Expert Systems with Applications 40 (2013) 6438–6446

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

Optimization of the weights and asymmetric activation function family of neural network for time series forecasting

Gecynalda S. da S. Gomes [a], Teresa B. Ludermir [b,*]

[a] Departamento de Estatística, Universidade Federal da Bahia, Salvador, Brazil
[b] Centro de Informatica, Universidade Federal de Pernambuco, Recife, Brazil

ARTICLE INFO

Keywords:
Neural networks
Asymmetric activation function
Free parameter
Simulated annealing
Tabu search
BPM algorithm
LM algorithm
Time series

ABSTRACT

The use of neural network models for time series forecasting has been motivated by experimental results that indicate high capacity for function approximation with good accuracy. Generally, these models use activation functions with fixed parameters. However, it is known that the choice of activation function strongly influences the complexity and neural network performance and that a limited number of activation functions has been used in general. We describe the use of an asymmetric activation functions family with free parameter for neural networks. We prove that the activation functions family defined, satisfies the requirements of the universal approximation theorem We present a methodology for global optimization of the activation functions family with free parameter and the connections between the processing units of the neural network. The main idea is to optimize, simultaneously, the weights and activation function used in a Multilayer Perceptron (MLP), through an approach that combines the advantages of simulated annealing, tabu search and a local learning algorithm. We have chosen two local learning algorithms: the backpropagation with momentum (BPM) and Levenberg–Marquardt (LM). The overall purpose is to improve performance in time series forecasting.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Artificial Neural Network (ANN) models are an important class of models that have attracted considerable attention and have been deployed in many applications. The use of these models in applied work is, generally, motivated by a mathematical result stating that, under mild regularity conditions, a relatively simple ANN model is capable of approximating any Borel measurable function from one finite dimensional space into $(0,1)^n$ to another to any desired degree of accuracy (Hornik, Stinchcombe, & White, 1989). Neural networks with a single hidden layer using sigmoid activation functions are universal approximators, i.e., these functions are capable of approximating any measurable function to any desired degree of accuracy (Funahashi, 1989; Hornik et al., 1989; Rumelhart, Hinton, & Williams, 1986).

Input–output mapping can be represented by neural networks through a combination of weighted connections between neurons (Rumelhart et al., 1986). Funahashi (1989) proved that any continuous mapping can be performed by a Multilayer Perceptron (MLP) with differentiable and monotonically increasing activation functions. In the author's work, sigmoid activation functions were used in the hidden layer and linear activation functions were used in the output layer.

Regarding the time series forecasting problem, many authors have used for decades different statistical methods ranging from moving averages and exponential smoothing and linear or nonlinear regressions to modeling and forecasting. Box and Jenkins (1990) developed autoregressive integrated moving average models to time series forecasting. To improve time series forecasting with nonlinear characteristics, several researchers have developed alternative methods that model such approximations, for example, Autoregressive Heteroscedastic models (ARCH) (Engle, 1982). Despite the improvements of these methods over linear methods, they tend to be specific to particular applications. The use of neural network models for forecasting time series is motivated by experimental results that show a high capacity of approximation to functions with high accuracy. Generally, these models use activation functions with fixed parameters, as seen in Zhang (2003) and Ghiassi, Saidane, and Zimbra (2005). As ANN models are used as universal function approximations (Hornik et al., 1989), many researchers use them to forecast diverse nonlinear models in a time series and for evaluating efficacy and performance compared to traditional forecasting methods (Gomes, Maia, Ludermir, Carvalho, & Araujo, 2006; Zhang, 2003).

Generally, the performance of ANNs depends on various factors such as the number of hidden layers, the number of hidden neurons, the learning algorithm and the activation function of each neuron.

* Corresponding author. Tel.: +55 81 21268430; fax: +55 81 21268438.
E-mail addresses: gecynaldassg@ufba.br (Gecynalda S. da S. Gomes), tbl@cin.ufpe.br, teresa.ludermir@gmail.com (T.B. Ludermir).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.05.053

The current emphasis in neural network research is on learning algorithms and architectures, therefore neglecting the importance of activation functions. However, the choice of activation functions can strongly influence complexity and performance of neural networks and have been said to play an important role in the convergence of the algorithms (Chandra & Singh, 2004; Duch & Jankowski, 1999; Gomes, Ludermir, & Lima, 2011; Singh & Chandra, 2003).

Several types of activation functions have been proposed. Hornik (1993) and Leshno, Lin, Pinkus, and Schocken (1993) used non-polynomial activation functions. Leung and Haykin (1993) used rational transfer functions with very satisfactory results. Rosen-Zvi, Biehl, and Kanter (1998) showed the general results of ANN models with periodic activation functions. Ma and Khorasani (2005) used Hermite polynomial activation functions. Gomes and Ludermir (2008) proposed the use of two new activation functions, complementary log–log and probit, which performed well in comparison to the logit activation function (log-sigmoid activation function). Gomes et al. (2011) implemented complementary log–log, probit and log–log activation functions, compared and evaluated their performance available using financial market data sets, through two learning algorithms. The authors also demonstrated that these functions are universal approximators of continuous functions and are suitable for regression problems. One characteristic of these functions is that they have fixed parameters and cannot therefore be adjusted to adapt to different problems.

There have been a limited number of studies with emphasis on activation functions with free parameters, i.e., adaptative activation functions. Some studies have shown that neural networks with activation functions of free parameters seems to provide better performance than classical architectures, with fixed activation function nodes. Guarnieri, Piazza, and Uncini (1999) presented a new activation function called spline adaptive, studied their properties and showed an improvement both in complexity and performance of the neural network in terms of generalization. Networks with such activation functions seem to provide better fitting properties than classical architectures, with fixed activation function neurons (Guarnieri et al., 1999). More recent studies have demonstrated the importance of activation functions for the learning of the neural network. For instance, Singh and Chandra (2003) proposed a new class of sigmoid functions and proved that these functions satisfy the requirements of the universal approximation theorem. Chandra (2003) proposed two parameterization methods that allow the construction of a class of sigmoid functions based on any given sigmoid function. It was demonstrated that all members of the proposed class satisfy the requirements for using an activation function in neural networks. Chandra and Singh (2004) use self-adaptive activation functions to assess the best sigmoid function from the class of sigmoid functions proposed in Singh and Chandra (2003). These and many other papers, surveyed here, demonstrate that the choice of activation functions is considered by some experts to be as important as the network architecture and learning algorithm.

The logit function assumes a continuous range of values from 0 to 1. It is sometimes desirable to have the activation function range from -1 to +1, in which case the activation function assumes an asymmetric form with respect to the origin, the hyperbolic tangent function is a good choice in such cases (Haykin, 2001). But, when the probability of a given binary response approaches 0 at a different rate than it approaches 1, the symmetric link is inappropriate (Chen, Dey, & Shao, 1999). Duch and Jankowski (1999) comment on the importance of symmetry in the activation functions, however, were not conducted empirical experiments that relate this with the behavior of the data. Based on these facts, a question remains as to precisely which features of the activation function determine the properties of the learning network: What is the relevance of symmetries in activation functions?.

