AN APPROACH TO REDUCE ERROR OF APPROXIMATION IN SCALON-WAVELON NEURAL NETWORKS

Adityakiran Jagannatham *Shashidhara H.L.* *Vikram M. Gadre*

Department of Electrical Engineering
Indian Institute of Technology Bombay
Powai, Mumbai-400 076 INDIA

ABSTRACT

The function approximation properties of scalon and wavelon neural networks(WNN) are studied from a multi-resolution analysis(MRA) perspective. It is then checked if a superposition of these properties applies to an integrated scalon-wavelon neural network(SWNN). Based on this, a heuristic alogrithm is suggested for reducing the error of approximation in SWNNs.

1. INTRODUCTION

Wavelet neural networks were first proposed and studied for function approximation in [1] and [2]. Wavelet analysis has proved to be a powerful tool for signal representation and analysis [3]. The WNN was motivated by an idea to combine this powerful technique with the neural network approach to approximate functions. Wavelets are well suited for function approximation because of the inherent Time-Frequency localization property of the wavelets. The WNNs were shown in [1] to achieve better approximations compared to the traditional sigmoid based neural networks. Following this, other WNN based architectures were proposed in [4],[5].

It has been reported in [1] that, given an appropriate wavelet function $\psi : \Re \rightarrow \Re$ there exists a denumerable family of the form

$$\Phi = \{d^{1/2}\psi(d_k x - t_k); t_k \epsilon \Re, d_k \epsilon \Re_+, k \epsilon Z\} \qquad (1)$$

(where the $d_k s$ are dilations and $t_k s$ are translations of the wavelet),satisfying the frame property. This class Φ is dense in $L^2(\Re)$ implying that given any $f : \Re \rightarrow \Re, f \epsilon L^2(\Re)$ there exists an approximation $\tilde{f}$ of the form

$$\tilde{f}(x) = \sum_{k=1}^{N} w_k d_k^{1/2} \psi(d_k x - t_k) \qquad (2)$$

such that $||f - \tilde{f}||$ can be made arbitrarily small by increasing N. This property naturally extends to the WNN since the parameter values in a WNN are not constrained to belong to countable families and can infact take values on the whole of $\Re$. This class hence includes the denumerable class Φ as a sub-class.

We define the following as being the **"Resources"** in an SWNN.

1. The Scalons
2. The Wavelons
3. The Training Time (proportional to the number of iterations)

In this work, a study has been made towards formulation of a neural network architecture and algorithm to optimize the use of these *resources*. By this we mean, making the best use of resources to reduce the mse error of approximation. The paper is organised as follows. Section II describes the architecture of the scalon network. III-V are about the experiments done and the final sections contain conclusions and scope for future work.

2. THE SCALON NETWORK STRUCTURE

The WNN is essentially based on the wavelet decomposition of a signal. Wavelets are constant Q filters (described in [6]). The higher the center frequency ω_o, the greater the band-width. Wavelets, depending on the scale, occupy different frequency bands. At the low-frequency bands, the BW of the basis wavelets shrinks considerably. Hence, the number of wavelets and consequently the number of wavelons needed to approximate the low frequency components in a signal increases considerably. Infact the DC component can never be approximated by wavelons. In the work in[1], the DC component was subtracted initially from the input-output relation and added as a constant bias later. A better solution to this problem can be found by taking recourse to MRA.

The MRA synthesis of a function begins with a projection of the function on a scalon basis at a coarse scale. It then subsequently adds projections of function on wavelet

