A new hybrid enhanced local linear neuro-fuzzy model based on the optimized singular spectrum analysis and its application for nonlinear and chaotic time series forecasting

Majid Abdollahzade a,*, Arash Miranian b, Hossein Hassani c,d, Hossein Iranmanesh e

a Department of Mechanical Engineering, Pardis Branch, Islamic Azad University, Pardis New City, Tehran, Iran
b Department of Electrical Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran
c The Statistical Research Centre, Executive Business Centre, Bournemouth University, UK
d Institute for International Energy Studies (IIES), No. 65, Sayeh St., Vali-e-asr Ave., Tehran 1967743711, Iran
e Department of Industrial Engineering, University of Tehran, Tehran, Iran

ABSTRACT

This paper develops a hybrid method for nonlinear and chaotic time series forecasting based on a local linear neuro-fuzzy model (LLNF) and optimized singular spectrum analysis (OSSA), termed OSSA–LLNF. Nonlinear and chaotic time series often exhibit complex behaviour and dynamics, turning their forecasting (particularly in multi-step ahead horizons) into a difficult task. In this paper, SSA is utilized for data processing, resulting in the elimination of noisy components and improvement of forecasting performance. The SSA parameters are fine-tuned using the particle swarm optimization algorithm. Then, the processed time series is modelled and forecasted via the LLNF model. The proposed OSSA–LLNF model is applied to forecast several well-known time series with different structures and characteristics. The comparison of the obtained results with those of several old and recently developed methods indicates the superiority and promising performance of the proposed OSSA–LLNF approach.

© 2014 Published by Elsevier Inc.

1. Introduction

1.1. Motivation and aims

Modelling and prediction of nonlinear and chaotic time series have always been a challenge to the research community. Such time series, which are commonly encountered in real world phenomena e.g. physics, biology, medicine, and engineering, usually exhibit seemingly unpredictable behaviour [28]. They are composed of various components such as trend, seasonality, impulse, steps, model exchange and other uncontrolled features, e.g. non-stationary behaviour, making traditional mathematical and statistical methods unsuitable for their modelling and prediction [33]. Hence, due to the wide application of time series analysis in fields such as engineering, economics and medicine [6], and the increasing importance of accurate modelling and prediction, this problem continues to remain open for further development of more sophisticated modelling approaches.

* Corresponding author at: No. 65, Sayeh St., Vali-e-asr Ave., Tehran 19395-4757, Iran. Tel.: +98 9124118422; fax: +98 21 22054853.
E-mail address: m.abdollahzade@pardisiau.ac.ir (M. Abdollahzade).

http://dx.doi.org/10.1016/j.ins.2014.09.002
0020-0255/© 2014 Published by Elsevier Inc.
1.2. Literature review

From a general perspective, time series prediction techniques can be categorized into two broad categories, namely statistical, e.g. auto-regressive (AR) and auto-regressive integrated moving average (ARIMA) models, and computational intelligence (CI) approaches, such as neural networks and fuzzy systems. An overview of time series prediction techniques, with the main focus on CI-based approaches, is presented in Table 1.

The statistical time series modelling approaches, among them the well-known AR, ARIMA, and generalized autoregressive conditional heteroskedasticity (GARCH) models employ some prominent features of time series, such as trend and seasonality to build the prediction model [28]. However, they often assume several restrictive assumptions, such as linearity, stationarity and normality, which are seldom satisfied in the case of nonlinear and chaotic time series. Hence, there is an ongoing need for the development of more dependable approaches which are able to model the complex behavior of nonlinear and chaotic time series.

Through the last two decades, CI techniques have been proposed as appropriate alternatives for identification, modelling, and prediction in complex dynamic systems and processes [11–44]. Artificial neural networks, neuro-fuzzy models and support vector machines are three well-known CI approaches widely used for time series prediction.

Among the available CI approaches, artificial neural networks (ANNs) and neuro-fuzzy models, as synergistic integration of ANNs and Takagi–Sugeno fuzzy inference systems, have gained a strong popularity in the area of nonlinear time series prediction [68,36,5,52,15]. These CI approaches are data-driven modelling techniques which construct a mapping between the past values of a time series and its future. They usually do not require any specific assumption about the nature of the time series, its behaviour, stationarity and complexity. With CI approaches, the prediction model is built upon the available historical data of the time series.

ANNs were originally motivated by the biological structures in the brains of humans and animals, and are extremely powerful for tasks such as information processing, learning, function approximation and prediction [47,32,17]. There are several variants of ANNs, such as multi-layered perceptron (MLP) neural networks, radial basis function (RBF) neural networks and recurrent neural networks. For instance, in [36] a RBF neural network was developed for chaotic and noisy time series prediction. In [67], Yan proposed a generalized regression neural network (GRNN) for effective modelling of the large-scale business time series, and modular neural networks were developed for time series forecasting in [41,42].

The neuro-fuzzy (NF) models, which inherit the learning and parallel processing capabilities of ANNs and incorporation of a priori knowledge in fuzzy systems, are data-driven fuzzy inference systems not solely designed by expert knowledge but instead partly learned from data [47]. Owing to these capabilities, the NF models have been proved as high-performance approaches for identification and prediction of complex nonlinear processes, e.g. time series [70,3,45,1].

In [70], a multi-input–multi-output-adaptive-network-based fuzzy inference system (MANFIS) was developed for chaotic time series prediction and applied to Mackey–Glass chaotic time series and a Duffing forced-oscillation system. Bodanyski and Vynokurova proposed a wavelet-neuro-fuzzy system with a five-layered structure for chaotic time series identification [3]. In this work, wavelets were used as membership functions in the antecedent layer, and the adaptive multidimensional wavelets as activation functions in the consequent layer [3]. In another recent study, Miranian and Abdollahzade [45] combined the powerful least-squares support vector machines with neuro-fuzzy models and applied their developed model to predict several different chaotic and nonlinear time series.

