Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT
Neurocomputing 55 (2003) 593–625

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Audio signal processing by neural networks☆

Aurelio Uncini*

Dipartimento INFOCOM, University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy

Accepted 11 March 2003

Abstract

In this paper a review of architectures suitable for nonlinear real-time audio signal processing is presented. The computational and structural complexity of neural networks (NNs) represent in fact, the main drawbacks that can hinder many practical NNs multimedia applications. In particular efficient neural architectures and their learning algorithm for real-time on-line audio processing are discussed. Moreover, applications in the fields of (1) audio signal recovery, (2) speech quality enhancement, (3) nonlinear transducer linearization, (4) learning based pseudo-physical sound synthesis, are briefly presented and discussed.

Keywords: Nonlinear audio signal processing; Neural networks for signal processing; Subband adaptive nonlinear filters; Spline neural networks; Speech enhancement; Signal recovery; Signal predistortion; Physical model sound synthesis

1. Introduction

In the last years the technologies related to multimedia applications have greatly increased and the neural networks (NNs) paradigm seems to be one of the best methodologies for the treatment of incomplete information and difficult nonlinear digital signal processing (DSP) problems [37]. NNs, in fact, represent in some way a central technology for many ill-posed data processing: due to universal approximation capabilities NNs are able to approximate unknown systems based on sparse sets of noisy data (see e.g. [29,27,11]). Although a lot of NN's applications concern classification problem, a growing interest has been devoted in nonlinear time series prediction and in

☆ This work is supported by PRIN project of the "Ministero della Ricerca Scientifica e Tecnologica" of Italy.

* Fax: +39-06-4873300.
E-mail address: aurel@ieee.org (A. Uncini).

0925-2312/03/$ - see front matter
doi:10.1016/S0925-2312(03)00395-3

complex nonlinear dynamic modeling [47]. Moreover, one of the main drawbacks that can hinder practical NNs application in multimedia, depends on their computational and structural complexity.

Classical approaches for nonlinear DSP are based on specific and efficient architectures e.g. median and bilinear filters, some spectral analysis techniques or on generic nonlinear architectures suitable for a large class of problems but usually complex e.g. Volterra filters, non linear state equations, polynomial filters, functional links, etc., [64,50,41,45,32]. In other words typical nonlinear DSP approaches consist of design specific algorithms for specifics problems.

Neural networks (the multi-layer perceptron (MLP) [14,30,9], the time delay neural networks (TDNN) [38,65], and recurrent neural networks (RNN) [69,68,66,7,2,21,25]), have been used extensively in the past for functional approximation of continuous nonlinear mappings, (see e.g. [29,27,11,38,65–69,6,7,2,21,25,19]). Successful functional approximation depends on appropriate selection of the parameter values. This selection is usually made through supervised learning where a training set of input-output pairs is available and the network is trained to match this set according to some pre-specified criterion function. When the criterion function is the sum of squared errors, a popular algorithm is the well-known backpropagation (BP) training procedure.

The MLP and RNNs represent an adaptive circuit which extend and generalize the simple adaptive linear filter in nonlinear domain. By adding in some way delay lines NN filters can be viewed as an extension of linear adaptive filters to deal with nonlinear modeling tasks [29,67,28,43]. It is well known, in fact, that a large amount of DSP techniques are based on linear models, but in some cases the nature of the problems are nonlinear and obviously in these cases nonlinear general purpose architectures are needed.

Despite the formal elegance of the neural model, several problems should be solved. First of all is the model selection. Given an input-output relation the problems are: (1) the determination of the inputs number, (2) the number of neurons in the hidden layers in order to have a correct approximation and (in the case of dynamic processes) (3) how put memory (delay line) in the model.

Although there are several papers dealing with the problem of network topology determination, usually the numbers of layers and neurons are specified by heuristic procedure.

The aim of this paper is the examination of some discrete-time neural architectures for real-time on-line nonlinear signal processing (especially audio processing) applications. We review some general models of time-delay multilayer NNs and in particular some kinds of recurrent networks with local feedback called locally RNN (LRNN). We review, also, NNs architectures with suitable flexible activation functions which allow: very small networks, easy and fast learning processes and mitigate the problem of topology determination.

The paper is organized as follows. Section 2 presents a review of some neural architectures for real-time on-line signal processing. Multilayer static and dynamic time-delay neural networks, adaptive spline neural networks, multirate subband neural networks and their on-line learning algorithms are also reviewed and discussed in the context of DSP applications.

Section 3 presents some NNs based nonlinear audio processing applications. In particular we discuss about: audio signal recovery, speech quality enhancement, nonlinear transducer linearization, and finally, a learning based pseudo-physical sound synthesis is presented.

2. Neural architectures for real-time audio DSP

2.1. MLP with external memory

Although linear adaptive filter theory is well-known and consolidated, its extension to the nonlinear domain is a field of great interest and in continuous expansion. In this Section some neural architectures suitable for adaptive nonlinear filtering are presented.

The formulation of transversal and recursive filters can be easily extended to the nonlinear domain: in the case of discrete-time sequences the filter can be described through a relationship between the input sequence $\{x[t], x[t-1], \ldots\}$ and the output sequence $\{y[t], y[t-1], \ldots\}$. The general form are expressed as

$$y[t] = \Phi\{x[t], x[t-1], \ldots, x[t-M+1]\}, \tag{1}$$

$$y[t] = \Phi\{x[t], x[t-1], \ldots, x[t-M+1], y[t-1], \ldots, y[t-N]\}. \tag{2}$$

In the first expression the output is a nonlinear function of the inputs (present and past samples): in other words Eq. (1) represents a nonlinear generalization of linear finite impulse response filter (FIR). The output signal $y[t]$ in equation (2) is also a function of past output signal: so it represents a nonlinear generalization of linear infinite impulse response filter (IIR). The equation (2) represents a general form usually called nonlinear autoregressive moving average (NARMA) model. The indexes M and N, represent the filter memory length and the couple (N,M) is defined as filter order.

The easiest way to get dynamics from a MLP network is the use of external tapped delay lines (TDL) [43], [see Figs. 1 and 2] subsuming many traditional signal processing structures, including FIR-IIR filters, and gamma memory NN [16], for which the delay operator, used in conventional TDL's, is replaced by a single pole discrete-time filter. These networks are universal approximators for dynamic systems [9,10], just as feedforward MLP's are universal approximators for static mappings [14,30].

Concerning the previous general structure we can assert: (1) the problem of the determination of the optimum filter order (N,M) requires some *a priori* knowledge of the statistics of the input signal; (2) filtering of high non-stationary signals requires that the filter free parameters ($\mathbf{w} \in R$) can vary fast so that it is possible to track the input's statistic variation. Moreover, if in equations (1) and (2) Φ is a linear function, there exists a huge number of methods for the determination of the free filters parameters (filter synthesis). A family of adaptive algorithms, suitable for transversal filters, is derived from the least square error minimization [29,28].

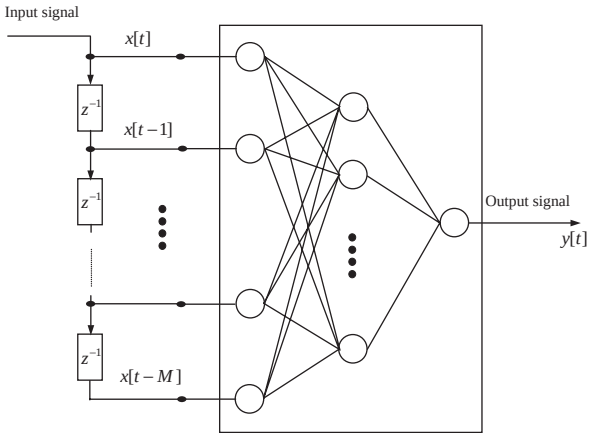

Fig. 1. Buffered MLP structure with input TDL.

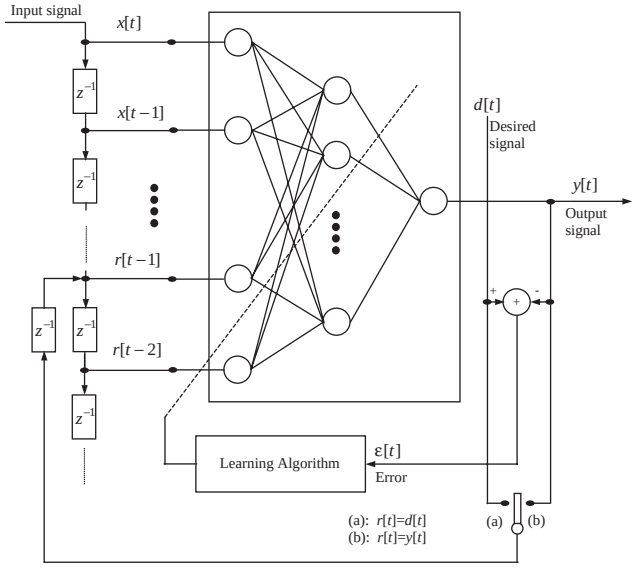

Fig. 2. Buffered MLP with input and output TDL. Switch position: (a) signal correction; (b) error correction.

2.2. Learning algorithm for MLP with external memory

When a MLP network is used as nonlinear adaptive filter its activity is defined from the input sequence $x[t]$ (with possible infinite length) and from the desired signal $d[t]$. For each sample t an error $\varepsilon[t]$ is defined as a difference between the desired signal $d[t]$ and the network's output $y[t]$: $\varepsilon[t]=d[t]-y[t]$. As an example in system identification

problems $d[t]$ represents the output of the system to be modeled while in prediction task $d[t]$ is equal to the input signal at time $t+\tau$ (such that, $d[t]=x[t+\tau]$).

The free network parameters $\mathbf{w}\in R$ (or weights) are determined by the minimization of a certain error norm (usually L_2). In the case of static networks the most adopted criterion is the so called Least Square minimization (LS) and the objective function to minimize is defined as

$$J(\mathbf{w})=\frac{1}{K}\sum_{p=1}^{K}\varepsilon(p)^2. \tag{3}$$

The learning algorithm minimizes the quantity $J(\mathbf{w})$ over a finite sample set K (learning phase); then the network is used with the "frozen" weight (forward mode).

In the case of adaptive filters the error minimization follows an on-line procedure: i.e. the error is minimized during the filtering operation. This means that: (1) the sequence is considered of infinite length, (2) the process can be time-variant. The learning, that can be viewed as a dynamical process, is implemented through a finite length memory mechanism (called forgetting mechanism). The time memory limitation is implemented through a sliding window of T_c length (usually of rectangular form). The functional to be minimized is then

$$J(t,\mathbf{w})=\frac{1}{2}\sum_{p=t-T_c+1}^{t}\varepsilon(p)^2. \tag{4}$$

The choice of T_c is strongly problem-dependent and it is correlated to the degree of input signal nonstationarity.

The learning algorithm for static neural networks with external delay lines is then the simple well-known backpropagation algorithm (e.g., see [29]). The dynamic behavior is, in fact, delegated to the network feed mechanism: the input sequence feeds the network through a sliding window (the delay line) and the input samples are translated at each learning step [43].

In parallel with the development of adaptive nonlinear filters with FIR structures, a recursive IIR architecture can be implemented (see Fig. 2). The motivation for such developments are similar to those arising in ordinary linear filtering applications. A recursive structure can potentially produce comparable results with far fewer coefficients, and consequently with a much lower computational burden. This potential is not easily obtained, however, since the computational gain is offset by increased problems in guaranteeing learning mode stability and convergence.

