Predicting time series using neural networks with wavelet-based denoising layers

Article *in* Neural Computing and Applications · March 2005

DOI: 10.1007/s00521-004-0434-z · Source: DBLP

2 authors:

Uros Lotric
University of Ljubljana
36 PUBLICATIONS **232** CITATIONS
SEE PROFILE

Andrej Dobnikar
University of Ljubljana
69 PUBLICATIONS **485** CITATIONS
SEE PROFILE

Neural Networks with Wavelet Based Denoising Layers for Time Series Prediction

UROS LOTRIC[1] AND ANDREJ DOBNIKAR

University of Ljubljana, Faculty of Computer and Information Science, Slovenia, e-mail: {uros.lotric, andrej.dobnikar}@fri.uni-lj.si

Abstract

To avoid preprocessing of noisy data, two special denoising layers based on wavelet multiresolution analysis are integrated into the layered neural networks. A gradient based learning algorithm is developed which uses the same cost function for setting both the neural network weights and the free parameters of denoising layers. The proposed layers, integrated into feedforward and recurrent neural networks, are validated on time series prediction problems: the Feigenbaum sequence, the rubber hardness time series and the yearly average sunspot number. It is shown that the introduced denoising layers improve the prediction accuracy in both cases.

Keywords: feedforward and recurrent neural networks, wavelet multiresolution analysis, denoising, gradient based threshold adaptation, time series prediction

[1] Corresponding author: Uros Lotric, University of Ljubljana, Faculty of Computer and Information Science, Trzaska 25, 1000 Ljubljana, Slovenia, e-mail: uros.lotric@fri.uni-lj.si, phone: +386 1 4768 874, fax: +386 1 4768 369

1. Introduction

Models of dynamical systems in which the structure and parameters are extracted directly from real data cannot avoid the problem of noise. Therefore, noise reduction has become an important issue in modeling with neural networks [1]. Usually, noise reduction is treated as a separate problem and preprocessing methods such as filtering [1, 2] based on statistical criteria are used. However, attempts have been made to achieve noise reduction by integrating various performance measures into the cost function without influencing the input data [3, 4]. Another approach proposes to include methods for input data reduction in the neural network learning process [5].

In this paper two special denoising layers that enable neural networks to deal with noise without separate data preprocessing are introduced. These layers apply novel filtering methods originating in the wavelet multiresolution analysis [6]. Such methods have already been reported in neural network prediction problems [7], however, only as a preprocessing technique and not integrated into the model itself. At the same time, models which integrate wavelets' features in neural networks can be found in literature [8, 9].

The design of the proposed denoising layers enables for their integration into the majority of layered neural network. By this integration, the same cost function is used as the model performance criterion and as the noise level estimation criterion. The parameters of denoising layers are set in the process of learning and any gradient based learning algorithm can be applied. In contrast to methods using cost functions with additional terms [3, 4] the proposed denoising layers can be added to any neural network without rewriting its learning process equations.

2. Wavelet Based Denoising

In wavelet multiresolution analysis a data sample is treated on different scales. It is decomposed into coefficients of approximation on a coarser scale and coefficients of remaining details on finer scales [10]. One-dimensional wavelet multiresolution analysis is based on the scaling function $\phi(t)$ and the

corresponding mother wavelet $\psi(t)$ fulfilling certain technical conditions. By dilation and translation of both functions on abscissa the basis functions $\phi_{j,k} = 2^{-j/2}\phi(2^{-j}t-k)$ and $\psi_{j,k} = 2^{-j/2}\psi(2^{-j}t-k)$, $j,k \in \mathbf{Z}$, are derived. An arbitrary continuous data sample can be written as a superposition of these basis functions

$$p(t) = \sum_{k\in\mathbf{Z}} a_{J,k}\phi_{J,k}(t) + \sum_{j\le J}\sum_{k\in\mathbf{Z}} d_{j,k}\psi_{j,k}(t) \quad , \tag{1}$$

where the sum with coefficients $a_{J,k}$, $k \in \mathbf{Z}$, represents the approximation of the data sample on the scale J and the sum with coefficients $d_{j,k}$, $k \in \mathbf{Z}$, represents the details of the data sample on scale j.