To study the relevance of symmetries in activation functions, we propose the use of new functions as Asymmetric Activation Functions Family with Free Parameter (AAFFFP), for neural networks based on the family of Aranda-Ordaz transformations with asymmetric alternatives (Aranda-Ordaz, 1981). This family of Aranda-Ordaz functions was proposed for binary data and is used as link functions in Generalized Linear Models (GLM), when data have binomial distribution. For details see (Nelder & Wedderburn, 1972).

To optimize the value of the parameter AAFFFP and the weights and bias of the neural network, we combine the advantages of simulated annealing, tabu search and the backpropagation training algorithm. This approach is based on the paper by Ludermir, Yamazaki, and Zanchettin (2006) where is generated automatic process for producing networks with high classification performance and low complexity, whose aim is the simultaneous optimization of MLP network weights and architectures. However, all network units implemented the hyperbolic tangent activation function. In our study, the architecture of the neural network is predefined to be evaluated the actual effect of weights optimization combined with the activation function optimization with free parameter.

Generally, existing ANN models for time series forecasting use MLP networks, in which the number of hidden layers, the number of nodes in the input and hidden layers and activation function are chosen, often by trial and error in order to find a plausible model for the specific application. Ghiassi and Saidane (2005) developed a neural network model - DAN2: architecture for dynamic ANNs - which employs a different architecture than traditional models. To demonstrate the effectiveness of the DAN2 model, the authors compared their performance with the performance of the traditional ANN and ARIMA, showing the superiority of the DAN2 model for time series forecasting. For this reason, we implemented the model DAN2 to serve as a reference and compare the results of the neural network model with those of AAFFFP proposal.

Hybrid expert systems have been largely used in many applications (Sahin, Tolun, & Hassanpour, 2012), as well as in time series forecasting (Khashei & Bijari, 2012), so the main goal of this paper is to find a hybrid neural network model that presents a good performance in adjustment and forecasting of time series, which have different behaviors.

This model combines the techniques of simulated annealing, tabu search and a local learning algorithm, backpropagation with the term momentum (BPM) or Levenberg–Marquardt (LM), whose activation function has a free parameter and whose network architecture contains a hidden layer and few hidden nodes, thus providing more reliable in the results of time series forecasting. The activation functions family to be used has as special cases the logit function and the complementary log–log function (Aranda-Ordaz, 1981) and satisfies the requirements of the universal approximation theorem.

This paper is organized in the following way. In Section 2, we present works on neural network optimization. In Section 3, we present the mathematical proof according to which the new functions satisfy the universal approximation theorem. The optimization methodology is presented in Section 4. In Section 5, we present the configuration of the experiments and the results. Finally, in Section 6, the conclusion is presented.

2. Neural network global optimization

Various optimization techniques have been used in the literature, in order, to improve the performance of the ANN, such as simulated annealing (SA), tabu search (TS), genetic algorithms (GAs) and others. These techniques are, generally, used as a hybrid approach to the training of the neural network. Generally, the goal is to minimize the main problem of gradient-based algorithms: the local convergence.

An integration of SA, TS and GAs was proposed by Mantawy, Abdel-Magid, Selim, algorithms, and search (1999). In Li, Ong, and Nee (2002), GAs and SA were combined for the optimization of processes in the engineering plans. In Ting, Li, and Lee (2003), a combination between GA and TS was used. Generally, it is known that global optimization techniques, like SA and TS, are relatively inefficient for fine tuning in local searches. As a result, it is important to investigate whether the generalization performance of networks can still be enhanced when the topologies generated by these techniques are trained with a local search approach, such as the backpropagation algorithm. In training ANN, these technique mixtures were used in various applications, simultaneously or not (Ludermir et al., 2006; Yamazaki & Ludermir, 2003; Zanchettin, Ludermir, & Almeida, 2011).

Tsai, Chou, and Liu (2006) used a hybrid Taguchi-genetic algorithm to solve the problem of tuning both network structure and parameters of a feedforward artificial neural networks. Gepperth and Roth (2006) used a multi-objective evolutionary process in order to optimize feedforward architectures. In Hamm et al. (2002) SA was used for optimizing neural network weights. Yamazaki and Ludermir (2003) made use of SA and TS, simultaneously, in order, to optimize the weights and architecture. In this case, the problem taken into consideration was the scent detection in an artificial nose. Ludermir et al. (2006) combined three techniques: SA, TS and the backpropagation training algorithm, in order, to create an automatic process that produces networks with good classification performance and low complexity. Zanchettin et al. (2011) present an optimization method that integrates four techniques: SA, TS, GA and the backpropagation training algorithm in order to find weights and architecture for a neural network. Ferreira, Ludermir, and Aquino (2013) propose an approach to reservoir computing design and training using an evolutionary strategy.

In most approaches, the authors use these techniques in order to optimize parameters and initial values for weight connections between processing units and network architecture, fixating an activation function commonly used in the literature, such as sigmoid logistic or hyperbolic tangent. For example, in Ludermir et al. (2006), Yamazaki and Ludermir (2003), Zanchettin et al. (2011) the activation function used in all the problems was a hyperbolic tangent with a fixed parameter. There are works where free parameter activation functions are used, though some authors employ an adaptation of the backpropagation algorithm as a search method for the best value of the parameter (Chandra & Singh, 2004). However, this type of approach continues facing problems with local optimization. Other authors use optimization methods like the line search (Chandra & Singh, 2004; Gomes, Ludermir, & Almeida, 2009). Thus arises the idea of optimizing, simultaneously, the activation function and the weights of the network.

3. Asymmetric activation function family

A sigmoid function can be defined as Chandra (2003).

Definition 1. : A real function, $f(x), f:\mathbb{R}\to\mathbb{R}$, with the properties $x \to \pm\infty$.

$$\lim_{x\to+\infty} f(x) = a; \quad \lim_{x\to-\infty} f(x) = b, \tag{1}$$

where a and b are real numbers and $a > b$. The usual values are $a = 1$ and $b = 0$ or -1.

The general class of sigmoid functions includes discontinuous functions such as the Heaviside's theta function, the step function and the sign function, as well as continuous functions such as the arctangent function, the hyperbolic tangent function and the logit function. Any function that is non-constant, bounded and monotonically increasing, satisfies (1) and therefore belongs to the set of all sigmoid functions. For sigmoid functions, including the family of asymmetric activation functions ($\mathcal{F}_a$), the universal approximation theorem (UAT) can be summarized as Haykin (2001).

