symmetry

MDPI

Article

Time Series Seasonal Analysis Based on Fuzzy Transforms

Ferdinando Di Martino * and Salvatore Sessa

Dipartimento di Architettura, Università degli Studi di Napoli Federico II, via Toledo 402, 80134 Napoli, Italy; sessa@unina.it

* Correspondence: fdimarti@unina.it; Tel.: +39-081-253-8907; Fax: +39-081-253-8905

Received: 27 September 2017; Accepted: 11 November 2017; Published: 17 November 2017

Abstract: We define a new seasonal forecasting method based on fuzzy transforms. We use the best interpolating polynomial for extracting the trend of the time series and generate the inverse fuzzy transform on each seasonal subset of the universe of discourse for predicting the value of an assigned output. In the first example, we use the daily weather dataset of the municipality of Naples (Italy) starting from data collected from 2003 to 2015 making predictions on mean temperature, max temperature and min temperature, all considered daily. In the second example, we use the daily mean temperature measured at the weather station "Chiavari Caperana" in the Liguria Italian Region. We compare the results with our method, the average seasonal variation, Auto Regressive Integrated Moving Average (ARIMA) and the usual fuzzy transforms concluding that the best results are obtained under our approach in both examples. In addition, the comparison results show that, for seasonal time series that have no consistent irregular variations, the performance obtained with our method is comparable with the ones obtained using Support Vector Machine- and Artificial Neural Networks-based models.

Keywords: ARIMA; forecasting; fuzzy partition; fuzzy transform; time series

1. Introduction

Time series forecasting methods are quantitative techniques that analyze historical data of a variable for predicting its values. Traditional forecasting methods [1–7] use statistical techniques to estimate the future trend of a variable starting from numerical datasets. Many time series data contain seasonal patterns that have regular, repetitive and predictable changes that happen sequentially in a period of time which could be a year, a season, a month, a week, etc.

Different approaches have been developed to deal with trend and seasonal time series. Traditional approaches, such the moving average method, additive and multiplicative models, Holt-Winters exponential smoothing, etc. [3,4,5,7], use statistical methods for removing the seasonal components: they decompose the series into trend, seasonal, cyclical and irregular components [8]. Other statistical approaches are based on the Box–Jenkins model, called Autoregressive Integrated Moving Average (ARIMA) [3,5,7,9]. In the first phase of ARIMA, an auto-correlation analysis is performed for verifying if the series is non-stationary. Then, the series is transformed into a stationary series formed by the differences between the value at the actual moment and the value at the previous moment.

The main limitation of the seasonal ARIMA model is the fact that the process is considered to be linear. Many soft computing models have been presented in the literature for capturing nonlinear characteristics in seasonal time series, e.g., Support Vector Machine (SVM) [10,11] used for wind speed prediction [12], air quality forecasting [13], and rainfall forecasting [14]. SVM utilizes a kernel function to transform the input variables into a multi-dimensional feature space, and then the Lagrange multipliers are used for finding the best hyperplane to model the data in the feature space.

The main advantage of an SVM method is that that the solution is unique and there are no risk to move towards local minima, but some problems remain as the choice of the kernel parameters which influences the structure of the feature space, affecting the final solution. Another method is based on an Artificial Neural Network (ANN) [8,15–17]. The most widely used ANN architectures for forecasting problems are given by multi-layer Feed Forward Network (FNN) architectures [18,19], where the input nodes are given by the successive observations of the time series; that is, target y_t is a function of the values y_{t-1}, y_{t-2}, …, y_{t-p}, where p is the number of input nodes.

A variation of FNN is the Time Lagged Neural Network (TLNN) architecture [20,21], where the input nodes are the time series values at some particular lags. For example, in a time series having the month as seasonal period, the neural network used for forecasting the parameter value at the time t can contain input nodes corresponding to the lagged values at the time t−1, t−2,..,t−12. The key point is that an ANN can be considered a nonlinear auto-regression model. ANNs are inherently nonlinear and can accurately model complex characteristics in data patterns with respect to linear approaches such as ARIMA models. One of the main problems in the ANN forecasting models is the selection of appropriate network parameters. This operation is crucial since it strongly affects the final results. Furthermore, the presence of a high number of network parameters in the model can produce overtraining of data, giving rise to incorrect forecasting solutions.

To reduce the problems present in the SVM and ANN approaches, some authors have recently developed some hybrid models, e.g., genetic algorithms and tabu search (GA/TS) [11] and the Modified Firefly Algorithm (MFA).

In [22], a Discrete Wavelet Transform (DWT) algorithm is used to decompose the time series into linear and nonlinear components. Afterwards, the ARIMA and the ANN models are used to forecasting separately the two components.

In [23], the authors present four seasonal forecasting models: an adaptive ANN model that uses a genetic algorithm for evolving the ANN topology and the back-propagation parameter called ADANN (Automatic Design of Artificial Neural Networks), a SVM seasonal forecasting model and two hybrid ANN and SVM models based on linguistic fuzzy rules. The authors compare the four methods with the traditional ARIMA method, showing that the results under the four methods are comparable with the ones obtained using ARIMA. The best results are obtained by using the ADANN algorithm based on linguistic fuzzy rules but ANN-based methods are complex to manage and require more computational effort.

Various fuzzy modeling approaches are proposed in literature and applied in different fields for data analysis and knowledge exploring (for examples, see [24–27]).

To overcome the high computational complexity and the heavy dependence on the input parameters of the ANN models, a soft computing forecasting method based on fuzzy transforms [28] (for short, F-transforms) is presented in [29]. This method was applied to the North Atlantic Oscillation (NAO) data time series. The results are better than the ones obtained using the well known Wang–Mendel method [30] and Local Linear Wavelet Neural Network (LLWNN) techniques.

F-transforms are a powerful flexible method to be applied in various domains such as image compression [31], detection of attribute dependencies in data analysis [32] and extraction of the trend cycle of time series [33].

Strictly speaking, in [29], a forecasting index is calculated with respect to the data of the training set: if this index is smaller than or equal to an assigned threshold, the algorithm is stopped; otherwise, it is iterated by taking a finer fuzzy partition of the variable domain.When a fuzzy partition is settled, the algorithm controls that the training dataset is sufficiently dense with respect to the fuzzy partition; that is, each fuzzy set of the fuzzy partition has at least a non-zero membership degree in some point.

Here, we give a forecasting method based on F-transforms for seasonal time series analysis called Time Series Seasonal F-transform (TSSF). We give a partition of the time series dataset into seasonal pattern components. A seasonal pattern is related to a fixed period of the time series fluctuations: the seasonality is set on a fixed period such as year, month, week, etc. In the TSSF model, we assume that the different components affect the time series. We use the best polynomial

fit for estimating the trend of the time series. After de-trending the data by subtracting the trend from the time series dataset, we find a fuzzy partition of the dataset into seasonal subsets to which we apply the F-transforms by checking that the chosen partition is optimal for the density of the training data. In [29,32], the authors have developed this process: in particular, four forecasting indexes are proposed for assessing the quality of the results: Mean Absolute Deviation (MAD), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE) and Mean Absolute Deviation Mean (MADMEAN). An optimal fuzzy partition is found by calculating the corresponding RMSE and MADMEAN: if at least one of the two indices does not exceed a respective threshold, then the process stops and the given fuzzy partition is considered optimal.

1.1 TSSF Method for Forecasting Analysis

In the TSSF method, we essentially adopt the same procedure, but we prefer to use the MADMEAN index because this index is more robust than other forecasting indexes, as proven in [34]. We emphasize that the dimension of the optimal fuzzy partition can vary with the seasonal subset. In Figure 1, the TSSF method is synthetized in detail.

In the assessment phase, the best polynomial fit is applied for determining the trend in the training data. After de-trending the dataset, the time series is decomposed into S seasonal subsets to which the F-transform forecasting iterative method is applied. Initially, a coarse grained uniform fuzzy partition is fixed. If the subset is not sufficiently dense with respect to the fuzzy partition, the F-transform sub-process applied on the seasonal subset stops, else the MADMEAN index is calculated and, if it is greater than an assigned threshold, the F-transform sub-process is iterated considering a finer uniform fuzzy partition. The output is given by the inverse F-transforms obtained for each seasonal subset at the end of the corresponding sub-process.

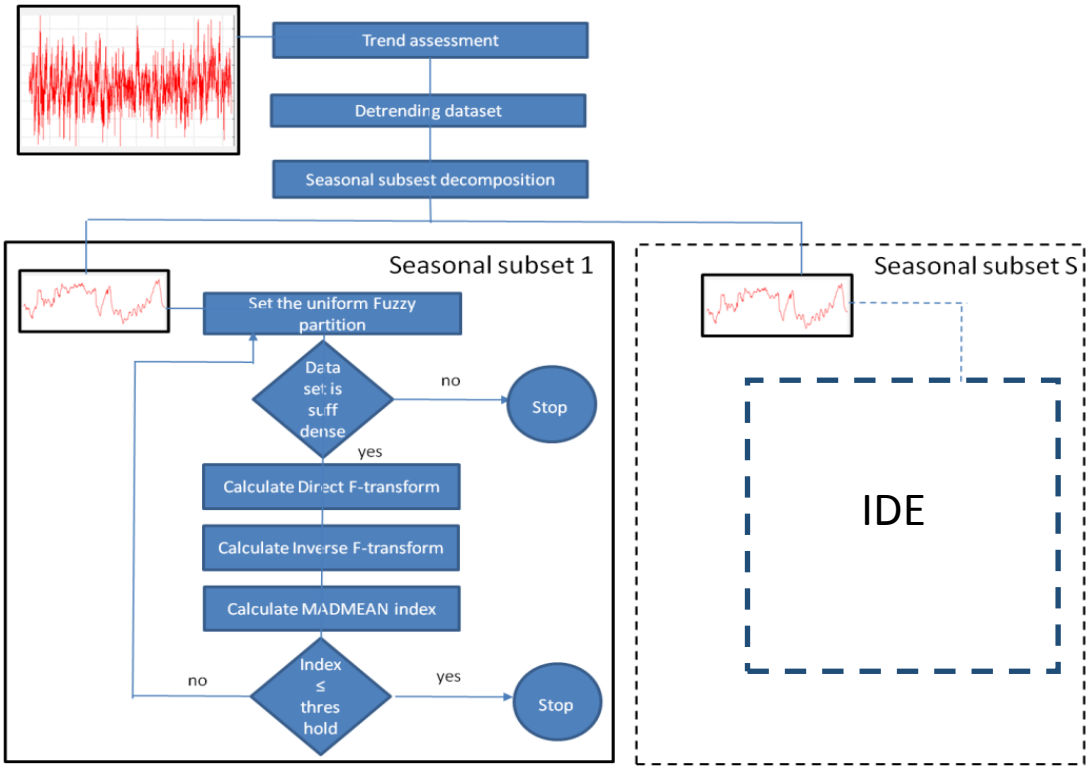


Figure 1. Schema of the TSSF method.

