Machine Learning with Applications 22 (2025) 100781

Contents lists available at ScienceDirect

Machine Learning with Applications

journal homepage: www.elsevier.com/locate/mlwa

A comparative framework for multi-horizon time series forecasting: Neural networks with adaptive preprocessing

Ana Lazcano [a], Julio E. Sandubete [a,*], Miguel A. Jaramillo-Morán [b]

[a] Faculty of Law, Business and Government, Universidad Francisco de Vitoria, Madrid, Spain
[b] Department of Electrical Engineering, Electronics and Automation, University of Extremadura, Badajoz, Spain

ARTICLE INFO

Keywords:
Multi-horizon forecasting
Neural networks
Preprocessing methods
Multiple-input multiple-output (MIMO)
Economic time series

ABSTRACT

Accurate multi-horizon time series forecasting remains a major challenge in predictive modeling due to cumulative error propagation over extended horizons. This study proposes a unified and reproducible framework that integrates adaptive signal decomposition with neural network architectures under a Multiple-Input Multiple-Output (MIMO) strategy, effectively removing recursive dependencies and improving training stability. Three representative neural models Multilayer Perceptron (MLP), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM) are systematically combined with both classical and adaptive preprocessing techniques, namely trend–fluctuation separation, Empirical Mode Decomposition (EMD), Variational Mode Decomposition (VMD), and Empirical Wavelet Transform (EWT). The framework is validated on three economic and energy-related datasets (electricity demand, natural gas prices, and CO_2 emission allowances), generating forecasts up to ten steps ahead and evaluated through RMSE, MAPE, and R^2 metrics. Experimental results show that adaptive decomposition, particularly VMD and EMD, yield the highest accuracy and stability across prediction horizons, while EWT provides consistent intermediate improvements and classical trend-based methods offer only marginal benefits. Moreover, computational analysis demonstrates that the proposed approach remains lightweight and efficient, with training and inference times suitable for real-world deployment. Overall, the findings confirm that coupling adaptive preprocessing with MIMO-based neural forecasting enhances accuracy, robustness, and interpretability without increasing architectural complexity, establishing a practical foundation for multi-horizon forecasting in economic and financial domains.

1. Introduction

The development of increasingly efficient models for time series forecasting is one of the most intensively studied areas in the last years (Lim & Zohren, 2021; Elsayed et al., 2021; Sezer et al., 2020). The demand for greater predictive accuracy, particularly in economic and financial contexts, arises from increasingly volatile markets where reliable forecasts are essential (Shaub, 2020). In this regard, Artificial Neural Network (ANN)–based models, such as the Multilayer Perceptron (MLP), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM), have consolidated themselves as robust alternatives to traditional linear approaches, including AutoRegressive Integrated Moving Average (ARIMA) and related statistical techniques (Siami-Namini & Namin, 2018). Their ability to capture nonlinear dynamics, temporal dependencies, and hidden structures within the data makes them especially suitable for multi-horizon forecasting, where the challenge is to anticipate several future steps based solely on past observations.

The aim of time series forecasting is to provide a model that can predict the next value of the series by properly processing the N previous ones:

$$y_{t+1} = f(x_t, \ldots, x_{t-N+1}) \tag{1}$$

where x_t is the observation at time t, y_{t+1} a predicted value at $t+1$ and $f(\cdot)$ a function describing the predictor.

Traditional linear models, such as ARIMA and its variants, have long dominated the field of time series forecasting. However, their limited capacity to capture nonlinear relationships and long-term dependencies often leads to suboptimal performance in complex real-world datasets (De Gooijer & Hyndman, 2006; Zhang & Qi, 2005). In contrast, ANNs provide a flexible, data-driven alternative that can approximate highly nonlinear functions and adapt to heterogeneous patterns within the data

* Corresponding author.
E-mail addresses: ana.lazcano@ufv.es (A. Lazcano), je.sandubete@ufv.es (J.E. Sandubete), miguel@unex.es (M.A. Jaramillo-Morán).

https://doi.org/10.1016/j.mlwa.2025.100781
Received 29 September 2025; Received in revised form 31 October 2025; Accepted 31 October 2025
Available online 1 November 2025

Table 1
Comparison of several multi-step-ahead forecasting methodologies.

Method	Description	Advantage	Limitation
Direct	Independent model per horizon	No error propagation	Multiple models required
Recursive	Recursive one-step forecasting	Simple, widely used	Error accumulation
DirRec	Hybrid of direct and recursive	Reduces error propagation	Embedding grows with horizon
MIMO	Multi-output model	Preserves temporal dependence	One model for all horizons
DirMO	Block-wise multi-output	Combines direct & MIMO	Complex training

Table 2
Hyperparameters used for the preprocessing algorithms.

	Window (TF)	Peaks (W)	Wavelet (W)	IMFs (EMD)	IMFs (VMD)	Modes (EWT)
EII	5	5	Db5	10	6	5
Gas	5	5	Db5	10	6	5
CO2	5	5	Db5	9	6	5

(Shiblee et al., 2009). This capability makes them particularly suitable for forecasting tasks in volatile environments, where both short- and long-range dependencies play a crucial role.

This development of neural network-based models is allowing us to explore different aspects, from the amount of data used in predictions (Chen et al., 2023; Cerqueira et al., 2019), processing speed (Bontempi et al., 2012), and the time horizon of the predictions (Zeng et al., 2023). This value determines how many values can be predicted in a single iteration of the model. Increasing the accuracy of this type of predictions is beneficial in terms of operational efficiency in many aspects.

The prediction horizon is one of the greatest challenges when making time series predictions. Most of the state of the art focuses on algorithms that develop a single output prediction, a single time step (Htike, 2013), since from several past data points it is possible to obtain a result with a high fit in a single future prediction. However, in many circumstances, multi-horizon forecasting, which predicts multiple stages in the future, is preferable, as it can provide significant resource guidance and long-term decision-making (Khuntia et al., 2016; Soman et al., 2010). However, this approach poses a major challenge for prediction models based on neural networks, since, in most cases, the results obtained over multiple horizons come from both the real historical information contained in the time series and the output variables obtained in the time horizon itself (Li et al., 2019). That is, a prediction problem with H future steps is addressed by iterating H times from a one-step-ahead prediction. In this way, after the prediction of the future value is obtained, it serves as feedback for the next prediction, so that the predicted values are used as inputs instead of using the actual values, which are not available, assuming an increase in error propagation.

When talking about multi-horizon, or multi-step, time series predictions, also called long-term predictions, it involves predicting the next H values ($y_{N+1}, \cdots, y_{N+H}$) of a time series with historical data ($y_1, \cdots, y_N$) composed of N observations, where $H > 1$ determines the prediction time horizon.

Despite the growing number of comparative studies exploring various ANN architectures for time series forecasting, few works have systematically examined how data preprocessing, particularly signal decomposition techniques, affects model accuracy and stability across extended prediction horizons. Prior research has mainly compared network structures or training strategies, overlooking the potential interaction between preprocessing stages and neural architectures. Consequently, it remains unclear whether improvements in predictive performance are primarily driven by architectural complexity or by the quality of input signal decomposition. This gap is particularly relevant for multi-horizon settings, where error accumulation and signal distortion can severely affect forecasting reliability.

In this work, we contribute to the field of multi-horizon forecasting by empirically assessing how the integration of preprocessing techniques with neural network architectures can improve accuracy and stability across extended prediction horizons. Specifically, our study addresses two open questions left unresolved by previous comparative analyses: *(i)* To what extent can advanced decomposition-based preprocessing methods enhance predictive accuracy across different horizons? and *(ii)* How does the interaction between preprocessing and forecasting architecture affect error propagation and long-term stability?

To this end, we adopt the Multiple-Input Multiple-Output (MIMO) strategy (Bontempi, 2008) which avoids recursive error propagation by training the model to provide all the future predictions simultaneously. This choice provides a simple and computationally efficient framework that is particularly well suited for disentangling the methodological contribution of data preprocessing from that of the forecasting architecture.

While alternative approaches such as the Direct prediction strategy (Zhang et al., 2013; Hamzaçebi et al., 2009; Kline, 2004), the Recursive strategy (Parlos et al., 2000; Su et al., 1992; Ng & Young, 1990), Direct-Recursive (DirRec) (Sorjamaa & Lendasse, 2006), or Direct-Multi-Output (DirMO) (Taieb et al., 2010) suffer from either error accumulation or structural complexity, our study demonstrates that combining the MIMO strategy with advanced decomposition methods and neural networks yields more robust and interpretable forecasts over longer horizons.

Therefore, the core innovation of this work lies in the design and validation of a unified experimental framework that systematically integrates adaptive signal decomposition with direct multi-horizon neural forecasting. Unlike prior studies that examined these components separately, our approach isolates the impact of preprocessing under a controlled MIMO setting, thereby revealing the specific contribution of decomposition techniques (EMD, VMD, and EWT) to the long-term stability of forecasts. This integration constitutes both a methodological and empirical advancement, as it establishes a replicable basis for evaluating the interplay between data transformation and neural modeling in complex economic time series.

Table 4
Time consumed to perform the decompositions proposed in this work. Time in seconds.

Model	TF	W	EMD	VMD	EWT
Time	0.0014	0.0026	0.0097	0.5545	0.0157

Table 3
Significant hyperparameters adjusted for the three neural models used. HL stans for “Hidden Layer” an OL for “Output Layer”.

Model	Hidden layers	Neurons in HL	Neurons in OL	Input data	Epochs	Minibatch size	Learning rate	Training
MLP	1	10	10	10	100			Levenberg-Marquard
LSTM/ BiLSTM	1	10	10	1	100	128	0.005	Adam

Table 5
Time consumed to perform training and validation with the MLP for each preprocessing technique. The time consumed for the direct prediction is also provided for comparison. Time in seconds.

MLP	Direct	TF	W	EMD	VMD	EWT
Training	7.51782	15.57213	15.5256	185.067	169.205	165.559
Validation	0.00613	0.012462	0.01240	0.13493	0.08418	0.08302

Table 6
Time consumed to perform training and validation with the LSTM for each preprocessing technique. The time consumed for the direct prediction is also provided for comparison. Time in seconds.

LSTM	Direct	TF	W	EMD	VMD	EWT
Training	1.22676	2.46107	2.30463	19.4984	16.041	16.6557
Validation	0.00937	0.01888	0.01762	0.13847	0.10384	0.11029

Table 7
Time consumed to perform training and validation with the BiLSTM for each preprocessing technique. The time consumed for the direct prediction is also provided for comparison. Time in seconds.

BiLSTM	Direct	TF	W	EMD	VMD	EWT
Training	2.496169	4.94243	4.60177	35,97,652	31,02,291	30,41,048
Validation	0.011524	0.02118	0.02126	0403,852	0357,653	0135,607

The remainder of the article is organized as follows. In Section 2, we present some of the basic techniques for multi-step-ahead time series prediction using ANN. Section 3 presents the methodology employed, explaining the preprocessing techniques and neural models used. Section 4 presents several experiments to compare the performance of the preprocessing methods and different time horizons. Section 5 offers discussions, conclusions, and observations for future work.

2. Multi-step forecasting methods

To delve deeper into multi-horizon forecasting, it is helpful to explore previous work on time series forecasting and its evolution over recent decades. Knowing that a time series is a sequence of historical measurements x_t of an observable variable at equal time intervals, we will explore the literature on time series forecasting and delve deeper into multi-horizon forecasting.

Time series forecasting has historically been the domain of linear statistical models such as ARIMA, their limitations becoming apparent in the 1980s, a period coinciding with the development of nonlinear models such as the heteroskedastic conditional autoregressive (ARCH) model (Engle, 1982). Subsequently, machine learning models gained a foothold in the scientific community, consolidating themselves as solid proposals in the field of time series forecasting, managing to surpass the results of classical models (Mahmud et al., 2025; Nasir et al., 2025; Gonçalves et al., 2023).

Among the different forecasting tools available in this field, neural networks appear as the most widely used for time series forecasting for their flexibility, robustness, and accuracy. They seem to be the more flexible option to carry out multi-step-ahead predictions. (Torres et al., 2021; Cheng & Wang, 2020). This is why we will use them as the prediction tool in this work.

The growing interest in using neural networks for multi-horizon time series forecasting, driven by their flexible learning capabilities, has led to experimentation with various deep learning approaches (Wang et al., 2022), showing significant performance improvements over traditional statistical models (Alaa & van der Schaar, 2019; Makridakis et al., 2020). Although many of these architectures are grounded in variations of RNNs, more recent research has incorporated attention mechanisms to optimize the selection of the most relevant time steps (Fan et al., 2019), particularly in Transformer-based models (Li et al., 2019).

In parallel, Graph Neural Networks (GNNs) have emerged as a promising alternative for capturing spatial or structural dependencies among multiple interconnected time series (Rangapuram et al., 2018). However, both attention-based and graph-based models present practical limitations (Grassi, 2024), either due to their high computational demands or the difficulty of generalizing their performance in contexts with limited data or without an explicit network structure.

For multi-step-ahead prediction several techniques have been proposed and one of them should be selected to deal with the problem at hand. According to research conducted by Taieb et al (2010), we can distinguish 5 types of algorithms for multi-horizon predictions. To provide a clearer overview of the methodological landscape, Table 1 summarizes the main multi-step forecasting strategies, highlighting their advantages and limitations as reported in the literature.

Each of these methods presents specific advantages, but they also involve important limitations when applied to long forecasting horizons.

Table 8
RMSE metric for the EII time series at different time steps.

Metrics							
Model	N° steps	Direct	TF	W	EMD	VMD	EWT
MLP	1	6602	6486	5428	5457	1714	4260
	5	15,470	15,664	16,163	9821	3828	5374
	10	18,605	15,578	17,486	12,012	4922	6656
LSTM	1	8505	7564	9404	5709	3098	4834
	5	15,722	19,636	17,529	11,949	6556	7107
	10	14,981	17,933	16,354	13,238	6224	8838
BiLSTM	1	5771	4170	4824	4073	2290	3291
	5	6671	5274	5241	4222	3600	3552
	10	9666	6668	7780	6438	4429	4174

Table 9
RMSE metric for the NG time series at different time steps.

Metrics							
Model	N° steps	Direct	TF	W	EMD	VMD	EWT
MLP	1	0740	0974	0651	0611	0310	0568
	5	1135	0865	1051	0728	0450	0790
	10	1,40	1027	1410	0848	0678	0864
LSTM	1	0545	0621	0560	0648	0345	0610
	5	0848	0881	0876	1038	0478	0863
	10	0949	0972	0975	1029	0746	1054
BiLSTM	1	0567	0495	0529	0515	0306	0511
	5	0625	0567	0651	0593	0436	0536
	10	0711	0648	0689	0586	0492	0573

Table 10
RMSE metric for the CO_2 emission allowance price time series at different time steps.

Metrics							
Model	N° steps	Direct	TF	W	EMD	VMD	EWT
MLP	1	22,593	6435	39,631	1667	21,797	2777
	5	57,345	28,545	48,662	2411	21,288	5301
	10	81,953	104,801	79,531	3080	17,393	5909
LSTM	1	13,016	17,234	13,790	3457	15,990	12,286
	5	17,077	14,324	13,955	3461	17,428	11,253
	10	18,750	18,450	19,540	3971	15,948	12,185
BiLSTM	1	24,668	17,773	23,959	3746	19,619	8617
	5	24,654	23,874	25,180	4446	29,102	11,473
	10	21,674	28,348	29,413	4389	23,164	11,611

In Direct methods, H (number of future predictions) forecasting models are defined to predict each time step of the time horizon. This results in a very complex structure that demands a lot of computation resources. Recursive forecasting suffers from progressive error accumulation, which severely deteriorates accuracy as the time horizon increases. MIMO models define a unique forecasting model that provides all the predictions for the entire time horizon at once, resulting in a likely large forecasting structure. Hybrid methods such as DirRec, which combines direct and recursive strategies, add further complexity, but does not avoid the problem of error accumulation. Similarly, DirMO strategies require predicting several time horizons simultaneously using multiple MIMO models. This creates a complex structure that must be defined through trial and error since the optimal number of MIMO models and time horizons cannot be determined a priori. Therefore, the MIMO method appears as the simplest structure that can avoid the error accumulation problem.

For these reasons, in this study we adopt the MIMO method. Under this approach, one only forecasting model is trained to forecast several future data, which allows predictions to be generated without relying on previously forecasted values. By avoiding recursive dependency, this method eliminates the propagation of errors and provides a straightforward yet effective framework for analyzing the contribution of preprocessing techniques and neural architectures to multi-step forecasting. In addition, one only forecasting model needs be used providing a very simple structure, what improves robustness, facilitates interpretability, and offers a clear methodological basis to isolate the effects of our proposed innovations.

