Advances in the Theory of Nonlinear Analysis and its Applications 7 (2023) No. 4, 35-46.
https://doi.org/10.17762/atnaa.v7.i4.280
Available online at www.https://atnaea.org/
Research Article

Advances in the Theory of Nonlinear Analysis and its Applications

ISSN: 2587-2648 Peer-Reviewed Scientific Journal

Seasonal Decomposition and Trend using Hybrid STL- FNN with Application

Zena A. Sultan[a], Nihad S. Khalaf Aljboori[a]

[a] *Department of Mathematics, College of Education for Women, Tikrit University, Tikrit, Iraq.*

Abstract

Most researchers in the field of time series follow one approach, which is to rely on time series models in general. Several models have emerged in engineering, economics, and health applications. These models are known as neural networks, which imitate human nerve cells. This study suggests a hybrid algorithm called STL-FNN. It combines two methods - seasonal trend decomposition STL and feed-forward neural network (FNN). The STL method analyzes the original data into three subseries: seasonality, trend, and residual. To predict each of the three sub-series, the FNN neural network uses the seasonal component and separates them. Finally, all predicted outputs are combined as a total time series product. The results indicate that the STL-FNN can check the performance of the hybrid model. It uses the relative absolute error (MAE) criterion. We analyzed actual data to test the hybrid model's ability to predict. The model used the average monthly spending of foreign visitors in the United Kingdom from January 1986 to February 2020.

Keywords: Decomposition; STL model; FNN; Smoother Loess; STL-FNN structure.
2010 MSC: 60G10

1. Introduction

Time series data can have various patterns. To improve understanding, divide the data into separate components that represent patterns. During our study, we'll talk about three types of time series patterns: trend, seasonality, and cycles. When we break down a time series, we usually put trend and cycle together into one component (sometimes called trend for simplicity). A time series has three parts: trend cycle, seasonal component, and remainder component. In this study, we will look at some common methods for extracting components from time series. People often use this method to better understand time series and make forecasts more accurate.

Email address: nihad.shreef16@tu.edu.iq (Nihad S. Khalaf Aljboori)

Received September 20, 2023; Accepted: November 16, 2023; Online: December 13, 2023.

STL is a strong method for breaking down time series data in economics and environmental studies. The STL method breaks down time series into trend, seasonal, and residual parts using regression models. You can use STL with any data set. But you'll only get useful results if the data has a repeating time pattern. For example, air quality goes down in warmer months or online shopping goes up in the last quarter of each year. The pattern is displayed in the STL results as the seasonal component. The STL algorithm smooths the time series using LOESS in two loops. The inner loop makes the seasonal trend smoother. The outer loop lessens the impact of outliers. During the inner loop, the seasonal component is calculated first and removed to calculate the trend component. To find the remainder, subtract the seasonal and trend components from the time series.

The STL model was first introduced in 1990 by Cleveland. Its purpose is to filter the analysis of time series into three components, which are trend, seasonality, and residual. This model has a simplicity of structure as well as allowing quadrant calculation even for very long series [1], in 2000, Stroud and Xirouchakis presented the study "STL and its Extension" in which three approaches to improving the free-form manufacturing process were discussed:

1. The use of approximate control parameters;
2. The development and use of extensions to the Standard Transform Language (STL);
3. Manipulation methods With precise geometric models at the rapid prototyping (RP) level [2].

As for Szilvśi and others in 2003, they presented a study entitled "STL file analysis." Physical objects are produced layer by layer, each layer being a 2D cross-section of the 3D mesh by performing STL decomposition [3], in 2011 Theodosiou presented the study "Forecasting monthly and quarterly time series using STL analysis". This study is a re-examination of the benefits and limitations of decomposition and combination techniques in the field of forecasting, and also a contribution to this field, offering a new method of forecasting [4], in 2013 Donzé presented a study titled "Efficient Robust Court Monitoring for STL" that presents an efficient algorithm for calculating the degree of robustness to which a segmental continuous signal meets or violates the STL formula [5]. Kelley followed up in the same year with a study titled "STL: Guiding the 21st Century" which was found to directly address specific technology literacy standards for STL [6], in 2017 Van Eijnatten presented the study "Effect of CT parameters on STL model accuracy". The objectives of this study were to evaluate the image quality and accuracy of STL models acquired using different CT scanners and acquisition parameters [7], and for Yin and others in 2020, they presented the study "STL-ATTLSTM: Forecasting vegetable prices using STL and LSTM based on attention mechanism" in which the STL model was linked with the short-term neural network LSTM [8]. In the same year, Ničković and Tomoya presented the study "RTAMT: Online STL Durability Monitors" [9]. Varnai and Dimos then presented a study titled "On Robustness Measures for Learning STL Tasks" in which they examined existing and potential robustness measures specifically from the perspective of how these might help learning algorithms. They have shown that different desirable properties constrain the shape of potential measures, introducing a new measure based on the results [10], and in 2020, Chen and Shixiong presented the study "Short-Term Metro Passenger Forecasting with Seasonal Decomposition and Trend Using Loess and LSTM Neural Networks" [11]. In 1948, Alan Turing proposed the idea of a neural network in his book "Intelligent Machines". He referred to them as "unstructured machines of type B" [12]. The benefit of artificial neural network models lies in the fact that they can be used to infer a function from the input and use it. Neural networks can learn input representations without supervision. These representations capture important properties of the input distribution. Deep learning algorithms can also implicitly learn a distribution function. Data learning in neural networks is helpful when the data and tasks are too complex to design by hand. Nonlinear networks are used in many fields [13]. The topics covered are system identification, control, game playing, decision making, pattern recognition, and sequence recognition. They are also used in medicine, finance, data analysis, visuals, and spam detection [14].