In further discussion, only discrete data samples $\mathbf{p}=(p_1, p_2, ..., p_N)^T$ are considered. The discrete wavelet transform and its inverse are linear operations, and as such, they can be written in matrix form as

$$\mathbf{c} = \mathbf{W}^D\mathbf{p} \quad and \quad \mathbf{p} = \mathbf{W}^R\mathbf{c} \quad , \tag{2}$$

respectively, where the C-dimensional vector $\mathbf{c} = (d_{1,1}, ..., d_{1,C_1}, ..., d_{J,1}, ..., d_{J,C_J}, a_{J,1}, ..., a_{J,C_J})^T$ represents the discrete wavelet transform of the data sample. The decomposition matrix $\mathbf{W}^D$ and the reconstruction matrix $\mathbf{W}^R$ can be constructed by applying the pyramidal convolution schemes [11].

As the wavelet coefficients of details having small absolute values are likely to contribute to noise, Donoho and Johnstone [6] proposed denoising by thresholding. Denoising is a three-step process, composed of discrete wavelet transformation of a data sample to the space of wavelet coefficients, scale dependent thresholding of the wavelet coefficients of details, and construction of a denoised data sample from thresholded wavelet coefficients in terms of inverse discrete wavelet transform.

Following the work of Donoho and Johnstone [6], the generalized soft thresholding function

$$T(d_{j,k},\tau_j)=d_{j,k}+\frac{1}{2}\left(\sqrt{(d_{j,k}-\tau_j)^2+s}-\sqrt{(d_{j,k}+\tau_j)^2+s}\right) \quad (3)$$

was introduced [12] with parameter $s \geq 0$ determining the level of smoothness (Figure 1).

Figure 1

The function removes the wavelet coefficients of details $d_{j,k}$ smaller than threshold τ_j and reduces the absolute values of larger wavelet coefficients of details. With the threshold τ_j any level of denoising can be chosen, from no denoising by setting $\tau_j = 0$ up to complete removal of information by setting $\tau_j = d_{j,\max}$, with $d_{j,\max}$ being the value of the absolutely largest wavelet coefficient of details on the j-th scale [12].

3. Denoising Layers

In neural networks terminology the wavelet denoising process can be described by two layers, the thresholding layer and the reconstruction layer, presented in Figure 2.

Figure 2

The thresholding layer has C computational units or thresholding neurons, split into $J+1$ groups, differing in thresholds τ_j. The number of scales J equals to the largest integer smaller than or equal to $\log_2 N$. While the thresholds of the first J groups, processing the wavelet coefficients of details, are allowed to change, the threshold of the last group, representing the approximation on scale J, is fixed to zero. The thresholding neuron is essentially the standard neuron [13] with nonlinear sigmoid activation function replaced by generalized soft thresholding function. When the input data sample $\mathbf{p}(q)$ at some discrete time q

is presented to the denoising layers, the output of the k-th thresholding neuron belonging to the group j is given by equations

$$\tilde{c}_k(q) = T(c_k(q), \tau_j) \quad , \quad c_k(q) = \sum_{l=1}^{N} \mathbf{W}^D_{k,l} p_l(q) \quad , \tag{4}$$

where the weighted sum $c_k(q)$ represents one of the wavelet coefficients ($a_{j,k}$ or $d_{j,k}$, $j \in$ □ , $k \in \mathbf{Z}$) and the thresholding neuron output $\tilde{c}_k(q)$ represents the matching thresholded wavelet coefficient. Thresholded wavelet coefficients are further passed to the reconstruction layer having N neurons with linear activation functions, which generate the denoised sample $\tilde{\mathbf{p}}(q)$ with elements

$$\tilde{p}_k(q) = \sum_{l=1}^{C} \mathbf{W}^R_{k,l} \tilde{c}_l(q) \quad , \quad k = 1, ..., N \quad . \tag{5}$$

3.1. Gradient Based Learning Algorithm for Thresholds

The goal of learning is to adapt the model's free parameters in order to minimize a cost function. When gradient based algorithms are used, the gradients of the cost function or its additive parts with respect to the model's free parameters need to be expressed analytically [14].