In this method a forecasting model is defined to processes N past data $(x_t, \ldots, x_{t-N+1})$ to provide H future predictions $\left(y_{t+1,\ldots}y_{t+H}\right)$ at once:

$$\left(y_{t+1,\ldots}y_{t+H}\right) = f(x_t, \ldots, x_{t-N+1}) \tag{2}$$

In this case, there is no accumulation of errors, since forecasted values are not used to make the predictions. In addition, only one forecasting model must be defined, which allows an easier adjustment of its parameters that facilitates the search for the structure performing the best and allows for an easier interpretation of the results.

3. Methodology

In this work, we propose a strategy that combines the time series preprocessing with a forecasting tool to provide multistep predictions. The forecasting models will be neural networks since they have been proven to predict time series and are often the preferred tool for time series forecasting (Agarwal et al., 2025; Mounir et al., 2023) accurately and reliably.

Two models will be tested: a feedforward network with no feedback among neurons, the Multilayer Perceptron, and a recurrent model with feedback loops to leverage the temporal relationship between the data of the time series presented as inputs, the Long Short-Term Memory. A third model, a more sophisticated version of the LSTM, will also be tested. These two neural network models were selected for their proven reliability and effectiveness in time series prediction, especially when working with datasets that have undergone prior transformations (Khedhiri, 2022; Paoli et al., 2010).

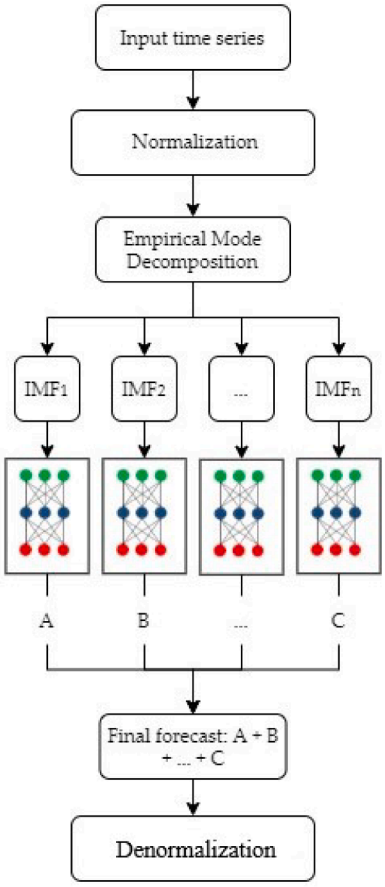

Fig. 1. Structure of the forecasting model proposed in this work. It has been peculiarized for the case of decomposition with EMD for better understanding. Why do all the figures appear in the middle of the appendix? We understand that in the final version they appear throughout the paper, distributed in the section corresponding to them, right?.

The need to preprocess the time series data before forecasting arises from the fact that many time series exhibit highly complex behaviors that hinder the ability of forecasting models to provide reliable and accurate predictions when applied directly. Numerous studies have demonstrated that preprocessing data before forecasting can significantly improve the performance of a forecasting tool (Li et al., 2019; Moghaddam et al., 2016;).

The purpose of this preprocessing is to transform the original time series into one or more subseries that are easier to predict. This strategy is particularly advantageous when the data have periodic or quasi-periodic components because preprocessing allows the series to be decomposed into subseries that can be associated with specific frequency components, making them easier to predict. Therefore, several preprocessing algorithms will be tested in this work along with the neural models. They range from the simple decomposition of the time series into its trend and fluctuations around it (two strategies will be proposed: to extract the trend by means of a moving average and by a noise rejection filter) to more sophisticated decomposition tools that decompose the series into subseries closely related to frequencies defining oscillatory modes, such as the Empirical Mode Decomposition (EMD), the Variational Mode Decomposition (VMD), and the Empirical Wavelet Transform (EWT).

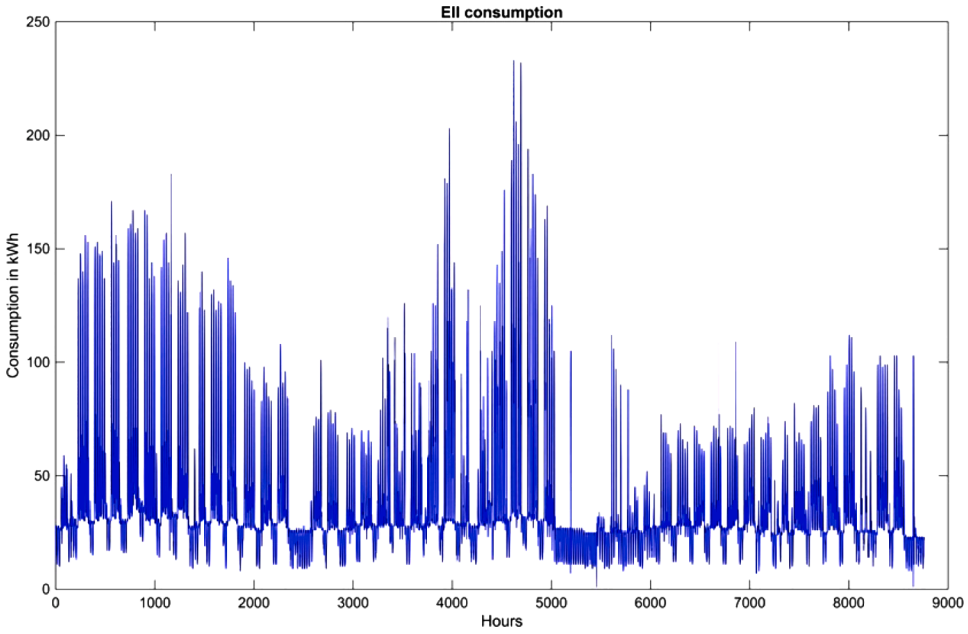

Fig. 2. EII time series prices. The figure displays the electricity consumption of the School of Industrial Engineering (EII) at the University of Extremadura.

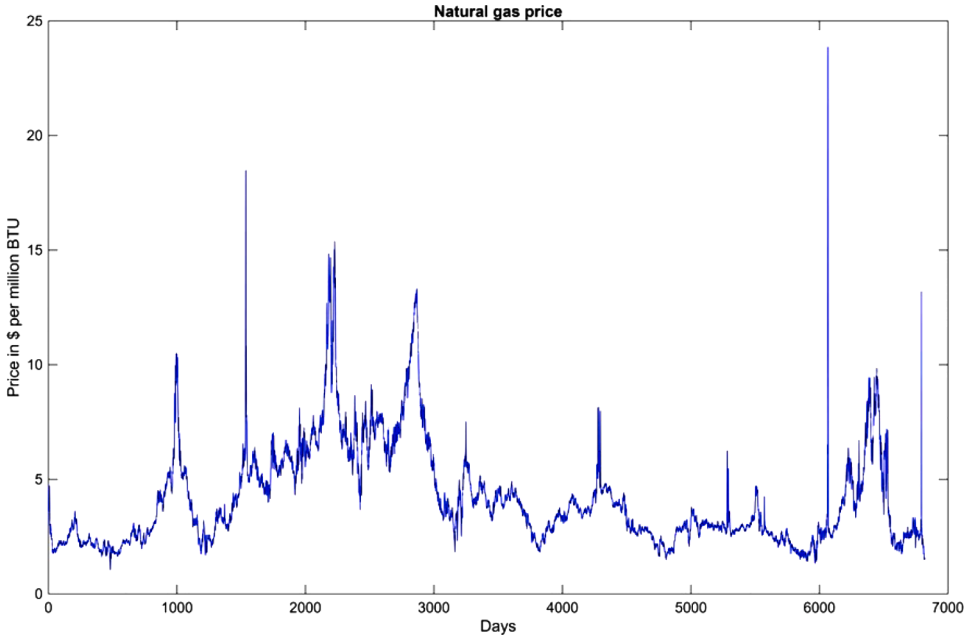

Fig. 3. Natural gas time series prices. The figure displays the daily Natural Gas prices (NG) recorded in the European energy market over the observation period.

The proposed forecasting model is a sequential structure which starts by normalizing the original time series values for better processing. Then, depending on the method, the normalized time series is decomposed into two or more subseries using one of the proposed preprocessing methods. Each subseries is forecasted independently using one of the selected neural networks. The predictions are then summed up to obtain an overall forecast, which is denormalized to predict the original time series.

The normalization and denormalization process is necessary for time series forecasting with a neural network because it adapts the dataset's range of values to the range of the neural network's inputs and outputs. Without this process, the neural model could fail to predict extreme values or have difficulty with time series that have a large range of values.

Fig. 1 illustrates this structure of the proposed forecasting model, in which the decomposition process is particularized to EMD for a better understanding of the model.

This workflow is consistently applied across all preprocessing techniques tested in this study (TF, W, EMD, VMD, and EWT), with only the decomposition stage varying according to the selected method. Fig. 1 illustrates the general structure of this MIMO-based forecasting framework, exemplified with the EMD case for clarity. Once this unified workflow was established, we proceeded to evaluate the performance of several neural architectures, specifically, Multilayer Perceptron, Long Short-Term Memory, and Bidirectional LSTM to analyze how different network topologies interact with the preprocessed inputs and influence multi-horizon prediction accuracy. Subsequently, we detail each of these models, followed by the mathematical description of the

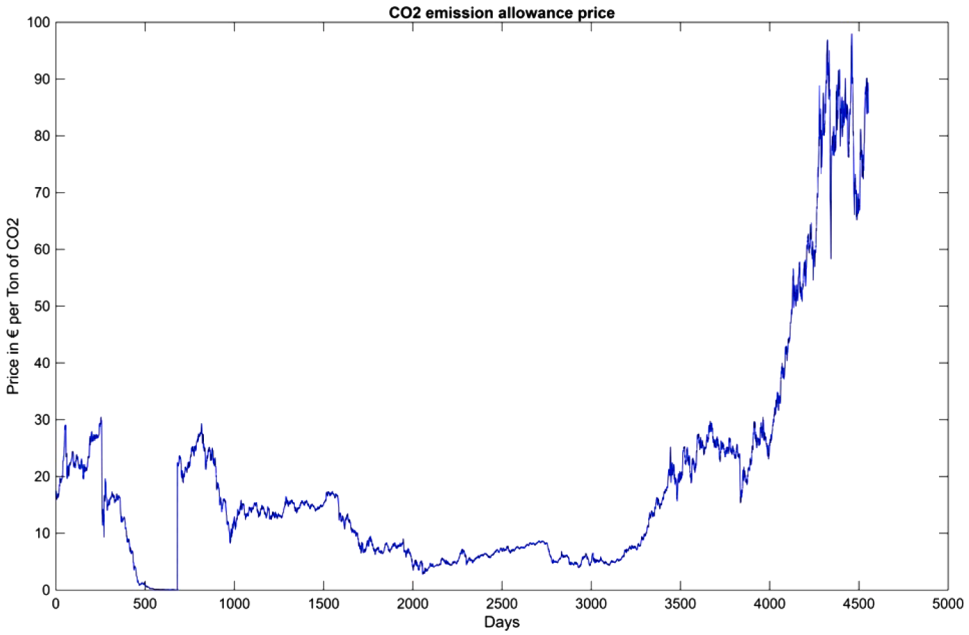

Fig. 4. CO_2 emission allowance time series prices. The figure displays the daily CO_2 emission allowance prices obtained from the European Energy Exchange (EEX) throughout the observation period.

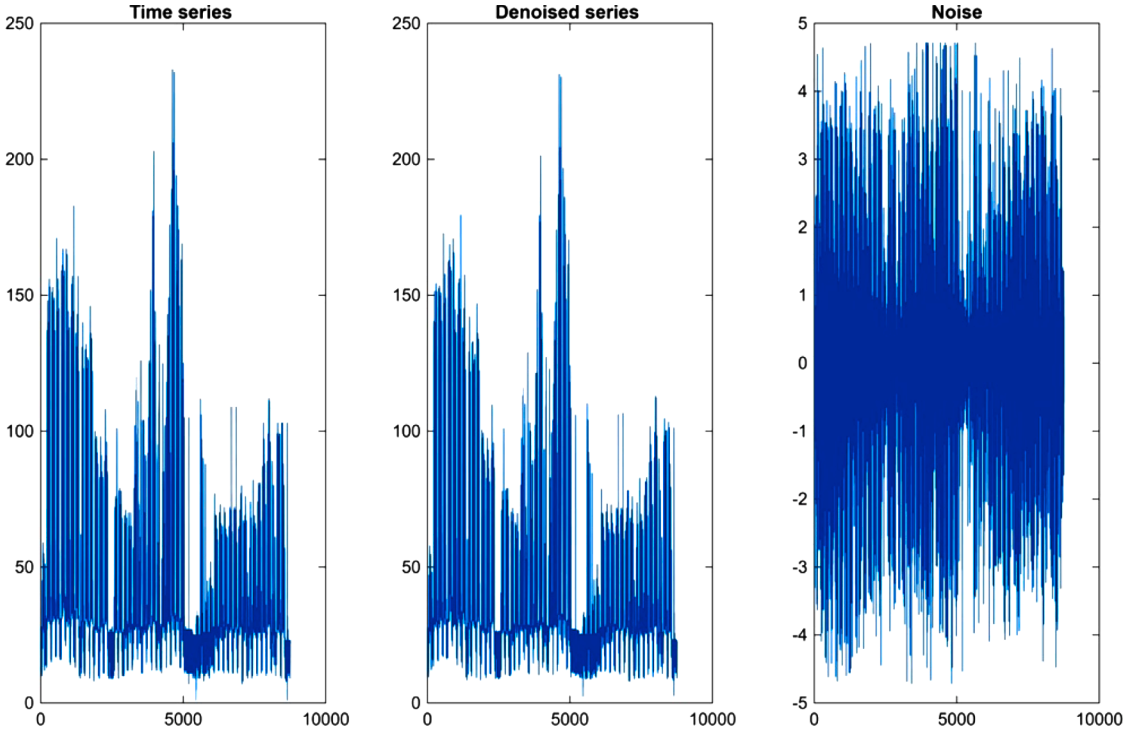

Fig. 5. Decomposition of the EII time series into trend and fluctuations using the moving average method. The x-axis represents the time index (observations), while the y-axis indicates the corresponding signal amplitude of the EII series.

preprocessing algorithms employed.

3.1. Multilayer perceptron (MLP)

Within the MIMO framework, the MLP maps an input window of length N to a vector of H future values: (). The MLP consists of an input layer, one or more hidden layers with nonlinear activation functions, and an output layer of dimension H. The model learns a nonlinear mapping that approximates the function between past and future observations: .

The proposed architecture consists of one hidden layer with ten neurons and an output layer of ten neurons, corresponding to the ten-step prediction horizon. This configuration was selected after testing multiple layer and neuron combinations, as it provided the most stable accuracy across datasets (see Section 5.1). The network uses the Levenberg–Marquardt algorithm for training, which is particularly efficient for shallow feedforward networks implemented in MATLAB. The activation function in the hidden layer is the hyperbolic tangent sigmoid,

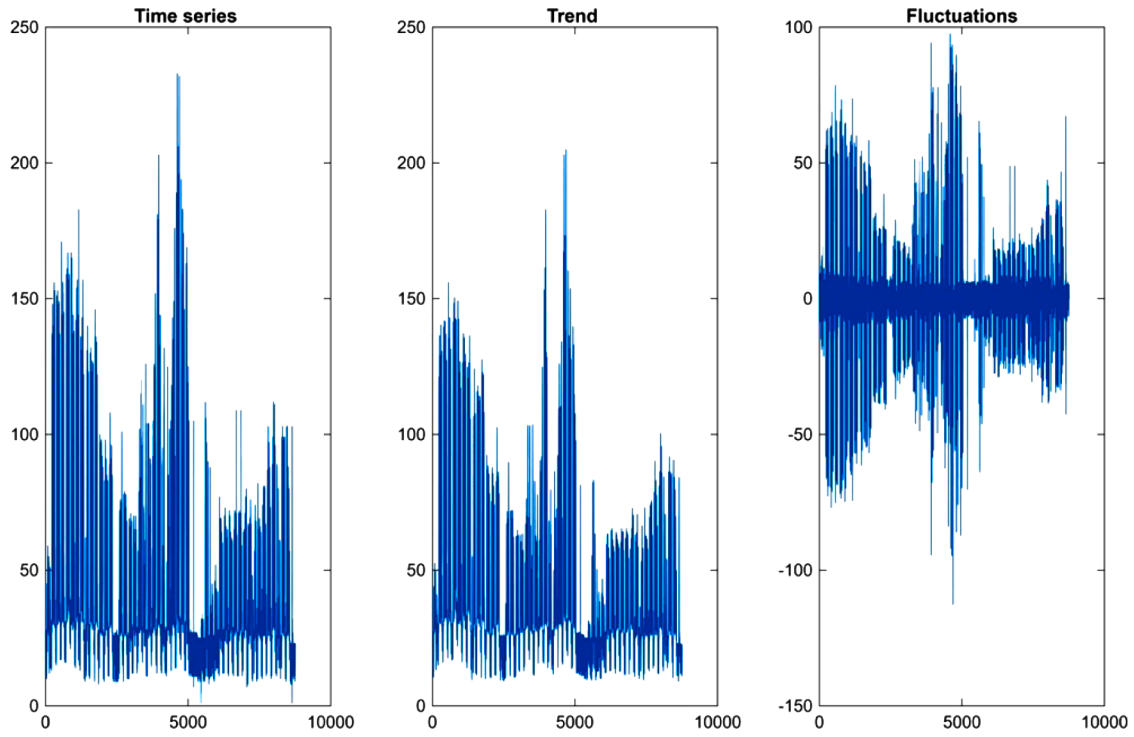

Fig. 6. Decomposition of the EII time series into trend and fluctuations using the wavelet denoising method (db5 wavelet, level 1). The x-axis represents the time index (observations), while the y-axis indicates the signal amplitude of the EII series.