Time series modelling and prediction by ensemble NF models has been also pursued by researchers in [50,43,71]. For instance, integration of type-2 fuzzy systems in ensemble neural networks for time series prediction has been proposed in [50]. In [51], a genetic algorithm-based optimization method was proposed for ensemble neural network models with fuzzy aggregation of responses for complex time series prediction. In [14], Gaxiola et al. employed a genetic algorithm to optimize the three neural networks in an ensemble mode. Their method was applied to time series prediction as well as

<table>
<thead>
<tr>
<th>Authors/years</th>
<th>Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Melin et al. (2006) [16]</td>
<td>Neural trees</td>
</tr>
<tr>
<td>Lin et al. (2009) [55]</td>
<td>Self-evolving type-2 fuzzy neural network</td>
</tr>
<tr>
<td>Melin et al. (2012) [22]</td>
<td>Ensemble neural network</td>
</tr>
<tr>
<td>Miranian and Abdollahzade (2013) [18]</td>
<td>LSSVM-based neuro fuzzy model</td>
</tr>
<tr>
<td>Gaxiola et al. (2013) [21]</td>
<td>Type-2 fuzzy neural network</td>
</tr>
<tr>
<td>Pulido and Melin (2014) [67]</td>
<td>Ensemble neural networks with fuzzy aggregation and PSO</td>
</tr>
</tbody>
</table>
pattern recognition. In another work, published by Melin et al. [43], an architecture for ensembles of adaptive network based fuzzy inference system (ANFIS) was developed and employed to predict the Mackey–Glass, Dow Jones and Mexican stock exchange time series. In this work, integrator by average and the integrator by weighted average were used for the integration of the ensembles of ANFIS.

Local linear neuro-fuzzy (LLNF) models, as an appealing class of NF models, have been applied to various modeling and time series prediction applications in recent years [47,1,24,20]. As an example, Abdollahzade et al. [1] employed a LLNF model for the practical problem of electricity price forecasting in several important power markets. Long-term energy demand forecasting by hybrid LLNF models have been reported in [24].

The LLNF models are constructed based on a fast and efficient divide-and-conquer strategy which breaks down a difficult modelling problem into a set of smaller and simpler sub-problems. The LLNF models are able to describe different operating regimes of a process by different local linear models (LLMs) and their associated validity functions. In the structure of the LLNF model, the contribution of each LLM to the final output is determined by its locally active validity function. In other words, the validity functions set the validity region (or operating regime) of their corresponding LLMs. This provides an accurate representation of the process by weighted aggregation of the LLMs’ outputs.

On the other hand, apart from the time series modelling approach, extraction of main features of time series, removal of noise and unpredictable components in a pre-processing stage can remarkably enhance prediction performance and accuracy. In fact, filtering of noisy and almost unpredictable components of nonlinear and chaotic time series by data processing techniques often leads to a series which is less complex and more predictable [55].

Singular Spectrum Analysis (SSA), as a relatively novel time series processing technique, incorporates the elements of classical time series analysis, multivariate statistics, multivariate geometry, dynamical systems and signal processing [16,21]. SSA works well for linear and nonlinear, stationary and non-stationary time series with different features and structure. It can efficiently identify and extract the trend, cyclic, quasi-cyclic and noise components of a time series [21] and then reconstruct a new series by elimination of noise components and hence improve prediction performance (this implies employing the SSA for time series filtering in the pre-processing stage). Moreover, if the parameters of SSA are optimally selected (optimal filtering of the time series) this will lead to further improvement in the time series prediction performance.

1.3. Contribution and highlights

In this paper, a combination of the optimized SSA and LLNF model is proposed as a powerful approach for the prediction of the nonlinear and chaotic time series. The LLNF model itself can efficiently construct a mapping which accurately describes the behavior of the time series. However, in case of complex, highly nonlinear and chaotic time series, the LLNF model may fail to completely capture the dynamics of the time series. SSA was originally developed to extract information from chaotic time series to produce an insight into the unknown dynamics of the time series. Hence preprocessing of the time series using SSA removes high-frequency noisy components of the time series and extracts the main features. It is worth noting that the LLNF model can build a more accurate mapping for the filtered time series since the high-frequency component of the original time series (which does not contain useful information) has been removed during the preprocessing stage by the SSA. Therefore, there is a strong motivation to combine SSA and LLNF models for nonlinear and chaotic time series prediction applications.

The whole procedure of the SSA technique depends upon two basic, yet very important, choices: (i) the window length L and (ii) the number of eigenvalues r, that one needs to select for reconstructing noise-free series from a noisy series of length N. Large values of L allow longer periodic oscillations to be resolved, but choosing a value large for L leaves too few observations from which to estimate the covariance matrix of the L variables. Although considerable attempts and various techniques have been considered for selecting the proper value of L, there are no sufficient algebraic and theoretical materials for choosing L [22]. No considerable attempts have also been made for the optimal selection of the SSA choice r. In this paper we propose a Particle Swarm Optimization (PSO) technique to fully exploit the signal processing capabilities of the SSA technique by intelligent detection of values relating to L and r. The proposed hybrid approach in this paper is established based on the PSO optimized SSA, as the time series preprocessing technique, and the LLNF network, as the forecasting model. The hybrid approach, termed OSSA–LLNF is therefore used for forecasting purposes.

In brief, the contributions and highlights of this paper can be summarized as follows.

1. Developing a hybrid approach based on the SSA data processing technique and LLNF model for time series forecasting.
2. Optimal intelligent selection of SSA’s parameters based on the particle swarm optimization algorithm.
3. Applying the proposed hybrid approach to several well-known benchmark time series and comparison of the results with recent and old studies.

1.4. Paper organization

This paper is organized as follows. The local linear neuro-fuzzy model and LOLIMOT learning algorithm are described in Section 2. The fundamentals of SSA are presented in Section 3. The proposed forecasting framework and the optimization procedure for SSA are outlined in Section 4. Simulation results obtained for four well-known time series, with different features, namely Box–Jenkins, Mackey–Glass, sunspot and the US accidental death series are reported in Section 5. Box–Jenkins
is a time series obtained from observations during a combustion process. The Mackey–Glass time series is a hand-designed benchmark time series exhibiting chaotic behavior and sunspot numbers is a real world time series which is of importance to space mission centers. The US accidental death series is a monthly data which shows strong seasonal patterns. Finally, conclusions are drawn in Section 6.

2. Local linear neuro-fuzzy model

The local linear neuro-fuzzy approach, based on the incremental tree and learning algorithm, starts from an initial optimal linear model and increases the complexity of the model as long as improvements occur. The local linear model tree (LOLIMOT) learning algorithm employed in this paper applies axis-orthogonal splits to divide the original input domain into sub-domains, each identified by a local linear model (LLM) and its associated validity function, which determines the validity region of the LLM. The LOLIMOT algorithm is fast and determines the optimal structure of the LLNF model heuristically without running time-consuming nonlinear optimization algorithms. It includes small number of adjustable parameters and does not require any initialization. While other approaches, such as nonlinear optimization techniques may be employed for training the LLNF model, they suffer from drawbacks such as being time-consuming (particularly in case of high-dimensional input space), requirement for initialization, large number of adjustable parameters and getting stuck in local minima [47].

