DESIGN OF MULTI-LAYER NEURAL NETWORKS FOR BUTTERWORTH FILTER OPTIMIZATION

Assist. Prof. Dr. Hanan A. R. Akkar
Department of Electrical & Electronic Engineering
University of Technology

E-mail: hnn_aaa@yahoo.com

Abstract

In this paper a proposed design of five multi-layer feed-forward Artificial Neural Networks (ANNs) is presented for optimized Butterworth filter. The first and second network perform Butterworth ideal Low Pass Filter (LPF) and typical LPF. The third ANN performs Band Pass Filter (BPF). The fourth network perform multi–BPF which consists of two layers, the first layer consists of six tansig neurons and the second layer consists of one purline neuron, and the fifth feed-forward network is designed to perform the High Pass Filter (HPF) which consists of three layers, the first layer consists of three tansig neurons, the second layer consists of three tansig neurons and the third layer consists of one purline neuron. Back-propagation training algorithm is used to train the proposed networks with Mean Square Error (MSE) equals 10^{-10}. Simulation and test programs are implemented by using MATLAB.

Key word: Artificial Neural Networks, Digital Signal Processing, Filters.

تصميم الشبكات العصبية متعددة الطبقات لتمثيل مثالي لمرشحات البترورث

أ.م.د. حنان عبد الرضا عكار
الجامعة التكنولوجية/قسم الهندسة الكهربائية والالكترونية
بغداد/العراق

الملخص:

تم في هذا البحث تصميم خمس شبكات عصبية متعددة الطبقات لتمثيل مثالي لمرشحات البترورث.حيث تنفذ الشبكتان العصبيتان الأولى و الثانية مرشح الترددات الواطئة المثالي والعملي . أما الشبكة العصبية الثالثة فتقوم بتنفيذ مرشح الترددات ألحزمي . أما الشبكة العصبية الرابعة, التي تنفذ مرشح الترددات ألحزمي المتعدد فتتكون من طبقتان, حيث تحتوي الطبقة الأولى على ست خلايا عصبية و تحتوي الطبقة الثانية على خلية عصبية واحدة فقط. وأخيرا الشبكة العصبية الخامسة التي تنفذ مرشح الترددات العالي فتتكون من ثلاث طبقات حيث تحتوي الطبقة الأولى على ثلاث خلايا عصبية و تحتوي الطبقة الثانية على ثلاث خلايا عصبية . أما الطبقة الثالثة فتحتوي على خلية عصبية واحدة فقط. تم استخدام الخوارزمية ذات الانتشار العكسي في تدريب الشبكات العصبية حيث تم الحصول على المعدل التربيعي للخطأ بحدود 10^{-10}. تم تدريب الشبكات و اختبارها باستخدام MATLAB.

Introduction

ANN has been studied for many years in the hope of achieving human like performance in the field of speech and image recognition (Stuart, 2007) . In the case of artificial net, the neuron is a node or processing element, which processes weighted inputs and produces outputs which might be used as inputs to other nodes (Sivanandam, 2006).

Digital filters are widely used in processing digital signals of many diverse applications, including speech processing and data communications, image and video processing, sonar, radar, seismic and oil exploration and consumer electronics (Madisettf, 1999). The design and realization of digital filters involve a blend of theory, applications and technologies. For most applications, it is desirable to design frequency selective filters which alter or pass unchanged different frequency components (Haykin, 1999).

In this paper, a proposed feed-forward ANN are designed to optimized common ideal & typical digital filter types (low pass, band pass, high pass and multi-band pass) with sharp cut-off edge that cannot be implemented directly. They must be approximated with realization system the sharp cut-off edges need to be replaced with transition bands in which the designed would change smoothly in going from one band to the other.