The UAT gives a mathematical justification for the approximation of an arbitrary continuous function opposed to its exact representation (Haykin, 2001). The UAT provides a set of conditions that an activation function should satisfy so that any neural network using them has the universal approximation property. The condition that the inputs belong to the unit hypercube can be generalized to any bounded hypercube. There may be additional algorithmic and efficiency requirements for the activation functions to be distinguishable and satisfy a simple differential equation for the evaluation of weight increments.

The Aranda-Ordaz asymmetric function is defined by Aranda-Ordaz (1981)

$$\upsilon = \log\left[\frac{(1-\pi)^{-\lambda}}{\lambda}\right], \tag{2}$$

where $\pi \in (0,1)$ and $\lambda > 0$. The inverse of (2) takes the form

$$\pi = 1 - (1 + \lambda e^{\upsilon})^{-1/\lambda}. \tag{3}$$

Therefore, for neural network models, we will utilize the function (3) in order for it to be an AAFFFP ($\mathcal{F}_\lambda$). Hence, in our context, we have

$$f_\lambda(x) = 1 - (1 + \lambda e^{x})^{-1/\lambda}, \quad \lambda > 0. \tag{4}$$

Fig. 1 presents the $f_\lambda(x)$ behavior as a function of x for the activation function for different values of λ. Observe that the sigmoid logistic functions and complementary log–log functions are special cases of the sigmoid family $\mathcal{F}_\lambda$, when $\lambda = 1$ and $\lambda \to 0$, respectively. For values of $\lambda > 1, f_\lambda(x)$ approximates more slowly to one than in the logistic sigmoid function.

For every member of the family $\mathcal{F}_\lambda$, the propositions below establish that they are non-constant, bounded and monotonically increasing.

Proposition 3.1. . *Every member of the class $\mathcal{F}_\lambda$ is a monotonically increasing (MI) function.*

Proof (*See Gomes et al. (2009)*). □

Proposition 3.2. *Every member $f_\lambda(x)$ of the family $\mathcal{F}_\lambda$ is bounded above by 1 and below by 0, that is, the following relations are true:*

$$\lim_{x\to+\infty} f_\lambda(x) = 1; \quad \lim_{x\to-\infty} f_\lambda(x) = 0. \tag{5}$$

Proof (*See Gomes et al. (2009)*). □

Proposition 3.3. . *Every member of the family $\mathcal{F}_\lambda$, i.e., $f_\lambda(x)$, satisfies the generalized equation given by equation $(1 + \lambda\, e^{x})^{-(1+\lambda)/\lambda} e^{x}$.*

Proof (*Differentiate Eq. (4)*). From the Propositions 3.1, 3.2, 3.3, we can see that every member of $\mathcal{F}_\lambda$ is non-constant, limited and monotonically increasing. So, every member of the family $\mathcal{F}_\lambda$ satisfies the properties required by the UAT, and therefore they can be used as activation functions of a neural network. □

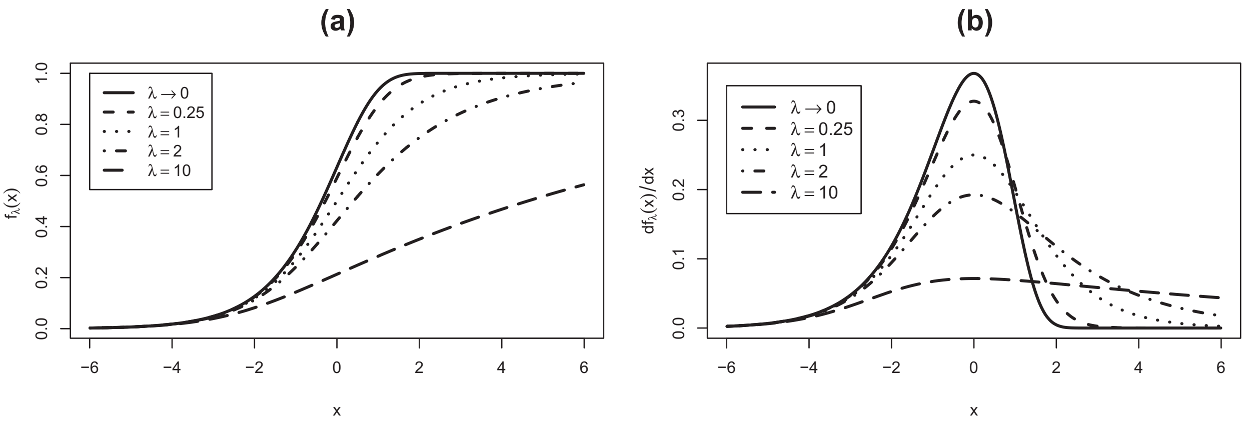

Fig. 1. (a) Activation function for the different values of λ and (b) the corresponding derivatives.

4. Optimization methodology

The solution to an optimization problem can be characterized as a local or global search process. The local search process aims to find the best solution within a set of solutions in a restricted space, taking into consideration that this solution depends on the starting point of the search process. In the global search process, the objective is to find the best possible solution, no matter what are the initial conditions of the search process. When there is an enumerable universe of possible combinations of elements, one intends to minimize or maximize, there is a class of combinational optimization characterized by the search strategy employed, use of information on the problem domain and complexity. The GAs (Goldberg, 1997), SA (Kirkpatrick, Jr, & Vecchi, 1983) and TS (Glover & Laguna, 1997) are iteractive algorithms that, in general, are used to solve combinational optimization problems.

Algorithm 1. Optimization methodology for MLP neural networks with AAFFFP

```
1: $s_0 \leftarrow$ initial solution
2: $T_0 \leftarrow$ initial temperature
3: Update $s_{best}$ with $s_0$ (best solution found so far)
4: for $i = 0$ to $I_{\max} - 1$ do
5:    if $i + 1$ is not a multiple of $I_T$ then
6:       $T_{i+1} \leftarrow T_i$
7:    else
8:       $T_{i+1} \leftarrow$ new temperature
9:       if stopping criteria is satisfied then
10:          Stop execution
11:      end if
12: end if
13: Generate a set of $K$ new solutions from $s_i$
14: Choose the best solution $s'$ from the set
15: if $f(s') < f(s_i)$ then
16:       $s_{i+1} \leftarrow s'$
17:    else
18:       $s_{i+1} \leftarrow s'$ with probability $e^{[f(s')-f(s_i)]/T_{i+1}}$
19:    end if
20:    Update $s_{best}$ (if $f(s_{i+1}) < f(s_{best})$)
21:    Keep the parameter of the AAFFFP contained in $s_{best}$
     constant and use the weights and bias as initial ones for
     training with the backpropagation learning algorithm with
     momentum and Levenberg–Marquardt learning algorithm.
22: endfor
```

The SA algorithm can be defined as a global search techniques, which approximates the maximum or minimum of an object function $f : S \rightarrow R$, over a finite set S. This algorithm was introduced in the literature by Kirkpatrick et al. (1983), which in turn was based on the ideas of Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953) about simulation of a system of particles experiencing temperature changes. Under perturbation, the system tries to find a balance point which minimizes the total energy.