The validity functions can be interpreted as the operating point dependent weighting factor, which determine the contribution of their associated LLMs to the final output. Thus, it is inferred that the LLNF models use a divide-and-conquer approach to solve a complex modelling problem by decomposing it into smaller and simpler sub-problems [47,1]. The general structure of the LLNF model for a $p$-dimensional input space and $M$ local linear models is illustrated in Fig. 1. Based on this Figure, each local linear model and its associated validity function provide a nonlinear neuron.

The global output of the LLNF model can be stated as follows,

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i \phi_i(u),$$

where $\hat{y}_i$ is the output of $LLM_i$, $\phi_i$ is the associated validity function, and $u = [u_1 \ u_2 \ldots \ u_p]^T$ is the input vector. The local output of each LLM is expressed below:

$$\hat{y}_i = \theta_{i0} + \theta_{i1}u_1 + \theta_{i2}u_2 + \cdots + \theta_{ip}u_p,$$

where $\theta_i = [\theta_{i0} \ldots \theta_{ip}]$ is the vector of parameters for $LLM_i$.

The validity functions must form a partition of unity, i.e. they must sum up to 1. This property ensures that contributions of all LLMs sum up to 100%. The validity functions are chosen as axis-orthogonal normalized Gaussian functions, expressed below,

$$\phi_i(u) = \frac{\mu_i(u)}{\sum_{j=1}^{M} \mu_j(u)}, \quad j = 1, \ldots, M,$$

$$\mu_i(u) = \exp \left( -\frac{1}{2} \left( \frac{(u_1 - c_{i1})^2}{\sigma_{i1}^2} + \cdots + \frac{(u_p - c_{ip})^2}{\sigma_{ip}^2} \right) \right),$$

where $c_{ij}$ and $\sigma_{ij}$ represent center coordinates and standard deviation of normalized Gaussian validity function associated with $i$-th local linear model.

![Fig. 1. Structure of the local linear model for a $p$-dimensional input vector.](image-url)
The LLNF model offers a transparent representation of the nonlinear system. It is worth noting that transparency is a measure of the human linguistic interpretability of the rules issued from the training of the neuro-fuzzy system. The LLNF model established based on the divide-and-conquer strategy divides a complex model into a set of sub-models which acts linearly and independently. Two types of parameters, i.e. rule consequent and rule premise parameters should be identified in the LLNF model. The rule consequent parameters \( \beta_i \), associated with local linear models, are estimated by linear least-square optimization procedure. However, both global and local least square estimations can be applied. In this paper, the former is employed since it considers the interaction between LLMs to a greater extent and therefore shows better performance [1].

For a LLNF model with \( M \) neurons and \( p \) inputs, the vector of linear parameters contains \( n = M(p + 1) \) elements, where each LLM has \( p \) parameters associated with its \( p \) inputs and one parameter associated with the bias term.

\[
\Theta = [\theta_{10} \theta_{11} \ldots \theta_{1p} \theta_{20} \theta_{21} \ldots \theta_{2p} \ldots \theta_{M0} \theta_{M1} \ldots \theta_{Mp}]^T. \tag{5}
\]

The corresponding regression matrix \( \mathbf{X}_{reg} \) for \( D \) measured data samples is,

\[
\mathbf{X}_{reg} = [\mathbf{X}_{reg,1} \mathbf{X}_{reg,2} \ldots \mathbf{X}_{reg,M}], \tag{6}
\]

where the regression sub-matrix \( \mathbf{X}_{reg,i} \) takes the following form,

\[
\begin{bmatrix}
\phi_i(u(1)) & u_1(u(1)) & \ldots & u_p(u(1)) \\
\phi_i(u(2)) & u_1(u(2)) & \ldots & u_p(u(2)) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_i(u(D)) & u_{1}(u(D)) & \ldots & u_p(u(D))
\end{bmatrix}.
\]

Hence,

\[
\hat{y} = \mathbf{X}_{reg} \cdot \hat{\Theta}; \quad \hat{\Theta} = \left( \mathbf{X}_{reg}^T \mathbf{X}_{reg} \right)^{-1} \mathbf{X}_{reg}^T \mathbf{y}. \tag{8}
\]

where \( \mathbf{y} = [y(1) y(2) \ldots y(D)]^T \) contains the \( D \) measured outputs.

The LOLIMOT algorithm is used for the estimation of the validity functions parameters. This algorithm is fast in convergence and computationally efficient and is therefore preferred over other optimization methods such as genetic algorithms and simulated annealing. LOLIMOT algorithm utilizes multivariate normalized axis-orthogonal Gaussian membership functions, as stated by (3) and (4).

In the LOLIMOT algorithm, the input space is divided into hyper-rectangles by axis-orthogonal cuts based on a tree structure. Each hyper-rectangle represents a LLM. In the original LOLIMOT algorithm, the LLM with worst performance is divided into two halves at each iteration. Thereafter, Gaussian membership functions are placed at the centre of the hyper-rectangles and standard deviations are selected proportional to the extension of hyper-rectangles (usually \( 1/3 \) of hyper-rectangle’s extension). Starting from an initial optimal linear model, the LOLIMOT algorithm adds nonlinear neurons provided that the regression sub-matrix \( \mathbf{X}_{reg} \) contains the \( D \) measured outputs.

A graphical representation of partitioning a two-dimensional input space by LOLMOT up to the first four iterations is illustrated in Fig. 2. The first two iterations in this figure are explained below,

1. In the first iteration, a single LLM is fitted over the whole input space (model A). For this model the validity function covers the whole input space and is equal to unity (\( \theta = 1 \)).
2. In the second iteration, the whole input space is divided horizontally and vertically into two equal hyper-rectangles, producing two different model structures, B and C. Then the validity functions are constructed for both horizontally- and vertically-partitioned input spaces. The Gaussian validity functions are placed at the centre of the hyper-rectangles and their standard deviations are selected proportional to the extension of hyper-rectangles.
3. Having estimated the validity functions, the local linear models are estimated using (8) for both model structures B and C.
4. Next, the overall performance of both model structures B and C are evaluated. The model structure with best performance is determined (in Fig. 2 model structure C) and considered for further refinement in the following iterations. Note that the LLNF model includes two LLMs with horizontally partitioned input space.

