\hat{\ell}_t = g(\ell_{t-1}, \dots, \ell_{t-m}, e_{t-1}, \dots, e_{t-n}), \quad s < n < t, \quad s < m < t$$
       where the linear function $g$ and residuals $e_{t-1}, \dots, e_{t-n}$ are obtained from fitting
       the ARIMA model to $\ell_t$.
5:     return the predictor of $y_t$, $t = 1, \dots, T$:
           $$\hat{y}_t = f(r_{t-1}, \dots, r_{t-p}, \hat{L}_t, y_{t-1}, \dots, y_{t-q}), \quad s < p < t, \quad s < q < t,$$
       where the nonlinear function $g$ is obtained from fitting the ANN model to $r_t$, $\hat{L}_t$
       and $y_t$.
6: end function
```

3.2 The boosting method

The various hybrid models exhibit distinct advantages depending on the data structure. To further enhance these models, in addition to applying the EWMA smoothing method to refine the linear components as discussed in Section 3.1, we also incorporate a boosting method to augment both the original hybrid models and their EWMA variants. Boosting is an ensemble learning technique that constructs a robust learner by combining multiple weak learners. In this section, we leverage the concept of ensemble learning to nonlinear modeling by iteratively training a series of weak classifiers and adjusting sample weights based on their performance.

Given the sequential nature of time series data, which reflects temporal dependencies and trends, directly applying weights to sample points could disrupt the inherent order and significance of these points. Therefore, integrating a weight-enhanced model directly into ARIMA may not be optimal. Instead, we focus on boosting the ANN component within the hybrid models. Let D_n^t represent the weight of the squared error loss for the tth predictor in the nth weighted ANN model within the boosting framework. Additionally, let W_n denote the weight associated with the aggregation of the nth weighted ANN model. The algorithm for the boosted ANN model is detailed in Algorithm B-ANN. In our study, we restricted the number of base learners in the boosted algorithm to $N = 5$, primarily to mitigate overfitting by controlling model complexity. This fixed N also facilitates a clearer assessment of the boosting method's impact on predictive performance. While it is possible to fine-tune N by evaluating predictive performance across various values in boosting, such exploration lies beyond this article's scope and is reserved for future work.

In the remainder of this article, for convenience, when the ANN modeling component in the aforementioned hybrid algorithms is replaced with Algorithm B-ANN, the algorithm names will be prefixed with "B-". For instance, the combination of Algorithm AH and Algorithm B-ANN will be designated as Algorithm B-AH. Consequently, detailed descriptions of all boosted hybrid models will not be provided.

Algorithm B-ANN

```
Require: $\mathcal{D} = \{(y_t, \boldsymbol{x}_t)\}_{t=1,\dots,T}$ with $y_t$ the response and $\boldsymbol{x}_t$ a covariate vector, representing the input residuals derived from fitting ARIMA models, and $N$ (set to 5 by default) denotes the number of base learners.
Ensure: $\hat{y}_t$, $t = 1, \dots, T$.
1: function B-ANN($\mathcal{D}$,$N$)
2:     Set $n = 1$ and the initial weight for the square error loss of the $t$th predictor by $D_n^t = 1/T$, for $t = 1, \dots, T$, to obtain the first weighted ANN predictor, denoted by $f_1(\boldsymbol{x}_t)$.
3:     Set $n = n + 1$ and update the weights $D_n^t$ for $n \geq 2$ by
           $$D_n^t = \frac{D_{n-1}^t \exp(\xi_{n-1}^t)}{\sum_{t=1}^T D_{n-1}^t \exp(\xi_{n-1}^t)}, \quad t = 1, \dots, T,$$
       where $\xi_n^t = |y_t - f_n(\boldsymbol{x}_t)|$ denotes the residual of $t$th predictor in the $n$th iteration.
4:     Obtain the $n$th $D_n^t$-based weighted ANN predictor, denoted by $f_n(\boldsymbol{x}_t)$.
5:     Go back 3 until $n = N$.
6:     Calculate the weights for aggregation of the $n$th ANN model, $W_n$, by
           $$w_n = \frac{1}{2} \log\left( \max\left( \frac{\widehat{\mathrm{Var}}(y_t) - \widehat{\mathrm{Var}}(\xi_n^t)}{\widehat{\mathrm{Var}}(\xi_n^t)}, 1 \right) \right),$$
           $$W_n = \frac{w_n}{\sum_{n=1}^N w_n},$$
       where for $\bar{y} = \sum_{t=1}^T y_t/T$ and $\bar{\xi}_n = \sum_{t=1}^T \xi_n^t/T$,
           $$\widehat{\mathrm{Var}}(y_t) \equiv \frac{1}{T-1} \sum_{t=1}^T (y_t - \bar{y})^2,$$
           $$\widehat{\mathrm{Var}}(\xi_n^t) \equiv \frac{1}{T-1} \sum_{t=1}^T (\xi_n^t - \bar{\xi}_n)^2, \quad n = 1, \dots, N.$$
7:     return the aggregated ANN predictor $\hat{y}_t$:
           $$\hat{y}_t = \sum_{n=1}^N W_n f_n(\boldsymbol{x}_t).$$
8: end function
```

4. Empirical studies

In this section, we evaluate the performance of the hybrid models using three metrics: the mean squared error (MSE), the mean absolute error (MAE), and the mean absolute scaled error (MASE). For an arbitrary predictor $\hat{y}_t$ of y_t, these metrics are defined as follows:

$$\begin{aligned}\text{MSE} &= \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2,\\ \text{MAE} &= \frac{1}{T}\sum_{t=1}^{T}|y_t - \hat{y}_t|,\\ \text{MASE} &= \frac{T-1}{T}\frac{\sum_{t=1}^{T}|y_t - \hat{y}_t|}{\sum_{t=2}^{T}|y_t - y_{t-1}|}.\end{aligned} \tag{8}$$

Among these metrics, MASE is particularly advantageous for assessing forecasting performance in data with seasonality due to its scale invariance, adjustment for seasonality, and capability for benchmark comparison.

