

Decomposition-Based Preprocessing Versus Architectural Complexity in Neural Forecasting of Economic and Financial Time Series: A Comparative Empirical Study

Ana Lazcano

Universidad Francisco de Vitoria

Julio E. Sandubete

`je.sandubete@ufv.es`

Universidad Francisco de Vitoria

Miguel A. Jaramillo-Morán

Universidad de Extremadura

Research Article

Keywords: Time series forecasting, Neural networks, Variational mode decomposition, Empirical mode decomposition, Hybrid forecasting models, Computational econometrics

Posted Date: October 27th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9800256/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Decomposition-Based Preprocessing Versus Architectural Complexity in Neural Forecasting of Economic and Financial Time Series: A Comparative Empirical Study

Ana Lazcano ¹ · Julio E. Sandubete ^{1,2,*} · Miguel A. Jaramillo-Morán ³

¹ Faculty of Law, Business and Government, Universidad Francisco de Vitoria, 28223 Madrid, Spain

² CRIA-BDHS Research Group, Higher Polytechnic School of Technology and Science, Universidad Camilo José Cela, 28036 Madrid, Spain

³ Department of Electrical Engineering, Electronics and Automation, School of Industrial Engineering, Universidad de Extremadura, 06006 Badajoz, Spain

* **Corresponding author:** Julio E. Sandubete, je.sandubete@ufv.es

Co-authors' emails: Ana Lazcano (ana.lazcano@ufv.es) · Miguel A. Jaramillo-Morán (miguel@uex.es)

ORCID: Ana Lazcano [0000-0002-6492-0012] · Julio E. Sandubete [0000-0002-1518-0417] · Miguel A. Jaramillo-Morán [0000-0003-4169-6983]

Abstract: Forecasting economic and financial time series with neural networks remains a methodologically active area of computational economics, particularly regarding the trade-off between predictive accuracy and model complexity. While the literature on hybrid forecasting models is extensive, the joint characterization of accuracy and architectural parsimony has received limited systematic attention. This paper examines under what conditions input preprocessing through mathematical signal decomposition can substitute for architectural complexity in neural models applied to nonstationary economic and financial series. Three neural architectures of increasing complexity are considered: a Multilayer Perceptron (MLP), a Long Short-Term Memory network (LSTM), and a Bidirectional LSTM (BiLSTM). Each architecture is estimated on raw series and on series preprocessed through five mathematically distinct decomposition techniques: trend–fluctuation decomposition, empirical mode decomposition, variational mode decomposition, empirical wavelet transform and wavelet denoising. The empirical analysis is conducted on three heterogeneous daily time series spanning equity, energy commodity and natural resource markets (Apple stock prices, Brent crude oil prices and natural gas prices), and predictive performance is evaluated through MAPE, RMSE, MSE and R². The results show that variational mode decomposition consistently provides the largest accuracy gains across all three architectures and reduces by up to one order of magnitude the number of neurons required by recurrent models to achieve competitive performance. These findings indicate that, for short-horizon forecasting of nonstationary economic and financial series, predictive performance depends more on appropriate signal preprocessing than on architectural depth, with direct implications for the design of computationally efficient forecasting pipelines in economic applications.

Keywords: Time series forecasting · Neural networks · Variational mode decomposition · Empirical mode decomposition · Hybrid forecasting models · Computational econometrics

JEL Classification: C45 · C53 · C58 · G17

1. Introduction

Time series forecasting has become a very active research area in recent years. Accurate and reliable tools have been developed, that are potentially very useful in several practical applications, ranging from the economic field to engineering and scientific applications. The first tools developed were mainly based on statistical or regression techniques that try to algorithmically approximate the dynamics of the time series. Probably, the most widely used of such tools was the well-known Autoregressive Moving Integrated Moving Average (ARIMA) (Wilson, 2016). Although it has been used to predict many time series for years, it has problems when dealing with hard nonlinear time series because of its linear structure. Therefore, other tools have been also proposed to try to overcome its performance: regression models (Wu et al., 2013), Kalman filters (Takeda et al., 2016) or Generalized Autoregressive Conditional Heteroscedasticity (GARCH) (Conrad et al., 2012). Another option has been also proposed: to combine two or more forecasting tools to provide a joint prediction that is assumed to be more accurate than those provided by each model independently. So ARIMA has been used along with Fourier Series Expansion and Particle Swarm Optimization (PSO) to forecast electricity consumption (Wang et al., 2012), or with Autoregressive Conditional Heteroscedasticity (ARCH) to forecast CO2 emissions (Dritsaki & Dritsaki, 2020).

Nevertheless, Machine Learning has emerged as a more feasible and accurate option to predict time series because of their ability to deal with any kind of complex problem, most of them closely related to human intelligence. They have been able to predict time series that the classical statistical tools had problems to deal with, especially strongly nonlinear time series. For this reason, machine-learning-based time series forecasting has become an active and fruitful research field.

There are several machine learning tools that have been used to predict different types of time series: Random Forest (RF), Gradient Boosting (GB) and Extreme Gradient Boosting (XGBoost) (Lucas et al., 2020), Gaussian Process Regression (Feng et al., 2022) or Support Vector Machine (SVM) (Zhu et al., 2017). Among these methods, Artificial Neural Networks (ANNs) have received particular attention because of their strong predictive capabilities, often surpassing those of classical statistical models. They have attracted the attention of researchers due to their strong predictive capabilities, often surpassing those of classical statistical models. This growing interest is largely attributed to their flexibility in modeling complex, nonlinear dynamics inherent in time series data, without relying on predefined assumptions about the structure of the input signals. ANNs have clearly demonstrated their effectiveness in time series prediction, with a vast literature on the subject.

Many studies highlight how ANNs significantly outperform classical forecasting tools. In (Hill et al., 1996), the accuracy of artificial neural network models is evaluated against traditional models used to predict time series, revealing a clear advantage of ANNs over classical techniques. This line of research later evolved to incorporate hybrid approaches, combining both methodologies to enhance prediction accuracy, yielding improved performance metrics (Khashei & Bijari, 2010). Some investigations have specifically focused on complex time series, where promising forecasting results have also been achieved (Bhardwaj et al., 2020; Di Piazza et al., 2021; Kumar et al., 2023).

ANNs have been widely used in real-world applications to forecast variables related to economy or energy. Thus, they have been used to forecast electric power transactions (Bak & Bae, 2020), financial time series (Fang et al., 2023), raw materials and electricity prices (Lazcano et al., 2024), stock market variables (Moghaddam et al., 2016), electricity prices (Keles et al., 2016) or CO2 emission allowance prices (Jaramillo-Morán et al., 2021).

Research on time series forecasting using ANN techniques encompasses various models, some of which have very simple structures, such as the Multilayer Perceptron (MLP) (Borghi et al., 2021; Chen et al., 2023; Voyant et al., 2015), which is widely used due to its ability to approximate measurable functions, even in the presence of nonlinearities or missing data.

Nevertheless, the development of new complex neural structures capable of handling large datasets with strong temporal dependencies has attracted the attention of researchers. Those models are known as Deep

Learning neural networks because they are made up of many layers and neurons (processing elements) in each one. A clear example is LSTM (Long Short-Term Memory) networks, initially designed for processing text and speech but later widely adopted for time series forecasting due to their remarkable predictive performance. Thus, it seems logical to suppose that they can also provide accurate predictions of time series, which usually are strongly time dependent. They have been used to predict electric energy load (Ciechulski & Osowski, 2021), consumption in building (Hwang et al., 2020), or CO2 emission allowance prices (Jaramillo-Morán et al., 2021).

Despite its proven accuracy and reliability as a forecasting tool, LSTM has been extended through an improved variant, the Bidirectional Long Short-Term Memory (BiLSTM) network. Unlike traditional LSTMs, BiLSTM processes input sequences twice, once in the forward direction and once in the backward direction, resulting in greater efficiency (Siarni-Namini et al., 2019).

LSTMs have been also combined with other structures to improve performance. They have been used along with a Convolutional Neural Network (CNN, another class of Deep Learning network) to forecast short-term electric load (Guo et al., 2020). In (Yang W. et al., 2022) it was combined with an MLP and an XGBoost to predict consumption. In (Yang D. et al., 2022) a Genetic Algorithm was used to optimize a LSTM. In (Dong & Zhou, 2024) an ARIMA-CNN-LSTM model was used to forecast gold prices.

Building on the strategy of combining techniques to develop hybrid forecasting models, some researchers have proposed alternative approaches that enhance predictions by preprocessing time series data. These preprocessing methodologies have been shown to significantly improve forecasting performance (Pabuçcu, 2017) by refining the input data before model training.

Selecting the optimal preprocessing strategy for a given time series requires the application of a variety of analytical techniques. As an initial step, it is essential to conduct a statistical evaluation of the dataset to identify and remove outliers or inconsistent entries that could bias the analysis (Famili et al., 1997). This stage can improve the quality and stability of the data, contributing to greater consistency and forecastability (Cortés-Ibáñez et al., 2020; Fong et al., 2013), but is not useful when the time series to be forecasted does not have such problems. However, preprocessing is potentially a very powerful strategy whenever it is applied in a wider way: to decompose the original series into subcomponents exhibiting more uniform dynamics, thereby simplifying subsequent modelling efforts.

This method relies on the assumption that many real-world time series are often shaped by recurring patterns linked to social or environmental influences. By isolating subseries aligned with specific cycles or frequencies, the decomposition facilitates a more accurate representation of the underlying structure and enhances forecasting precision. As a result, this decomposition yields a structure that is easier to model and predict, ultimately leading to more precise forecasts (Kotsiantis et al., 2006). The aim is to decompose the original dataset into several subseries that can be independently forecasted and then summed up to provide a prediction of the original time series. These subseries are supposed to be related to specific frequencies embedded in the overall behavior of the time series, and, consequently, can be more accurately predicted.

One of those techniques is the Empirical Mode Decomposition (EMD), which splits a time series into several subseries, each one with proper oscillatory behavior. As the whole process has a heuristic behavior, it suffers from a lack of mathematical support. The Variational Mode Decomposition has been developed to overcome this problem. Both algorithms have been applied to different forecasting tools such as ARIMA (Angamuthu Chinnathambi et al., 2018), SVM (Bak & Bae, 2020), GARCH and SVM (Zhu et al., 2018), MLP (Jaramillo-Morán et al., 2021; Yang D. et al., 2022), CNN (Shen et al., 2021) or LSTM (Agarwal et al., 2025; Yang D. et al., 2022).

Another tool that has also attracted the attention of researchers is the Empirical Wavelet Transform (EWT), which represents a generalization of the Fourier transform to both time and space. It is a fruitful research field of applied mathematics that explores the periodic spatial-temporal behavior of datasets. Therefore, several

authors have explored its use as a preprocessing tool for time series forecasting: in (Joo & Kim, 2015) with an ARIMA, and in (Yemets et al., 2025) with an LSTM.

All these works showed that including a preprocessing stage in the process of time series forecasting provides an improvement of accuracy, so that models with preprocessing can outperform those without it. Thus, it has become a step to be considered in the forecasting process, because, as the works have proved, it has improved performance of the forecasting tools when properly selected and applied to the time series to be predicted.