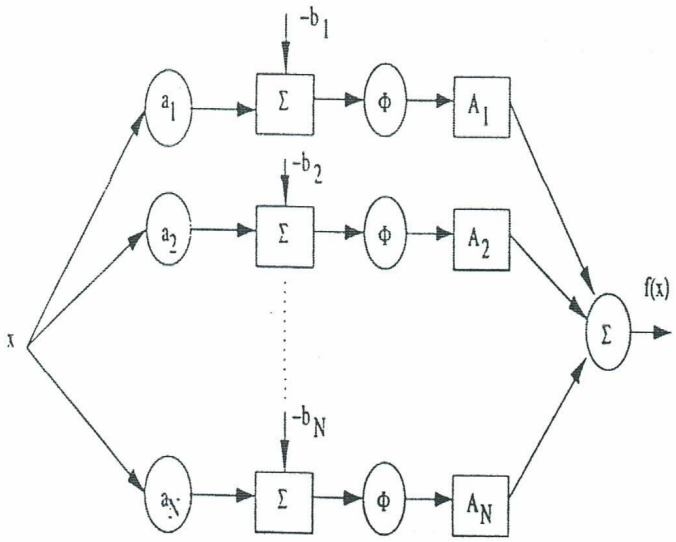

Fig. 1. Scalon Network

subspaces at finer and finer scales. Motivated by this idea, a scalon neural network (SNN) has been proposed analogous to the WNN, with the idea of approximating the low-frequency components. Its structure is shown in Fig.1.

The output relation of an SNN is given as

$$f(x) = \sum_{k=1}^{N} A_{sk}\phi(a_{sk}x - b_{sk}) \tag{3}$$

where N is the number of scalons and A_{sk}, a_{sk}, b_{sk} are the weights, translations and dilations respectively. All of them are adaptable. The scaling function (ϕ) employed was e^{-x^2}.

In the most general case, all the a_{sk}s can be made to vary for different scalons independently. However, in this work, keeping in consonance with the MRA idea of approximating the signal by a scalon basis of a given scale, the scale parameter a_{sk} was taken to be equal for all scalons. This uniform scale parameter is denoted by a_s. Hence the actual output relation is described by

$$f(x) = \sum_{k=1}^{k=N} A_{sk}e^{-(a_s x - b_{sk})^2} \tag{4}$$

This network architecture is aimed at reducing the number of wavelons necessary to represent the low-frequency bands by replacing wavelons with scalons.

3. ADAPTION ALGORITHM

The algorithm analogous to the one described in [1] was used in training. It is similar to the traditional *back-propagation* algorithm for multi-layer perceptrons in an artificial neural network([7]). Denoting the network output by $f(x)$ and the image of x by y, the instantaneous squared error is given by

$$\epsilon = (y - f(x))^2 \tag{5}$$

The algorithm minimizes the error by correcting parameters in the direction of steepest descent. The direction of the steepest descent is found by evaluating the instantaneous gradient of the error with respect to the network parameters. The final parameter updating relations are given by

$$A_{sk} = A_{sk} + 2\eta\epsilon e^{-(a_s x - b_{sk})^2} \tag{6}$$

$$a_s = a_s - 4\eta\epsilon x \sum_{k=1}^{N} A_{sk}(a_s x - b_{sk})e^{-(a_s x - b_{sk})^2} \tag{7}$$

$$b_{sk} = b_{sk} + 4\eta\epsilon A_{sk}(a_s x - b_{sk})e^{-(a_s x - b_{sk})^2} \tag{8}$$

where η, the *Learning Constant*, determines the rate of training.

The error of approximation E, is given by

$$E = (\sum_{\forall x}(y - f(x))^2 * step\_size)^{\frac{1}{2}} \tag{9}$$

where *step_size* is the interval length of training data sampling and the summation is evaluated over all the training input-output pairs. This measure is an absolute error measure, in that it is a measure of the total area under the error curve and is not an average error measure like the *mse*. But it corresponds directly to the *mse* and is a more stringent error measure.

4. INITIALIZATION

The scalons and wavelons were distributed uniformly over the domain of approximation. The initial weight was in some cases taken to be 1 and in others taken to be the image of the training function at $x = b_{sk}/a_{sk}$. The dilation parameter a for the scalons was set such that the low-frequency band-width of scaling function was equal to that of the training function. Analogously for all the wavelons, it was set equal to that corresponding to the high-frequency band-width. Following this, the translation parameters. b_{sk}s for the scalons and wavelons were initialized using the relation,

$$b_{sk} = a_{sk}\left[LOW + \frac{(k + 0.5)(HIGH - LOW)}{UNIT\_NO.}\right] \tag{10}$$