Opposed to standard neurons with adaptable weights, the weights of thresholding neurons are fixed as given by the decomposition matrix $\mathbf{W}^D$. Similarly, the weights of linear neurons in reconstruction layer are given by the reconstruction matrix $\mathbf{W}^R$. Hence, the only adaptable parameters of the denoising layers are the thresholds, determining the shape of thresholding neurons' activation functions. Although the thresholding function with parameter $s = 0$ exhibits nice statistical properties [6], it can lead to a situation where gradient based adaptation of thresholds is unintentionally stopped [12]. Therefore, the parameter $s = 0.01 \cdot d^2_{j,\max} > 0$, found empirically, is used in thresholding neuron activation functions. Moreover, to comply with the gradient based algorithms, unconstrained thresholds τ^{∞}_j, $j = 1, ..., J$, which are mapped to the bounded thresholds τ_j by the sigmoid function $\tau_j = d_{j,\max}/(1 + e^{-\tau^{\infty}_j})$, are introduced [12].

Let us suppose that the cost function depends on some function ε, which on the other hand depends on the denoised data sample, $\varepsilon=\varepsilon(\tilde{\mathbf{p}}(q),...)$. To calculate gradients of the function ε with respect to the thresholds, an approach similar to the one of Trentin [15] has been used. Having in mind that each element of the denoised data sample is obtained from all wavelet coefficients, one can write

$$\frac{\partial \varepsilon}{\partial \tau_j^{\infty}} = \sum_{i=1}^{N} \frac{\partial \varepsilon}{\partial \tilde{p}_i(q)} \frac{\partial \tilde{p}_i(q)}{\partial \tau_j} \frac{\partial \tau_j}{\partial \tau_j^{\infty}} \quad . \tag{6}$$

The first factor in the sum is obtained from the model to which the denoised data sample is passed. Considering the Equation (5) and the mapping of thresholds, the second and the third factor can be expressed as

$$\frac{\partial \tilde{p}_i(q)}{\partial \tau_j} \frac{\partial \tau_j}{\partial \tau_j^{\infty}} = \sum_{l=1}^{C} \mathbf{W}_{i,l}^{R} \frac{\partial T(c_l(q),\tau_j)}{\partial \tau_j} \tau_j (1-\tau_j/d_{j,\max}) \tag{7}$$

with

$$\frac{\partial T(c_l(q),\tau_j)}{\partial \tau_j} = -\frac{1}{2}\left(\frac{c_l(q)-\tau_j}{\sqrt{(c_j(q)-\tau_j)^2+s}} + \frac{c_l(q)+\tau_j}{\sqrt{(c_j(q)+\tau_j)^2+s}} \right). \tag{8}$$

4. Neural Network Model with Denoising Layers

The denoising layers are usually placed between the neural network model inputs and its first nonlinear layer. To demonstrate their universality, the integration of the denoising layers into the feedforward two-layered perceptron and the two-layered perceptron with global recurrent connections is presented. Only equations of the recurrent model are given, since they can be easily reduced to the feedforward version. The original models and their modifications are shown in Figure 3.

Figure 3

4.1. Two-Layered Perceptron with Recurrent Connections

The two-layered perceptron with global recurrent connections starts by merging the inputs $\mathbf{x}(q) = (x_1(q), ..., x_N(q))^T$ to the last known model outputs $\mathbf{y}^o(q-1)^T$. Obtained vector $\mathbf{z}(q) = (\mathbf{y}^o(q-1)^T, \mathbf{x}(q)^T)^T$ is passed further to the N_h nonlinear neurons in the hidden layer with outputs

$$y_k^h = \varphi^h(s_k^h(q)) \quad , \quad s_k^h(q) = \sum_{l=0}^{N_o+N} \omega_{k,l}^h z_l(q) \quad , \quad k = 1, ..., N_h \quad , \tag{9}$$