Building on this literature, this paper makes the following contributions. First, we incorporate new knowledge in the field of time series forecasting focusing on the integration of preprocessing tools, tailored to the characteristics of time-dependent data, with ANNs. The study involves a comparative analysis of various preprocessing methodologies and their impact on the performance of neural models. Experimental findings suggest that appropriate preprocessing allows for the design of simpler, and therefore more computationally efficient, network architectures than those applied directly to unprocessed data. Second, in this work, a comprehensive review of several preprocessing techniques suitable to be combined with the neural models that have been proved to be accurate and reliable forecasting tools will be provided. Third, several forecasting models obtained by combining those processing techniques with the selected neural models will be tested to prove that models with preprocessing provide more accurate predictions than those without it. In fact, we want to test whether a high number of neurons in a neural network model implies (or does not) an improvement in evaluation metrics, whereas proper data preprocessing does lead to better outcomes. The validity of the proposed approach is empirically assessed on three economic and financial time series representative of different price-generating processes, with implications for the design of computationally efficient forecasting pipelines in applied computational economics.

This document is divided into the following sections: the methodology section describes in detail the different architectures employed, and the preprocessing techniques targeted by this analysis. The experimentation section provides information on the data used, the hyperparameters selected, and the metrics used to evaluate the model. The results section presents the metrics obtained in the analysis and the graphical results. Finally, the discussion and conclusions sections summarize the key findings of the study and propose future research directions.

2. Methodology

This study examines whether preprocessing can improve the forecasting performance of neural networks and, at the same time, reduce the model complexity required to achieve competitive accuracy in economic and financial time series prediction (Khedhiri, 2022; Paoli et al., 2010). To address this question, three neural forecasting models are considered: a Multilayer Perceptron, a Long Short-Term Memory network, and a Bidirectional Long Short-Term Memory network. Each architecture is evaluated under two alternative settings: a direct specification, in which forecasts are generated from the raw series, and a hybrid specification, in which the original series is first decomposed into a set of subseries with a more or less homogeneous behavior through preprocessing and the resulting components are forecast separately. This design allows us to assess whether preprocessing provides better accuracy than direct forecasting.

Let x_t denote the observed value of an economic or financial price series at time t . In the direct approach, the one-step-ahead forecast is obtained from the original series as

$$\hat{x}_{t+1} = f_{\theta}(\mathbf{z}_t) \quad (1)$$

where $f_{\theta}(\cdot)$ denotes the neural forecasting model and $\mathbf{z}_t$ is the corresponding vector or sequence of lagged inputs. In the hybrid approach, the original series is first decomposed into M components:

$$x_t = \sum_{m=1}^M c_{m,t} \quad (2)$$

where $c_{m,t}$ denotes the m -th component delivered by the preprocessing stage. Each component is then forecast independently using the same class of neural model, and the final prediction is reconstructed by aggregation:

$$\hat{x}_{t+1} = \sum_{m=1}^M \hat{c}_{m,t+1} \quad (3)$$

Accordingly, each hybrid forecasting structure combines one preprocessing method with as many neural sub-models as components generated by the decomposition. Since the number of components depends on the preprocessing method and on the specific time series under analysis, the final hybrid structure may differ across series.

2.1 Neural network architectures

The empirical analysis includes three neural architectures that represent different levels of structural complexity in time series forecasting.

2.1.1 Multilayer Perceptron

The first model is a Multilayer Perceptron (MLP). It is implemented as a feedforward neural network with an input layer followed by one or several hidden layers and a linear output layer (Fan et al., 2015; Göçken et al., 2016; Han et al., 2019; Hornik et al., 1989). In time series forecasting the input layer is made up of several past elements that will be processed by the network to provide a prediction of a future value. Although several future predictions can be provided if desired, in this work, a single value is forecast. Therefore, the output linear layer has a single neuron. The number of elements of the input layer (input data) must be empirically fixed by testing alternative input sizes (Madhusudhanan et al., 2024; Wanchen, 2020; Zhang T. et al., 2022).

2.1.2 Long Short-Term Memory

The second estimator is a Long Short-Term Memory (LSTM) network. LSTM models are designed to process sequential information by combining the inputs to the network with a sort of memory of past inputs. This allows them to capture temporal dependence more flexibly than standard feedforward networks (Gers et al., 2000; Hochreiter & Schmidhuber, 1997). In the present study, the LSTM specification consists of one or several recurrent layers (LSTM layers) followed by a fully connected output layer. The input to the model is defined as the observation immediately preceding the one to be predicted, while the recurrent structure provides the temporal updating mechanism through its internal state. As with the MLP, the fully connected layer has a single neuron, since a one-step-ahead prediction is provided.

2.1.3 Bidirectional Long Short-Term Memory

The third architecture is a Bidirectional Long Short-Term Memory (BiLSTM) network. This architecture extends the standard LSTM by processing the input sequence in both forward and backward directions before combining the resulting representations in the output layer (Kim & Moon, 2019). In the context of this study, BiLSTM is included as a higher-capacity recurrent alternative to test whether additional architectural complexity improves forecasting performance relative to both the conventional LSTM and the simpler MLP benchmark. As in the LSTM case, the empirical implementation is configured for one-step-ahead prediction.

2.2 Signal decomposition techniques

Five preprocessing strategies are considered to transform the original series into several components with potentially simpler dynamics: trend–fluctuation decomposition, empirical mode decomposition (EMD),

variational mode decomposition (VMD), empirical wavelet transform (EWT), and wavelet denoising. The rationale is the same in all cases: if the original signal is first decomposed into smoother or more homogeneous components, each component may become easier to forecast than the raw series itself. The hybrid forecast is then obtained by aggregating the component-level predictions.

2.2.1 Trend–fluctuation decomposition

The first preprocessing strategy separates the original series into a low-frequency trend and a residual fluctuation component by means of a moving-average filter (Arianos & Carbone, 2007; Osborn, 1995). Let x_t denote the original series and T_t its smoothed trend component. The fluctuation component is then defined as the difference between the observed value and the extracted trend:

$$F_t = x_t - T_t \quad (4)$$

The original series can therefore be represented as the sum of a slowly varying component and a short-run fluctuation component. In the empirical implementation, both series are forecast separately, and their predictions are added to reconstruct the final forecast of the original series. This decomposition is intended to reduce the burden on the forecasting model by isolating long-run movement from short-run variability (Gonzalez-Romera et al., 2006).

2.2.2 Empirical Mode Decomposition (EMD)

The second preprocessing strategy is Empirical Mode Decomposition (EMD), which decomposes the original series into a set of intrinsic mode functions (IMFs) and a residual term (Huang et al., 2008; Ren et al., 2014). EMD is especially suitable for nonlinear and nonstationary signals because it separates the original series into oscillatory components associated with different time scales. The decomposition can be expressed as

$$x_t = \sum_{k=1}^K IMF_{k,t} + r_t \quad (5)$$

where $IMF_{k,t}$ denotes the k -th intrinsic mode function and r_t is the residual component. In the hybrid forecasting framework used in this study, each IMF and the residual are predicted separately, and the component forecasts are then summed to obtain the prediction of the original time series. The EMD algorithm is detailed in Appendix A.1.

2.2.3 Variational Mode Decomposition (VMD)

The third preprocessing strategy is Variational Mode Decomposition (VMD). Unlike EMD, which is based on an empirical sifting procedure, VMD formulates the decomposition as an optimization problem in which the signal is split into a finite set of modes with limited bandwidth around specific center frequencies (Boyd et al., 2011; Dragomiretskiy & Zosso, 2014). Formally, the original series is represented as

$$x_t = \sum_{k=1}^K u_{k,t} \quad (6)$$

where each mode $u_{k,t}$ captures a different frequency band of the original signal. In practical terms, VMD aims to produce a decomposition into smoother and more homogeneous subseries, which may be easier to forecast with neural models than the raw signal. In the empirical design of this study, each mode is forecast separately, and the final prediction is obtained by aggregating the mode-specific forecasts. The VMD algorithm is detailed in Appendix A.2.

2.2.4 Empirical Wavelet Transform (EWT)

The fourth preprocessing strategy is the Empirical Wavelet Transform (EWT), which builds an adaptive wavelet filter bank after partitioning the Fourier spectrum of the signal into data-driven frequency bands, each associated to a frequency peak. This procedure yields a set of empirical modes associated with different spectral regions of the original series. As in the previous decomposition-based methods, the idea is to replace a single complex forecasting task with several simpler component-level forecasting tasks. Once the EWT components are obtained, each of them is predicted separately and the final forecast is reconstructed by aggregation (Gilles, 2013; Penedo et al., 2019). The EWT algorithm is detailed in Appendix A.3.

2.2.5 Wavelet denoising

The fifth preprocessing strategy is based on wavelet denoising. In this case, the series is first transformed into the wavelet domain, where the coefficients associated with high-frequency noise are filtered out. The signal is then reconstructed from the retained coefficients, yielding a smoothed version of the original series. This produces a low-frequency approximation component, interpreted as the denoised signal, and a high-frequency detail component, which is discarded. This denoised series is then subtracted from the original series to obtain the corresponding noise component. Both denoised and noise series are independently forecasted, and their predictions added. This preprocessing strategy is detailed at the end of Appendix A.3.

For all preprocessing methods, the hybrid forecasting procedure follows the same sequence. First, the original series is transformed into a set of components. Second, one of the three selected neural networks is chosen to forecast all components. Third, each network produces a one-step-ahead forecast for each component. Finally, the individual component forecasts are added to recover the forecast of the original series. This unified procedure ensures that differences across hybrid models are attributable to the preprocessing method itself rather than to changes in the forecasting logic.

The comparison between direct and hybrid specifications is therefore organized around a common forecasting task, a common prediction horizon, and a common model selection strategy. This makes it possible to evaluate, in a controlled way, whether preprocessing improves predictive performance and whether such improvements are accompanied by a reduction in network size. In this sense, the methodology is explicitly designed to test the central hypothesis of the paper: decomposing the original time series into subseries with homogeneous behaviors provides an improvement of performance of the forecasting tools.

3. Experimental setup

The main aim of this paper is to identify the advantages of applying preprocessing techniques to a time series before forecasting it with neural networks. This defines a hybrid model consisting of a preprocessing tool and a neural model. Different structures will be proposed by combining the preprocessing tools and neural networks described in the previous section. These hybrid models will be tested alongside the neural models alone to determine whether the hybrid structures provide greater accuracy than models without preprocessing.

The forecasting process performed by the hybrid models starts with decomposing the original time series into a set of subseries. Each subseries is then forecasted independently by a neural network. Although all the neural networks are of the same model with the same structure, they differ because they are trained to predict different subseries. Finally, the predictions are summed up to provide a prediction for the original time series. Thus, each hybrid forecasting tool consists of a preprocessing tool and several neural networks equal to the number of subseries provided by this preprocessing tool. It is worth noting that the preprocessing tool can provide different numbers of subseries for different time series, and, therefore, a hybrid model can have different numbers of neural networks depending on the time series it is predicting.

Forecasting will be carried out with three financial time series with different trends and behaviors, which will be forecasted with each one of the different hybrid models defined. The three neural models alone will also be used. The results provided by the hybrid models and the neural models alone will be compared to

determining which structure performs better for each time series. To properly evaluate performance, a set of error metrics will be used.