In this section, we compare the boosting-enhanced hybrid methods introduced in Section 2 with the EWMA-modified hybrid methods discussed in Section 3.1. This comparison aims to evaluate the improvement in predictive accuracy that boosting imparts to hybrid models. The comparison is conducted across two candidate model sets: $\mathcal{M}_1 = \{$AH, HNN, AHMA, SHNN, B-AH, B-HNN, B-AHMA, B-SHNN$\}$ and $\mathcal{M}_2 = \{$EWAH, EWHNN, EWAHMA, EWSHNN, B-EWAH, B-EWHNN, B-EWAHMA, B-EWSHNN$\}$.

Given the variability in empirical performance, we employ a selection mechanism to identify more robust forecasting methods. To demonstrate the robustness of this selection approach, we apply the k-means clustering algorithm to classify these hybrid models based on their performance across the three metrics defined in (8). We anticipate that models exhibiting similar predictive performance will be grouped accordingly. To select the most competitive models from $\mathcal{M}_1$ and $\mathcal{M}_2$, evaluating them based on a combined assessment of three metrics given in (8), each assigned equal weight to determine the most robust model, we utilize a validation technique known as electoral college cross-validation (ECCV; Zhan and Yang 2022). The ECCV method addresses the potential bias of relying on a single training set for performance evaluation. It enhances model evaluation by combining multiple validation criteria across subsets of data, which are then aggregated in a voting-like process. Unlike standard cross-validation, which can be skewed by fold-specific anomalies, ECCV provides a more robust model assessment by balancing evaluations from diverse criteria. In this study, we further divide the training set into two segments—a training subset and a validation subset—using a range of ratios, then perform repeated training and validation across these splits. This method allows for the evaluation of model stability and consistency by assessing performance across multiple data partitions. Models that achieve consistently high rankings across different ratios are considered robust. We refer to this procedure as multiple split cross-validation (MS-CV), detailed in Algorithm MS-CV.

Algorithm MS-CV

```
Require: $\mathcal{D} = \{y_t\}_{t=1,\ldots,T}$, $L \in \mathcal{L} = \{\text{MSE,MAE,MASE}\}$ given in (8), $M \in \mathcal{M}$ with $\mathcal{M}$ the candidate model set.
Ensure: $\hat{M}$, the optimal model with the smallest rank.
1: function MS-CV($\mathcal{D}, \mathcal{L}, \mathcal{M}$)
2:     Consider $\delta \in \mathcal{S} = \{0.5, 0.6, 0.7, 0.8, 0.9\}$, where the first $\delta \times 100\%$ of the data $\mathcal{D}$ is set as the training set $\mathcal{D}_{T,\delta}$, and the remainder is used as the validation set $\mathcal{D}_{V,\delta}$.
3:     For each $\delta$ in $\mathcal{S}$, train each model $M \in \mathcal{M}$ based on $\mathcal{D}_{T,\delta}$ and calculate its prediction loss for each metric $L \in \mathcal{L}$ on $\mathcal{D}_{V,\delta}$. We denote the corresponding loss by $\rho_\delta(M, L)$.
4:     Calculate the over all predictive performance for the model $M$ under the splitting ratio $\delta$:
           $$\bar{\rho}_\delta(M) = \sum_{L \in \mathcal{L}} \frac{\rho_\delta(M, L)}{\sum_{M \in \mathcal{M}} \rho_\delta(M, L)}. \qquad (9)$$
5:     Rank the values of $\bar{\rho}_\delta(M)$ for all the models $M \in \mathcal{M}$ that are denoted by $R_\delta(M)$. A smaller rank indicates a smaller overall predictor performance among $L \in \mathcal{L}$.
6:     Calculate $\bar{R}(M) = \sum_{\delta \in \mathcal{S}} R_\delta(M)$, the summation of ranks of all possible ratio $\delta \in \mathcal{S}$.
7:     return the optimal model $\hat{M}$ by
           $$\hat{M} = \arg\min_{M \in \mathcal{M}} R(M).$$
8: end function
```

Note that in Algorithm MS-CV, the training set will include all data from the earlier portion of the dataset, along with the samples added as the ratio increases. Consequently, the training set always comprises a continuous segment of the data, ensuring that earlier observations are included when moving to higher split ratios. In addition, it performs only a single split for each specified ratio, as resampling methods can disrupt the inherent order of time series data. Accordingly, in our study, we split the time series data into two segments based on the specified ratio. In addition, we use the MS-CV algorithm to identify the model with the best overall performance across three metrics. However, in empirical analysis, if users have a particular preference for a specific metric, they can still apply the MS-CV algorithm focused on that metric to obtain the top-ranked model across different ratios. To further evaluate the predictive performance of the models selected through Algorithm MS-CV,we re-fit each hybrid model on the entire training set and subsequently compute the prediction errors using the three metrics on the testing set. The k-means clustering method is then employed to categorize the models into two groups: the leading group and the lagging group, based on these metrics. We anticipate that the final selected model will consistently be classified in the leading group during subsequent data analyses. Additionally, to enhance the robustness of the k-means clustering results, we normalize the three metrics for all models before applying the k-means method. Note that the normalization of the three metrics was implemented due to significant differences in scale across different models, which could affect the variation between metrics. To balance the influence of these variations, we adopted normalization prior to applying the k-means clustering method. Additionally, it is generally recommended to use normalized data for k-means clustering to ensure that the algorithm treats each metric with equal importance, as k-means is sensitive to the scale of the input features.