The procedure outlined above continues until desired level of validation error or model complexity is obtained.

3. Singular spectrum analysis

The main theme of SSA is decomposition of the original time series into a sum of series, so that each component can be identified as trend, periodic, quasi-periodic components or noise. The SSA technique is performed in four stages, which include embedding, singular value decomposition (SVD), grouping and diagonal averaging. The time series decomposition step is carried out through the first two stages, whilst the reconstruction step is performed by stages three and four [21]. A brief review of SSA is presented in the following (for more information, see [16]).
Consider time series $Y(t)$, available at $N$ point,

$$Y = [y_1, y_2, \ldots, y_N]^T.$$  

(9)

The embedding procedure involves transformation of the original time series into a trajectory matrix of $L$-dimensional vectors,

$$X = [X_1, X_2, \ldots, X_K],$$  

(10)

where

$$X_i = [y_i, y_{i+1}, \ldots, y_{i+L-1}]^T, 1 \leq i \leq K,$$

$$K = N - L + 1.$$  

(11)

Vectors $X_i$ are called $L$-lagged vectors and are supposed to describe the local state of the underlying time series. Note that $X$ is a Hankel matrix (i.e. the value of its $(i, j)$ entries only depend on sum $i + j$).

Now consider the matrix $XX^T$. The SVD is applied to produce $L$ eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_L$ and corresponding eigenvectors $U_1, U_2, \ldots, U_L$. The SVD of the trajectory matrix can be stated as,
\[ X = S_1 + \cdots + S_d, \] (12)

where \( S_i = \sqrt{\lambda_i} U_i V_i^T \) are elementary matrices (have rank one), \( d = \max \{ i, \text{such that } \lambda_i > 0 \} \) and \( V_i = X^T U_i / \lambda_i \). The vectors \( V_1, \ldots, V_d \) are called principle components (PCs) and the collection \((\lambda_i, U_i, V_i)\) is termed as the \( i \)-th eigentriple of matrix \( X \).

In the final stage, the time series is reconstructed by grouping and then diagonal averaging. The grouping step corresponds to splitting the elementary matrices \( S_i \) into several groups and summing the matrices within each group. Denote \( I = \{i_1, \ldots, i_p\} \) as a group of indices. Then the trajectory matrix \( X_i \), corresponding to group \( I \), is defined as \( X_i = S_{i_1} + S_{i_2} + \cdots + S_{i_p} \). In this paper, we consider splitting the set of indices \( J = \{1, \ldots, d\} \) into two disjoint subsets of \( I_1 = \{1, \ldots, r\} \) and \( I_2 = \{r+1, \ldots, d\} \). This leads to the following representation of the trajectory matrix,

\[ X = X_{I_1} + X_{I_2}. \] (13)

In representation (5), the trajectory matrix \( X \) is the sum of two resultant matrices. Next, via diagonal averaging each matrix is transformed into a time series by means of the following procedure. Let \( Z \) be a \((L \times n)\) matrix with elements \( z_{ij}, 1 \leq i \leq L, 1 \leq j \leq n \). Set \( L^* = \min(L, n) \), \( n^* = \max(L, n) \) and \( N = n + L - 1 \). Let \( z_{ij}^* = z_{ij} \) if \( L < n \); otherwise \( z_{ij}^* = z_{kj} \). Diagonal averaging transforms matrix \( Z \) into a series \( \{z_1, z_2, \ldots, z_N\} \) by means of the following equation,

\[
\bar{Y}_k = \begin{cases} 
\frac{1}{k+1} \sum_{q=1}^{k} z_{q,k-q+1}, & 1 \leq k \leq L^* \\
\frac{1}{L^*} \sum_{q=L^*+1}^{n^*} z_{q,k-q+1}, & L^* < k \leq K^* \\
\frac{1}{n-K^*+1} \sum_{q=K^*+1}^{N} z_{q,k-q+1}, & K^* < k \leq N
\end{cases}
\] (14)

Note that Eq. (6) relates to averaging the matrix elements along diagonals \( i + j = k + 1 \). The described diagonal averaging applied to matrix \( X_{I_1}, n = 1, 2 \) produces a time series \( Y \) of length \( N \). Hence, the original noisy times series \( \bar{Y} \) is decomposed into two series,

\[ \bar{Y} = Y_1 + Y_2. \] (15)

In this paper, we consider \( Y_2 \) as a noisy series and the original noisy time series is reconstructed by the first \( r \) eigentriples, i.e.

\[ \bar{Y} \sim Y_1. \] (16)

4. Structure of the proposed forecasting framework

The proper selection of the window length \( L \) as well as number of eigentriples, \( r \), used for time series reconstruction has a crucial effect on the de-noising performance of SSA. In this paper, we apply a heuristic and intelligent search algorithm, i.e. particle swarm optimization, for fine tuning SSA choices. PSO algorithms are fast, computationally efficient and easy to code search techniques established based on the constructive cooperation between the particles [34,13].

In PSO, particles flow in a multi-dimensional search space and the position of each particle is tuned based on the experiences gained by him and his neighbours. In this paper, we adopt a global best (gbest) PSO algorithm. In the gbest algorithm, the new position of the particle is achieved by adding the velocity component, as follows [13]:

\[
x_i(t+1) = x_i(t) + v_i(t+1), \quad v_i(t+1) = v_i(t) + c_1 r_{ij}(t)[y_j(t) - x_j(t)] + c_2 r_{ij}(t)[y(t) - x_i(t)],
\] (17)

where \( x_i(t) \) is position of particle \( i \) at time \( t \), \( v_i(t) \) is velocity of particle \( i \) at dimension \( j \) and time \( t \), \( y_j(t) \) is the best position of particle \( i \) up to time \( t \), \( y(t) \) is the best position of entire swarm up to time \( t \), \( c_1, c_2 \) are acceleration constants, and \( r_{ij}(t) \) and \( r_{ij}(t) \) are uniformly distributed numbers in \([0,1]\).