All predictive models applied to the time series in this study are configured to generate a one-step-ahead forecast, utilizing historical data to estimate the next value in the sequence. Specifically, future price prediction is performed by processing past prices.

3.1 Datasets

Three time series were selected for prediction purposes. These three series were chosen for their distinct characteristics, which will allow the validity of the hypothesis to be assessed using the selected models and preprocessing techniques. 75% of the time series data were used for training the chosen models, while the remaining 25% was reserved for assessing their performance.

Apple stock prices complete the first selected series, which yielded a total of 5,215 observations, corresponding to a daily period from May 7, 2004, for 20 years, ending on May 8, 2024. This series, represented in Figure 1, displays a marked upward trend, which poses a significant forecasting challenge due to the inherent difficulty in modeling long-term growth patterns. Moreover, the fluctuations present throughout the series increase its complexity, particularly when using neural network models. In this context, the application of preprocessing techniques may prove valuable, as they can enhance the interpretability and modeling capacity when dealing with time series exhibiting such nonstationary and volatile behavior.

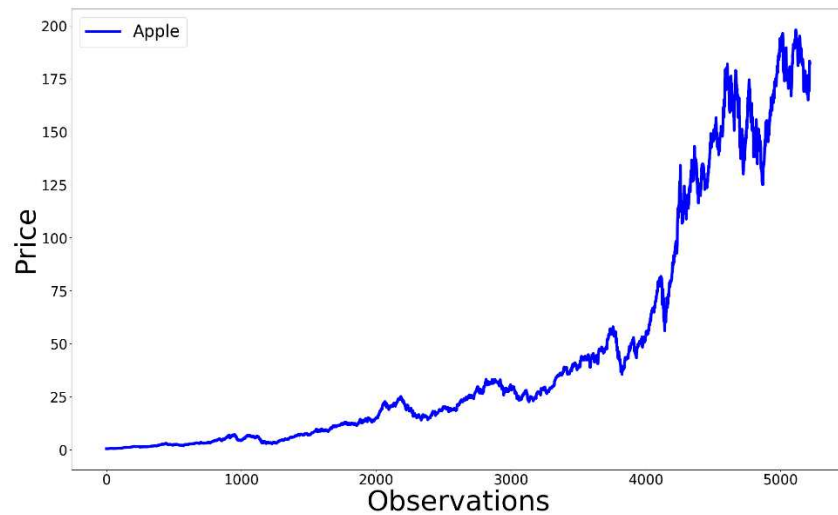


Figure 1. Apple time series prices.

A second time series with a different behavior from the previous one has been also forecasted: the Brent oil prices. It comprises a total of 9,318 observations collected from May 20, 1987, to May 5, 2024. This series exhibits significant variations in its trend throughout its entire span, see Figure 2.

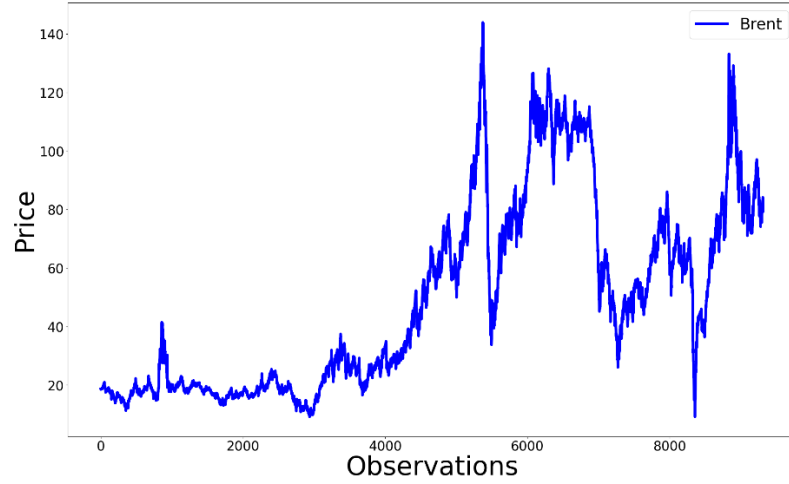


Figure 2. Brent time series prices.

The last of the time series included in this study corresponds to natural gas prices from January 8, 1997, to February 22, 2024. This series exhibits high volatility, as it is impacted by supply and demand, as well as political factors, highlighting the need to focus on adaptive models to improve predictions, see Figure 3.

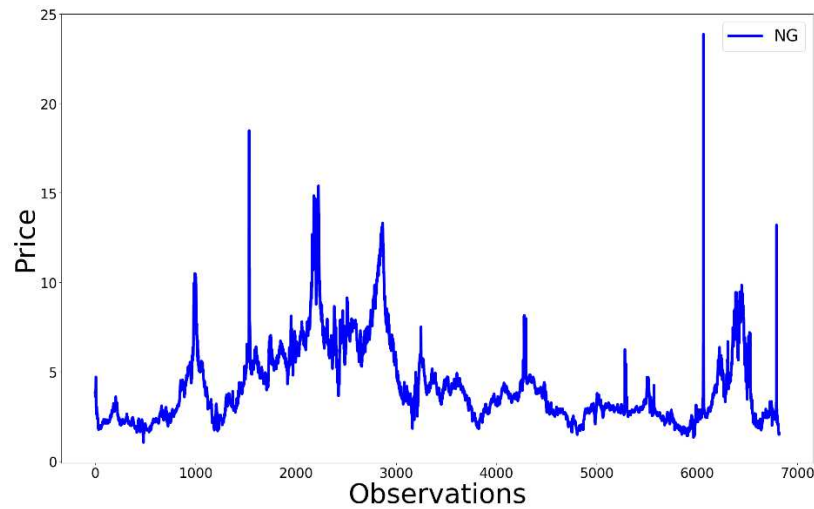


Figure 3. Natural Gas time series prices

3.2 Network configuration and hyperparameters

When working with neural networks, it is necessary to define the structure of the model, e.g., number of layers, neurons in each layer, and a set of hyperparameters that control the learning process of the neural network.

The first step when defining the network is to fix the number of inputs and outputs of the models. As we are performing a regression process the number of inputs will be the number of past data to be provided to the network. Several sizes were tested for the MLP, ranging from 1 to 10, and the best performance was achieved with 3 inputs. In the case of the LSTM and BiLSTM the input consists of a single input (that immediately precedes the one to be predicted), as the recurrent structure of LSTM networks inherently captures dependencies across sequences through their internal memory. The output was a single value since only one-step-ahead predictions are provided for the three neural models.

Regarding model architecture, a single hidden layer was selected for the Multi-Layer Perceptron (MLP). This decision is grounded in prior research indicating that a one-hidden-layer architecture, if composed of a

sufficient number of neurons, is capable of approximating the behavior of any dynamic system with high accuracy [54]. Since the preprocessing stage provides series that can have positive and negative values, the hyperbolic tangent has been selected as the activation function of the neurons in the hidden layer, while a linear activation function was chosen for the output layer, the most usual configuration for regression problems.

The architectures of the LSTM and BiLSTM networks are slightly different. These models consist of one or several LSTM layers, followed by a fully connected layer as the output one. Alternative configurations involving up to three LSTM layers were explored and the best results were obtained with a single layer. Models with two and three hidden layers not only did not improve predictive performance but also required longer training times. As with the MLP, the regression layer has a single neuron to provide a one-step-ahead prediction.

Along with the number of layers, the number of neurons in each layer is a key issue when defining a neural network's structure. Currently, the identification of the optimal number of neurons can be performed using grid search models (Luo & Tian, 2020), which are computationally expensive and time-consuming. As a result, these approaches tend to be inefficient and offer only limited improvements in predictive performance. Instead, a trial-and-error procedure is preferred in most cases. Therefore, predictions will be generated by varying the number of neurons across different models. This variation will be applied in a structured manner by increasing the number of neurons until no improvement in accuracy is achieved. In this work values ranging from 1 to 200 were tested and the configuration that gave the best accuracy was identified. This process was repeated for each preprocessing-neural model configuration, as well as with the neural models alone. The results show that preprocessing can produce more accurate predictions than the best performing neural configuration alone, and in most cases with fewer neurons.

Another step that is needed when simulating neural networks is adjustment of a series of hyperparameters that determine how the learning process is performed. A correct selection of these values is key to achieving a well-performing and precise model. All programming environments that provide neural models for simulation allow for adjusting quite a lot of hyperparameters for the users to customize his model. They provide default values for them all, although the user can modify them. For the sake of simplicity, we have only modified those with a higher influence in accuracy:

- Learning rate: Determines the weight update process throughout training based on gradient descent. A lower value results in greater convergence but slows down the process, while a higher rate results in faster performance but compromises accuracy.
- Batch size: Identifies the number of observations to be analyzed before updating the parameters. This depends largely on memory and convergence speed.
- Epochs: Reflects how many times the entire data set will be processed by the model. A high value for this parameter improves learning but increases the risk of overfitting.
- Optimization Algorithm: Determines the method used to adjust weights based on the computed gradients. One of the most frequently adopted optimizers for LSTM models is Adam, which integrates the benefits of RMSProp and momentum-based techniques to improve convergence efficiency (Sun, 2019). In this study, this approach has been employed. For the MLP architecture, the Levenberg-Marquardt algorithm has been selected as the training method. This algorithm merges features from both gradient descent and Gauss-Newton methods, making it a classical and effective choice for training this type of neural network (Bishop, 2006). However, it does not allow the training dataset to be divided into batches.

Several values were tested by trial-and-error. Those performing best are presented in Table 1.

Table 1. Parameters selected.

<i>Model</i>	<i>Learning rate</i>	<i>Batch</i>	<i>Epoch</i>	<i>Hidden Layers</i>	<i>Optimizer</i>
LSTM	0.001	30	100	1	Adam
MLP	0.001		100	1	Levenberg-Marquardt
BiLSTM	0.001	30	100	1	Adam

All simulations were conducted in the MATLAB R2024a environment, executed on a personal computer equipped with an Intel i9 processor and an RTX4070 graphics card, which enabled efficient execution of the computational experiments.

The purpose of this configuration is not to claim universal optimality for any neural architecture, but to ensure a stable and consistent empirical framework in which the relative effect of preprocessing can be assessed. In this sense, the methodological comparison is intentionally centered on two dimensions: forecast accuracy and architectural parsimony.

3.3 Evaluation metrics

The error metrics used to evaluate the accuracy of the models and preprocessing are described below, having been selected due to their wide use in the existing literature (Botchkarev, 2019).

Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{N} \sum_{t=1}^N (x_t - \hat{x}_t)^2}$$

Mean Squared Error (MSE):

$$MSE = \frac{1}{N} \sum_{t=1}^N (x_t - \hat{x}_t)^2$$

Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{1}{N} \sum_{t=1}^N \frac{|x_t - \hat{x}_t|}{|x_t|} \times 100$$

R-squared (R^2):

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

The time series over time t is denoted as x_t the predicted values are identified as $\hat{x}_t$, the mean of the series is $\bar{x}_t$, and the total number of observations is represented as N .

The evaluation of the predictive capacity of the models is measured through MAPE and RMSE; a lower result indicates greater accuracy. Model performance is measured through MSE and R^2 , which allow us to analyze the explained variability of the predictions.