where $\varphi^h(s) = \tanh(s)$ is the nonlinear activation function, $\omega_{k,l}^h$ are the adaptive weights and the bias $z_0(q) = 1$. The outputs of the hidden layer are further fed to the N_o neurons in the output layer, giving the model outputs

$$y_k^o = \varphi^o(s_k^o(q)) \quad , \quad s_k^o(q) = \sum_{l=0}^{N_h} \omega_{k,l}^o y_l^h(q) \quad , \quad k = 1, ..., N_o \quad , \tag{10}$$

with the nonlinear activation $\varphi^o(s) = \tanh(s)$, the adaptive weights $\omega_{k,l}^o$ and the bias $y_0^h(q) = 1$.

Neural network models are commonly trained on a set of known input–output pairs, $\{\mathbf{p}(q), \mathbf{t}(q)\}$, with $\mathbf{t}(q)$ being the vector of target values. The cost function usually depends on the errors of each input–output pair, $e_k = t_k - y_k^o$, $k = 1, ..., O$, $O \le N_o$, and therefore its gradients can be expressed with the gradients of the model outputs.

According to the derivation of the Real Time Recurrent Learning [16], two dynamical systems are obtained for the two-layered perceptron with global recurrent connections, determining the gradients of outputs,

$$\rho_{i,j}^k(q) = \varphi^o(s_k^o(q)) \sum_{l=1}^{N_h} \omega_{k,l}^o \varphi^h(s_l^h(q)) \{\delta_{il} z_j(q) + \sum_{m=1}^{N_o} \omega_{l,m}^h \rho_{i,j}^m(q-1)\} \tag{11}$$

$$\pi_{i,j}^k(q) = \varphi^o(s_k^o(q)) \{\delta_{ik} y_j^h(q) + \sum_{l=1}^{N_h} \omega_{k,l}^o \varphi^h(s_l^h(q)) \sum_{m=1}^{N_o} \omega_{l,m}^h \pi_{i,j}^m(q-1)\} \quad , \tag{12}$$

with $\rho_{i,j}^{k}(q) = \partial y_k^o(q)/\partial \omega_{i,j}^h$ and $\pi_{i,j}^{k}(q) = \partial y_k^o(q)/\partial \omega_{i,j}^o$. In Equations (11) and (12), $\varphi^h(s)$ and $\varphi^o(s)$ denote the derivatives of the corresponding activation functions and δ_{ij} is the Kronecker symbol. At the beginning of the learning process $\rho_{i,j}^{k}(0) = 0$ and $\pi_{i,j}^{k}(0) = 0$ is set.

By setting the number of outputs N_o to zero in Equations (9), (11) and (12), the equations governing the two-layered perceptron without recurrent connections are obtained.

4.2. Integration of Denoising Layers

The two-layered perceptron with global recurrent connections and integrated denoising layers is shown in Figure 3b. The input sample $\mathbf{p}(q)$ is first passed to the thresholding layer and then further to the reconstruction layer in order to obtain the denoised sample $\tilde{\mathbf{p}}(q)$. The process is governed by Equations (4) and (5), respectively. The inputs of the two-layered perceptron with global recurrent connections are then substituted with the denoised data sample, $\mathbf{x}(q) = \tilde{\mathbf{p}}(q)$. Applying the Equations (9) and (10) the model outputs are finally computed.

As pointed out in section 1, the gradients of a cost function can be obtained from the gradients of the model outputs. The gradients of the model outputs with respect to the weights of the neurons in the hidden layer and to the weights of the neurons in the output layer are given by Equations (11) and (12). The gradients of outputs with respect to the thresholds are obtained from Equation (6) by setting the model output $y_k^o(q)$ as the function ε. In this case, the first factor in Equation (6) becomes

$$\frac{\partial y_k^o(q)}{\partial \tilde{p}_i(q)} = \varphi^o(s_k^o(q)) \sum_{l=1}^{N_h} \omega_{k,l}^o \varphi^h(s_l^h(q)) \omega_{l,N_o+i}^h \quad . \tag{13}$$

Again, the equations of the feedforward two-layered perceptron with the denoising layers are obtained by setting N_o to zero.