For optimal selection of SSA choices, two dimensional particles are randomly distributed in the search space. Since \( L \) and \( r \) are integer numbers, we apply the following simple modification to obtain a discrete PSO search algorithm.

\[
x_i(t+1) = \text{round}(x_i(t) + v_i(t+1)),
\] (18)

where function \text{round()} rounds the \( i \)-th argument to the nearest integer. Furthermore, through the optimization of SSA, the following constraints must be satisfied,

\[
L < \frac{N}{2}, \quad r \leq L.
\] (19)
The overall framework of the proposed OSSA–LLNF model is depicted in Fig. 3. Based on Fig. 3, the time series is first decomposed by the optimized SSA algorithm into a number of eigentriples. Then the informative eigentriples are selected for time series reconstruction while the remaining eigentriples (which are often high-frequency and worthless components), are eliminated. Finally, the filtered time series, which is a smoothed version of the original series, is used as the input vector for the LLNF model. It must be noted that the best structure of the LLNF model is chosen using validation data. For each time series a portion of training data is used as validation data set to select the best structure of the model.

The procedure of optimizing SSA by the PSO algorithm is illustrated in Fig. 4. In this procedure, the optimal parameters of SSA are selected based on the validation error of the LLNF model.

There are some points regarding SSA filtering and LLNF forecasting which are worth noting here; when applying SSA to the time series, the test series should not be filtered. Only the training data are utilized for filtering since the test points are assumed to be unknown. However, it is arbitrary whether to filter training output or not.

![Fig. 3. The proposed forecasting framework. Forecasting of time series at period \( t \), using a window length \( W \).](image)

![Fig. 4. SSA optimization procedure by PSO.](image)
As shown in Fig. 3, a moving window with a proper length of $N$ is considered in the proposed forecast framework. This window embraces the last $N$ points of the time series, preceding the test data. At each forecasting instant, the data points embraced by the moving window are filtered through the PSO optimized SSA and then used as the input series for training the LLNF model. This procedure is repeated until the whole forecasting interval is covered. For instance, consider a case when the last 20 points of time series $x(t)$ with length of 1000, is going to be forecasted (with one step ahead forecasting $h = 1$, where $h$ is the forecasting horizon) as test data with the following input and output vectors,

<table>
<thead>
<tr>
<th>No.</th>
<th>Input vector</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$x(1), x(2), x(3), x(4)$</td>
<td>$x(5)$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>976</td>
<td>$x(976), x(977), x(978), x(979)$</td>
<td>$x(980)$</td>
</tr>
</tbody>
</table>

Then, the window moves one point forward for predicting the second test point, i.e. $x(982)$. This procedure is carried on until the last test point ($x(1000)$) is forecasted. For this point the training inputs and outputs are as follows,

<table>
<thead>
<tr>
<th>No.</th>
<th>Input vector</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$x(20), x(21), x(22), x(23)$</td>
<td>$x(24)$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>976</td>
<td>$x(995), x(996), x(997), x(998)$</td>
<td>$x(999)$</td>
</tr>
</tbody>
</table>

Furthermore, the input vector for forecasting of $x(1000)$ is as stated below,

<table>
<thead>
<tr>
<th>Input vector</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x(996), x(997), x(998), x(999)$</td>
<td>$x(1000)$</td>
</tr>
</tbody>
</table>

### 5. Empirical results

In this section, the performance of the proposed hybrid approach is evaluated based on four different case studies, including a hand-designed as well as three real-world systems. First, forecasting of benchmark time series of Mackey–Glass as a hand-designed system is carried out and the forecasting ability of the proposed hybrid method is compared to a variety of previously proposed methods in literature. The proposed method is also applied for forecasting of other three real time series; the Box–Jenkins time series, solar activity, and the US accidental death series. The following error criteria are used for performance evaluation.

- **Root mean square of error (RMSE)**
  \[
  \text{RMSE} = \sqrt{\frac{1}{T} \sum_{t=1}^{h-T} (P_t - \hat{P}_t)^2}.
  \] (21)

- **Normalized mean square of error (NMSE)**
  \[
  \text{NMSE} = \sum_{h=1}^{h-T} (P_t - \hat{P}_t)^2 \left/ \sum_{h=1}^{h-T} (P_t - \bar{P})^2 \right. .
  \] (22)

- **Mean absolute error (MAE)**
  \[
  \text{MAE} = \frac{1}{T} \sum_{h=1}^{h-T} |P_t - \hat{P}_t|.
  \] (24)
Mean absolute percentage error (MAPE)

\[
MRAE = \frac{100}{T} \sum_{h=1}^{b-1} \frac{|P_t - \hat{P}_t|}{P_t},
\]

(25)

where \(P_t\) and \(\hat{P}_t\) are the target and predicted values of the time series at time \(t\), and \(T\) is the number of forecasted data points. Furthermore, we will consider the ratio of the RMSE and NMSE of the proposed method to those of other methods as another criterion.

Moreover, to enable statistical evaluations the modified Diebold–Mariano (DM) test will be computed for each case study. The DM test for one-step ahead forecasts are defined by [19]

\[
DM = \bar{d} \sqrt{N(N - 1) \left( \sum_{t=1}^{T} (d_t - \bar{d})^2 \right)^{-1/2}},
\]

(26)

where

\[
\bar{d} = \frac{1}{T} \sum_{t=1}^{T} d_t,
\]

(27)

and,

\[
d_t = (P_t - \hat{P}_t)^2 - (P_{t-1} - \hat{P}_{t-1})^2.
\]

Large (in absolute values) negative values of DM statistic indicate superiority of the approach [19].

It must be noted that at the optimization stage, 30 particles, each being two-dimensional, are used for optimal selection of SSA choices by PSO algorithm through 100 iterations.

5.1. Box–Jenkins time series

Let us now consider the performance of the proposed method for forecasting Box–Jenkins time series (one step ahead forecasting \(h = 1\)). This time series is generated by the combustion process of methane–air mixture, with gas flow rate \(u(t)\) as the process input and the CO\(_2\) concentration as the output of combustion process \(y(t)\) [8]. For forecasting the Box–Jenkins time series, we have considered two cases of input features. In the first case, the input variables \(u(t - 4)\) and \(y(t - 1)\) are considered for forecasting output \(y(t)\), whilst in the second case inputs \(u(t - 4), u(t - 5), y(t - 1)\) and \(y(t - 2)\) are used. It must be noted that for the first case of inputs, we did not filter training output by SSA. However, for the second case of inputs the training outputs were filtered. The original time series includes 296 data points. The first 200 observations are used for the training period and the remaining 92 observations are used as test data. In order to assess the model’s accuracy, RMSE is used. In addition, we have compared the results obtained using our proposed method with those obtained through the models proposed in [68,8–10].