Using R^2 provides an assessment of how the model performed. Its interpretability makes this metric one of the most valuable tools for comparing multiple studies. Combining it with the other metrics used in this study provides a comprehensive view of model fit, with results that allow for easy communication (Olawoyin

& Chen, 2018). The application of the R^2 metric in nonlinear contexts provides extensive information about the model's explanation based on the dependent variable using variance. This metric contributes to obtaining clear results and a more rigorous evaluation when used in combination with other metrics (Li, 2017).

4. Experimental results

The time series used are processed with the neural network models described, which are recognized in the literature as benchmark models for predicting economic time series (Pirani et al., 2022; Yang & Wang, 2022). For each of these models, tests are conducted with different numbers of neurons and combinations of preprocessing techniques to identify the most efficient way to make predictions.

The EMD algorithm generated a different number of IMFs and its corresponding residue, depending on the time series analyzed. Specifically, it produced 7 components (IMFs and residue) for the Apple stock series, 9 for the Brent crude oil series, and 10 for the natural gas series. VMD also generates a variable number of modes for different times series. It provides the same number for the three series used: 6. For the EWT algorithm several values of the maximum number of peaks -each associated to a frequency band- were tested and the best performance was achieved with 5. For the Moving Average decomposition, several sizes of the sliding window were tested, ranging from 1 to 10, and the best performances were achieved with five preceding data points. Regarding the denoising process using wavelets, the Daubechies wavelet with five vanishing moments (db5) was selected, employing a single-level decomposition.

Table 2. Summary results with error metrics for the time series.

Series	Model	Neurons	Preprocessing technique	Metrics			
				MAPE	RMSE	MSE	R2
Brent	MLP	3	VMD	0,919	0,802	0,644	0,998
	MLP	1	Direct	1,912	1,674	2,802	0,992
	LSTM	5	VMD	1,453	1,254	1,574	0,996
	LSTM	11	Direct	2,484	2,090	4,370	0,988
	BiLSTM	20	VMD	0,862	0,895	0,802	0,997
	BiLSTM	100	Direct	1,465	1,259	1,585	0,995
Natural Gas	MLP	3	VMD	2,683	0,307	0,094	0,968
	MLP	35	Direct	4,382	0,520	0,270	0,908
	LSTM	5	VMD	2,951	0,300	0,090	0,969
	LSTM	13	Direct	4,413	0,550	0,303	0,897
	BiLSTM	50	VMD	1,336	0,096	0,009	0,996
	BiLSTM	200	Direct	2,450	0,241	0,058	0,980
Apple	MLP	3	EWD	6,485	14,265	203,496	0,890
	MLP	1	Direct	12,909	24,939	621,957	0,666
	LSTM	100	VMD	2,398	4,534	20,564	0,989
	LSTM	160	Direct	2,643	4,411	19,464	0,989
	BiLSTM	50	EMD	1,139	2,556	6,533	0,996
	BiLSTM	130	Direct	1,456	3,594	12,920	0,993

Table 2 summarizes the results for each time series with each neural network model, highlighting the best results across the three time series and the preprocessing method associated with them. It shows how the BiLSTM model, in combination with VMD (Brent and Natural Gas) and EMD (Apple) preprocessing, delivers the best results. Furthermore, it allows for a significant reduction in the number of neurons and simplifies the model architecture. It is worth noting that in most cases the hybrid model (preprocessing-neural network) has

been able to outperform the forecasting tool alone. In most cases, VMD was the preprocessing technique associated with the best forecasting accuracy. Detailed results for each of the time series and preprocessing methods studied are presented in the following tables.

Table 3. Error metrics for the Apple time series when preprocessing is applied.

		Metrics			
Model	Preprocessing technique	MAPE	RMSE	MSE	R2
LSTM (100 neurons)	TF	3,234	5,956	35,475	0,981
	Wavelet	2,557	4,636	21,498	0,988
	EMD	2,660	4,757	22,637	0,987
	VMD	2,398	4,534	20,564	0,989
	EWD	5,879	10,079	101,593	0,945
MLP (3 neurons)	TF	30,035	56,298	3,169e+03	-0,6999
	Wavelet	31,918	59,773	3,572e+03	-0,916
	EMD	8,793	18,069	326,522	0,824
	VMD	26,065	49,904	2,490e+03	-0,335
	EWD	6,485	14,265	203,496	0,890
BiLSTM (50 neurons)	TF	3,557	7,346	53,970	0,971
	Wavelet	4,888	9,745	94,976	0,949
	EMD	1,139	2,556	6,533	0,996
	VMD	10,088	18,834	354,746	0,809
	EWD	10,462	17,835	318,092	0,829

Table 4. Error metrics for the Apple time series without preprocessing.

		Metrics			
Model	No. Neurons	MAPE	RMSE	MSE	R2
LSTM	160	2,643	4,411	19,464	0,989
MLP	1	12,909	24,939	621,957	0,666
BiLSTM	130	1,456	3,594	12,920	0,993

The analysis of the results shown in Tables 3, 5, and 7 presents the outcomes of applying preprocessing techniques to the data, indicating the different models used and the number of neurons required in each case. For comparative purposes, Tables 4, 6, and 8 are included, showing predictions made to the data without applying any prior preprocessing and the number of neurons that yielded the best result in each series. In all cases, each table corresponds to a specific time series.

Table 5. Error metrics for the Brent time series when preprocessing is applied.

		Metrics			
Model	Preprocessing technique	MAPE	RMSE	MSE	R2
LSTM (5 neurons)	TF	3,288	2,618	6,854	0,982
	Wavelet	3,176	2,685	7,211	0,981
	EMD	8,811	6,011	36,132	0,907
	VMD	1,453	1,254	1,574	0,996
	EWD	2,216	1,986	3,945	0,989
MLP	TF	1,945	1,688	2,850	0,992

(3 neurons)	Wavelet	1,259	1,264	1,598	0,995
	EMD	4,875	3,778	14,279	0,963
	VMD	0,919	0,802	0,644	0,998
	EWD	1,925	1,662	2,763	0,992
BiLSTM (20 neurons)	TF	1,333	1,244	1,549	0,996
	Wavelet	1,580	1,367	1,870	0,995
	EMD	2,708	1,824	3,327	0,991
	VMD	0,862	0,895	0,802	0,997
	EWD	1,103	1,005	1,011	0,997

Table 6. Error metrics for the Brent time series without preprocessing.

Metrics					
Model	Nº Neurons	MAPE	RMSE	MSE	R2
LSTM	11	2,484	2,090	4,370	0,988
MLP	1	1,912	1,674	2,802	0,992
BiLSTM	100	1,465	1,259	1,585	0,995

Table 7. Error metrics for the Natural Gas time series when preprocessing is applied.

Metrics					
Model	Preprocessing	MAPE	RMSE	MSE	R2
LSTM (5 neurons)	TF	4,712	0,590	0,348	0,882
	Wavelet	3,239	0,536	0,287	0,902
	EMD	6,220	0,632	0,400	0,865
	VMD	2,951	0,300	0,090	0,969
	EWD	4,103	0,547	0,299	0,899
MLP (3 neurons)	TF	4,420	0,600	0,361	0,878
	Wavelet	3,225	0,558	0,311	0,895
	EMD	5,194	0,597	0,357	0,879
	VMD	2,683	0,307	0,094	0,968
	EWD	4,029	0,540	0,291	0,901
BiLSTM (50 neurons)	TF	2,250	0,168	0,028	0,990
	Wavelet	2,704	0,345	0,119	0,959
	EMD	2,468	0,293	0,086	0,970
	VMD	1,336	0,096	0,009	0,996
	EWD	1,310	0,213	0,045	0,984

Table 8. Error metrics for the Natural Gas time series without preprocessing.

Metrics					
Model	Nº Neurons	MAPE	RMSE	MSE	R2
LSTM	13	4,413	0,550	0,303	0,897
MLP	35	4,382	0,520	0,270	0,908
BiLSTM	200	2,450	0,241	0,058	0,980

Figures 4, 6, and 8 describe in detail the evolution of MAPE, RMSE, and R^2 values as the number of neurons increases. MSE metrics have not been included in the graphs, as their high error in some series would negatively impact the visualization of the figures. The values shown in the tables clearly indicate that the use of VMD preprocessing produces the best results in most series, with EMD preprocessing only predominating in the series related to Apple, as can be seen more clearly when graphically representing the best performance indices for both prediction without and with preprocessing (Figures 5, 7, and 9, where MSE is not represented for the same reason it was not included in Figures 4, 6, and 8). On the other hand, when comparing these results from the perspective of the number of neurons, it can be inferred that RNN-based models (LSTM and BiLSTM) generally require a larger number of neurons than MLP models to achieve their best performance.

A detailed analysis of each series reveals that, in the case of the Apple series (Tables 3 and 4 and Figure 5), the LSTM and BiLSTM models require more than 120 neurons to achieve optimal performance when predicting without preprocessing, which entails significant computational costs. In contrast, the MLP model achieves its best performance with only a single neuron (Figure 4). However, after preprocessing the Apple series, the LSTM model attains a lower MAPE with 100 neurons while preserving the same R^2 as the direct specification, although its RMSE and MSE are slightly higher. In contrast, the preprocessed BiLSTM model improves all error metrics with only 50 neurons, resulting in substantial computational savings. Although the preprocessed MLP model uses more neurons (3 versus 1 in the raw model), its efficiency improves considerably, whereas increasing the number of neurons in the non-preprocessed tests degrades the model's performance. Overall, the preprocessed Apple series demonstrates improved results with lower resource requirements.