5. Results

The proposed denoising layers were integrated into the two-layered perceptron (TLP) and the two-layered perceptron with global recurrent connections (RTLP). The performance of the models enhanced with the denoising layers, the DTLP model and the DRTLP model, respectively, was tested on three one-step-ahead prediction problems: the logistic map, the hardness of rubber compound and the yearly average sunspot number.

The time series model usually connects a value of a time series on model output with N previous values presented on model inputs. To establish such connection, the model is trained on a set of known input–output pairs. From each time series, first 85% of input–output pairs were included in the training set, used to set free parameters of the models, while the remaining 15% of input–output pairs formed the testing set, used for model comparison. The optimal values of topological parameters, which cannot be included in the gradient based algorithms, were found by inspecting the free parameter space. The model configurations were allowed to have from 4 to 20 inputs (N) with the number of free parameters not exceeding 35% of the number of all input–output pairs. The wavelets from symlet family [10], S^M, $2M \leq N+1$, were used in the construction of the denoising layers.

As the performance measures, the root mean squared error, normalized to standard deviation of a time series, (NRMSE) and the mean absolute percentage error (MAPE) were considered. On each model configuration, the Levenberg-Marquardt gradient algorithm [14] was used to adapt weights and thresholds. The learning was repeated 20 times and only the configurations with the smallest NRMSE were used in further analysis.

5.1. Logistic Map

The logistic map is known as a simple model which yields chaos [17]. It is given by the recursive relation $x_t = 4\,x_{t-1}(1-x_{t-1})$. In the following experiments, 250 values were used, beginning with $x_1 = 0.01$.

Detailed comparison of the applied models, each having 4 inputs, 10 neurons in the hidden layer and 1 neuron in the output layer, is given in Table 1.

Table 1

The number of neurons in thresholding and reconstruction layers is established by the number of inputs N and the wavelet order S^M, in these models the numbers are 9 and 4, respectively. All models succeeded to determine the relationship between the consecutive values. Small values of thresholds, i.e., $\tau_1 = 7.4 \cdot 10^{-5} d_{1,\max}$ and $\tau_2 = 2.0 \cdot 10^{-4} d_{2,\max}$ of the DRTLP model, indicate that the models with denoising layers are capable of distinguishing between chaotic data and noise, which is not the case when denoising methods based on statistical criteria are applied [12]. Slightly higher errors observed at the models with the denoising layers originate in the mapping of the unconstrained thresholds τ_j^{∞} to the bounded thresholds τ_j, not allowing the latter to become zero in finite number of learning steps.

5.2. Hardness of Rubber Compound

The quality of rubber products, such as bicycle and motorcycle tubes, greatly depends on the quality of its components, particularly of a rubber compound. Thus, in the rubber industry the quality of a rubber compound after each mixing is permanently monitored. An important indicator for the quality of a rubber compound is its hardness.

Variation of hardness in 199 successive mixings together with some denoised data samples, obtained by the denoising layer of the DTLP and the DRTLP models, is shown in Figure 4. Quite substantial denoising performed by both, the DTLP model and the DRTLP model, reveals the noisy nature of the time series.

Figure 4

The prediction errors given in Table 2 show that the denoising layers improve prediction: the DTLP model has an advantage over the TLP model and the DRTLP model over the RTLP model. Within the testing set, the NRMSE measure is decreased by more than 6% and the MAPE measure by at least 3%. Larger prediction errors on the training set may be explained by higher variations of the training set data.

Table 2

The success of the DTLP and the DRTLP models can be attributed to the fact that the denoising layers have simplified the relations between input and output data samples by removing some noise from the input data samples. Mutual comparison of the models TLP and RTLP shows that recurrent connections do not necessarily improve prediction. This is most probably due to the learning algorithms which are far more complex for recurrent neural network models.

5.3. Sunspot number

The number of sunspots, dark areas of concentrated magnetic field on the Sun, is highly correlated with the Sun activity. The yearly average of the sunspot number, collected between the years 1700 and 2001, was used in the analysis [18].