Although the learning in transversal FIR networks is univocally defined, for recursive networks with output feedback delay line, we have two learning modalities [48]: (1) signal correction; (2) error correction. In signal correction modality the network is fed with desired signal $\{d[t-1],d[t-2],\ldots,d[t-N]\}$ while it the error correction mode we use the delayed output signal $\{y[t-1],\ldots,y[t-N]\}$.

For linear case the error correction mode shows better convergence property [48], but the initial weights condition may produce some numerical stability problems. So in a practical case we start the learning in signal correction mode and after few iteration we switch on error correction mode.

2.3. MLP with internal memory

In order to have NNs with dynamic behavior the buffer can be applied to the input of each neuron. In this way each NN's weight (or synapse) can be implemented as an FIR (MA model) or IIR filter (ARMA model).

The main example of implementation of feedback is the classical fully recurrent neural network [69,68], i.e., a single layer of neurons fully interconnected with each other, or several such layers. Such recurrent networks however exhibit some well known disadvantages: a large structural complexity ($O(n^2)$ weights are necessary for n neurons) and a slow and difficult training. In fact they are very general architectures which can model a large class of dynamical systems, but on specific problems simpler dynamic neural networks which make use of available priori knowledge can be better used.

In the past few years, a growing interest has been devoted to methods which allow introduction of temporal dynamics into the multilayer neural model. In fact the related architectures are less complex and easier to train with respect to the fully recurrent networks. The major difference among these methods lies in how the feedbacks are included in the network: different architectures arise depending on how the ARMA model is included in the network.

The first architecture is the IIR-MLP proposed by Back and Tsoi [2], where static synapses are substituted by conventional IIR adaptive filters (see Fig. 3(a)). The second architecture is the local feedback recurrent multi-layer network (LF-MLN) studied by Frasconi et al. [21]. The output of the neuron summing node is filtered by an autoregressive (AR) adaptive filter (all poles transfer function) before feeding the activation function (activation feedback); in the most general case the synapses are FIR adaptive filters (see Fig. 3(b)). The LF-MLN is a particular case of the IIR-MLP, when all the synaptic transfer functions of the same neuron have the same denominator.

The third structure is the output-feedback LF-MLN by Gori et al. [25]. In this architecture the IIR filter is not simply placed in the classical neuron model but is modified to make the feedback-loop pass through the nonlinearity, i.e., the one time step delayed output of the neuron is filtered by a FIR filter whose output is added to the inputs contributions, providing the activation. Again in the general model the synapses can be FIR filters (see Fig. 3(c)).

At last Fig. 3(d) shows the architecture proposed by Mozer in [42] (with one delay feedback dynamic units in the first layer only) and by Leighton and Conrath in [39] (multiple delays and no restriction on the position of dynamic units). It is again a multilayer network where each neuron has FIR filter synapses and an AR filter after the activation function (AR-MLP). It is easy to see that this network is a particular case of the IIR-MLP, followed by linear all-pole filters.

Due to the use of powerful dynamic neuron models, one of the major advantages of locally recurrent neural networks (suitable for audio DSP applications) with respect to buffered MLP's or fully recurrent networks is that a smaller number of neurons are required for a given problem.

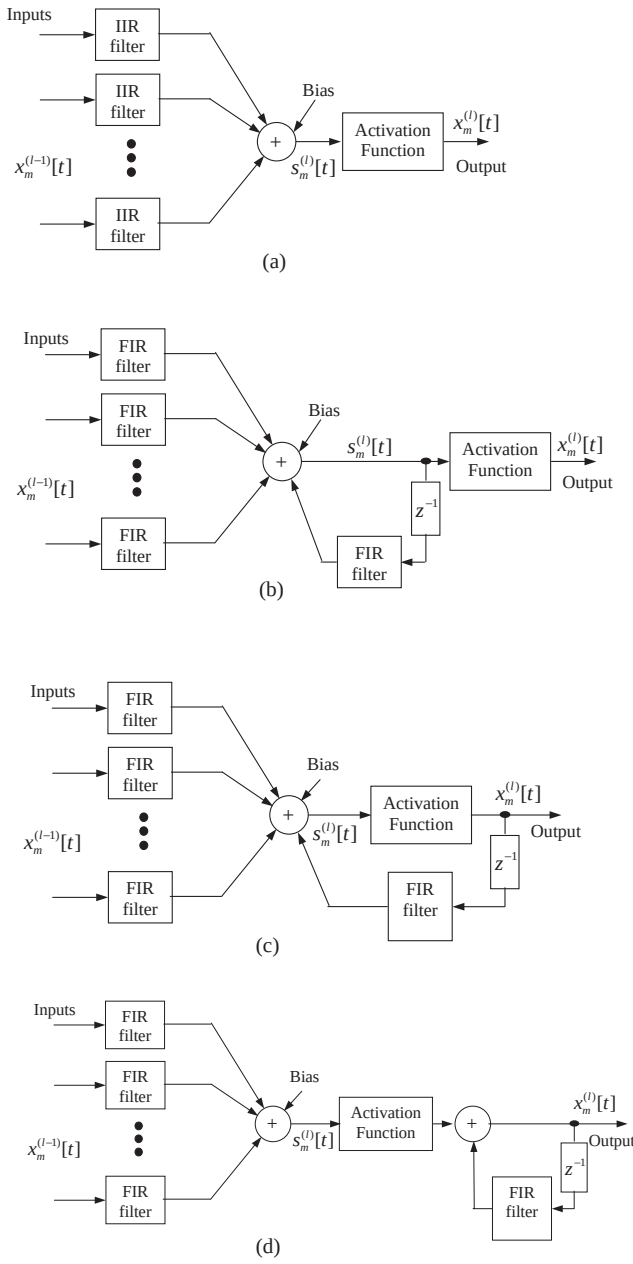

Fig. 3. Dynamic neurons using: (a) IIR/FIR synapses (FIR-IIR/MLP); (b) locally feedback multilayer network (LF-MLN); (c) output feedback; (d) autoregressive MLP.

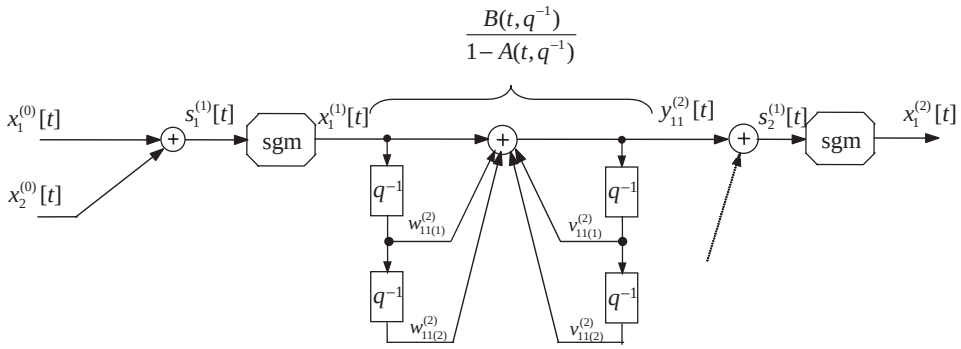

Fig. 4. Example of a simple IIR-MLP which shows the notation.

2.4. Learning algorithm for MLP with internal memory

In the following we will consider only locally recurrent neural networks, particularly IIR-MLP and output feedback MLNs (see Fig. 3(c)) which are the most general and interesting architectures (concerning the learning algorithm FIR and activation feedback structures they can, in fact, be viewed as theirs particular cases).

Some algorithms to train such networks exist, in this paper we describe a gradient-based algorithm for locally recurrent neural networks presented in [7], called recursive back-propagation (RBP) whose on-line version, (causal recursive backpropagation (CRBP)) presents some advantages in audio DSP on-line applications.

An IIR-MLP consists in a neural network where each synapse is replaced with a transfer function with poles and zeros, which are the AR and MA parts respectively.

Using a notation introduced in [67], extended in [7], and with reference to Fig. 4 the forward phase can be described as

$$y_{nm}^{(l)}[t] = \sum_{p=0}^{L_{nm}^{l}-1} w_{nm(p)}^{(l)} x_m^{(l-1)}[t-p] + \sum_{p=1}^{I_{nm}^{l}} v_{nm(p)}^{(l)} y_{nm}^{(l-1)}[t-p], \tag{5}$$

$$s_n^{(l)}[t] = \sum_{m=0}^{N_{l-1}} y_{nm}^{(l)}[t]; \quad x_n^{(l)}[t] = \mathrm{sgm}(s_n^{(l)}[t]), \tag{6}$$

where the indexes $(L_{nm}^{l}-1)$ and I_{nm}^{l} represent the MA and AR parts, respectively, of the nth neuron of the lth layer relative to the output of the $(l-1)$th layer. N_l represents the number of neurons of the lth layer. The quantities $w_{nm(p)}^{(l)}$ and $v_{nm(p)}^{(l)}$ represent respectively the coefficients of the MA and AR parts of the corresponding synapse (the weights $w_{n0}^{(l)}$ are the bias terms).

For (5), the direct form I of the IIR filter has been used [28], but other structures are possible. In particular, direct form II structures allow reduction in the storage complexity as well as in the number of operations [28], both in forward and backward computation. For the sake of clarity the expression corresponding to (5), in the IIR

filter usual notation is reported

$$y[t] = \sum_{p=0}^{M-1} w[t]x[t-p] + \sum_{p=1}^{N-1} v[t]y[t-p], \tag{7}$$

where $y[t]$ is the output, $x[t]$ the input of the IIR filter, $w[t]$ are the coefficients of the MA part, $v[t]$ of the AR part and the orders of the MA and AR parts are $(N-1)$ and $(M-1)$, respectively. Using different notation the input–output relation expressed by Eq. (7) can be written as

$$y[t] = \left(\frac{B(t, q^{-1})}{1 - A(t, q^{-1})} \right) x(t), \tag{8}$$

where

$$A(t, q^{-1}) = \sum_{p=1}^{N-1} v_p[t] q^{-p}; \quad B(t, q^{-1}) = \sum_{p=0}^{M-1} w_p[t] q^{-p} \tag{9}$$

and the term q^{-1} is the delay operator, i.e. $q^{-\tau}(s[t]) = s[t-\tau]$.

In order to derive the learning algorithm, let $\varepsilon_n[t] = d_n[t] - x_n^{(M)}[t]$, the functional (4) to be minimized (called in this case global error) becomes

$$J(\theta) = E^2 = \sum_{t=1}^{T} \sum_{n=1}^{N_M} \varepsilon_n^2[t], \tag{10}$$

where θ, represents both $\mathbf{w}$ and $\mathbf{v}$ weights ($\theta = \mathbf{w} \cup \mathbf{v}$), T is the duration sequence and the index N_M represents the number of net outputs i.e. the number of neurons in the Mth layer.

Let us define the usual quantities "*error*" and "*delta*" of the backpropagation algorithm

$$e_n^{(l)}[t] = -\frac{1}{2} \frac{\partial E^2}{\partial x_n^{(l)}[t]}; \quad \delta_n^{(l)}[t] = -\frac{1}{2} \frac{\partial E^2}{\partial s_n^{(l)}[t]} \tag{11}$$

and, as in the static case, it holds

$$\delta_n^{(l)}[t] = e_n^{(l)}[t] \mathrm{sgm}'(s_n^{(l)}[t]). \tag{12}$$

It is possible to derive the RBP learning algorithm which is described in the signal flow graph (SFG) of Fig. 5 (see [7] for the proof).

