Multilayer Spline Neural Networks for Speech Denoising in Frequency Domain

GIOVANNI COSTANTINI, MASSIMO CAROTA
Department of Electronics Engineering
University of Rome “Tor Vergata”
Via del Politecnico 1, 00133 Rome
ITALY

Abstract: - The speech denoising Neural Network architecture we propose in this paper is based on Adaptive Spline Neural Network (ASNN). It is an architecture for real-time oriented applications, due to its low size complexity and high parallelism. Results show improvements in Signal to Noise Ratio (SNR) and better performances in comparison with classical denoising neural networks.

Key-Words: - Speech Enhancement, Noise Reduction, Adaptive Filters

1 Introduction

Today sophisticated communication and recording systems require better performances in speech denoising. Most of all, we need to improve noisy speech intelligibility in man to man or man-machine communication. The presence of noise dramatically reduces performances of modern automatic speech recognition systems operating in noisy environments.
The superimposed noise can be impulsive, continuous, correlated or uncorrelated. Obviously, a general solution to the problem of speech denoising is not available; a specific noise-oriented solution is needed. In this paper we focus our attention to continuous uncorrelated disturbs (background noise).
The classical Spectral Subtractive approach to the problem is still one of the most employed one; its main drawback is the residual “musical noise”, which causes metallic voice distortion [1]. In recent years, neural networks were successfully applied in audio signal processing. However, the huge size of implemented structures and the low generalization capabilities have discouraged their use.
We aim to obtain low size real-time oriented neural architectures using Adaptive Spline Neural Network (ASNN) to improve SNR in background-noise corrupted speech signals. Comparison is made with classical Multi Layer Perceptron (MLP) neural networks used in denoising applications [2,3].

2 Spline Networks

In classical multilayer feed-forward neural networks, each neuron is characterized by a fixed non-liner activation function, such as $a\left(1-e^{-bx}\right)/\left(1+e^{-bx}\right)$.
To reduce the network size, at the cost of an acceptably grater neuron complexity, it is possible to define adaptive activation functions. The simplest way is to use polynomial functions [4]. The main drawback of this solution regards the adaptation of coefficients in the learning phase, due to spurious minima and maxima. In addition, a polynomial is a non-bounded function, resulting in a generally poorly smooth approximation.
Later, to solve the above problems, adaptive spline activation function was introduced [5-7]. Each neuron is characterized by a different activation function, whose shape can be modified through some control points. It has been proved that such a neuron improves flexibility, approximation and generalization capabilities of the network.
In this work, we implement activation functions through cubic spline interpolation of control points. In every training step, we update both activation function shape and input weight values. In the following, we introduce spline network theory [5,7].

2.1 Spline curve

A planar spline curve is a two-dimensional array, whose components are piecewise polynomial, univariate functions of the same degree. Its mathematical formulation ensures both continuity and existence of derivatives, along the curve and in correspondence to the joining points between the curve spans. We define the curve as follows:

$$F(u)=\left[F_x(u) \quad F_y(u)\right]=\mathop{\mathbf{C}}_{i=0}^{N-3} F_i(u) \qquad (1)$$

where $\mathbf{C}$ is the concatenation operator and $F_i(u)$ the i-th curve span. The indices of the C operator in (1) are valid only for cubic polynomials that are good tradeoffs between the requested properties and computational complexity.
The domain of parameter u is $0 \le u \le 1$ for every curve span; it has the property of being *local*. So, we can evaluate, from the abscissa global parameter, the local parameter u, as well as the curve span i, thru a unique mapping. In this way, we can match any point on the spline curve $F(u)$ with a point on the i-th $F_i(u)$ curve span, that can be described as follows (see [5]):

$$F_i(u) = \begin{bmatrix} F_{xi}(u) & F_{yi}(u) \end{bmatrix}^T = \sum_{j=0}^{3} Q_{i+j} C_j(u) \qquad (2)$$