Figure 5 shows the variations of the yearly average of the sunspot number and some denoised data samples obtained by the models DTLP and DRTLP.

Figure 5

Denoised data samples differ substantially. The DTLP model found almost no noise in the time series, i.e., the details were removed mainly on the first scale, $\tau_1 = 0.87\, d_{j,\max}$, while the details on the other scales were practically left intact. The DRTLP model, on the other hand, performed denoising on all three scales,

i.e., $\tau_1 = 0.11 d_{1,\max}$, $\tau_2 = 0.18 d_{2,\max}$ and $\tau_3 = 0.35 d_{3,\max}$, which reflects in reduction of amplitudes in denoised data samples.

Table 3

Detailed comparison of the models given in Table 3 shows that the prediction improvement of the DTLP model compared to the TLP model is similar to the prediction improvement of the DRTLP model over the RTLP model, regardless of the completely different concepts of denoising. In both cases, the NRMSE and MAPE performance measures are reduced from 5% and even up to 25% on training sets. Furthermore, the recurrent connections of the models RTLP and DRTLP had reduced the prediction errors on the training set for at least 7% in comparison to the models TLP and DTLP, respectively. Very large MAPE errors on the training set, compared to the MAPE errors on the test set, are due to the nature of the time series, having many small values in the training set. Namely, the MAPE performance measure extremely penalizes errors in prediction of small values.

6. Conclusion

The denoising based on wavelet multiresolution analysis can be formulated as a generalization of neural network layers. Opposed to standard neurons, in which free parameters determine the weighted sum, the free parameters of thresholding neurons determine the shape of thresholding functions.

A gradient based algorithm for setting the free parameters of thresholding neurons was developed. This algorithm enables for the same cost function to be used for setting both the weights of standard neurons and the thresholds of thresholding neurons. In this way, the denoising layers can be completely integrated into many types of layered neural network and denoising is not treated as a separate process. Although the learning algorithm is designed for gradient based methods, it can easily be generalized to various non-gradient algorithms, including evolutionary.

The presented examples of application on one-step-ahead prediction problems revealed that the denoising layers improve prediction on noisy data sets. The utilization of the cost function which minimizes prediction error ensures that denoising is performed in such a way that the optimal prediction is achieved. Unlike statistical methods, the neural networks with denoising layers can distinguish between noise and chaos and practically no denoising is performed in the latter case.

Acknowledgements

The project has been funded in part by the Slovenian Ministry of Education, Science and Sport under Grant No. Z2-3040.

References

1. Masters T (1995) Neural, Novel & Hybrid Algorithms for Time Series Prediction. John Wiley & Sons, Toronto.
2. Weigend A S, Gershenfeld N A (1994) Time Series Prediction: Forecasting the Future and Understanding the Past. Addison Wesley, Reading.
3. Forsee F D, Hagan M T (1997) Gauss-Newton Approximation to Bayesian Regularization. In Proceedings of the 1997 IEEE International Joint Conference on Neural Networks, Huston, pp 1930-1935.
4. Drucker H, Cun Y L (1992) Improving Generalization Performance Using Double Backpropagation. IEEE Trans. Neural Netw. 3 (6): 991-997.
5. Ster B (2003) Latched recurrent neural network. Electrotechnical Review 70 (1-2): 46-51, http://ev.fri.uni-lj.si/online.html.
6. Donoho D L (1995) De-Noising by Soft-Thresholding. IEEE Trans. Inf. Theory 41: 613-627.
7. Prochazka A, Mudrova M, Storek M. (1998) Wavelet Use for Noise Rejection and Signal Modelling. In Prochazka A et al. (eds). Signal Analysis and Prediction, Birkhauser, Boston, pp 215-226.
8. Zhang Q, Benveniste A (1992) Wavelet Networks. IEEE Trans. Neural Netw. 3: 889-898.
9. Quipeng L, Xiaoling Y, Quanke F (2003) Fault Diagnosis Using Wavelet Neural Networks, Neural Processing Letters 18: 115-123.
10. Daubechies I (1992) Ten Lectures on Wavelets, SIAM, Philadelphia.
11. Lotric U, Dobnikar A (2003) Matrix Formulation of Multilayered Perceptron with Denoising Unit. Electrotechnical Review 70, in press, http://ev.fri.uni-lj.si/online.html.
12. Lotric U (2000) Using Wavelet Analysis and Neural Networks for Time Series Prediction, Ph. D. Thesis. University of Ljubljana, Faculty of Computer and Information Science, Ljubljana.
13. Haykin S (1999) Neural Networks: A Comprehensive Foundation, 2nd edition. Prentice-Hall, New Jersey.
14. Press W H, Teukolsky S A, Vetterling W T, Flannery B P (1992) Numerical Recipes in C, 2nd edition, Cambridge University, Cambridge.