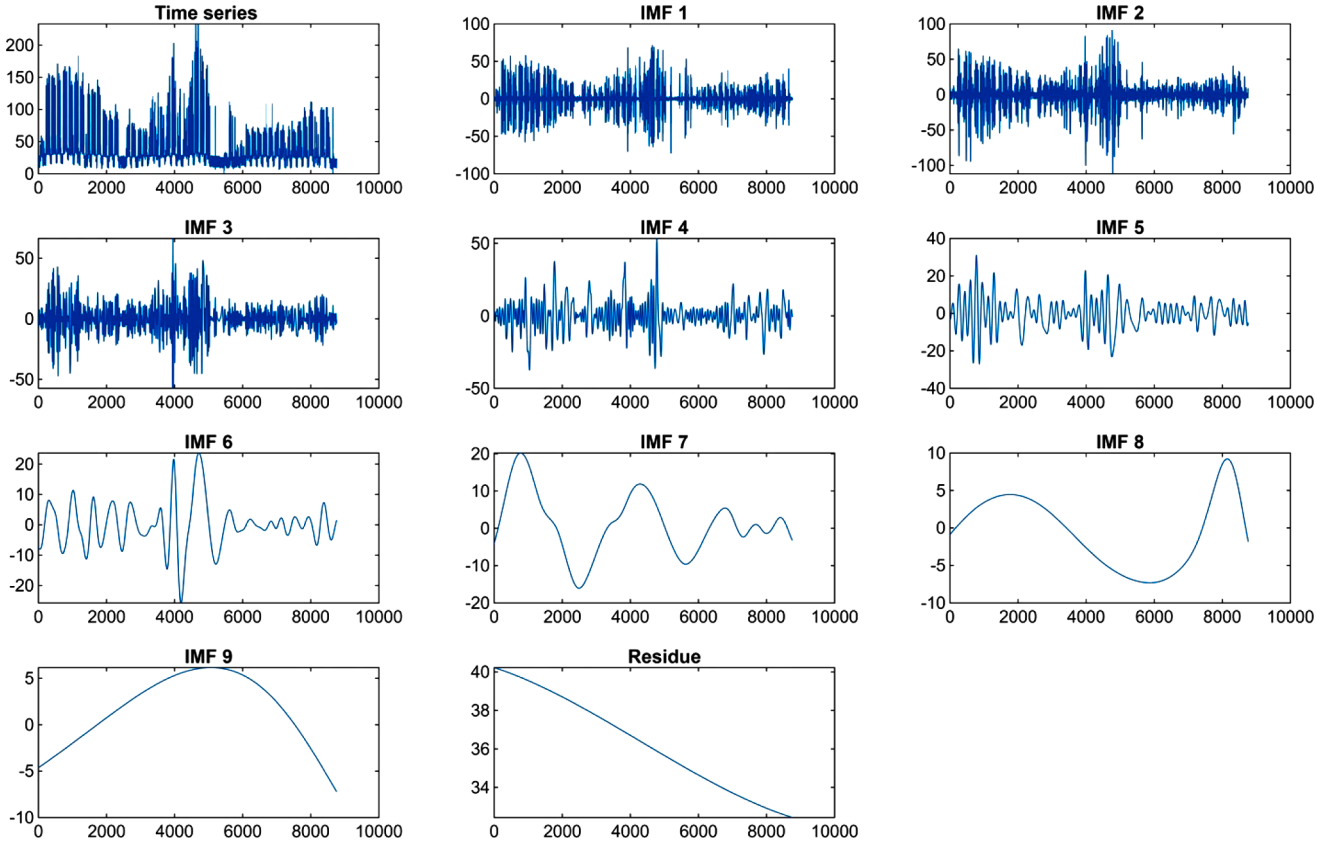

Fig. 7. Decomposition of the EII time series into Intrinsic Mode Functions (IMFs) using the Empirical Mode Decomposition (EMD) method. The x-axis represents the time index (observations), and the y-axis corresponds to the amplitude of each extracted IMF component.

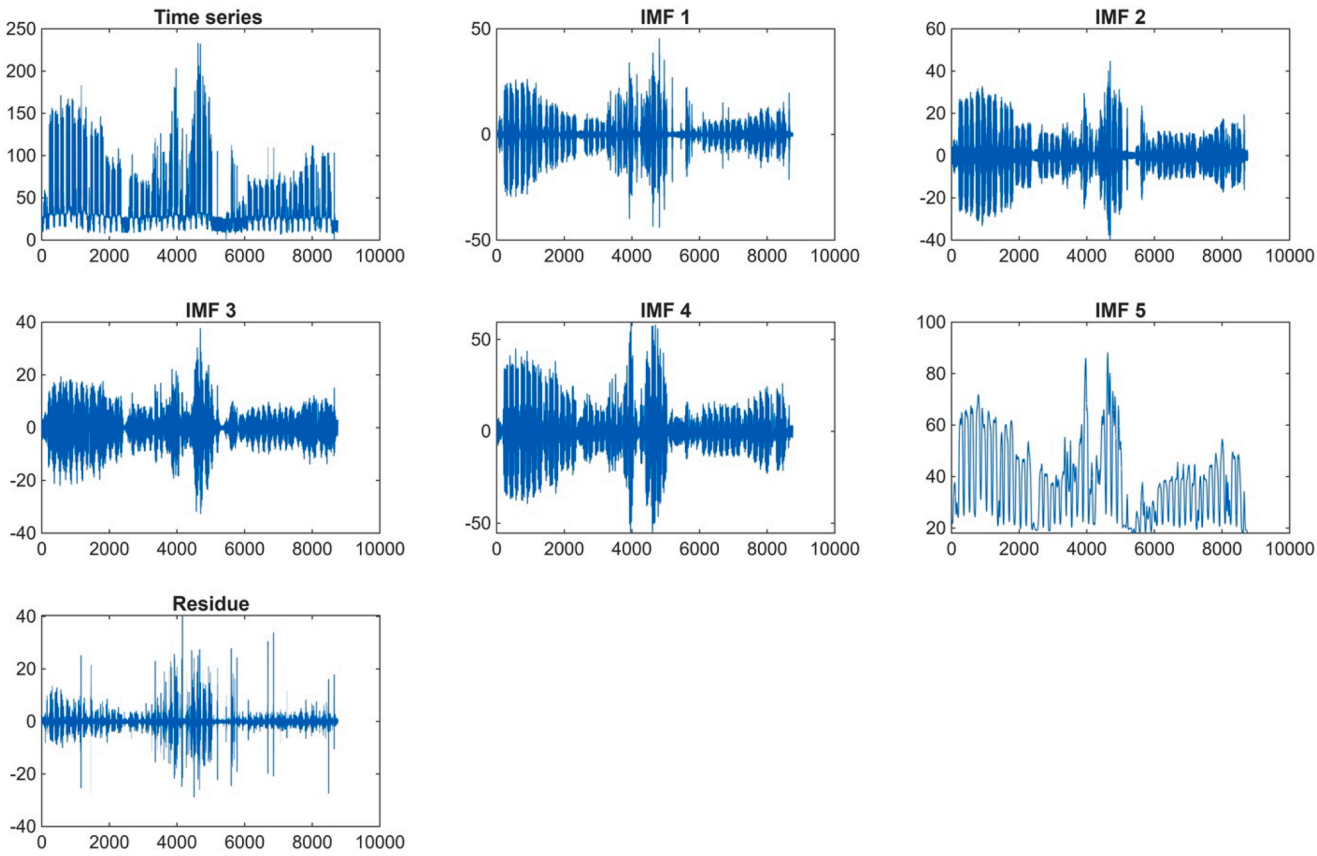

Fig. 8. Decomposition of the EII time series into modes using the Variational Mode Decomposition (VMD) technique. The x-axis represents the time index (observations), while the y-axis shows the amplitude of each VMD mode extracted from the original signal.

and the output layer uses a linear activation to predict continuous values. The model is trained for up to 100 epochs, as convergence is consistently achieved before this limit. Classical MLP formulations are provided in Appendix A (Hornik et al., 1989; Zhang et al., 2022).

3.2. Long short-term memory (LSTM)

The LSTM is a recurrent neural network capable of capturing temporal dependencies by maintaining internal memory states. Unlike MLPs, LSTMs process sequences step-by-step, allowing information to persist through time. In our framework, a stacked LSTM directly outputs H steps without recursion (MIMO configuration), avoiding cumulative error propagation. Input windows of length N are used, and the model outputs H values per sample: $\hat{y}_{t+1:t+H} = \mathrm{LSTM}(x_t, x_{t-1}, \ldots, x_{t-N+1})$.

The model includes a single LSTM layer with ten units followed by a fully connected output layer of size ten. Training is performed using the Adam optimizer with a learning rate of 0.005, batch size 128, and early stopping to prevent overfitting. This design efficiently captures both short- and long-term dependencies while maintaining computational simplicity and comparability with the MLP. Gate equations governing input, forget, and output dynamics are summarized in Appendix A.

3.3. Bidirectional LSTM (BiLSTM)

The BiLSTM extends the unidirectional LSTM by processing the input sequence in both forward and backward directions, allowing the network to learn from past and future contexts simultaneously. The architecture consists of one bidirectional LSTM layer with ten units in each direction (20 total outputs concatenated), followed by a linear layer of ten neurons to produce the final forecast. That is, the BiLSTM replaces the LSTM encoder with bidirectional layers to capture patterns with asymmetric temporal dependencies while still producing a single H-dimensional output via a linear head (MIMO configuration): $\hat{y}_{t+1:t+H} = \mathrm{Linear}([\rightarrow\mathrm{LSTM}(x_t), \leftarrow\mathrm{LSTM}(x_t)])$.

As with the LSTM, the BiLSTM is trained using the Adam optimizer under the same settings (learning rate 0.005, batch size 128, 100 epochs). This configuration balances model expressiveness and training efficiency, ensuring consistent comparison with the other architectures (Siami-Namini et al., 2019; Kim & Moon, 2019). Formal bidirectional recurrences and references appear in Appendix A.

3.4. Data preprocessing

3.4.1. Trend-flutuations decomposition

Trend-flutuations decomposition separates a time series into low-frequency (trend) and high-frequency (fluctuation) components, allowing independent modeling of each. The trend is estimated using a moving average: $T(t) = (1/(N+1)) * \Sigma_{i=0}^{N} x(t-i)$ while fluctuations are obtained as $F(t) = x(t) - T(t)$.

Both $T(t)$ and $F(t)$ can be forecasted independently, improving the capacity of neural networks to learn specific behaviors. In our implementation, these components serve as parallel inputs to the MIMO networks, and after prediction, their outputs are recombined to produce the final forecast at the original scale.

Another option to carry out the trend-fluctuation decomposition is to obtain the trend series from a noise-rejection filter developed with the wavelet transform. The fluctuation series will be the corresponding noise series obtained by subtracting the denoised series from the original one.

The mathematical formulation and complete derivation of the moving average–based trend and the noise-rejection trend decompositions are provided in Appendix A for reference.

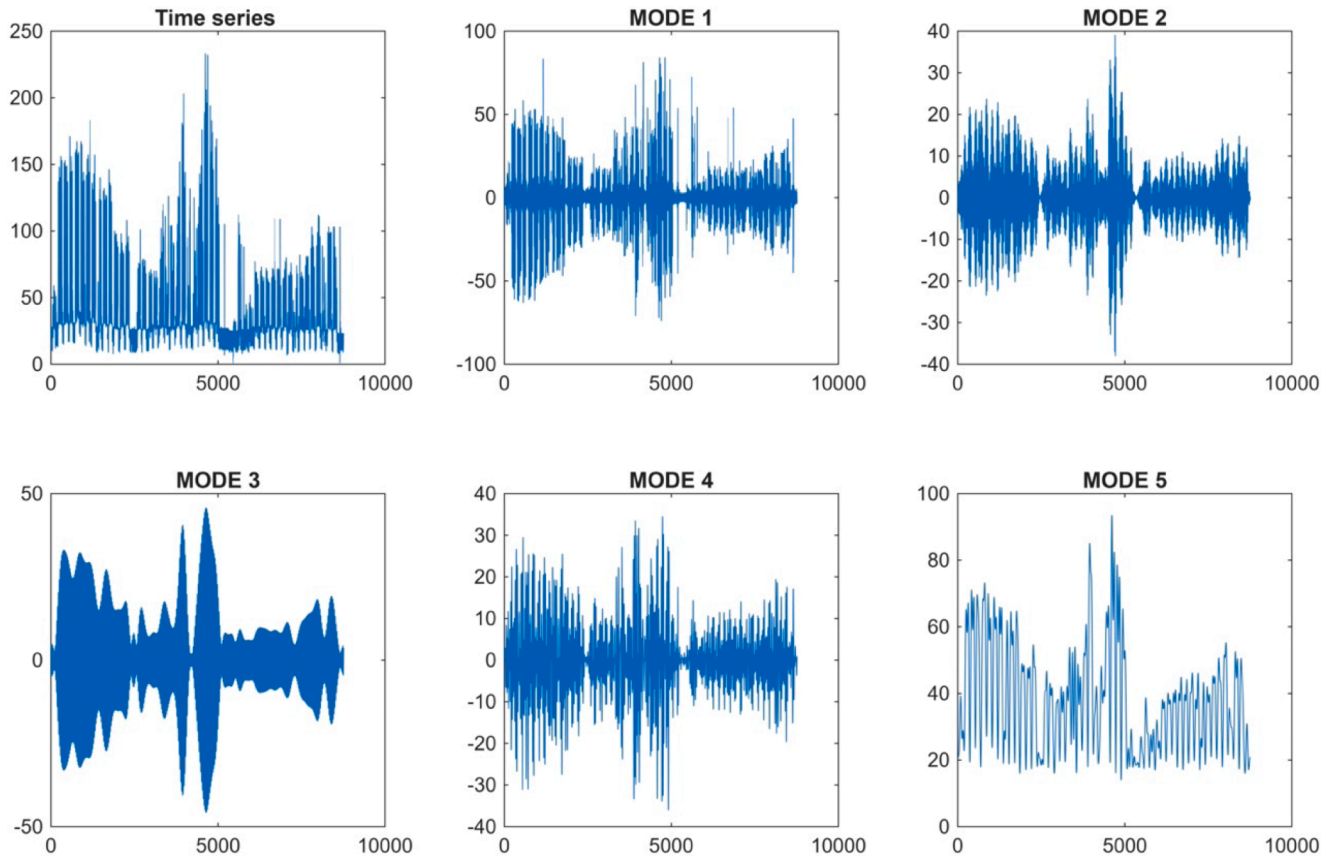

Fig. 9. Decomposition of the EII time series into spectral modes using the Empirical Wavelet Transform (EWT) method. The x-axis represents the time index (observations), and the y-axis displays the amplitude of each EWT mode derived from the original signal.

3.4.2. Empirical model decomposition

EMD decomposes a time series into a finite set of Intrinsic Mode Functions (IMFs) and a residual term. Each IMF represents oscillatory behavior at a characteristic time scale. The decomposition is adaptive and data-driven, allowing nonlinear and non-stationary signals to be effectively represented.

Formally, the original signal can be reconstructed as: $x(t) = \Sigma_i c_i(t) + r(t)$. In our implementation, each IMF and the residual are fed as separate input channels to the forecasting model. The forecasts for each component are recombined by summation, which yields the final multi-horizon prediction. A full description of the EMD sifting process and the corresponding equations governing the extraction of intrinsic mode functions can be found in Appendix A.

3.4.3. Variational mode decomposition

VMD is an optimization-based technique that decomposes the input signal $\mathrm{x(t)}$ into several modes $\mathrm{u_i(t)}$, each associated with a narrow frequency band centered around ω_i. The signal reconstruction is given by: $\mathrm{x(t)} = \Sigma_{i=1}^{\mathrm{K}} \mathrm{u_i(t)} = \Sigma_{i=1}^{\mathrm{K}} \mathrm{A_i(t)} * \cos(\varphi_\mathrm{i}(\mathrm{t}))$ and $\omega_\mathrm{i}(\mathrm{t}) = \partial\varphi_\mathrm{i}(\mathrm{t}) / \partial \mathrm{t}$.