where Q_i are the *control points* $\begin{bmatrix} q_{x,i} & q_{y,i} \end{bmatrix}^T$, with constraints $q_{x,0} < q_{x,1} < \ldots < q_{x,N}$, and $C_j(u)$ are the spline polynomials:

$$\begin{array}{ll} C_0(u) = \frac{1}{2}\left(-u^3 + 2u^2 - u\right) & C_2(u) = \frac{1}{2}\left(-3u^3 + 4u^2 + u\right) \\ C_1(u) = \frac{1}{2}\left(3u^3 - 5u^2 + 2\right) & C_3(u) = \frac{1}{2}\left(u^3 - u^2\right) \end{array} \qquad (3)$$

2.2 The SG (Sigmoid Generalized) neuron

Once the weighted sum s_k^l is computed, a correspondence in the parametric curve has to be find (that is to determine the span of the curve); then the value must be mapped through the curve. In Figure 1, the *k*-th neuron SG in the *l*-th layer of the network is shown:

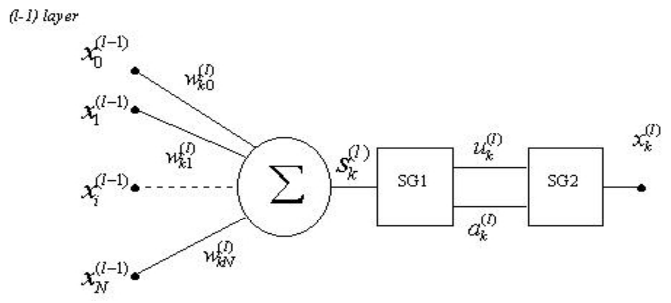


Figure 1: Representation of SG neuron.

Block SG1 represents the inversion of the *x-axis* component of the parametric spline function. By choosing uniform distribution of samples along *x-axis*:

$$F_{xi}(u) = u\Delta x + q_{xj+1} \qquad (4)$$

the inversion of *x-axis* becomes trivial. We assume:

$$z_k^{(l)} = \frac{s_k^{(l)}}{\Delta x} + \frac{n-2}{2} \quad a_k^{(l)} = \left\lfloor z_k^{(l)} \right\rfloor \quad u_k^{(l)} = z_k^{(l)} - \left\lfloor z_k^{(l)} \right\rfloor \qquad (5)$$

where $\lfloor \bullet \rfloor$ is the *floor* operator, *n* is the number of control points, $a_k^{(l)}$ and $u_k^{(l)}$ are respectively the index of the considered span and its internal parameter *u* (*a=0* stands for the first span of the curve). All the control points $\{q_{y,0}, \ldots, q_{y,n-1}\}$ are saved in a Look Up Table (LUT) and are shown in Figure 2.

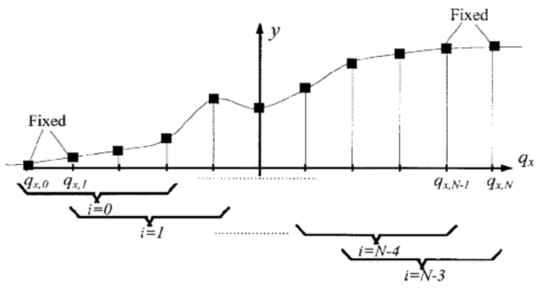


Figure 2: control points with a fixed step Δx.

Block SG2 represents Eq. (2) that can be rewritten as:

$$F_i(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} Q_i \\ Q_{i+1} \\ Q_{i+2} \\ Q_{i+3} \end{bmatrix} \qquad (6)$$

for Catmull-Rom spline, where $F_i(u)$ is the span *a* evaluated in block SG1 with Eq. (5).

3 Network Architecture

The chosen architecture is a Multilayer Spline Neural Network: three layers of N spline neurons each (N:N:N network). It operates on samples in the frequency domain (noisy samples) and outputs samples in the same domain (clean samples).