To forecast a value of a parameter y_0 at the time t, we calculate the *sth* seasonal subset of cardinality n(s), in which the time t is inserted. Then, we consider the *sth* inverse F-transform $f^F_{n(s)}(t)$. The forecasted value of the parameter is given by the formula $f^F_{n(s)}(t) + trend(t)$, where

trend(t) is the value of the polynomial trend at the time *t*. Figure 2 illustrates the application of the TSSF for forecasting analysis.

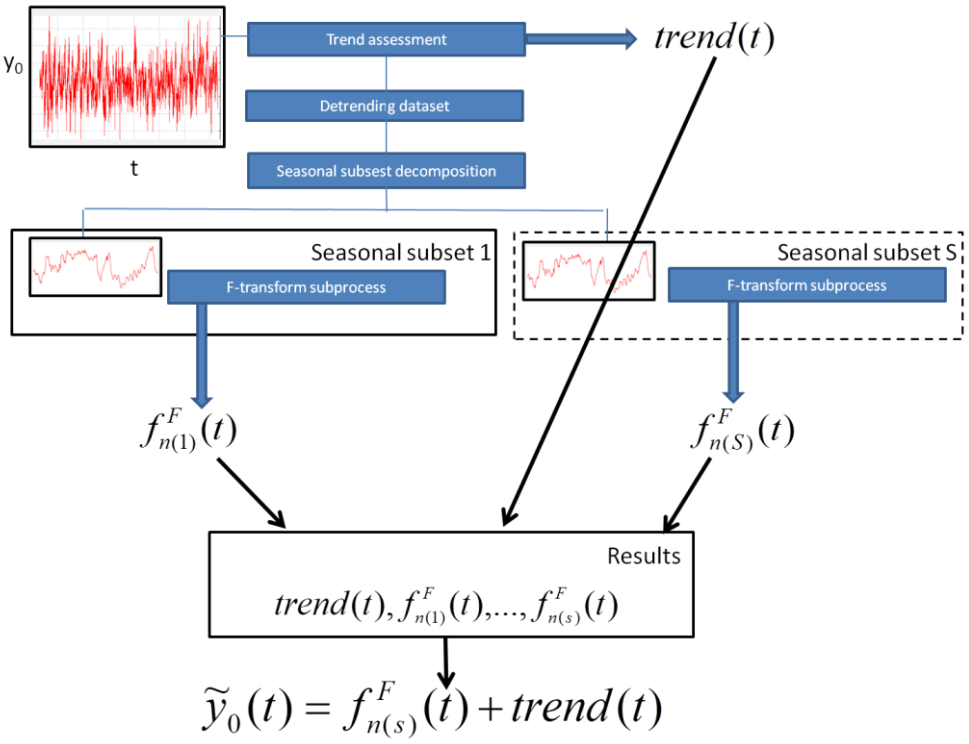


Figure 2. The TSSF method for forecasting analysis.

For sake of completeness, we recall the concept of F-transform in Section 2. Section 3 contains the F-transform-based prediction method. In Section 4, we describe our TSSF method applied on the climate dataset of Naples (Italy) (www.ilmeteo.it/portale/archivio-meteo/Napoli) whose results are given in Section 5 that contain also comparisons with average seasonal variation, ARIMA and the traditional prediction F-transform methods. Section 6 reports the conclusions.

2. Direct and Inverse F-Transform

Following the definitions and notations of [28], let $n \geq 2$ and $x_1, x_2, \ldots, x_n$ be points of [a,b], called nodes, such that $x_1 = a < x_2 < \ldots < x_n = b$. The family of fuzzy sets $A_1, \ldots, A_n$: [a,b] → [0,1], called basic functions, is a fuzzy partition of [a,b] if the following hold:

- $A_i(x_i) = 1$ for every $i = 1,2, \ldots,n$;
- $A_i(x) = 0$ if $x \notin]x_{i-1},x_{i+1}[$ for $i = 2, \ldots,n-1$;
- $A_i(x)$ is a continuous function on [a,b];
- $A_i(x)$ strictly increases on $[x_{i-1}, x_i]$ for $i = 2, \ldots, n$ and strictly decreases on $[x_i,x_{i+1}]$ for $i = 1, \ldots, n-1$;
- $A_1(x) + \ldots + A_n(x) = 1$ for every $x \in [a,b]$.

The fuzzy sets $\{A_1(x),\ldots,A_n(x)\}$ form an uniform fuzzy partition if $n \geq 3$ and $x_i = a + h\cdot(i-1)$, where $h = (b-a)/(n-1)$ and $i = 1, 2, \ldots, n$ (that is the nodes are equidistant);

- $A_i(x_i - x) = A_i(x_i + x)$ for every $x \in [0,h]$ and $i = 2, \ldots, n-1$;
- $A_{i+1}(x) = A_i(x - h)$ for every $x \in [x_i, x_{i+1}]$ and $i = 1,2, \ldots, n-1$.

Here, we are only interested in the discrete case, i.e., for functions f defined on the set P of points $p_1, \ldots, p_m$ of [a,b]. If P is sufficiently dense with respect to the given partition $\{A_1, A_2, \ldots, A_n\}$, i.e., for each $i \in \{1,\ldots,n\}$ there exists an index $j \in \{1,\ldots,m\}$ such that $A_i(p_j) > 0$, we define $\{F_1, F_2, \ldots, F_n\}$ as the discrete direct F-transform of f with respect to $\{A_1, A_2, \ldots, A_n\}$, where F_i is given by

$$F_i = \frac{\sum_{j=1}^{m} f(p_j)A_i(p_j)}{\sum_{j=1}^{m} A_i(p_j)} \quad (1)$$

for i = 1, …, n. Similarly we define the discrete inverse F-transform of f with respect to {A_1, A_2, …, A_n} by setting

$$f_n^F(p_j) = \sum_{i=1}^{n} F_i A_i(p_j) \quad (2)$$

for every j $\in$ {1, …, m}. We can extend the above concepts to functions in k (≥2) variables. In the discrete case, we assume that the function f(x_1,x_2, …, x_k) is defined on m points $p_j = (p_{j1}, p_{j2}, \ldots, p_{jk}) \in [a_1,b_1] \times [a_2,b_2] \times \ldots \times [a_k,b_k]$ for j = 1,…,m. We say that P = {(p_{11}, p_{12}, …, p_{1k}), …, (p_{m1}, p_{m2}, …,p_{mk})} is sufficiently dense with respect to the fuzzy partitions

$$\{A_{11}, A_{12}, \ldots, A_{1n_1}\}, \ldots, \{A_{k1}, A_{k2}, \ldots, A_{kn_k}\} \quad (3)$$

of [a_1,b_1], …, [a_k,b_k], respectively, if for each k-tuple {h_1, …, h_k} $\in$ {1, …, n_1} × … × {1, …, n_k}, there exists a point $p_j = (p_{j1}, p_{j2}, \ldots, p_{jk})$ in P, j = 1, …, m, such that $A_{1h_1}(p_{j1}) \cdot \ldots \cdot A_{kh_k}(p_{jk}) > 0$.

In this case, we define the (h_1,h_2,…,h_k)-*th* component $F_{h_1h_2\ldots h_k}$ of the discrete direct F-transform of f with respect to the basic Function (3) as

$$F_{h_1h_2\ldots h_k} = \frac{\sum_{j=1}^{m} f(p_{j1}, p_{j2}, \ldots p_{jk}) \cdot A_{1h_1}(p_{j1}) \cdot A_{2h_2}(p_{j2}) \cdot \ldots \cdot A_{kh_k}(p_{jk})}{\sum_{j=1}^{m} A_{1h_1}(p_{j1}) \cdot A_{2h_2}(p_{j2}) \cdot \ldots \cdot A_{kh_k}(p_{jk})} \quad (4)$$

Now, we define the discrete inverse F-transform of f with respect to the Function (3) to be the following function by setting for each point $p_j = (p_{j1}, p_{j2}, \ldots, p_{jk}) \in [a_1,b_1] \times \ldots \times [a_k,b_k]$:

$$f_{n_1\ldots n_k}^F(p_{j1}, \ldots, p_{jk}) = \sum_{h_1=1}^{n_1} \sum_{h_2=1}^{n_2} \ldots \sum_{h_k=1}^{n_k} F_{h_1h_2\ldots h_k} \cdot A_{1h_1}(p_{j1}) \ldots A_{kh_k}(p_{jk}) \quad (5)$$

for j = 1, …, m. The following theorem holds [28]:

Theorem 1. *Let f(x_1,x_2,…,x_k) be given on the set of points P = {(p_{11},p_{12}, …, p_{1k}) ,(p_{21}, p_{22}, …, p_{2k}), …,(p_{m1}, p_{m2}, …, p_{mk})} $\in$ [a_1,b_1] × [a_2,b_2] × … × [a_k,b_k]. Then, for every ε > 0, there exist k integers $n_1 = n_1(\varepsilon)$, …, $n_k = n_k(\varepsilon)$ and k related fuzzy partitions (3) of [a_1,b_1], …, [a_k,b_k], respectively, such that the set P is sufficiently dense with respect to them and for every $p_j = (p_{j1}, p_{j2}, \ldots, p_{jk})$ in P, j = 1, …, m, the following inequality holds:*

$$\left| f(p_{j1}, \ldots, p_{jk}) - f_{n_1\ldots n_k}^F(p_{j1}, \ldots, p_{jk}) \right| < \varepsilon. \quad (6)$$

3. F-Transform Forecasting Method

We now describe the F-transform forecasting algorithm presented in [29]. Let M be assigned input–output pairs data ($x^{(j)}$, $y^{(j)}$) in the following form:

$$
\begin{aligned}
&(\mathbf{x}^{(1)}, \mathrm{y}^{(1)}) = (x_1^{(1)}, ..., x_i^{(1)} ..., x_n^{(1)}, \mathrm{y}^{(1)}) \\
&... \\
&(\mathbf{x}^{(j)}, \mathrm{y}^{(\mathrm{j})}) = (x_1^{(j)}, ..., x_i^{(j)} ..., x_n^{(j)}, \mathrm{y}^{(\mathrm{j})}) \\
&... \\
&(\mathbf{x}^{(M)}, \mathrm{y}^{(\mathrm{M})}) = (x_1^{(M)}, ..., x_i^{(M)}, ..., x_n^{(M)}, \mathrm{y}^{(\mathrm{M})})
\end{aligned}
\tag{7}
$$

for j = 1, 2, …, M. The task is to generate a fuzzy rule-set from the M pairs $(\mathrm{x}^{(\mathrm{j})}, \mathrm{y}^{(\mathrm{j})})$ to determine a mapping f from the input-space R^{n} into the output-space R. We assume that the $x_i^{(1)}, ..., x_i^{(j)}, ..., x_i^{(M)}$ lie in $\left[x_i^-, x_i^+\right]$ for every i = 1, …, n, and $\mathrm{y}^{(1)}, \mathrm{y}^{(2)}, \ldots, \mathrm{y}^{(\mathrm{M})}$ in $[\mathrm{y}^-, \mathrm{y}^+]$. The F-transform forecasting method calculates a function $y = f(x_1, ..., x_n)$, which approximates the data. Similar to in [32], we create a partition of $\mathrm{n_i}$ fuzzy sets for each domain $[x_i^-, x_i^+]$, hence we construct the respective direct and inverse F-transforms (Equations (3) and (4)) to estimate an approximation of f. We illustrate this forecasting method in the following steps:

(1) Give a uniform partition composed by $\mathrm{n_i}$ fuzzy sets ($\mathrm{n_i} \geq 3$) $\{A_{i1}, ..., A_{ini}\}$ of the domain $[x_i^-, x_i^+]$ of each variable $\mathrm{x_i}$, i = 1, …, n. If $X_{i1}, ..., X_{is}, ..., X_{in_i}$ are the nodes of each interval $[X_i^-, X_i^*]$, each function $\mathrm{A_{is}}$ is defined in the following way for s = 1, …, $\mathrm{n_i}$:

$$
\begin{aligned}
A_{i1}(x) &= \begin{cases} 0.5 \cdot (1 + \cos\frac{\pi}{s_i}(x - x_{i1})) & \text{if } \mathrm{x} \in [\mathrm{x_{i1}}, \mathrm{x_{i2}}] \\ 0 & \text{otherwise} \end{cases} \\
A_{is}(x) &= \begin{cases} 0.5 \cdot (1 + \cos\frac{\pi}{s_i}(x - x_{is})) & \text{if } \mathrm{x} \in [\mathrm{x_{i(s-1)}}, \mathrm{x_{i(s+1)}}] \\ 0 & \text{otherwise} \end{cases} \\
A_{in_i}(x) &= \begin{cases} 0.5 \cdot (1 + \cos\frac{\pi}{s_i}(x - x_{i(n_i-1)})) & \text{if } \mathrm{x} \in [x_{i(n_i-1)}, x_{in_i}] \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
\tag{8}
$$

where $\mathrm{x_{i1}} = x_i^-$, $X_{in_i} = X_i^+$, $s_i = \left(x_i^+ - x_i^-\right)/\left(n_i - 1\right)$ and $x_{is} = x_i^- + s_i \cdot (s-1)$.

(2) If the dataset is not sufficiently dense with respect to the fuzzy partition, i.e., if there exists a variable $\mathrm{x_i}$ and a fuzzy set $\mathrm{A_{is}}$ of the corresponding fuzzy partition such that $A_{is}(x_i^{(r)}) = 0$ for each r = 1, …, M, the process stops otherwise calculate the $\mathrm{n_1 \cdot n_2 \cdot \ldots \cdot n_k}$ components of the direct F-transform of f with Equation (3) by setting k = n, $p_{j1} = x_1^{(j)}$, …, $p_{jn} = x_n^{(j)}$ and $y(j) = f(x_1^{(j)}, x_2^{(j)}, ..., x_n^{(j)})$, obtaining the following quantity:

$$
F_{h_1 h_2 ... h_n} = \frac{\sum_{j=1}^{M} y^{(j)} \cdot A_{1h_1}(x_1^{(j)}) \cdot ... \cdot A_{nh_n}(x_n^{(j)})}{\sum_{j=1}^{M} A_{1h_1}(x_1^{(j)}) \cdot ... \cdot A_{nh_n}(x_n^{(j)})}
\tag{9}
$$

(3) Calculate the discrete inverse F-transform as

$$
f^F_{n_1 ... n_n}\left(x_1^{(j)}, ..., x_n^{(j)}\right) = \sum_{h_1=1}^{n_1} \sum_{h_2=1}^{n_2} ... \sum_{h_n=1}^{n_n} F_{h_1 h_2 ... h_n} \cdot A_{1h_1}(x_1^{(j)}) \cdot ... \cdot A_{nh_n}(x_n^{(j)})
\tag{10}
$$

to approximate the function f.

(4) Calculate the forecasting index as

$$MADMEAN = \frac{\sum_{j=1}^{M} \left| f^{F}_{n_1 n_2 \ldots n_n}(x_1^{(j)}, \ldots, x_n^{(j)}) - y^{(j)} \right|}{\sum_{j=1}^{M} y^{(j)}} \tag{11}$$

If Equation (11) is less or equal to an assigned threshold, then the process stops; otherwise, a finer fuzzy partition is taken and the process restarts from the step 2.

In [29], four forecasting indices, RMSE, MAPE, MAD, MADMEAN, were proposed, but we prefer to use MADMEAN, which is the best one in terms of accuracy, as proven in [34].

4. TSSF Method

We consider time series of data formed from observations of an original parameter y_0 measured at different times. The dataset is formed by M measure pairs as $\left(t^{(0)}, y_0^{(0)}\right), \left(t^{(1)}, y_0^{(1)}\right), \ldots, \left(t^{(M)}, y_0^{(M)}\right)$.

Our aim is to evaluate seasonal fluctuations of a time series by using the F-transform method. First, we use a polynomial fitting for calculating the trend of the phenomenon with respect to the time. Afterwards, we subtract the trend from the data obtaining a new dataset of the fluctuations $y^{(j)}$, being $y^{(j)} = y_0^{(j)} - trend(t^{(j)})$. After de-trending the dataset, we use a partition into S subsets, being S considered as seasonal period. Each subset represents the seasonal fluctuation with respect to the trend.

The seasonal data subset is composed by M_s pairs, expressing the fluctuation measures of the parameter y_0 at different times: $(t^{(1)}, y^{(1)})$, $(t^{(2)}, y^{(2)})$ .... $(t^{(M_s)}, y^{(M_s)})$, where $y^{(j)}$ is given by the original measure $y_0^{(j)}$ at the time $t^{(j)}$ minus the trend calculated at that time. The formulae of the corresponding one-dimensional directed and inverse F-transforms, considering n basic functions, are given as

$$F_h = \frac{\sum_{j=1}^{M_S} y^{(j)} \cdot A_h(t^{(j)})}{\sum_{j=1}^{M_S} A_h(t^{(j)})} \tag{12}$$

$$f_n^F(t) = \sum_{h=1}^{n} F_h \cdot A_h(t) \tag{13}$$

respectively. The inverse F-transform (13) approximates the seasonal fluctuation at the time t. In our method, we start with three basic functions and verify that the subset of data is sufficiently dense with respect to this fuzzy partition. After calculating the directed and inverse F-transform with Equations (12) and (13), respectively, we calculate the $(MADMEAN)_s$ index for the *sth* fluctuation subset of data given as

$$(MADMEAN)_s = \frac{\sum_{j=1}^{M_s} \left| f_n^F(t^{(j)}) - y^{(j)} \right|}{\sum_{j=1}^{M_s} y^{(j)}} \tag{14}$$

If Equation (14) is greater than an assigned threshold, than the process is iterated considering a fuzzy partition of dimension n:= n + 1; otherwise, the iteration process stops. Our algorithm is illustrated in Figure 3.

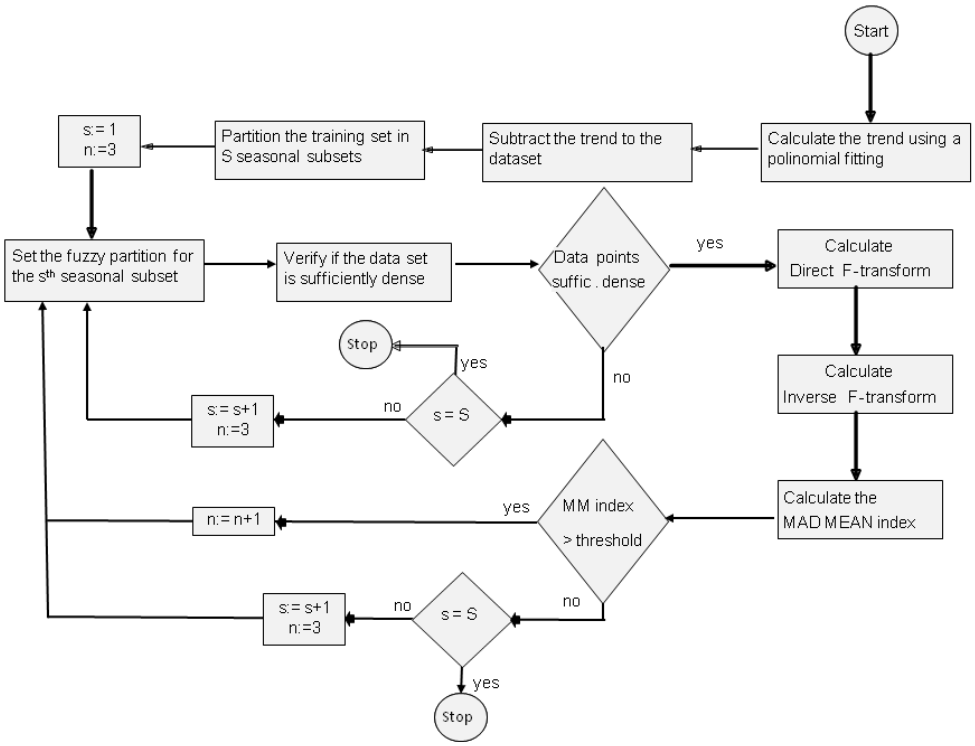


Figure 3. Flux diagram related of the TSSF method.

For the *sth* fluctuation subset, we obtain the inverse F-transform by using the following n(s) basic functions:

$$f_{n(s)}^{F}(t) = \sum_{h=1}^{n(s)} F_h \cdot A_h(t) \tag{15}$$

For evaluating the value of the parameter y_0 at the time t in the *sth* seasonal period, we add to $f_{n(s)}^{F}(t)$ the trend calculated at the time t, i.e., trend(t), obtaining the following value:

$$\tilde{y}_0(t) = f_{n(s)}^{F}(t) + trend(t) \tag{16}$$

For evaluating the accuracy of the results, we can use the MADMEAN index given by

$$\text{MADMEAN} = \frac{\sum_{j=1}^{M} \left| \tilde{y}_0(t^{(j)}) - y_0^{(j)} \right|}{\sum_{j=1}^{M} y_0^{(j)}} \tag{17}$$

For sake of completeness, we describe also the values of the above-cited indices:

- Root Mean Square Error (RMSE) is defined as:

$$\text{RMSE} = \left(\frac{\sum_{j=1}^{M} \left(\tilde{y}_0(t^{(j)}) - y_0^{(j)} \right)^2}{M} \right)^{1/2} \tag{18}$$

- Mean Absolute Percentage Error (MAPE) is defined as:

$$\mathrm{MAPE} = \left[\frac{1}{M} \sum_{j=1}^{M} \left| \frac{\tilde{y}_0(t^{(j)}) - y_0^{(j)}}{y_0^{(j)}} \right| \right] \times 100 \tag{19}$$

- Mean Absolute Deviation (MAD) is defined as:

$$\mathrm{MAD} = \frac{\sum_{j=1}^{M} \left| \tilde{y}_0(t^{(j)}) - y_0^{(j)} \right|}{M} \tag{20}$$

The TSSF algorithm is schematized in Appendix A. To evaluate the MADMEAN threshold, we consider a pre-processing phase in which we apply the k-cross-validation technique to control the presence of over-fitting in the learning data. We set a seasonal subset and we apply a random partition in k folds. The union of k−1 subsets is used as training set and the other subset is used as validation set. Then we consider a fuzzy partition with n = 3 and apply Equation (17) to the training set calculating the MADMEAN index; then we apply Equation (18) to the validation test to calculate the RMSE index. We repeat this process for all the k folds, obtaining the mean MADMEAN and RMSE indexes as

$$\mathrm{MADMEAN}(n) = \frac{\sum_{i=1}^{k} \mathrm{MADMEAN}_i(n)}{k} \tag{21}$$

$$\mathrm{RMSE}(n) = \frac{\sum_{i=1}^{k} \mathrm{RMSE}_i(n)}{k} \tag{22}$$

where MADMEAN(n) and RMSE(n) are the mean MADMEAN and RMSE using the fuzzy partition size n, respectively.