The PSO algorithm was first executed to identify the optimal choices of the SSA algorithm. The PSO optimized SSA algorithm resulted in the decomposition of Box–Jenkins time series into 9 components. We then reconstruct the noise-free series using the first 5 components through the fourth stage of SSA, for the first case of inputs. For the second case of inputs, 9 out of 10 components were used for the reconstruction stage. The singular values for Box–Jenkins time series, for \(L = 10\), are shown in Fig. 5. Next, the filtered series was used to generate the input features in the LLNF model. Fig. 6 shows the predicted and actual values of the test series obtained using four inputs for the OSSA–LLNF model. Fig. 6 implies that the hybrid method has accurately identified and modelled the dynamics of the Box–Jenkins time series by means of the training data. The forecasting errors, the difference between actual and forecasted values generated by the LLNF model and OSSA–LLNF are illustrated in Fig. 7, indicating a noticeable improvement in forecasting, obtained through SSA’s de-noising procedure. The 95% confidence intervals, obtained by the proposed OSSA–LLNF approach are also shown in Fig. 8. Furthermore, a numerical comparison to other methods as well as the LLNF model, in terms of RMSE, is summarized in Table 2. The ratio of RMSE of the proposed method to the RMSE of the compared methods is also presented in Table 2. If the ratio of RMSE < 1, then the OSSA–LLNF procedure outperforms the alternate prediction method. On the other hand, if the ratio >1 it indicates that the performance of the OSSA–LLNF procedure is worse than the predictions of the competing method. The small values for RMSE ratio in Table 2 indicate much better performance of the proposed approach. The comparison of results reveals the superior performance of the proposed hybrid method with respect to the previously developed methods and the LLNF model without SSA filtering. We therefore conclude that removal of noisy components of the Box–Jenkins time series using SSA improves the forecasting performance.

For the statistical assessment, the results for the DM statistic for the OSSA–LLNF model and the LLNF model for both two-input and four-input case studies are presented in Table 3. The superiority of the proposed approach can be easily seen from the results of the DM test.
Fig. 5. Singular spectrum of Box–Jenkins time series.

Fig. 6. Actual and predicted values for Box-Jenkins time series ($h = 1$).

Fig. 7. Forecasting error using LLNF and OSSA–LLNF – Box–Jenkins time series.
5.2. Mackey–Glass time series

Mackey–Glass time series is generated using the following time-delay differential equation, introduced as a model for white blood cell production,

\[
\frac{dx(t)}{dt} = \frac{ax(t - \tau)}{1 + x^{10}(t - \tau)} + \beta x(t).
\]  

This time series is chaotic for \( \tau > 16.8 \) and exhibits no clearly defined period. This benchmark series has been used in numerous neural network and fuzzy modelling research (see, for example [68,8]). The series is very sensitive to the initial conditions. Here, we choose \( x(0) = 1.2, a = 0.2, \beta = -0.1, \tau = 17 \) and evaluate six-step ahead forecasting results.

The standard input variables for this case are \( x(t - 18), x(t - 12), x(t - 6), x(t) \) for forecasting \( x(t + 6) \) (six step ahead forecasting \( h = 6 \)). From 1000 observations, the first 500 data are used as the training set and the last 500 are employed.

### Table 2
The forecasting results of Box-Jenkins time series (\( h = 1 \)).

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>RMSE ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARMA [32]</td>
<td>0.843</td>
<td>0.011</td>
</tr>
<tr>
<td>Tong’s model [33]</td>
<td>0.685</td>
<td>0.014</td>
</tr>
<tr>
<td>Pedrycz’s model [34]</td>
<td>0.566</td>
<td>0.017</td>
</tr>
<tr>
<td>Xu’s model [35]</td>
<td>0.573</td>
<td>0.017</td>
</tr>
<tr>
<td>Sugeno’s model [36]</td>
<td>0.596</td>
<td>0.016</td>
</tr>
<tr>
<td>Surmann’s model [37]</td>
<td>0.400</td>
<td>0.024</td>
</tr>
<tr>
<td>Lee’s model [59]</td>
<td>0.638</td>
<td>0.015</td>
</tr>
<tr>
<td>Lin’s model [39]</td>
<td>0.511</td>
<td>0.019</td>
</tr>
<tr>
<td>Nie’s model [32]</td>
<td>0.412</td>
<td>0.023</td>
</tr>
<tr>
<td>ANFIS [48]</td>
<td>0.085</td>
<td>0.113</td>
</tr>
<tr>
<td>FuNN [42]</td>
<td>0.0226</td>
<td>0.425</td>
</tr>
<tr>
<td>HyFIS [43]</td>
<td>0.0205</td>
<td>0.468</td>
</tr>
<tr>
<td>Neural tree model [44]</td>
<td>0.0265</td>
<td>0.362</td>
</tr>
<tr>
<td>WNN + gradient [31]</td>
<td>0.084</td>
<td>0.114</td>
</tr>
<tr>
<td>WNN + hybrid [31]</td>
<td>0.081</td>
<td>0.119</td>
</tr>
<tr>
<td>LWNN + gradient [31]</td>
<td>0.01643</td>
<td>0.584</td>
</tr>
<tr>
<td>LNN + hybrid [31]</td>
<td>0.01378</td>
<td>0.697</td>
</tr>
<tr>
<td>Recurrent ANFIS [31]</td>
<td>0.019</td>
<td>0.505</td>
</tr>
<tr>
<td>TNFIS [46]</td>
<td>0.0230</td>
<td>0.417</td>
</tr>
<tr>
<td>FWNN-S (2MFs) [6]</td>
<td>0.03085</td>
<td>0.311</td>
</tr>
<tr>
<td>FWNN-S (3 MFs) [6]</td>
<td>0.02778</td>
<td>0.346</td>
</tr>
<tr>
<td>FWNN-R (2MFs) [6]</td>
<td>0.03171</td>
<td>0.303</td>
</tr>
<tr>
<td>FWNN-R (3 MFs) [6]</td>
<td>0.02794</td>
<td>0.344</td>
</tr>
<tr>
<td>FWNN-M (2 MFs) [6]</td>
<td>0.02963</td>
<td>0.324</td>
</tr>
<tr>
<td>FWNN-M (3 MFs) [6]</td>
<td>0.02324</td>
<td>0.413</td>
</tr>
<tr>
<td>LLNF (two inputs)</td>
<td>0.0462</td>
<td>0.208</td>
</tr>
<tr>
<td>OSSA–LLNF (two inputs)</td>
<td>0.0321</td>
<td>0.300</td>
</tr>
<tr>
<td>LLNF (four inputs)</td>
<td>0.0235</td>
<td>0.409</td>
</tr>
<tr>
<td>OSSA–LLNF (four inputs)</td>
<td>0.0096</td>
<td>–</td>
</tr>
</tbody>
</table>
as test series. Based on the optimization performed by the PSO algorithm, the series was decomposed into 18 components; the first 5 were used to reconstruct the filtered series, signal reconstruction (i.e. \( L = 18 \) and \( r = 5 \)). It must be noted that during training process for this case, the output of training data are filtered by the optimized SSA algorithm.