15. Trentin E (2001) Networks with Trainable Amplitude of Activation Functions. Neural Netw. 14: 471-493.
16. Williams R J, Zipser D (1989) A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. Neural Computation 1: 270-280.
17. Schuster H G (1984) Deterministic Chaos. An Introduction,. Physik, Weinheim.
18. Sunspot Index Data Center, Royal Observatory of Belgium (2001) Yearly Definitive Sunspot Number, http://sidc.oma.be.

Tables:

Table 1. Model comparison in prediction of the logistic map.

Model	Structure [Free parameters]		NRMSE Training	NRMSE Testing	MAPE Training	MAPE Testing
TLP	4-10-1	[61]	0.0065	0.0076	0.026	0.025
DTLP	4-(9-4)-10-1, S^2	[64]	0.0066	0.0078	0.049	0.059
RTLP	4-10-1	[71]	0.0032	0.0037	0.024	0.022
DRTLP	4-(9-4)-10-1, S^2	[74]	0.0033	0.0044	0.025	0.027

Table 2. Model comparison in prediction of rubber hardness.

Model	Structure [Free parameters]		NRMSE Training	NRMSE Testing	MAPE Training	MAPE Testing
TLP	18-1-1	[21]	0.75	0.67	0.0140	0.0121
DTLP	19-(53-19)-1-1, S^2	[27]	0.75	0.63	0.0140	0.0117
RTLP	4-2-1	[15]	0.82	0.69	0.0150	0.0128
DRTLP	18-(76-18)-1-4, S^8	[36]	0.74	0.62	0.0135	0.0119

Table 3. Model comparison in prediction of the yearly average sunspot number.

Model	Structure [Free parameters]		NRMSE Training	NRMSE Testing	MAPE Training	MAPE Testing
TLP	9-3-1	[34]	0.35	0.42	0.64	0.30
DTLP	9-(36-9)-8-1, S^5	[93]	0.23	0.40	0.35	0.23
RTLP	6-6-1	[55]	0.23	0.39	0.53	0.21
DRTLP	11-(37-11)-3-9, S^5	[103]	0.28	0.37	0.39	0.19

Figure legends:

Figure 1. Influence of the parameter s on the shape of the generalized soft thresholding function.

Figure 2. Wavelet based denoising layers.

Figure 3. a) The two-layered perceptron with global recurrent connections N - N_h - N_o and b) its enhancement with the wavelet based denoising layers N -(C - N)- N_h - N_o . When the recurrent connections (dashed) are omitted, the feedforward two-layered perceptron (a) and the two-layered perceptron with the denoising layers (b) are obtained.

Figure 4. Variation of a rubber hardness with examples of the denoised data samples $\tilde{\mathbf{p}}(q)$, obtained by the denoising layers of the models DTLP and DRTLP.

Figure 5. Variation of the yearly average sunspot number with examples of the denoised data samples $\tilde{\mathbf{p}}(q)$, obtained by the denoising layers of the models DTLP and DRTLP.