We repeat this process using many values of n. Then, we plot in a graph the RMSE(n) by varying n. The threshold value is set as the mean MADMEAN obtained in correspondence of the value n for which there is a plateau in the RMSE graph.

In next section, we present the results of tests in which the TSSF algorithm is applied to seasonal time series. In our tests, we apply the pre-processing phase described below in which we partition the dataset in k folds, setting k = 10. Comparisons are pointed out with respect to ARIMA, the F-transform methods and SVM and ADANN proposed in [23].

5. Experiments on Time Series Data

In a first experiment, we use a dataset composed from climate data (mean, maximum and minimum temperature, pressure, speed of the wind, etc., measured every day) of Naples (Italy) collected at the webpage: www.ilmeteo.it/portale/archivio-meteo/Napoli. The following main climate parameters are measured every thirty minutes: Min temperature (°C), Mean temperature (°C), Max temperature (°C), Dew point (°C), Mean humidity (%), Mean view (km), Mean wind speed (km/h), Max wind speed (km/h), Gust of wind (km/h), Mean pressure on sea level (mb), Mean pressure (mb), and Millimeters of rain (mm).

For sake of brevity, we limit the results for the parameters mean, max and min temperature. As training dataset, we consider these data recorded in the months July and August from 01 July 2003 to 16 August 2015, hence for 806 days represented as abscissas (via a number ID) in Figures 4 and 5, respectively. The daily mean and max temperature is represented on the ordinate axis in Figures 4 and 5, respectively. We obtain the best fit polynomial of nine degree ($y = \sum_{i=1}^{9} a_i \cdot x^i$) (red color in Figures 4 and 5) whose coefficients are given in Appendix B.

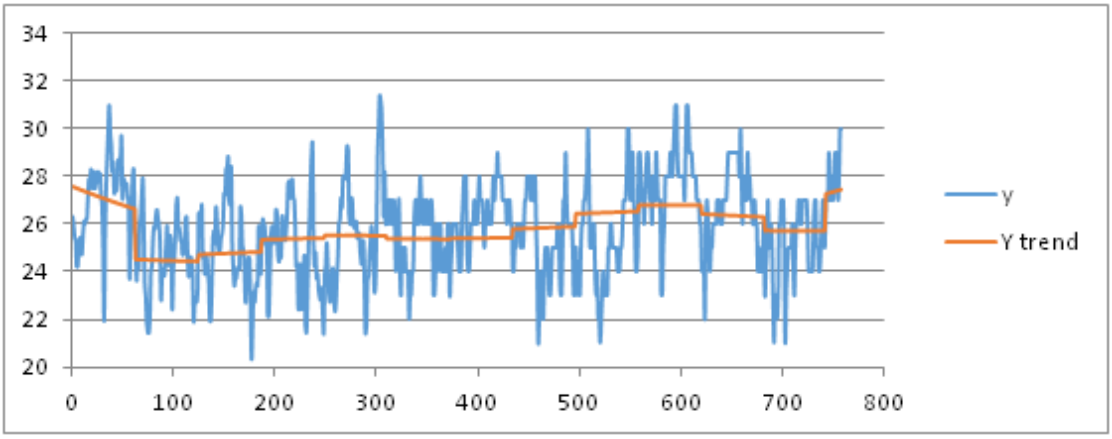


Figure 4. Trend of the mean temperature in the months of July and August (from 1 July 2003 to 16 August 2015) obtained by using a ninth-degree polynomial fitting.

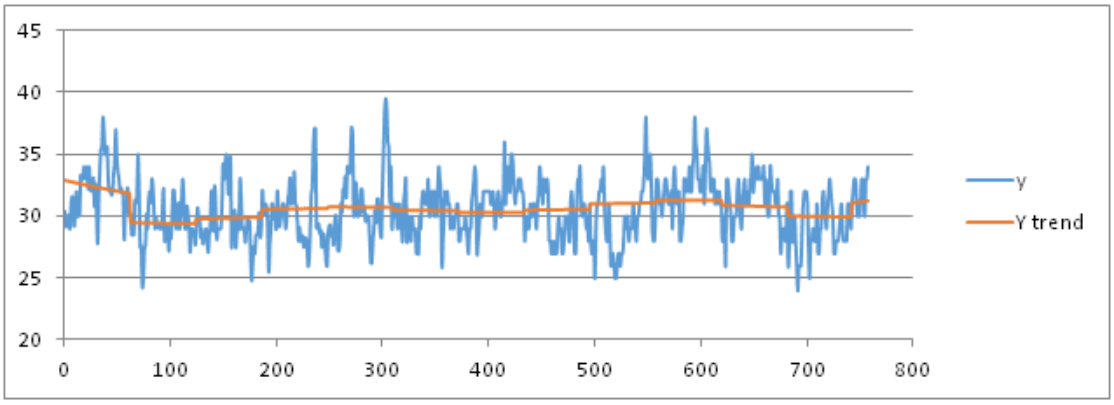
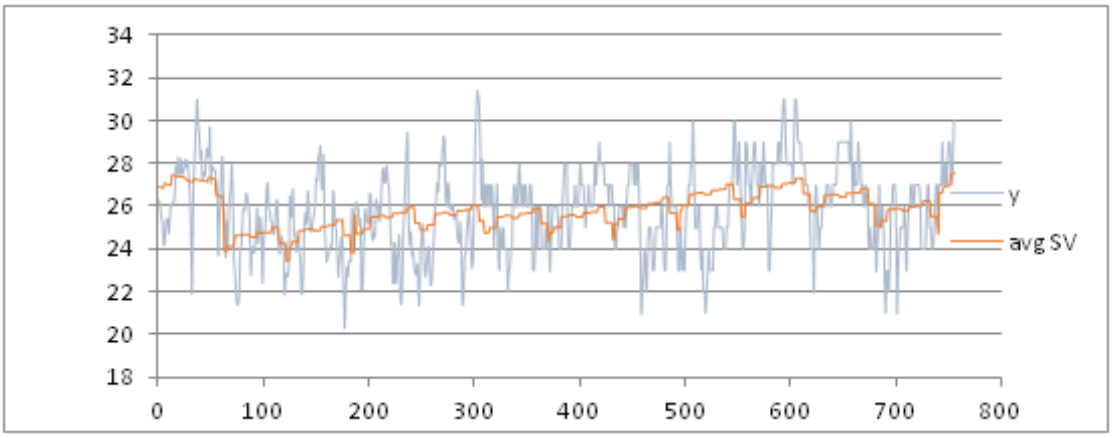


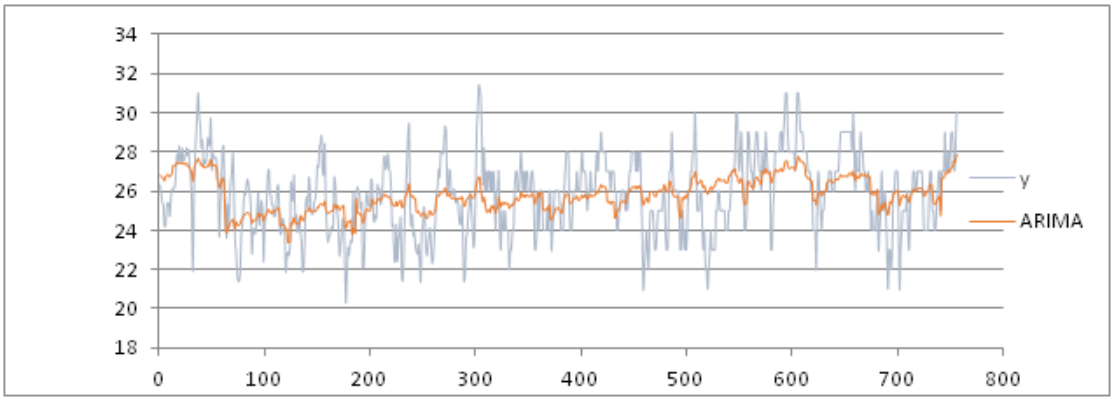
Figure 5. Trend of the max temperature in the months of July and August (from 1 July 2003 to 16 August 2015) obtained by using a best fit polynomial of nine degree.

We consider the week as seasonal period, partitioning the data set into nine seasonal subsets: Week 27 (July 1) to Week 36 (August 31). After applying the pre-processing phase, we set the threshold of the MADMEAN index to the value 5. Then, we apply the TSSF algorithm to the daily mean and max temperature. We also give the results of the other three methods, plotted in Figures 6a–d and 7a–d, respectively:

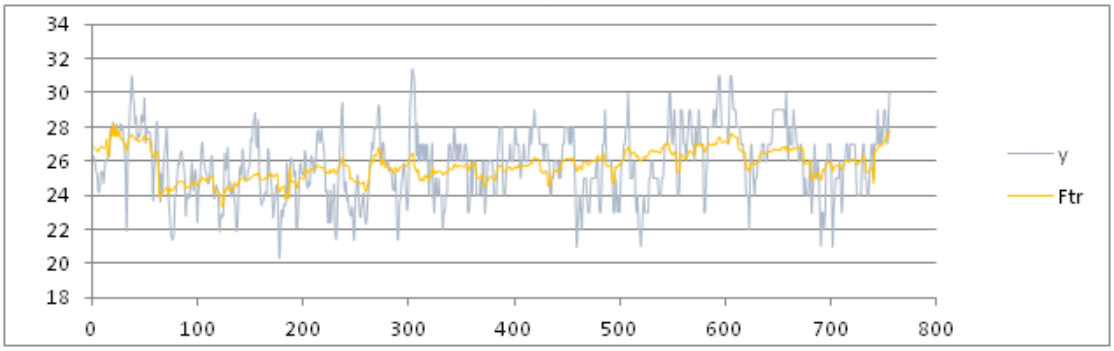
(1) Average seasonal variation method [3] (labeled as avgSV): This method calculates the mean seasonal variation for each seasonal period and adds the mean seasonal variation to the trend value.
(2) Seasonal ARIMA method: In our experiments, we used the forecasting tool ForecastPro [35]. As highlighted in [23], this tool is well known for ensuring the best performance in the use of forecasting ARIMA models.
(3) F-transform prediction method [29]: This is applied to the complete dataset (labeled as F-transforms).



