Wavelet Based Denoising Integrated into Multilayered Perceptron

Uroš Lotrič [1]

University of Ljubljana, Faculty of Computer and Information Science, Tržaška 25, 1000 Ljubljana, Slovenia

Abstract

A denoising unit based on wavelet multiresolution analysis is added ahead of the multilayered perceptron. The cost function used in neural network learning is also applied as the denoising criterion and hence denoising itself is treated as a part of the integrated model. By introducing continuously derivable generalized soft thresholding function and infinite thresholds, a gradient based learning algorithm for simultaneous setting of all free parameters of the model is derived. The proposed model outmatches the classical multilayered perceptron and the multilayered perceptron with statistical denoising in noisy time series prediction problems.

Key words: Multilayered perceptron, Wavelet multiresolution analysis, Denoising, Gradient descent threshold adaptation, Time series prediction

1 Introduction

Noise is inherently present in the majority of real world time series and in their analysis the question arises whether it should be removed or not. Due

[1] Corresponding author. Tel.: +386-1-4768874; fax: +386-1-4768369.
E-mail address: uros.lotric@fri.uni-lj.si.

to its generalization ability, a neural network removes noise from a time series to a certain extent. Moreover, injecting artificial noise can even improve its generalization performance [4]. However, denoising prior to the modelling can greatly improve its ability to capture valuable information. When noise can be identified, e.g. its equations are known, noise reduction can be based on this knowledge [20]. When the system equations are not known, only the methods estimating and removing noise on the basis of time series themselves can be used [8]. The basic problem faced in this case is to find the best denoising criterion for the given time series.

In this paper, a neural network model is proposed, which does not use denoising as data preprocessing, but, rather, includes it in the prediction model, thus using the same cost function as the criterion for prediction and noise level estimation. The dilemma of removing noise or not and to what extent is no longer a matter of data preprocessing, but is included into the modelling itself: the model removes noise adaptively so that prediction error is minimized. In the case of layered neural networks the gradient based learning algorithms are widely recognized as a powerful tool for setting the model free parameters. In order to integrate denoising into the layered neural network, the gradient based learning algorithms have to be expanded to the denoising. The denoising method based on the wavelet multiresolution analysis [7] was chosen, with its mathematical formulation being very similar to those of layered neural networks.

The pioneering work in wavelet based denoising was done by Donoho and Johnstone [9, 10]. Their work was founded on the basic idea of the wavelet multiresolution analysis: a time series is observed on different time scales and on each described by wavelet coefficients of its approximation and wavelet

coefficients of the remaining details. They transformed the wavelet coefficients of the details using a thresholding function and thus on each scale removed the wavelet coefficients of the details that mainly contributed to noise. To set the thresholds, determining the shape of the thresholding function, Donoho and Johnstone used the minmax principle in conjunction with Besov norm and signal continuity properties. Later on, several other denoising criteria were proposed. Nason used cross validation [15], Abramovich Bayesian hypothesis testing [1], and Cherkassky and Shao the Vapnik – Chervonenkis theory [6]. All these methods are based on statistical measures and were developed for pure denoising problems. They may be, however, also used as methods of time series preprocessing in prediction problems. Procházka, Mudrová and Štorek [16], for example, applied the method of Donoho and Johnstone in combination with a multilayered perceptron.

In time series prediction with neural networks, wavelets have also been used to decompose the original time series in order to reduce the complexity of the problem. One possibility is to generate time series on different scales, model each separately and then sum them up. Thomason [22] decomposed the time series into two time series: one was obtained by reconstruction from wavelet coefficients on smaller and the other on larger scales. Both time series were then modelled by a multilayered perceptron. On the other hand, the prediction can be performed also in the space of wavelet coefficients [3, 17, 24]. Aussem and Murtagh [3] proposed two different approaches to such prediction. In both cases the original time series was transformed by the à trous wavelet transform [19]. Then (i) the prediction of the wavelet coefficients was made separately for each scale and then summed up or (ii) the original time series was predicted from wavelet coefficients across all considered resolution scales simultaneously. Their methods yielded good results in prediction of both social

and natural phenomena [2, 3]. The second approach, which appears to be superior, was further developed by Renaud, Starck and Murtagh [17].

The approach presented in this study is similar to the second of the two approaches of Aussem and Murtagh and to the approach of Renaud, Starck and Murtagh [17]. However, in the space of wavelet coefficients wavelet based denoising is also applied, as proposed by Donoho and Johnstone [9, 10]. In contrast to Procházka, Mudrová and Štorek [16], denoising is not used to preprocess the entire time series, but is integrated into the neural network and works locally only on the time series window, currently presented to the model inputs. This permits dealing also with time series containing non-stationary noise. Thresholding is performed on the wavelet coefficients of the details on each scale selectively. And, most importantly, the cost function used in neural network learning is taken as the denoising criterion. The optimal thresholds are obtained by gradient based learning algorithm. This algorithm is similar to the one Chen and Chang [5] and Trentin [23] used to autotune the shape of activation functions in layered neural networks.

A brief background on wavelet based denoising is given in the next section. In the third section wavelet denoising is integrated into the multilayered perceptron and the gradient based learning algorithm for the thresholds is derived. The capability of the integrated model is evaluated in the fourth section by comparing its performance with the performance of the classical multilayered perceptron and the multilayered perceptron with the statistical denoising as preprocessing. The main conclusions are drawn in the last section.

2 Denoising with Wavelets

Classical time series denoising approaches rooted in Fourier analysis assume noise to be manifested mainly as high frequency oscillations. With this in mind, a time series is decomposed into sinusoidal waveforms of different frequencies and only low-frequency components are left in the denoised time series. Recent approaches make more flexible assumptions about the nature of a time series [6]. Namely, most relevant basis functions in a denoised time series are not specified a priori, but, rather, selected from a time series itself. The wavelet based denoising, for example, assumes that analyses of time series at different resolutions might improve the separation of the true underlying signal from noise.

2.1 Wavelet Multiresolution Analysis

The wavelet multiresolution analysis is based on the scaling function $\phi(t)$ and the corresponding mother wavelet $\psi(t)$, fulfilling specific technical conditions [7]. The scaling function and the mother wavelet are localized both in time and frequency domain (Fig. 1), which allows for explicit capture of the local dynamics within a time series. By dilation and translation of the scaling function and the mother wavelet the following functions are derived:

$$\phi_{j,k} = 2^{-j/2}\phi(2^{-j}t - k) \quad \text{and} \quad \psi_{j,k} = 2^{-j/2}\psi(2^{-j}t - k) \quad , \tag{1}$$

with the subscript j denoting the scale or corresponding resolution of the functions and the subscript k localizing the functions in time. On each scale j, the functions $\phi_{j,k}(t)$, $k \in \mathbb{Z}$, and $\psi_{j,k}(t)$, $k \in \mathbb{Z}$, form an orthonormal basis in the spaces of the square integrable functions $L^2(\mathbb{R})$ [7]. An arbitrary time

series $x(t) \in L^2(\mathbb{R})$ can be written as

$$x(t) = \sum_{k \in \mathbb{Z}} a_{J,k} \phi_{J,k}(t) + \sum_{j \leq J} \sum_{k \in \mathbb{Z}} d_{j,k} \psi_{j,k}(t) \quad , \tag{2}$$

where the first term represents the approximation on the scale J and the second term the details on the scale J and all finer scales. Together the wavelet coefficients $a_{J,k}$ of the approximation and the wavelet coefficients $d_{j,k}$ of the details form the discrete wavelet transform of the original time series $x(t)$.

In the wavelet multiresolution analysis the wavelet coefficients $a_{j,k}$ of the approximation and the wavelet coefficients $d_{j,k}$ of the details on adjacent scales are related by the decomposition

$$a_{j,k} = \sum_{n \in \mathbb{Z}} h^*_{n-2k} a_{j-1,n} \quad , \quad d_{j,k} = \sum_{n \in \mathbb{Z}} g^*_{n-2k} a_{j-1,n} \quad , \tag{3}$$

as well as by the reconstruction

$$a_{j-1,k} = \sum_{n \in \mathbb{Z}} (h_{k-2n} a_{j,n} + g_{k-2n} d_{j,n}) \quad , \tag{4}$$

where * denotes complex conjugates. Pyramidal schemes of decomposition and reconstruction are graphically presented in Fig. 2.