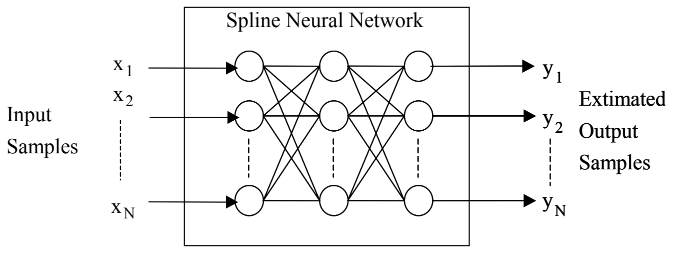


Figure 3: the frequency domain network.

The input frequency samples are obtained by evaluating the Short Time Fourier Transform (STFT) of the input signal to be filtered. We isolate N samples of the noisy signal, by a Hamming windowing operation. We first tried to exploit the speed and efficiency of the FFT algorithm, obtaining N/2 frequency samples of Spectrum Phase and as many of Spectrum Modulus. It is well known that FFT gives a constant resolution in frequency sampling, equal to the sampling rate divided by the window size in samples: so, for the speech signal, we have too little information at low frequencies and too much at the high ones.

One possible solution to this problem is the well-known constant Q Frequency Transform (QFT) described in [8]. This solution exhibits two unacceptable drawbacks: first, the evaluation of the complex values of the spectral components is too much time consuming; second, the so called IQFT (inverse transform) doesn't give back a reconstructed signal of acceptable quality.

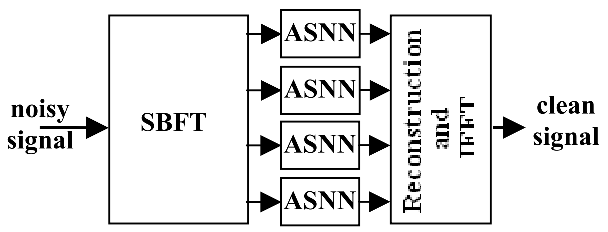


Figure 4: the denoising sistem.

Our proposal is to preserve the variable resolution (high at the low frequencies and low at the high ones), by evaluating FFT on "sub-bands" of constant resolution. We call it Sub-Band Fourier Transform (SBFT, see Figure 4).
We evaluate the modulus of the FFT of a windowed chunk of samples on four adjacent frequency segments ("sub-bands"). At the output, we rejoin the sub-bands, in order to obtain the Modulus samples of the estimated clean signal. In the reconstruction step, we maintain the frequency samples of the noisy signal spectrum Phase, thanks to the insensibility of the human ear to phase information.

3.1 The SBFT Algorithm

Chunks of M (M=256) windowed time samples are isolated and their FFT is evaluated. This algorithm is characterized by a special window function, obtained extending one 32 frequency sample long Hamming window with its last sample up to a length of 256 samples, as depicted in Figure 5.

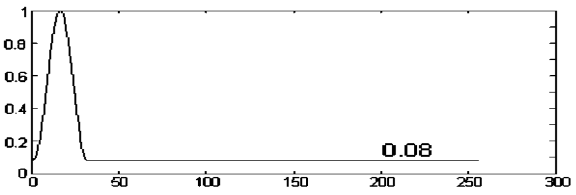
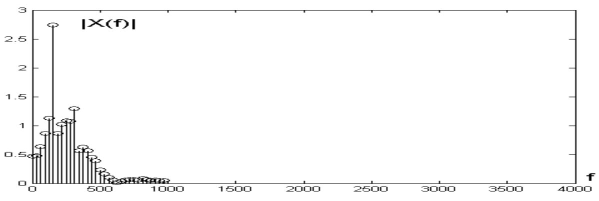


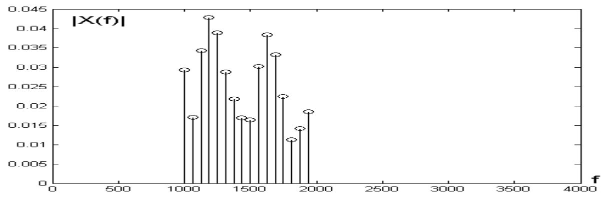
Figure 5: the window function.