Butterworth Polynomial Filter Characteristics

The Butterworth filter provides the best Taylor series approximation to the ideal LPF response at analog frequencies (0 and ∞) for any order n. The Butterworth polynomials are polynomials of order n whose magnitude is given by (Stein, 2000):-

$$\left| B_n\left(\frac{\omega}{\omega_a}\right) \right| = \sqrt{1+\left(\frac{\omega}{\omega_a}\right)^{2n}} \qquad (1)$$

Where ω is $2\pi f$, ω_a is $2\pi f_0$ and f_0 is a resonant frequency.
LPF is formed by taking the reciprocal of these polynomials (Kara, 2001).

$$\left| \frac{A_V}{A_{VO}} \right| = \frac{1}{\sqrt{1+\left(\frac{\omega}{\omega_a}\right)^{2n}}} \qquad (2)$$

Where A_V is the gain of filter and A_{VO} is resonant frequency gain.
As n increases, $|A_v/A_{vo}|$ becomes closer to one for $\omega \langle \omega_a$, and $|A_v/A_{vo}|$ falls off more sharply for $\omega \rangle \omega_a$ .When $\omega = \omega_a$, $|A_v/A_{vo}| = 1/\sqrt{2}$ regardless of n.

Back-Propagation Training Algorithm

In this section a trainable layered neural networks employing the input data is presented. In the case of layered network training, the error can be propagated into hidden layers so that the output error information passes back-ward. This mechanism of back ward error transmission is used to modify the synaptic weights of internal and input layers. The back-propagation algorithm is used throughout this paper for supervised training of multilayer FFNN (Kabir, 2005).

The back propagation designed to minimize the Mean Square Error (MSE) between the actual output of a multi-layer FFNN and the desired output. The following steps give summery of back-propagation algorithm(Hsu, 2005):-

Step 1: $\eta > 0$, E_{max} chosen , Where weights W and V are initialized at small random values: W is (k x J) and V is (J x I).

Step 2: Training step starts here, Input is presented and layers outputs computed [f(net)].

$$f(net) = \frac{2}{1+\exp(-\lambda net)} - 1$$

$y_j \leftarrow f(v_j^t x)$, for j=1,2,...J

Where v_j is the j'th row of V weights, and $o_k \leftarrow f(w_k^t y)$, for $k = 1,2,...K$

Where w_k is the k'th row of W weights.

Step 3: Error value is computed:

$$E \leftarrow \frac{1}{2}(d_k - o_k)^2 + E, \quad \text{for } k = 1,2,...K$$

Step 4: Error signal vectors δ_o and δ_y of both layers are compute. Vector δ_o is (Kx1), and δ_y is (Jx1). The error signal terms of the output layer in this step are:

$\delta_{ok} = \frac{1}{2}(d_k - o_k)(1 - o^2)$, for $k = 1, 2,... K$ The error signal terms of the hidden layer in this step are: $\delta_{yj} = \frac{1}{2}(1 - y_j^2)\sum_{k=1}^{K} \delta_{ok} w_{kj}$ for j=1, 2… J.

Step5: Output layer weights are adjusted:

$w_{kj} \leftarrow w_{kj} + \eta\delta_{ok} y_j$, for k=1, 2… K and j=1, 2… J.

Step 6: Hidden layer weights are adjusted:

$v_{ji} \leftarrow v_{ji} + \eta\delta_{yj} x_i$ for j=1,2,…J and i=1,2,.. I.

Step 7: Repeat by going to step 2.

Step 8: The training cycle is completed. for $E<E_{max}$ terminate the training session. If $E>E_{max}$, then $E \leftarrow 0$, and initiate the new training cycle by going to step 2.

Description and Computer Simulation Results

The feed-forward back-propagation neural network which optimized the ideal LPF and typical polynomial LPF is shown in **Figure (1).** The network is simulated using MATLAB and its output is plotted against the target as shown in **Figure (2)**, where cut frequency is 0.1 rad/sec. The weights transpose vector between inputs and first hidden layer are:

$V= [-1.6970\ 1.6970\ -1.6970\ -1.7017\ -0.5252\ 30.9583]^t$

The weights of **bias(b)** are:

[-1.6970 1.6970 -1.6970 -1.7017 -0.5252 30.9583].