Coefficients h_n and g_n used in the decomposition and the reconstruction formulae are given as $h_n = \int_{-\infty}^{+\infty} \phi(t) \phi_{-1,n}(t) dt$ and $g_n = (-1)^n h^*_{1-n}$. The values for the most popular wavelet families can be found in literature [7]. The number C_j of the wavelet coefficients of the details on the scale j is given by the number of the wavelet coefficients of the details on the previous scale and by the number of nonzero coefficients g_n and h_n, determined by the chosen wavelet family and the wavelet order [12]. Throughout this work, the family of orthogonal least asymmetric wavelets or symlets [7] is used, because of its

near linear phase characteristics, favored in time series processing [14].

In further discussion, a discrete time series with N values x_k acquired at times $t_k = k\Delta t$, $k = 1, \ldots, N$, with the constant sampling time Δt is considered. To simplify the calculation on a discrete time series $a_{0,k} = x_k$ is set [7]. In this case, the number of the scales J is limited to the largest integer smaller than or equal to $\log_2 N$.

For the finite number of scales J, the discrete wavelet transform, which is a linear transformation, can be written as $\mathbf{c} = \mathbf{W}^{\mathrm{D}}\mathbf{x}$ and its inverse as $\mathbf{x} = \mathbf{W}^{\mathrm{R}}\mathbf{c}$. The vector $\mathbf{x}$ represents the discrete time series, $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$, while the vector $\mathbf{c} = [d_{1,1}, \ldots, d_{1,C_1}, d_{2,1}, \ldots, d_{J,C_J}, a_{J,1}, \ldots, a_{J,C_J}]^T$ combines the wavelet coefficients of the details on the scales $j = 1, \ldots, J$ and the wavelet coefficients of the approximation on the scale J. The elements of the decomposition matrix $\mathbf{W}^{\mathrm{D}} = \mathbf{W}^{\mathrm{D}}(N, J, h_n)$ and of the reconstruction matrix $\mathbf{W}^{\mathrm{R}} = \mathbf{W}^{\mathrm{R}}(N, J, h_n)$ can be derived from the pyramidal decomposition and reconstruction schemes given by Eqs. (3) and (4), respectively.

2.2 Wavelet Denoising

The discrete wavelet transform is linear and orthogonal, thus transforming white noise in the time space to white noise in the space of the wavelet coefficients [8]. It also enables compact coding, since the wavelet coefficients of the details possess high absolute values only in the intervals of rapid time series change. These properties led Donoho and Johnstone to propose denoising with thresholding [9], which consists of the following steps.

- A time series is transformed to the wavelet coefficients $a_{J,k}$ of the approximation and the wavelet coefficients $d_{j,k}$, $j = 1, \ldots, J$, of the details.

- The wavelet coefficients $d_{j,k}$ of the details on each scale, $j = 1, \ldots, J$, are separately thresholded as

$$\tilde{d}_{j,k} = T(d_{j,k}, \tau_j) \quad , \tag{5}$$

where the soft thresholding function [9]

$$T(d, \tau) = \begin{cases} 0, & |d| \leq \tau \\ d - \text{sign}(d)\tau, & \text{otherwise} \end{cases} \tag{6}$$

removes wavelet coefficients d of the details, that are absolutely smaller than the threshold τ, and reduces absolute values of those wavelet coefficients of the details which exceed the threshold.

- From the wavelet coefficients $a_{J,k}$ of the approximation and the modified wavelet coefficients $\tilde{d}_{j,k}$, $j = 1, \ldots, J$, of the details the denoised time series is reconstructed.

Setting the thresholds is the essential part of denoising. Assuming that most of the wavelet coefficients of the details contribute to the Gaussian white noise, Donoho and Johnstone set the thresholds to $\tau_j = \hat{\sigma}_j \sqrt{\ln N_{\text{C}}}$, where $\hat{\sigma}_j$ is the estimated standard deviation of the wavelet coefficients of the details on the scale j and N_{C} is the number of all wavelet coefficients. This method, however, tends to underfit the data [10] and was therefore further enhanced by a criterion based on the Stein's unbiased risk estimate [21]. The so obtained method, called the SureShrink, is most widely used in denoising with wavelets. Most methods proposed by other authors [1, 15] differ from the foregoing in the way thresholds are estimated.

3 Multilayered Perceptron with Denoising Unit

A new approach towards threshold estimation which integrates both preprocessing and modelling with a standard neural network into an integrated model is proposed. It can be shown that the formulation of wavelet based denoising is in fact an analog of the formulation of the layers of the multilayered perceptron. The denoising layers are topologically set in front of the first hidden layer of the classical multilayered perceptron and a learning algorithm for threshold setting is derived.

3.1 Model

The multilayered perceptron [11] as a general function estimator can be used to find associations between the specified input–output pairs $\{(\mathbf{p}(1), \mathbf{r}(1)), \ldots, (\mathbf{p}(Q), \mathbf{r}(Q))\}$ by minimizing the specified cost function. Neurons in the multilayered perceptron, shown in Fig. 3, are arranged in L layers and each of the N_l neurons in the l-th layer is connected to all neurons in adjacent layers. When the q-th input–output pair is presented to the multilayered perceptron, the output $y_n^l(q)$ of the n-th neuron in the l-th layer is given by

$$y_n^l(q) = \varphi_n^l(v_n^l(q)) \quad , \quad n = 1, \ldots, N_l \quad , \tag{7}$$

where φ_n^l is the activation function and the internal activity level $v_n^l(q)$ denotes the sum of weighted outputs from the previous layer

$$v_n^l(q) = \sum_{i=0}^{N_{l-1}} \omega_{n,i}^l y_i^{l-1}(q) \quad . \tag{8}$$

For $i > 0$, $y_i^{l-1}(q)$ are the outputs from the previous layer, whereas the constant term $y_0^{l-1}(q) = -1$ is used to introduce a bias into the internal activity level.

The weights $\omega_{n,i}^l$ of neural network connections are free parameters of the model. The same activation function is commonly used for all neurons, in this case the hyperbolic tangent, $\varphi_n^l(v) = \tanh(v)$.

In the multilayered perceptron with the denoising unit the denoising procedure described in section 2.2 is applied to each input vector separately. Using the matrix notation of the wavelet transform, the input vector $\mathbf{p}(q)$ of the q-th input–output pair with the elements $p_i(q)$, $i = 1, \ldots, N_0$, is first transformed into the wavelet coefficients,

$$c_n(q) = \sum_{i=1}^{N_0} W_{n,i}^{\mathrm{D}} p_i(q) \quad , \quad n = 1, \ldots, N_{\mathrm{D}} \quad , \tag{9}$$

where $W_{n,i}^{\mathrm{D}}$ are the elements of the decomposition matrix $\mathbf{W}^{\mathrm{D}} = \mathbf{W}^{\mathrm{D}}(N_0, J, h_n)$ and N_{D} the number of rows in the decomposition matrix. The number of scales J is the largest integer, smaller than or equal to $\log_2 N_0$.

Next, the modified wavelet coefficients $\tilde{c}_n(q)$ are obtained by separately thresholding the wavelet coefficients of the details on each scale $j = 1, \ldots, J$, with the thresholding function given by Eq. (6),

$$\tilde{c}_n(q) = T(c_n(q), \tau_j) \quad . \tag{10}$$

The index n runs over the wavelet coefficients of the details on the scale j, $n = S(j-1)+1, \ldots, S(j)$. The function $S(j) = \sum_{k=1}^{j} C_k$ gives the number of the wavelet coefficients of the details on all scales up to the scale j. The wavelet coefficients of the approximation on the largest scale J remain unchanged, $\tilde{c}_n(q) = c_n(q)$, $n = S(J)+1, \ldots, N_{\mathrm{D}}$.