We tune the number of modes K and the penalty factor α via cross-validation. Each mode is standardized and treated as an independent channel for the forecasting model. After prediction, the reconstructed signal is obtained by summing the predicted modes. The complete variational formulation of the VMD algorithm, including the augmented Lagrangian and ADMM optimization steps, is detailed in Appendix A.

3.4.4. Empirical wavelet transform

EWT performs a spectral decomposition of $\mathrm{x(t)}$ by constructing adaptive wavelet filters based on empirical frequency boundaries. The process divides the Fourier spectrum into N bands, each defining a wavelet $\psi_\mathrm{i}(\mathrm{t})$ and a scaling function $\varphi_1(\mathrm{t})$. The decomposition and reconstruction follow: $\mathrm{x(t)} = \mathrm{x_0(t)} + \Sigma_{i=1}^{N-1} \mathrm{x_i(t)}$ where $\mathrm{x_0(t)} = \mathrm{W_f^\varepsilon(0,t)} * \varphi_1(\mathrm{t})$ and $\mathrm{x_i(t)} = \mathrm{W_f^\varepsilon(i,t)} * \psi_\mathrm{i}(\mathrm{t})$.

Empirical modes are used as inputs to the forecasting models. High-frequency, low-energy modes can be discarded to reduce noise. All transformations are fitted using the training set only, ensuring that no data leakage occurs, and inverse transforms are applied to reconstruct forecasts on the original scale. The formal definitions of the empirical wavelet filters and the reconstruction equations used in the EWT are comprehensively presented in Appendix A.

4. Experimentation

Throughout this research, we have attempted to identify methodologies capable of producing multi-horizon predictions without compromising error metrics, eliminating the techniques traditionally used for this purpose, which were typically associated with error accumulation.

This study aims to identify forecasting methodologies capable of generating stable multi-horizon predictions while minimizing the propagation of error. Specifically, it explores whether combining preprocessing techniques with direct multi-horizon prediction strategies can improve the accuracy and stability of neural network–based forecasting models.

Using preprocessing techniques in combination with a direct multi-horizon prediction technique will demonstrate that greater stability in error metrics can be achieved, thereby simplifying the process of obtaining multi-horizon predictions in a single forecasting step.

The central hypothesis is that integrating signal decomposition or transformation techniques before training enhances the model's learning process, allowing it to generate multi-horizon forecasts in a single step without the need for recursive prediction and without cumulative degradation of performance.

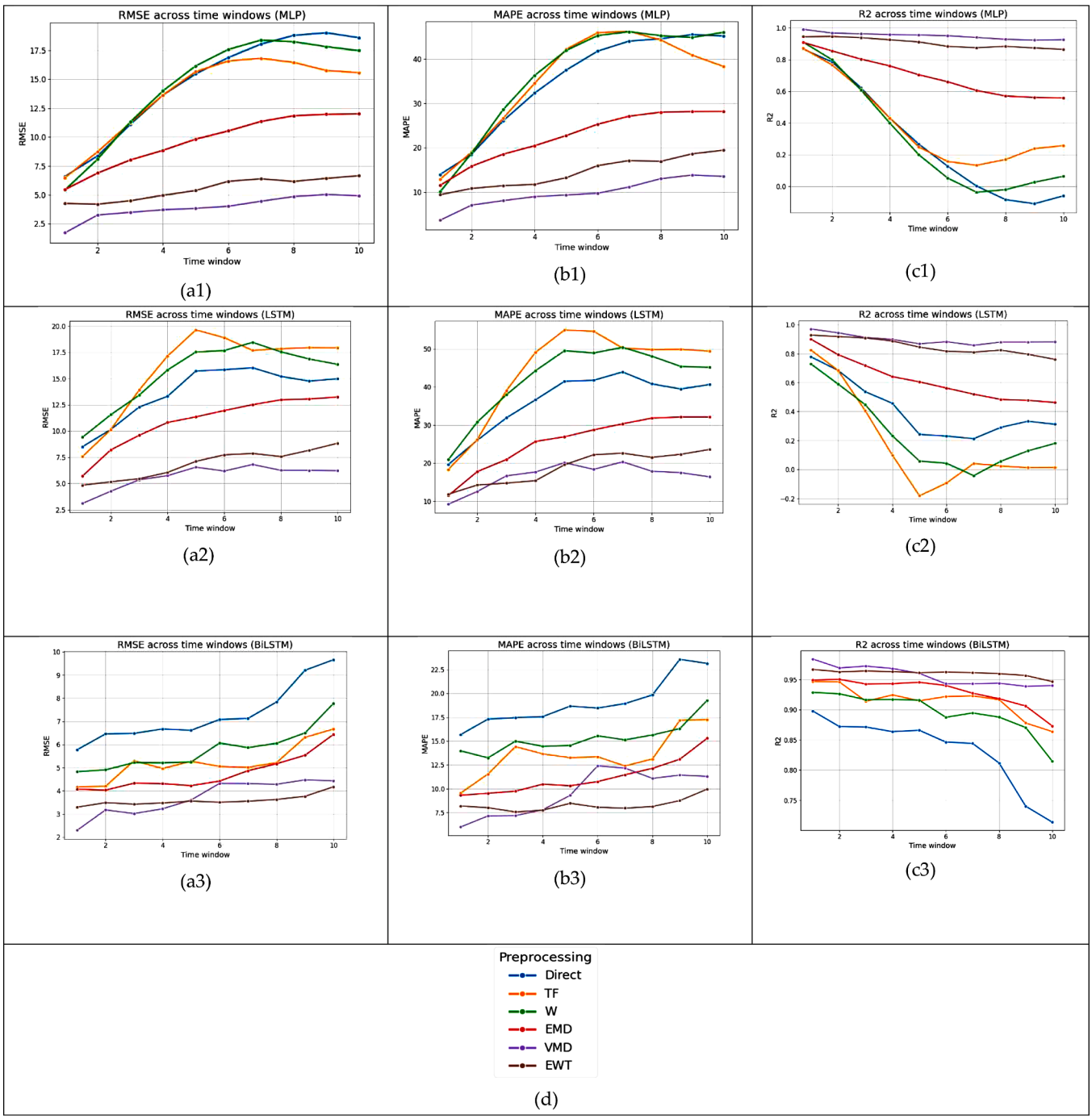

Fig. 10. Evolution of the error metrics with the number of forecasting steps for the EII time series. Subplots (a1–c1) correspond to the MLP model, (a2–c2) to the LSTM model, and (a3–c3) to the BiLSTM model. For each model, the plots display the variation of RMSE, MAPE, and R^2 metrics across different forecasting horizons and preprocessing techniques (Direct, TF, W, EMD, VMD, and EWT). The x-axis represents the time window (forecasting step), while the y-axis indicates the corresponding error metric value.

To test this hypothesis, multiple tests are performed combining different preprocessing models with neural network techniques such as MLP, LSTM, and BiLSTM. The evolution of performance metrics is measured over different time horizons. The results achieved by the different structures tested will be compared, obtaining an approximation of the preprocessing and model that allows for a longer time horizon without compromising the metrics.

To validate this hypothesis, we performed a comprehensive set of experiments combining several preprocessing strategies (trend–-fluctuation decomposition, EMD, VMD, and EWT) with three neural network architectures: MLP, LSTM, and BiLSTM. The experiments evaluate how each preprocessing method affects predictive performance as the forecast horizon increases. By comparing the evolution of error metrics (RMSE, MAPE, and R^2) across different horizons, we determine which model–preprocessing combination yields the best trade-off between forecast length and accuracy.

To achieve this, tests are performed on 1 to 10 output data points, identifying the outcome metrics for each, allowing for a comparison of the evolution based on the preprocessing technique used. Specifically, each model generates forecasts for 1 to 10 future time steps, enabling a

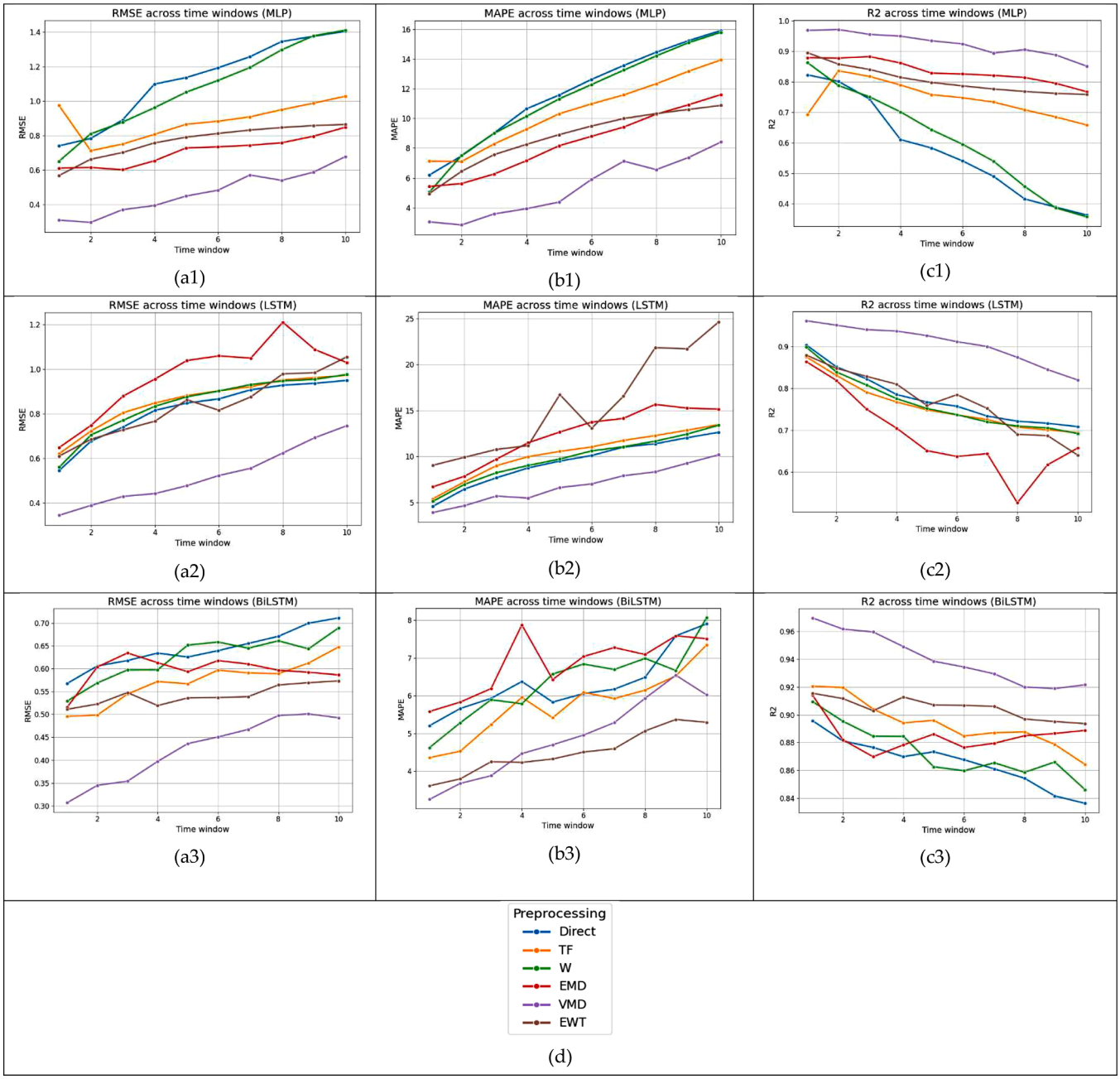

Fig. 11. Evolution of the error metrics with the number of forecasting steps for the Gas time series. Subplots (a1–c1) correspond to the MLP model, (a2–c2) to the LSTM model, and (a3–c3) to the BiLSTM model. For each model, the plots depict the variation of RMSE, MAPE, and R^2 metrics across different forecasting horizons and preprocessing techniques (Direct, TF, W, EMD, VMD, and EWT). The x-axis represents the time window (forecasting step), while the y-axis indicates the corresponding error metric value.

fine-grained comparison of how predictive quality degrades or remains stable over longer horizons depending on the preprocessing applied.

For all models, an architecture based on a single hidden layer with 10 neurons was chosen, allowing the subsequent 10-time steps to be obtained from a single observation. Other numbers of neurons were tested, but this number generally yielded the best performance in most cases.

To ensure comparability and computational efficiency, all neural networks were implemented using a standardized architecture with one hidden layer of 10 neurons, sufficient to generate 10-step forecasts from a single observation window. Preliminary tests with alternative configurations confirmed that this setup provided the most consistent and interpretable results across datasets.

4.1. Dataset

Three representative economic and energy-related time series were selected to evaluate the proposed approach.[1] Each dataset was divided into 75 % for training and 25 % for validation to ensure balanced learning and fair performance comparison across models.

The first series that is represented is the electricity consumption of the School of Industrial Engineering (EII) at the University of Extremadura. Data was collected hourly between January 1 and December 31,

[1] To access the dataset follow this link: https://universidadfv-my.sharepoint.com/:f:/g/personal/ana_lazcano_ufv_es/Eqm7Ag0PZcVHkBA4snAoX-ZAB5Z9vNgrtCjM0Jk0VD6z3Bg?e=fdAjMw

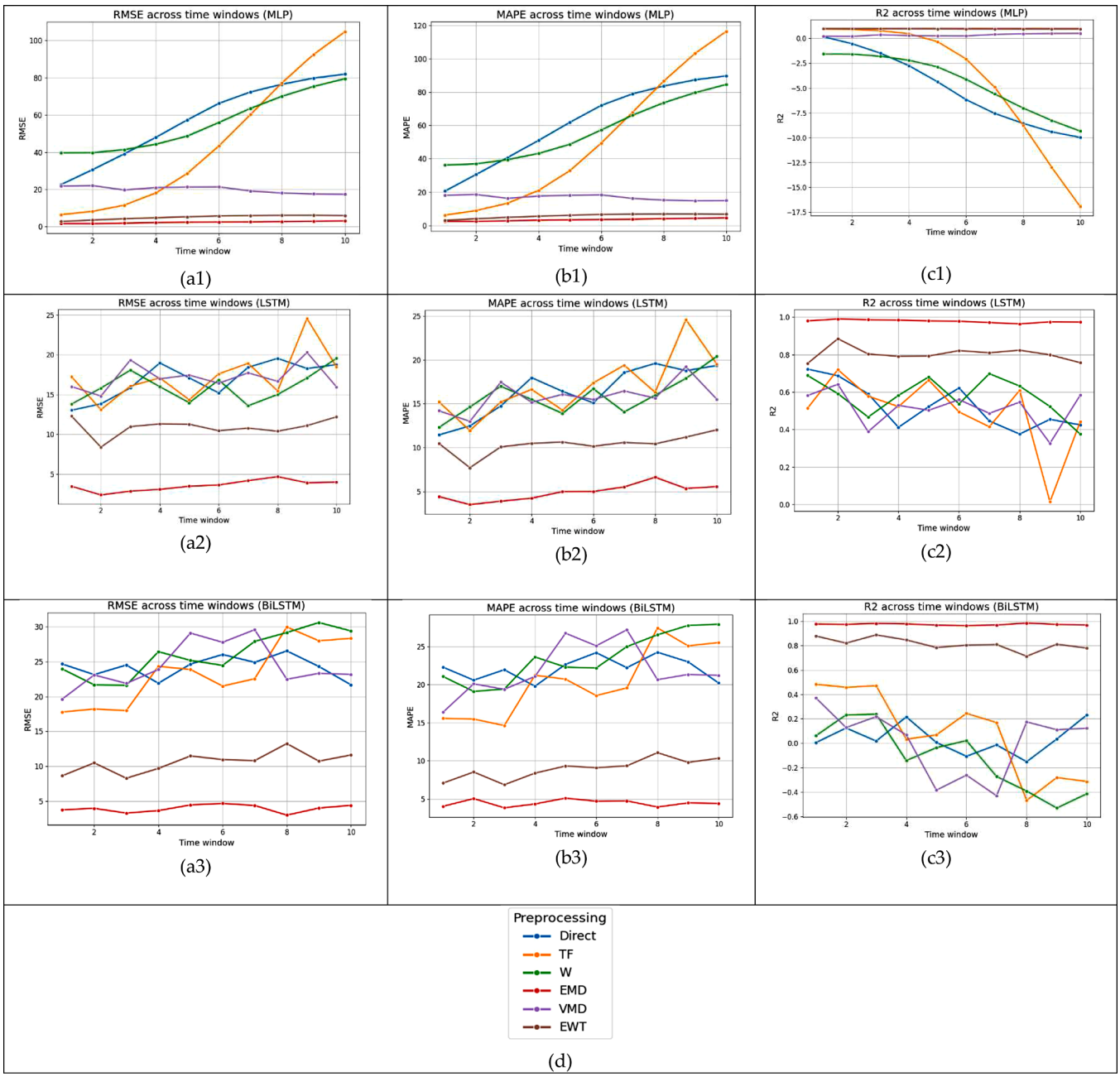

Fig. 12. Evolution of the error metrics with the number of forecasting steps for the CO_2 emission allowances time series. Subplots (a1–c1) correspond to the MLP model, (a2–c2) to the LSTM model, and (a3–c3) to the BiLSTM model. Each group of plots shows the behavior of RMSE, MAPE, and R^2 metrics across different forecasting horizons and preprocessing strategies (Direct, TF, W, EMD, VMD, and EWT). The x-axis represents the time window (forecasting step), while the y-axis displays the corresponding error metric value.