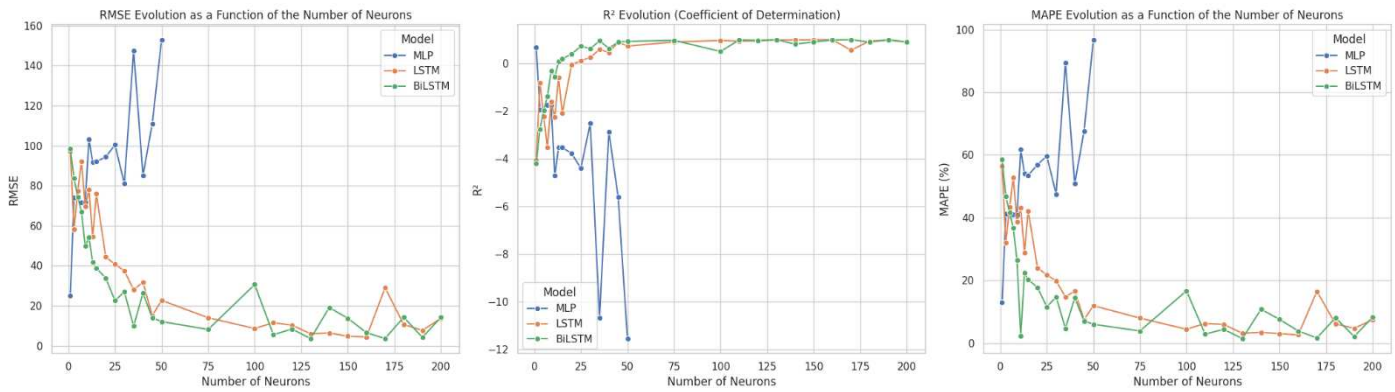
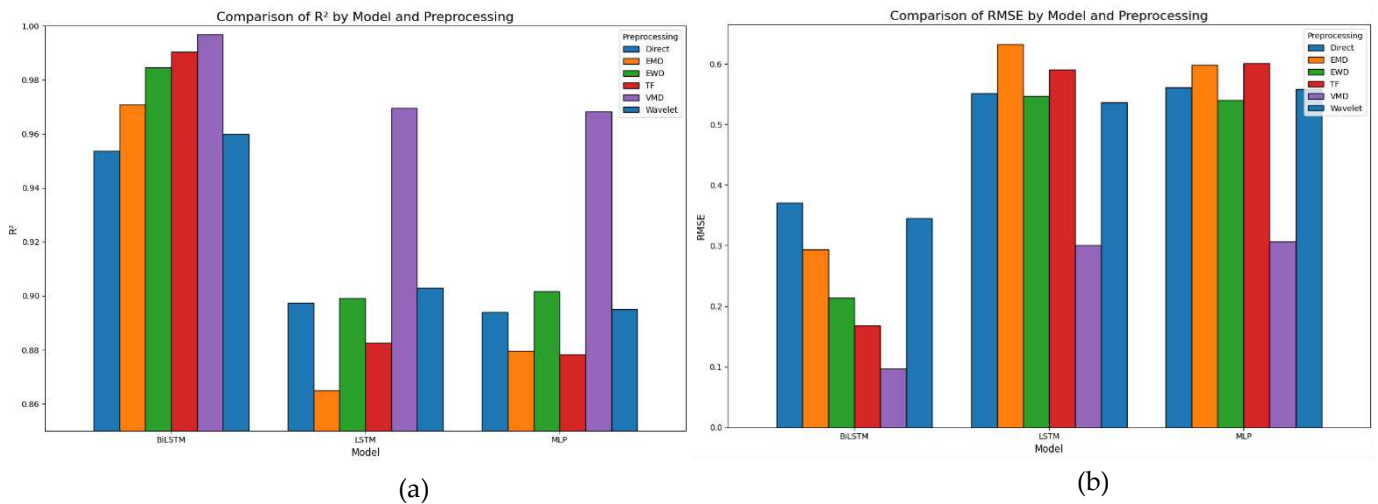
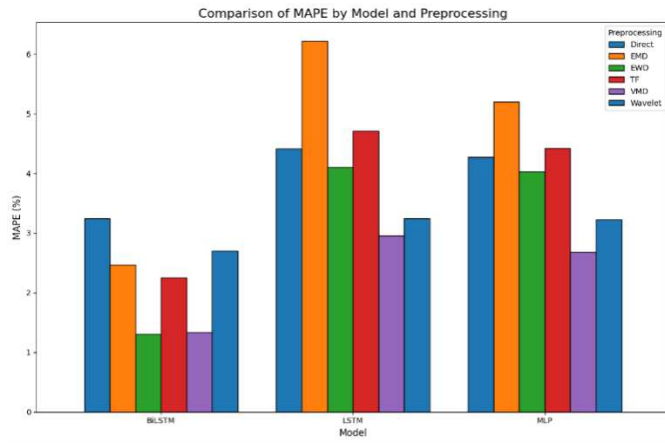


Figure 4. Evolution of the error metrics with the number of neurons for the Apple time series without preprocessing.





(c)

Figure 5. R^2 , RMSE and MAPE values for the Apple time series when preprocessing is applied.

A similar behavior is observed when predicting the Brent crude oil price series. Once again, VMD (Figures 6 and 7) preprocessing proves more effective across all models, producing very low error metrics (Tables 5 and 6 and Figure 7). For the non-preprocessed data, the MLP model achieves its best performance with a single neuron, while the LSTM and BiLSTM models perform best with 11 and 100 neurons, respectively (Figure 6). In all cases, preprocessing improves performance: LSTM achieves optimal results with just 5 neurons, and BiLSTM with 20, highlighting a significantly more efficient modeling process. The MLP model shows a consistent trend: when using 3 neurons after preprocessing, it continues providing slightly worse accuracy than BiLSTM, but clearly better than LSTM.

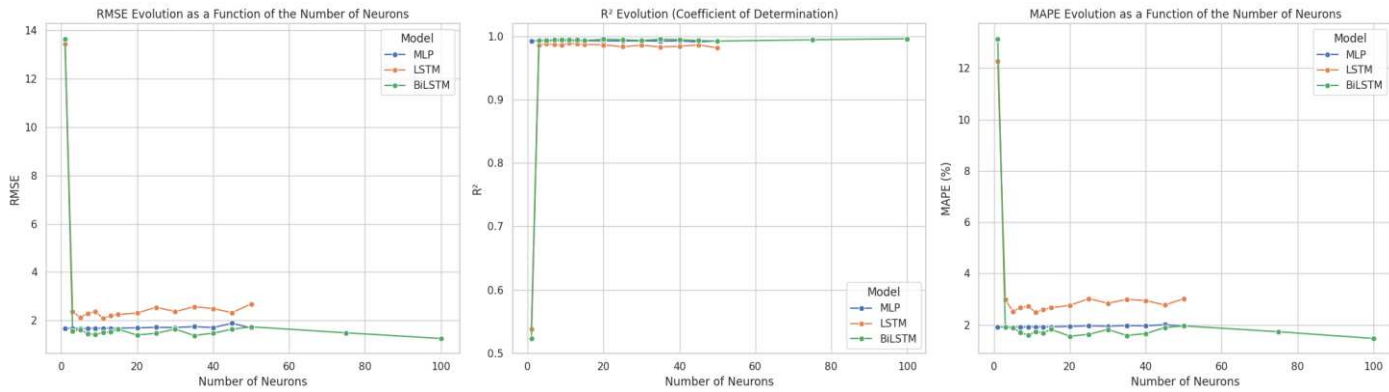


Figure 6. Evolution of the error metrics with the number of neurons for the Brent time series without preprocessing.

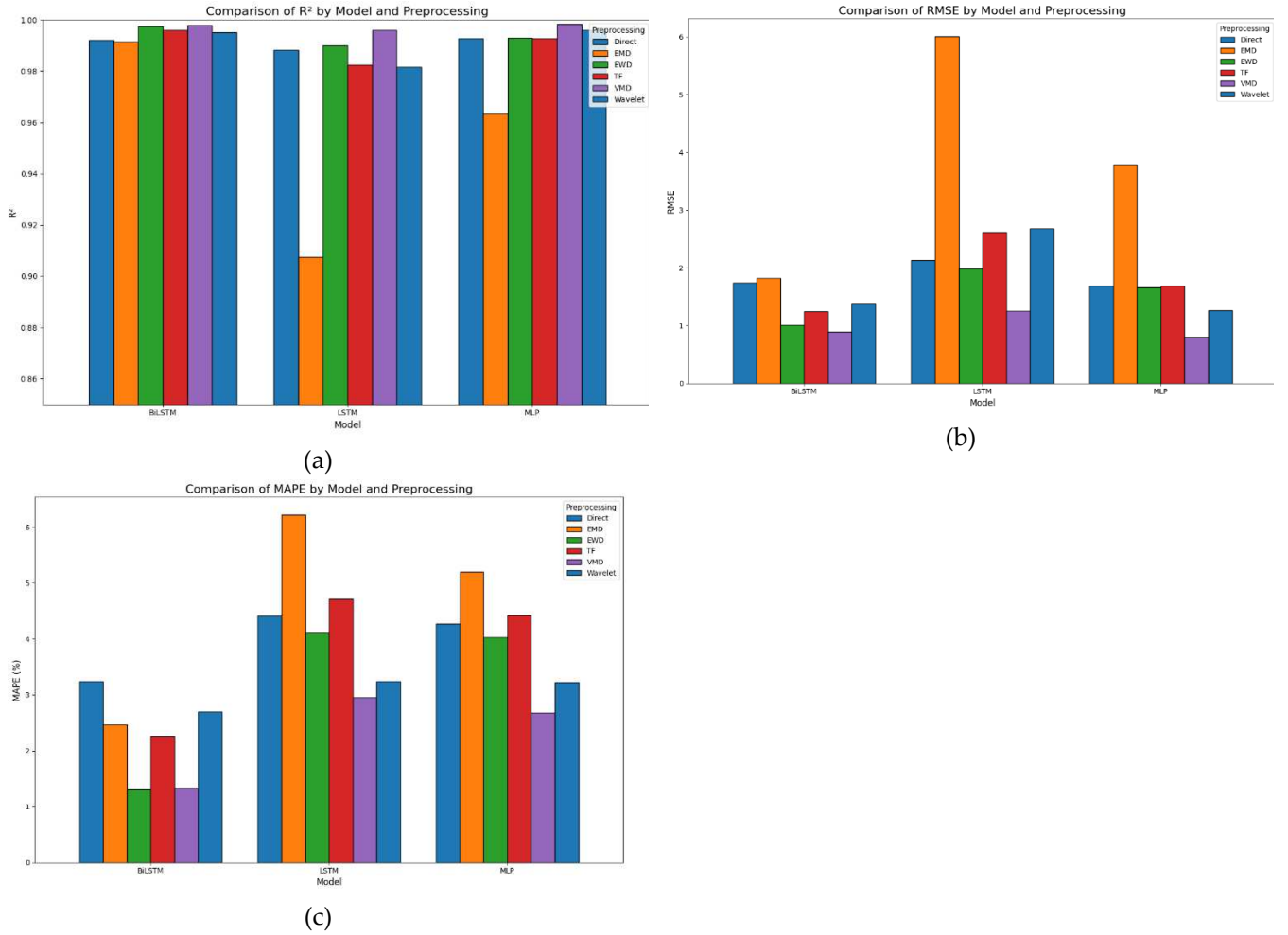


Figure 7. R^2 , RMSE and MAPE values for the Brent time series when preprocessing is applied.

Lastly, the natural gas time series follows a similar trajectory, with VMD preprocessing once again providing the lowest error metrics compared to other methods (Figures 8 and 9). Without preprocessing, LSTM requires 13 neurons to achieve optimal results, whereas only 5 neurons are needed after preprocessing. The MLP model performs best with 35 neurons on raw data but requires only 3 neurons after preprocessing to achieve significant improvements. The BiLSTM model reaches optimal performance with 200 neurons on the unprocessed data, but preprocessing reduces this requirement to just 50 neurons while further enhancing the metrics (Figure 8).