In the SFG of Fig. 5 there is a feedback path with a positive index delay operator (q^1) so the RBP algorithm is *non-causal* (i.e. the weights variation depend upon the future time index $t+1$). For this reason RBP can be implemented only in the so called *batch* mode, i.e. the weight adaptation can be performed at the end of the window time T by the accumulation of the weight variations computed at each learning step t. For many DSP (in particular audio) applications due to the intrinsic input-output delay, batch algorithms can not be used. For this reason an on-line algorithm derived

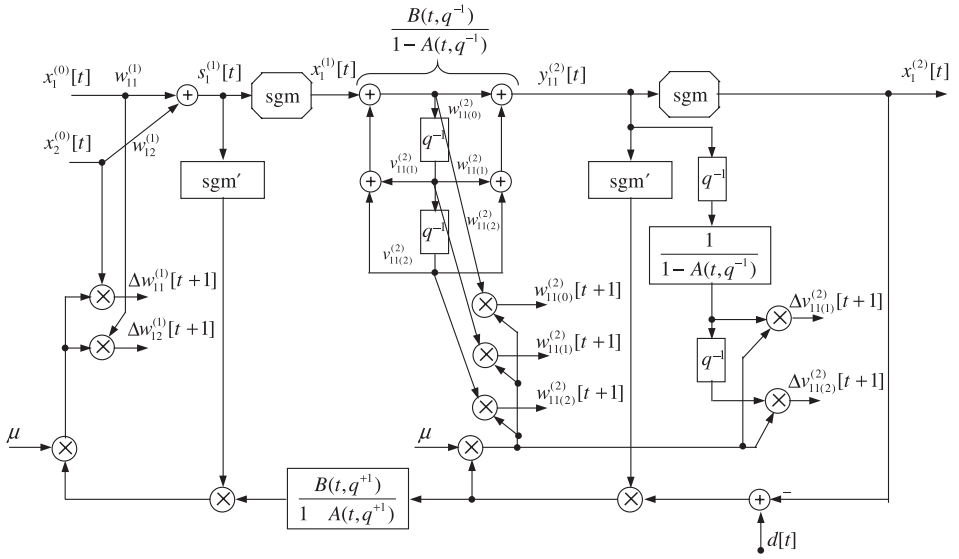

Fig. 5. Signal Flow Graph of RBP of the IIR-MLP of Fig. 4.

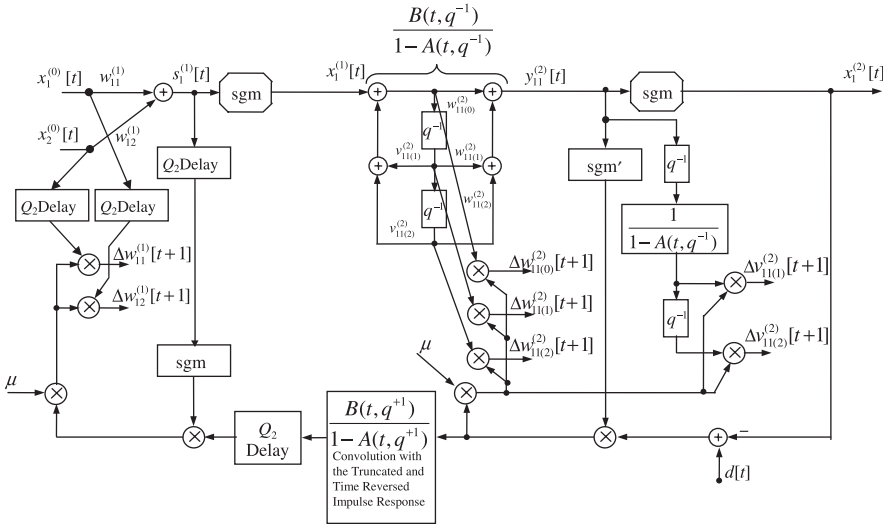

Fig. 6. Signal Flow Graph of causalized recursive backpropagation CRBP of the IIR-MLP of Fig. 4.

from the RBP and called causal recursive backpropagation (CRBP) has been proposed in [6,7]. With reference to the SFG reported in Fig. 6, CRBP algorithm introduces a suitable number of delays in the weights adaptation scheme in order to remove the noncausality of the RBP. In this way the weights-variation ($\nabla\theta$) for the lth layer can

be written as

$$\theta^{(l)}[t+1] = \theta^{(l)}[t] + \nabla\theta^{(l)}[t+1-D_l], \tag{13}$$

where D_i represents a delay

$$D_l = \begin{cases} 0, & \text{if } l = M, \\ \sum_{i=l+1}^{M} Q_i & \text{if } l < M. \end{cases} \tag{14}$$

The CRBP algorithm includes as particular cases backpropagation (BP) [29,67], temporal backpropagation (TBP) [69], backpropagation for sequences (BPS) [25], and Back–Tsoi algorithm [2]. Moreover it allows the training of generalized output and activation feedback MLN's which have no constraint on the position of the dynamic units, implementing communications among them, as suggested in [51] for a better modeling.

2.5. Neural network with flexible spline activation function

Introduced in [62,26] the adaptive spline activation function neural networks (ASNNs) are built using neurons with flexible activation function (FAF). This FAF is implemented by a small look-up-table (LUT) containing some spline control points. The neuron's output is then computed by a simple interpolation scheme over the LUT parameters using Catmull-Rom (CR) or B-spline cubic basis. During the learning phase, the shape of the activation function can be modified by adapting the spline control points.

It is well-known that under certain regularity conditions, the NN representation capabilities depend on the number of free parameters, whatever the structure of the network [1]. Hence, FAF can reduce the number of interconnections and therefore the overall network complexity, since they now contain free parameters. It follows that very small networks can be able to solve difficult nonlinear problems: so ASNN represents a suitable paradigm for real and complex domain signal processing applications [58] and due to this interesting performance more recently they have been successfully used in an unsupervised context for blind signal processing problems [54]. Moreover, from theoretical points of view, the authors in [62] and in their further developments [55] demonstrated that such neuron architecture can improve approximation and generalization capabilities of the whole network. In particular, this neuron architecture, sometimes called generalized sigmoidal (GS) neuron, presents several interesting features: (1) it is easy to adapt, (2) it can retain the squashing property of the sigmoid, (3) it has the necessary smoothing characteristics, (4) it is easy to implement both in hardware and as software simulations.

The ASNN has a multilayer structure and can be implemented by using both standard and dynamic neurons like in IIR-MLP.

Referring to papers [62,26] for more details, here we give some notes on the realization of an adaptive flexible activation function.

The splines are, generally, smooth parametric curves, divided into multiple tracts; they are also able to preserve the continuity of derivatives at the joining points. In the

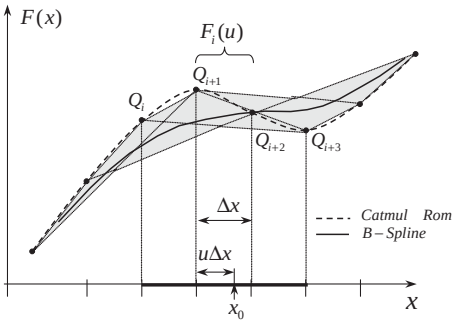

Fig. 7. Catmul-Rom and B-spline interpolation of control points examples.

planar case the graph $F_i(u)$ of the ith curve span is represented by

$$F_i(u) = [F_{xi}(u), F_{yi}(u)]^{\mathrm{T}}, \tag{15}$$

where $u \in [0,1)$ is the local span parameter, T is the transpose operator and the two polynomial functions $F_{xi}(.)$, $F_{yi}(.)$ describe the curve tract behavior in the two coordinates x and y. The ith curve *spline basis functions* tract can be written in the form

$$F_i(u) = \sum_{k=0}^{d} Q_{i+k} b_{i,d}(u) = \begin{bmatrix} \sum_{k=0}^{d} q_{x,i+k} b_{i,d}(u) \\ \sum_{k=0}^{d} q_{y,i+k} b_{i,d}(u) \end{bmatrix}, \tag{16}$$

where $b_{i,d}(u)$ is the ith element of the spline basis (a polynomial of degree d in the variable u), and $Q_{i+k} = [q_{x,i+k}, q_{y,i+k}]^{\mathrm{T}}$ are the $(d+1)$ *control points* of the ith curve tract: moving such points on the real plane will affect the curve shape.

We have chosen to represent the activation functions through the concatenation of even more local spline basis functions, controlled by only four coefficients. To keep the cubic characteristic, we have used a Catmull-Rom [8] or B-spline [59] cubic spline.

Referring to Fig. 7, the ith curve span in (16), expressed in matrix form can be rewritten as

$$F_i(u) = \mathbf{T} \cdot \mathbf{M} \cdot \mathbf{Q}_i, \tag{17}$$

where $\mathbf{T} = [u^3 \ u^2 \ u \ 1]$, $\mathbf{Q}_i = [Q_i \ Q_{i+1} \ Q_{i+2} \ Q_{i+3}]^{\mathrm{T}}$, and the matrix $\mathbf{M}$ assumes the value:

$$M = \frac{1}{2} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix}, \tag{18}$$

for Catmull–Rom spline base and

$$\mathbf{M} = \frac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}, \tag{19}$$

for the B-spline base.

From Eq. (17) such a spline (see Fig. 7 for details) interpolates the points $\mathbf{Q}_{i+1}$ ($u=0$) and $\mathbf{Q}_{i+2}$ ($u=1$) and has a continuous first derivative (B-spline also the second), useful for the backpropagation-like learning algorithm.

In general, Eq. (17) represents a curve and to obtain a function we have ordered the x-coordinates according to the rule $q_{x,i} < q_{x,i+1} < q_{x,i+2} < q_{x,i+3}$.

To find the value of the local parameter u we have to solve the equation $F_{x,i}(u)=s_0$, where s_0 is the activation of the neuron. This is a third degree equation, whose solution can make the numerical burden of the learning algorithm heavier. The easiest alternative consists in setting the control points uniformly spaced along the x-axis (Δx is the step): this choice allows us to reduce the third degree polynomial $F_{x,i}(u)$ to a first degree polynomial and the equation for u becomes linear.

$$F_{x,i}(u) = u\Delta x + q_{x,i+1}. \tag{20}$$

Moreover, the fixed parameter Δx is the key tool for smoothness control. Now we can calculate the output of the neuron by $F_{y,i}(u)$.

As we decided to adapt only the y-coordinates of the spline knots, we must initialize them before starting the backpropagation learning: for this reason we take, along the x-axis, $N+1$ uniformly spaced samples from a sigmoid, or from another function assuring universal approximation capability [14,30].

Referring to Fig. 8 for the formalism, the FAF is composed of two functional blocks. The first block, called GS1, performs the mapping of the linear combiner output to the parametric spline domain, i.e. the x-axes inversion (equations inside block GS1 of Fig. 8).

The block GS2 computes the neuron output by using the activation function's control points, stored in the GS2-LUT, and the polynomial coefficients of Eq. (17). It follows

$$x_k^l = F^{(l)}_{k,i_k^{(l)}}(u_k^{(l)}), \tag{21}$$

where the function $F^{(l)}_{k,i_k^{(l)}}$ is the $i_k^{(l)}$th tract of the activation spline curve of the kth neuron of the lth layer.