2022, resulting in a total of 8607 observations, see Fig. 2.

The second series uses data corresponding to daily Natural Gas prices (NG), with data collected between January 1997 and February 2024, for a total of 6821 observations. This series is characterized by high volatility, highlighting a greater need for a robust methodology that allows for greater stability in multi-horizon forecasts, see Fig. 3.

The third dataset represents daily CO_2 emission allowance prices in the European Union, obtained from the European Energy Exchange (EEX), Leipzig, Germany. The data ranges span from April 2005 to December 2022, with a total of 4553 observations, see Fig. 4. This series exhibits a strong trend, reinforcing the need to be able to make multi-step predictions without compromising the metrics to achieve a better adjustment to market prices.

These three datasets collectively provide a diverse testing ground combining cyclical (electricity), highly volatile (natural gas), and strongly trending (CO_2 allowances) behaviors allowing us to assess the adaptability and robustness of each modeling strategy under distinct dynamic conditions.

4.2. Error metrics

The performance of the models is evaluated using three standard error metrics (Botchkarev, 2018): Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and R^2.

Root Mean Squared Error (RMSE):

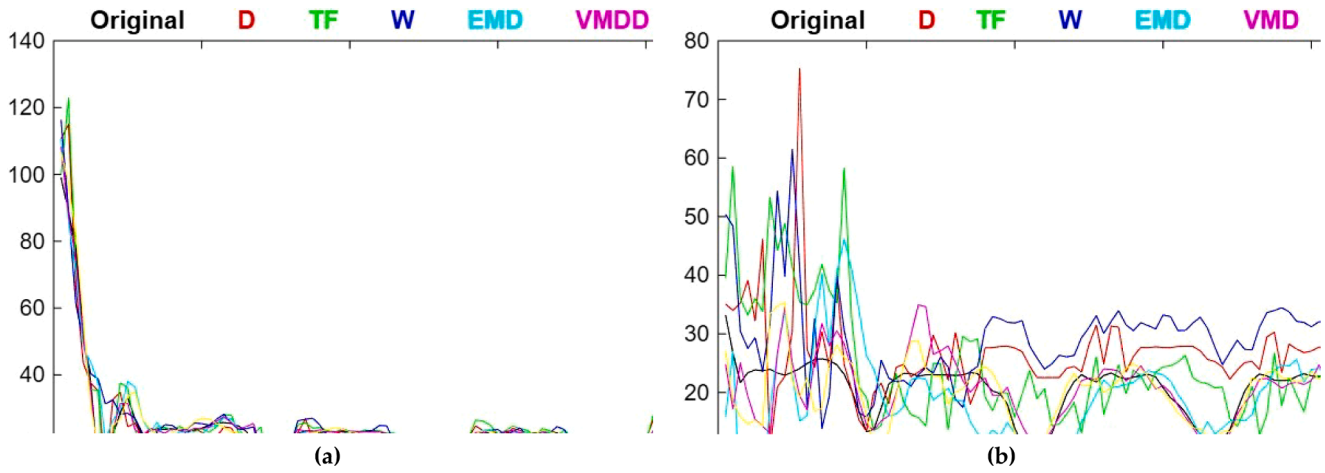

Fig. 13. The last 100 predictions of the EII time series obtained using the MLP model. The figure displays the results for the direct prediction, the desired (true) values, and the outputs obtained with the different preprocessing strategies (TF, W, EMD, VMD, and EWT). Subplot (a) shows the one-step-ahead ($t + 1$) predictions, while subplot (b) presents the ten-step-ahead ($t + 10$) forecasts. The x-axis represents the time index (observations), and the y-axis indicates the signal amplitude of the EII series.

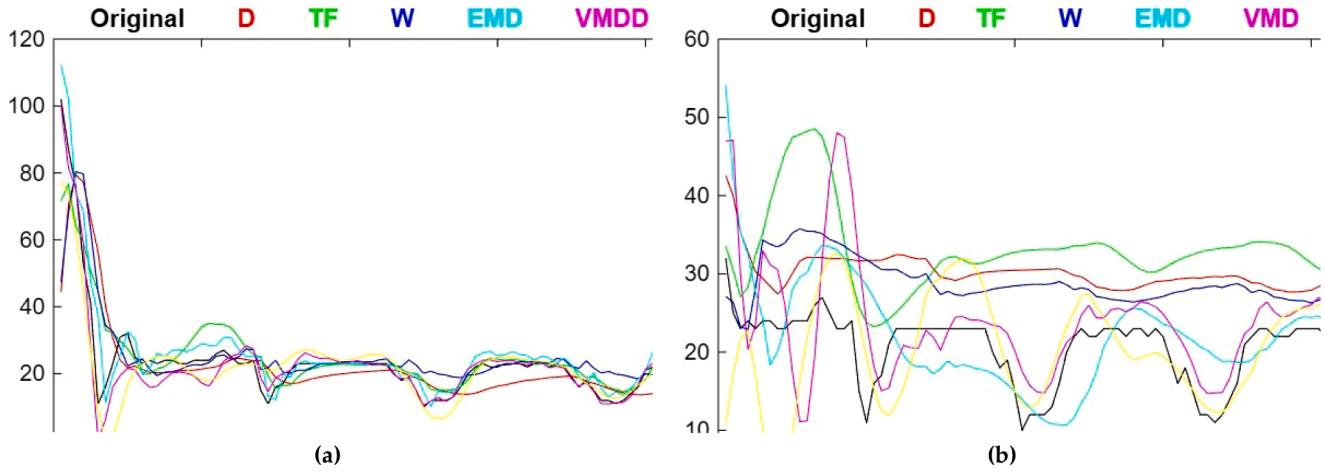

Fig. 14. The last 100 predictions of the EII time series obtained using the LSTM model. The figure illustrates the direct prediction, the desired (true) values, and the outputs derived from the different preprocessing strategies (TF, W, EMD, VMD, and EWT). Subplot (a) shows the one-step-ahead ($t + 1$) predictions, whereas subplot (b) presents the ten-step-ahead ($t + 10$) forecasts. The x-axis represents the time index (observations), and the y-axis denotes the signal amplitude of the EII series.

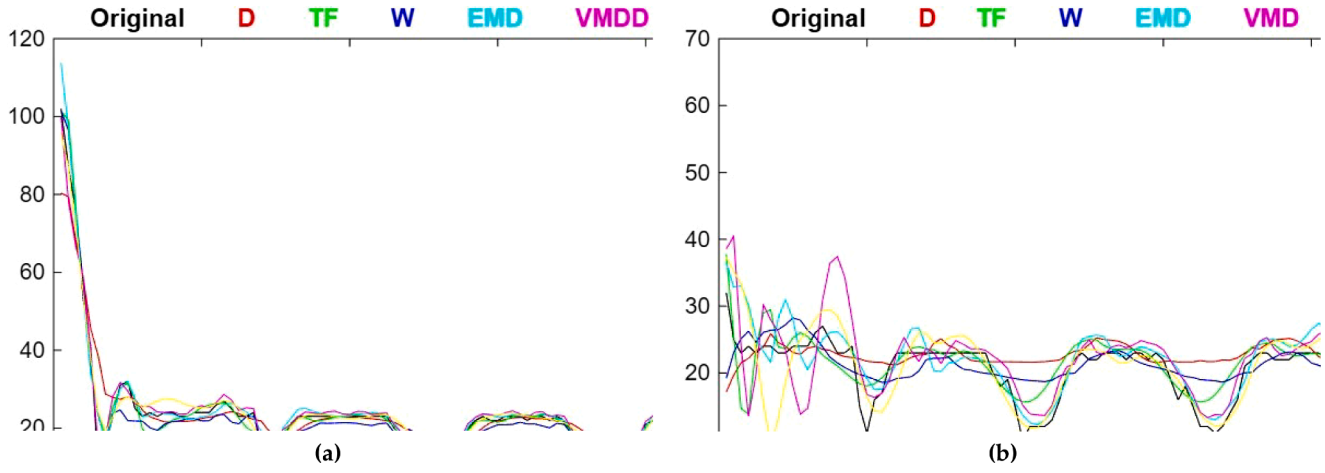

Fig. 15. The last 100 predictions of the EII time series obtained using the BiLSTM model. The figure compares the direct prediction, the desired (true) values, and the results obtained through the various preprocessing methods (TF, W, EMD, VMD, and EWT). Subplot (a) displays the one-step-ahead ($t + 1$) predictions, while subplot (b) shows the ten-step-ahead ($t + 10$) forecasts. The x-axis corresponds to the time index (observations), and the y-axis represents the amplitude of the EII signal.

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(x(t) - \hat{x}(t))^2}. \quad (3)$$

Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\frac{|x(t) - \hat{x}(t)|}{|x(t)|} \times 100. \quad (4)$$

R-squared (R^2):

$$R^2 = 1 - \frac{\sum_{t=1}^{N}(x(t) - \hat{x}(t))^2}{\sum_{t=1}^{N}(x(t) - \bar{x}(t))^2}. \quad (5)$$

In these expressions, the original time series is denoted as $x(t)$, and the predicted series is identified as $\hat{x}(t)$. The mean of the original time series is $\bar{x}(t)$, and the total number of observations is N.

The predictive capacity of the models is evaluated using MAPE and RMSE. Lower values indicate greater accuracy. The model performance is evaluated using R^2, which explains how closely the predictions align with the original time series. Its value ranges from 1 to 0, though values lower than 0 are also possible albeit unusual. Negative values of R^2 mean that there is virtually no relationship between the predictions and the actual data.

Thus, the closer its value is to one, the better the fit. Using R^2 alongside the other metrics employed in this work provides a more rigorous analysis of the results provided by the forecasting models, allowing for easier comparison (Olawoyin & Chen, 2018). Using all these metrics together provides a more robust and rigorous evaluation of predictions (Botchkarev, 2019; Li, 2017).

5. Experimental results

The proposed experimentation was carried out using three representative time series from the economic field that were presented as datasets. In the initial phase, the preprocessing techniques described above were applied to each dataset: trend-fluctuation decomposition, empirical mode decomposition, variational mode decomposition, and empirical wavelet transform. Trend-fluctuation decomposition was performed in two ways: extracting the trend by means of a moving average (TF) and using a noise rejection filter with wavelets (W in the tables and figures).

These preprocessing techniques were selected to evaluate how different forms of signal transformation ranging from simple smoothing to adaptive decomposition affect forecasting stability and accuracy across models. After preprocessing, each subseries obtained using these techniques was independently forecasted using three selected neural network models: MLP, LSTM, and BiLSTM. Predictions for the original series were obtained by summing the predictions for each subseries.

These models are widely recognized for their effectiveness in predicting economic time series (Yang et al., 2022; Pirani et al., 2022; Yang & Wang, 2022). For comparison purposes, the three-time series were also predicted directly without preprocessing (Direct). This allows us to analyze in detail the actual improvement in accuracy that preprocessing can induce in multi-horizon forecasting.

The comparative setup thus distinguishes between models trained directly on raw data and those trained on decomposed subcomponents, isolating the effect of preprocessing on learning stability and error propagation.

The experimental design contemplates generating multi-horizon predictions, that is, several future time steps will be provided in each forecast. Specifically, outputs ranging from 1 to 10-time steps were provided for each model to analyze the robustness of the predictions as the time horizon increases.

5.1. Simulation environment

5.1.1. Computational setup and software environment

All simulations were conducted using MATLAB R2024a on a personal computer equipped with an Intel i9 processor and an NVIDIA RTX 4070 GPU. Both the preprocessing methods and the neural network models were implemented using MATLAB's built-in toolboxes, which allow for flexible adjustment of key hyperparameters. The values assigned to the main hyperparameters are detailed in the following subsections.

5.1.2. Data normalization and preprocessing methods

Before the decomposition process, all time series were normalized by subtracting the mean and dividing by the standard deviation of each dataset. The inverse normalization was applied to rescale predictions to their original units. Although min–max normalization was also tested, the mean–standard deviation normalization provided superior forecasting results. This step prevents extreme values from negatively affecting the learning capability of the neural networks, as is common practice in time series forecasting.

Figs. 5–9 illustrate, for the EII dataset, how each preprocessing technique decomposes the original series into multiple subseries (trend/fluctuation, wavelet-based trend, IMFs, VMD modes, or EWT modes). Each subseries was individually forecasted and subsequently aggregated and denormalized to reconstruct the final multi-horizon prediction. Table 2 summarizes the main preprocessing hyperparameters.

(a) Trend–Fluctuation and Wavelet Decompositions

For the moving-average trend decomposition, a window size of 5 yielded the most accurate forecasts across datasets (Fig. 5). For the wavelet-based noise-reduction method, a Daubechies-5 (db5) wavelet with a single-level decomposition was used, retaining only the low-frequency component to obtain the trend (Fig. 6). Other wavelets were tested, but no significant improvement was observed.

(b) Empirical Mode, Variational Mode, and Empirical Wavelet Decompositions

Both EMD and VMD automatically determine the number of intrinsic mode functions (IMFs) and modes. EMD produced 8 and 9 IMFs plus a residue, while VMD yielded 5 modes plus a residue for all series. For the EWT, the maximum number of peaks was set to 5, since higher values did not lead to noticeable accuracy gains. The corresponding decompositions for the EII series are shown in Figs. 7–9.

5.1.3. Neural network architectures and training configuration

Three neural network architectures were implemented in this study: a Multilayer Perceptron, a Long Short-Term Memory network, and a Bidirectional LSTM. The MLP model was designed with a single hidden layer and one output layer, following the theoretical foundation established by Hornik et al (1989), who demonstrated that a single hidden layer is sufficient for approximating any nonlinear mapping when an adequate number of neurons is used. In our implementation, both the hidden and output layers comprised 10 neurons, corresponding to the 10-step forecasting horizon. Several configurations were tested for the hidden layer but increasing or reducing the number of neurons consistently led to decreased accuracy or less stable convergence, confirming that this setting represented the optimal trade-off between complexity and generalization.

The LSTM and BiLSTM architectures were designed to allow direct comparison with the MLP while leveraging their capacity to model temporal dependencies. Each model consisted of one LSTM layer followed by a fully connected output layer, both with 10 neurons. Using this structure ensured comparable complexity across models. Because LSTM-based networks incorporate internal memory mechanisms to retain information about past values, only one input feature was required. Additional input values were also tested but did not improve predictive performance and occasionally led to degradation due to interference with the models' internal state representation.

Regarding training procedures, the MLP was optimized using the

Levenberg–Marquardt algorithm, which is particularly efficient for shallow networks, whereas the LSTM and BiLSTM models were trained with the Adam optimizer using a learning rate of 0.005. Training was conducted for 100 epochs, a number empirically determined to guarantee convergence since no significant reduction in training error was observed beyond that point. For the LSTM-based models, data were processed in minibatches of 128 samples, consistent with MATLAB's default configuration, which proved effective for maintaining stable convergence without sacrificing accuracy. The main hyperparameters corresponding to each neural network configuration are summarized in Table 3.

5.1.4. Configuration computational performance and efficiency

To ensure transparency and reproducibility, all hyperparameters were systematically tested on the training set. The selected configurations (Tables 2 and 3) represent the most stable setups, balancing performance, and computational cost. The computational efficiency of each stage was also analyzed. The decomposition times, summarized in Table 4, show that all preprocessing steps completed in less than one second, confirming their negligible impact on total simulation time.

Tables 5,6,7 present the training and validation times for the three models using the EII dataset (the largest one). Validation was consistently completed in under one second, demonstrating the framework's lightweight nature and operational feasibility.

Although the MLP exhibits the simplest structure, its training time was longer due to the global optimization process of the Levenberg–Marquardt algorithm. In contrast, LSTM and BiLSTM models benefited from minibatch training, achieving faster convergence. Nonetheless, since MLP training is performed only once, its longer training time does not represent a practical limitation.

Overall, the proposed framework proves computationally tractable and can be executed efficiently on standard desktop hardware without requiring specialized computational resources. The decomposition-based preprocessing further enhances interpretability, as each subseries can be analyzed independently to understand its contribution to the final prediction.

5.2. Results assessment

Tables 8–10 present a summary of the results obtained, focusing on the RMSE metric for all models used, with and without the application of the different preprocessing techniques. The predictions were evaluated at 1-, 5-, and 10-time steps to analyze the multi-horizon behavior of the models.