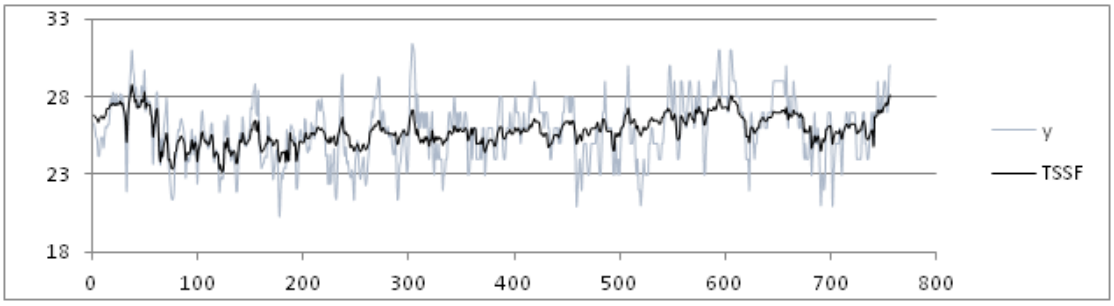
(**a**)



(**b**)



(**c**)



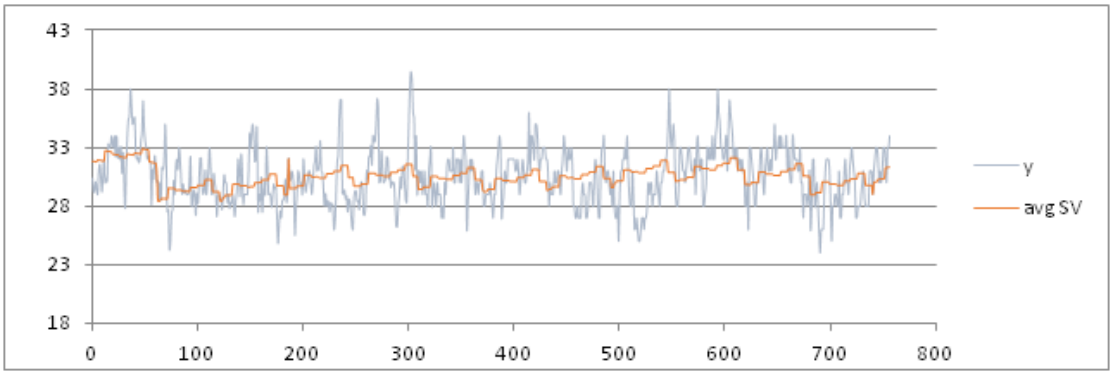
(**d**)

Figure 6. (**a**) Results obtained for the mean temperature by using the avgSV method; (**b**) results obtained for the mean temperature by using the ARIMA method; (**c**) results obtained for the mean temperature by using the F-transforms method; and (**d**) results obtained for the mean temperature by using the TSSF method.

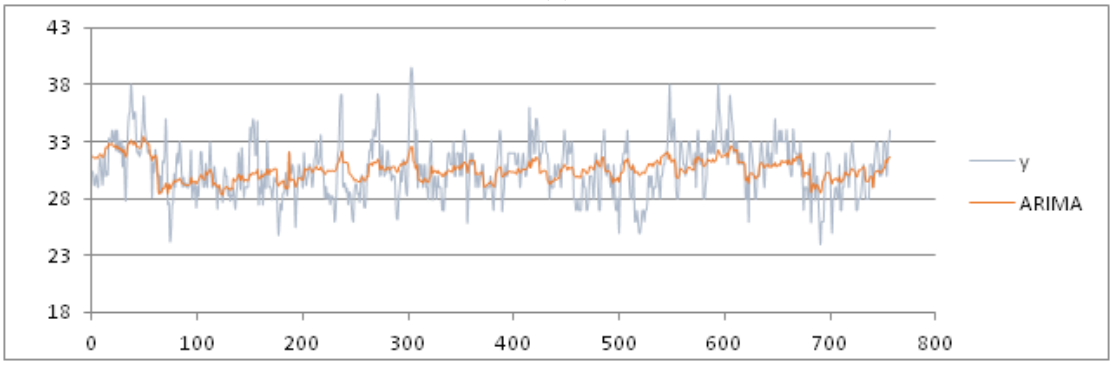
In Table 1, we show the four indices obtained using the four methods. The best results for the mean temperature are obtained using the TSSF method, with a MADMEAN of 4.22.

Table 1. RMSE, MAPE, MAD, and MADMEAN indices for the mean temperature.

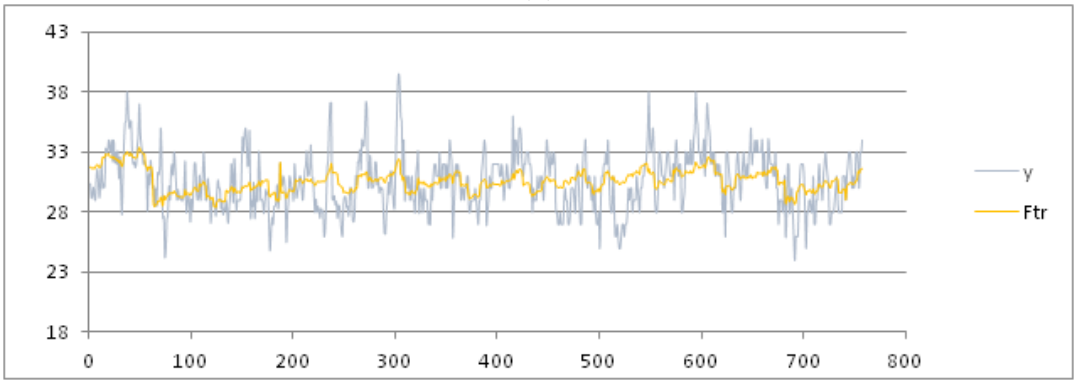
Forecasting Method	RMSE	MAPE	MAD	MADMEAN
avgSV	1.78	5.60%	1.42	5.50
ARIMA	1.55	4.89%	1.24	4.80
F-transforms	1.61	4.96%	1.28	5.05
TSSF	1.37	4.29%	1.09	4.22



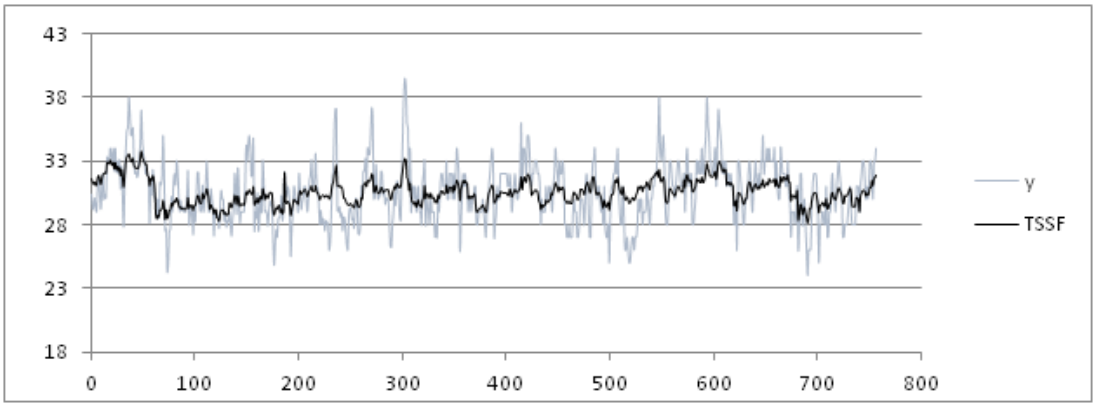
(**a**)



(**b**)



(**c**)



(**d**)

Figure 7. (**a**) Results for the max temperature in the months of July and August under avgSV method; (**b**) results for the max temperature in the months of July and August under ARIMA method; (**c**) results for the max temperature in the months of July and August under the F-transforms method; and (**d**) results for the max temperature in the months of July and August under the TSSF method.

In Table 2, we show the four indexes obtained using the four methods. The best results for the max temperature are obtained using TSSF, with a MADMEAN of 4.53.

Table 2. RMSE, MAPE, MAD and MADMEAN indices for the max temperature.

Forecasting Method	RMSE	MAPE	MAD	MADMEAN
avgSV	2.21	5.74%	1.74	5.70
ARIMA	1.93	5.03%	1.53	4.99
F-transforms	1.98	5.17%	1.56	5.13
TSSF	1.76	4.57%	1.39	4.53

We now present the results of other experiments in which the variation of the min temperature is explored during the years. We assume the month as seasonal period: the training dataset is formed by all the measures recorded from 1 January 2003 to 31 December 2015. It is partitioned into 12 seasonal subsets corresponding to 12 months of a year. After applying the pre-processing phase, we set the threshold of the MADMEAN index to the value 6. In Figure 8, we show the data and the trend obtained using a best fit polynomial of nine degree (see Appendix A).

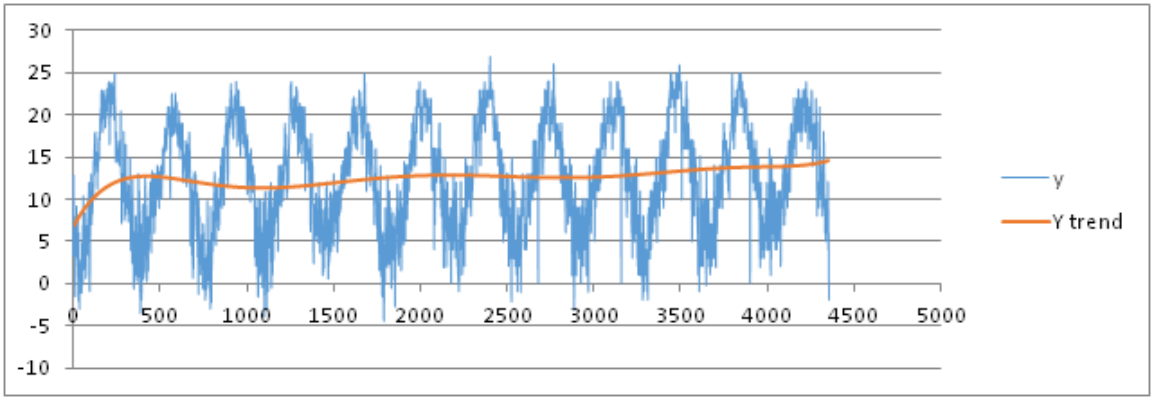
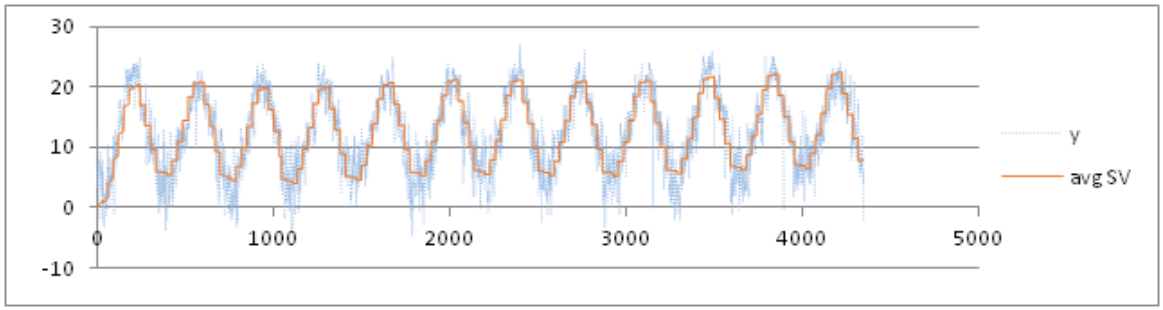
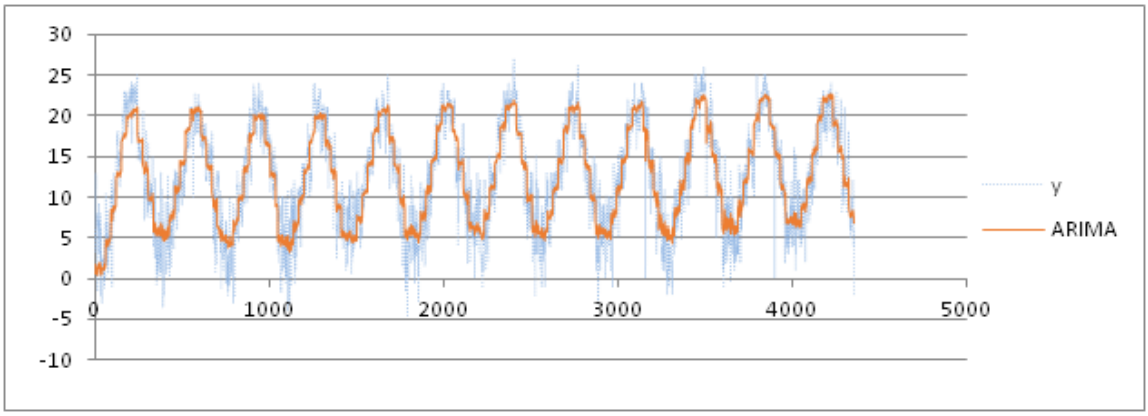