2.6. Learning algorithm for the ASNN

The learning algorithm for ASNN can be based on the classical backpropagation where, referring to Eq. (10), the functional J to be minimized is a function of both the weights $w_{kj}^{(l)}$ and the local activation function free parameters $Q_{ki}^{(l)}$. Using the same

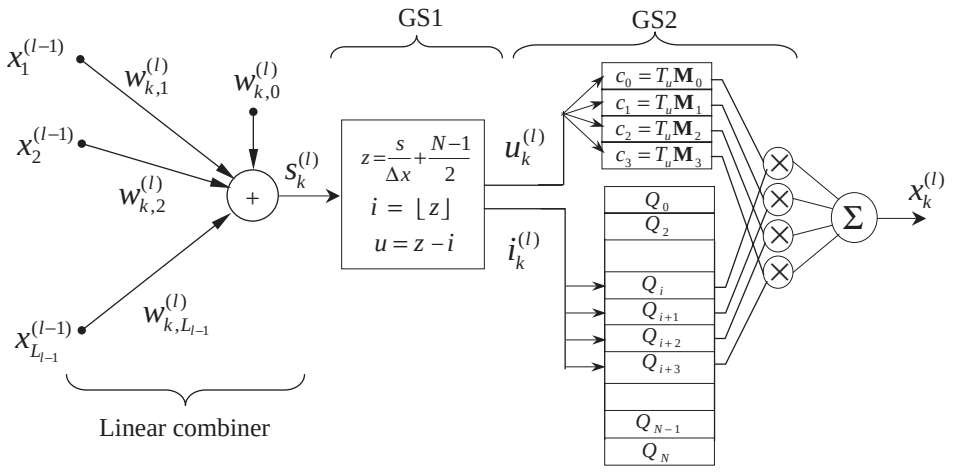

Fig. 8. The GS neuron architecture. Block GS1 computes the parameters $u_k^{(l)}$ and $i_k^{(l)}$, while GS2 computes the neuron's output through the spline patch determined by GS1. In the block GS2, the $\mathbf{M}i$ are the columns of the matrix (16) or (17) depending on the selected basis.

notation proposed in previous Section, for the tth step of the learning phase we have the following recursive equation:

FOR $l = M, \ldots, 1$
FOR $k = 0, \ldots, N_l$,
FOR $j = 0, \ldots, N_l - 1$,

$$e_k^{(l)} = \begin{cases} (d_k - x_k^{(l)}), & l = M, \\ \sum_{p=1}^{N_{l+1}} \delta_k^{(l+1)} w_{pk}^{(l+1)}, & l = M-1, \ldots, 1, \end{cases} \tag{22}$$

$$\delta_k^{(l)} = e_k^{(l)} \left(\left. \frac{\mathrm{d}F_{k,i_k^{(l)}}^{(l)}(u)}{\mathrm{d}u} \right|_{u=u_k^{(l)}} \right) \frac{1}{\Delta x}, \tag{23}$$

$$\Delta w_{kj}^{(l)} = \mu \delta_k^{(l)} x_j^{(l-1)}, \tag{24}$$

$$w_{kj}^{(l)}[t+1] = w_{kj}^{(l)}[t+1] + \Delta w_{kj}^{(l)}[t], \tag{25}$$

FOR $m = 0, \ldots, 3$

$$\Delta Q_{k,(i_k^{(l)}+m)}^{(l)} = e_k^{(l)} \left(\frac{\partial F^{(l)}}{\partial Q_{k,(i_k^{(l)}+m)}} \right)^{(l)} = \mu_q e_k^{(l)} b_{k,m}^{(l)}(u_k^{(l)}), \tag{26}$$

$$Q_{k,(i_k^{(l)}+m)}^{(l)}[t+1] = Q_{k,(i_k^{(l)}+m)}^{(l)}[t+1] + \Delta Q_{k,(i_k^{(l)}+m)}^{(l)}[t], \tag{27}$$

(in Eqs. (22)–(26), time index t is omitted) where Δ represents the local approximation of the function error gradient, the terms μ and μ_q are the learning rate for the weights and the activation function parameters respectively and the derivative of $F(u)$ is simply a second order polynomial.

The terms $\Delta w_{kj}^{(l)}[t]$ and $\Delta Q_{k,(i_k^{(l)}+m)}^{(l)}[t]$ are obtained by computing the error derivative with respect to the weights and to the control points of the activation function, respectively. Note that the generalization to IIR-MLP architecture is straightforward.

For the control point adaptation (27), the parameter m rang from 0 to 3, restricting the update to only 4 points. In this step, we consider the parametric value u fixed (i.e. $u = u_k^{(l)}$), so the GS2 block is represented by a function of only four variables (the four control points).

2.7. Subband neural networks

It is well know from linear and adaptive filter theory that subband techniques present several advantages with respect to the full-band approach [70,13,18,44,22,46].

First of all, they achieve computational efficiency by decimating the signal before the adaptive processing. In fact the subband linear adaptive filters present impulse responses that are shorter than full-band adaptive filter although the total number of the free parameters remains the same.

A second interesting property is due to the splitting of the input signal: the eigenvalues spread of the subband-signals' autocorrelation function is reduced and consequently least-squares-like adaptation algorithms present better convergence performance [70,46].

More recently, a subbands multirate architecture has been extended in NN context. It is well known, in fact, that the needed long training can hinder many real-time NN applications. So, since smaller networks are needed for each subband, speed-up both the convergence time and the forward-backward computation, the multirate approach has been used in a on-line (or in continuous learning) mode as a simple nonlinear adaptive filter [12].

An important topic of multirate signal processing regards the choice of the filter banks. Filter banks, in fact, decompose full-band signal spectra in a number of directly adjacent frequencies subbands and recombine the signal spectra by the use of low-pass, band-pass, and high-pass filters. Moreover, in the last two decades several techniques and topologies for the design of filter banks have been proposed. A key part of the design can concern perfect vs. almost perfect reconstruction or uniform vs. non-uniforms bands. [13,18].

The uniform filter bank consists of band-pass filters, partitioning signal spectra into directly adjacent bands of equal width (see Fig. 9(a)). All filters have the same bandwidth and the central frequencies are uniformly spaced on the frequency axis (see for example [13]). The octave filter bank belongs to the Q constant filter bank family, i.e. the ratio between nominal bandwidth amplitude Δ_{f_k} and its central frequency f_k is constant. The bank is implemented in a tree structure as Fig. 9(b) shows: two-channels filter banks, consisting of a low-pass filter $H_{\mathrm{LP}}(z)$ and a complementary high-pass filter $H_{\mathrm{HP}}(z)$, are used as a band separating filter [18].

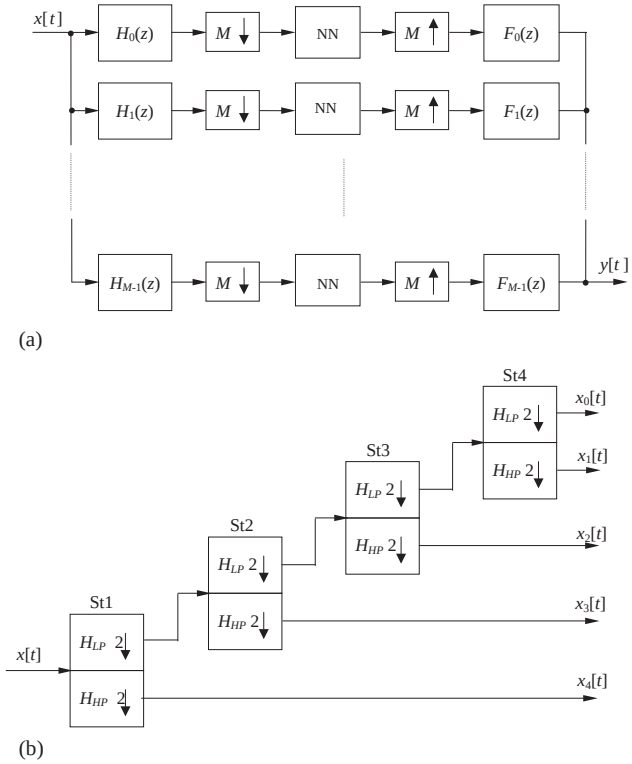

Fig. 9. (a) The M-channel maximally decimated uniform filter bank. (b) A tree-structured octave analysis filter bank.

A subband neural prediction architecture used for audio signal recovery and described in next Section is shown in Fig. 10.

3. Neural network applications to audio signal processing

3.1. Audio signal recovery

The introduction of digital systems in audio signal field let grow the problem of digital audio restoration. Due to the media changing of the new audio-video technology the old analog supports are replaced by digital samples that can be stored as a streaming of numerical data.

Audio restoration is necessary whenever the original signal is corrupted by background or impulsive noise, or whenever a sequence of consecutive samples is missing. Situations of the first type are common with old gramophone recordings: the background noise is due to transducer equipment, while impulsive noise appears as a result of natural surface irregularities, scratches and dust particles. Situations of the second

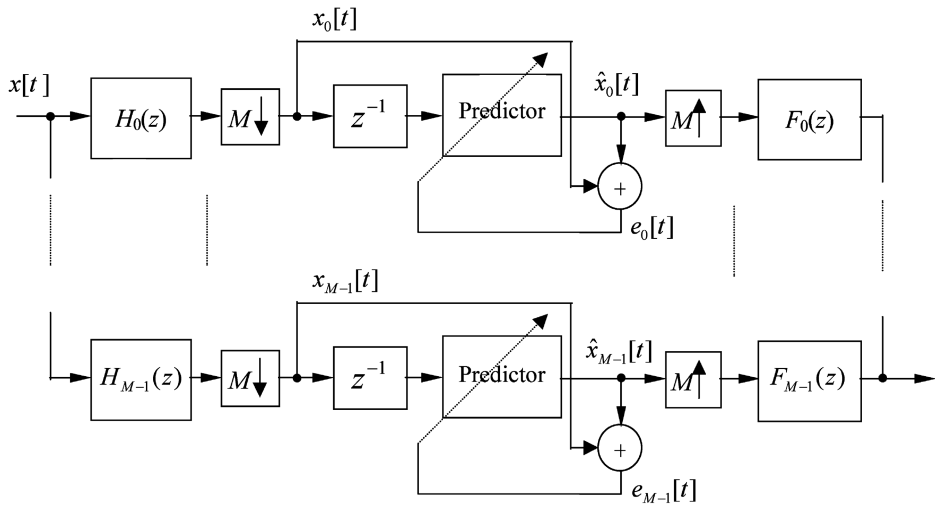

Fig. 10. Subband neural prediction scheme for on-line audio signal processing. Whether output signals are taken from the uniform analysis filter bank or from the octave analysis filter bank, the situation is very similar.

type are common with old magnetic tapes, when due to demagnetization effects a fragment of a musical track is completely missing. The case of impulsive noise and the case of a missing sequence are similar, because only few data are involved, while background noise involves all samples. So two different approaches must be adopted. We concerned on degradations of the first type, in which only a localized sequence must be processed to reconstruct the original signal. This operation is also necessary when reduction of impulsive noise is required: a preprocessing stage localizes noise positions and then a reconstruction stage of corrupted samples is needed [60,61,24].

In [63] a method for missing data reconstruction based on the assumption that signal can be modeled as P order autoregressive process is presented. The model parameters, which are the taps of a linear predictor filter, are calculated using a signal block including missing data. Then the missing samples are predicted in forward and backward mode and the attained results are combined in suitable manner to give missing sequence. The main drawback of autoregressive model based approach is an increasing model order for long sequence reconstruction. Model order must be two or three times the length of missing data sequence. This method is not very appropriate for reconstructing long sequence (over about a hundred of samples at 44.1 KHz sampling rate [60]).

Linear prediction methods have been extended to the nonlinear case using NNs [15]. A P inputs MLP NNs has been used, with hidden layer neurons having bipolar sigmoidal activation function, and a linear neuron in the output layer. The net has been trained using several examples extracted randomly from different uncorrupted signals. A forward and a backward prediction are performed in a similar manner as a linear case. Some experiments have shown performances of the same level of the linear predictor.

Improvement has been obtained adding a training stage using samples before and after the missing data, allowing the net to learn the local characteristics of the signal. Results remain nevertheless comparable with those obtained using linear prediction.

A dual approach is missing data reconstruction in the frequency domain [40]. Spectral components of signal are analyzed using a short time Fourier transform (STFT) analysis with overlapping windows.