4.1 Forecasts for sunspot dataset

The sunspot data is collected from the Royal Observatory of Belgium during 1700 to 1987, comprising a total of 288 observations, with the first 221 used as the training set and the remaining 67 as the testing set. The dataset has been analyzed by Zhang (2003), Khashei and Bijari (2011), Babu and Reddy (2014) and Büyükahin and Ertekin (2019). In this article, ARIMA model parameters were selected based on the ACF and PACF plots of the sunspot data. As shown in Figure 1, the ACF plot showed a periodic effect of five, and the PACF plot suggested an AR(9) model fit, as it cut off after lag 9. To assess model adequacy, we applied the Ljung-Box Test to check for residual autocorrelation. The test showed p-values exceeding 0.05, indicating no significant residual correlation, and thus confirming the model's adequacy for future predictions. Therefore, we consider an AR(9) with a periodicity of 5 time units, for fitting the ARIMA model to predict the linear component of Algorithm AH and Algorithm HNN. In addition, for generating the low-volatility components in Algorithm AHMA and Algorithm SHNN, the length of the fixed MA filter is set as $m = 15$. As to the ANN model for each hybrid model, the tuning parameters are data-driven and are given in Table 1. Here we note that for the corresponding boosted hybrid models, the setting for the ARIMA and ANN parts in the algorithms are the same.

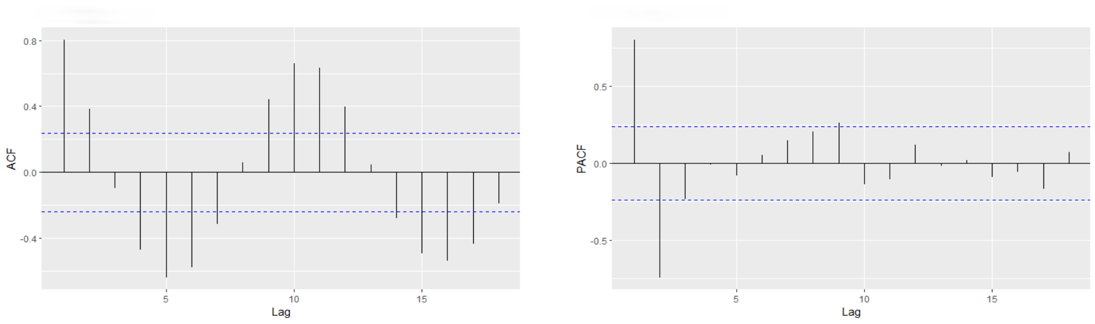


Figure 1: Autocorrelation and partial autocorrelation functions of sunspot training dataset.

Table 1: Hyper-parameter settings for the ANN model in the hybrid model of sunspot data.

Model	Neuron	Initializer	Activation function	Optimizer	Loss function
AH	$15 \times 14 \times 1$	normal	tanh	Adam	MSE
HNN	$24 \times 24 \times 1$	uniform	Relu	Adam	MSE
AHMA	$9 \times 9 \times 1$	uniform	Relu	Adam	MSE
SHNN	$18 \times 17 \times 1$	uniform	Relu	Adam	MSE

Table 2: Overall ranking of sunspot data under different hybrid models and varying training and validation split ratios, where $R_\delta(M)$ and $\bar{R}(M)$ are defined in Algorithm MS-CV with $M \in \mathcal{M}_1$.

Model	$R_{0.5}(M)$	$R_{0.6}(M)$	$R_{0.7}(M)$	$R_{0.8}(M)$	$R_{0.9}(M)$	$\bar{R}(M)$
AH	6	6	7	8	7	34
B-AH	8	8	8	7	8	39
HNN	7	7	6	6	6	32
B-HNN	5	4	5	5	2	21
AHMA	4	5	3	4	3	19
B-AHMA	1	1	1	3	5	11
SHNN	3	3	2	2	4	14
B-SHNN	2	2	4	1	1	**10**

Now, we apply Algorithm MS-CV over the set $\mathcal{M}_1$ to select an optimal model in overall prediction performance among the three metrics defined in (8). The ranks of the candidate models in $\mathcal{M}_1$ is illustrated in Table 2, which selects the B-SHNN model as the optimal model for the predicting the sunspots. Note that the variability in rankings across different split ratios likely stems from the boosted hybrid model's inherent complexity, as it integrates a linear ARIMA model with a nonlinear ANN model. Since ARIMA captures linear patterns while ANN is adept at modeling non-linear structures, changes in training sample proportions can lead ANN to respond more variably to distinct dataset features, impacting the ranking stability. Additionally, the nested nature of training samples under different splits can introduce variations that significantly influence ANN's performance, causing fluctuations in the overall rankings. Finally, the boosting process itself may adapt differently to unique data splits, further contributing to inconsistency in model rankings. To verify the validity of the B-SHNN model for the prediction, we re-fit each hybrid model on the entire training set and then calculate the prediction errors for three metrics on the testing set. As shown in Table 3, the predictive performance of the B-SHNN indeed falls into the leading group from using the k-means grouping method. While the hybrid model incorporates linear components to enhance interpretability, the use of ANN and subsequent boosting still results in considerable model complexity, which may lead to overfitting in the data analysis process. Although we applied the MS-CV algorithm to alleviate overfitting through validation, a certain degree of overfitting persists, which occasionally causes certain (boosted) hybrid models to exhibit unexpectedly poor performance on the testing set. This residual overfitting likely explains why some models (e.g., B-AHMA in Table 2),

Table 3: The three metrics, MAE, MSE and MASE, for each model in $\mathcal{M}_1$ based on the sunspot testing dataset, where the Group variable represents the classification results of k-means method based on the three normalized metrics with the normalized center point $(0.0715, 0.0505, 0.0715)$ for the leading Group 1 and the normalized center point $(0.7857, 0.7419, 0.7857)$ for the lagging Group 2.