Since neural networks were specifically introduced in 1943 by McCulloch and Walter, who introduced the concept of neural networks, by design, a neural network consists of a set of simple neurons interconnected with each other and associated with training weights [15]. From studies, for example: The most important

question of which network size is most appropriate for a given problem was posed by Pepys and Michael in a 1994 study titled "Feedforward Neural Networks" [16], in 2016, Eldan and Ohad presented the study "Depth Power of Feedback Neural Networks." They showed that there is a simple (almost radial) function, which can be expressed by small neural networks with three layers [17], Lui and William in 2019 also presented a numerical methodology for a specific problem titled "Building reduced-order models of fluid flows using deep feedforward neural networks" [18], Ozanich et al. studied "Feedforward Neural Network for Direction of Arrival Estimation" and their goal is to develop a nonlinear feed-forward deep neural network (FNN) for direction of arrival (DOA) estimation in 2020 [19], Chen presented a 2022 study titled "Atrial Fibrillation Detection Using Feedforward Neural Network" that used a feed-forward neural network [20], in 2023, Chen et al presented the study "Residual Strength Prediction of Corroded Pipelines Using Multilayer Perceptron and Modified Feed-Forward Neural Network." In this study, an artificial neural network (ANN) was used to predict the residual strength of corroded pipelines. Due to insufficient training data from previous experiments and severe iterations that were necessary to ensure accurate outcome prediction [21]. In the same year, Konar et al. presented a "classical and quantum spiked hybrid shallow neural network for noise-robust image classification" in which feed-forward neural networks were used [22].

2. Time series decomposition

Time series data can show a variety of patterns, and it is often useful to break a time series into several components, each representing a basic pattern category. There are several common methods of decomposition: Classical decomposition, X11 decomposition, Seasonal Extraction in ARIMA Time Series decomposition, Seasonal-Trend decomposition using Regression (STR), and Seasonal and Trend decomposition using Loess (STL). This is often done to help improve understanding of time series, but it can also be used to improve forecast accuracy. To extract these components from a time series, but what we are going to focus on is a specific type, which is STL. This is often done to help improve understanding of time series, but it can also be used to improve forecast accuracy.

If we assume additive decomposition, we can write the time series by decomposition in the form:

$$y_t = S_t + T_t + R_t \tag{1}$$

So that S_t, T_t and R_t represent both the seasonality component, the cyclical trend, and the residual, respectively, for all cycles of t. Instead, the time series will be written by multiplicative decomposition as:

$$y_t = S_t \times T_t \times R_t \tag{2}$$

If the seasonal fluctuations stay the same or the variance is close to the trend cycle, more decomposition is needed. If the pattern changes with the seasons or if the variation is related to the overall level of the data, then multiplicative analysis is better. Multiplicative analyses are common in economic time series. Instead of breaking it down using multiplication, you can change the data to make it look the same over time. Then, you can use additive decomposition. We can transform this by taking the logarithm of both sides. This is the same as using multiplicative decomposition because. $y_t = S_t \times T_t \times R_t$ Rewards $\log y_t = \log S_t + \log T_t + \log R_t$ [23].

3. STL decomposition

Seasonal and Trend decomposition using Loess (STL) is a versatile and robust method for time series analysis. The STL method was developed by Cleveland in 1990. Eq.(1) represents the STL equation.

STL has a simple design in which the series consists of the seasonal and trend parts and the remainder based on the Smoother Loess, which allows analysis of the properties of the parts as well as quick calculation even if the time series is very long and has large amounts of trend and seasonal smoothing. STL has many advantages and design goals compared to other decomposition methods, which can be summarized as:

1. STL will handle any type of seasonality, because the design is simple and straightforward to use
2. The seasonal component is allowed to change over time for flexibility in determining amounts of variation, and the rate of change can be controlled by the user.
3. The user can also control the smoothness of the direction cycle.
4. Limit the number of observations for each cycle of the seasonal component to an integer greater than one.
5. It can be robust to outliers (i.e. the user can specify a strongly decomposed value), and missing values, so that unusual incidental observations do not affect estimates of the trend cycle and seasonal components. However, they will affect the remaining component.
6. Ease of implementation using computer programs.