More recently a subbands signal reconstruction method, which realizes a trade off among techniques based on time and frequency domains has been proposed [12]. Two different filter banks multirate architectures have been implemented: M channels uniform filter bank and octave filter bank. Each subband is processed by a nonlinear predictor realized using the ASNN architecture (see Fig. 10).

While previous NN approaches involve a long training process, thanks to the small network architecture, needed for each subband and to the FAF, which speeds-up the convergence time and improves the generalization performances, the ASNNs are able to work in on-line (or in continuous learning) mode as simple nonlinear adaptive filters.

The reconstruction of L consecutive missing samples in an audio signal may be considered an extrapolation problem, as represented in Fig. 11. Given the signal trends on the left and on the right of the missing sequence boundaries our task is to fill the gap.

Input signal $x[t]$ is properly selected so that missing sequence is centered. At the output of analysis filter bank we have M signals, $x_1[t],\ldots,x_M[t]$, where M is the number of channels. A forward and backward prediction algorithm is applied to every $x_i[t]$ signal and results are combined using weighing windows as in a cross fade operation. Signals with missing data gap filled by prediction results are then passed to synthesis filter bank to obtain full band reconstructed signal $y[t]$. Fig. 12 shows experimental results of a 45 ms of signal reconstruction using a octave filter bank and ASNNs (for more details see [12]).

3.2. Quality enhancement of speech signal

In this section a system for speech quality enhancement (SQE) is presented. A SQE system attempts to recover the high and low frequencies from a narrow-band telephone speech signal, usually working as a post-processor at the receiver side of a transmission system. The system operates directly in the frequency domain using complex-valued spline neural networks [58].

It is known that a signal sampled at 16 KHz (wide-band speech) has a nominal frequency band from 0 to 8 KHz, while the narrow band telephone speech is limited between 300 and 3400 Hz. The problem is therefore to recover from this narrow band signal the two missing frequency bands: nominally from 0 to 300 Hz and from 3400 to 8000 Hz. This should be made possible by the human speech production physical mechanism, which relates the frequency contents of different bands.

Let $s[t]$ be the narrow-band speech signal whose short-time Fourier transform (STFT) is $S_t(e^{j\omega_k})$, with $S_t(e^{j\omega_k}) \neq 0$ only for $\omega \in [\omega_1, \omega_2]$. Let $\tilde{s}[t]$ be the corresponding wide-band signal; its STFT is now $\tilde{S}_t(e^{j\omega_k}) \neq 0$ for $\omega \in [\omega_0, \omega_N]$ with the position $\omega_0 < \omega_1$ and $\omega_2 < \omega_N$.

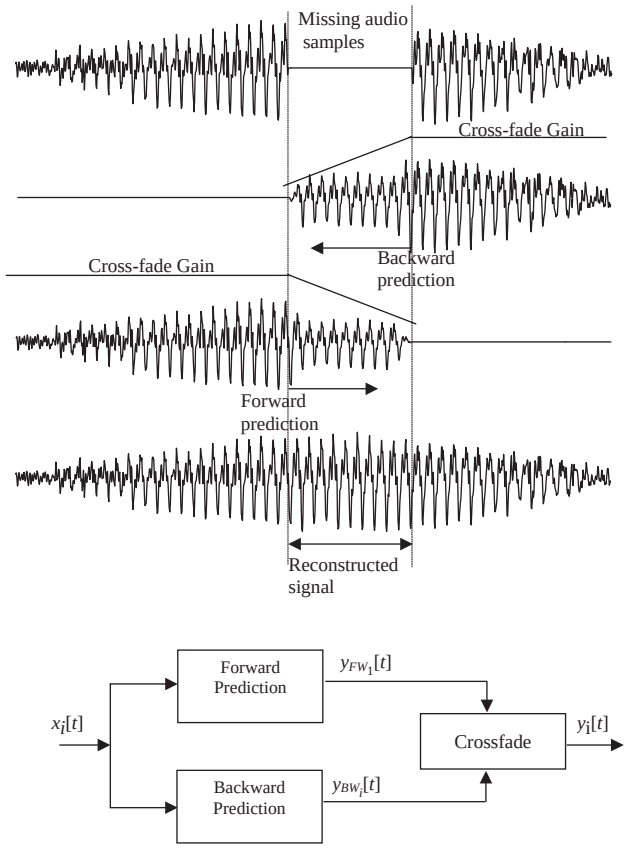

Fig. 11. Forward–backward prediction scheme: example 18 ms (about 800 samples) of missing and reconstructed samples of music audio signal.

A SQE system postulates the existence of an operator Ψ (in general nonlinear), called quality enhancement operator (QEO), such that:

$$\tilde{S}_t(e^{j\omega_k}) = \Psi[S_t(e^{j\omega_k})] \tag{28}$$

or, in terms of STFT:

$$\begin{aligned}\tilde{s}[t] &= \sum_m \left[\sum_k \Psi[S_t(e^{j\omega_k})] e^{j\omega_k t} \right] \\ &= \sum_m \left[\sum_k \Psi \left[\sum_\tau s[\tau] w[m-\tau] e^{-j\omega_k \tau} \right] e^{j\omega_k t} \right],\end{aligned} \tag{29}$$

where $w[.]$ represents the overlapping time window.

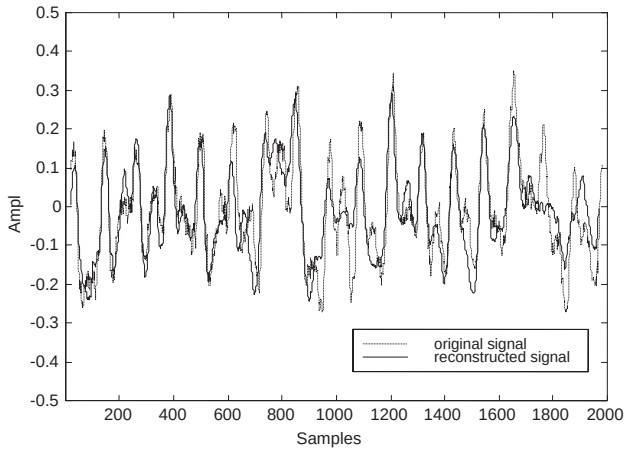

Fig. 12. Example of reconstruction of 2000 samples (45 ms) by octave filter bank and spline neural networks.

With reference to Eq. (29), the SQE system performs a direct STFT on a narrow-band signal up-sampled to 16 KHz, recovers the broad-band signal through the nonlinear operator, and performs an inverse STFT. However, this simple scheme does not attain good performances, since the recovery processes for the lower and the higher band are very different. Moreover the Ψ operator could degrade the narrow-band frequency contents of the original speech signal. Better performances hence can be obtained by splitting the Ψ operator in two different operators Ψ_{L} and Ψ_{H}, for the lower and higher frequencies, respectively. The original narrow-band information are sent to the output without any processing.

The system implements both the Ψ_{L} and Ψ_{H} operators with properly trained complex adaptive spline neural networks, respectively CASNN1 for the first operator and CASNN2 for the second. However, since it is known that the frequency contents of the higher band (3600–8000 Hz) is strongly related to the contents of the narrow-band frequencies mainly for voiced sounds, our SQE system processes differently high frequency voiced and unvoiced sounds. For the first the CASNN2 properly trained only on voiced speech is employed, while for the unvoiced speech the scheme proposed in [71] is also used. The overall SQE scheme is shown in Figs. 13 and 14 (see [57] for more details).

3.3. Loudspeaker linearization by predistortion

Loudspeaker is an electrodynamic transducer used to convert electrical power in-to acoustic power. A general scheme of a loudspeaker is shown in Fig. 15.

An ideal transducer has in the motor mode a constant relationship between electrical input (the current) and mechanical force output. Let F the force, v the velocity, I the current, l the wire of the coil length, E the electromotive force and B the flux density;

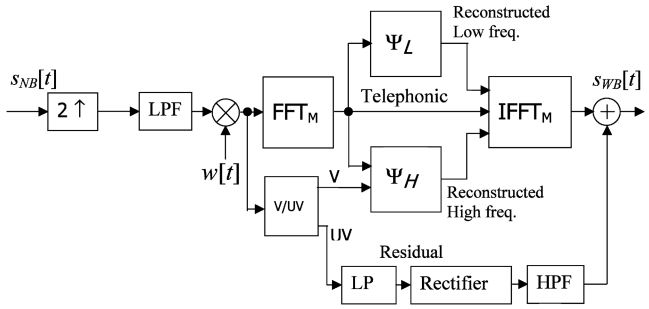

Fig. 13. Scheme of the SQE system. V/UNV is a voiced/unvoiced selector and $w[n]$ represents the overlapping time window.

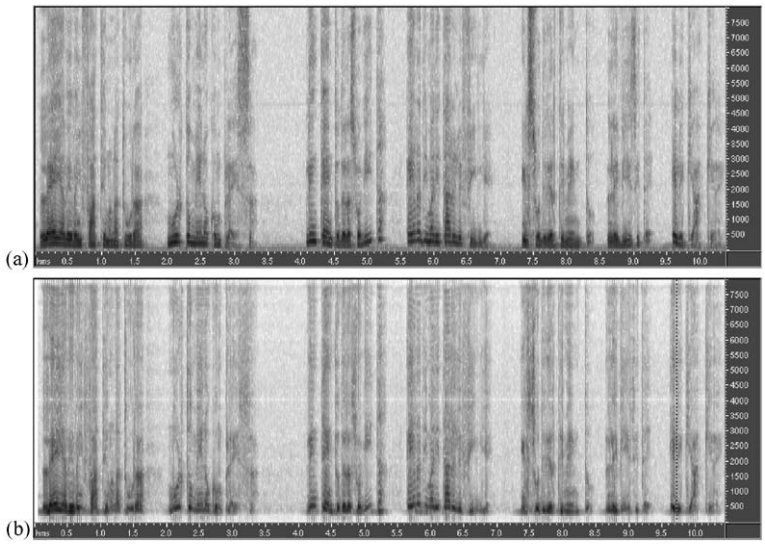

Fig. 14. Time-frequency plots of speech signal containing utterances from different female speakers: (a) original wide-band signal; (b) output signal of the proposed SQE system.

in elementary treatments, the behavior of moving-coil systems is often represented by $F = BlI$ and $E = Blv$. The quantity (Bl) is sometimes called force factor or motor coupling factor.

Real loudspeakers are hard nonlinear dynamic devices. In fact, there are several nonlinearity sources that can be grouped in three principal classes: electromagnetic, mechanical and acoustical.

As an example, an electromagnetic nonlinearity source can be due to the fact that in the case of constant-current drive the force applied to the coil is dependent on its position inside the magnetic circuit. The force $(F = \int_l B\,\mathrm{d}l)$ is a function of the voice-coil excursion x i.e. the coil can go away from the region where the magnetic

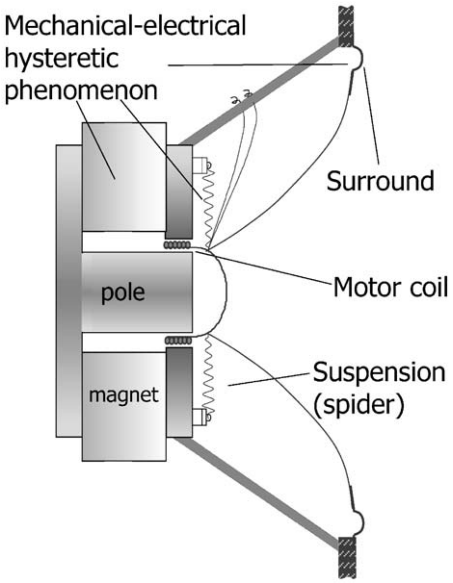

Fig. 15. Simple loudspeaker scheme.

flux density B can be considered constant. A typical force versus displacement is shown in Fig. 16(a).