$$W_1 = \begin{bmatrix} -1.2 & -0.5 & -0.5 & 0.6 & 4.3 & -0.1 \\ -1.7 & -0.2 & -1.2 & 1.6 & 2.1 & -2.8 \\ -1.4 & 0.8 & 0.2 & -0.4 & 3.2 & 0.3 \\ 1.2 & -0.5 & 0.7 & 1.8 & -0.7 & -0.6 \\ -0.2 & 1.2 & -0.8 & -0.4 & -0.4 & -1.4 \end{bmatrix}$$

$$W_2 = \begin{bmatrix} -0.9 & 1.2 & -0.5 & 0.7 & 1.3 \end{bmatrix}^t$$

The performance training is measured according to the Mean Square Errors MSE, which equals to 10^{-10} and learning parameter used is (1). The network is trained for (165) epochs with time of (10.094) sec, as shown in **Figure (3)**. For typical LPF the simulated FFNN is shown in **Figure (4)** at cut-off frequency of (1.5) rad/sec.The network is trained for (1109) epochs to reach the performance goal which is (10^{-10}) with elapsed time equals (21.844) sec as shown in **Figure (5).**

Other FFNN is designed to perform the ideal BPF, it consists of three layers, the first layer consists of six tansig neurons, the second consists of four tansig neurons and third layer consists of one purline neuron, as shown in **Figure (6).** The output simulation result of the FFNN for BPF network is shown in **Figure (7).** The adaptation is done with trains, which updates weights with specified learning function. The network is trained for (285) epochs with elapsed time equals (30.39) sec, as shown in **Figure (8).**

A proposed FFNN design of Multi-band pass filter is shown in **Figure (9).** The network consists of two layers. The first layer consists of six tansig neurons and the second layer consists of one purline neuron. The output of the simulated network is shown in **Figure (10).** The MSE (10^{-10}) of the network with (303) epochs and elapsed time equals to (14.56) sec is shown in **Figure (11).**

The last proposed design FFNN of HPF is shown in **Figure (12).** The network consists of three layers. The first layer consists of three tansig neurons, the second layer consists of three tansig neurons, and the third layer consists of one purline neuron. The output of the simulated network is shown in **Figure (13).** The MSE (10^{-10}) of the simulated network with (303) epochs and elapsed time equals to (14.56) sec. is shown in **Figure (14).**

The following table illustrates the structure of FFNN for ideal LPF, typical LPF, ideal BPF, Multi-band PF and ideal HPF with MSE, No. of epochs, Elapsed time and cut-off frequency for each network. The cut-off frequency illustrated in **Table (1)** is corresponding to the location of a sharp edge of the output response of the filter as shown in **Figures (2, 4, 7, 10, and 13).**

Conclusions

In this paper, five feed-forward neural networks are proposed for Butterworth filter optimization. These networks learn the basic analog prototype for different types of classical Butterworth filter and summarize major characteristics. The created networks are trained using back-propagation algorithm that minimizes the MSE between the actual output and desired output). The use of Multi-layer Feed Forward Neural Networks (MFFNN) with Butterworth filter shows fast response for producing the estimated output signals. The estimated results are near to the actual values according to the value of Mean Square Error (MSE) based on the use of back- propagation training algorithm. This simplifies the development with better performance and fast computations of such type of applications. These proposed FFNN are very efficient and fast because of very low MSE (10^{-10}) is achieved, which gives high learning to the network to response for any selected frequency.

References

Haykin, S., (1999) " **Signal and Systems**", John Willy & Sons, Inc.

Hsu, D., (2005) " **Competitive Learning with Floating Gate Circuits** ", IEEE on circuits and systems.

Kabir, A.,(2005) " **Implementation of Multi-layer Neural Network on FPGA",** College of Technology Indiana State University, March.

Karu, Z. Z, (2001).," **Signals and Systems Made Ridiculously Simple**", Zizi Press, Cambridge MA.

Madisettf, V. K, (1999), "**Digital Signal Processing Handbook",** Chapman & Hall / CRC Press LLC.

Sivanandam, S. N, (2006)," **Introduction to Ann's**", VIKAS publishing house PVT LTD.

Stein, J.Y, (2000) " **Digital Signal Processing** a **Computer Science Perspective**", Widely Inter-Science Publication John Wiley & Sons, Inc.

Stuart, R., (2007) " **Artificial Intelligence**: **a Modern Approach**", 3rd Edition, Prentice Hall.

Table (1) Simulation results of the proposed design.