Finally, the denoised vector $\mathbf{y}^0(q)$ is obtained from the modified wavelet coefficients by the inverse discrete wavelet transform, given by the reconstruction

matrix $\mathbf{W}^{\mathrm{R}} = \mathbf{W}^{\mathrm{R}}(N_0, J, h_n)$ with the elements $W^{\mathrm{R}}_{i,n}$,

$$y^0_i(q) = \sum_{n=1}^{N_{\mathrm{D}}} W^R_{i,n} \tilde{c}_n(q) \quad , \quad i = 1, \ldots, N_0 \quad , \tag{11}$$

and fed forward to the first hidden layer of the multilayered perceptron.

Calculation of the denoised wavelet coefficients by Eq. (9) and Eq. (10) is similar to the calculation of neuron outputs by Eq. (8) and Eq. (7). The smooth thresholding described by Eq. (10) is just a special form of the activation function given by Eq. (7). Therefore, the discrete wavelet transform together with the thresholding can be thought of as a layer of N_{D} special neurons with denoising task, called the decomposition and thresholding layer (Fig. 3). The reconstruction of the denoised vector described by Eq. (11) is again similar to Eq. (8). Consequently, it can be represented as an additional layer of N_0 reconstruction neurons with linear activation functions, $I(v) = v$, termed reconstruction layer (Fig. 3).

Although the equations are very similar, the denoising unit differs from the multilayered perceptron in positions of free parameters. While the free parameters of the multilayered perceptron are the weights on connections between the neurons, the free parameters of the denoising unit are hidden within the thresholding functions.

3.2 Backpropagation Learning

The backpropagation learning is the most popular algorithm for adapting free parameters of the multilayered perceptron. The goal of learning is to find a set of the model free parameters which minimize the cost function. Usually, the cost function is given by the mean squared error (MSE) between target and

calculated outputs on all Q input–output pairs $E = Q^{-1} \sum_{q=1}^{Q} E(q)$, where $E(q)$ is the squared error of the q-th input–output pair $E(q) = \frac{1}{2} \sum_{n=1}^{N_L} e_n^2(q)$ and $e_n(q) = r_n(q) - y_n^L(q)$ the difference between the n-th element of the output vector $\mathbf{r}(q)$ and the value, calculated on the n-th output neuron.

In the backpropagation learning the weights $\omega_{n,i}^l$ of the multilayered perceptron are updated following the delta rule

$$\Delta\omega_{n,i}^l(q) = -\eta \partial E(q)/\partial\omega_{n,i}^l \tag{12}$$

when the q-th input–output pair is presented. The constant η determines the rate of learning. This equation can be rewritten as $\Delta\omega_{n,i}^l(q) = \eta\delta_n^l(q)y_i^{l-1}(q)$ where the local gradient $\delta_n^l(q) = -\partial E(q)/\partial v_n^l(q)$ of each neuron can be analytically calculated [11].

The backpropagation learning is designed for optimization of free parameters without constraints. For the thresholds, however, only the nonnegative values smaller than absolutely the largest wavelet coefficient of the details on each scale $c_{j,\max}$ are reasonable [12, 13]. To extend the backpropagation learning to the denoising unit the unlimited thresholds τ_j^∞ are introduced as

$$\tau_j = c_{j,\max}(1 + \mathrm{e}^{-\tau_j^\infty})^{-1} \quad , \quad j = 1, \ldots, J \quad . \tag{13}$$

The delta rule (Eq. (12)) can be applied for updating the unlimited thresholds in the following way:

$$\Delta\tau_j^\infty(q) = -\eta\frac{\partial E(q)}{\partial\tau_j^\infty} = -\eta\frac{\partial E(q)}{\partial\tau_j}\frac{\partial\tau_j}{\partial\tau_j^\infty} = \eta\frac{\partial E(q)}{\partial\tau_j}\tau_j\left(\frac{\tau_j}{c_{j,\max}} - 1\right) \quad . \tag{14}$$

The thresholds τ_j of the denoising unit affect the elements of the denoised vector and so indirectly the outputs of the multilayered perceptron and the

cost function. Thus, the chain rule can be applied,

$$\frac{\partial E(q)}{\partial \tau_j} = \sum_{i=1}^{N_0} \frac{\partial E(q)}{\partial y_i^0(q)} \frac{\partial y_i^0(q)}{\partial \tau_j} \quad . \tag{15}$$

The first factor in the sum can be written as

$$\frac{\partial E(q)}{\partial y_i^0(q)} = \sum_{n=1}^{N_1} \frac{\partial E(q)}{\partial v_n^1(q)} \frac{\partial v_n^1(q)}{\partial y_i^0(q)} = -\sum_{n=1}^{N_1} \delta_n^1(q) \omega_{n,i}^1 \quad . \tag{16}$$

The second factor in Eq. (15) can be derived from Eqs. (10) and (11), taking into account that the threshold τ_j affects only the wavelet coefficients of the details on the scale j,

$$\frac{\partial y_i^0(q)}{\partial \tau_j} = \sum_{n=S(j-1)+1}^{S(j)} W_{i,n}^R \frac{\partial \tilde{c}_n(q)}{\partial \tau_j} = \sum_{n=S(j-1)+1}^{S(j)} W_{i,n}^R \frac{\partial T(c_n(q), \tau_j)}{\partial \tau_j} \quad . \tag{17}$$

In denoising applications the soft thresholding function, given by Eq. (5) and shown in Fig. 4 (thin line), is commonly used. However, it is not suitable for the backpropagation learning. As illustrated in Fig. 4b, its partial derivatives $\partial T(c_n(q), \tau_j)/\partial \tau_j$ equal zero for the wavelet coefficients of the details absolutely smaller than the threshold, $|c_n(q)| \leq \tau_j$. If the threshold τ_j is sufficiently high, a special situation may occur, where all partial derivatives $\partial \tilde{c}_n(q)/\partial \tau_j$, $q = 1, \ldots, Q$, become zero. This results in $\Delta \tau_j(q) = 0$, $q = 1, \ldots, Q$, which prevents the backpropagation learning from changing the threshold τ_j any further. For this reason, the following generalized soft thresholding function shown in Fig. 4 (thick line) is introduced

$$T^{\mathrm{G}}(c, \tau) = c + \frac{1}{2}\left(\sqrt{(c-\tau)^2 + s} - \sqrt{(c+\tau)^2 + s}\right) \quad , \tag{18}$$

with c being the wavelet coefficient and τ denoting the threshold. The constant s determines the deviation from the soft thresholding function given by Eq. (6).

For $s = 0$ both thresholding functions are identical. The larger the constant s, the greater the interval, in which the partial derivative of the generalized soft thresholding function with respect to the threshold

$$\frac{\partial T^{\mathrm{G}}(c,\tau)}{\partial \tau} = -\frac{1}{2}\left(\frac{c-\tau}{\sqrt{(c-\tau)^2+s}} + \frac{c+\tau}{\sqrt{(c+\tau)^2+s}}\right) \tag{19}$$

considerably differs from zero, augmenting the rate of convergence of the back-propagation learning. The value of $s = 0.01$ was used.

The foregoing description of learning of the multilayered perceptron with the denoising unit can be easily expanded to other quasi Newtonian minimization methods, such as conjugate gradient or Levenberg-Marquardt schemes [12]. The approach used for setting the thresholds τ_j is similar to the approach Trentin [23] used for setting the amplitudes of activation functions. However, Trentin used the same parameter for all neurons in a layer, while there are J groups of neurons with the same threshold in the decomposition and thresholding layer.

3.3 Computational requirements

Integration of the denoising and thresholding layer and the reconstruction layer into the classical multilayered perceptron unavoidably increases the computational complexity of the model.

With addition of the two layers, the architecture of the multilayered perceptron having N_0 inputs and N_l, $l = 1, \ldots, L$, neurons in each nonlinear layer, i.e., $N_0 - N_1 - \ldots - N_L$, changes to $N_0 - N_D - N_0 - N_1 - \ldots - N_L$. If the decomposition and reconstruction matrix are pre-computed, each of the added layers leads to additional $N_0 N_D$ multiplications and N_D generalized

soft thresholding function calculations. As the number of thresholding neurons $N_D = S(J)$ increases approximately linearly with the number of model inputs N_0, the number of additional multiplications required by the first two layers is approximately proportional to N_0^2.