From the mechanical point of view the force versus displacement curve of the loudspeaker spider[1] and outer rim are not straight lines and show hysteresis. A typical displacement versus force curve is shown in Fig. 16(b). Moreover, the excursion capability of the voice coil is limited (mechanical clipping): this non linearity only occurs at extreme drive level.

The acoustical nonlinearity sources are due to nonlinear acoustic wave radiation, Doppler effect, etc. Usually they can be neglected respect to the electromagnetic and mechanical more evident nonlinearity sources.

Fig. 17 shows the frequency response and distortions of an hi-fi woofer (model SIPE-ASW300) enclosed in a 25 l box used in the following described experiments.

In the past some loudspeaker predistortion architectures were developed either in closed-loop or open-loop control approaches [31,35]. However the design of controllers dedicated to loudspeaker linearization are based on: (1) the identification of a precise loudspeaker model (for example based on the Volterra expansion [64,31,36]); (2) the learning in some way (closed form or using adaptive learning based approach) the predistorter inverse model.

[1] In early loudspeakers, the flexible hinge that held the moving system in place (at the voice coil) was made of leather, and die cut in the shape of a spider's legs. This part came to be known as a "spider", and the part is so named to this day, though the shape of the part has metamorphosed completely. Spiders are generally made of cotton and steam-pressed. The spider can introduce noise into a loudspeaker system from air moving through its holes.

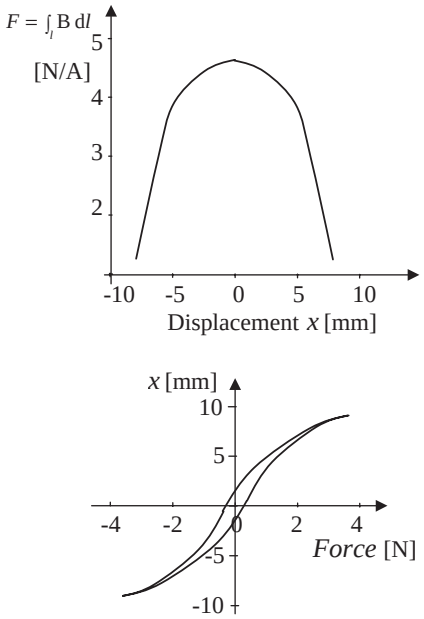

Fig. 16. Typical loudspeaker small signal behavior.

For the loudspeaker identification problem it is well-known that a suitable neural network, thanks to its universal approximation capabilities, can be used for identification of a dynamical nonlinear system [43,9]. So with NN approach we can characterize the whole behavior of the direct-radiation in a loudspeaker independently from its nonlinearity sources. To completely characterize a nonlinear dynamic system by a functional model we need a infinite set of inputs. In our experiments we used some linear sinusoidal sweeps of different amplitudes. In particular we used 3 linear sweeps from 10 to 500 Hz, sampled at 2 kHz, with different amplitude. Several output loudspeaker responses have been acquired in anechoic chamber.

Concerning the predistortion model, several researchers have demonstrated how neural networks can be trained to compensate for nonlinear signal distortion. In particular in digital satellite communication systems predistortion technique inserts a nonlinear module between the input signal and the nonlinear radio frequency high power amplifiers [33,34,4,3]. Using a similar architecture of that one proposed for digital satellite communications systems in [3], in our experiments the loudspeaker model is based on an adaptive spline neural network (ASNN2 in Fig. 18) with 20 inputs (10 delays of the inputs and 10 autoregressive of the desired output), 12 hidden spline neurons (with 28 control points each), a single linear output; it has been trained using a simple (normalized) backpropagation algorithm. The Fig. 19, presents a response of medium amplitude sweep signal of the woofer SIPE-AS300 (up) and response of the neural network loudspeaker model to the same excitation signal (down), after 2500 learning epochs (each epoch consists in the presentation of the whole training set signals).

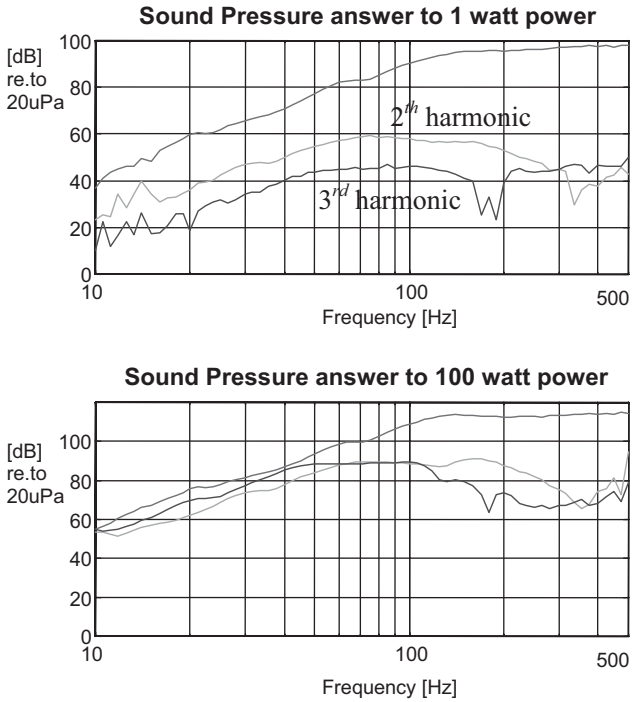

Fig. 17. Frequency response and distortions of the woofer SIPE-ASW300 enclosed in a 25 l box. The microphone is placed at the distance of 1 m from the loudspeaker in anechoic chamber.

Fig. 18 shows the overall loudspeaker-model and predistorter-model learning scheme. Although the loudspeaker-model has been independently estimated (pre-training phase in anechoic chamber), the scheme of Fig. 18 shows that in order to train the networks ASNN1 the error (ε_2 in the figure) is backpropagated through the ASNN2 (i.e. the neural loudspeaker model).

It is important to underline that this scheme can be used to re-train the ASNNs in the place where the loudspeaker is positioned, taking into account the deviations of the loudspeaker characteristics due to age and other environmental factors.

Fig. 20 shows the harmonic distortion at 200Hz with and without predistorter.

3.4. Physical-like NN sound synthesizer

The sound synthesis by physical or physical-like model seems to be one of the best way to produce interesting and high quality sounds. The physical model paradigms are generally based on the subdivision of the synthesizer in a nonlinear excitation part in connection with other linear parts as delay lines and/or filters [53,49,5]. The most famous model-based technique is the so-called digital waveguide filter [53]. The basic idea of this approach is to simulate the vibration-transmitting component of an acoustic

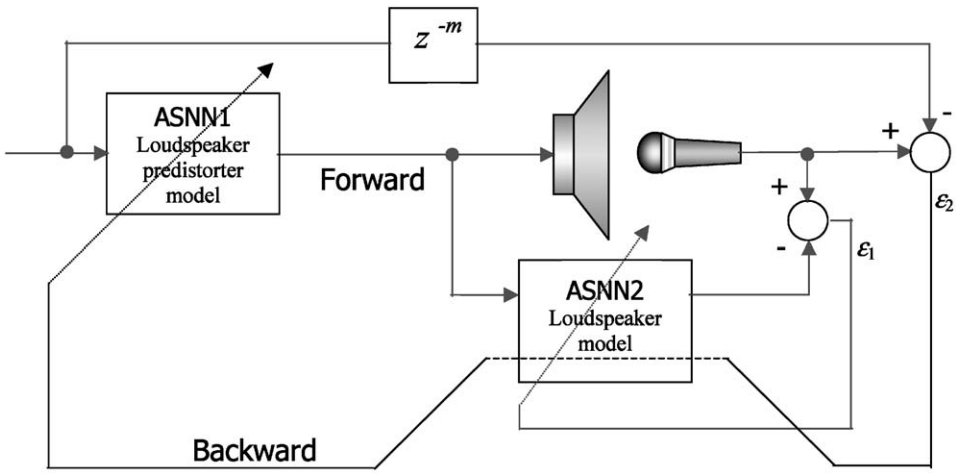

Fig. 18. ASNN predistorter learning scheme. The ASNN2 learns the loudspeaker model that is used for error back-propagation needed for the learning of ASNN1 network. Both ASNNs have 20 inputs (10 MA and 10 AR) 12 spline hidden neurons and a linear neuron output.

musical instrument such as the membrane of a drum, a string of a stringed instrument, and a bore of a woodwind instrument. One of the main problems with model-based synthesis techniques is the determination of the synthesizer parameters.

Usually, a spectrum analysis of the original signal is necessary in order to correctly design the filters, and many simplifications are made in order to describe the nonlinear excitation mechanism (NLEM). The NLEM, in fact, is a very important characteristic of the timber of the instrument.

In the last years, several attempts have been made for modeling the NLEM. As an example, in [52], Smith proposed classical identification techniques for the violin model parameters estimation. More recently, Drioli and Rocchesso in [17], proposed an interesting learning-based approach for pseudo-physical model for sound synthesis. They proposed the use of Radial Basis Function universal approximation scheme in order to off-line learns the static or dynamic curve of NLEM.

In this section a new recurrent-network-based synthesis model for single reed NLEM is proposed. Although the idea of using neural networks for sound synthesis is not new (see, for example, [17] and the reference therein) our work addresses a new particularly efficient scheme.

In the proposed approach the structure of the network is designed on the basis of a physical model of nonlinear excitation of the single-reed woodwind instrument. In general the NLEM, as in the vibrating reed, is a nonlinear system with memory so, static networks cannot adequately model this system. In order to take into account this nonlinear dynamic, IIR/FIR-MLP (TDNN) [7] with flexible activation function are used [62]. Moreover, in order to obtain an efficient hardware/software implementation, the synaptic weights are constrained to be a power-of-two terms while the nonlinear

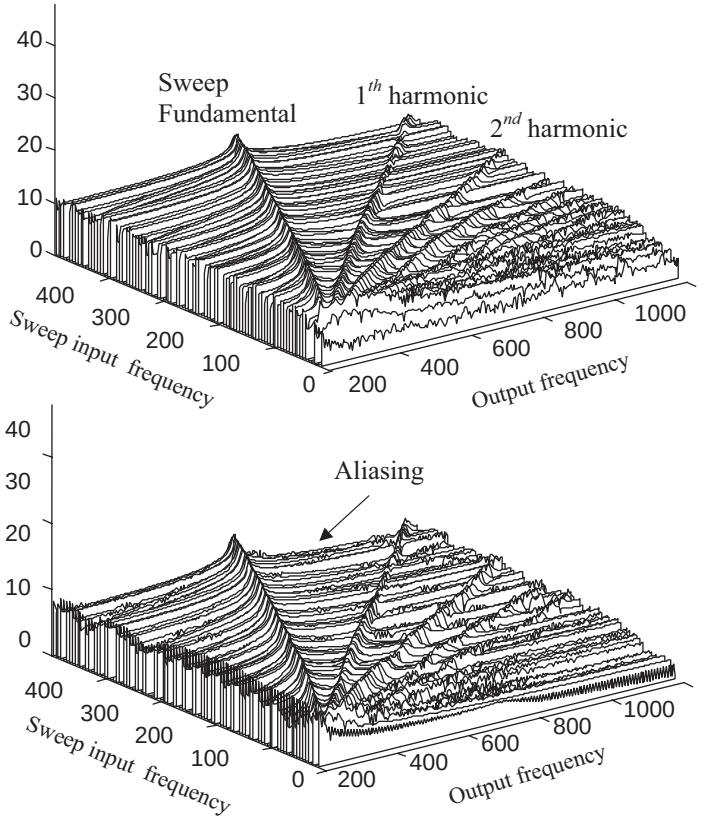

Fig. 19. Response of sweep signal of the woofer SIPE-AS300 (up) and response of the neural network loudspeaker model to the same signal (down).

function can be implemented as a simple spline interpolation scheme or through a lookup table.