To escape the local minima, the SA algorithm differentiates itself from the other search methods, by accepting a new solution that increases the cost (Dowsland, 1993). The search process consists of a sequence of iterations. Each iteration consists in randomly changing the current solution in order to create a new solution in its neighborhood. Once a new solution is created, the corresponding alteration in the cost function is calculated in order to determine if the new solution can be accepted. If the cost of a new solution is less than the cost of the current solution, the new solution is accepted. If the contrary occurs, the Metropolis criterion is verified (Metropolis et al., 1953), based on the Boltzmann probability. This probability is regulated by a parameter called temperature, which decreases during the optimization process. In this manner, the parameter T is referred to as the temperature and the temperature reduction process is called a cooling process.

In this paper, the cooling strategy chosen is the logarithmic cooling rule obtained in Belisle (1992). According to this rule, the new temperature equals the current temperature multiplied by a reduction factor given by $1/\log\left(\lfloor (i-1)/I_T \rfloor \times I_T + \exp(1)\right)$ where $\lfloor a \rfloor$ represents the whole part of the division. The initial temperature T_0, the number of functions evaluated at each temperature, I_T, and the maximum number of iterations, $I_{\max}$, are parameters of the implementation. In many cases, the method may take a very long time to converge if the temperature reduction rule is too slow. However, a slow rule is often necessary, in order to allow an efficient exploration in the search space.

Tabu search tries to avoid this limitation by evaluating many new solutions in each iteration, instead of only one solution, as performed by simulated annealing. The best solution (i.e., the one with lower cost) is always accepted as the current solution. This strategy makes tabu search faster than simulated annealing, but it demands implementing a list containing a set of recently visited solutions (the tabu list) in order to avoid the acceptation of previously evaluated solutions. Using the tabu list for comparing new solutions to the prohibited (tabu) solutions increases the computational cost of tabu search when compared to simulated annealing (Glover & Laguna, 1997).

The pseudo-code of the methodology is presented in the Algorithm 1. A set of new solutions is generated each iteration, and the best one is selected according to the cost function, as performed by tabu search. However, the best solution is not always accepted since this decision is guided by a probability distribution, which is the same used by simulated annealing. During the execution of the methodology, the activation function and the weights are optimized, and the best solution found so far is stored. At the end of this process, the MLP architecture contained in is kept constant, and the weights are taken as the initial ones for training with the local learning algorithm, in order to perform a fine-tuned local search. Two algorithms were chosen: backpropagation with term momentum and the Levenberg Marquardt. The backpropagation with term momentum (Rumelhart et al., 1986) was chosen for being one of the connecting models most used in the literature (Haykin, 2001) and the Levenberg–Marquardt was chosen for being an algorithm designed for rapid training without using a Hessian matrix (Hagan & Menhaj, 1994). The original description of the Levenberg–Marquardt learning algorithm is given in Marquardt (1963).

An important factor is the definition of the network topology. For these reasons, several studies are being carried out for network architecture optimization. Nevertheless, in this study, in order to be able to evaluate the real effects of the activation function optimization combined with the weights, we opted for setting an architecture with few hidden nodes, thus decreasing the complexity of the network.

In this study, the MLP architecture has only one hidden layer, containing all the possible connections between adjacent layers, without any connections between non-adjacent layers. As a result, the quantity of connections is yielded by

$$N = pq + qm$$

where p is the number of input nodes, q is the number of hidden nodes and m is the number of output nodes.

Considering a set of solutions S and a real cost function f, the methodology used seeks the global minimum s, so that $f(s) \leqslant f(s')$, $\forall s' \in S$. The initial solution s_0 is a MLP network with a predefined MLP architecture with a maximum of 4 hidden nodes. The activation function in the initial solution of FFAAPL is the function with the parameter $\lambda = 1$ which represents the logit function. The initial weights are randomly extracted from a uniform distribution $U(0,1)$. The cost function is defined by $f(s) = \frac{1}{n}\sum_{j=1}^{n} e_j(s)$, where $e_j = -t_j - y_j$, t_j and y_j represent, respectively, the desirable output value and the output value of the network associated with the j-ith output unit and the training pattern i. The process ends after $I_{\max}$ iterations or if the stopping criterion based on the validation set is satisfied. Accordingly, the best solution s_{best} found is returned. The cooling scheme refreshes the temperature T_i of the iteration i at each I_T iteration of the algorithm. At each iteration, K new solutions are generated from the current one. Each solution contains information about the weights of the network MLP and, in the case of the AAFFFP, the parameter value is λ.

5. Experiment results

In these experiments, we used a combination of the global optimization techniques SA and TS in order, to optimize the parameter λ of the AAFFFP family ($\mathcal{F}_\lambda$) and the weights and bias of the neural network. Subsequently, we will present the description of the data sets, the parameters chosen for the optimization methodology experiments and the results found in these experiments.

5.1. Description of data sets

In order, to show evidence of the efficacy of the neural network models, that have an asymmetric activation functions family with free parameter, we used six time series data sets with non-linear behaviors. In the time series presented, there are characteristics (a priori) important for the modeling, such as seasonality and constant trend. There are also time series with behaviors irregular behaviors, in other words, non-stationary, non-seasonal or an additive and multiplicative seasonal, non-Gaussian and time series that do not present a stochastic trend. These examples have been used as benchmarks in the literature for time series forecasting. Table 1 shows the descriptive statistics for the time series used in the experiments.

5.1.1. Airline passenger data set

The first series corresponds to the logarithm of the total of passengers in an international airline from January, 1949 to December, 1960 (Airline series). The Airline series corresponds to the classic data used by Box and Jenkins (1990) and by Ghiassi, Saidane, and Zimbra (2005) in the DAN2 models. The Airline series in it is original form displays non-linear behavior and exhibits a multiplicative seasonal behavior. For this reason, the data is transformed through logarithmic transformation to convert the multiplicative seasonality to an additive one. This series has 144 observations and, as in various researches involving this time series, we used data from the first 11 years (132 observations) in order to adjustment the model (training set) and the last 12 observations for forecasting (test set).

5.1.2. USAccDeaths data set

The second series corresponds to the monthly quantity of death-resulting accidents in the United States in the period between January, 1973 and December, 1978 (USAccDeaths series). These data were used by Brockwell and Davis (1986). The USAccDeaths series displays a similar behavior to the transformed Airline series, though it does not demonstrate an increasing trend. Consequently, performing alterations in the data is unnecessary. This series has 72 observations. For the network training, we used the first 5 years (60 observations) and for the network testing, the last 12 observations.