The model’s performance for this case study is compared with the results reported in [68,8–59,64,23]. Among the compared methods in this case study, the fuzzy wavelet neural network (FWNN) method developed in [68] has shown the best performance. Three different consequent parts of the fuzzy rules were considered for developing FWNN, resulting in FWNN-S, FWNN-M and FWNN-R.

The actual and forecasted values of the test series and the forecasting error of the LNNF model and OSSA–LNNF are depicted in Figs. 9 and 10 respectively. Remarkable improvement in the performance of the LNNF model using SSA is obvious when comparing forecasting errors obtained from LNNF and OSSA–LNNF approaches. The 95% confidence intervals are illustrated in Fig. 11. It is visible that both upper and lower intervals are very close to the target time series. For a better illustration, a small part of Fig. 11 is magnified in the top-left box.

A comparison between the proposed OSSA–LNNF model and other methods, in terms of RMSE is tabularized in Table 4. The comparison reveals the better performance of the OSSA–LNNF model with respect to the other methods. Furthermore, the ratio of RMSE of the proposed method to the RMSE of the compared methods is also computed in Table 4, as a measure of accuracy.

<table>
<thead>
<tr>
<th>Method</th>
<th>DM statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLNF (two inputs)</td>
<td>-2.86</td>
</tr>
<tr>
<td>OSSA–LLNF (two inputs)</td>
<td>-4.02</td>
</tr>
<tr>
<td>LLNF (four inputs)</td>
<td>-4.89</td>
</tr>
<tr>
<td>OSSA–LLNF (four inputs)</td>
<td>-6.80</td>
</tr>
</tbody>
</table>

Table 3
The results of DM statistics for Box–Jenkins time series.

Fig. 9. Six-step ahead forecasting of Mackey–Glass time series.

Fig. 10. Forecasting error using LLNF and OSSA–LNNF (six-step ahead forecasting of Mackey–Glass time series).
indicating the performance improvement. For instance, for FWNN-S, FWNN-M, FWNN-R, the value of RMSE ratio is 0.92, 0.88 and 0.43, respectively. Another interesting result is the RMSE ratio for LLNF without SSA filtering, with the value of 0.59. This indicates the effect of including SSA for forecasting purposes. In addition, the results of the DM test for the proposed approach and the LLNF model are summarized in Table 5.

5.3. Sunspot series

Let us now consider the sunspot time series which is a real-world highly complex and non-stationary time series [63,58]. The solar cycle series is very difficult to predict due to the intrinsic complexity of the related time behaviour [54]. This series

![Mackey–Glass time series](image)

**Fig. 11.** 95% Confidence intervals for Mackey–Glass time series.

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>RMSE ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product T-norm [47]</td>
<td>0.0907</td>
<td>0.01</td>
</tr>
<tr>
<td>ANFIS [48]</td>
<td>0.0015</td>
<td>0.67</td>
</tr>
<tr>
<td>Classical RBF network [49]</td>
<td>0.0114</td>
<td>0.09</td>
</tr>
<tr>
<td>GA and fuzzy system [50]</td>
<td>0.049</td>
<td>0.02</td>
</tr>
<tr>
<td>PG-RBF [51]</td>
<td>0.0028</td>
<td>0.36</td>
</tr>
<tr>
<td>Neural tree model [44]</td>
<td>0.0069</td>
<td>0.15</td>
</tr>
<tr>
<td>Linear model [31]</td>
<td>0.55</td>
<td>0.001</td>
</tr>
<tr>
<td>Auto-regressive model [31]</td>
<td>0.19</td>
<td>0.01</td>
</tr>
<tr>
<td>Cascade correlation NN [31]</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>Sixth-order polynomial [31]</td>
<td>0.04</td>
<td>0.03</td>
</tr>
<tr>
<td>Back-propagation NN [31]</td>
<td>0.02</td>
<td>0.05</td>
</tr>
<tr>
<td>WNN + gradient [31]</td>
<td>0.0071</td>
<td>0.14</td>
</tr>
<tr>
<td>WNN + hybrid [31]</td>
<td>0.0059</td>
<td>0.17</td>
</tr>
<tr>
<td>LWNN + gradient [31]</td>
<td>0.0041</td>
<td>0.24</td>
</tr>
<tr>
<td>LWNN + hybrid [31]</td>
<td>0.0036</td>
<td>0.28</td>
</tr>
<tr>
<td>RBF [53]</td>
<td>0.0015</td>
<td>0.67</td>
</tr>
<tr>
<td>Recurrent ANFIS [31]</td>
<td>0.0013</td>
<td>0.77</td>
</tr>
<tr>
<td>SEIT2FNN [53]</td>
<td>0.0034</td>
<td>0.3</td>
</tr>
<tr>
<td>NFIS-SEELA [54]</td>
<td>0.00747</td>
<td>0.14</td>
</tr>
<tr>
<td>FLNNF-CCPso [55]</td>
<td>0.00842</td>
<td>0.12</td>
</tr>
<tr>
<td>IT2FNN [56]</td>
<td>0.0020</td>
<td>0.5</td>
</tr>
<tr>
<td>MSBFNN [57]</td>
<td>0.0024</td>
<td>0.42</td>
</tr>
<tr>
<td>FWNN-S [6]</td>
<td>0.00109</td>
<td>0.92</td>
</tr>
<tr>
<td>FWNN-R [6]</td>
<td>0.00232</td>
<td>0.43</td>
</tr>
<tr>
<td>FWNN-M [6]</td>
<td>0.00114</td>
<td>0.88</td>
</tr>
<tr>
<td>LLNF</td>
<td>0.0017</td>
<td>0.59</td>
</tr>
<tr>
<td>OSSA–LLNF</td>
<td>0.001003</td>
<td>–</td>
</tr>
</tbody>
</table>