Due to power-of-two weights constraints and difficult derivative computation (for some particular synthesis scheme) standard or time-delay back-propagation learning algorithm cannot be developed. Therefore the learning phase has been carried out by an efficient combinatorial optimization algorithm called Tabu Search (TS), firstly proposed by Glover and Laguna [23] and recently used for power-of-two adaptive filter [56]. Moreover, in order to demonstrate the effectiveness of the proposed model, experiments on a single-reed woodwind instruments have been carried out.

The single-reed and mouthpiece arrangements act as a pressure-controlled valve, which transmits energy into the instrument for the initialization and maintenance of oscillations in the acoustic resonator. In woodwind instruments the reed generator is generally driven into a highly nonlinear part of its characteristic and the reed can be modeled as a damped nonlinear oscillator so that the motion of a second-order

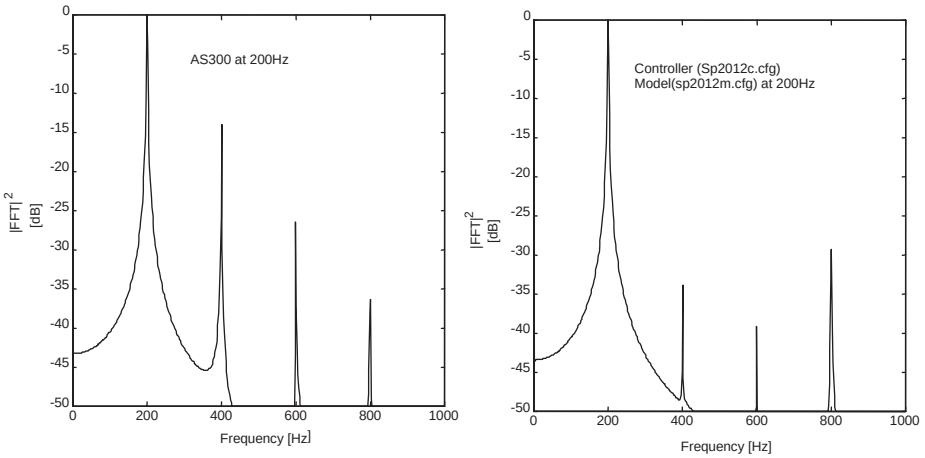

Fig. 20. Harmonic distortion of the woofer with (right) and without (left) neural predistorter. About 20dB of 2nd and 3rd harmonic attenuation at 200 Hz.

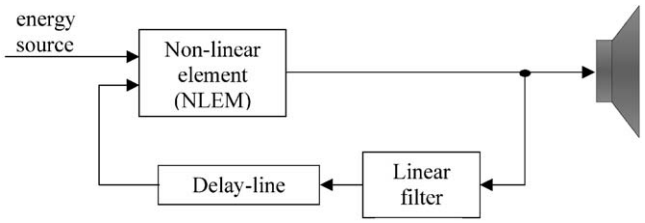

Fig. 21. General scheme of physical model synthesizer.

mass-spring system is given by:

$$m_r\left[\frac{\mathrm{d}^2x}{\mathrm{d}t^2}+\mu\omega_r\frac{\mathrm{d}x}{\mathrm{d}t}+\omega_r^2(x-x_0)\right]=g(p_\Delta(t),U(t)), \tag{30}$$

where m_r is the equivalent reed mass, μ is the damping factor, ω_r is fundamental reed frequency, $p_\Delta(t)$ is the difference between the player's oral cavity and the pressure in the reed channel $p_\Delta=(p_{oc}-p_r)$ and $U(t)$ is the steady volume flow through the reed and $g(.)$ is an hard nonlinear function [49,20]. The reed-excitation mechanism [20] is a nonlinear system with memory and its parameters estimation can be very difficult.

A general model for a woodwind instrument is reported in Fig. 21. It is composed of a delay line, a linear element (filter) and a NLEM. In our approach, in order to define a more general nonlinear model, we make use of neural networks with FIR-IIR synapses and activation functions implemented through an adaptive CR-spline interpolated curve.

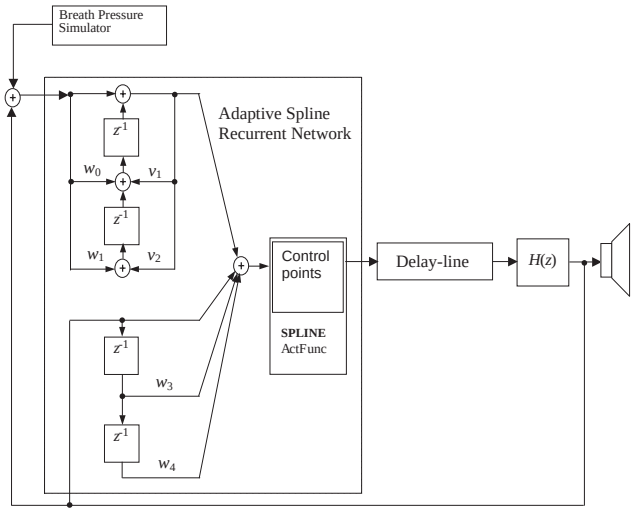

Fig. 22. Proposed scheme of neural physical model synthesizer where the NLEM is here implemented with a spline-TDNN.

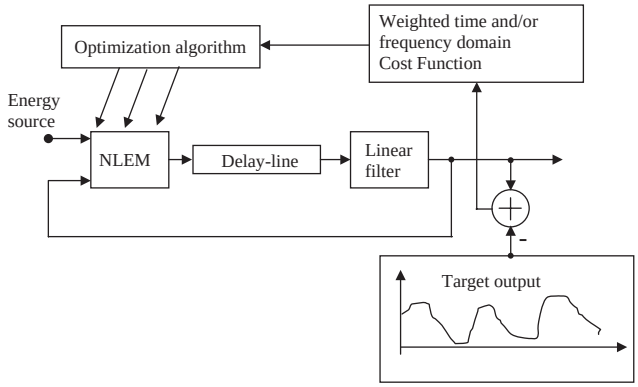

Fig. 23. Learning scheme of the neural physical model synthesizer.

Hence the parameters we optimize are the filter's weights and the control points of the CR-spline activation curve.

FAF greatly reduces the structural complexity to approximate the NLEM. So the synthesis network (here called Adaptive Spline Recurrent Network (ASRN)) constructed on the basis of the previously described physical model of a single reed instrument is shown in Fig. 22. Once the parameters are fixed with the learning algorithm (see Fig. 23), the instrument works as a typical physical model instrument.

In order to obtain a computationally efficient synthesizer, the ASRN makes use of IIR-FIR synapse with power-of-two (or a sum of power-of-two) coefficients. This

represents a great advantage in the case of hardware realization. Multipliers in fact can be built by using few simple and fast shift registers instead of slower floating-point arithmetic, such a strategy can reduce both the VLSI silicon area and the computational time. Moreover, as specified in [26], the activation function can be easily and efficiently implemented both in hardware and in software or after the learning, simply realized through a lookup-table.

We have tested this model with clarinet and saxophone sounds. The learning phase consisted of 1000 TS cycles for the clarinet, having taken into account a cost function over a window of 1024 samples. This cost function depends on a weighted difference between the real instrument sound and the generated one. In particular, the cost function J_{cost} minimized by the previous described learning algorithm is defined as

$$J_{\cos t} = w[t]J_1 + (1 - w[t])J_2, \tag{31}$$

where J_1 and J_2 are defined as

$$J_1 = \frac{1}{N}\sum_{t=0}^{N}(d[t] - x[t])^2 \tag{32}$$

$$J_2 = \sum_{\omega\in\Omega}(|D(e^{j\omega})| - |X(e^{j\omega})|)^2, \tag{33}$$

where $d[t]$, $x[t]$ and $D(e^{j\omega})$, $X(e^{j\omega})$; represent the desired and actual sound signals in time and frequency domain, respectively. The term $w[t]$ represents a weighting function designed in order to take into account the initial instrument transient: $w[t]$ can assume an hyperbolic shape (e.g. $w[t]=c_0/(t+c_1)$ where c_0 and c_1 are suitable constant terms). The time-domain approach has, in fact, the advantage that it is able to treat quite simply the starting transients of the sound. This is important, since transients make a very large contribution to the individuality of musical sounds. The frequency-domain approach, in contrast, works best for steady sounds (e.g. see [20]).

We have tested the model with several FIR-IIR delay line lengths. We obtained good results for clarinet and acceptable results for saxophone-like sound. Currently we are testing new solutions to improve saxophone model, such as to use adaptive all-pass filters as termination of the delay-line.

Our goal is not to exactly reproduce the target sound, that is impossible without excessively complicating the model, but to make the instruments to learn the parameters, so that they can reproduce different types of sound. In this case the same model can be used to reproduce a class of different instrument characteristics. In fact, once fixed all the parameters for a single instrument by minimizing the cost function (learning stage) the instrument can be played (forward stage) as a normal physical-model instrument.

4. Conclusions

A review of some neural architectures for real-time DSP and some audio applications have been described and discussed. Nonlinear signal processing represents in fact a

central issue for a lot of applications of intelligent signal processing (that is a key tools for many emerging multimedia technologies).

In particular flexible activation functions, IIR-MLP, multirate NNs have been studied in order to reduce the structural-computational complexity and make possible real-time low cost NN audio applications.

The described experiments demonstrate that NNs can be considered as a well established methodology for audio or more general nonlinear signal processing applications.

References

[1] S. Amari, A universal theorem on learning curves, Neural Networks 6 (2) (1993) 161–166.
[2] A.D. Back, A.C. Tsoi, FIR and IIR synapses, a new neural network architecture for time series modeling, Neural Comput. 3 (1991) 375–385.
[3] N. Benvenuto, F. Piazza, A. Uncini, A neural network approach to data predistorter with memory in digital radio system, Proceedings of ICC '93: International Communications Conference, Geneva, Switzerland, May 1993.
[4] A. Bernardini, S.D. Fina, A new predistortion technique using neural nets, Signal Process. 34 (1993) 231–243.
[5] G. Borin, G De Poli, A. Sarti, Sound Synthesis by dynamic systems interaction, in: D. Baggi (Ed.), Readings in Computer-Generated Music, IEEE Comp. Soc. Press, Silver Spring, MD, 1992, pp. 139–160.
[6] P. Campolucci, A. Uncini, F. Piazza, A signal-flow-graph approach to on-line gradient calculation, 2000 Massachusetts Institute of Technology Neural Computation, Vol. 12, August 2000, pp. 1901–1927.
[7] P. Campolucci, A. Uncini, F. Piazza, B.D. Rao, On-line learning algorithms for locally recurrent neural networks, IEEE Trans. Neural Network 10 (2) (1999) 253–271.
[8] E. Catmull, R. Rom, A class of local interpolating splines, in: R.E. Barnhill, R.F. Riesenfeld (Eds.), Computer Aided Geometric Design, Academic Press, New York, 1974, pp. 317–326.
[9] T. Chen, H. Chen, Approximation of continuous functionals by neural networks with application to dynamic systems, IEEE Trans. Neural Networks 4 (6) (1993) 910–918.
[10] T. Chen, H. Chen, R. Liu, Approximation capability in C(Rn) by multilayer feedforward networks and related problems, IEEE Trans. Neural Networks 6 (1) (1995) 25–30.
[11] A. Cichocki, R. Unbehauen, Neural Networks for Optimization and Signal Processing, Wiley, B.G. Teubner, Stuttgart, 1993.
[12] G. Cocchi, A. Uncini, Subband neural networks prediction for on-line audio signal recovery, IEEE Trans. Neural Network 13 (4) (2002) 867–876.
[13] R.E. Crochiere, L.R. Rabiner, Multirate Digital Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, 1983.
[14] G. Cybenko, Approximation by superposition of a sigmoidal function, in: Mathematical Control Signals Systems, Vol. 2, Springer, New York, 1989.
[15] A. Czyzewsky, Artificial intelligence-based processing of old audio recordings, A.E.S. Reprint 3885 (F-6), San Francisco 1994.
[16] B. De Vries, J.C. Principe, The gamma model—A new neural model for temporal processing, Neural Networks 5 (1992) 565–576.
[17] C. Drioli, D. Rocchesso, Learning pseudo-physical models for sound synthesis trasformation, IEEE International Conference on Systems, Man, and Cybernet. 2 (1998) 1085–1090.
[18] N.J. Fleige, Multirate Digital Signal Processing (Multirate systems, Filter Banks, Wavelet), Wiley, New York, 1994.
[19] L.A. Feldkamp, G.V. Puskorius, A signal processing framework based on dynamic neural networks with application to problems in adaptation, filtering, and classification, Proc. IEEE 86 (11) (1998) 2259–2277.
[20] N.H. Fletcher, T.D. Rossing, The Physics of Musical Instruments, Springer, New York, 1981.