Complementary metrics RMSE, MAPE and R^2 are presented graphically (Figs. 10–12) to offer a clearer visual understanding of the trade-off between predictive accuracy and model generalization. This structure provides both quantitative rigor and interpretative clarity.

We report only RMSE values in the tables because this metric provides an absolute measure of forecast error that facilitates direct comparison across models and preprocessing methods. In contrast, the complementary metrics R^2 and MAPE are included in the graphical analysis (Figs. 10, 11 and 12), as they provide additional insights into explanatory power and relative error magnitude, respectively, but are less convenient to synthesize in tabular form. This combined presentation ensures both clarity and completeness in the evaluation of forecasting performance.

These experimental results show that, in most of the analyzed cases, the incorporation of preprocessing significantly contributes to the stability of the error metrics (MAPE, RMSE, and R^2), especially in scenarios with longer prediction horizons. Only preprocessing with TF and W present a few cases in which they cannot outperform direct prediction. This behavior suggests that the applied preprocessing transformations (especially with EMD, VMD and EWT) allow the models to better capture the underlying structure of the series, maintaining greater prediction accuracy.

In contrast, the direct approach, which does not include a prior transformation phase, performs notably poorly compared to models with preprocessing. This suggests that the model has difficulty capturing the internal time evolution of the forecasted time series. The model can only provide a rough estimation of the time series' overall behavior, not a more detailed prediction. However, the models with preprocessing, except for TF and W, provide more accurate predictions that increase slightly as the time horizon increases, as can expected when forecasting farther time horizons. These findings reinforce the hypothesis that preprocessing is a key element in improving the quality and stability of multi-horizon forecasts in economic environments characterized by high complexity and nonlinearity.

The results presented in Tables 8, 9, and 10 show that the VMD and EMD decomposition techniques stand out for offering considerably lower error values compared to the other preprocessing methods and the option without preprocessing. One of them always provides the best performance. The EWT decomposition, although almost never can provide the best performance, in all cases provides accurate predictions that outperform those of the models that are not the best. In almost all cases the EWT decomposition is the second in performance.

It is worth noting that the EMD technique achieves the best results among all the evaluated configurations for the CO2 emission allowance price series, with significantly greater accuracy than the other preprocessing models. This behavior could be explained by the specific characteristics of the series, which has a high data density and values close to zero, conditions under which EMD is especially effective in decomposing the signal into components that better preserve its original dynamics.

Figs. 10–12 illustrate the evolution of forecasting performance across the three-evaluation metrics (RMSE (a), MAPE (b), and R^2 (c)) and prediction horizons (1–10 steps). Each subplot corresponds to one of the neural network architectures MLP, LSTM, or BiLSTM while panel (d) provides the color coding used to distinguish preprocessing techniques. This visual framework facilitates the comparison of performance patterns among methods and models as forecasting horizons extend.

In Fig. 10, representing the EII electricity consumption data, all models display the expected gradual decline in accuracy with increasing forecast horizon. However, the direct approach suffers from the steepest deterioration, whereas preprocessing-based models maintain a notably smoother trajectory of error growth, indicating enhanced stability over time.

Among the preprocessing methods, VMD achieves the best balance between accuracy and stability, followed closely by EMD and EWT. These three adaptive decompositions consistently outperform both trend–fluctuation methods (TF and W) and the direct baseline. TF and W, while capable of minor smoothing effects, fail to produce sustained accuracy gains over longer horizons.

These differences in performance can be attributed to the nature of the preprocessing applied. EMD, EWT, and VMD share the characteristic of decomposing time series into intrinsically simpler or stationary components (IMFs in the case of EMD, modes in EWT and VMD), which facilitates learning by the neural model by reducing the structural complexity of the original signal. These techniques better capture the multiscale dynamics of nonlinear and nonstationary series, common in real-world data such as electricity consumption.

In summary, the EII dataset highlights the superiority of adaptive and data-driven decompositions over rigid filtering methods, particularly in signals combining periodicity and irregular fluctuations. This confirms that the model's predictive stability is closely tied to the representational richness of the preprocessing stage.

As shown in Fig. 11, the NG series presents a more volatile and nonlinear pattern, which naturally amplifies forecast uncertainty at longer horizons. Despite this, the VMD-based models consistently achieve the best overall accuracy (lowest RMSE and MAPE, highest R^2) across all horizons. This result indicates that VMD's frequency-centered decomposition is particularly effective at isolating oscillatory

components typical of commodity price dynamics.

Interestingly, the EMD and EWT methods also perform competitively, especially in the short- and mid-term horizons (1 to 5 steps), where they maintain comparable accuracy to VMD. However, their performance deteriorates slightly for 10-step forecasts, possibly due to mode mixing or loss of resolution in high-frequency components. While the temporary performance drop of the EMD method at intermediate horizons may be caused by the increased variability introduced by the decomposition, which can make multi-step forecasting more challenging for LSTM and BiLSTM models, it may subsequently allow for error stabilization.

This behaviour illustrates a general limitation of adaptive decompositions: by separating intrinsic oscillations too finely, they may increase short-term instability, especially in models that rely on temporal recurrence. Nonetheless, these transient effects tend to smooth out at longer horizons, where the aggregated signal components improve robustness and long-term accuracy. Trend-based methods (TF and W), again, contribute minimal benefit and occasionally even increase prediction variance.

Overall, the NG series results confirm the robustness of adaptive preprocessing methods, particularly VMD, in stabilizing forecasts for volatile markets. They also highlight that the effectiveness of decomposition depends on both the signal's spectral characteristics and the forecast horizon under consideration.

Finally, Fig. 12 presents the results for the CO_2 price series. In this case, the comparative analysis of preprocessing techniques reveals that both EMD and EWT consistently provide the best performance across the three neural network architectures, yielding the lowest error levels and highest stability in multi-horizon forecasting. In contrast, the baseline methods without preprocessing (direct and the two trend–fluctuation variants) clearly underperform, as they fail to capture the underlying oscillatory dynamics of the series.

The relative underperformance of VMD in this case suggests that it has difficulty dealing with the steep-rising trend at the end of the time series, which is mainly used for validation, as it does not consider the higher order modes (IMFs higher than 5) which account for the lower frequency of the original signal. Nonetheless, VMD remains considerably more effective than the non-adaptive approaches, maintaining a consistent level of precision even under high nonstationary.

Therefore, when computational efficiency or interpretability is prioritized, VMD represents a viable intermediate solution that balances adaptiveness with numerical stability, whereas EMD and EWT maximize performance in accuracy-critical contexts.

Collectively, these results confirm that preprocessing is not merely a complementary step but a fundamental component of the forecasting pipeline. Adaptive, data-driven decompositions (particularly EMD and EWT) consistently improve the representation of temporal structures, enabling neural models to preserve accuracy and reliability even at extended prediction horizons.

From an applied perspective, the choice of preprocessing method is not only a question of raw predictive accuracy but also of computational cost, robustness across datasets, and operational constraints. The timing analysis in Tables 4–7 shows that all adaptive decomposition methods introduce a non-negligible computational overhead during training when compared with the Direct approach or with simple trend–-fluctuation filtering (TF/W).

For example, training times for the LSTM increase from approximately 1–2 s in the Direct case to 16–20 s when using VMD, EWT, or EMD. A similar pattern is observed for the MLP and BiLSTM. However, this additional cost is compensated by a marked reduction in forecast error at longer horizons, where Direct prediction tends to deteriorate sharply. In practical terms, this indicates that preprocessing is especially valuable in scenarios where forecast stability over multiple future steps is more critical than retraining speed.

Within the adaptive methods, our results suggest different usage profiles. EMD and EWT generally achieve the best absolute accuracy, particularly in the CO_2 allowance price series, where both methods capture slow-varying structure and localized oscillations more effectively than other techniques. For applications in which accuracy at longer horizons is the primary objective (e.g., regulatory planning, long-term procurement strategies), these methods are the preferred choice. VMD, on the other hand, offers a favorable trade-off between performance and computational demands.

Although VMD does not always reach the minimum RMSE achieved by EMD/EWT, it consistently ranks among the top methods in all datasets and exhibits particularly strong behavior in highly volatile conditions such as the natural gas price series. At the same time, VMD requires shorter training times than EMD in all three neural architectures, and its forecasts remain stable as the horizon increases. This makes VMD attractive in operational environments that require regular model updating under time or hardware constraints for example, short-term market monitoring or rolling intraday forecasts.

TF and W represent low-cost baselines: they are extremely fast to compute (their decomposition runs in milliseconds and adds only a modest increase in training time) but generally do not deliver the same multi-horizon stability as the adaptive methods. These approaches may still be useful in embedded or resource-limited scenarios in which models must be retrained very frequently and the acceptable forecast horizon is short $(t+1)$, but our results show that they are less reliable for longer-range projections.

Taken together, the comparative results provide a practical guideline for real-world implementation: EMD and EWT are recommended when achieving maximum long-horizon accuracy is the main priority and higher training costs are acceptable; VMD offers the best compromise between accuracy, robustness across diverse time series, and moderate computational demands; and TF/W or Direct approaches are best suited for contexts where computational efficiency or frequent retraining takes precedence over multi-step forecast stability. This interpretation connects the empirical evidence with specific operational scenarios and demonstrates how the proposed framework can effectively guide model selection in applied forecasting tasks.

Figs. 13–15 connect the preprocessing stage with forecasting performance. For the dataset EII, they compare the predicted trajectories obtained with each preprocessing method (TF, W, EMD, VMD, EWT) and each neural architecture (MLP, LSTM, BiLSTM) against the ground truth, for horizons $t+1$ and $t+10$. This side-by-side view illustrates how the choice of decomposition affects short- and long-horizon accuracy. Only the results for one time series (EII) are shown to avoid excessive figures. For clarity, only the last 100 predictions are shown.

As these figures show, all models can provide accurate predictions for the first-time horizon, as shown in figures (a). However, accuracy decreases for tenth future values, as shown in figures (b). While the Direct, TF, and W predictions lose accuracy over time, the EMD, VMD, and EWT predictions follow the time series' evolution, providing reliable, albeit slightly less accurate, predictions. This proves their robustness and reliability.

Overall, these findings confirm that the combination of adaptive preprocessing and neural networks constitutes a methodological advance that significantly improves the robustness and reliability of multi-horizon forecasting in economic environments characterized by volatility and nonlinear dynamics. Beyond their empirical relevance, the results contribute to clarifying the comparative advantages and trade-offs among different preprocessing techniques, offering practical guidelines for selecting the most appropriate strategy depending on the characteristics of the data under analysis.

6. Conclusions

Multi-horizon predictions continue to be one of the most relevant challenges in the field of time series forecasting using neural networks. Accurately anticipating multiple future steps based on previously predicted data is a key strategic capability, especially in contexts where

anticipating critical events or planning ahead is essential. However, this approach still faces a significant limitation: the progressive degradation of accuracy as the time horizon expands. This loss is particularly significant when using autoregressive models, as the error tends to accumulate with each iteration. Furthermore, many of the techniques proposed to mitigate this problem involve significant computational complexity or significantly more sophisticated architecture.

This work explores an alternative based on traditional signal preprocessing techniques (TF, Wavelet, EMD, VMD, and EWT), with the aim of improving the stability and accuracy of multi-horizon predictions without substantially increasing the computational burden or model complexity. These techniques allow the original signal to be decomposed into simpler components, facilitating the extraction of underlying patterns that might be difficult for the model to capture directly.

The results obtained confirm that appropriate preprocessing significantly improves the performance of neural models, allowing the prediction horizon to be extended up to ten steps with a much smaller loss of accuracy than that obtained with the direct methodology. However, the benefits vary depending on the type of technique used.

Classical transformations such as trend–fluctuation decomposition, whether using moving average or wavelet-based noise rejection, proved insufficient to yield consistent improvements. This limitation can be attributed to their rigid filtering nature, which lacks adaptability to the intrinsic nonstationary and multiscale dynamics of real-world time series.

In contrast, adaptive decomposition methods EMD, VMD, and EWT demonstrated a superior ability to disentangle latent oscillatory patterns at distinct frequency bands, thereby facilitating more efficient learning by neural networks. Among these, VMD exhibited the most stable performance across prediction horizons, likely due to its controlled mode separation mechanism, which preserves frequency coherence and reduces overfitting risk.

Conversely, the temporary performance degradation observed in EMD for intermediate horizons can be linked to the higher variability and mode mixing introduced by its fully data-driven nature, which may complicate temporal dependency learning in recurrent architectures such as LSTM and BiLSTM.

Within real-world financial applications, these findings are especially relevant, since anticipating several future steps enables more accurate forecasting and improved risk management. However, the performance of the proposed methods still depends on the appropriate tuning of the decomposition parameters and the specific dynamics of each time series.

Nevertheless, it is important to note that these conclusions are based on experiments conducted in an offline environment using historical datasets. While the obtained results demonstrate strong potential for real-world use, applying the proposed framework in real-time forecasting contexts would require additional validation to address challenges such as continuous data inflow, model retraining frequency, and latency constraints. Future research will therefore focus on adapting and testing this framework under online and real-time conditions to assess its practical deployment capabilities.

As a future direction, it is also proposed to explore extending the prediction horizon beyond ten steps and to combine these preprocessing strategies with more advanced or hybrid neural architectures to optimize the balance between accuracy, stability, and computational cost.

CRediT authorship contribution statement

Ana Lazcano: Conceptualization, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration. **Julio E. Sandubete:** Conceptualization, Investigation, Resources, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration. **Miguel A. Jaramillo-Morán:** Conceptualization, Methodology, Software, Validation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Julio E. Sandubete reports a relationship with Francisco De Vitoria University Foundation that includes: employment. Ana Lazcano reports a relationship with Francisco De Vitoria University Foundation that includes: employment. Miguel A. Jaramillo-Moran reports a relationship with University of Extremadura that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Mathematical Details

This appendix compiles the mathematical formulations included in Section 3, providing complete equations and explanatory context for each model and preprocessing method.

A.1 Multilayer Perceptron (MLP)

One of the most widely used models in the literature for time series prediction is the classical Multilayer Perceptron (Lazcano et al., 2024; Han et al., 2019; Göçken et al., 2016; Keles et al., 2016; Fan et al., 2015), which is probably the simplest commonly used neural model nowadays. Its lasting relevance stems from its capability to behave as a universal approximator (Hornik et al., 1989). Furthermore, the MLP's relatively simple architecture facilitates programming (Zhang et al., 2022; Borghi et al., 2021), while still providing accurate and reliable results, as demonstrated by recent empirical studies (Madhusudhanan et al., 2024).

An MLP consists of several layers, each containing a variable number of neurons. These neurons act as the processing units. The input data are assumed as the input layer, which will be processed by a first hidden one that will pass the results to the following hidden layer. Although several hidden layers can be implemented in an MLP, often only one layer is sufficient to provide accurate predictions (Hornik et al., 1989). The network's output will be given by an output layer following the last hidden one.

Each neuron processes the information it receives from the previous layer:

$$x_j^l = f\left(\sum_i w_{ji}^l x_i^{l-1} + b_j\right). \tag{A.1}$$

In this expression x_j^l represents the output of neuron j, located in layer l. This neuron processes the outputs of all neurons in the preceding layer, x_i^{l-1}, after multiplying them by a weigh matrix, w_{ji}^l, that represents the strength of those connections. The result of this product, along with a bias term, b_j, is then processed by an activation function, $f()$, that provides the neuron's output. This function is typically a sigmoid, a hyperbolic tangent or a

rectified linear unit (ReLU) in the hidden layers. However, it used to be a linear one in the output layer. Due to the full interconnectivity of neurons between layers, they are usually referred to as dense layers (Wanchen et al., 2020), and neural networks with this structure are known as fully connected networks.

The capability of neural networks to mimic the behavior of a dynamic system arises from their capability to learn that behavior from data describing its evolution. To do so, however, they must first be trained. Therefore, the available dataset must be divided into two subsets: one for training and one for subsequent network validation.

The MLP is trained using an algorithm that seeks to minimize the prediction error. It is the well-known Backpropagation (Zhang et al., 2023), which, despite being originally developed for the MLP, has become the standard procedure for training several neural models. It works as follows. An input pattern (a set of past data in the context of time series forecasting) is presented to the network. The network processes this input pattern and provides an output (a prediction in time series forecasting). This output is then compared with a desired prediction (which has been previously associated with the input pattern) to obtain an error. The network parameters are then updated based on this error to minimize it by backpropagating the error from the output layer to the previous layers. This process repeats for each input pattern in the training dataset until a predefined minimum error is achieved or a certain number of iterations is reached.