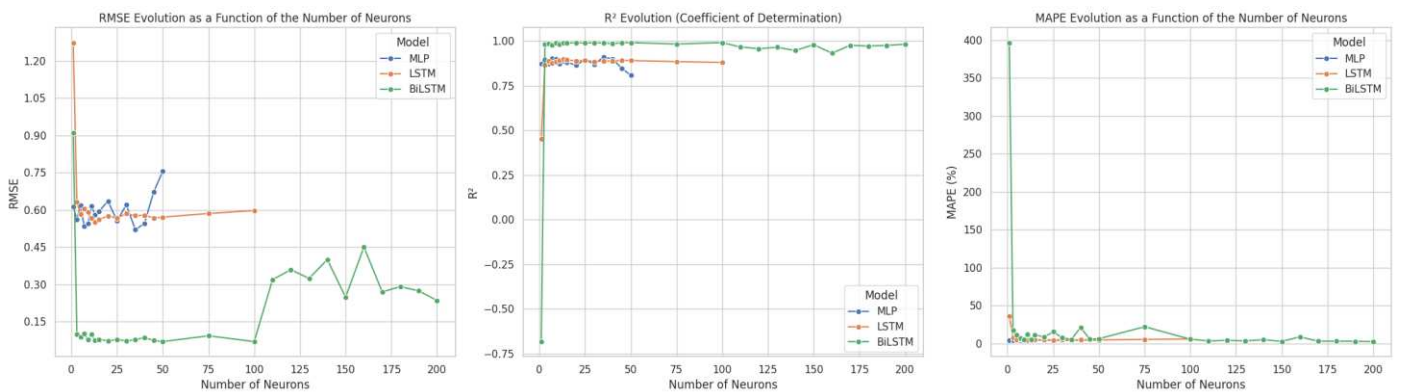


Figure 8. Evolution of the error metrics with the number of neurons for the Natural Gas time series without preprocessing.

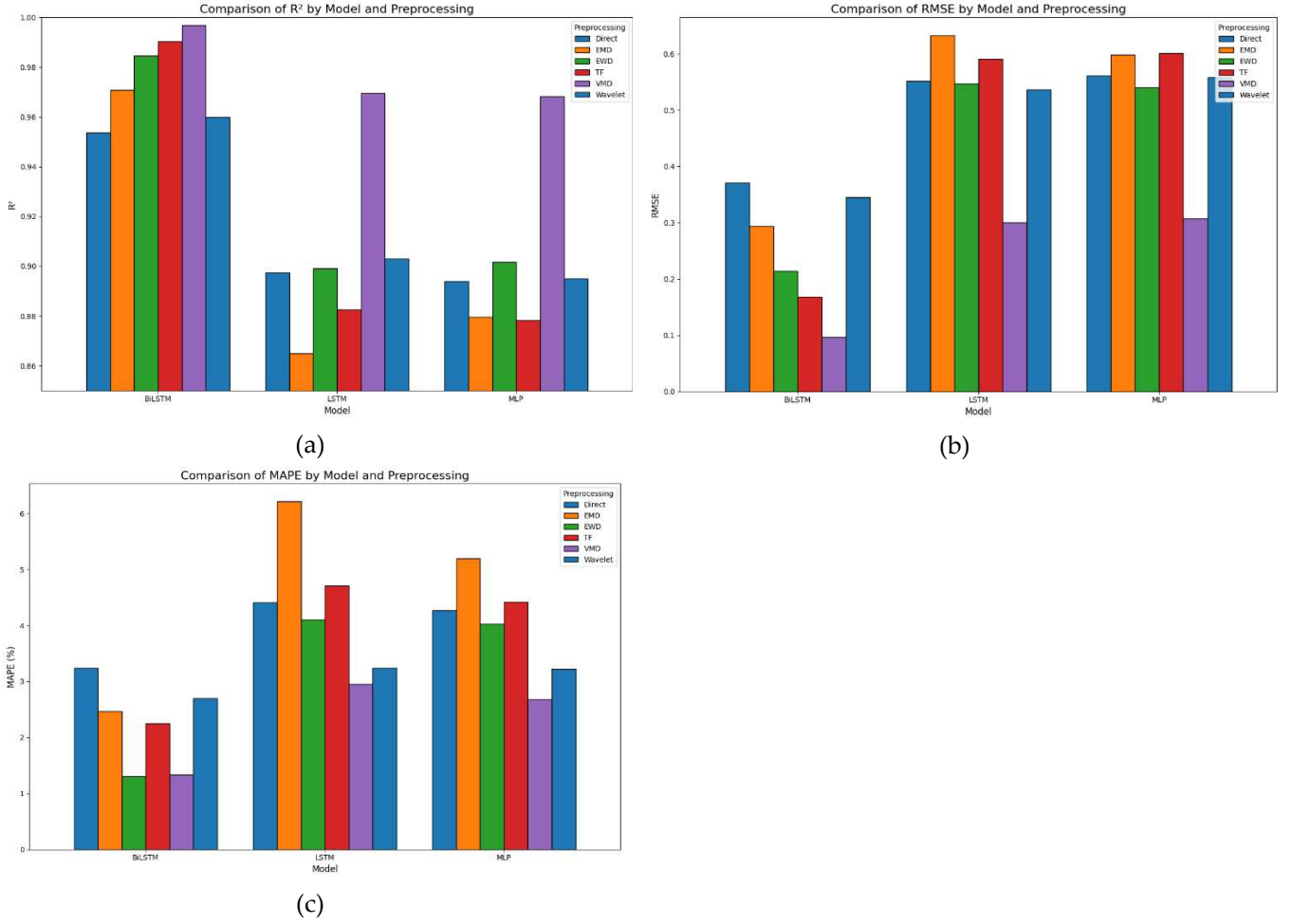


Figure 9. R^2 , RMSE and MAPE values for the Natural Gas time series when preprocessing is applied.

5. Discussion and limitations

Time series preprocessing is a topic of great interest in the literature, given its ability to achieve improvements in the performance of prediction models (Lazcano & Jaramillo-Morán, 2025). Several studies have pointed out that proper preprocessing prior to training neural networks improves prediction accuracy, as reflected in lower error metrics. However, the relationship between preprocessing, model complexity, and the optimal number of neurons remains an underexplored topic.

The results obtained in this study contribute to this discussion by showing how different architectures specifically MLP, LSTM, and BiLSTM respond to decomposition-based preprocessing, with VMD delivering the most consistent gains overall and EMD providing the best hybrid results for Apple. This effect is particularly evident in the Brent and natural Gas series, where VMD improves forecasting accuracy while reducing the number of neurons required by the recurrent models. It has been shown that LSTM and BiLSTM models, due to their internal memory mechanisms and gating structures, typically require a larger number of neurons to effectively capture temporal dependencies, especially when working with raw (unprocessed) series. However, after applying VMD, these models achieve higher accuracy with a significantly smaller number of neurons, which results in a considerable reduction in computational cost.

Some of the main limitations found in this study lie in the choice of the number of selected hyperparameters, on which performance closely depends. Although a systematic approach to neuron selection was followed in this work, variations in other parameters may have influenced the results and would require additional optimization in future research.

Second, the models evaluated in this study are purely data-driven and do not incorporate exogenous variables or structural constraints that could improve their robustness against economic shocks or external events. This limits their applicability in highly volatile or policy-sensitive contexts, where interpretability and adaptability are key factors.

Third, although reducing the number of neurons leads to greater computational efficiency, it is important to balance this with the risk of underfitting, particularly in highly complex time series. Further research is needed to establish a more robust methodology for determining the optimal number of neurons based on both the model architecture and the preprocessing strategy used.

Lastly, from the perspective of computational economics, these findings have direct implications for the design of forecasting pipelines in applied economic and financial settings. The substitution between signal preprocessing and architectural complexity quantified here suggests that practitioners can build accurate forecasting systems for nonstationary economic and financial series with substantially smaller neural networks than typically reported, which translates into shorter training times, lower memory and computational requirements, and improved suitability for deployment in operational economic decision-support environments.

6. Conclusions

The inherent complexity of economic time series demands not only robust predictive models but also efficient strategies that minimize computational costs without compromising accuracy. This study highlights the importance of aligning preprocessing techniques with forecasting tools to achieve such a balance. In most cases, the application of Variational Mode Decomposition has proven to be a simple yet effective method for enhancing the performance of neural network models, especially those designed to capture temporal dependencies, such as LSTM and BiLSTM.

The results obtained indicate that decomposition-based preprocessing, and especially VMD in most cases, leads to improved forecasting accuracy while reducing the number of neurons required to achieve competitive performance. This finding is especially relevant in scenarios where computational efficiency is critical, such as real-time forecasting or resource-constrained environments. Furthermore, the study demonstrates that even simpler neural architectures, such as MLP, benefit from the application of VMD, thereby confirming its versatility across different levels of model complexity. These results contribute to the computational economics literature by providing empirically grounded guidance on when signal preprocessing can substitute for architectural complexity in neural forecasting pipelines applied to economic and financial data.

Future research could explore the integration of this approach in multistep forecasting contexts, where maintaining accuracy across extended prediction horizons remains a major challenge. Adapting neural models to address these scenarios could substantially enhance their applicability to economic planning and policy-making processes, where long-term forecasts are often essential.

7. Statements and Declarations

Funding. The authors did not receive support from any organization for the submitted work.

Competing Interests. The authors have no relevant financial or non-financial interests to disclose.

Author Contributions. All authors contributed to the study conception and design. Methodology and software implementation were performed by Miguel A. Jaramillo-Morán. Data collection and curation were performed by Miguel A. Jaramillo-Morán and Ana Lazcano. Investigation and validation were performed by Ana Lazcano and Julio E. Sandubete. The first draft of the manuscript was written by Miguel A. Jaramillo-Morán, Ana Lazcano and Julio E. Sandubete, and all authors commented on and revised previous versions of the manuscript. All authors read and approved the final version.

Data Availability Statement. The financial time series used in this study (Apple stock prices, Brent crude oil prices and natural gas prices) are publicly available from Yahoo Finance. The MATLAB code used to generate the results is available from the corresponding author upon reasonable 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, 9464938. <https://doi.org/10.1155/jpas/9464938>
- Angamuthu Chinnathambi, R., Mukherjee, A., Campion, M., Salehfar, H., Hansen, T. M., Lin, J., et al. (2018). A multi-stage price forecasting model for day-ahead electricity markets. *Forecasting*, 1(1), 26–46.
- 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>
- Bak, G., & Bae, Y. (2020). Predicting the amount of electric power transaction using deep learning methods. *Energies*, 13(24), 6649. <https://doi.org/10.3390/en13246649>
- Bhardwaj, S., Chandrasekhar, E., Padiyar, P., & Gadre, V. M. (2020). A comparative study of wavelet-based ANN and classical techniques for geophysical time-series forecasting. *Computers & Geosciences*, 138, 104461. <https://doi.org/10.1016/j.cageo.2020.104461>
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.
- 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. (2019). Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14, 45–76. <https://doi.org/10.28945/4184>
- 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/10.1561/22000000016>
- Chen, S.-A., Li, C.-L., Yoder, N., Arik, S. O., & Pfister, T. (2023). TSMixer: An all-MLP architecture for time series forecasting. *arXiv*. <https://doi.org/10.48550/arXiv.2303.06053>
- Ciechulski, T., & Osowski, S. (2021). High precision LSTM model for short-time load forecasting in power systems. *Energies*, 14(11), 2983. <https://doi.org/10.3390/en14112983>
- Conrad, C., Rittler, D., & Rotfuß, W. (2012). Modeling and explaining the dynamics of European Union Allowance prices at high-frequency. *Energy Economics*, 34(1), 316–326. <https://doi.org/10.1016/j.eneco.2011.02.011>
- Cortés-Ibáñez, J. A., González, S., Valle-Alonso, J. J., Luengo, J., García, S., & Herrera, F. (2020). Preprocessing methodology for time series: An industrial world application case study. *Information Sciences*, 514, 385–401. <https://doi.org/10.1016/j.ins.2019.11.027>
- Di Piazza, A., Di Piazza, M. C., La Tona, G., & Luna, M. (2021). An artificial neural network-based forecasting model of energy-related time series for electrical grid management. *Mathematics and Computers in Simulation*, 184, 294–305. <https://doi.org/10.1016/j.matcom.2020.05.010>
- Dong, Z., & Zhou, Y. (2024). A novel hybrid model for financial forecasting based on CEEMDAN-SE and ARIMA-CNN-LSTM. *Mathematics*, 12(15), 2434. <https://doi.org/10.3390/math12162434>
- Dragomiretskiy, K., & Zosso, D. (2014). Variational mode decomposition. *IEEE Transactions on Signal Processing*, 62(3), 531–544. <https://doi.org/10.1109/TSP.2013.2288675>
- Dritsaki, M., & Dritsaki, C. (2020). Forecasting European Union CO2 emissions using autoregressive integrated moving average-autoregressive conditional heteroscedasticity models. *International Journal of Energy Economics and Policy*, 10(5), 411–423. <https://doi.org/10.32479/ijee.9186>
- Famili, A., Shen, W.-M., Weber, R., & Simoudis, E. (1997). Data preprocessing and intelligent data analysis. *Intelligent Data Analysis*, 1(1), 3–23. <https://doi.org/10.3233/IDA-1997-1102>
- 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.