The algorithm is as follows:

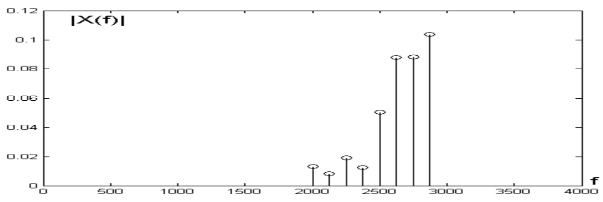
1) FFT of the current chunk of 256 samples is evaluated and its first 32 values of the Modulus are extracted; these values exactly span from 0 Hz to 1000Hz



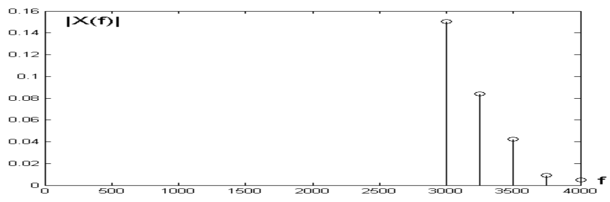
2) same as above, but for 128 samples, and values from 17-th to 32-th are extracted (1 KHz - 2 KHz)



3) same as above, but for 64 samples, and values from 17-th to 24-th are extracted (2 KHz - 3 KHz)



4) same as above, but for 32 samples, and values from 13-th to 17-th are extracted (3 KHz - 4 KHz)



The computations of these four FFTs are independent, so they can be run in parallel.
The 32 values from the first FFT are the input pattern of a 32:32:32 ASNN. The 16 values from the second FFT are the input pattern of a 16:16:16 ASNN. The 8 values from the third FFT are the input pattern of an 8:8:8 ASNN. The 5 values from the fourth FFT (the 5-th value is the zero frequency component) are the input pattern of a 5:5:5 ASNN. The operations performed by the four neural networks are independent, too; therefore, they can operate in parallel on their own frequency sub-band.
In order to obtain the time samples of the estimated clean signal, the denoising operation ends with the IFFT of the estimated Modulus, in couple with the noisy Phase.

4 Network Training

To build up a proper training set, we collect some speech signals, considered clean, and sum them to as many noise signals, provided they are uncorrelated with respect to the clean ones. The noise signals are:

1) noise from the real world (noise produced by the fan of power supply)
2) white noise (computer generated)
3) pink noise (computer generated)

The artificial noisy signals represent the input patterns (Training Set, i.e. training signals), while their clean versions are the desired outputs.

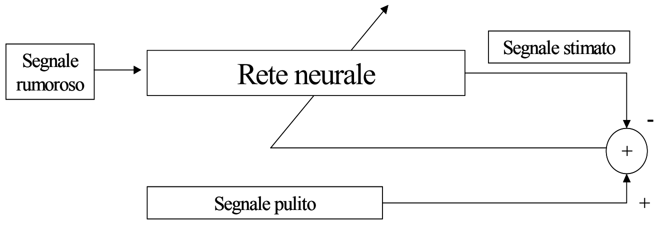


Figure 6: the adaptive learning process.

The learning algorithm is based on the classical back-propagation rule: at every step, clean signals estimated by the ASNNs are compared with the desired clean signals, in order to evaluate the Mean Square Error, as shown in Figure 6. Connection weights and control points are then updated by minimizing this error. The updating step can be done in parallel inside each sub-network. The first training cycle ends when input signal end is reached. The training epoch ends when no relevant changes can be appreciated.

5 Simulation Results

The working sampling frequency was $F_s = 8$ KHz. Different speech samples from different speakers with different utterance were employed. Noisy signals at different SNR were processed. The developed architectures shows better SNR improvements in comparison with classical perceptron neural approach [2,3].