A.2 Long short-term memory (LSTM)

The Long Short-Term Memory is a neural model designed to solve complex problems involving time-dependent data, such as natural language processing, speech processing, and machine translation tasks (Hochreiter & Schmidhuber, 1997). To do this, LSTMs incorporate an internal memory and recurrent feedback between neurons within the same layer to be able to capture long-term dependencies in the data. Thus, a complex inner structure must be defined for each neuron, which is why they are usually referred to as cells instead of neurons.

The structure of the typical LSTM cell is shown in Fig. A.1. The variables describing the cell's time evolution are the inputs x_t, the outputs y_t, the memory states c_{t-1} and c_t, and the recurrent feedback connections y_{t-1}. Three control gates are also defined to control how these variables are combined and processed to provide the cell's output: an input gate, a forget gate, and an output gate. These gates provide control signals obtained by processing the new inputs received by the cell, x_t, together with the feedback from neurons in the same layer This feedback consists of the outputs of these neurons in the previous time step, y_{t-1}. The control signals are provided by functions with a form like that of the neuron of an MLP, Eq. (A.1), with the sigmoid as transfer function to give values in the range [0,1].

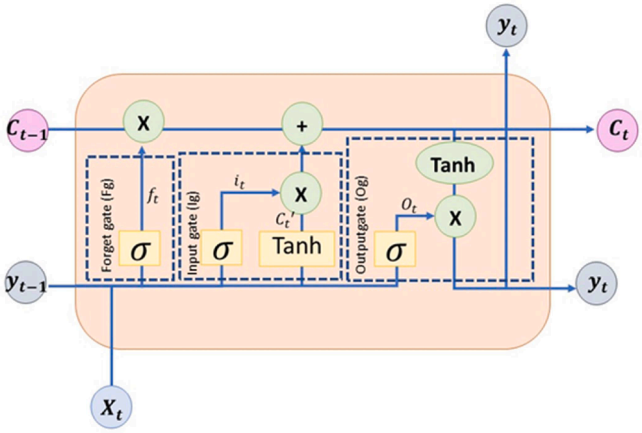

Fig. A.1. LSTM neural network.

The input gate determines the proportion of the new inputs, x_t, and feedback, y_{t-1}, that should be added to the internal state of the cell, c_{t-1}. The forgetting gate controls how much of the stored information, c_{t-1}, should be forgotten, or retained. Finally, the output gate determines how much of the new internal state, c_t, will be released as the cell's output. The equations describing these three gates are:

$$i_t = \sigma\left(W^i[y_{t-1}, \boldsymbol{x}_t] + b^i\right), \tag{A.2}$$

$$f_t = \sigma\left(\boldsymbol{W}^f[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^f\right), \tag{A.3}$$

$$o_t = \sigma(\boldsymbol{W}^o[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^o). \tag{A.4}$$

For clarity, in these equations, the new inputs and the pasts states of the cells in the same layers, feedback, are merged into a single vector, $[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t]$. The parameters $\boldsymbol{W}^i$, $\boldsymbol{W}^f$, and $\boldsymbol{W}^o$are weight matrices, while b^i, b^f, and b^o represent bias terms.

To produce a new output from the signals it receives, the cell works as follows. First, a new temporal internal state must be provided from the cell's inputs:

$$c_t^{'} = \tanh(\boldsymbol{W}^c[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^c), \tag{A.5}$$

where $\boldsymbol{W}^c$ and b^c are again the corresponding weight vector and bias, respectively. A hyperbolic tangent function is used to give values within the interval $[-1, 1]$.

Then, the cell's internal state in time step *t-1*, c_{t-1}, is combined with the temporal internal state, $c_t^{'}$, by means of the corresponding input and forget gates to obtain a new internal state:

$$c_t = f_t c_{t-1} + i_t c_t^{'}. \tag{A.6}$$

Finally, the output of the cell is a fraction of the hyperbolic tangent of the new internal state, c_t, as decided by the output gate:

$$y_t = o_t \tanh(c_t). \tag{A.7}$$

Like other neural models, the LSTM must be trained before it can be used. This is done using a modified version of the backpropagation algorithm. This algorithm is modified to handle the internal memory and feedback between neurons in the same layer. Two different procedures are used to adjust all the parameters in each cell (Gers et al., 2000). The first, Truncated Backpropagation Through Time (BPTT), adjust the parameters of the output gate and the activation function by limiting the depth of the time steps considered, to manage computational complexity. The second, Real-Time Recurrent Learning (RTRL), optimizes the input gate, the forget gate, and the input activation function. RTRL is suited for updating parameters in recurrent connections in real-time conditions.

A.3 Bidirectional LSTM (BiLSTM)

The bidirectional LSTM is an extension of the LSTM model. It defines two parallel processing paths for data: one that processes the input sequence in chronological order and another that processes it in reverse. This structure is intended to identify deep, time-dependent relationships in data, thereby improving the LSTM's forecasting capability.

Siami-Namini et al (2019) have argued for the advantages of adding layers to neural models, claiming that BiLSTM outperforms standard LSTM because of its bidirectional processing. BiLSTMs have been used to predict commercial time series and have reported improved performance (Kim & Moon, 2019).

Fig. A.2 shows how the BiLSTM works. For the sake of clarity, only three-time steps are presented. The network has two layers that process data in opposite directions to achieve simultaneous learning in different contexts. One layer processes information from left to right (forward time), and the other processes information from right to left (backward time). At time step t, the input $\boldsymbol{x}_t$ enters both the forward and backward layers through weight matrices $\boldsymbol{W}^1$ and $\boldsymbol{W}^3$, respectively. Each layer receives its own previous internal state (c^f_{t-1} for forward and c^b_{t+1} for backward) through matrices $\boldsymbol{W}^2$ and $\boldsymbol{W}^5$. The combined result from both layers, $\boldsymbol{y}_t$, is computed using weight matrices $\boldsymbol{W}^4$ and $\boldsymbol{W}^6$.

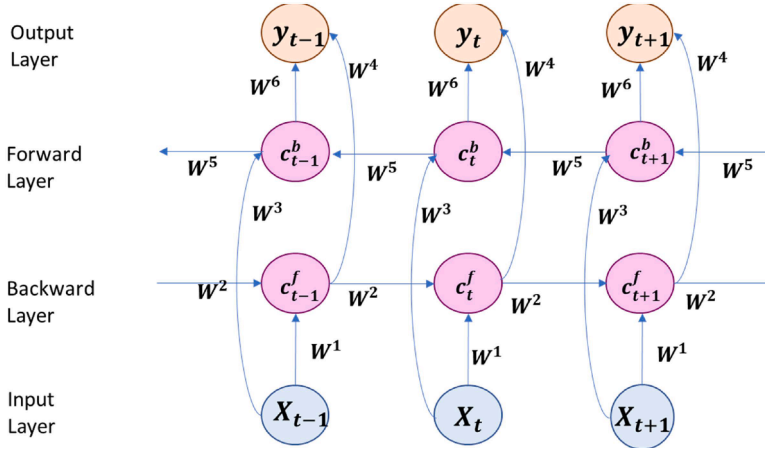

Fig. A.2. BiLSTM neural network.

The equations that describe the network behavior are:

$$\boldsymbol{c}^f_t = f\left(\boldsymbol{W}^1\boldsymbol{x}_t + \boldsymbol{W}^2\boldsymbol{c}^f_{t-1}\right), \tag{A.8}$$

$$\boldsymbol{c}^b_t = f\left(\boldsymbol{W}^3\boldsymbol{x}_t + \boldsymbol{W}^5\boldsymbol{c}^b_{t+1}\right), \tag{A.9}$$

$$\boldsymbol{y}_t = g\left(\boldsymbol{W}^4\boldsymbol{c}^f_t + \boldsymbol{W}^6\boldsymbol{c}^b_t\right). \tag{A.10}$$

In these equations f$(\cdot)$ and $g(\cdot)$ are saturating functions: sigmoid or hyperbolic tangent.

A.4 Data preprocessing

A.4.1 Trend decomposition

(Arianos & Carbone, 2007; Osborn, 1995) have proposed decomposing a time series into trend and fluctuations. This decomposition aims to facilitate independent predictions of both subseries. It was proven in Gonzalez-Romera et al (2006) that this results in greater accuracy because the forecasting models will better learn the behavior of each subseries. One of the most widely used tools for extracting the trend of a time series is the moving average. It replaces each datum with the average of N preceding data points and itself:

$$T(t) = \frac{1}{N+1}\sum_{i=0}^{N} x(t-i), \tag{A.11}$$

In this expression $x(t)$ is the original time series, and $T(t)$ is the resulting trend series. The trend series loses the first N data points due to the way it is obtained.

The fluctuation series, F(t), is obtained by subtracting $T(t)$ from $x(t)$. These two series can be forecasted independently, and their predictions then added to obtain the forecast of the original time series. However, the first N elements are lost because of how the trend series was obtained.

A.4.2 Empirical model decomposition

The Empirical Model Decomposition algorithm decomposes the original time series into subseries with periodic behaviour, facilitating prediction (Ren et al., 2014).

During the process of extracting the IMFs, they must meet two criteria to properly characterize the periodic components of the series: the difference between the number of zero-crossing points and the number of extremum points must be not greater than 1, and the average of the upper and lower envelopes, obtained by connecting the corresponding maximum and minimum points, must equal zero.

The IMFs and the residual are obtained by an iterative procedure that stops when no more IMFs can be obtained (Huang et al., 2008).

1. Upper and lower envelopes are defined from the original time series $x(t)$ by connecting its maxima $u(t)$ and minima $l(t)$.
2. The mean of the upper and lower envelopes is then obtained:

$$m(t) = \frac{u(t) + l(t)}{2}. \tag{A.12}$$

3. An IMF candidate $h(t)$ is obtained by subtracting $m(t)$ from $x(t)$:

$$h(t) = -m(t). \tag{A.13}$$

4. If $h(t)$ meets the two criteria for being an IMF, it is selected as the first IMF, $c_1(t)$, and the algorithm moves on to the next step. Otherwise, $h(t)$ is considered a new time series, and it will be processed following steps 1 to 3 until an IMF is obtained.
5. A new series is defined by subtracting $c_1(t)$ from $x(t)$, which will be subsequently processed following steps 1 to 4:

$$f_n(t) = x(t) - c_1(t). \tag{A.14}$$

This process will be iterated until no new IMFs are obtained. Once the algorithm stops, the residual is obtained by subtracting the last IMF from the original time series: $h(t) = r(t)$.

The algorithm uses two stopping conditions that prevent it from falling into an infinite loop. The first condition is evaluated at step 4 and is applied when the candidate IMF meets the IMF conditions, or its variance falls below a certain threshold. The second condition is evaluated at the final stage and is applied when the residual becomes a constant, exhibits a steady linear trend, or has only one extremum.

The original time series can be decomposed as the sum of the IMFs and the residual:

$$x(t) = \sum_i c_i(t) + r(t). \tag{A.15}$$

The IMFs and the residual can be forecasted independently, and their predictions can be added together to obtain a prediction for the original time series.

A.4.3 Variational Mode Decomposition

The aim of the Variational Mode Decomposition (Dragomiretskiy & Zosso, 2013) is like that of the EMD, since it decomposes a time series $x(t)$ into a set of subseries (modes) $u_i(t)$ with specific bandwidths that must be mostly compact around a central frequency ω_i:

In this expression $A_i(t)$ represents the amplitude and $\phi_i(t)$ the phase of the corresponding mode $u_i(t)$, which will be associated with a central frequency $\omega_i(t)$:

$$x(t) = \sum_{i=1}^{N} u_i(t) = \sum_{i=1}^{N} A_i(t)\cos\left(\phi_i(t)\right) \tag{A.16}$$

The VMD algorithm seeks to define an ensemble of modes with the smallest possible bandwidth whose sum can accurately reproduce the original signal. This is an optimization problem that can be transformed into a variational problem:

$$\omega_i(t) = \frac{\partial \phi_i(t)}{\partial t}. \tag{A.17}$$

$$\min_{\{u_i\},\{w_i\}} \left\{ \sum_i \| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \|_2^2 \right\}, \tag{A.18}$$

where $\delta(t)$ is de Dirac distribution function and * the convolution operator. The squared L2-norm is used. This is a constrained variational problem that can be converted into an unconstrained optimization problem, in which an augmented Lagrangian function including a quadratic penalty and Lagrange multipliers are defined:

$$\mathscr{L}(\{u_i\},\{w_i\},\lambda) = \alpha \sum_i \| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \|_2^2 + \| x(t) - \sum_i u_i(t) \|_2^2 + \left\langle \lambda(t), f(t) - \sum_i u_i(t) \right\rangle. \tag{A.19}$$

The problem can be solved by identifying saddle points in this function. An iterative technique can be used to do this: the Alternating Direction Multiplier Method (Boyd et al., 2011). According to this algorithm, the modes and their corresponding central frequencies can be updated using the iterative expressions:

$$u_i^{n+1} = \arg\min_{u_i} \left\{ \alpha \| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \|_2^2 + \| x(t) - \sum_i u_i(t) + \frac{\lambda(t)}{2} \|_2^2 \right\} \tag{A.20}$$

$$\omega_i^{n+1} = \arg\min_{\omega_i}\left\{\sum_i \| \partial_t\left[\left(\delta(t)+\frac{j}{\pi t}\right) * u_i(t)\right]e^{-j\omega_i t}\|_2^2\right\}. \tag{A.21}$$

A.4.4 Empirical wavelet transform

The Empirical Wavelet Transform (Gilles, 2013) performs a decomposition like those provided by the EMD and the VMD. However, it differs in that it operates on the Fourier spectrum of the time series to identify frequency peaks that will be associated with wavelets to make up a filter bank that can be subsequently used to carry out the decomposition process. The subseries produced by this filter bank define a compact support Fourier spectrum.

To build this filter bank, the Fourier spectrum is divided into N consecutive segments, which are limited to the $[0,\ \pi]$ interval to guarantee that the Shannon criterium is accomplished. Each segment will be associated with a central frequency ω_i and a width of $2\tau_i$, with $\tau_i = \gamma\omega_i$ and $0 < \gamma < 1$.

Each element of the filter ban filter bank is defined by means of a set of empirical scaling functions $\widehat{\varphi}_n(\omega)$ (low-pass filter) and empirical wavelets $\widehat{\psi}_n(\omega)$ (band-pass filters):

$$\widehat{\varphi}_n(\omega) = \begin{cases} 1 & \text{if } |\omega| \le (1-\gamma)\omega_n \\ \cos\left[\frac{\pi}{2}\,\beta\left(\frac{1}{2\gamma\omega_n}(|\omega| - (1-\gamma)\omega_n)\right)\right] & \text{if } (1-\gamma)\omega_n \le |\omega| \le (1+\gamma)\omega_n \\ 0 & \text{otherwise} \end{cases} \tag{A.22}$$

$$\widehat{\psi}_n(\omega) = \begin{cases} 1 & \text{if } (1+\gamma)\omega_n \le |\omega| \le (1-\gamma)\omega_{n+1} \\ \cos\left[\frac{\pi}{2}\,\beta\left(\frac{1}{2\gamma\omega_n}(|\omega| - (1-\gamma)\omega_{n+1})\right)\right] & \text{if } (1-\gamma)\omega_{n+1}+1 \le |\omega| \le (1+\gamma)\omega_{n+1} \\ \sin\left[\frac{\pi}{2}\,\beta\left(\frac{1}{2\gamma\omega_n}(|\omega| - (1-\gamma)\omega_n)\right)\right] & \text{if } (1-\gamma)\omega_n \le |\omega| \le (1+\gamma)\omega_n \\ 0 & \text{otherwise} \end{cases} \tag{A.23}$$

$\beta(x)$ must be defined for each specific case, although it usually has the form:

$$\beta(x) = \begin{cases} 0 & \text{if } x \le 0 \\ x^4\,(35 - 84x + 70x^2 - 20x^3) & \text{if } 0 < x < 1 \\ 1 & \text{if } x \ge 1 \end{cases} \tag{A.24}$$

To properly define the filter bank, each frequency peak identified in the Fourier spectrum will be associated with one of the previously defined N segments. However, since the number of peaks *M* and the number of segments *N* do not have to be the same, two cases may arise:

$M \ge N$: only the first *M-1* peaks will be considered (the first segment will be associated to $\omega = 0$).

$M < N$: only M segments will be defined.