Type of filter	FFNN layers	MSE	No. of epochs	Elapsed time sec.	Cut-off frequency rad/sec
Ideal LPF	6-5-1	10^{-10}	165	10.094	0.1
Typical LPF	6-5-1	10^{-10}	1109	21.84	1.5
Ideal BPF	6-4-1	10^{-10}	285	30.39	0.3-0.8
Multi-band PF	6-1	10^{-10}	303	14.56	0.1-0.4 0.5-0.8
Ideal HPF	3-3-1	10^{-10}	3045	47.2	0.1

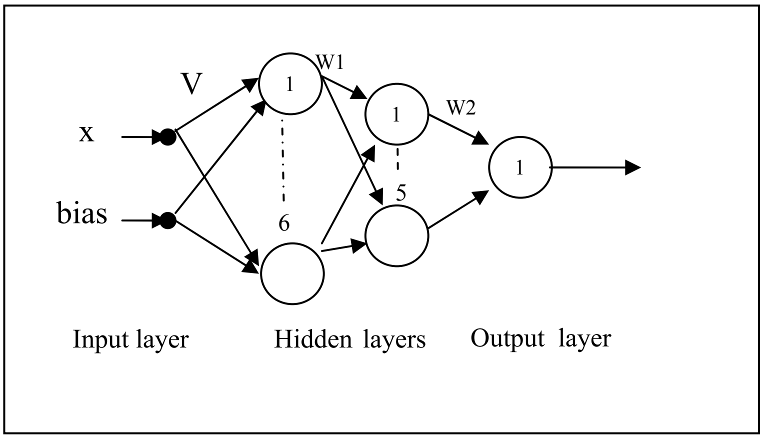

Figure (1) The first and second FFNN for ideal and typical LPF.

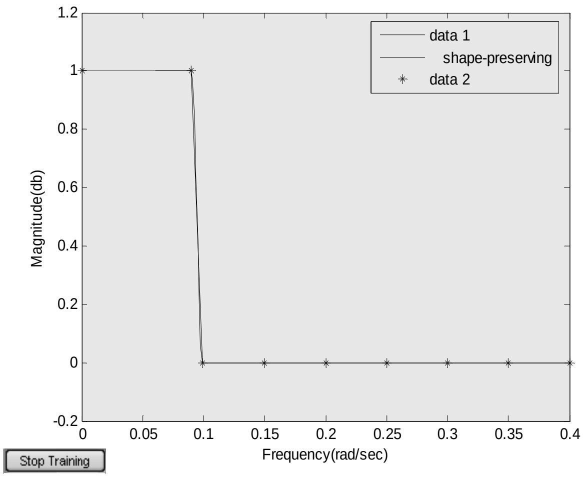

Figure (2) The output of ideal LPF FFNN.

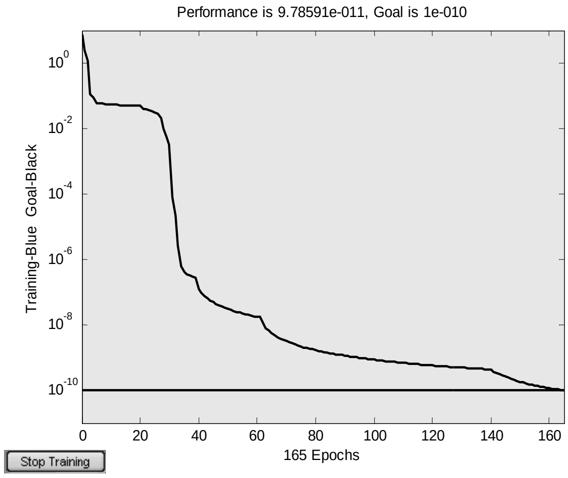

Figure (3) The MSE of ideal LPF FNNN.

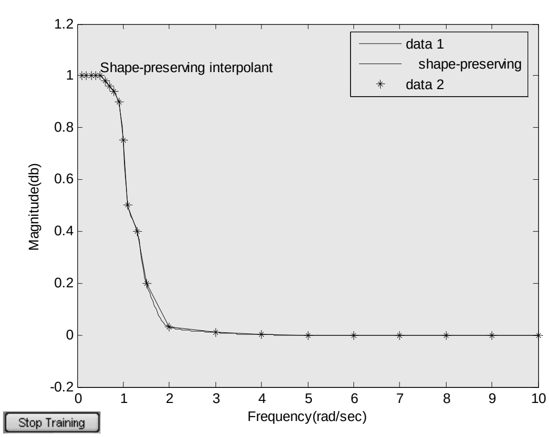

Figure (4) The output of typical LPF FFNN.