Model	MAE	MSE	MASE	Group
AH	12.9972	301.7276	0.5617	2
B-AH	12.7214	313.1442	0.5498	2
HNN	12.6458	297.6040	0.5465	2
B-HNN	13.4168	348.4417	0.5798	2
AHMA	12.1766	302.4926	0.5262	2
B-AHMA	13.2909	312.9930	0.5744	2
SHNN	11.2687	206.3916	0.4870	1
B-SHNN	10.9100	220.7400	0.4715	1

although highly ranked by MS-CV during training and validation, fall to a second-tier level in actual testing performance.

Next, we apply Algorithm MS-CV over the set $\mathcal{M}_2$ to select an optimal model in overall prediction performance among the three metrics defined in (8). Note that for models in $\mathcal{M}_2$ and their boosted variants, they share the same tuning parameters of ANN models that are data-driven and are given in Table 4.

The ranks of the candidate models in $\mathcal{M}_2$ is illustrated in Table 5, which selects the B-EWSHNN model as the optimal model for the predicting the sunspots. To verify the validity of the B-EWSHNN model for the prediction, we re-fit each hybrid model on the entire training set and then calculate the prediction errors for three metrics on the testing set. As shown in Table 6, the predictive performance of the B-EWSHNN indeed falls into the leading group from using the k-means grouping method.

Table 4: Hyper-parameter settings for the ANN model in the hybrid model of sunspot data.

Model	Neuron	Initializer	Activation function	Optimizer	Loss function
EWAH	$10 \times 9 \times 1$	normal	tanh	Adam	MSE
EWHNN	$20 \times 19 \times 1$	uniform	Relu	Adam	MSE
EWAHMA	$9 \times 9 \times 1$	uniform	Relu	Adam	MSE
EWSHNN	$15 \times 14 \times 1$	uniform	Relu	Adam	MSE

Table 5: Overall ranking of sunspot data under different hybrid models and varying training and validation split ratios, where $R_\delta(M)$ and $\bar{R}(M)$ are defined in Algorithm MS-CV with $M \in \mathcal{M}_2$.

Model	$R_{0.5}(M)$	$R_{0.6}(M)$	$R_{0.7}(M)$	$R_{0.8}(M)$	$R_{0.9}(M)$	$\bar{R}(M)$
EWAH	6	7	7	5	8	33
B-EWAH	8	8	8	4	7	35
EWHNN	7	6	6	8	6	33
B-EWHNN	5	5	4	7	5	26
EWAHMA	4	3	3	2	2	14
B-EWAHMA	3	4	5	1	1	14
EWSHNN	2	2	2	6	4	16
B-EWSHNN	1	1	1	3	3	**9**

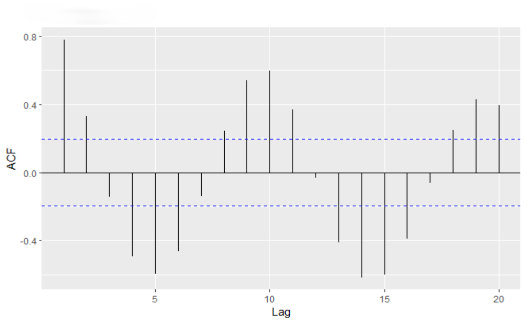
Table 6: The three metrics, MAE, MSE and MASE, for each model in $\mathcal{M}_2$ based on the sunspot testing dataset, where the Group variable represents the classification results of k-means method based on the three normalized metrics with the normalized center point $(0.2298, 0.0871, 0.2298)$ for the leading Group 1 and the normalized center point $(0.8771, 0.7819, 0.8771)$ for the lagging Group 2.

Model	MAE	MSE	MASE	Group
EWAH	13.6287	307.4490	0.5890	2
B-EWAH	13.7487	336.0213	0.5942	2
EWHNN	11.9983	227.3876	0.5185	1
B-EWHNN	13.4030	366.8180	0.5793	2
EWAHMA	13.1607	316.8685	0.5688	2
B-EWAHMA	13.0775	347.3039	0.5652	2
EWSHNN	11.0979	198.9242	0.4796	1
B-EWSHNN	10.3899	190.0374	0.4490	1

4.2 Forecasts for lynx dataset

The lynx dataset collects the number of lynx trapped per year in the Mackenzie River district of Northern Canada during 1821-1934, comprising a total of 114 observations, with the first 100 used as the training set and the remaining 14 as the testing set. The dataset has been analyzed by Khashei and Bijari (2011), Babu and Reddy (2014) and Büyükahin and Ertekin (2019). As shown in Figure 2, the ACF plot showed a periodic effect of five, and the PACF plot suggested an AR(11) model fit, as it cut off after lag 11. To assess model adequacy, we applied the Ljung-Box Test to check for residual autocorrelation. The test showed p-values exceeding 0.05, indicating no significant residual correlation, and thus confirming the model's adequacy for future predictions. Therefore, we consider an AR(11) with a periodicity of 5 time units, for fitting the ARIMA model to predict the linear component of Algorithm AH and Algorithm HNN. In addition, for generating the low-volatility components in Algorithm AHMA and Algorithm SHNN, the length of the fixed MA filter is set as $m = 3$. As to the ANN model for each hybrid model, the tuning parameters are data-driven and are given in Table 7. Here we note that for the corresponding boosted hybrid models, the setting for the ARIMA and ANN parts in the algorithms are the same.