The EWT will decompose the signal (time series) $x(t)$ into a set of empirical modes:

$$x(t) = x_0(t) + \sum_{i=1}^{N-1} x_i(t). \tag{A.25}$$

In this expression each mode $x_n(t)$ will be defined as:

$$x_0(t) = W_f^{\varepsilon}(0,t) * \varphi_1(t), \tag{A.26}$$

$$x_i(t) = W_f^{\varepsilon}(i,t) * \psi_i(t), \tag{A.27}$$

where $\varphi_1(t)$ and $\psi_i(t)$ are wavelets in the time domain that corresponds to the scaling function and empirical wavelets $\widehat{\varphi}_n(\omega)$ and $\widehat{\psi}_n(\omega)$, respectively, in the Fourier domain, while $W_f^{\varepsilon}(0,t)$ is the approximation coefficient associated with the lowest frequency, and $W_f^{\varepsilon}(i,t)$ is the detail coefficients associated with higher frequencies. The operator * defines the convolution of two functions. These coefficients are obtained as the inner products:

$$W_f^{\varepsilon}(0,t) = \langle x(t), \varphi_1(t)\ \rangle = \int x(\tau)\overline{\varphi}_1(\tau - t)d\tau = \ \mathscr{F}^{-1}\{\widehat{x}(\omega)\ \overline{\widehat{\varphi}_1}(\omega)\}, \tag{A.28}$$

$$W_f^{\varepsilon}(i,t) = \langle x(t), \psi_i(t)\ \rangle = \int x(\tau)\overline{\psi}_i(\tau - t)d\tau = \mathscr{F}^{-1}\{\widehat{x}(\omega)\ \overline{\widehat{\psi}}_i(\omega)\}, \tag{A.29}$$

where the bar over a function stands for its complex conjugate and $\mathscr{F}^{-1}\{\}$ represents the inverse Fourier transform.

The EWT provides a segmentation in the frequency space, which allows for the definition of a filter bank, as mentioned above. In this way, if the coefficients $W_f^{\varepsilon}(i,t)$ are adjusted appropriately, different filters can be obtained and then used for signal filtering (Penedo et al., 2019). So, a low-pass filter can be designed to extract the trend of a time series by removing the higher frequencies, which are usually associated with noise due to their lower significance. This filter will behave like a noise rejection filter. Therefore, after performing the EWT process, modes $x_n(t)$ whose wavelet coefficients $W_f^{\varepsilon}(i,t)$ are lower than a certain threshold can be removed. This provides an approximation of the original signal, Eq. (A.25), with only the elements, modes $x_n(t)$, associated with the lower frequencies. This approximation should define the series' trend. Thus, a fluctuation time series can be obtained by subtracting it from the original series as it was done with the trend-fluctuation decomposition proposed above.

Data availability

Data will be made available on request.

References

Agarwal, S., Sharma, S., Faisal, K. N., & Sharma, R. R. (2025). Time-series forecasting using SVMD-LSTM: A hybrid approach for stock market prediction. *Journal of Probability and Statistics, 2025*(1), Article 9464938.

Alaa, A. M., & van der Schaar, M. (2019). Attentive state-space modeling of disease progression. *Advances in neural information processing systems* (p. 32).

Arianos, S., & Carbone, A. (2007). Detrending moving average algorithm: A closed-form approximation of the scaling law. *Physica A: Statistical Mechanics and its Applications, 382*(1), 9–15. https://doi.org/10.1016/j.physa.2007.02.074

Bontempi, G. (2008). Long term time series prediction with multi-input multi-output local learning. In *Proc. 2nd ESTSP* (pp. 145–154).

Bontempi, G., Ben Taieb, S., & Le Borgne, Y. A. (2012). Machine learning strategies for time series forecasting. *European big data management and analytics summer school (pp. 62-77)*. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-36318-4_3

Borghi, P. H., Zakordonets, O., & Teixeira, J. P. (2021). A COVID-19 time series forecasting model based on MLP ANN. *Procedia Computer Science, 181*, 940–947. https://doi.org/10.1016/j.procs.2021.01.250

Botchkarev, A. (2018). Performance metrics (error measures) in machine learning regression, forecasting and prognostics: properties and typology. arXiv preprint arXiv:1809.03006. https://doi.org/10.28945/4184.

Botchkarev, A. (2019). A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Interdisciplinary Journal of Information, Knowledge, and Management, 14*, 045–076.

Boyd, S., Parikh, N., Chu, E., Peleato, B., & Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning, 3*(1), 1–122. https://doi.org/10.1561/2200000016

Cerqueira, V., Torgo, L., & Soares, C. (2019). Machine learning vs statistical methods for time series forecasting: size matters. arXiv preprint arXiv:1909.13316.

Chen, Z., Ma, M., Li, T., Wang, H., & Li, C. (2023). Long sequence time-series forecasting with deep learning: A survey. *Information Fusion, 97*, Article 101819. https://doi.org/10.1016/j.inffus.2023.101819

Cheng, Z., & Wang, J. (2020). A new combined model based on multi-objective salp swarm optimization for wind speed forecasting. *Applied Soft Computing, 92*, Article 106294. https://doi.org/10.1016/j.asoc.2020.106294

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. *International Journal of Forecasting, 22*(3), 443–473. https://doi.org/10.1016/j.ijforecast.2006.01.001

Dragomiretskiy, K., & Zosso, D. (2013). Variational mode decomposition. *IEEE Transactions on Signal Processing, 62*(3), 531–544. https://doi.org/10.1109/TSP.2013.2288675

Elsayed, S., Thyssens, D., Rashed, A., Jomaa, H.S., & Schmidt-Thieme, L. (2021). Do we really need deep learning models for time series forecasting? arXiv preprint arXiv:2101.02118.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*, 987–1007. https://doi.org/10.2307/1912773

Fan, C., Zhang, Y., Pan, Y., Li, X., Zhang, C., Yuan, R., Wu, D., Wang, W., Pei, J., & Huang, H. (2019). Multi-horizon time series forecasting with temporal attention learning. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2527–2535. https://doi.org/10.1145/3292500.3330662.

Fan, X., Li, S., & Tian, L. (2015). Chaotic characteristic identification for carbon price and a multi-layer perceptron network prediction model. *Expert Systems with Applications, 42*(8), 3945–3952. https://doi.org/10.1016/j.eswa.2014.12.047

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: continual prediction with LSTM. *Neural Computation, 12*(10), 2451–2471. https://doi.org/10.1162/089976600300015015

Gilles, J. (2013). Empirical wavelets transform. *IEEE Transactions on Signal Processing, 61*(16), 3999–4010. https://doi.org/10.1109/TSP.2013.2265222

Göçken, M., Özçalıcı, M., Boru, A., & Dosdoğru, A.T. (2016). Integrating metaheuristics and artificial neural networks for improved stock price prediction. Expert systems with applications, 44, 320–331. https://doi.org/10.1016/j.eswa.2015.09.029.

Gonçalves, J. V. M., Alexandre, M., & Lima, G. T. (2023). ARIMA and LSTM: A comparative analysis of financial time series forecasting. *FEA/USP*. https://repec.eae.fea.usp.br/documentos/Goncalves_Alexandre_Lima_13WP.pdf.

Gonzalez-Romera, E., Jaramillo-Moran, M. A., & Carmona-Fernandez, D. (2006). Monthly electric energy demand forecasting based on trend extraction. *IEEE Transactions on Power Systems, 21*(4), 1946–1953.

Grassi, S. (2024). Examining the limitations and challenges of using Transformers for time series forecasting.

Hamzaçebi, C., Akay, D., & Kutay, F. (2009). Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. *Expert Systems with Applications, 36*(2), 3839–3844.

Han, M., Ding, L., Zhao, X., & Kang, W. (2019). Forecasting carbon prices in the Shenzhen market, China: the role of mixed-frequency factors. *Energy, 171*, 69–76. https://doi.org/10.1016/j.energy.2019.01.009

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8

Htike, Z. Z. (2013). Multi-horizon ternary time series forecasting. *2013 Signal processing: Algorithms, architectures, arrangements, and applications (SPA)*, 337–342.

Huang, N. E., Wu, Z., & Long, S. R. (2008). *Hilbert-Huang Transform, 3* p. 2544). Scholarpedia.

Keles, D., Scelle, J., Paraschiv, F., & Fichtner, W. (2016). Extended forecast methods for day-ahead electricity spot prices applying artificial neural networks. *Applied Energy, 162*, 218–230. https://doi.org/10.1016/j.apenergy.2015.09.087

Khedhiri, S. (2022). Comparison of SARFIMA and LSTM methods to model and to forecast Canadian temperature. *Regional Statistics, 12*(02), 177–194. https://doi.org/10.15196/RS120204

Khuntia, S. R., Rueda, J. L., & van Der Meijden, M. A. (2016). Forecasting the load of electrical power systems in mid-and long-term horizons: A review. *IET Generation, Transmission & Distribution, 10*(16), 3971–3977. https://doi.org/10.1049/iet-gtd.2016.0340

Kim, J., & Moon, N. (2019). BiLSTM model based on multivariate time series data in multiple fields for forecasting trading area. *Journal of Ambient Intelligence and Humanized Computing*, 1–10. https://doi.org/10.1007/s12652-019-01398-9

Kline, D. M. (2004). Methods for multi-step time series forecasting neural networks. *Neural networks in business forecasting* (pp. 226–250). IGI Global Scientific Publishing. https://doi.org/10.4018/978-1-59140-176-6.ch012

Lazcano, A., Jaramillo-Morán, M. A., & Sandubete, J. E. (2024). Back to basics: the power of the multilayer perceptron in financial time series forecasting. *Mathematics, 12*(12), 1920. https://doi.org/10.3390/math12121920

Li, J. (2017). Assessing the accuracy of predictive models for numerical data: not r nor r2, why not? Then what? *PloS One, 12*(8), Article e0183250. https://doi.org/10.1371/journal.pone.0183250

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y. X., & Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems* (p. 32). https://proceedings.neurips.cc/paper_files/paper/2019/file/6775a0635c302542da2c32aa19d86be0-Paper.pdf.

Lim, B., & Zohren, S. (2021). Time-series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A, 379*(2194), Article 20200209. https://doi.org/10.1098/rsta.2020.0209

Madhusudhanan, K., Jawed, S., & Schmidt-Thieme, L. (2024). Hyperparameter tuning MLP's for probabilistic time series forecasting. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 264–275). Singapore: Springer Nature Singapore. https://doi.org/10.48550/arXiv.2403.04477.

Mahmud, A., Noor, S. H. N. S. H., Musa, K. I., Hamzah, F. M., Yudin, Z. M., Kamaruddin, N., … Nawi, M. A. A. (2025). Hybrid ARIMA-LSTM for COVID-19 forecasting: A comparative AI modeling study. *PeerJ Computer Science, 11*, e3195.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting, 36*(1), 54–74. https://doi.org/10.1016/j.ijforecast.2019.04.014

Moghaddam, A. H., Moghaddam, M. H., & Esfandyari, M. (2016). Stock market index prediction using artificial neural network. *Journal of Economics, Finance and Administrative Science, 21*(41), 89–93.

Mounir, N., Ouadi, H., & Jrhilifa, I. (2023). Short-term electric load forecasting using an EMD-BI-LSTM approach for smart grid energy management system. *Energy and Buildings, 288*, Article 113022. https://doi.org/10.1016/j.enbuild.2023.113022

Nasir, J., Iftikhar, H., Aamir, M., Iftikhar, H., Rodrigues, P. C., & Rehman, M. Z. (2025). A hybrid LMD–ARIMA–machine learning framework for enhanced forecasting of financial time series: evidence from the NASDAQ Composite Index. *Mathematics,, 13*(15), 2389. https://doi.org/10.3390/math13152389

Ng, C. N., & Young, P. C. (1990). Recursive estimation and forecasting of non-stationary time series. *Journal of Forecasting, 9*(2), 173–204. https://doi.org/10.1002/for.3980090208

Olawoyin, A., & Chen, Y. (2018). Predicting the future with artificial neural network. *Procedia Computer Science, 140*, 383–392. https://doi.org/10.1016/j.procs.2018.10.300

Osborn, D. R. (1995). Moving average detrending and the analysis of business cycles. *Oxford Bulletin of Economics & Statistics, 57*(4). https://doi.org/10.1111/j.1468-0084.1995.tb00039.x

Paoli, C., Voyant, C., Muselli, M., & Nivet, M. L. (2010). Forecasting of preprocessed daily solar radiation time series using neural networks. *Solar Energy, 84*(12), 2146–2160. https://doi.org/10.1016/j.solener.2010.08.011

Parlos, A. G., Rais, O. T., & Atiya, A. F. (2000). Multi-step-ahead prediction using dynamic recurrent neural networks. *Neural Networks, 13*(7), 765–786. https://doi.org/10.1016/S0893-6080(00)00048-4

Penedo, S. R., Netto, M. L., & Justo, J. F. (2019). Designing digital filter banks using wavelets. *EURASIP Journal on Advances in Signal Processing, 2019*(1), 33. https://doi.org/10.1186/s13634-019-0632-6

Pirani, M., Thakkar, P., Jivrani, P., Bohara, M. H., & Garg, D. (2022). A comparative analysis of ARIMA, GRU, LSTM and BiLSTM on financial time series forecasting. In *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)* (pp. 1–6). IEEE. https://doi.org/10.1109/ICDCECE53908.2022.9793213.

Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., & Januschowski, T. (2018). Deep State space models for time series forecasting. *Advances in Neural Information Processing Systems*, 31.

Ren, Y., Qiu, X., & Suganthan, P. N. (2014). Empirical mode decomposition based adaboost-backpropagation neural network method for wind speed forecasting. In *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL)* (pp. 1–6). IEEE. https://doi.org/10.1109/CIEL.2014.7015741.

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005-2019. *Applied Soft Computing, 90*, Article 106181. https://doi.org/10.1016/j.asoc.2020.106181

Shaub, D. (2020). Fast and accurate yearly time series forecasting with forecast combinations. *International Journal of Forecasting, 36*(1), 116–120. https://doi.org/10.1016/j.ijforecast.2019.03.032

Siami-Namini, S., & Namin, A.S. (2018). Forecasting economics and financial time series: ARIMA vs. LSTM. arXiv preprint arXiv:1803.06386.

Shiblee, M., Kalra, P. K., & Chandra, B. (2009). Time series prediction with multilayer perceptron (MLP): A new generalized error-based approach. In *International Conference on Neural Information Processing* (pp. 37–44). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-03040-6_5.

Siami-Namini, S., Tavakoli, N., & Namin, A. S. (2019). The performance of LSTM and BiLSTM in forecasting time series. In *2019 IEEE International conference on big data (Big Data)* (pp. 3285–3292). IEEE. https://doi.org/10.1109/BigData47090.2019.9005997.

Soman, S. S., Zareipour, H., Malik, O., & Mandal, P. (2010). A review of wind power and wind speed forecasting methods with different time horizons. In *North American power symposium* (pp. 1–8). IEEE.

Sorjamaa, A., & Lendasse, A. (2006). Time series prediction using DirRec strategy. *In Esann, 6*, 143–148. https://www.esann.org/sites/default/files/proceedings/legacy/es2006-77.pdf.

Su, H. T., McAvoy, T. J., & Werbos, P. (1992). Long-term predictions of chemical processes using recurrent neural networks: A parallel training approach. *Industrial & engineering chemistry research, 31*(5), 1338–1352.

Taieb, S. B., Sorjamaa, A., & Bontempi, G. (2010). Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomputing, 73*(10–12), 1950–1957. https://doi.org/10.1016/j.neucom.2009.11.030

Torres, J. F., Hadjout, D., Sebaa, A., Martínez-Álvarez, F., & Troncoso, A. (2021). Deep learning for time series forecasting: A survey. *Big Data, 9*(1), 3–21. https://doi.org/10.1089/big.2020.0159

Wanchen, L. (2020). Analysis on the weight initialization problem in fully connected multi-layer perceptron neural network. In *2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE)* (pp. 150–153). IEEE. https://doi.org/10.1109/ICAICE51518.2020.00035.

Wang, X., Cai, Z., Luo, Y., Wen, Z., & Ying, S. (2022). Long time series deep forecasting with multiscale feature extraction and seq2seq attention mechanism. *Neural Process Lett, 54*(4), 3443–3466.

Yang, M., & Wang, J. (2022). Adaptability of financial time series prediction based on BiLSTM. *Procedia Computer Science, 199*, 18–25. https://doi.org/10.1016/j.procs.2022.01.003

Yang, W., Shi, J., Li, S., Song, Z., Zhang, Z., & Chen, Z. (2022). A combined deep learning load forecasting model of single household resident user considering multi-time scale electricity consumption behavior. *Applied Energy, 307*, Article 118197. https://doi.org/10.1016/j.apenergy.2021.118197

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2023). Are transformers effective for time series forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence, 37*(9), 11121–11128. https://doi.org/10.1609/aaai.v37i9.26317

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2023). *Dive into deep learning*. Cambridge University Press.

Zhang, G. P., & Qi, M. (2005). Neural network forecasting for seasonal and trend time series. *European Journal of Operational Research, 160*(2), 501–514. https://doi.org/10.1016/j.ejor.2003.08.037

Zhang, L., Zhou, W. D., Chang, P. C., Yang, J. W., & Li, F. Z. (2013). Iterated time series prediction with multiple support vector regression models. *Neurocomputing, 99*, 411–422. https://doi.org/10.1016/j.neucom.2012.06.030

Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., Zheng, S., & Li, J. (2022). Less is more: fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv*. arXiv preprint arXiv:2207.01186.