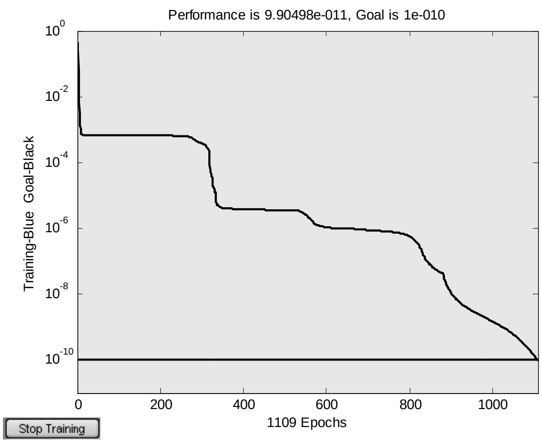

Figure (5) The MSE of the typical LPF FFNN.

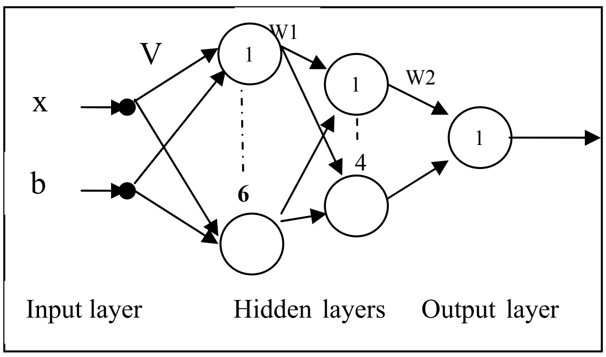

Figure (6) The FFNN for BPF .

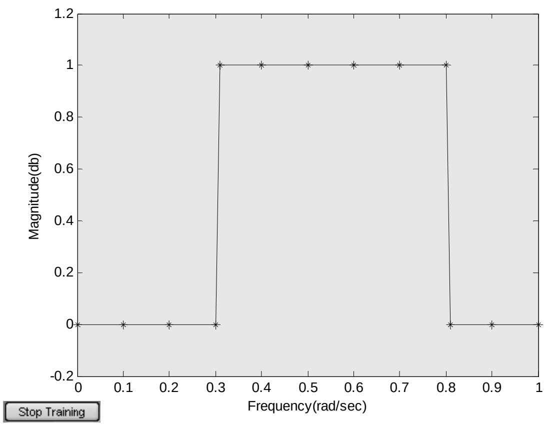

Figure (7) The output of BPF FFNN.

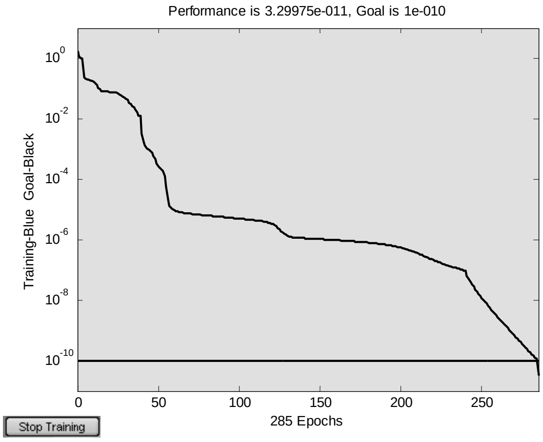

Figure (8) The MSE of BPF FFNN.

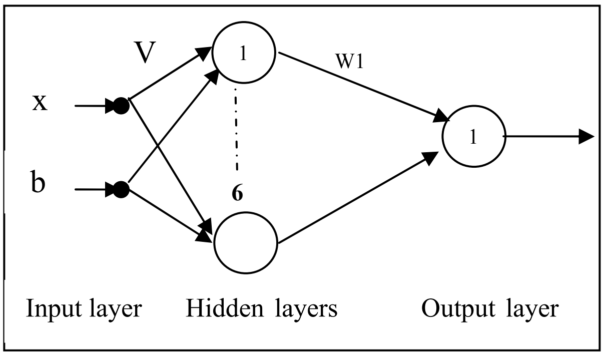

Figure (9) The FFNN of Multi-band PF.