**Table 4**  
The results of six-step ahead forecasting of Mackey–Glass time series ($h = 6$).

<table>
<thead>
<tr>
<th>Method</th>
<th>DM statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLNF</td>
<td>–18.12</td>
</tr>
<tr>
<td>OSSA–LLNF</td>
<td>–20.33</td>
</tr>
</tbody>
</table>

**Table 5**  
The results of DM statistics for Mackey–Glass time series.
corresponds to annual average relative number of sunspots observed. A considerable number of CI-based approaches have studied the sunspot numbers (see, for example, [68,40,2]). The annually recorded sunspot time series from 1700 to 1955 is considered here [46]. The input variables are \( x(t - 4), x(t - 3), x(t - 2), x(t - 1) \) where the output variable is \( x(t) \) (one step ahead forecasting \( h = 1 \)). The whole sunspot series is divided into a training set form 1700 to 1920 and two test sets from 1921 to 1955 and 1956 to 1979 respectively [68]. It must be noted that the second test series is typical of the whole dataset.

For forecasting the first test series, we have employed 20 out of 27 components for the reconstruction stage and the LLNF model with 14 neurons yielding the best performance. The original and residual (the difference between the original and filtered series) series are depicted in Fig. 12. It is obvious that the residual series contains very high frequency components which are unpredictable (they are actually noisy components). For the second test series, 12 out of 15 components were used in the reconstruction stage. Furthermore, the output of training data was filtered by the optimized SSA algorithm. A graphical comparison between the actual and forecasted values obtained by OSSA–LLNF for test series 1 and 2 are shown in Fig. 13.

Similar to the previous case studies, the 95% confidence intervals are also shown in Fig. 14.

Furthermore, the comparison results with other methods, including FWNN models developed in [68] and other methods [62,2] are presented in Table 6. Based on the results presented in Table 6, one can conclude that the proposed OSSA–LLNF model has outperformed the three FWNN and other models in test case 1. The proposed OSSA–LLNF model has also shown promising performance for test series 2. The ratio of NMSE between the benchmark methods and the proposed method is also provided in Table 6. In test series 1, the NMSE ratio for RFNN model [2] (as the best among compared methods) is 0.81. The NMSE ratio of the best benchmark method for the test series 2, i.e. FWNN-S, is 0.56. The ratio values for both test series 1 and 2 indicate surpassing performance of the proposed OSSA–LLNF method over other approaches. In addition, NMSE ratios for LLNF without SSA filtering demonstrate the positive effects on time series processing via SSA. A comparison between the proposed approach and the LLNF model, in terms of DM statistics, can be found in Table 7.

![Fig. 12. (a) Actual and (b) residual of the sunspot series (obtained from training data).](image)

![Fig. 13. Forecasting results of the sunspot time series \((h = 1)\).](image)
5.4. The US accidental deaths series

The US accidental deaths series (Death series) provides information about monthly accidental deaths in the United States [21]. The death time series includes number of accidental deaths from Jan. 1973 to Jun. 1979. The last six data points are considered as the test series and the remaining are employed for training purposes (six step ahead forecasting $h = 6$). Here we perform a six-step ahead forecast and all the test points (Jan. 1979 to Jun. 1979) will be forecasted using the information prior to Jan. 1979.

Prior to constructing the forecasting model, appropriate input variables should be determined. A simple, but efficient, auto-correlation analysis is carried out to select the most relevant input variables. Based on the auto-correlation analysis, input variables $x(t - 24), x(t - 18), x(t - 12), x(t - 2), x(t - 1)$ are selected to predict $x(t)$. The last two inputs represent the trend component while the first four inputs exhibit periodic behaviour of the series. It should be noted that for forecasting the test data, an iterative approach is adopted for unavailable input variables. For instance, when predicting the second test point, the actual value of the input variable $x(t - 1)$ is unknown. Hence, the predicted value for the test point is used. We also use 10 out of 22 eigentriples of the original time series to reconstruct the signal components (filtered series). The LLNF model with 3 neurons showed the best performance. Fig. 15 shows the actual and predicted values for training and forecasting periods. According to Fig. 15, the forecasted values match the actual value for both training and forecasting periods. A numerical evaluation in terms of MAE and MAPE are presented in Table 8. The comparison with some other methods in
literature is also conducted and the results are given in Table 8. The results indicate that the performance of the LLNF model has been significantly improved via the SSA filtering approach (MAE decreased from 269 to 162 and MAE ratio for LLNF model equals to 0.62). Furthermore, the proposed OSSA–LLNF approach has outperformed all the methods described in [21].

6. Conclusion

In this paper, a hybrid approach, based on the local linear neuro-fuzzy models and the SSA technique was proposed for the forecasting of non-stationary, nonlinear and chaotic time series. By means of the SSA technique, the high frequency components of the noisy time series were removed in order to enhance the forecasting performance of the LLNF model. The SSA algorithm was optimized by the PSO to obtain the optimal filtering performance. The use of the OSSA–LLNF approach for the forecasting four well-known benchmark data sets, namely Box–Jenkins, Mackey–Glass, sunspot and death time series, and comparison with the several old and recent methods, confirmed the outstanding forecasting performance of the proposed approach.

References


Table 8
The forecasting results of the US death series (h = 6).

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>MAPE (%)</th>
<th>MAE ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model I [27]</td>
<td>524</td>
<td>6</td>
<td>0.32</td>
</tr>
<tr>
<td>Model II [27]</td>
<td>415</td>
<td>5</td>
<td>0.40</td>
</tr>
<tr>
<td>HWS [27]</td>
<td>351</td>
<td>4</td>
<td>0.48</td>
</tr>
<tr>
<td>LLNF</td>
<td>269</td>
<td>3.2</td>
<td>0.62</td>
</tr>
<tr>
<td>ARAR [27]</td>
<td>227</td>
<td>3</td>
<td>0.74</td>
</tr>
<tr>
<td>SSA [27]</td>
<td>180</td>
<td>2</td>
<td>0.93</td>
</tr>
<tr>
<td>OSSA–LLNF</td>
<td>162</td>
<td>1.9</td>
<td>–</td>
</tr>
</tbody>
</table>

Fig. 15. Six step-ahead forecasting of the US. death time series (h = 6).

Table 8
The forecasting results of the US death series (h = 6).


Further reading