5.1.3. WWWusage data set

The third series corresponds to the number of users connected to the Internet through a server every minute (WWWusage series), in relation to 100 min (100 observations). In the analysis of these data by the authors Makridakis, Wheelwright, and Hyndman (2005), this series is non-stationary. For the network training we used the first 88 min (88 observations) and for network testing, the last 12 observations.

5.1.4. Lynx data set

The fourth series corresponds to the number of Canadian lynx trapped per year in the Mackenzie River district of Northern Canada, for the period 1821 to 1934 (114 observations). This series can be obtained in Brockwell and Davis (1991) and was studied by Zhang (2003). For the network training, we used the first 102 observations and for testing it, the last 12 observations.

5.1.5. Nile data set

The fifth series corresponds to the measurements of the annual flow of the river Nile at Ashwan in the period between 1871 and 1970 (100 observations). This series was used by Balke (1993). For training the network, we used the first 88 measurements and for testing the network, the last 12 measurements.

Table 1
Descriptive statistics of time series.

Time series	N	Minimum	Maximum	Mean	Std. deviation	Skewness
Airline	144	4.64	6.43	5.54	0.44	−0.12
USAccDeaths	100	83.00	228.00	137.08	40.00	0.39
Lynx	72	6892.00	11317.00	8788.79	957.75	0.35
WWWusage	100	456.00	1370.00	919.35	169.23	0.33
Nile	114	39.00	6991.00	1538.02	1585.84	1.37
PetroPrice	192	0.08	0.13	0.10	0.01	−0.13

5.1.6. PetroPrice data set

Finally, the sixth and last series corresponds to the petrol price in Great Britain in the period from January, 1969 to December, 1984 (PetroPrice series). This series has 198 observations. The first 168 observations were used to adjustment the models and the remaining 12 observations for forecasting. This series was used by Gomes et al. (2006).

5.2. Lag selection

For the Airline, USAccDeaths, WWusage, Lynx and PetroPrice time series, we executed autoregressive models (AR) (Box & Jenkins, 1990) in order to select the lags. The quantity selected was used as input nodes in all the evaluated models. For the Nile series, the lag number selected by the AR model was insufficient for the adjustment and the forecasting of the DAN2 model. For this reason, the number of lags chosen is eight. The selected values are: $lag = \{5,3,4,4,8,3\}$ for the time series Airline, USAccDeaths, WWWusage, Lynx, Nile, PetroPrice, respectively.

The AR model was chosen by convenience, although a linear model. Therefore, the lag does not reflect the nonlinear structure. A better way is to use the NN to select these lags. However, main focus in this paper is not on the best lag structure.

5.3. Experimental parameters

In the global optimization technique, which combines SA and TS (SA + TS), the initial temperature equals 1 and the temperature decreased at each ten iterations of the optimization algorithm according to equation (10). The maximum number of iterations permitted equals 10,000. These values were chosen empirically. 100 executions of the algorithm were carried out with different random initializations of a uniform distribution $U(0,1)$ for weights and bias. The lambda value was initialized with 1, which represents the logit function. For each initialization, 10 executions of SA + TS were carried out and average values were obtained from these 10 executions. The stopping criterion GL_5 defined in Proben1 (Prechelt, 1994) was also used.

The performance of SA algorithm is influenced by the choice of the cooling scheme and by the choice of mechanism for new solutions generations. Yet, there are no objective rules to adjustment the configuration in order to obtain the best possible results. Normally, different configurations of the parameters for evaluating performance are employed (Sexton, Dorsey, & Johnson, 1999). Hence, the configuration used in this work was empirically chosen and may not be optimal for the problem approached. The purpose of this approach is demonstrating that the SA algorithm reached good results for the optimization problem presented, despite its difficulty for parameter adjustment.

To verify if the performance of the final networks generated by SA + TS could be improved, the values of the final connections and of the parameter λ were used in the MLP networks. They were trained through the backpropagation with term momentum and Levenberg–Marquardt algorithms, which correspond to the following configurations SA + TS + BPM and SA + TS + LM, respectively. The training is concluded when 10,000 epochs are reached, or if the validation error increases by 5 consecutive epochs. The learning rate and the term momentum used were of 0.001 and 0.9, respectively. All MLP topologies have a single hidden layer, containing only connections between adjacent layers. The experimental parameters used in this paper, which were chosen after some preliminary experiments, may not have been optimal for the problem. A more rigorous parameter exploration may have generated better results, but this paper does not intend to present an exhaustive exploration of the adjustable parameters. This paper aims to show that good results have been achieved by SA + TS with the addition of a training phase with a learning algorithm. For further details on the architecture of the networks used in the different data sets and of the average value found for the optimized parameter (λ) of each series, refer to Table 2.

As an evaluation criterion of the models, we used the mean square error (MSE) and mean absolute percentage error of forecast (MAPE).

The results and discussions will be presented next. These results will be discussed in a general and also in blocks organized in the following manner:

- **first block:** SA + TS (Aranda), SA + TS (Logit) and SA + TS (Cloglog) models;
- **second block:** SA + TS + BPM (Aranda), SA + TS + BPM (Logit) and SA + TS + BPM (Cloglog) models; and
- **third block:** SA + TS + LM (Aranda), SA + TS + LM (Logit) and SA + TS + LM (Cloglog) models.

5.4. Results and discussions

We present summary data, only for the test set, due to the volume of data. In Table 3, we present the results of the average performances of the ARIMA, AR and DAN2 models and the average performances and the respective standard deviations referring to the 100 initializations of the SA + TS models, SA + TS + BPM models and the SA + TS + LM models. The latter three models were executed with the Aranda activation function with λ free parameter, with the $\lambda = 1$ parameter (which corresponds to the logit function) and $\lambda \to 0$ (which corresponds to the complementary log–log function). The ARIMA, AR and DAN2 models do not display standard deviations because they do not possess random initialization.

For the ARIMA model, we tried different models of the ARIMA $(p,1,q)$ kind, varying $p = 0,1,2,3,4$ and $q = 0,1,2,3,4$. The best model of each time series was selected through the smallest AIC (Akaike Information Criterion), which is the most commonly used criterion (Mills, 1990). There are other criteria, for example, information criteria BIC (Bayesian Information Criterion). Unlike Akaike Information Criteria, BIC is derived within a Bayesian framework as an estimate of the Bayes factor for two competing models. However, main focus in this paper is not on the best information criteria.

In Table 4, we present the p-values of the t-Student tests[1] for the MSE and MAPE measures. These tests compare the average perfor-

[1] The t-Student test is a parametric test used in statistics in order to compare two or more independent samples aiming at verifying the existence of statistically significant difference between the metric averages of these samples (Lehman, 1986).

Table 2
Architectural details of the networks used.