Now, we apply Algorithm MS-CV over the set $\mathcal{M}_1$ to select an optimal model in overall prediction performance among the three metrics defined in (8). The ranks of



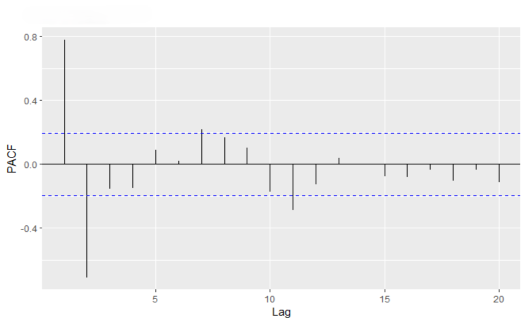


Figure 2: Autocorrelation and partial autocorrelation functions of Lynx training dataset.

Table 7: Hyper-parameter settings for the ANN model in the hybrid model of Lynx data.

Model	Neuron	Initializer	Activation function	Optimizer	Loss function
AH	$10 \times 9 \times 1$	uniform	Relu	Adam	MSE
HNN	$22 \times 21 \times 1$	uniform	Relu	Adam	MSE
AHMA	$11 \times 10 \times 1$	uniform	Relu	Adam	MSE
SHNN	$21 \times 20 \times 1$	uniform	Relu	Adam	MSE

Table 8: Overall ranking of Lynx data under different hybrid models and varying training and validation split ratios, where $R_\delta(M)$ and $\bar{R}(M)$ are defined in Algorithm MS-CV with $M \in \mathcal{M}_1$.

Model	$R_{0.5}(M)$	$R_{0.6}(M)$	$R_{0.7}(M)$	$R_{0.8}(M)$	$R_{0.9}(M)$	$\bar{R}(M)$
AH	4	7	4	3	5	23
B-AH	5	8	6	4	6	29
HNN	7	5	7	5	7	31
B-HNN	8	6	8	6	8	36
AHMA	2	4	2	1	1	10
B-AHMA	1	1	1	2	2	**7**
SHNN	3	2	3	8	3	19
B-SHNN	6	3	5	7	4	25

the candidate models in $\mathcal{M}_1$ is illustrated in Table 8, which selects the B-AHMA model as the optimal model for the predicting the sunspots. To verify the validity of the B-AHMA model for the prediction, we re-fit each hybrid model on the entire training set and then calculate the prediction errors for three metrics on the testing set. As shown in Table 9, the predictive performance of the B-AHMA indeed falls into the leading group from using the k-means grouping method.

Next, we apply Algorithm MS-CV over the set $\mathcal{M}_2$ to select an optimal model in overall prediction performance among the three metrics defined in (8). Note that for models in $\mathcal{M}_2$ and their boosted variants, they share the same tuning parameters of ANN models that are data-driven and are given in Table 10.

The ranks of the candidate models in $\mathcal{M}_2$ is illustrated in Table 11, which selects the B-EWAHMA model as the optimal model for the predicting the sunspots. To verify the validity of the B-EWAHMA model for the prediction, we re-fit each hybrid model on the entire training set and then calculate the prediction errors for three metrics on

Table 9: The three metrics, MAE, MSE and MASE, for each model in $\mathcal{M}_1$ based on the Lynx testing dataset, where the Group variable represents the classification results of k-means method based on the three normalized metrics with the normalized center point $(0.3374, 0.6067, 0.3375)$ for the leading Group 1 and the normalized center point $(0.8915, 0.7542, 0.8915)$ for the lagging Group 2.

Model	MAE	MSE	MASE	Group
AH	0.1047	0.0137	0.4682	1
B-AH	0.1203	0.0196	0.5380	2
HNN	0.1090	0.0174	0.4877	2
B-HNN	0.0789	0.0098	0.3528	1
AHMA	0.0940	0.0177	0.4207	1
B-AHMA	0.0882	0.0176	0.3945	1
SHNN	0.0919	0.0163	0.4112	1
B-SHNN	0.0900	0.0184	0.4025	1

Table 10: Hyper-parameter settings for the ANN model in the hybrid model of Lynx data.

Model	Neuron	Initializer	Activation function	Optimizer	Loss function
EWAH	$10 \times 9 \times 1$	uniform	Relu	Adam	MSE
EWHNN	$22 \times 21 \times 1$	uniform	Relu	Adam	MSE
EWAHMA	$11 \times 10 \times 1$	uniform	Relu	Adam	MSE
EWSHNN	$23 \times 22 \times 1$	uniform	Relu	Adam	MSE

Table 11: Overall ranking of Lynx data under different hybrid models and varying training and validation split ratios, where $R_{\delta}(M)$ and $\bar{R}(M)$ are defined in Algorithm MS-CV with $M \in \mathcal{M}_2$.

Model	$R_{0.5}(M)$	$R_{0.6}(M)$	$R_{0.7}(M)$	$R_{0.8}(M)$	$R_{0.9}(M)$	$\bar{R}(M)$
EWAH	6	8	8	6	8	36
B-EWAH	5	7	7	4	7	30
EWHNN	8	6	6	1	4	25
B-EWHNN	7	5	5	2	3	22
EWAHMA	4	3	2	3	1	13
B-EWAHMA	1	1	1	5	2	**10**
EWSHNN	2	4	4	8	5	23
B-EWSHNN	3	2	3	7	6	21

Table 12: The three metrics, MAE, MSE and MASE, for each model in $\mathcal{M}_2$ based on the Lynx testing dataset, where the Group variable represents the classification results of k-means method based on the three normalized metrics with the normalized center point $(0.2298, 0.0871, 0.2298)$ for the leading Group 1 and the normalized center point $(0.8771, 0.7819, 0.8771)$ for the lagging Group 2.