In a first test session, the neural network was tested by comparing its behavior with respect to the training signals and to other signals not employed in training process (Test Set). In both cases, we estimated the improvement of SNR in dB and the quality of the output signal. In the following table, test results have been reported, where superimposed noises are: real noise, pink noise and white noise. At all events, the starting SNR was approximately 6 dB, where, signal 1 is the training signal, signal 2 is the same speaker pronouncing phrases different from those of the training step, signal 3 is the speaker of opposite sex.

	signal 1	signal 2	signal 3
real	17.6	17	15.6
white	15.5	14.7	15.1
pink	15.3	15.2	14

Table 1

In a second test session, the neural network was compared with respect to classical perceptron multilayer neural networks (MLP) [2,3]. An SNR improvement of at most 8 dB is achieved on Training Set (starting SNR = 8.7 dB), as shown in Figure 7.

The proposed architecture requires a more complex training procedure, because of the overload introduced by the adaptation of the spline activation function. Nevertheless, in the operating phase, the spline neuron exhibits the same complexity as the classical neuron, at remarkable lower network dimension and topology. In fact, classical Perceptron Neural Networks do employ a higher number of neurons (up to 240 in 60:60:60:60 structure), with absolutely worst performances (see Figure 7).

On the contrary, the proposed architecture requires only 32+16+8+5=61 neurons per layer (183 as a whole), with better performances (SNR improvement of 17.6 dB).

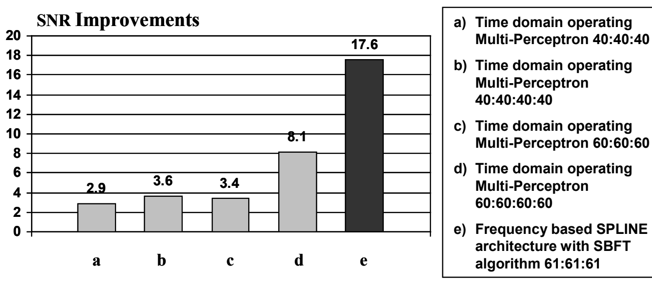


Figure 7: SNR improvement.

6 Conclusion

A new neural architecture for speech signal denoising was developed. An adaptive spline neuron model was employed and a frequency based approach was explored. Results show better performances compared to classical MLP neural networks, with SNR improvements up to 17.6 dB. The developed architecture is real-time application oriented because of its low complexity and its very high degree of parallelism.

References:

[1] S. F. Boll, *Suppression of Acoustic Noise in Speech Using Spectral Subtraction', IEEE Trans. Acoustic. Speech and Signal Preocessing, vol ASSP-27, no.2, pp. 113-120, 1979.*

[2] T. T. Le, *Speech Enhancement Using Non-Linear Prediction', TENCON'93. Proceedings. Computer, Communication, Control and Power Engineering 1993, vol.3, pp. 306-309.*

[3] S. Tamura, A. Waibel, *Noise Reduction Using Connectionist Models', Acoustics, Speech and Signal Processing 1988. ICASSP-88, vol.1, pp. 553-556.*

[4] G. P. Scavone, P. R. Cook, *Combined linear and non-linear prediction in calibrating models of musical instruments to recordings', Proceedings of International Computer Music Conference, ICMC'94, pp. 433-434.*

[5] S. Guarnieri, F. Piazza, A. Uncini, *Multilayer Feedforward Networks with Adaptive Spline Activation Function', IEEE Transaction on Neural Networks, Vol.10, no.3, May 1999.*

[6] 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. on Signal Processing, Vol. 47, No. 2, February 1999.*

[7] L. Vecci, F. Piazza, A. Uncini, *Learning and Approximation Capabilities of Adaptive Spline Activation Function Neural Networks', Neural Networks, Vol. 11, No. 2, March 1998.*

[8] Brown J.C., *Calculation of a constant Q spectral transform*. J. Acoust. Soc. Am., vol. 89, January, 1991, pages 425-434