-
- Fang, Z., Ma, X., Pan, H., Yang, G., & Arce, G. R. (2023). Movement forecasting of financial time series based on adaptive LSTM-BN network. *Expert Systems with Applications*, 213, 119207.
- Feng, Y., Yao, J., Li, Z., & Zheng, R. (2022). Uncertainty prediction of energy consumption in buildings under stochastic shading adjustment. *Energy*, 254, 124145. <https://doi.org/10.1016/j.energy.2022.124145>
- Fong, S., Lan, K., Sun, P., Mohammed, S., & Fiaidhi, J. (2013). A time-series pre-processing methodology for biosignal classification using statistical feature extraction. In *Proceedings of the 10th IASTED International Conference on Biomedical Engineering (Biomed'13)* (pp. 207–214).
- 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 wavelet 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>
- Gonzalez-Romera, E., Jaramillo-Morán, M. A., & Carmona-Fernandez, D. (2006). Monthly electric energy demand forecasting based on trend extraction. *IEEE Transactions on Power Systems*, 21(4), 1946–1953. <https://doi.org/10.1109/TPWRS.2006.883666>
- Guo, X., Zhao, Q., Zheng, D., Ning, Y., & Gao, Y. (2020). A short-term load forecasting model of multi-scale CNN-LSTM hybrid neural network considering the real-time electricity price. *Energy Reports*, 6, 1046–1053.
- 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>
- Hill, T., O'Connor, M., & Remus, W. (1996). Neural network models for time series forecasts. *Management Science*, 42(7), 1082–1092. <https://doi.org/10.1287/mnsc.42.7.1082>
- 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](https://doi.org/10.1016/0893-6080(89)90020-8)
- Huang, N. E., Wu, Z., & Long, S. R. (2008). Hilbert-Huang transform. *Scholarpedia*, 3(7), 2544. <https://doi.org/10.4249/scholarpedia.2544>
- Hwang, J., Suh, D., & Otto, M.-O. (2020). Forecasting electricity consumption in commercial buildings using a machine learning approach. *Energies*, 13(22), 5885. <https://doi.org/10.3390/en13225885>
- Jaramillo-Morán, M. A., Fernández-Martínez, D., Garcia-Garcia, A., & Carmona-Fernández, D. (2021). Improving artificial intelligence forecasting models performance with data preprocessing: European Union allowance prices case study. *Energies*, 14(23), 7845. <https://doi.org/10.3390/en14237845>
- Joo, T. W., & Kim, S. B. (2015). Time series forecasting based on wavelet filtering. *Expert Systems with Applications*, 42(8), 3868–3874. <https://doi.org/10.1016/j.eswa.2015.01.026>
- 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>
- Khashei, M., & Bijari, M. (2010). An artificial neural network (p,d,q) model for time series forecasting. *Expert Systems with Applications*, 37(1), 479–489. <https://doi.org/10.1016/j.eswa.2009.05.044>
- Khedhiri, S. (2022). Comparison of SARFIMA and LSTM methods to model and to forecast Canadian temperature. *Regional Statistics*, 12(2), 177–194. <https://doi.org/10.15196/RS120204>
- Kim, J., & Moon, N. (2019). BiLSTM model based on multivariate time series data in multiple field for forecasting trading area. *Journal of Ambient Intelligence and Humanized Computing*. <https://doi.org/10.1007/s12652-019-01398-9>

-
- Kotsiantis, S. B., Kanellopoulos, D., & Pintelas, P. E. (2006). Data preprocessing for supervised learning. *International Journal of Computer Science*, 1(2), 111–117.
- Kumar, B., Sunil, & Yadav, N. (2023). A novel hybrid model combining SARMA and LSTM for time series forecasting. *Applied Soft Computing*, 134, 110019. <https://doi.org/10.1016/j.asoc.2023.110019>
- Lazcano, A., & Jaramillo-Morán, M. A. (2025). Data preprocessing techniques and neural networks for trended time series forecasting. *Applied Soft Computing*, 174, 113063. <https://doi.org/10.1016/j.asoc.2025.113063>
- 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 r^2 , why not? Then what? *PLOS ONE*, 12(8), e0183250. <https://doi.org/10.1371/journal.pone.0183250>
- Lucas, A., Pegios, K., Kotsakis, E., & Clarke, D. (2020). Price forecasting for the balancing energy market using machine-learning regression. *Energies*, 13(20), 5420. <https://doi.org/10.3390/en13205420>
- Luo, S., & Tian, C. (2020). Financial high-frequency time series forecasting based on sub-step grid search long short-term memory network. *IEEE Access*, 8, 203183–203189. <https://doi.org/10.1109/ACCESS.2020.3037102>
- Madhusudhanan, K., Jawed, S., & Schmidt-Thieme, L. (2024). Hyperparameter tuning MLPs for probabilistic time series forecasting. *arXiv*. <https://doi.org/10.48550/arXiv.2403.04477>
- 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. <https://doi.org/10.1016/j.jefas.2016.07.002>
- 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 and Statistics*, 57(4), 547. <https://doi.org/10.1111/j.1468-0084.1995.tb00039.x>
- Pabuççu, H. (2017). Neural network data preprocessing: Is it necessary for time series forecasting? *Kilis 7 Aralık Üniversitesi İktisadi ve İdari Bilimler Fakültesi Akademik Araştırmalar ve Çalışmalar Dergisi*, 9(17). <https://doi.org/10.20990/kilisiibfakademik.329247>
- 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>
- Penedo, S. R. M., Netto, M. L., & Justo, J. F. (2019). Designing digital filter banks using wavelets. *EURASIP Journal on Advances in Signal Processing*, 2019, 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>
- 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>
- Shen, Y., Ma, Y., Deng, S., Huang, C.-J., & Kuo, P.-H. (2021). An ensemble model based on deep learning and data preprocessing for short-term electrical load forecasting. *Sustainability*, 13(4), 1694. <https://doi.org/10.3390/su13041694>
- Siarni-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>
- Sun, R. (2019). Optimization for deep learning: Theory and algorithms. *arXiv*. <https://doi.org/10.48550/arXiv.1912.08957>
- Takeda, H., Tamura, Y., & Sato, S. (2016). Using the ensemble Kalman filter for electricity load forecasting and analysis. *Energy*, 104, 184–198. <https://doi.org/10.1016/j.energy.2016.03.070>

-
- Voyant, C., Nivet, M. L., Paoli, C., Muselli, M., & Notton, G. (2015). Meteorological time series forecasting based on MLP modelling using heterogeneous transfer functions. *Journal of Physics: Conference Series*, 574, 012064. <https://doi.org/10.1088/1742-6596/574/1/012064>
- 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, Y., Wang, J., Zhao, G., & Dong, Y. (2012). Application of residual modification approach in seasonal ARIMA for electricity demand forecasting: A case study of China. *Energy Policy*, 48, 284–294. <https://doi.org/10.1016/j.enpol.2012.05.026>
- Wilson, G. T. (2016). Time series analysis: Forecasting and control, 5th edition, by George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel and Greta M. Ljung, 2015. *Journal of Time Series Analysis*, 37(5), 709–711. <https://doi.org/10.1111/jtsa.12194>
- Wu, J., Wang, J., Lu, H., Dong, Y., & Lu, X. (2013). Short term load forecasting technique based on the seasonal exponential adjustment method and the regression model. *Energy Conversion and Management*, 70, 1–9. <https://doi.org/10.1016/j.enconman.2013.02.010>
- Yang, D., Guo, J., Sun, S., Han, J., & Wang, S. (2022). An interval decomposition-ensemble approach with data-characteristic-driven reconstruction for short-term load forecasting. *Applied Energy*, 306, 117992. <https://doi.org/10.1016/j.apenergy.2021.117992>
- 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, 118197. <https://doi.org/10.1016/j.apenergy.2021.118197>
- Yemets, K., Izonin, I., & Mitoulis, S. A. (2025). Multi-family wavelet-based feature engineering method for short-term time series forecasting. *Scientific Reports*, 15, 39081. <https://doi.org/10.1038/s41598-025-26685-8>
- Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (n.d.). Dive into deep learning. <https://d2l.ai/>
- Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., & Zheng, S. (2022). Less is more: Fast multivariate time series forecasting with light sampling-oriented MLP structures. *arXiv*. <https://doi.org/10.48550/arXiv.2207.01186>
- Zhu, B., Han, D., Wang, P., Wu, Z., Zhang, T., & Wei, Y.-M. (2017). Forecasting carbon price using empirical mode decomposition and evolutionary least squares support vector regression. *Applied Energy*, 191, 521–530. <https://doi.org/10.1016/j.apenergy.2017.01.076>
- Zhu, B., Ye, S., Wang, P., He, K., Zhang, T., & Wei, Y.-M. (2018). A novel multiscale nonlinear ensemble learning paradigm for carbon price forecasting. *Energy Economics*, 70, 143–157. <https://doi.org/10.1016/j.eneco.2017.12.030>

Appendix A

Appendix A.1. Empirical Mode Decomposition

The Empirical Mode Decomposition (EMD) technique decomposes a time series into a set of intrinsic mode functions (IMF) and a final residue that should have a periodical behaviour that should facilitate their predictions.

Each IMF must meet two criteria to be considered as properly characterizing the periodical components of the series:

- One is the maximum value resulting from the difference between the total number of maximums and minimums obtained.
- The average of the envelopes at each of the signal points, both upper and lower, through the resulting interpolation of maxima and minima, must equal zero.