Model	MAE	MSE	MASE	Group
EWAH	0.1637	0.0348	0.7325	2
B-EWAH	0.1593	0.0348	0.7125	2
EWHNN	0.1444	0.0365	0.6460	2
B-EWHNN	0.1396	0.0283	0.6247	2
EWAHMA	0.1131	0.0184	0.5059	1
B-EWAHMA	0.0976	0.0118	0.4367	1
EWSHNN	0.0972	0.0138	0.4350	1
B-EWSHNN	0.0771	0.0090	0.3447	1

the testing set. As shown in Table 12, the predictive performance of the B-EWAHMA indeed falls into the leading group from using the k-means grouping method.

4.3 Forecasts for Gbp/Usd dataset

The Gbp/Usd dataset collects the exchange rate between British pound and US dollar which contains weekly observation during 1980-1993, comprising a total of 730 observations in the time series, with the first 678 used as the training set and the

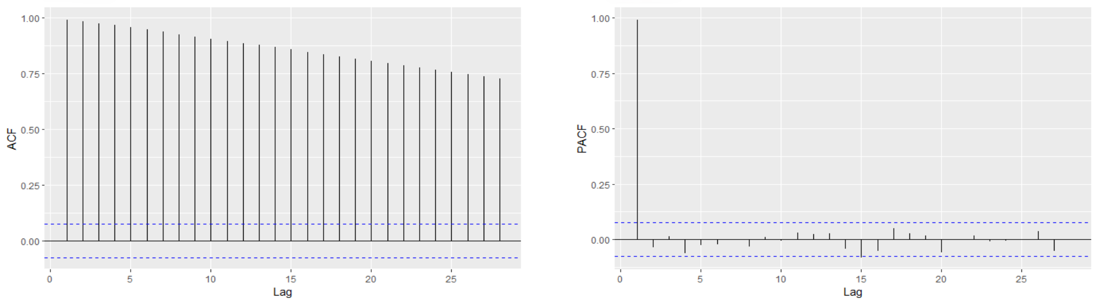


Figure 3: Autocorrelation and partial autocorrelation functions of Gbp/Usd training dataset.

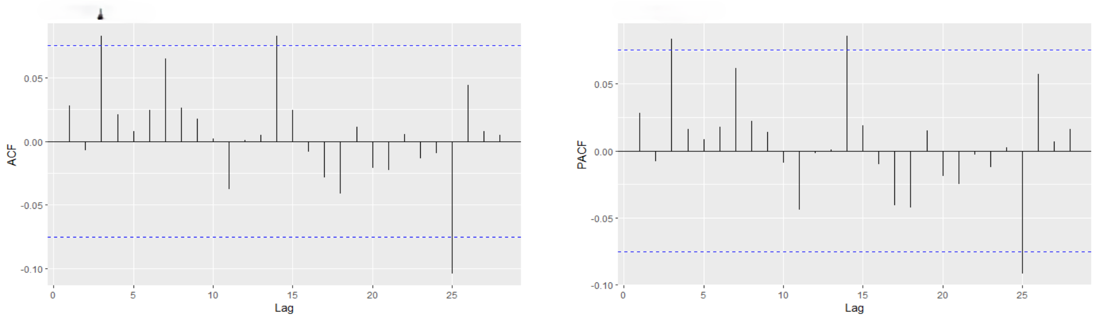


Figure 4: Autocorrelation and partial autocorrelation functions of Gbp/Usd training dataset after differencing.

remaining 52 as the testing set. The dataset has been analyzed by Zhang (2003), Khashei and Bijari (2011) and Büyükahin and Ertekin (2019). From Figure 3, we observe that the ACF does not exhibit exponential decay, and the PACF at lag 1 is close to 1, suggesting that differencing may be necessary to determine an appropriate model. Figure 4 shows the results after differencing, where the differenced PACF resembles the ACF, indicating that the series is stationary after differencing. Therefore, we consider an ARIMA(0,1,0) (a random walk), which is also suggested by Büyükahin and Ertekin (2019), for fitting the ARIMA model to predict the linear component of Algorithm AH and Algorithm HNN. In addition, for generating the low-volatility components in Algorithm AHMA and Algorithm SHNN, the length of the fixed MA filter is set as $m = 30$. As to the ANN model for each hybrid model, the tuning parameters are data-driven and are given in Table 13. Here we note that for the corresponding boosted hybrid models, the setting for the ARIMA and ANN parts in the algorithms are the same.

Now, we apply Algorithm MS-CV over the set $\mathcal{M}_1$ to select an optimal model in

Table 13: Hyper-parameter settings for the ANN model in the hybrid model of Gbp/Usd data.

Model	Neuron	Initializer	Activation function	Optimizer	Loss function
AH	$14 \times 13 \times 1$	uniform	Relu	RMSProp	MSE
HNN	$10 \times 9 \times 1$	uniform	Relu	RMSProp	MSE
AHMA	$9 \times 9 \times 1$	uniform	Relu	Adam	MSE
SHNN	$9 \times 9 \times 1$	uniform	Relu	RMSprop	MSE

Table 14: Overall ranking of Gbp/Usd data under different hybrid models and varying training and validation split ratios, where $R_\delta(M)$ and $\bar{R}(M)$ are defined in Algorithm MS-CV with $M \in \mathcal{M}_1$.