Activation function	λ	Architecture	No. of adjustable parameters	Activation function	λ	Architecture	No. of adjustable parameters
Airline series				*Lynx series*			
Aranda	2.11	5 – 2 – 1	16	Aranda	1.76	4 – 4 – 1	26
Logit	1	5 – 2 – 1	15	Logit	1	4 – 4 – 1	25
Cloglog	→0	5 – 2 – 1	15	Cloglog	→ 0	4 – 4 – 1	25
USAccDeaths series				*Nile series*			
Aranda	1.97	3 – 3 – 1	17	Aranda	1.87	8 – 4 – 1	42
Logit	1	3 – 3 – 1	16	Logit	1	8 – 4 – 1	41
Cloglog	→0	3 – 3 – 1	16	Cloglog	→ 0	8 – 4 – 1	41
WWWusage series				*PetroPrice series*			
Aranda	3.94	4 – 4 – 1	26	Aranda	1.15	3 – 4 – 1	22
Logit	1	4 – 4 – 1	25	Logit	1	3 – 4 – 1	21
Cloglog	→0	4 – 4 – 1	25	Cloglog	→ 0	3 – 4 – 1	21

Table 3
Results of the average performance to forecasting (test set) for the six time series.

Model	Airline series				USAccDeaths series			
	MSE		MAPE		MSE		MAPE	
	Average	SD	Average	SD	Average	SD	Average	SD
A – ARIMA (4,0,2)	0.02634	–	2.31566	–	942386	–	9.47	-
B – AR (5)	0.03067	–	2.38605	–	1109648	–	10.22	–
C – DAN2	0.00957	–	1.36551	–	406723	–	6.25	–
D – SA + TS (Aranda)	0.01669	1.46E-03	1.87142	1.42E-02	301938	4.90E + 02	5.06	4.87E-03
E – SA + TS + BPM (Aranda)	0.01243	1.34E-04	1.46304	1.09E-02	273074	7.17E + 04	4.70	8.66E-01
F – SA + TS + LM (Aranda)	0.01065	3.15E-04	1.38931	2.83E-02	251394	9.78E + 04	4.55	1.03E + 00
G – SA + TS (Logit)	0.01376	5.13E-05	1.56788	2.53E-03	321085	6.27E + 02	5.35	4.28E-03
H – SA + TS + BPM (Logit)	0.01552	4.61E-04	1.59407	2.17E-02	314162	2.99E + 04	5.19	4.90E-01
I – SA + TS + LM (Logit)	0.01109	2.00E-04	1.43849	1.68E-02	308305	4.27E + 04	4.99	4.27E-01
J – SA + TS (Cloglog)	0.01388	7.08E-05	1.57521	2.73E-03	377373	2.58E + 02	5.75	2.97E-03
K – SA + TS + BPM (Cloglog)	0.01733	3.32E-04	1.75207	1.58E-02	335508	4.91E + 04	5.34	4.97E-01
L – SA + TS + LM (Cloglog)	0.00962	3.95E-04	1.30864	2.42E-02	324220	6.64E + 04	5.26	5.63E-01
	WWWusage series				Lynx series			
	MSE		MAPE		MSE		MAPE	
	Average	SD	Average	SD	Average	SD	Average	SD
A – ARIMA (4,0,1)	375.61	–	7.55	–	1631166	–	77.24	–
B – AR (4)	3661.64	–	24.31	–	1638461	–	77.95	–
C – DAN2	12.43	–	1.51	–	146393	–	21.77	–
D – SA + TS (Aranda)	109.13	8.65E-01	4.11	1.62E-02	118519	2.66E + 03	18.35	3.01E-01
E – SA + TS + BPM (Aranda)	25.17	8.88E + 00	2.05	1.65E-01	112365	1.33E + 03	17.48	9.02E-02
F – SA + TS + LM (Aranda)	12.06	2.69E + 00	1.47	9.07E-02	59414	8.87E + 02	15.09	2.91E-02
G – SA + TS (Logit)	228.96	2.57E + 00	5.40	3.06E-02	175619	1.68E + 03	22.67	2.29E-01
H – SA + TS + BPM (Logit)	50.37	4.12E + 01	3.00	3.14E-01	161573	1.08E + 03	18.83	6.73E-02
I – SA + TS + LM (Logit)	17.25	1.33E + 01	1.53	2.04E-01	148846	1.96E + 03	17.20	9.49E-02
J – SA + TS (Cloglog)	62.78	5.96E-01	3.07	1.44E-02	172587	1.64E + 03	20.01	2.27E-01
K – SA + TS + BPM (Cloglog)	13.83	2.63E + 01	1.46	3.45E-01	165572	1.22E + 03	18.02	9.42E-02
L – SA + TS + LM (Cloglog)	16.37	1.31E + 01	1.52	2.46E-01	115527	1.22E + 03	17.05	5.43E-02
	Nile series				PetroPrice series			
	MSE		MAPE		MSE*		MAPE	
	Average	SD	Average	SD	Average	SD	Average	SD
A – ARIMA (3,0,2)	25948	–	13.58	–	0.00794	–	0.65	–
B – AR (4)	21880	–	13.77	–	0.19507	–	3.29	–
C – DAN2	14601	–	12.10	–	0.00016	–	0.78	–
D – SA + TS (Aranda)	19759	1.29E + 01	13.53	4.46E-03	0.00013	9.20E-08	0.75	3.73E-04
E – SA + TS + BPM (Aranda)	17005	5.80E + 01	12.85	4.24E-02	0.00013	9.90E-08	0.70	8.97E-04
F – SA + TS + LM (Aranda)	13062	1.64E + 02	11.75	9.41E-02	0.00012	3.00E-08	0.69	9.18E-05
G – SA + TS (Logit)	18894	5.71E + 00	13.53	6.78E-04	0.00014	9.50E-08	0.79	3.41E-04
H – SA + TS + BPM (Logit)	19050	7.67E + 01	13.12	5.15E-02	0.00014	1.67E-07	0.73	9.14E-04
I – SA + TS + LM (Logit)	16994	2.38E + 02	12.43	6.80E-02	0.00013	1.27E-07	0.65	5.55E-04
J – SA + TS (Cloglog)	19113	5.86E + 00	13.56	1.46E-03	0.00013	2.22E-07	0.72	8.16E-04
K – SA + TS + BPM (Cloglog)	17360	4.09E + 02	12.98	1.70E-01	0.00013	2.99E-07	0.72	9.27E-04
L – SA + TS + LM (Cloglog)	13341	2.52E + 02	11.83	1.03E-01	0.00013	6.10E-08	0.70	3.43E-04

SD – Standard deviation.
* Values multiplied by 10^4 for better visualization.

Table 4
Results of the *p*-values of *t*-Student tests with significance level of 5% for the MSE and MAPE measures.