STL design and parameter selection in practice depend on understanding which part of the time series variance becomes a seasonal component and which part becomes a trend component. This understanding comes from eigenvalue and frequency response analyses. On the other hand, it has some disadvantages. In particular, it does not handle calendar change automatically, and only provides facilities for additive decomposition.

To get a multiplicative decomposition, start by finding the logarithm of the data. Then transform the components again. You can find decompositions between the additive and the multiplier by using the Box-Cox transformation of data. The value of corresponds to double decomposition while corresponds to additive decomposition [23, 24].

4. Feed-forward neural networks using the back-error propagation algorithm

The propagation of signals entering the network is always forward if the interconnection lines run in one direction from the input layer to the output layer. Therefore, the signal coming out of each neuron depends only on the input signal.

In our study, we will use a specific type of FFNN when using the backpropagation algorithm for error, called it for short (BPA), as it is one of the most widespread algorithms and aims to reduce the square of the total error when training and thus obtain the optimal weights that can be relied upon to obtain predictions for new data. You have not undergone training or education.

The stages of training neural networks using BPA can be summarized by:

- Feed forward calculation of training samples.
- Back propagation to (output layers and hidden layers).
- And adjust the network weights.

The steps for training neural networks using BPA can be summarized as follows:

1. Initialize the initial values of the weights and choose the training pair from the training set.
2. The forward propagation stage of the error, as this stage is divided into several sections:
   (a) Each neuron in the input layer receives its own input signal $y_j, j = 1, 2, \ldots, m$ and then sends it to the set of neurons in the hidden layer.
   (b) Each neuron in the hidden layer $\xi_i, i = 1, 2, \ldots, p$ collects its values and weighted input signals as in the equation:

$$\xi_i = \chi_{0i} + \sum_{j=1}^{m} y_j \chi_{ji} \tag{3}$$

   Where χ_{ji} are the weights of the input layers of the hidden layers. We apply the activation function with the equation:

$$\hbar_i = f\left(\xi_i - \vartheta_i\right) \tag{4}$$

Where ξ_i represents the hidden layer unit of index i, $\hbar_i$ represents the output of this (activated) unit, while ϑ is the threshold (a non-linear function known as the activation function through which the outputs of each neuron are processed).
To search for their output values, all activation values are sent to the neurons of the output layer.

(c) Each neuron in the hidden layer combines its values and weighted input signals as follows:

$$\varphi_k = w_{0k} + \sum_{i=1}^{p} \hbar_i w_{ik},\ k = 1, 2, \ldots, n \tag{5}$$

Then apply the activation function to calculate the output for each neuron in the output layer as follows:

$$x_k = f\left(\varphi_k - \vartheta_k\right) \tag{6}$$

Where φ_k represents the output layer unit of indicator k, while x_k represents the output of this unit (activated).

3. The backpropagation stage of the error is also divided into several sections:

(a) Calculate the error of the output neuron through the relation:

$$E_k = t_k - x_k \tag{7}$$

Where t_k represents the true value of the neuron and x_k represents the output value of the neuron. Then compare the neural network outputs with the real values to estimate the error through the model:

$$\delta_k = \left(t_k - x_k\right).\ f'\left(\varphi_k - \vartheta_k\right) \tag{8}$$

As follows, we calculate the amount of change in the size of the error according to the equation:

$$\Delta w_{ik} = \gamma.\delta_k.\hbar_j \tag{9}$$

Where δ_k is the error correction factor for adjusting the weight w_{ik}, while γ represents the learning unit and is used to accumulate the weight at each step of training.
Then calculate the bias correction term used to subsequently update the weight w_{0k}, where w_{0k} is the bias on the unit of the respective output layer.

$$\Delta w_{0k} = \gamma.\delta_k$$

(b) Each neuron in the hidden layer collects weighted input signals $\delta_k as$ in the formula:

$$\Delta_k = \sum_{k=1}^{m} \delta_k w_{ik}$$

Then we calculate $\delta_k = \Delta_k.f'\left(\varphi_k - \vartheta_k\right)$, in order to calculate the change in the size of the error through the equation:

$$\Delta\chi_{ji} = \gamma.\delta_i.y_j$$

Where δ_i is the weight adjustment error correction factor χ_{ji}.

(c) Then calculate the bias correction term used to update the weight χ_{0i} later, where χ_{0i} is the bias on the hidden layer unit:

$$\Delta\chi_{0i} = \gamma.\delta_i$$

4. The stage of updating weights and biases. This stage includes two parts:

(a) The weights and biases of each neuron in the hidden layer and the output layer are updated by the following equations, respectively:

$$\chi_{ji}\left(N\right) = \chi_{ji}\left(O\right) + \Delta\chi_{ji},\ j = 1, 2, \ldots, m \tag{10}$$

$$w_{ik}\left(N\right) = w_{ik}\left(O\right) + \Delta w_{ik},\ i = 1, 2, \ldots, p \tag{11}$$

The activation function is then applied to estimate the hidden layer neurons.