Model	$R_{0.5}(M)$	$R_{0.6}(M)$	$R_{0.7}(M)$	$R_{0.8}(M)$	$R_{0.9}(M)$	$\bar{R}(M)$
AH	5	4	1	4	1	15
B-AH	3	3	3	2	2	13
HNN	2	2	4	3	3	14
B-HNN	4	8	5	8	5	30
AHMA	1	1	2	1	4	**9**
B-AHMA	8	5	7	6	8	34
SHNN	7	6	8	7	7	35
B-SHNN	6	7	6	5	6	30

Table 15: The three metrics, MAE, MSE and MASE, for each model in $\mathcal{M}_1$ based on the Gbp/Usd testing dataset, where the Group variable represents the classification results of k-means method based on the three normalized metrics with the normalized center point $(0.0914, 0.1099, 0.0932)$ for the leading Group 1 and the normalized center point $(0.7675, 0.7534, 0.7676)$ for the lagging Group 2.

Model	MAE	MSE	MASE	Group
AH	0.0204	0.0007	0.9859	1
B-AH	0.0204	0.0006	0.9840	1
HNN	0.0207	0.0007	0.9990	1
B-HNN	0.0203	0.0006	0.9804	1
AHMA	0.0205	0.0007	0.9892	1
B-AHMA	0.0197	0.0006	0.9485	1
SHNN	0.0275	0.0011	1.3259	2
B-SHNN	0.0239	0.0009	1.1504	2

Table 16: Hyper-parameter settings for the ANN model in the hybrid model of Gbp/Usd data.

Model	Neuron	Initializer	Activation function	Optimizer	Loss function
EWAH	$14 \times 13 \times 1$	uniform	tanh	RMSProp	MSE
EWHNN	$20 \times 19 \times 1$	uniform	Relu	RMSprop	MSE
EWAHMA	$4 \times 4 \times 1$	uniform	Relu	Adam	MSE
EWSHNN	$10 \times 9 \times 1$	uniform	Relu	RMSprop	MSE

Table 17: Overall ranking of Lynx data under different hybrid models and varying training and validation split ratios, where $R_\delta(M)$ and $\bar{R}(M)$ are defined in Algorithm MS-CV with $M \in \mathcal{M}_2$.

Model	$R_{0.5}(M)$	$R_{0.6}(M)$	$R_{0.7}(M)$	$R_{0.8}(M)$	$R_{0.9}(M)$	$\bar{R}(M)$
EWAH	1	1	1	1	1	**5**
B-EWAH	6	6	3	3	8	26
EWHNN	2	2.5	7	2	6	19.5
B-EWHNN	3	8	4	4	4	23
EWAHMA	7	5	6	7	5	30
B-EWAHMA	8	7	8	8	7	38
EWSHNN	4	2.5	2	5	2	15.5
B-EWSHNN	5	4	5	6	3	23

Table 18: The three metrics, MAE, MSE and MASE, for each model in $\mathcal{M}_2$ based on the Gbp/Usd testing dataset, where the Group variable represents the classification results of k-means method based on the three normalized metrics with the normalized center point $(0.1785, 0.0707, 0.1870)$ for the leading Group 1 and the normalized center point $(0.8154, 0.8282, 0.8196)$ for the lagging Group 2.

Model	MAE	MSE	MASE	Group
EWAH	0.0198	0.0006	0.9559	1
B-EWAH	0.0191	0.0006	0.9187	1
EWHNN	0.0223	0.0008	1.0767	2
B-EWHNN	0.0212	0.0006	1.0198	1
EWAHMA	0.0240	0.0009	1.1584	2
B-EWAHMA	0.0247	0.0009	1.1913	2
EWSHNN	0.0206	0.0007	0.9956	1
B-EWSHNN	0.0201	0.0007	0.9717	1

overall prediction performance among the three metrics defined in (8). The ranks of the candidate models in $\mathcal{M}_1$ is illustrated in Table 14, which selects the B-AHMA model as the optimal model for the predicting the sunspots. To verify the validity of the B-AHMA model for the prediction, we re-fit each hybrid model on the entire training set and then calculate the prediction errors for three metrics on the testing set. As shown in Table 15, the predictive performance of the B-AHMA indeed falls into the leading group from using the k-means grouping method.

Next, we apply Algorithm MS-CV over the set $\mathcal{M}_2$ to select an optimal model in overall prediction performance among the three metrics defined in (8). Note that for models in $\mathcal{M}_2$ and their boosted variants, they share the same tuning parameters of ANN models that are data-driven and are given in Table 16.

The ranks of the candidate models in $\mathcal{M}_2$ is illustrated in Table 17, which selects the EWAH model as the optimal model for the predicting the sunspots. To verify the validity of the EWAH model for the prediction, we re-fit each hybrid model on the entire training set and then calculate the prediction errors for three metrics on the testing set. As shown in Table 18, the predictive performance of the EWAH indeed falls into the leading group from using the k-means grouping method.

5. Discussion

This study focuses on advancing time series analysis through the application of boosting methods, the EWMA smoothing filter, and various hybrid models. These techniques are employed to enhance the predictive performance of traditional linear ARIMA models and nonlinear ANN models. Note that in our hybrid methods, we adopt a “simultaneous construction” approach for the ANN component. After modeling the data’s linear structure with a preliminary model, we use an ANN to capture and predict the nonlinear patterns present in the residuals. This ANN model is trained on the entire residual series, focusing on learning the broader, overall nonlinear structure in the data, rather than fitting separate models for each individual time point. Thus, our approach does not involve sequentially building ANN models at each time point. Instead, we treat the residuals as a unified dataset to capture overarching nonlinear characteristics, ensuring the model’s robustness across the dataset. The study further incorporates a rigorous model selection procedure to evaluate the proposed hybrid models and their variants. Specifically, the impact of boosting on predictive accuracy is assessed by applying the boosting method to both the established hybrid models in the literature and their EWMA-modified counterparts.