Time series	$\mu_E = \mu_H$	$\mu_E = \mu_K$	$\mu_H = \mu_K$	$\mu_F = \mu_I$	$\mu_F = \mu_L$	$\mu_I = \mu_L$
MSE						
Airline	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
USAccDeaths	0.0000	0.0000	0.0003	0.0000	0.0000	0.0452
WWWusage	0.0000	0.0001	0.0000	0.0002	0.0015	0.6374
Lynx	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Nile	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
PetroPrice	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
MAPE						
Airline	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
USAccDeaths	0.0000	0.0000	0.0261	0.0001	0.0000	0.0002
WWWusage	0.0000	0.0000	0.0000	0.0083	0.0589	0.7622
Lynx	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Nile	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
PetroPrice	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

mances between the models of the second block and between the models of the third block, shown previously. Therefore, the comparison between the SA + TS + BPM (Aranda) and SA + TS + BPM (Logit) models will be represented by the null hypothesis $\mu_E = \mu_H$, between the SA + TS + BPM (Aranda) and SA + TS + BPM (Cloglog) models it will be represented by the null hypothesis $\mu_E = \mu_K$, between the SA + TS + BPM (Logit) and SA + TS + BPM (Cloglog) models it will be represented by the null hypothesis $\mu_H = \mu_K$, between the SA + TS + LM (Aranda) and SA + TS + LM (Logit) it will be represented by the null hypothesis $\mu_F = \mu_I$, between the SA + TS + LM (Aranda) and SA + TS + LM (Cloglog) models, it will be represented by the null hypothesis $\mu_F = \mu_L$, between the SA + TS + LM (Logit) and SA + TS + LM (Cloglog) models, it will be represented by the null hypothesis $\mu_I = \mu_L$. In order, to verify if the difference between the averages is statistically significant, the *p*-value must be lower than the relevance level α. In this paper, the value of α is equal to 5% (or 0.05).

For the Airline series, we can observe that the performance improves with the employment of the SA + TS + BPM and SA + TS + LM models, independently the activation function. However, the SA + TS + LM (Aranda) model presented the best result in relation to the other models, except in comparison to the DAN2 model, in which the results were equivalent (Table 3).

For the USAccDeaths series, all the SA + TS, SA + TS + BPM and SA + TS + LM models display average performances superior in relation to the ARIMA, AR and DAN2 models. We can see that the models that combine the global and local optimization techniques show improvements in their average results, in relation to the models that utilize only global optimization techniques. The SA + TS + LM (Aranda) model showed the best result (Table 3).

For the WWWusage series, we can observe that the models that combine global and local optimization techniques demonstrate improvements in their average results, when compared to the models that only use global optimization techniques. The DAN2 model surpassed all the models in question, except the SA + TS + LM (Aranda) model. Therefore, the performance of the SA + TS + LM (Aranda) model was superior in relation to the performance of all other models (Table 3).

For the Lynx series, the SA + TS + BPM and the SA + TS + LM models presented substantial improvement in their average results, when compared to the models that only use global optimization techniques (SA + TS). We observe that only the models with Aranda activation function demonstrated better performances than the ARIMA, AR and DAN2 models. The SA + TS + LM (Aranda) model displayed the best result in relation to the other models (Table 3).

For the Nile series, only the SA + TS + LM (Aranda) and SA + TS + LM (Cloglog) models presented superior average performances than those presented by the DAN2 model. It is worth mentioning that the models that combine global and local optimization techniques display improvements in their average results, when compared to the models that use only global optimization techniques. It is also worth mentioning that the performance of the SA + TS + LM model was better than the performance of the other models (Table 3).

Finally, for the PetroPrice series, we observed that the SA + TS + BPM and the SA + TS + LM models display substantial improvements in their average performances, when compared to the SA + TS models. We further noticed that all the average performances obtained by the models with the used methodology were superior in relation to the performances obtained by the ARIMA, AR and DAN2 models. The performance of the SA + TS + LM model was better to the performance of all other models (Table 3).

6. Conclusions

The results presented show that the models, which combine global and local optimization techniques, display improvements in their average results compared to the models that use only global optimization techniques. Results also show that the SA + TS + LM (Aranda) model performance, in the six examples of time series studied, was superior to the performance of all other models, including the DAN2 models, which are extremely efficient in the time series forecasting.

For all the problems approached, when comparing the models per block, we observe that the models with an Aranda activation function displayed better average performances than the models with logit and complementary log–log activation functions, and this difference is statistically significant since all the *p*-values in Table 4 are lower than 0.05.

Therefore, it is reasonable to conclude that the employment of a methodology combining the main favorable characteristics of the SA and TS algorithms, using a local learning algorithm, is capable of producing very satisfactory results for optimization of the activation function and of the weights of the MLP networks, for the time series problems approached. It is worth mentioning that all the results exhibited may not have been optimal for each problem, in other words, it is possible that the ARIMA, AR and DAN2 models reach better results than the ones presented in this study, by simply changing the lag numbers. However, the aim of this paper is to show that it is possible to improve the results for time series forecasting on using SA and TS for the optimization of MLP connection weights and activation functions simultaneously, generating networks with low complexity and high generalization performance.

As future work, the methodology to optimize, simultaneously, the weights, the activation function and neural network architectures could be used to improve the cost function.

Acknowledgments

The authors thank CNPq, CAPES and FACEPE (Brazilian Agencies) for their financial support.

References

Aranda-Ordaz, F. J. (1981). On two families of transformations to additivity for binary response data. *Biometrika, 68*, 357–364.
Balke, N. S. (1993). Detecting level shifts in time series. *Journal of Business and Economic Statistics, 11*, 81–92.
Belisle, C. J. P. (1992). Convergence theorems for a class of simulated annealing algorithms on R^d. *Annals of Applied Probability, 29*, 885–895.
Box, G. E. P., & Jenkins, G. (1990). *Time series analysis, forecasting and control*. Holden-Day. Incorporated.

Brockwell, P. J., & Davis, R. A. (1986). *Time series: Theory and methods*. New York, NY, USA: Springer-Verlag New York, Inc.

Brockwell, P. J., & Davis, R. A. (1991). *Time series and forecasting methods* (2nd ed.). New York, NY, USA: Springer. Series G.

Chandra, P. (2003). Sigmoidal function classes for feedforward artificial neural networks. *Neural Processing Letters, 18*(3), 205–215.

Chandra, P., & Singh, Y. (2004). An activation function adapting training algorithm for sigmoidal feedforward networks. *Neurocomputing, 61*, 429–437.

Chandra, P., & Singh, Y. (2004). A case for the self-adaptation of activation functions in ffanns. *Neurocomputing, 56*, 447–454.

Chen, M. H., Dey, D. K., & Shao, Q. M. (1999). A new skewed link model for dichotomous quantal response data. *Journal of the American Statistical Association, 94*(448), 1172–1186.