(b) The neural network continues to update the weights until the optimal weights are obtained, and then the desired output is obtained (stop state testing) [25, 26, 27].

5. Structure of the STL-FFNN model

The decomposition model is composed by STL, including an inner loop and an outer loop. The inner ring is nested inside the outer ring. The main steps of the inner loop are seasonal smoothing and trend smoothing, including six steps for this decomposition, which are (removal, obtaining a new series by subtracting the trend values from the original values, smoothing the sub-series of the cycle. Each sub-cycle obtained from the removal step is undone by Loess, low Pass Filtering The filtering involves three steps, the first step is a moving average of length. The next step is also a moving average of length. The last step is a moving average of length 3. After that, Loess is applied to the low-pass filtering results. Removal. Get the seasonal series. For an additional step, Cancel.

Seasonality: We obtain a non-seasonal series, and finally, trend smoothing (the trend series is obtained for an additional step after applying Loess to the non-seasonal series).

STL-FFNN is a hybrid algorithm that combines the STL model with the FFNN neural network to improve prediction performance for the time series studied. The specific steps are as follows:

1. Enter the values of the time series to be analysed.
2. Set the decomposition period and use the STL model to decompose the original series into three subseries (seasonality, trend, and residual).
3. The data is divided into a training set and a test set, and the time step and prediction step are set before training starts. The three subsequences obtained in (2) are trained using the same parameters of the FFNN neural network, and their test sets are predicted separately.
4. Add the prediction results for the sub-series as the prediction results for the original data.
5. Collect the sub-prediction results to obtain the original series using Eq.(1) if the decomposition used is the additive decomposition, or Eq.(2) if the decomposition used is the multiplicative decomposition.
6. Then we calculate R^2, the mean absolute error (MAE) and the root mean square error (RMSE).

$$R^2 = 1 - \sum_j (\acute{y_j} - y_j)^2 / \sum_j (\bar{y}_j - y_j)^2 \tag{12}$$

$$MAE = \frac{1}{n} \sum_{j=0}^{n} |\acute{y_j} - y_j| \tag{13}$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{j=0}^{n} (\hat{y}_j - y_j)^2} \tag{14}$$

where $\acute{y_j}$ denotes the predicted value, y_j is the real value, $\bar{y}_j$ is the mean value of y_j, and n is the number of data in the test set.

7. Adjust the parameters to find the optimal decomposition frequency and time step for FFNN so that the indexes can.
8. The model with the best indicators in the test set can be used to predict the original series. Figure 1 shows the flow of the proposed STL-FFNN hybrid model

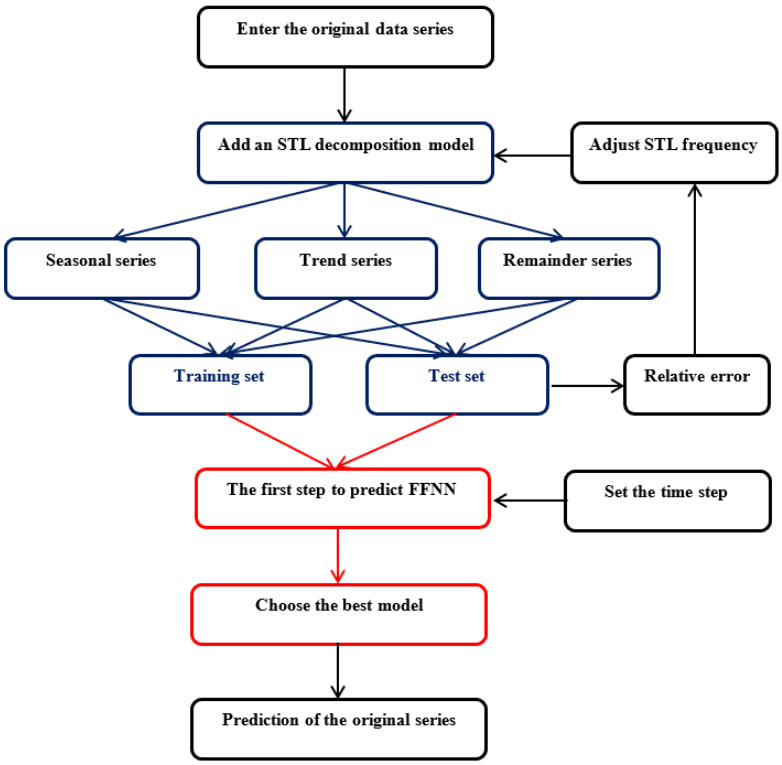

Figure 1: The flow of the proposed STL-FFNN hybrid model

6. Analysis using STL-FFNN model

The data used in the study represented real data on the monthly average of British pounds (in millions) spent by foreign visitors in the United Kingdom for the period from January 1986 to February 2020, with 410 views, obtained from the website https://www.kaggle.com/datasets.