Figures

Figure 1

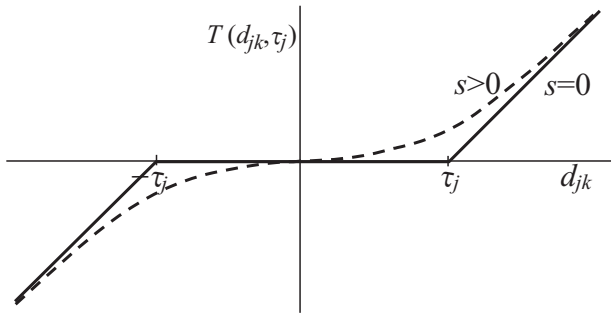

Figure 2

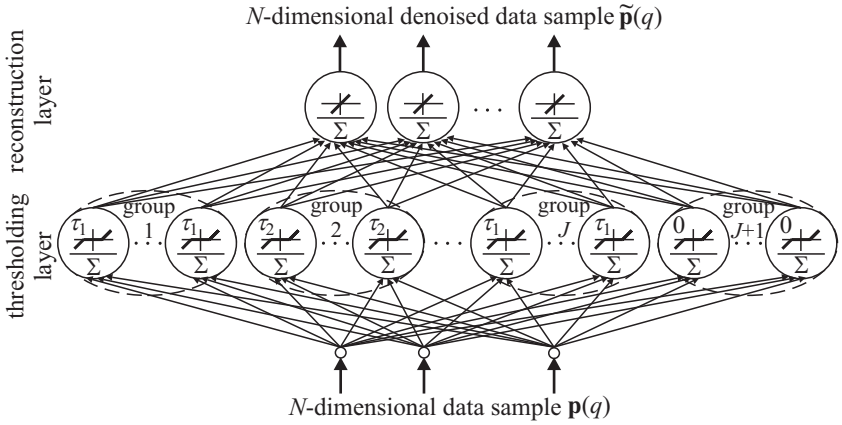

Figure 3

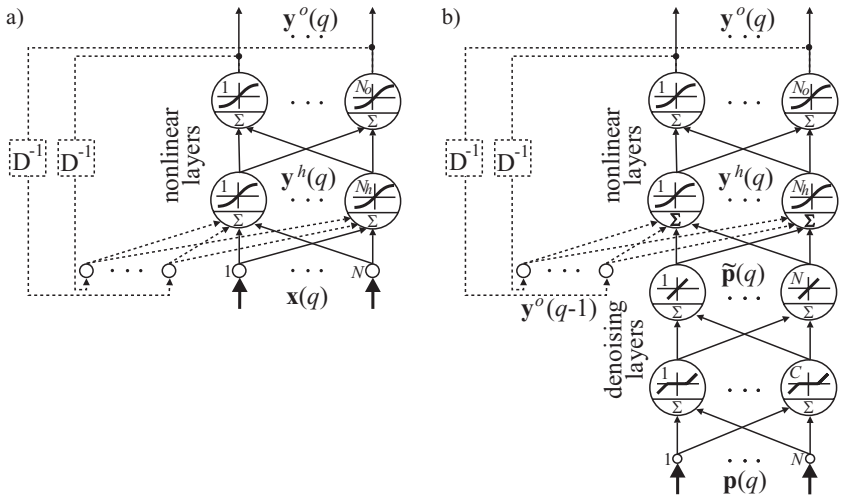

Figure 4

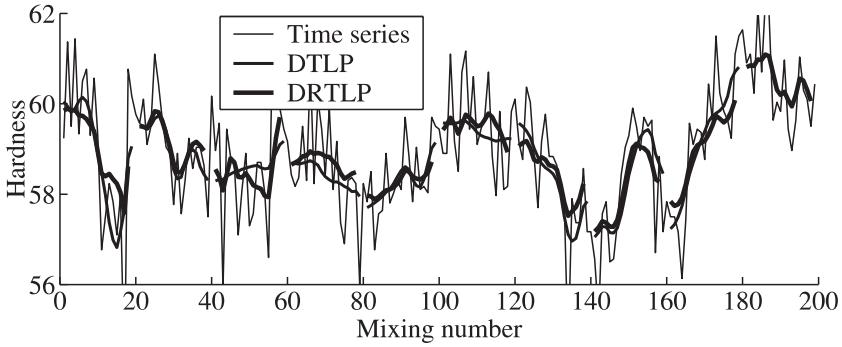

Figure 5