Similarly, the number of multiplications needed to update each threshold τ_j, $j = 1, \ldots, J$, during the back propagation learning is proportional to N_0^2. Thus, the learning process requires additional multiplications proportional to $N_0^2 \log_2 N_0$ and N_D additional calculations of generalized soft thresholding function derivatives.

4 Results

The proposed multilayered perceptron with the denoising unit (dMLP) was applied to one step ahead prediction problems, where the model relates a certain value of a time series with its previous values. Its performance was tested on several time series: the second order process, the Feigenbaum sequence and the time series obtained from the quality control of rubber compounds from a local producer. To provide a reference for the comparison, the prediction by the multilayered perceptron was applied to the same problems in two ways: without preprocessing (MLP) and with the wavelet denoising based on a statistical criterion (p+MLP).

4.1 Experimental Setup

The input–output pairs were generated from the time series with elements x_k, $k = 1, \ldots, N$, normalized to the zero mean and the unity variance. In the case of the dMLP and the MLP models, the elements of the q-th input–output

pair $(\mathbf{p}(q), \mathbf{r}(q))$ were prepared directly from the time series, $p_i(q) = x_{q+i-1}$, $i = 1, \ldots, N_0$, and $r_1(q) = x_{q+N_0}$, N_0 being the number of inputs. In the case of the p+MLP model, the time series values $x_1, \ldots, x_{q+N_0-1}$ were used for the wavelet denoising, while only the last N_0 denoised values formed $\mathbf{p}(q)$. The SureShrink denoising method with the symlet wavelets S^8 on eight scales was applied for denoising [12]. The first 85% of the input–output pairs formed the training set for setting free parameters of the models, while the remaining 15% of the input–output pairs formed the test set used for comparison of model performace. The first 85% of samples in training set were used for updating model free parameters and the last 15% of samples in training set were used for learning algorithm early stopping [11].

The structural parameters or architecture design of the models cannot be included in the gradient based learning algorithms. Their optimal values can be found by inspecting the parameter space. Three layers with adaptable parameters were allowed in each model: the reconstruction and the thresholding layer and two layers of neurons in the dMLP model and three layers of neurons in the MLP and the p+MLP models. The number of model inputs N_0 was limited to 20. The number of the neurons in the hidden layers was tuned in such a way that the total number of free parameters, including weights and thresholds, did not exceed 30% of the number of the input–output pairs included in the training set. For short input vectors the wavelet order has a strong impact on the denoising and was therefore included in the dMLP model as a structural parameter. Wavelets $S^1, \ldots, S^M$ were tried in each configuration, with the wavelet order M being the largest integer smaller than or equal to $N_0/2$.

For each configuration of the structural parameters the learning was repeated

20 times, each time randomly initializing model free parameters. While the weights of neural network connections were initialized on the interval $[-1, +1]$, the thresholds τ_j, $j = 1, \ldots, J$, were initialized on the intervals $[0.05\, c_{j,\max}$, $0.15\, c_{j,\max}]$. With such initialization only slight denoising of input samples was favored at the beginning of the learning process.

The configurations with the smallest performance measure on the test set were used further in the model comparison. As a performance measure the normalized root mean squared error

$$\mathrm{NRMSE} = \sqrt{\frac{1}{\sigma^2 Q_\mathrm{S}} \sum_{q=1}^{Q_\mathrm{S}} (r(q) - y_1^L(q))^2} \quad , \tag{20}$$

was considered, with σ denoting the standard deviation of a time series and Q_S denoting the number of input–output pairs in the observed set.

4.2 Second Order Process

The second order process given by the equation $\mathrm{d}^2x(t)/\,\mathrm{d}t^2 + \omega^2 x(t) = u(t)$ with $\omega^2 = 5\,\mathrm{s}^{-2}$ and the Gaussian white noise $u(t)$ with the standard deviation $\sigma_p = 0.1\,\mathrm{s}^{-2}$ was considered. To simulate measurements, Gaussian distributed error with standard deviation $\sigma_m = 0.3$ was added to the samples, taken 250 times, once every 0.2 s.

The second order process, simulated measurements and one step ahead predictions with all three models are shown in Fig. 5. Predictions with the dMLP model are closest to the underlying process, which also leads to the smallest errors with respect to the simulated measurements, as shown in Table 1. The performance of the dMLP model is superior to the MLP and the p+MLP models on the test set, while differences are less pronounced on the training

set. The p+MLP model yields even smaller error on the training set. This is due to the overfitting of the p+MLP model, which, on the other hand, reduces its ability of generalization and therefore leads to poorer performance on the test set. Moreover, the number of free parameters in the dMLP model is considerably smaller than the number of free parameters in the MLP and the p+MLP models. The smallest errors with respect to the simulated measurements were obtained when the dMLP model with the symlet wavelet S^5, 14 inputs, 39 neurons in the decomposition and thresholding layer, 14 neurons in reconstruction layer, 1 neuron in the hidden layer and 1 neuron in the output layer was used.

The influence of the denoising unit on an input vector is presented in Fig. 6. The central elements of the denoised vector lie close to the underlying process. Larger deviations at both edges are caused by the end effect, which was slightly reduced by using symmetric padding [12]. The denoising with the dMLP model was performed on 3 scales. At the end of the learning process the thresholds $0.84\,c_{1,\max}$, $0.44\,c_{2,\max}$ and $0.13\,c_{3,\max}$ were obtained, thus removing almost all wavelet coefficients of the details on the first scale and slightly modifying the majority of the wavelet coefficients of the details on the third scale.

4.3 Feigenbaum Sequence

The Feigenbaum sequence or the logistic map is given by the recursive relation $x_k = r\,x_{k-1}(1 - x_{k-1})$. The relation with $r = 4$ was considered, where the sequence becomes chaotic [18]. From the initial value $x_1 = 0.01$, 250 values were calculated with 15-digit precision (Fig. 7).

Table 2 gives the comparison of the three models on the Feigenbaum sequence.

It is obvious from the large prediction errors of the p+MLP model that statistical denoising is not appropriate when chaotic time series are considered, since it does not distinguish between chaotic time series and pure noise. Both the dMLP model and the MLP model, managed to learn the relationship between the consecutive values. At the end of the learning process, the thresholds of the dMLP model on the two scales reached $1.6 \cdot 10^{-3} c_{1,\max}$ and $4.9 \cdot 10^{-4} c_{2,\max}$. Obviously, the dMLP model recognized that denoising is not necessary. However, the mapping of the infinite thresholds τ_j^{∞} to the thresholds τ_j, given by Eq. (13), does not allow the thresholds τ_j to become zero and thus leave the input vector completely intact. Therefore, the errors of the dMLP model are slightly higher than the errors of the MLP model.

To illustrate the robustness and stability of the algorithm, the distribution of performance measure on testing set for 1000 runs, using the architectural design given in Table 2, is presented in Figure 8a for the MLP model and the dMLP model. The distribution of the dMLP model is shifted to the right compared to the MLP model, indicating poorer convergence of the dMLP model. To further reveal the source of the dMLP model convergence problems, the distribution of thresholds τ_1 and τ_2 end values for 1000 runs are presented in Figure 8b. These values should be close to zero, but in approximately one fourth of the runs their values are too large to yield good results. Apparently the whole model was caught in a local minimum for these cases and based on the similarity of the MLP model and the dMLP model distributions it may be hard to ascribe these convergence problems solely to thresholds.

4.4 Quality Control Time Series

An important characteristic determining quality of a rubber compound is its hardness. It is measured in Shore units on the scale reading from 0 to 100. Variation of hardness of the rubber compound used for bicycle and motorcycle tubes in 199 successive mixings is shown in Fig. 9a. A detail, together with predictions obtained with all three models, is given in Fig. 9b.