The observations were divided into two groups, a training group and an evaluation group, where the training group consisted of observations from the beginning of the time series to February 2019, amounting to 398 observations, while the evaluation group kept the last 12 observations from the time series as an evaluation group for the prediction values of the models used.

First, we draw the data series to see the nature of the time series and which component it belongs to, which is represented by the monthly average of British pounds (in millions) spent by foreign visitors in the United Kingdom. Figure.2 represents the data series.

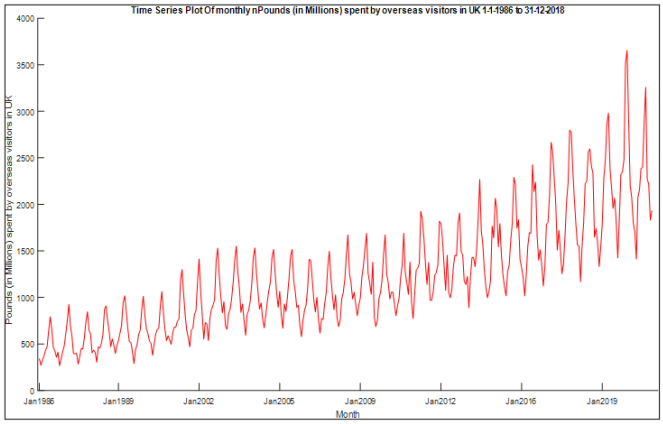

Figure 2: Data series

It is noted in Figure 2 that the time series follows the trend and seasonal components at the same time, as it follows the seasonal component because it is increasing at specific times of the year, as well as the trend component because the series is in an increasing pattern.

The next step after drawing the time series is to draw the total autocorrelation functions (ACF) and partial autocorrelation functions (PACF), because they are useful tools in the Box-Jenkins methodology. They are also useful in knowing which time differences fall outside the confines of the confidence interval set at $\pm\frac{1.96}{\sqrt{n}}$, where n represents the number of observations of the series. Temporality. Figure 3 represents the values of the ACF and PACF.

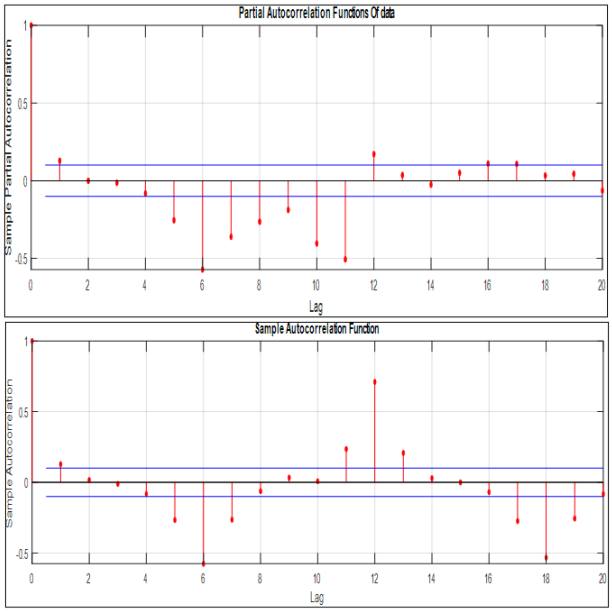

Figure 3: Values of the ACF and PACF

It is noticeable from plotting the ACF and PACF functions that there are lags time differences that fall outside the confines of the confidence interval, meaning that the ACF for $lags = 5, 6, 7, 8, 9, 10, 11, 12, 16, 17$ and the PACF for $lag = 1, 5, 6, 7, 11, 12, 13, 17, 18, 19$ fall outside the confidence interval, meaning that the series suffers from high fluctuation.

The next step in the data flow diagram for the hybrid model shown in Figure 1 is to divide the test set of data series into three subseries, which are the seasonal component series, the trend component series, and the remainder component series. Where Figure.4 represents the substrings of the test set of the data series.

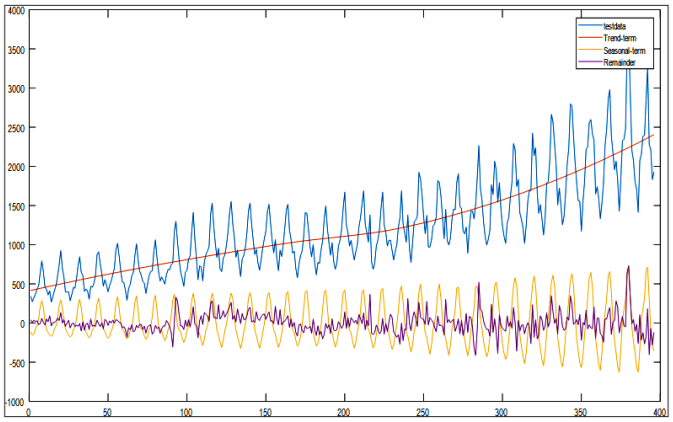

Figure 4: Substrings of the test set of the data series

Now that the time series has been decomposed into three sub-series, the resulting values for each sub-series are divided into two groups, a test set and a training set, with 80% training set and 20% test set. Then the FFNN neural network is run three times so that it runs once for each component of The three decomposition components (seasonality component, trend component, and residual component) are made up of 150 hidden layers. Where Table 1 represents the values of the training set.