Dowsland, K. A. (1993). Simulated annealing (pp. 20–69).

Duch, W., & Jankowski, N. (1999). Survey of neural transfer functions. *Neural Computing, 2*, 163–212.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica, 50*(4), 987–1007.

Ferreira, A. A., Ludermir, T. B., & Aquino, R. R. B. (2013). An approach to reservoir computing design and training. *Expert Systems With Applications, 40*(10), 4172–4182.

Funahashi, K. (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks, 2*(3), 183–192.

Gepperth, A., & Roth, S. (2006). Applications of multi-objective structure optimization. *Neurocomputing, 69*(7-9), 701–713.

Ghiassi, M., & Saidane, H. (2005). A dynamic architecture for artificial neural network. *Neurocomputing, 63*, 397–413.

Ghiassi, M., Saidane, H., & Zimbra, D. K. (2005). A dynamic artificial neural network model for forecasting time series events. *International Journal of Forecasting, 21*, 341–362.

Ghiassi, M., Saidane, H., & Zimbra, D. K. (2005). A dynamic artificial neural network model for forecasting time series events. *Journal of Forecasting, 21*.

Glover, F., & Laguna, M. (1997). *Tabu search*. Boston, USA: Kluwer Academic Publishers.

Goldberg, D. E. (1997). *Genetic algorithms in search optimization and machine learning*. Massachusets, USA: Addison-Wesley Co.

Gomes, G. S. da S., Ludermir, T. B., & Almeida, L. M. (2009). Neural networks with asymmetric activation function for function approximation. In *IEEE – INNS – ENNS international joint conference on neural networks* (Vol. 0, pp. 980–987).

Gomes, G. S. S., & Ludermir, T. B. (2008). Complementary log-log and probit: Activation functions implemented in artificial neural networks. *Eighth international conference on hybrid intelligent systems* (Vol. 1, pp. 939–942). Barcelona, Spain: Computer Society and IEEE.

Gomes, G. S. S., Ludermir, T. B., & Lima, L. M. M. R. (2011). Comparison of new activation functions in neural network for forecasting financial time series. *Neural Computing and Applications, 20*(3), 417–439.

Gomes, G. S., Maia, A. S., Ludermir, T., Carvalho, F. D., & Araujo, A. (2006). Hybrid model with dynamic architecture for forecasting time series. *International joint conference on neural networks* (Vol. 1, pp. 3742–3747). Vancouver, Canadian.

Guarnieri, S., Piazza, F., & Uncini, A. (1999). Multilayer feedforward networks with adaptive spline activation function. *IEEE-NN, 10*(3), 672–683.

Hagan, M. T., & Menhaj, M. (1994). Training feed-forward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks, 5*(6), 989–993.

Hamm, L. Brorsen, B. W., & Hagan, M. T. (2002). Global optimization of neural network weights. In *Proceedings of international joint conference on neural networks* (Vol. 2, pp. 1228–1233).

Haykin, S. (2001). *Neural networks: A comprehensive foundation* (2nd ed.). New Jersey: Prentice Hall.

Hornik, K. (1993). Some new results on neural network approximation. *Neural Networks, 6*(9), 1069–1072.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2*(5), 359–366.

Khashei, M., & Bijari, M. (2012). A new class of hybrid models for time series forecasting. *Expert Syst. Appl., 39*(4), 4344–4357.

Kirkpatrick, S., Jr, C. D. G., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*(4598), 671–680.

Lehman, E. (1986). *Testing statistical hypothesis* (2nd ed.). New York: John Wiley & Sons.

Leshno, M., Lin, V. Y., Pinkus, A., & Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks, 6*(6), 861–867.

Leung, H., & Haykin, S. (1993). Rational function neural network. *Neural Computation, 5*(6), 928–938.

Li, W. D., Ong, S. K., & Nee, A. Y. C. (2002). Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts. *International Journal of Production Research, 40*(8), 1899–1922.

Ludermir, T. B., Yamazaki, A., & Zanchettin, C. (2006). An optimization methodology for neural network weights and architectures. *IEEE Transactions on Neural Networks, 17*(6), 1452–1459.

Ma, L., & Khorasani, K. (2005). Constructive feedforward neural networks using hermite polynomial activation functions. *IEEE Transactions on Neural Networks, 16*.

Makridakis, S., Wheelwright, S. C., & Hyndman, R. J. (2005). Forecasting: Methods and applications. *Neurocomputing, 63*, 397–413.

Mantawy, A. H., Abdel-Magid, Y. L., Selim, S. Z., algorithms, Integrating genetic, & search, tabu (1999). and simulated annealing for the unit commitment problem. *IEEE Transactions on Power Systems, 14*(3), 829–836.

Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics, 11*, 431–441.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics, 21*, 1087–1092.

Mills, T. D. (1990). *Times series techniques for economists*. Cambridge: Cambridge University Press.

Nelder, J. A., & Wedderburn, R. W. M. (1972). *Journal of the Royal Statistical Society. Series A (General), 135*(3), 370–384.

Prechelt, L. (1994). Proben1 – a set of neural network benchmark problems and benchmarking rules, Tech. rep., Fakultat Fur Informatik.

Rosen-Zvi, M., Biehl, M., & Kanter, I. (1998). Learnability of periodic activation functions: General results. *Physical Review E, 58*(3), 3606–3609.

Rumelhart, D., Hinton, G., & Williams, R. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition*. Cambridge, MA, USA: MIT Press.

Sahin, S., Tolun, M. R., & Hassanpour, R. (2012). Hybrid expert systems: A survey of current approaches and applications. *Expert Syst. Appl., 39*(4), 4609–4617.

Sexton, R. S., Dorsey, R. E., & Johnson, J. D. (1999). Optimization of neural networks: A comparative analysis of the genetic algorithm and simulated annealing. *European Journal of Operational Research, 114*(3), 589–601.

Singh, Y., & Chandra, P. (2003). A class +1 sigmoidal activation functions for FFANNs. *Journal of Economic Dynamics and Control, 28*(1), 183–187.

Ting, C. K., Li, S. T., & Lee, C. (2003). On the harmonious mating strategy through tabu search. *Information Sciences, 156*(3-4), 189–214.

Tsai, J. T., Chou, J. H., & Liu, T. K. (2006). Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm. *IEEE Transactions on Neural Networks, 17*(1), 69–80.

Yamazaki, A., & Ludermir, T. B. (2003). Neural network training with global optimization techniques. *International Journal of Neural Systems, 13*, 77–86.

Zanchettin, C., Ludermir, T. B., & Almeida, L. (2011). Hybrid training method for MLP: Optimization of architecture and training. *IEEE Transactions on Systems, Man and Cybernetics. Part B. Cybernetics, 41*, 1097–1109.

Zhang, G. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing, 50*, 159–175.