Figure 8. Trend of min temperature under a best fit polynomial of nine degree.

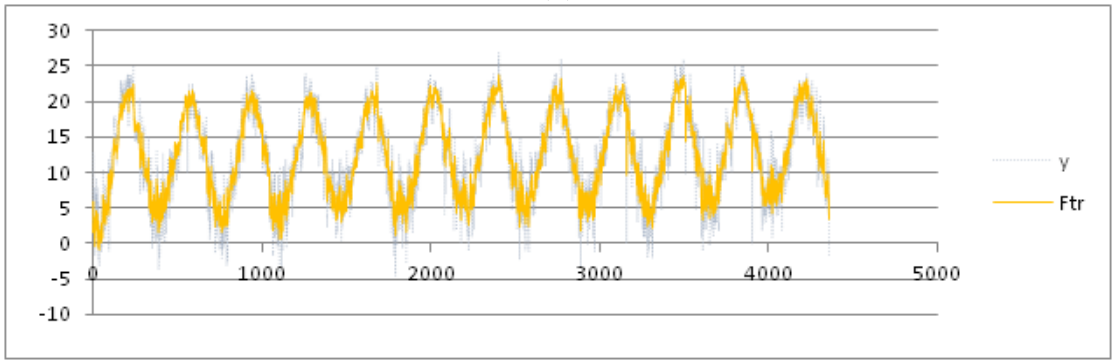
As we can observe in Figure 8, the seasonality of the data seems to be more evident than in the two previous examples. In Figure 9a–d, we plot the final results using the four forecasting methods.



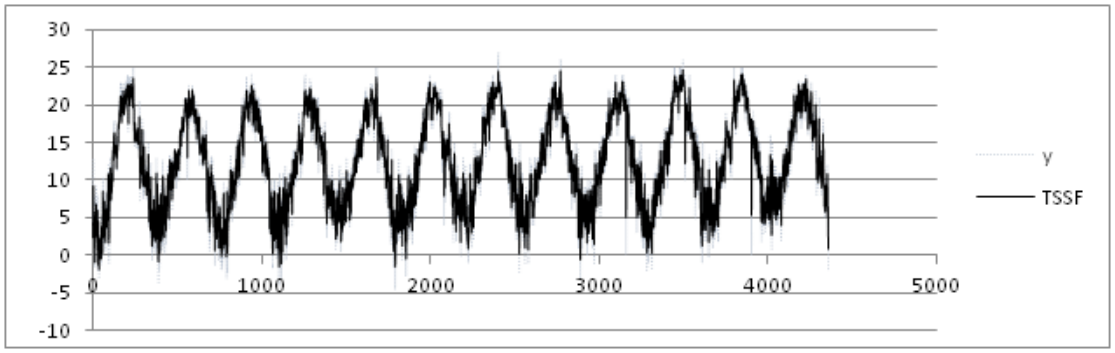
(**a**)



(**b**)



(**c**)



(**d**)

Figure 9. (**a**) Results obtained for the variation of the min temperature by using the avgSV method; (**b**) results obtained for the variation of the min temperature by using the ARIMA method; (**c**) results obtained for the variation of the min temperature by using the F-transforms method; and (**d**) results obtained for the variation of the min temperature by using the TSSF method.

In Table 3, we show the indices obtained using the four methods. MAPE is not measurable because there are measures of min temperature equal to 0.00. The best results for the min temperature are obtained using the TSSF, with a MADMEAN of 5.26.

Table 3. RMSE, MAPE, MAD and MADMEAN indices for the min temperature.

Forecasting Method	RMSE	MAPE	MAD	MADMEAN
avgSV	2.97	-	2.34	18.66
ARIMA	1.25		0.97	7.46
F-transforms	1.56	-	1.09	8.73
TSSF	0.87	-	0.66	5.26

The results in Table 3 show that the TSSF is more efficient when the seasonality of the data is more pointed, such as the mean temperature time series. To test the reliability of the forecasting results obtained using the four forecasting methods, we have considered test datasets containing the measure of the analyzed parameter in a next time period and calculating the RMSE obtained with respect to the forecasted values. For the mean and max temperature parameters, we use a test dataset formed by the mean and max daily temperatures measured in the temporal interval 17 August 2015–31 August 2015. For the min temperature parameter, we use a test dataset formed by the min daily temperatures measured in the temporal interval 1 January 2015–31 August 2015. We apply the test datasets, measuring the RMSE obtained using the four methods. In addition, we also show the RMSE calculated by applying the SVM and ADANN algorithms proposed in [23], as shown in Table 4

Table 4. RMSE measured by using six different methods for the mean, max, min temperature.

Parameter	Training Dataset Dimension	Test Dataset Dimension	Seasonal Period	RMSE					
				avgSV	ARIMA	F-transf.	TSSF	SVM	ADANN
Temp. mean	757	15	Week	1.83	1.58	1.62	1.39	0.88	0.87
Temp. max	757	15	Week	2.20	1.92	1.99	1.68	1.31	1.26
Temp. min	4354	212	Month	2.90	1.27	1.54	0.94	0.96	1.01

These results confirm that the performances obtained using the TSSF algorithm are better than the ones obtained using the avgSV, ARIMA and F-transform methods: the most reliable results are obtained for the third parameter, in which the seasonality is more regular. In addition, the performance obtained by applying SVM and ADANN algorithms to the first two time series improve those obtained by adopting the TSSF algorithm, but, in the third case, where the time series has best regularity, RMSE obtained using the TSSF, SVM and ADANN algorithms are indeed comparable.

We repeated these comparison tests by using other climatic datasets. For brevity, we show the results obtained by analyzing the trend of the daily mean temperature measured at the weather station *Chiavari Caperana* in the Italian Region Liguria; the data can be downloaded at the website (http://www.cartografiarl.regione.liguria.it/SiraQualMeteo/script/PubAccessoDatiMeteo.asp) of the webpage "Ambiente in Liguria".

The analyzed data concern the measures (°C) of the daily mean temperature during the period 1 January 2006–31 December 2016. The month is considered as the seasonal period; the dataset is partitioned into 12 seasonal subsets corresponding to the 12 months of a year. After applying the pre-processing phase, we set the threshold of the MADMEAN index to the value 5.2.

In Figure 10, we show the data and the trend obtained using a best fit polynomial of nine degree (see Appendix). In Figure 11a–d, the results obtained using the four methods are plotted.