Table 1: Training set values.

Unit	Initial value	Stopped value	Target value
Epoch	0	7	1000
Elapsed Time	-	00:00:02	-
Performance	6.1	0.0018	0
Gradient	14.2	0.00132	1e-07
Mu	0.001	1e-06	1e+10
Validation Checks	0	6	6

The forms of the training group were as follows:
Figure 5 represents a drawing of the test set and the training and validation set

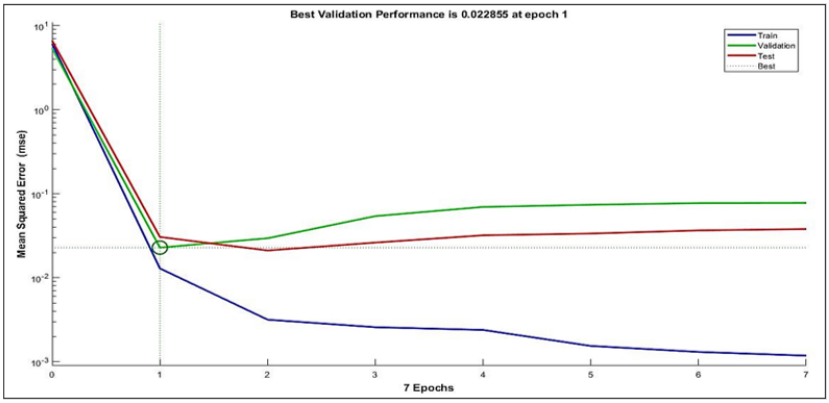

Figure 5: Drawing of the test set and the training and validation set

While Figure 6 represents a plot of regression values, Mu values, and validation test values

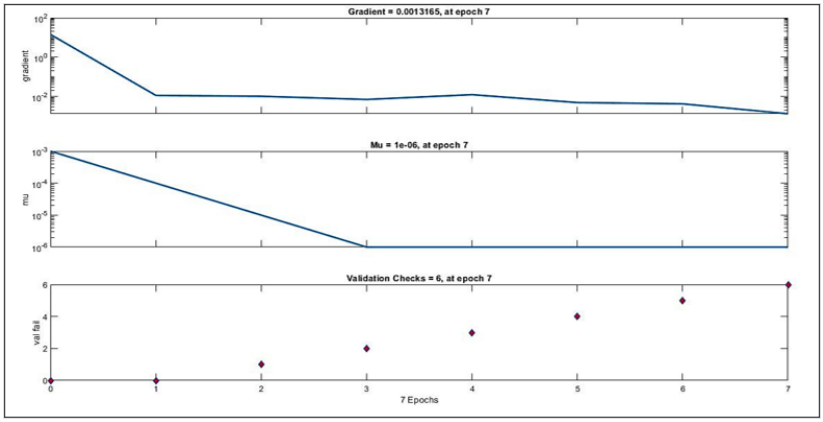

Figure 6: Plot of regression values, Mu values, and validation test values

While Figure 7 represents the regression plot for the test set and the training and validation set

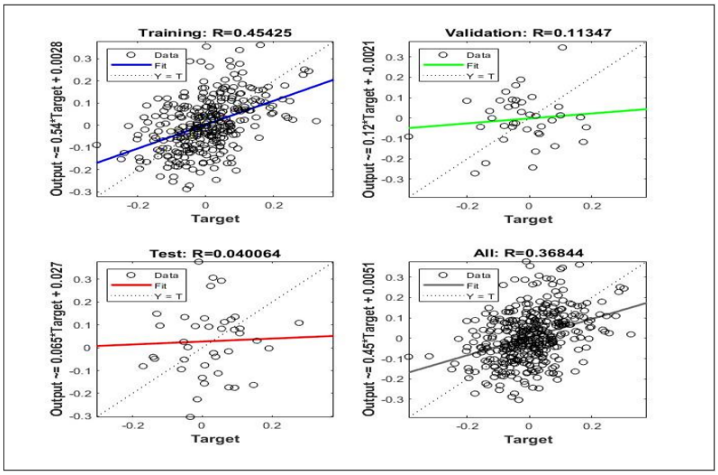

Figure 7: Regression plot for the test set and the training and validation set

The next step is after the neural network of the three components has completed its work in obtaining output values, the output series is drawn for the original series and the original series in addition to the sub-series. Figure 8 represents the output series for the original series and the original series in addition to the sub-sequences.