From the screening, consisting of an iterative algorithm, the IMFs will be obtained as a result. This is thanks to the decomposition of the time series $x(t)$ until all the IMFs are obtained (Joo & Kim, 2015):

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

$$m(t) = \frac{u(t) + l(t)}{2}. \quad (\text{A1})$$

3. A detailed component is obtained by subtracting $m(t)$ from $x(t)$ and:

$$h(t) = x(t) - m(t). \quad (\text{A2})$$

4. If $h(t)$ meets the criteria for an IMF, it is chosen as the first IMF ($h(t) = c_1(t)$). Then, the algorithm proceeds to the next step. If either condition is not met, $h(t)$ is assumed to be a new time series and it will be processed following steps 1 to 3 until an IMF is obtained.
5. A new series is then defined by subtracting $c_1(t)$ from $x(t)$, which will not have the frequencies associated with $c_1(t)$:

$$f_n(t) = x(t) - c_1(t). \quad (\text{A3})$$

6. These steps are iterated through until no new IMFs are obtained.

The residual is obtained at the end of this process by subtracting the last IMF from the original time series, i.e., $h(t)=r(t)$.

To avoid infinite iteration, the algorithm employs two stopping conditions. The first is assessed at step 4 and is fulfilled either when the candidate IMF meets both IMF conditions or when its variance falls below a set threshold. The second condition applies at the final stage and is satisfied if the residual becomes a constant value, exhibits a steady linear trend, or presents only a single extremum.

The original time series can be reconstructed as a product of the sum of the IMFs and the residual:

$$x(t) = \sum_i c_i(t) + r(t). \quad (\text{A4})$$

Once this process is complete, it is possible to make predictions for each of the IMFs independently, and then add the individual predictions together to obtain the prediction for the primary time series.

Appendix A.2. Variational Mode Decomposition

The Variational Mode Decomposition (VMD) algorithm shares characteristics with the EMD process, since it performs the decomposition of a signal $x(t)$ into a set of subsignals (modes), $u_k(t)$, with specific bandwidths which must be mostly compact around a center frequency ω_k :

$$x(t) = \sum_{i=1}^N u_i(t) = \sum_{i=1}^N A_i(t) \cos(\phi_i(t)) \quad (\text{A5})$$

In this expression $A_i(t)$ and $\phi_i(t)$ represent the time dependent amplitude and phase. The central frequencies of each mode are:

$$\omega_i(t) = \frac{\partial \phi_i(t)}{\partial t}. \quad (\text{A6})$$

In this way each mode $u_i(t)$ will be associated with a central frequency $\omega_i(t)$.

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

$$\min_{\{u_i\}, \{\omega_i\}} \left\{ \sum_i \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \right\|_2^2 \right\}. \quad (\text{A7})$$

In this expression $\delta(t)$ represents the Dirac distribution function and $*$ the convolution operator. Here the squared L^2 -norm has been used.

This constrained variational problem can be converted into an unconstrained optimization problem by defining an augmented Lagrangian function that includes a quadratic penalty and Lagrange multipliers:

$$\begin{aligned} \mathcal{L}(\{u_i\}, \{\omega_i\}, \lambda) = & \alpha \sum_i \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \right\|_2^2 + \left\| x(t) - \sum_i u_i(t) \right\|_2^2 \\ & + \langle \lambda(t), f(t) - \sum_i u_i(t) \rangle. \end{aligned} \quad (\text{A8})$$

By identifying saddle points in the augmented Lagrangian function, the initial minimization problem can be solved. The technique used is iterative optimization, the Alternating Direction Multiplier Method (ADMM) (Paoli et al., 2010). According to this algorithm, the modes and their corresponding central frequencies can be updated using the iterative expressions:

$$u_i^{n+1} = \arg \min_{u_i} \left\{ \alpha \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \right\|_2^2 + \left\| x(t) - \sum_i u_i(t) + \frac{\lambda(t)}{2} \right\|_2^2 \right\} \quad (\text{A9})$$

$$\omega_i^{n+1} = \arg \min_{\omega_i} \left\{ \sum_i \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_i(t) \right] e^{-j\omega_i t} \right\|_2^2 \right\}. \quad (\text{A10})$$

This problem is solved in the spectral domain providing solutions for each mode and frequency:

$$\hat{u}_i^{n+1}(\omega) = \frac{\hat{x}(\omega) - \sum_{k \neq i} \hat{u}_k(\omega) + \frac{\hat{\lambda}(\omega)}{2}}{1 + 2\alpha(\omega + \omega_k)^2}, \quad (\text{A11})$$

$$\omega_i^{n+1} = \frac{\int_0^\infty \omega |\hat{u}_i(\omega)|^2 d\omega}{\int_0^\infty |\hat{u}_i(\omega)|^2 d\omega}, \quad (\text{A12})$$

$$\hat{\lambda}(\omega)^{n+1} = \hat{\lambda}(\omega)^n + \tau \left(\hat{x}(\omega) - \sum_i \hat{u}_i^{n+1}(\omega) \right), \quad (\text{A13})$$

where $\hat{x}(\omega)$, $\hat{u}_i(\omega)$ and $\hat{\lambda}(\omega)$ are the Fourier transforms of $x(t)$, $u_i(t)$ and $\lambda(t)$. τ is a parameter that must be adjusted to ensure the iterative process converges.

Appendix A.3. Empirical Wavelet Transform

The Empirical Wavelet Transform (EWT) performs an adaptive multispectral decomposition that splits a certain signal into an ensemble of sub signals closely related to inner frequencies of the original signal. In this way it performs a decomposition like that provided by EMD and VMD. It differs from these two methods in that it works on the Fourier spectrum of the signal to identify frequency peaks and then builds a filter bank made up of wavelets to perform the decomposition process. The set of sub signals provided by this filter bank define a compact support Fourier spectrum.

To build that filter bank, the Fourier spectrum is divided into N consecutive segments, each of which is limited to the interval $\omega \in [0, \pi]$. This ensures that the Shannon criterium is met. Each segment will be associated with a specific central frequency ω_i and will have a width $2\tau_i$. For simplicity it is assumed that $\tau_i = \gamma\omega_i$ with $0 < \gamma < 1$.

The filter bank will be defined in the Fourier domain by means of a set of empirical scaling functions $\hat{\phi}_n(\omega)$ (low-pass filter) and empirical wavelets $\hat{\psi}_n(\omega)$ (band-pass filters):

$$\hat{\phi}_n(\omega) = \begin{cases} 1 & \text{if } |\omega| \leq (1 - \gamma)\omega_n \\ \cos\left[\frac{\pi}{2} \beta\left(\frac{1}{2\gamma\omega_n}(|\omega| - (1 - \gamma)\omega_n)\right)\right] & \text{if } (1 - \gamma)\omega_n \leq |\omega| \leq (1 + \gamma)\omega_n \\ 0 & \text{otherwise} \end{cases} \quad (\text{A14})$$

$$\hat{\psi}_n(\omega) = \begin{cases} 1 & \text{if } (1 + \gamma)\omega_n \leq |\omega| \leq (1 - \gamma)\omega_{n+1} \\ \cos\left[\frac{\pi}{2} \beta\left(\frac{1}{2\gamma\omega_n}(|\omega| - (1 - \gamma)\omega_{n+1})\right)\right] & \text{if } (1 - \gamma)\omega_{n+1} + 1 \leq |\omega| \leq (1 + \gamma)\omega_{n+1} \\ \sin\left[\frac{\pi}{2} \beta\left(\frac{1}{2\gamma\omega_n}(|\omega| - (1 - \gamma)\omega_n)\right)\right] & \text{if } (1 - \gamma)\omega_n \leq |\omega| \leq (1 + \gamma)\omega_n \\ 0 & \text{otherwise} \end{cases} \quad (\text{A15})$$

$\beta(x)$ is commonly defined as:

$$\beta(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x^4(35 - 84x + 70x^2 - 20x^3) & \text{if } 0 < x < 1 \\ 1 & \text{if } x \geq 1 \end{cases} \quad (\text{A16})$$

In order to properly define the N segments in the Fourier domain, frequency peaks should be identified and associated to each segment. If M peaks are identified two cases can appear:

- $M \geq N$: only the first $N-1$ peaks will be considered (a first segment will be associated to the $\omega = 0$ frequency).
- $M < N$: the signal has less peaks than expected and only M segments will be defined.

To guarantee that the empirical scaling functions and the empirical wavelets represent a tight frame of $L^2(R)$, γ must accomplish with:

$$\gamma < \min_n \left(\frac{\omega_{n+1} - \omega_n}{\omega_{n+1} + \omega_n} \right). \quad (\text{A17})$$

Now the EWT can be defined to provide a set of empirical wavelets that can provide a decomposition of a signal $f(t)$ into a set of empirical modes:

$$x(t) = x_0(t) + \sum_{i=1}^{N-1} x_i(t). \quad (\text{A18})$$

Each mode $x_n(t)$ will be defined as:

$$x_0(t) = W_f^\varepsilon(0, t) * \varphi_1(t), \quad (\text{A19})$$

$$x_i(t) = W_f^\varepsilon(i, t) * \psi_i(t), \quad (\text{A20})$$

where $\varphi_1(t)$ and $\psi_i(t)$ represents wavelets in the time domain corresponding to $\hat{\varphi}_n(\omega)$ and $\hat{\psi}_n(\omega)$ in the Fourier domain, and operator $*$ stands for convolution.

On the other hand, $W_f^\varepsilon(0, t)$ represents the approximation coefficients (that associated to the lowest frequency) while $W_f^\varepsilon(i, t)$ represent the detail ones (those associated to the higher frequencies). They are obtained as the inner products:

$$W_f^\varepsilon(0, t) = \langle x(t), \varphi_1(t) \rangle = \int x(\tau) \bar{\varphi}_1(\tau - t) d\tau = \mathcal{F}^{-1} \{ \hat{x}(\omega) \overline{\hat{\varphi}_1(\omega)} \}, \quad (\text{A21})$$

$$W_f^\varepsilon(i, t) = \langle x(t), \psi_i(t) \rangle = \int x(\tau) \bar{\psi}_i(\tau - t) d\tau = \mathcal{F}^{-1} \{ \hat{x}(\omega) \overline{\hat{\psi}_i(\omega)} \}, \quad (\text{A22})$$

where the bar over a variable means the complex conjugate of this variable. The equivalence between the Fourier and time domains yields from the convolution theorem.

Since the EWT is performed by defining a segmentation in the frequency space, a signal can be processed in the frequency domain. This means that, if the coefficients $W_f^\varepsilon(i, t)$ are adjusted appropriately, the wavelet transform can behave like a filter. This characteristic makes wavelets suitable for filtering applications (Yemets et al., 2025). In this way it can be used to provide a trend-fluctuations decomposition, like that previously described, by removing the higher frequencies, which are usually associated with noise due to their lower significance, and retaining only the lower frequencies, which define the trend. Thus, the representation in the transformed space can be adjusted to create a noise rejection filter. Through a three-step process, it is possible to eliminate noise, referring to the highest frequencies:

1. The wavelet coefficients $W_f^\varepsilon(i, t)$ obtained are the result of transforming the signal $x(t)$ into a wavelet domain.
2. Next, thresholding is applied to remove those coefficients with values lower than a predefined threshold (the less significant wavelet components are removed).
3. The preserved coefficients allow the series to be reconstructed, approximating the original but without the noise.

As a result, a denoised time series is obtained. By subtracting this from the original series, the noise component is isolated:

$$N(t) = x(t) - X_{DN}(t), \quad (\text{A23})$$

where $X_{DN}(t)$ represents the denoised series and $N(t)$ the noise component. The original time series can be expressed as the sum of these two components:

$$x(t) = X_{DN}(t) + N(t). \quad (\text{A24})$$