The most robust predictive model is identified from each set based on performance metrics obtained from the testing set. A multiple split cross-validation approach is utilized for model selection; see more details in Algorithm MS-CV. This approach involves partitioning the entire training set into multiple smaller training subsets and validation sets to rank the models according to various prediction metrics. The model with the highest ranking is selected for prediction on the testing set. Subsequently, k-means clustering is employed to verify the performance of the selected models on the testing set across different data analyses. The empirical results illustrate the stability and effectiveness of the model selection procedure across all hybrid methods examined in this study. Note that Algorithm MS-CV selects models based on an overall ranking evaluation with equal weighting across various metrics. In the preliminary stages of our data analysis, we found that it was not possible to achieve consistent model selection results under different individual metrics. When considering different weights, the models selected by MS-CV are likely to differ. Therefore, adjusting the MS-CV algorithm with different weights on various metrics based on domain knowledge of the data represents an important task in the data analysis.

There is potential for further improvement of the boosted hybrid models. Future research could also explore alternative neural networks, such as Long Short-Term Memory (LSTM) networks, which offer advantages over traditional ANN models by better capturing and storing long-term time dependencies and selectively remembering or forgetting past information. Although LSTM networks are more complex and may complicate subsequent analysis, they could provide different predictive results when integrated with the MS-CV algorithm based on time series patterns and dependencies. Additionally, as suggested by Büyükahin and Ertekin (2019), incorporating multi-scale decomposition techniques such as empirical mode decomposition (EMD; Huang *et al.* 1998) could further enhance model performance. EMD decomposes a time series into intrinsic mode functions (IMFs), which can be analyzed individually by any of the hybrid models introduced in this study. Given that each IMF has a unique local time scale and is relatively stationary, aggregating predictions from each IMF is expected to yield improved performance. However, exploring these extensions is beyond the scope of this article and will be addressed in future research.

In summary, in this article, we propose an algorithm that consists of three main components: (i) a hybrid model incorporating both linear and nonlinear structural models; (ii) enhanced learning for the hybrid model; and finally, (iii) model selection based on the MS-CV algorithm. Each of these components can be adapted to utilize specific linear or nonlinear models in the algorithm's modules, based on the data's long-term trends or cyclic patterns. Additionally, for cross-validation of time series data, the Time Series Split method employed in this paper can be substituted with common validation methods in time series analysis, such as Rolling Forecast or Expanding Window. However, the effectiveness of these alternative methods in forecasting must be determined based on domain knowledge of the data or user requirements; thus, we do not attempt or adjust these alternatives individually in this paper. Rather, we propose a boosted hybrid model framework, allowing users to flexibly modify and replace components to achieve optimal analysis and forecasting results in various data analysis contexts.

References

[1] Babu, C. N. and Reddy, B. E. (2014). A moving-average filter based hybrid ARIMA–ANN model for forecasting time series data. *Applied Soft Computing*, 23, pages 27-38.

[2] Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3), pages 307-327.

[3] Box, G. E. P. and Jenkins, G. M. (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day.

[4] Büyükahin, Ü. Ç. and Ertekin, Ş. (2019). Improving forecasting accuracy of time series data using a new ARIMA–ANN hybrid method and empirical mode decomposition. *Neurocomputing*, 361(7), pages 151-163.

[5] Dickey, D. A. and Fuller, W. A. (1981). Likelihood ratio statistics for autoregressive time series with a unit root. *Econometrica*, 49(4), pages 1057-1072.

[6] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), pages 1735-1780.

[7] Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C., and Liu, H. H. (1998). The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society A*, 454(1971), pages 903-995.

[8] Khashei, M. and Bijari, M. (2011). A new hybrid methodology for nonlinear time series forecasting. *Modelling and Simulation in Engineering*, 2011(1), 379121.

[9] Roberts, S. W. (1959). Control chart tests based on geometric moving averages. *Technometrics*, 1(3), pages 239-250.

[10] Sarkar, T. (2022). XBNet: An extremely boosted neural network. *Intelligent Systems with Applications*, 15, 200097.

[11] Zhan, Z. and Yang, Y. (2022). Profile electoral college cross-validation. *Information Sciences*, 586, pages 24-40.

[12] Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50, pages 159-175.

[Received August 2024; accepted December 2024.]

提升混合法之非平穩數據分析

張志浩 [1†] **黃士峰** [2] **羅育聖** [3]

[1] 國立政治大學統計學系

[2] 國立中央大學統計研究所

[3] 國立高雄大學統計學研究所

摘　要

在這篇論文中，我們提出了一個對混合模型的提升方法，並應用於分析非平穩時間序列資料。混合模型通過將原始時間序列分解為平穩部分和非平穩部分，並分別使用線性統計模型（如自迴歸模型）和非線性機器學習技術（如神經網絡方法）進行擬合。為了提高這些混合模型的預測效果，我們提出了一種提升方法，旨在改善非線性模型對非平穩部分的預測。同時，我們使用指數加權移動平均法進行數據平滑處理，作為現有混合模型中使用濾波方法分解時間序列的競爭替代方法。我們使用多個指標評估了混合模型及其提升和濾波變體的分析效果。為了選出在預測分析上較為穩健的混合模型，我們實施了一種基於多次分割交叉驗證技術的模型選擇流程對所有候選模型的預測指標總分進行排名後進行選取。最後我們通過三項實證研究展示該選模流程在預測分析上的穩健性。

關鍵詞：提升方法、混合模型、線性模型、模型選取、神經網絡模型、非線性模型。

JEL classification: C53, E17.

[†]通訊作者：張志浩
E-mail: jhow@nccu.edu.tw