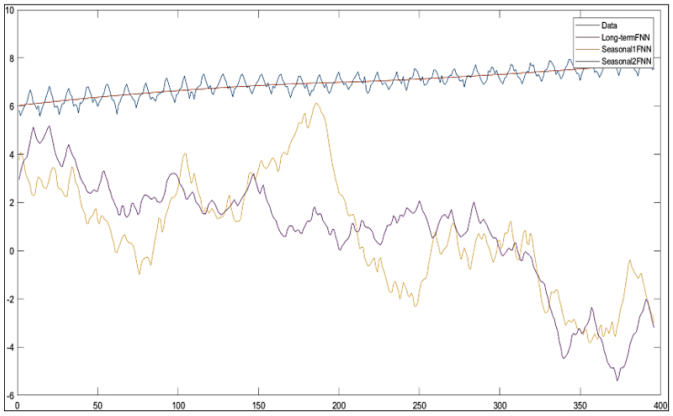

Figure 8: Output series for the original series and the original series in addition to the sub-sequences

Then the outputs of the three subnetworks are found, so that Figure 9 represents the outputs of the three subnetworks.

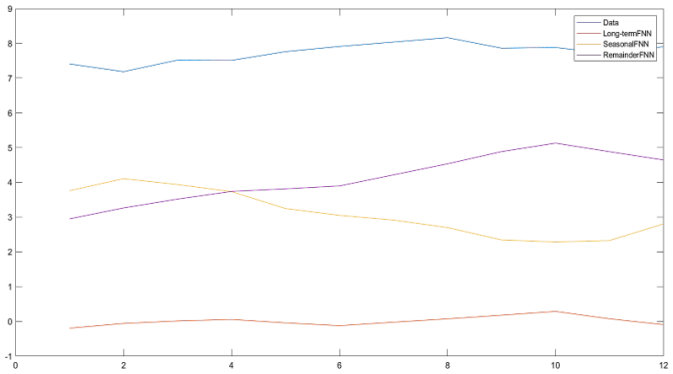

Figure 9: Outputs of the three subnetworks

The final step of the analysis is to use decomposition Eq.(1) to collect the outputs of the neural network to obtain a prediction of the values of the original time series. Figure 10 represents the predictions of the STL-FFNN hybrid model for the last 12 observations of the data series that were set as an evaluation set for the model prediction.

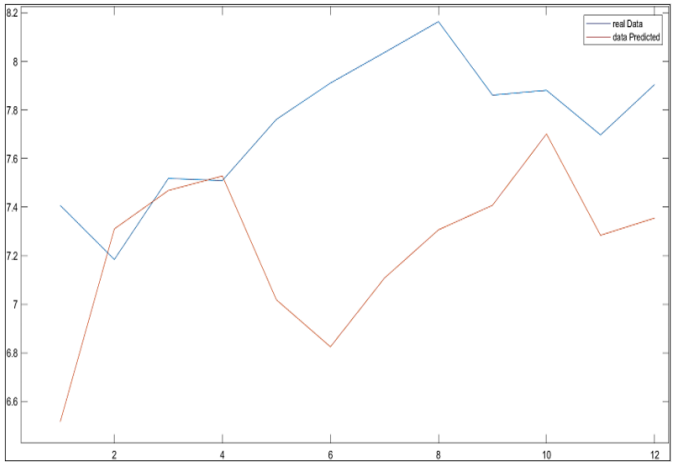

Figure 10: Predictions of the orthographic group relative to the original series

Table 2 represents the values of the last 12 observations in tens of thousands from the data series that were left as an evaluation group to determine the adequacy of the proposed model, while using Eqs. (13) and (14) to find the values of MAE and RMSE.

Table 2: Real data to Forecasting by STL-FFNN and value of MAE and RMSE

Real Data	Forecasting by STL-FFNN
7.4061	6.5164
7.1839	7.3098
7.5175	7.4679
7.5088	7.5276
7.7609	7.0178
7.9102	6.8245
8.0359	7.1073
8.1634	7.3058
7.8606	7.4070
7.8804	7.7007
7.6967	7.2835
7.9040	7.3543
MAE	0.5246
RMSE	0.6356293627

7. Conclusions

The proposed hybrid STL-FFNN model demonstrated high efficiency and high prediction accuracy on analysing the data series used, especially since this series suffers from fluctuation in the result of following several components of the time series, where the MAE and RMSE error values appeared very small compared to the size of a single observation, as segmentation of the series led to into several parts to give an accurate description of the state of the time series, which received these parts, the neural network to work four times

separately to find a prediction for each of the parts, then collect these parts to give a prediction for the proposed model.