Detailed comparison of the models is given in Table 3. The uncommonly high errors on the training set compared to the test set are rooted in the nature of time series. Namely, the spikes in the time series, particularly those at mixing 17 and around mixings 50 and 140 greatly contribute to prediction error on training set. The proposed dMLP model improves the prediction of the MLP model, while the statistical denoising in the p+MLP model does not lead to a better prediction. As can be seen in Fig. 10, the key to success of the dMLP model was in the denoising which was less severe than in the case of the p+MLP model. At the end of the learning process, four thresholds of the dMLP model were set to $0.87\,c_{1,\max}$, $0.67\,c_{2,\max}$, $0.13\,c_{3,\max}$ and $0.54\,c_{4,\max}$, from which it may be concluded that the most informative details were those on the third scale.

5 Conclusion

A model integrating the wavelet based denoising technique into the multilayered perceptron is proposed. The denoising unit not only becomes a part of the multilayered perceptron in the topological sense, but is simultaneously included in the learning algorithm. In this way, the same cost function is used

for setting free parameters of both the denoising unit and the multilayered perceptron, and it can be termed an integrated model. To extend the gradient based learning algorithm, commonly used in multilayered perceptron, to the denoising unit an enhancement of the soft thresholding function to continuously derivable generalized soft thresholding function and the mapping of thresholds to their unlimited counterparts were introduced.

The model was tested on one step ahead prediction problems, where the prediction error is used as the cost function. It is shown that the prediction of noisy time series is more successful with the integrated model which uses prediction error as the denoising criterion than with the classical multilayered perceptron and the multilayered perceptron with statistical denoising as preprocessing. Moreover, the integrated model was able to detect the case of chaotic time series where, in contrast to statistical denoising methods, practically no denoising was applied, thus enabling the model to perform equally well as the classical multilayered perceptron. Therefore, the multilayered perceptron with the denoising unit can be considered as a generalization of the classical multilayered perceptron. Its ability to handle noise is however limited by the minimization of prediction error, i.e., there is no guarantee that the global minimum will be reached and noise optimally reduced.

There are numerous possibilities for future improvements and applications. The performance and convergence of the integrated model can, for example, be improved by a heuristic initialization of free parameters and enhancements of the learning algorithm. For example, adding a regularization term to the cost function, may improve generalization. The batch learning was used throughout this paper, however, the approach is also suitable for on-line learning and in this case also non-stationary noise can be handled. Apart from the multilayered

perceptron, other types of neural networks could be considered. Furthermore, the same approach can be expanded to two or more dimensional problems, such as pattern recognition.

Acknowledgments

This work is supported by the Slovenian Ministry of Education, Science and Sport under the grant Z2-3040. The author is grateful to referees for their most constructive comments which help to improve the manuscript.

References

[1] F. Abramovich, T. Sapatinas, B. W. Silverman, Wavelet thresholding via Bayesian approach, J. R. Stat. Soc., B 60 (1998) 723–749.

[2] A. Aussem, J. Campbell, F. Murtagh, Wavelet-based feature extraction and decomposition strategies for financial forecasting, J. of Computational Intelligence in Finance 4 (2) (1998) 5–12.

[3] A. Aussem, F. Murtagh, Combining neural network forecasts on wavelet-transformed time series, Connect. Sci. 9 (1997) 113–121.

[4] C. M. Bishop, Training with noise is equivalent to Tikhonov regularization, Neural Comput. 7(1) (1995) 108–116.

[5] C.-T. Chen, W.-D. Chang, A feedforward neural network with function shape autotuning, Neural Netw. 9 (1996) 627–641.

[6] V. Cherkassky, X. Shao, Signal estimation and denoising using VC-theory, Neural Netw. 14 (2001) 37–52.

[7] I. Daubechies, Ten Lectures on Wavelets, SIAM, Philadelphia, 1992.

[8] D. L. Donoho, De-noising by soft-thresholding, IEEE Trans. Inf. Theory 41 (3) (1995) 613–627.

[9] D. L. Donoho, I. M. Johnstone, Ideal spatial adaptation by wavelet shrinkage, Biometrika 81 (1994) 425–455.

[10] D. L. Donoho, I. M. Johnstone, Adapting to unknown smoothness via wavelet shrinkage, J. Am. Stat. Assoc. 90 (432) (1995) 1200–1224.

[11] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, Prentice-Hall, New Jersey, 1999.

[12] U. Lotrič, Using wavelet analysis and neural networks for time series prediction, Ph.D. thesis, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana (2000).

[13] U. Lotrič, A. Dobnikar, Wavelet based smoothing in time series prediction with neural networks, in: V. Kůrková, N. Steele, R. Nerudá, M. Kárný (Eds.), Artificial Neural Nets and Genetic Algorithms: Proceedings of the International Conference in Prague, Springer, Wien, 2001, pp. 43–46.

[14] T. Masters, Neural, Novel & Hybrid Algorithms for Time Series Prediction, John Wiley & Sons, Toronto, 1995.

[15] G. P. Nason, Wavelet shrinkage using cross-validation, J. R. Stat. Soc., B 58 (1996) 463–479.

[16] A. Procházka, M. Mudrová, M. Štorek, Wavelet use for noise rejection and signal modelling, in: A. Procházka, J. Uhlíř, P. J. W. Rayner, N. G. Kingsbury (Eds.), Signal Analysis and Prediction, Birkhäuser, Boston, 1998, pp. 215–226.

[17] O. Renaud, J.-L. Starck, F. Murtagh, Prediction based on a multiscale decomposition, International Journal of Wavelets, Multiresolution and Information Processing 1(2) (2003), 217–232.

[18] H. G. Schuster, Deterministic Chaos. An Introduction, Physik, Weinheim,

1984.

[19] M. J. Shensa, The discrete wavelet transform: Wedding the à trous and Mallat algorithms, IEEE Trans. Signal Process. 40 (10) (1992) 2464–2482.

[20] J.-L. Starck, F. Murtagh, A. Bijaoui, Image and Data Analysis: The Multiscale Approach, Cambridge University Press, Cambridge, 1998.

[21] C. M. Stein, Estimation of the mean of a multivariate normal distribution, Ann. Stat. 9 (6) (1981) 1135–1151.

[22] M. R. Thomason, Financial forecasting with wavelet filters and neural networks, J. of Computational Intelligence in Finance 5 (2) (1997) 27–32.

[23] E. Trentin, Networks with trainable amplitude of activation functions, Neural Netw. 14 (2001) 471–493.

[24] F.-C. Tsui, S. Mingui, C.-C. Li, R. J. Sclabassi, Recurrent neural networks and discrete wavelet transform for time series modeling and prediction, in: Proceedings ICASSP-95, Detroit, 1995, pp. 3359–3362.

List of Tables

List of Figures

Table 1

Model	Structure	Number of free parameters	NRMSE	
			Training Set	Test Set
MLP	12-2-2-1	35	0.473	0.540
p+MLP	10-2-4-1	39	0.459	0.537
dMLP	14-(39-14)-1-1, S^5	20	0.465	0.508

Table 2

Model	Structure	Number of free parameters	NRMSE	
			Training Set	Test Set
MLP	4-10-1	61	$6.51 \cdot 10^{-3}$	$7.56 \cdot 10^{-3}$
p+MLP	11-2-3-1	37	0.995	0.974
dMLP	S^2, 4-(9-4)-10-1	63	$6.62 \cdot 10^{-3}$	$7.84 \cdot 10^{-3}$

Table 3

Model	Structure	Number of free parameters	NRMSE	
			Training Set	Test Set
MLP	18-1-1	21	0.78	0.67
p+MLP	7-3-1	28	0.97	0.76
dMLP	S^5, 19-(53-19)-1-1	26	0.87	0.63

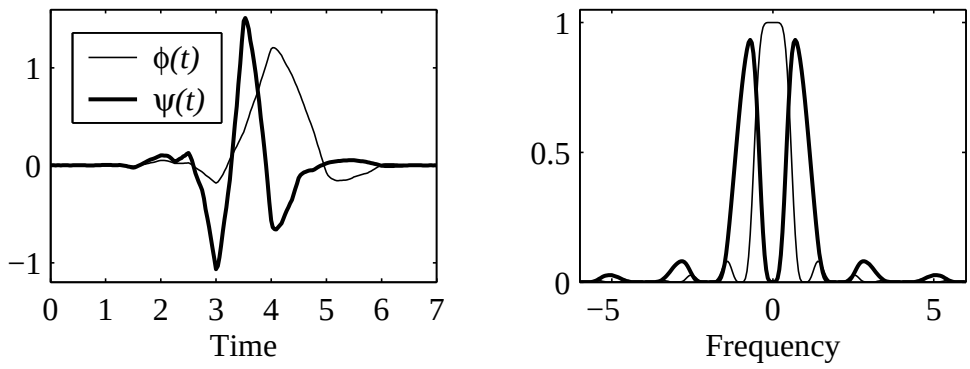

Fig. 1.