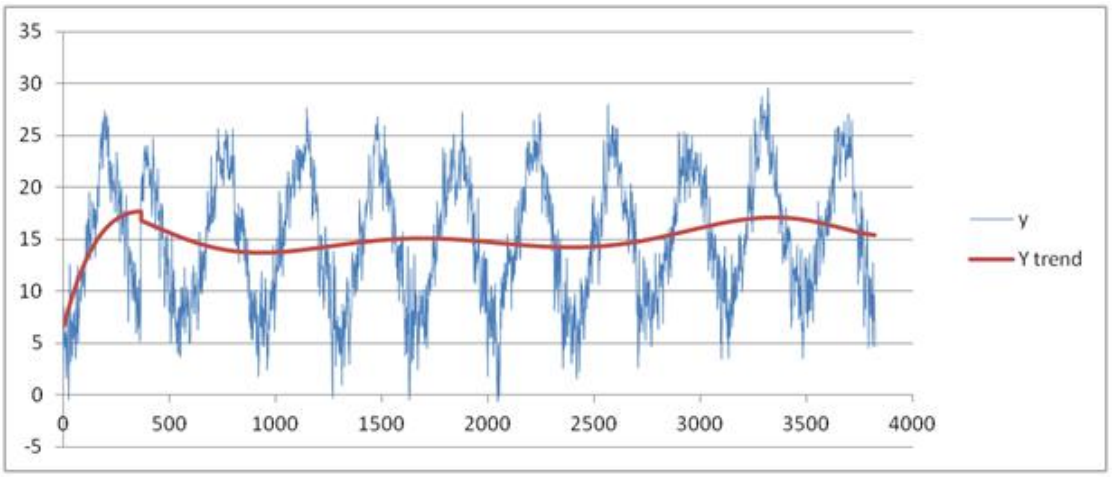
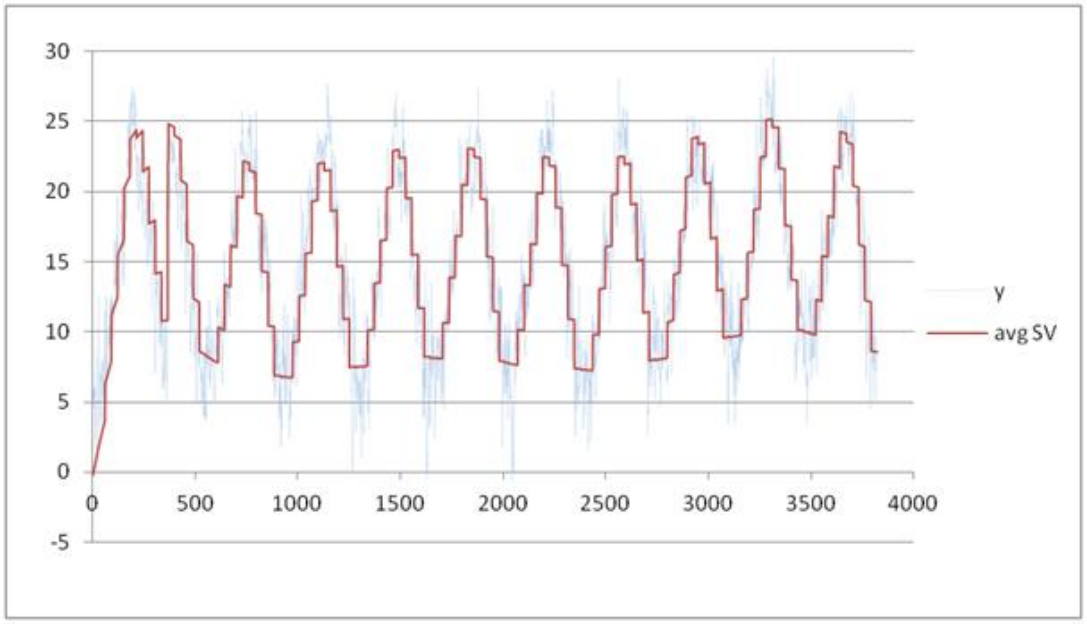
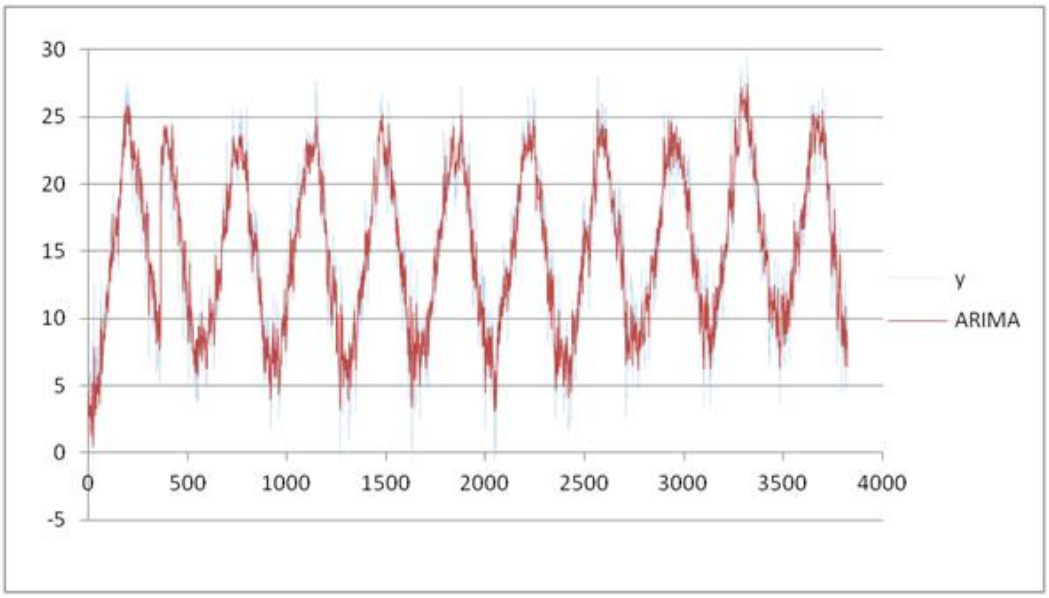


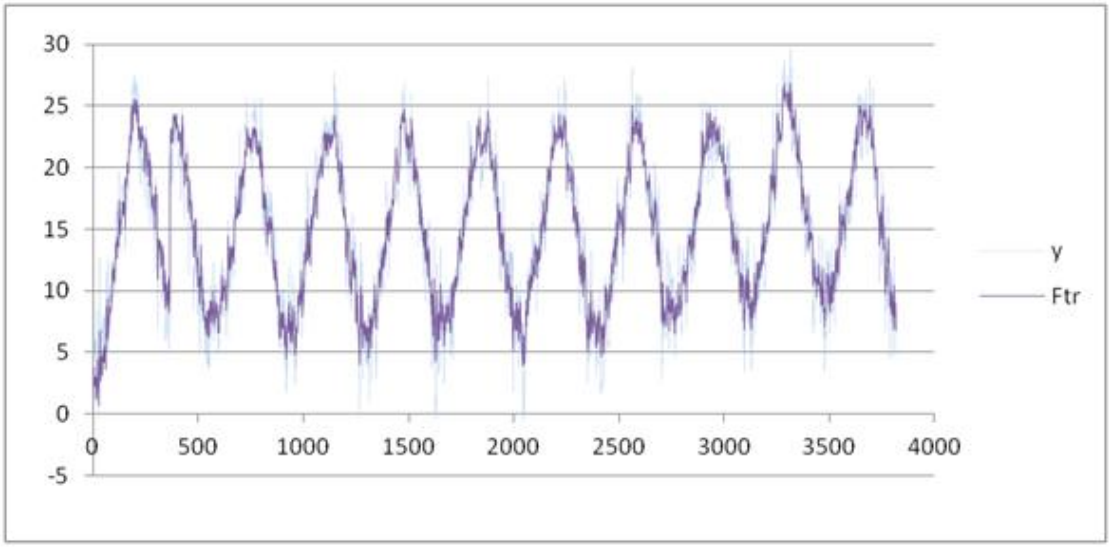
Figure 10. Trend of mean temperature under a best fit polynomial of nine degree Chiavari-Caperana weather station dataset.



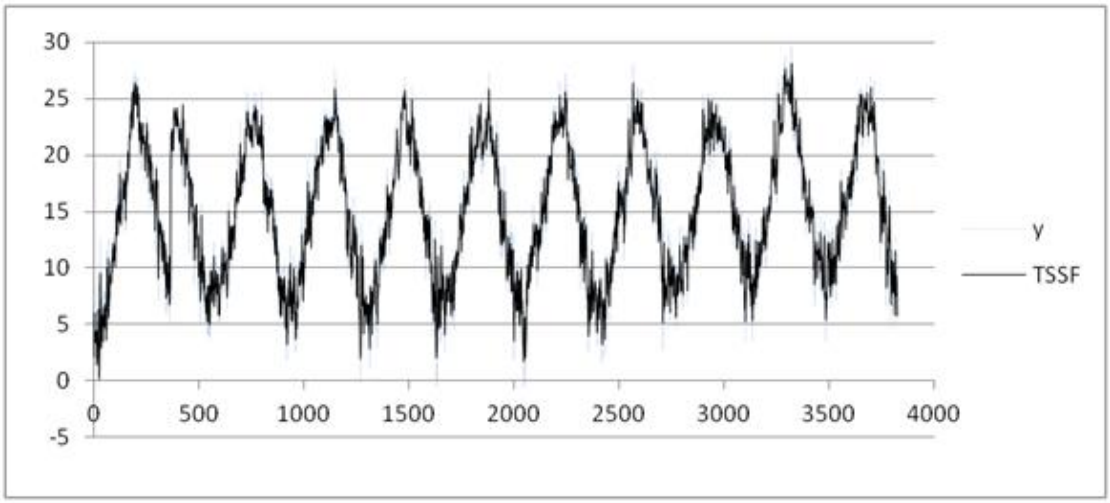
(**a**)



(**b**)



(**c**)



(**d**)

Figure 11. (**a**) Results obtained for the variation of the mean temperature by using the avgSV method; (**b**) results obtained for the variation of the mean temperature by using the ARIMA method; (**c**) results obtained for the variation of the mean temperature by using the F-transforms method; and (**d**) results obtained for the variation of the mean temperature by using the TSSF method.

In Table 5, we show the four indexes obtained using the four methods. The best results for the mean temperature are obtained using TSSF, with a MADMEAN of 3.83.

Table 5. RMSE, MAPE, MAD, MADMEAN indices for the mean temperature (Chiavari-Caperana weather station).

Forecasting Method	RMSE	MAPE	MAD	MADMEAN
avgSV	2.87	27.74%	2.14	16.19
ARIMA	1.13	12.37%	0.89	5.95
F-transforms	1.38	15.18%	1.10	7.30
TSSF	0.73	7.96%	0.58	3.83

The results in Table 5 confirm the ones obtained for the min temperature parameters in the weather dataset of Naples (Table 3): the TSSF gives better results than the ARIMA and F-transform algorithms, and is more efficient when the seasonality of the data is more regular. To test the reliability of the results obtained in Table 5, we have considered a test dataset containing the measure of the daily mean temperature during the period 1 January 2017–19 March 2017 and

calculating the RMSE obtained with respect to the forecasted values. In Table 6, we show the RMSE measured in the ix methods for each parameter.

Table 6. RMSE in six methods for the mean temperature (Chiavari-Caperana weather station).

Parameter	Training Dataset Dimension	Test Dataset Dimension	Seasonal Period	RMSE					
				avgSV	**ARIMA**	**F-transf.**	**TSSF**	**SVM**	**ADANN**
Temp. mean	3822	78	Month	2.85	1.15	1.40	0.85	0.88	0.87

The results in Table 6 confirm that the performances obtained using the TSSF algorithm are better than the ones obtained using the avgSV, ARIMA and F-transform methods. Moreover, the performance of the TSSF algorithm is comparable with the ones obtained using the SVM and ADANN algorithms, as the time series shows a regular seasonality

In Table 7, we show the tests results obtained for the mean temperature using the seasonal datasets of the daily mean temperature measured by other stations in the district of Genova; the data were downloaded from the Liguria Region webpage "Ambiente in Liguria". In each experiment, the season parameter is given by the month of the year.

Table 7. RMSE in six methods for the mean temperature in various stations in Genova district (Italy).

Station	RMSE					
	avgSV	**ARIMA**	**F-transf.**	**TSSF**	**SVM**	**ADANN**
Alpe Gorreto	2.98	1.20	1.49	0.84	0.81	0.83
Campo Ligure	2.74	1.09	1.34	0.76	0.71	0.76
Barbagelata	3.25	1.30	1.57	0.89	0.84	0.90
Camogli	3.39	1.38	1.68	0.95	0.88	0.86
Campo ligure	3.02	1.20	1.49	0.83	0.77	0.79
Carlasco	2.91	1.15	1.42	0.80	0.77	0.76
Chiavari	2.78	1.12	1.39	0.78	0.73	0.77
Genova Bolzaneto	2.95	1.16	1.41	0.81	0.77	0.75
Genova Pegli	3.34	1.29	1.64	0.94	0.89	0.88
Panesi	3.20	1.29	1.56	0.87	0.84	0.83
Rapallo	2.71	1.08	1.33	0.75	0.78	0.84
Rovegno	2.94	1.18	1.45	0.82	0.82	0.80
Tigliolo	3.06	1.24	1.52	0.85	0.80	0.85
Viganego	3.17	1.28	1.57	0.88	0.82	0.83

Table 7 confirms the results of Table 6. The performances of the TSSF algorithm are better than the ones obtained using the avgSV, ARIMA and F-trasform methods and comparable with the ones obtained with the SVM and ADANN algorithms: the reason for this is that none of the time series in Table 7 has significant irregular variations.