References

[1] R. B. Cleveland, W. S. Cleveland, J. E. McRae, I. Terpenning, Stl: A seasonal-trend decomposition, Journal of Official Statistics 6 (1) (1990) 3–73.
[2] I. Stroud, P. Xirouchakis, Stl and extensions, Advances in Engineering Software 31 (2) (2000) 83–95.
[3] M. Szilvási-Nagy, G. Matyasi, Analysis of stl files, Mathematical and computer modelling 38 (7-9) (2003) 945–960.
[4] M. Theodosiou, Forecasting monthly and quarterly time series using stl decomposition, International Journal of Forecasting 27 (4) (2011) 1178–1195.
[5] A. Donzé, T. Ferrere, O. Maler, Efficient robust monitoring for stl, in: Computer Aided Verification: 25th International Conference, CAV 2013, Springer, Saint Petersburg, Russia, 2013, pp. 264–279.
[6] T. R. Kelley, Stl guiding the 21st century, Technology and engineering teacher 73 (4) (2013) 18.
[7] M. Van Eijnatten, F. H. Berger, P. De Graaf, J. Koivisto, T. Forouzanfar, J. Wolff, Influence of ct parameters on stl model accuracy, Rapid Prototyping Journal 23 (4) (2017) 678–685.
[8] H. Yin, D. Jin, Y. H. Gu, C. J. Park, S. K. Han, S. J. Yoo, Stl-attlstm: vegetable price forecasting using stl and attention mechanism-based lstm, Agriculture 10 (12) (2020) 612.
[9] D. Ničković, T. Yamaguchi, Rtamt: Online robustness monitors from stl, in: International Symposium on Automated Technology for Verification and Analysis, Springer, Hanoi, Vietnam, 2020, pp. 564–571.
[10] P. Varnai, D. V. Dimarogonas, On robustness metrics for learning stl tasks, in: 2020 American Control Conference (ACC), IEEE, Denver, CO, USA, 2020, pp. 5394–5399.
[11] D. Chen, J. Zhang, S. Jiang, Forecasting the short-term metro ridership with seasonal and trend decomposition using loess and lstm neural networks, IEEE Access 8 (2020) 91181–91187.
[12] A. M. Turing, The essential turing, Oxford University Press, Oxford, United Kingdom, 2004.
[13] S. A. Billings, Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains, John Wiley & Sons, Hoboken, United States, 2013.
[14] E. Demidova, A. Zaveri, E. Simperl, Semantic image-based profiling of users' interests with neural networks, Emerging Topics in Semantic Technologies: ISWC 2018 Satellite Events 36 (2018) 179.
[15] W. S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity, The bulletin of mathematical biophysics 5 (1943) 115–133.
[16] G. Bebis, M. Georgiopoulos, Feed-forward neural networks, Ieee Potentials 13 (4) (1994) 27–31.
[17] R. Eldan, O. Shamir, The power of depth for feedforward neural networks, in: Conference on learning theory, PMLR, New York, USA, 2016, pp. 907–940.
[18] H. F. Lui, W. R. Wolf, Construction of reduced-order models for fluid flows using deep feedforward neural networks, Journal of Fluid Mechanics 872 (2019) 963–994.
[19] E. Ozanich, P. Gerstoft, H. Niu, A feedforward neural network for direction-of-arrival estimation, The journal of the acoustical society of America 147 (3) (2020) 2035–2048.
[20] Y. Chen, C. Zhang, C. Liu, Y. Wang, X. Wan, Atrial fibrillation detection using a feedforward neural network, Journal of Medical and Biological Engineering 42 (1) (2022) 63–73.
[21] Z. Chen, X. Li, W. Wang, Y. Li, L. Shi, Y. Li, Residual strength prediction of corroded pipelines using multilayer perceptron and modified feedforward neural network, Reliability Engineering & System Safety 231 (2023) 108980.
[22] D. Konar, A. D. Sarma, S. Bhandary, S. Bhattacharyya, A. Cangi, V. Aggarwal, A shallow hybrid classical–quantum spiking feedforward neural network for noise-robust image classification, Applied Soft Computing 136 (2023) 110099.
[23] R. J. Hyndman, G. Athanasopoulos, Forecasting: principles and practice, OTexts, Melbourne, Australia, 2018.
[24] R. B. Cleveland, W. S. Cleveland, J. E. McRae, I. Terpenning, Stl: A seasonal-trend decomposition, J. Off. Stat 6 (1) (1990) 3–73.
[25] C. Gupta, S. G. Sharma, M. G. Bansal, Implementation of back propagation algorithm (of neural networks) in vhdl, Phd. thesis, Deemed University, India (2007).
[26] P. A. Idowu, C. Osakwe, A. K. Aderonke, E. R. Adaguno do, Prediction of stock market in nigeria using artificial neural network, International Journal of Intelligent Systems and Applications 4 (11) (2012) 68.
[27] M. Cilimkovic, Neural networks and back propagation algorithm, Institute of Technology Blanchardstown, Blanchardstown Road North Dublin 15 (1) (2015) 1–12.