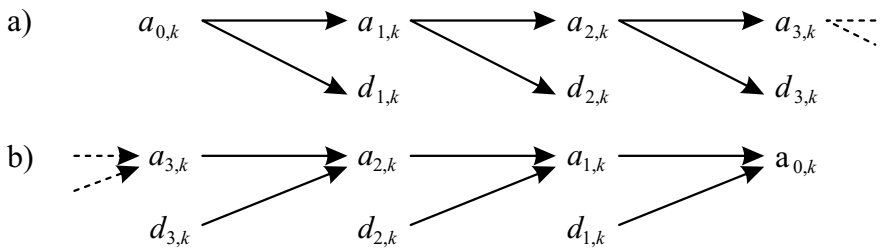

Fig. 2.

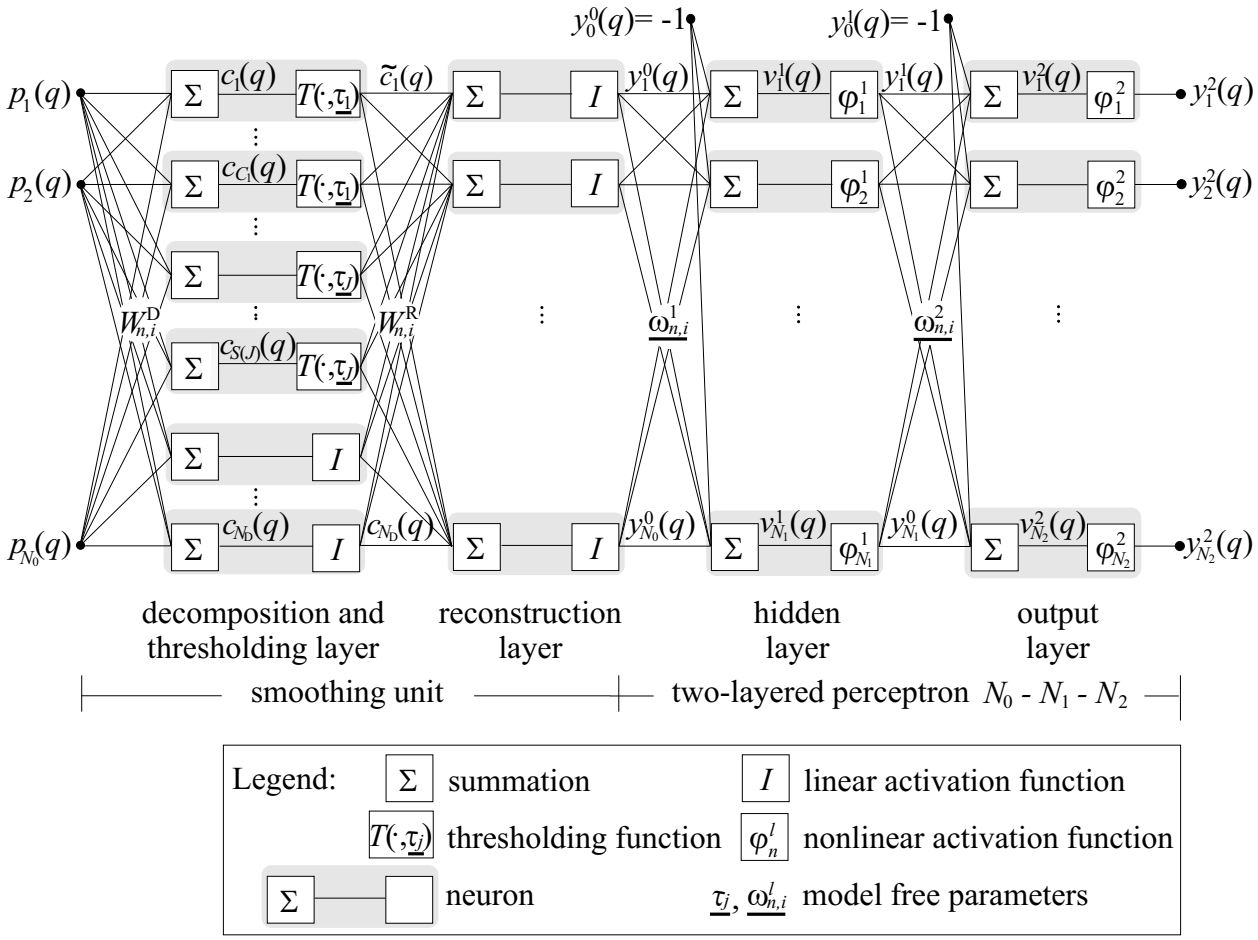
$y_0^0(q)$= -1
$y_0^1(q)$= -1
$p_1(q)$
$p_2(q)$
$p_{N_0}(q)$
$c_1(q)$
$c_{C_1}(q)$
$c_{S(J)}(q)$
$c_{N_D}(q)$
$T(\cdot,\underline{\tau}_1)$
$T(\cdot,\underline{\tau}_J)$
$\tilde{c}_1(q)$
$\tilde{c}_{N_D}(q)$
$W_{n,i}^D$
$W_{n,i}^R$
$y_1^0(q)$
$y_{N_0}^0(q)$
$v_1^1(q)$
$v_{N_1}^1(q)$
φ_1^1
$\varphi_{N_1}^1$
$y_1^1(q)$
$y_{N_1}^0(q)$
$\underline{\omega}_{n,i}^1$
$\underline{\omega}_{n,i}^2$
$v_1^2(q)$
$v_{N_2}^2(q)$
φ_1^2
φ_2^2
$\varphi_{N_2}^2$
$y_1^2(q)$
$y_2^2(q)$
$y_{N_2}^2(q)$
decomposition and thresholding layer
reconstruction layer
hidden layer
output layer
smoothing unit
two-layered perceptron N_0 - N_1 - N_2
Legend:
summation
linear activation function
thresholding function
nonlinear activation function
neuron
model free parameters

Fig. 3.

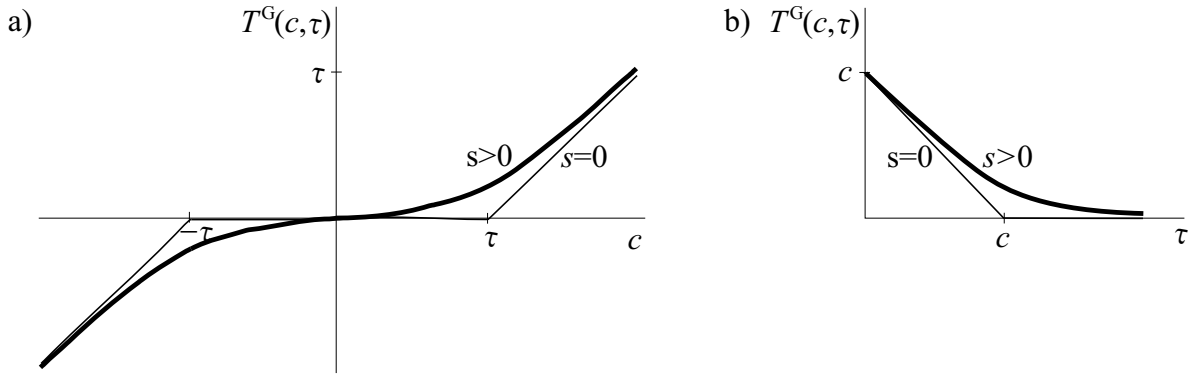

Fig. 4.