6. Conclusions

We present a new method based on F-transforms for seasonal forecasting. The goal was to adapt the F-transform based forecasting method in [29] for the analysis of seasonal times series, improving its performances with respect to other known methods. Usually, ARIMA model has lower performance than models based on ANN and SVM, but these models are complex to manage: for instance, the choice of the input parameters often affects the reliability of the final results. Our results show that the TFSS method improves the performances of ARIMA, avgSV and F-transform forecasting methods. In addition, for time series in which the seasonality of the data is more regular,

the performance obtained using the TFSS algorithm is comparable with the one obtained with SVM and ADANN models.

In future works, we shall try to improve TSSF for the management of time series that also contain irregular variations, comparing it with other soft computing algorithms.

Author Contributions: The authors have contributed equally to write this paper.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Schema of the TSSF prediction algorithm

(1) Calculate the trend using a polynomial fitting
(2) Subtract to the data the trend value obtaining a new dataset
(3) Partition the dataset into subsets: each data subset contains the measured data in a season.
(4) FOR each seasonal subset
(5) n:= 3
(6) Set the fuzzy partition (8)
(7) IF the subset is sufficiently dense with respect to the fuzzy partition THEN
(8) Calculate the direct F-transform (15)
(9) Calculate the inverse F-transform (16)
(10) Calculate the MADMEAN index
(11) IF MADMEAN > Threshold THEN
(12) n:= n + 1
(13) Go to 6)
(14) END IF
(15) END IF
(16) NEXT
(17) STORE the direct F-transform components
(18) END

Appendix B

Coefficients of the polynomial of 9th degree used for finding the best fit to training data of the mean temperature during the period 1 July 2003–16 August 2015.

a9 = 1.34428E-33
a8 = −2.82732E-28
a7 = 2.32945E-23
a6 = −9.06821E-19
a5 = 1.45202E-14
a4 = 0
a3 = 0
a2 = −0.066427015
a1 = 0
a0 = 17,699,903.37

Coefficients of the polynomial of 9th degree used for finding the best fit to training data of the max temperature during the period 1 July 2003–16 August 2015.

a9 = 1.45368E-33
a8 = −3.06035E-28
a7 = 2.52386E-23

a6 = −9.83437E-19
a5 = 1.57619E-14
a4 = 0
a3 = 0
a2 = −0.072310129
a1 = 0
a0 = 19,302,936.01

Coefficients of the polynomial of 9th degree used for finding the best fit to training data of the min temperature during the period 1 January 2003–31 December 2015.

a9 = 4.40678E-32
a8 = −1.11212E-26
a7 = 1.14529E-21
a6 = −5.94437E-17
a5 = 1.42772E-12
a4 = 0
a3 = −0.000762063
a2 = 13.0645932
a1 = 0
a0 = −1160591164

Coefficients of the polynomial of 9th degree used for finding the best fit to training data of the mean temperature measured at the weather station Chiavari-Caperana during the period 1 January 2006–31 December 2016.

a9 = 1.53589E-31
a8 = −4.0454E-26
a7 = 4.41907E-21
a6 = −2.5338E-16
a5 = 7.78147E-12
a4 = −1.06189E-07
a3 = 0
a2 = 0
a1 = 519825.0932
a0 = −7088111243

References

1. Abraham, B.; Ledolter, J. *Statistical Methods for Forecasting*; John Wiley & Sons: New York, NY, USA, 1983; p. 445, ISBN 978-0-471-86764-3.
2. *Principles of Forecasting: A Handbook for Researchers and Practitioners*; Armstrong, J.S. Ed.; Springer: Berlin, Germany, 2001; p. 841.
3. Box, G.E.P.; Jenkins, G.M.; Reinsel, G.C. *Time Series Analysis: Forecasting and Control*, 5th ed.; Prentice Hall: Englewood Cliffs, NJ, USA, 2015; p. 712, ISBN 978-1-118-67502-1.
4. Chatfield, C. *Time Series Forecasting*; Chapman & Hall/CRC: Boca Raton, FL, USA, 2001; p. 267, ISBN 1-58488-063-5.
5. Hymdam, R.J.; Athanasopoulos, G. *Forecasting Principles and Practice*; OText Publisher: Melbourne, Australia, 2013; p. 291, ISBN 9780987507105.
6. Lu, W.Z.; Wang, W.J. Potential assessment of the Support Vector Machine method in forecasting ambient air pollutant trends. *Chemosphere* **2005**, *59*, 693–701, doi:10.1016/j.chemosphere.2004.10.032.
7. Makridakis, S.G.; Wheelwright, S.C.; Hyndman, R.J. *Forecasting: Methods and Applications*, 3rd ed.; J. Wiley & Sons: New York, NY, USA, 1998; p. 656, ISBN 978-0-471-53233-0.
8. Zhang, G.P.; Qi, M. Neural network forecasting for seasonal and trend time series. *Eur. J. Oper. Res.* **2005**, *160*, 501–514, doi:10.1016/j.ejor.2003.08.037.
9. Pankratz, A. *Forecasting with Dynamic Regression Models*; Wiley: Hoboken, NJ, USA, 2012; p. 392.
10. Miller, K.; Smola, A.J.; Ratsch, G.; Scholkopf, B.; Kohlmorgen, J.; Vapnik, V. Predicting time series with support vector machines. In Proceedings of the 7th International Conference on Artificial Neural Networks, Lecture

Notes in Computer Sciences, Switzerland, 8–10 October 1997; Springer: Berlin, Germany, 1998; Volume 1327, pp. 999–1004.
11. Pai, P.F.; Lin, K.P.; Lin, C.S.; Chang, P.T. Time series forecasting by a seasonal support vector regression model. *Exp. Syst. Appl.* **2010**, *37*, 4261–4265, doi:10.1016/j.eswa.2009.11.076.
12. Mohandes, M.A.; Halawani, T.O.; Rehman, S.; Hussain, A.A. Support vector machines for wind speed prediction. *Renew. Energy* **2004**, *29*, 939–947, doi:10.1016/j.renene.2003.11.009.
13. Ittig, P.T. A seasonal index for business. *Decis. Sci.* **1997**, *28*, 335–355, doi:10.1111/j.1540-5915.1997.tb01314.x.
14. Hong, W.C.; Pai, P.F. Potential assessment of the support vector regression technique in rainfall forecasting. *Water Resour. Manag.* **2007**, *21*, 495–513, doi:10.1007/s11269-006-9026-2.
15. Crone, S.F.; Hibon, M.; Nikolopoulos, K. Advances in forecasting with neural networks? Empirical evidence from the NN3 competition on time series prediction. *Int. J. Forecast.* **2011**, *27*, 635–660, doi:10.1016/j.ijforecast.2011.04.001.
16. Hamzacebi, C. Improving artificial neural networks performance in seasonal time series forecasting. *Inf. Sci.* **2008**, *178*, 4550–4559, doi:10.1016/j.ins.2008.07.024.
17. Zhang, G.P.; Kline, D.M. Quarterly time-series forecasting with neural networks. *IEEE Trans. Neural Netw.* **2007**, *18*, 1800–1814, doi:10.1109/TNN.2007.896859.
18. Zhang, G.; Patuwo, B.E.; Hu, M.Y. Forecasting with artificial neural networks: The state of the art. *Int. J. Forecast.* **1998**, *14*, 35–62, doi:10.1016/S0169-2070(97)00044-7.
19. Zhang, G.; Zhang, G.P. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* **2003**, *50*, 159–175, doi:10.1016/S0925-2312(01)00702-0.
20. Faraway, J.; Chatfield, C. Time series forecasting with neural networks: A comparative study using the airline data. *J. Royal Stat. Soc. Ser. C Appl. Stat.* **1998**, *47*, 231–250, doi:10.1111/1467-9876.00109.
21. Kihoro, J.M.; Otieno, R.O.; Wafula, C. Seasonal time series forecasting: A comparative study of ARIMA and ANN models. *Afr. J. Sci. Technol.* **2004**, *5*, 41–49, doi:10.4314/ajst.v5i2.15330.
22. Khandelwal, I.; Adhikari, R.; Verma, G. Time series forecasting using hybrid ARIMA and ANN models based on DWT decomposition. *Procedia Comput. Sci.* **2015**, *48*, 173–179, doi:10.1016/j.procs.2015.04.167.
23. Štepnicka, M.; Cortez, P.; Peralta Donate, J.; Štepnickova, L. Forecasting seasonal time series with computational intelligence: On recent methods and the potential of their combinations, *Exp. Syst. Appl.* **2013**, *40*, 1981–1992, doi:10.1016/j.eswa.2012.10.001.
24. Kumar, A.; Kumar, D.; Jarial, S.K. A hybrid clustering method based on improved artificial bee colony and fuzzy C-Means algorithm. *Int. J. Artif. Intelli*. **2017**, *15*, 40–60.
25. Medina J.; Ojeda-Aciego, M. Multi-adjoint t-concept lattices. *Inf. Sci.* **2010**, *180*, 712–725, doi:10.1016/j.ins.2009.11.018.
26. Nowaková, J.; Prílepok, M.; Snášel, V. Medical image retrieval using vector quantization and fuzzy S-tree. *J. Med. Syst.* **2017**, *41*, 1–16, doi:10.1007/s1091.
27. Pozna, C.; Minculete, N.; Precup, R.; Kòczy, L.T.; Ballagi, A. Signatures: Definitions, operators and applications to fuzzy modeling. *Fuzzy Sets Syst.* **2012**, *201*, 86–104, doi:10.1016/j.fss.2011.12.016.
28. Perfilieva, I. Fuzzy transforms: Theory and applications. *Fuzzy Sets Syst.* **2006**, *157*, 993–1023, doi:10.1016/j.fss.2005.11.012.
29. Di Martino, F.; Loia, V.; Sessa, S. Fuzzy transforms method in prediction data analysis, *Fuzzy Sets Syst.* **2011**, *180*, 146–163, doi:10.1016/j.fss.2010.11.009.
30. Wang, L.X. Mendel, J.M. Generating fuzzy rules by learning from examples. *IEEE Trans. Syst. Man Cybern.* **1992**, *22*, 1414–1427, doi:10.1109/21.199466.
31. Di Martino, F.; Loia, V.; Perfilieva, I.; Sessa, S. An image coding/decoding method based on direct and inverse fuzzy transforms. *Int. J. Approx. Reason.* **2008**, *48*, 110–131, doi:10.1016/j.ijar.2007.06.008.
32. Di Martino, F.; Loia, V.; Sessa, S. Fuzzy transforms method and attribute dependency in data analysis. *Inf. Sci.* **2010**, *180*, 493–505, doi:10.1016/j.ins.2009.10.012.
33. Novák, V.; Pavliska, V.; Perfilieva, I., Štepnicka, M. F-transform and fuzzy natural logic in Time Series Analysis. In Proceedings of the 8th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT) Milan, Italy, 11–13 September 2013; Atlantic Press: Pleasantville, NJ, USA, pp. 40–47.
34. Kolassa, W.; Schutz, W. Advantages of the MADMEAN ratio over the MAPE. *Foresight* **2007**, *6*, 40–43.
35. Goodrich, R.L. The Forecast Pro methodology. *Int. J. Forecast.* **2000**, *16*, 533–535, doi:10.1016/S0169-2070(00)00086-8.