[21] P. Frasconi, M. Gori, G. Soda, Local feedback multilayered networks, Neural Comput. 4 (1992) 120–130.
[22] A. Gilloire, M. Vetterli, Adaptive filtering in subbands with critical sampling: analysis, experiments, and application to acoustic echo cancellation, IEEE Trans. Signal Process. 40 (8) (1992) 1862–1875.
[23] F. Glover, M. Laguna, Tabu Search, Kluver Academic Publisher, Dordrecht, 1997.
[24] S.J. Godsill, P.J.W. Rayner, Digital Audio Restoration, Springer, Berlin, 1998.
[25] M. Gori, Y. Bengio, R. De Mori, BPS: A learning algorithm for capturing the dynamic nature of speech, in Proceedings of International Joint Conference on Neural Networks, Washington, DC, Vol. II, 1989 pp. 417–423.
[26] S. Guarnieri, F. Piazza, A. Uncini, Multilayer feedforward networks with adaptive spline activation function, IEEE Trans. Neural Network 10 (3) (1999) 672–683.
[27] S. Haykin, Neural Networks Expand SP's Horizons, IEEE Signal Process. Mag. 13 (2) (1996) 24–49.
[28] S. Haykin, Adaptive Filter Theory, 3th Edition, Prentice-Hall, Englewood Cliffs, NJ, 1996.
[29] S. Haykin, Neural Networks (A comprehensive Foundation), 2nd Edition, Prentice-Hall, Englewood Cliffs, NJ, 1999.
[30] K. Hornik, M. Stinchombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (1989) 359–366.
[31] A.J.M. Kaizer, Modeling of the nonlinear response of an electrodynamic loudspeaker by a Volterra series expansion, J. Audio Eng. Soc. 35 (6) (1987) 421–433.
[32] S. Kalluri, G.A. Arce, A general class of nonlinear normalized adaptive filtering algorithms, IEEE Trans. Signal Process. 47 (8) (1999).
[33] G. Karam, H. Sari, Analysis of predistortion, equalization, and ISI cancellation techniques in digital radio systems with non linear transmit amplifiers, IEEE Trans. Commun. COM-37 (1989) 1245–1253.
[34] G. Karam, H. Sari, Data predistortion technique with memory for QAM radio systems, IEEE Trans. Commun. COM-39 (1991) 336–343.
[35] W. Klippel, Dynamic measurement and interpretation of the nonlinear parameters of electrodynamic loudspeakers, J. Audio Eng. Soc. 38 (12) (1990) 944–955.
[36] W. Klippel, Nonlinear large-signal behavior of electrodynamic loudspeakers at low frequencies, J. Audio Eng. Soc. 40 (1992) 483–496.
[37] S.Y. Kung, J.N. Hwang, Neural networks for intelligent multimedia processing, Proc. IEEE 86 (6) (1998) 1244–1272.
[38] K.J. Lang, G.E. Hinton, The development of the time-delay neural networks architecture for speech recognition", Technical Report CMU-CS-88-152, Carnegie Mellon University Pittsburgh, PA.
[39] R.R. Leighton, B.C. Conrath, The autoregressive backpropagation algorithm in: Proceedings of the International Joint Conference on Neural Networks, Seattle, WA, 1991, pp. 369–377.
[40] R.C. Maher, A method for extrapolation of missing digital audio data, J. Audio Eng. Soc. 42 (5) (1994) 350–357.
[41] V.J. Mathews, G.L. Sicuranza, Polynomial Signal Processing, Wiley Publishers, New York, ISBN, 0-471-03414-2, 2000.
[42] M.C. Mozer, A focused backpropagation algorithm for temporal pattern recognition, University of Toronto, Technical Report CRG-TR-88-3, Canada, 1988; Complex Systems, 3, 1989, 349–381.
[43] K.S. Narendra, K. Parthasarathy, Identification and control of dynamical systems containing neural networks, IEEE Trans. Neural Networks 1 (1990) 4–27.
[44] T.Q. Nguyen, Near-perfect reconstruction pseudo-QMF banks, IEEE Trans. Signal Process. 42 (1) (1994) 65–76.
[45] Y.H. Pao, Adaptive Pattern Recognition and Neural Networks, Addison-Wesley, Reading, MA, 1989.
[46] M.R. Petraglia, S.K. Mitra, Performance analysis of adaptive filter structures based on subband decomposition, Proceedings of the IEEE International Symposium on Circuit and Systems, Chicago, IL, May 1993, pp. 60–63.
[47] J.C. Principe, A. Rathie, J.M. Kuo, Prediction of chaotic time series with neural networks and the issue of dynamic modeling, Int. J. Bifurcation Chaos 2 (4) (1992) 989–996.

[48] P.A. Regalia, Adaptive IIR Filtering in Signal Processing and Control, Marcel Dekker Inc., New York, 1995.

[49] G.P. Scavone, An acoustical analysis of single-reed woodwind instruments with an emphasis on design and performance issues and digital waveguide modeling techniques Ph.D. Thesis, Music Department, Stanford University, March 1997.

[50] M. Schetzen, Nonlinear system modeling based on the Wiener theory, Proc. IEEE 69 (12) (1981) 1557–1573.

[51] J. Schmidhuber, Learning complex, extended sequences using the principle of history compression, Neural Comput. 4 (2) (1992) 234–242.

[52] J.O. Smith, Technique for Digital Filter Design and System Identification with Application to the Violin, Ph.D. Thesis, CCRM, Standford University, Report N. STAN-M-14, 1983.

[53] J.O. Smith, Physical modeling using digital wave-guides, Comput. music J. 16 (4) (1992) 74–91.

[54] M. Solazzi, F. Piazza, A. Uncini, An adaptive spline nonlinear function for blind signal processing, Proceedings of the Workshop on Neural Networks for Signal Processing, Vol. X, December 2000, pp. 396–404.

[55] M. Solazzi, A. Uncini, Artificial neural network with adaptive multidimensional spline activation functions, IEEE-INNS-ENNS International Joint Conference on Neural Networks IJCNN2000, Como, Italy, 24–27 July 2000.

[56] S. Traferro, A. Uncini, Power-of-two adaptive filters using tabu search, IEEE Trans. Circ. Syst.—II: Analog Digital Signal Process. 47 (6) (2000) 566–569.

[57] A. Uncini, F. Gobbi, F. Piazza, Frequency recovery of narrow-band speech using adaptive spline neural networks, Proceedings of IEEE International Conference on Acoustic Speech and Signal Processing, ICASSP'99, Phoenix AR, Vol. 2, March 15–19, 1999 pp. 997–1000.

[58] A. Uncini, L. Vecci, P. Campolucci, F. Piazza, Complex-valued neural networks with adaptive spline activation function for digital radio links nonlinear equalization, IEEE Trans. Signal Process. 47 (2) (1999).

[59] M. Unser, A Perfect Fit for Signal and Image Processing, IEEE Signal Process. Mag. (1999) 22–38.

[60] S.V. Vaseghi, Advanced Signal Processing and Digital Noise Reduction, Wiley, New York, Teubner, Stuttlgorte, 1996.

[61] S.V. Vaseghi, R. Franyling-Cork, Restoration of old gramophone recordings, J. Audio Eng. Soc. 40 (10) (1992).

[62] L. Vecci, F. Piazza, A. Uncini, Learning and Approximation Capabilities of Adaptive Spline Activation Function Neural Networks, Neural Networks 11 (2) (1998) 259–270.

[63] S.V. Vaseghi, P.J.W. Rayner, Detection and suppression of impulsive noise in speech communication system, IEE Proc. 1371 (1) (1990) 38–46.

[64] V. Volterra, Sopra le Funzioni che Dipendono de altre Funzioni, Rend. R. Accad. Lincei pp. 97–105, 141–146, 153–158 2 Sem., 1887.

[65] A.T. Waibel, T. Hanazawa, G.E. Hinton, K. Shikano, K.J. Lang, Phoneme recognition using time-delay neural networks, IEEE Trans. Acoustics, Speech and Signal Processing 37 (3) (1989) 328–339.

[66] E.A. Wan, Temporal backpropagation fir FIR neural networks, in Proceedings of the International Joint Conference on Neural Network, Vol. 1, 1990, pp. 575–580.

[67] B. Widrow, M. Lehr, 30 years of adaptive neural networks: perceptron, adaline and backpropagation, Proc. IEEE 78 (9) (1990).

[68] R.J. Williams, J. Peng, An efficient gradient-based algorithm for on-line training of recurrent network trajectories, Neural Comput. 2 (1990) 490–501.

[69] R.J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, Neural Comput. 1 (1989) 270–280.

[70] Y.G. Yang, N.I. Cho, S.U. Lee, On the performance analysis and applications of the subband adaptive digital filter, Signal Process., 1994.

[71] H. Yasukawa, Signal restoration of broad band speech using nonlinear processing, Proceedings of EUSIPCO'96, Trieste, Italy, Sept. 1996.

Aurelio Uncini, was born in Cupra Montana, Italy, in 1958. He received the laurea degree in electronic Engineering from the University of Ancona, Italy, on 1983 and the Ph.D. degree in Electrical Engineering in 1994 from University of Bologna, Italy.

From 1984 to 1986 he was with the "Ugo Bordoni" Foundation, Rome, Italy, engaged in research on digital processing of speech signals. From 1986 to 1987 he was at Italian Ministry of Communication. From 1987 to 1993 he has been a free researcher affiliated at the Department of Electronics and Automatics—University of Ancona and where from 1994 to 1998 he was assistant professor. Since November 1998, he is Associate Professor at the INFOCOM Department—University of Rome "La Sapienza" where he is teaching Circuits Theory. He his author of more than 100 papers in the field of circuits theory, optimisation algorithms for circuits design, neural networks and signal processing. His present research interests include also adaptive filters, audio processing, neural networks for signal processing, blind signal processing. Prof. Uncini is a member of the IEEE Neural Networks for Signal Processing Technical Committee, of the Associazione Electronicnica ed Elettronica Italiana (AEI), of the International Neural Networks Society (INNS) and the Societá Italiana Reti Neuroniche (SIREN).