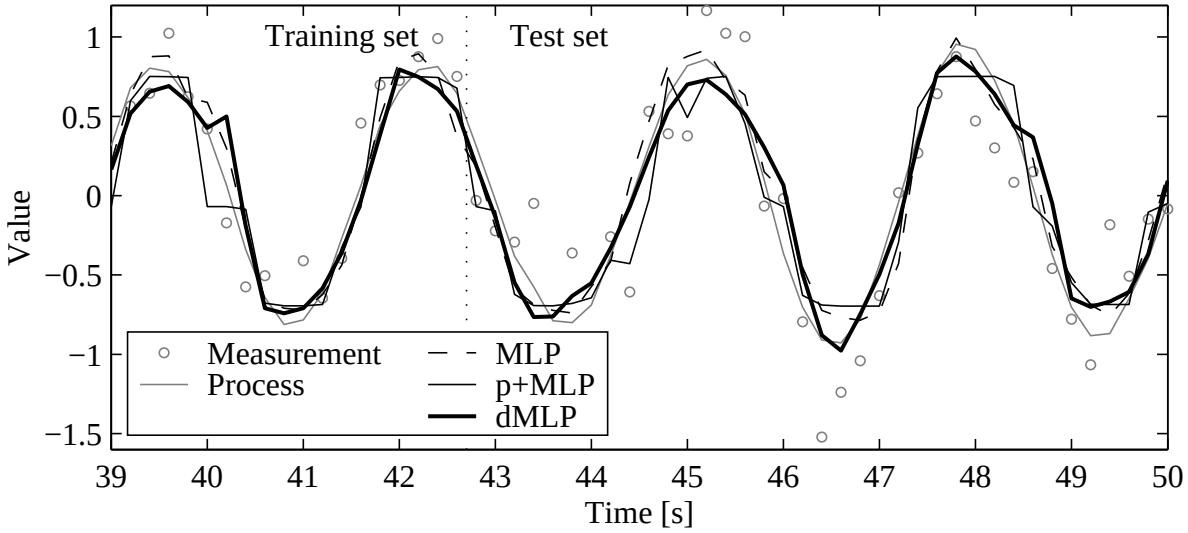
Training set
Test set
Measurement
Process
MLP
p+MLP
dMLP
Value
Time [s]
1
0.5
0
−0.5
−1
−1.5
39
40
41
42
43
44
45
46
47
48
49
50

Fig. 5.

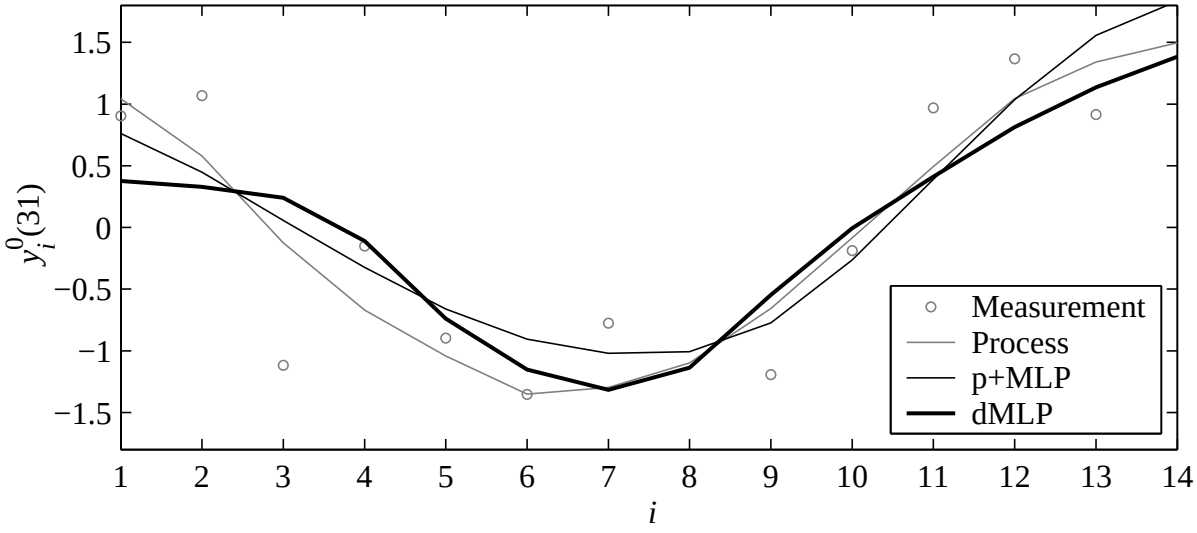
1.5
1
0.5
0
−0.5
−1
−1.5
$y_i^0(31)$
1
2
3
4
5
6
7
8
9
10
11
12
13
14
i
Measurement
Process
p+MLP
dMLP

Fig. 6.

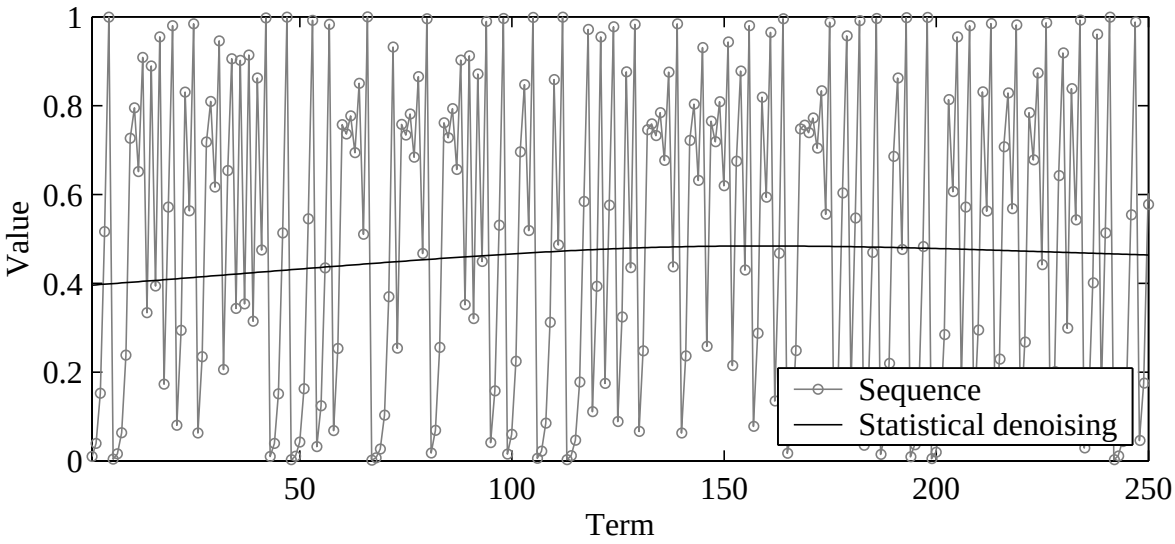
1
0.8
0.6
0.4
0.2
0
Value
50
100
150
200
250
Term
Sequence
Statistical denoising

Fig. 7.

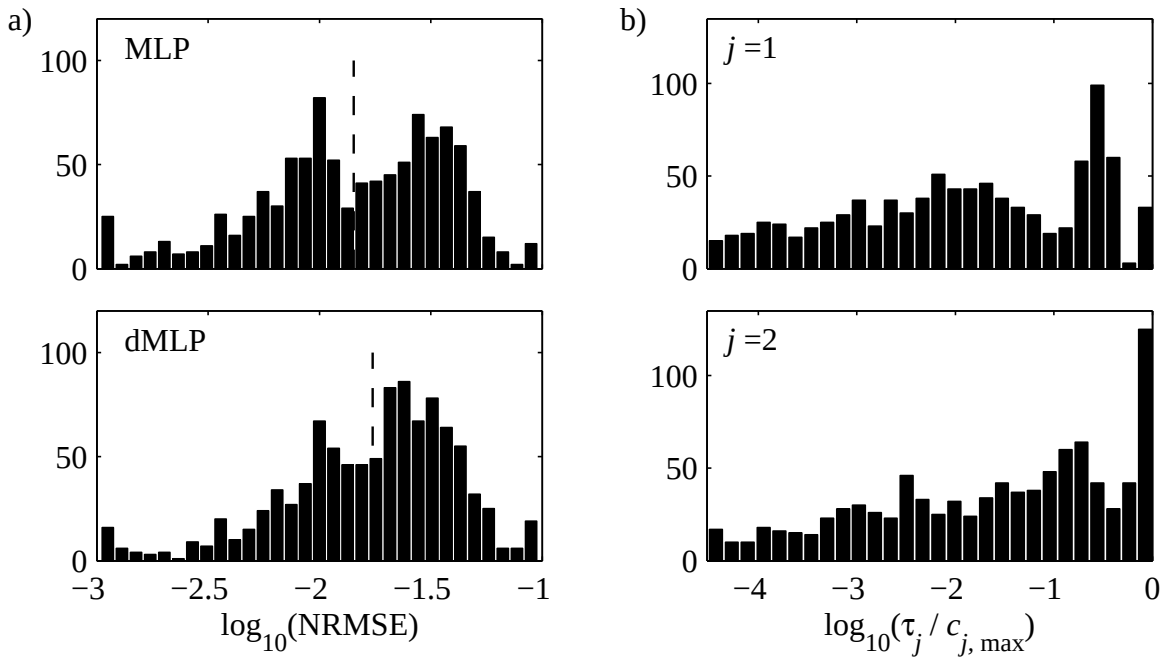
a)
MLP
dMLP
100
50
0
−3
−2.5
−2
−1.5
−1
log10(NRMSE)
b)
j =1
j =2
−4
−3
−2
−1
0
log10(τj / cj, max)