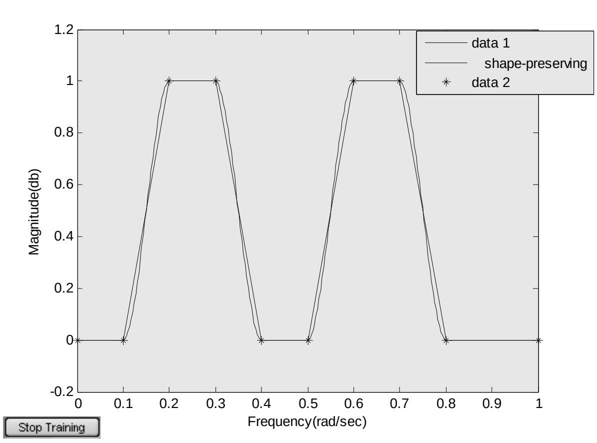

Figure (10) The output of Multi-band PF FFNN.

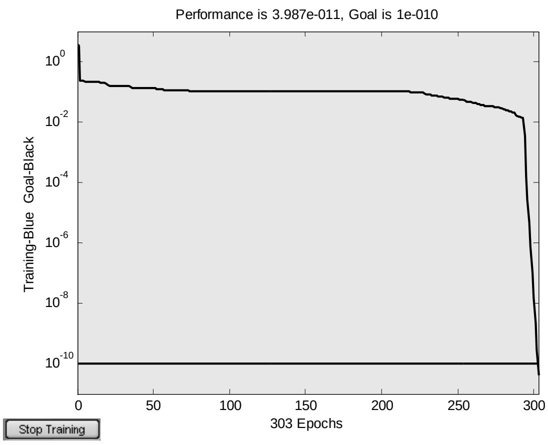

Figure (11) The MSE of Multi-band PF FFNN

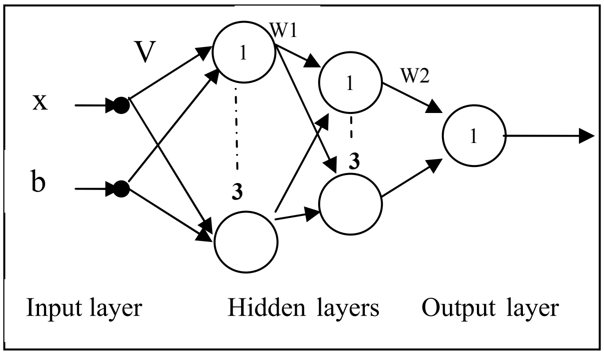

Figure (12) The FFNN for HPF.

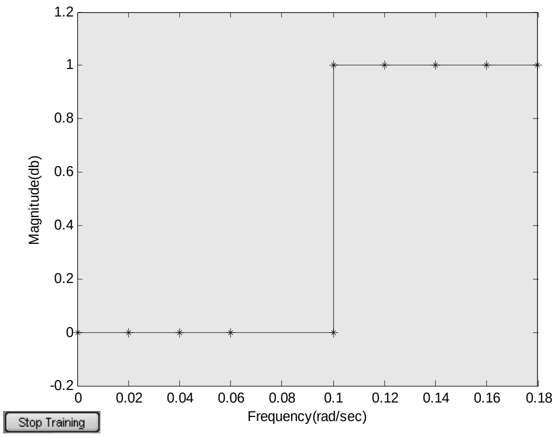

Figure (13) The output of HPF FFNN.

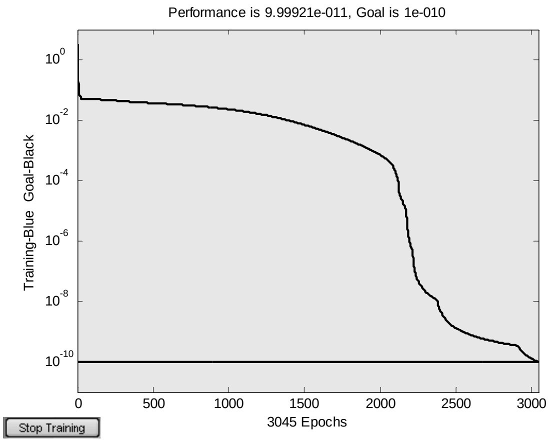

Figure (14) The MSE of HPF FFNN.