Fig. 8.

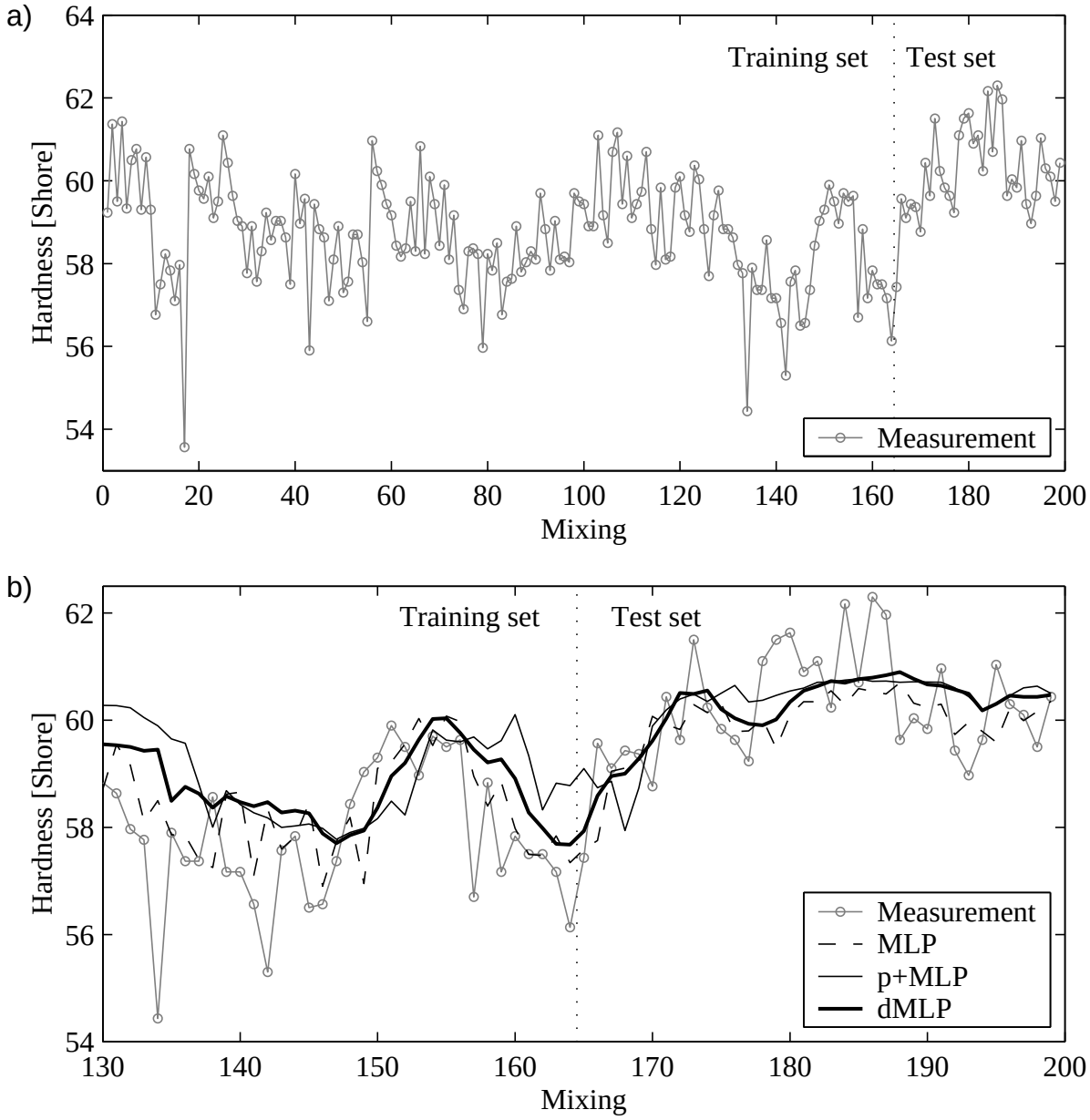
a)
Training set
Test set
Measurement
Hardness [Shore]
Mixing
b)
Training set
Test set
Measurement
MLP
p+MLP
dMLP
Hardness [Shore]
Mixing

Fig. 9.

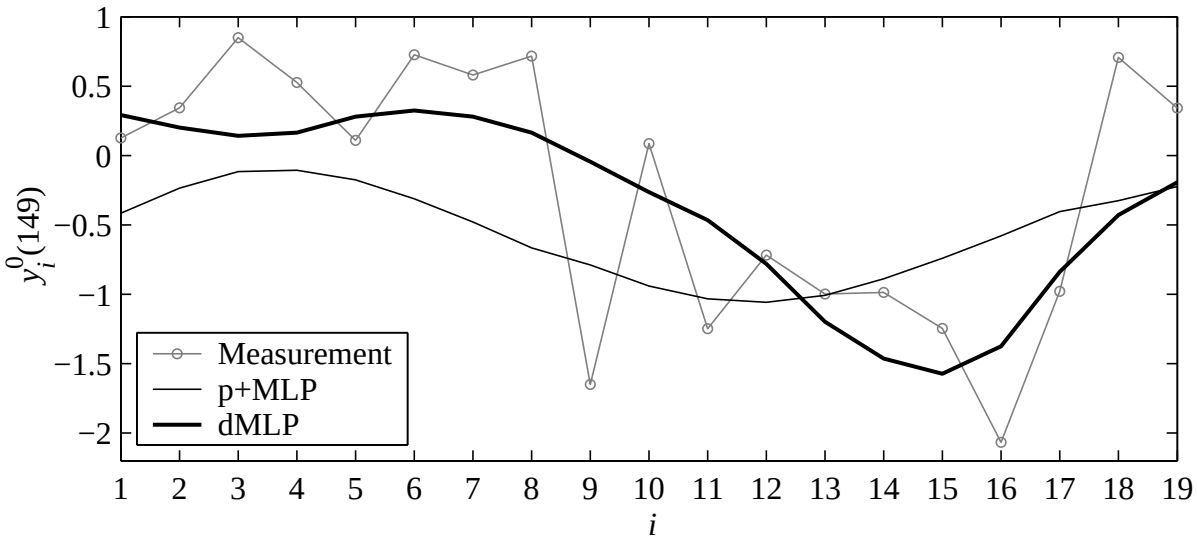
1
0.5
0
−0.5
−1
−1.5
−2
$y_i^0(149)$
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
i
Measurement
p+MLP
dMLP

Fig. 10.

Biography

Uroš Lotrič received his B. Sc. degree in physics and M. Sc. and Ph. D. degrees in computer science from the University of Ljubljana, Slovenia in years 1994, 1997 and 2000, respectively. He currently holds the position of teaching assistant on the Faculty of Computer and Information Science, University of Ljubljana, Slovenia. His research interests include neural networks, wavelets and their applications.