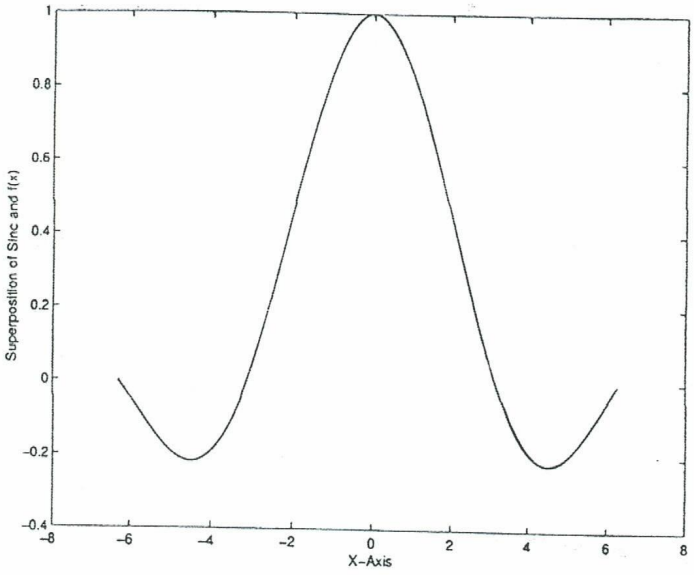

Fig. 2. Scalon Network Approximation to Sinc Function with 20 Scalons after 5000 Learning Iterations

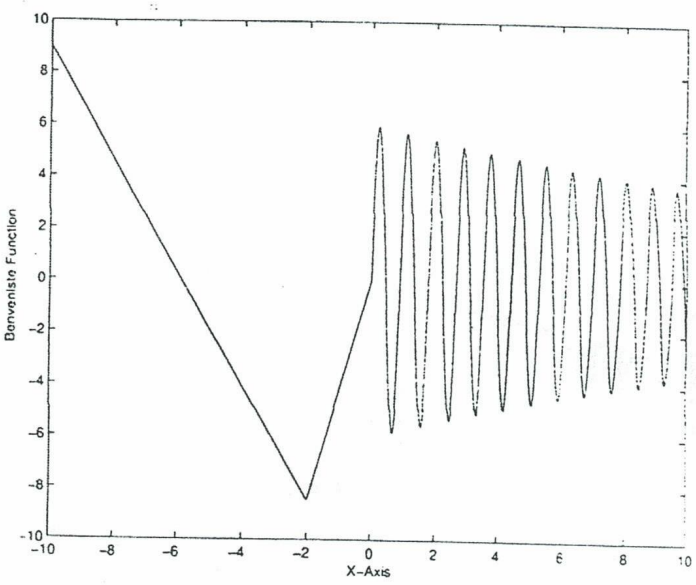

Fig. 3. The 'Benvensite' Function

where LOW and HIGH are the corresponding bounds of the domain of approximation and $UNIT\_NO.$ corresponds to the scalon or wavelon units respectively. This initialization procedure was formulated with an aim of allocating the scalon and wavelon units uniformly over the domain of approximation. The precise initial parameter values and other data about the experiment are given alongwith the corresponding descriptions of the experiments. The network output is also plotted against the training function.

4.1. Experiment I

This was carried out with the aim of testing the hypothesis that the scalon network well approximates input-output relations having solely low frequency components. The sinc function, which is a perfect band-limited function, was used to train the scalon network. The input-output pairs used for training were samples of the training function taken at intervals of length 0.005 over $(-2\pi, 2\pi)$, the domain of approximation.

The scalons were initialized uniformly over the domain. The initial weights were taken to be 1. Initial scale a was set equal to 0.71 and the translation parameters, b_{sk}s were set by the relation mentioned previously. Scalon number was 10 or 20. Iterations varied from 2000 to 30000. η, the learning constant was 0.001.

The results obtained were in good confirmation with the hypothesis. The approximation was quite good even for a few scalons and few iterations. In fact, in the plots shown, the approximation to the function is so good than one can hardly distinguish between the actual image of x and the network output. The plot shown (Fig.2.) actually shows both the function and its approximation, which almost exactly superpose over each other. The error of approximation is of the order of 10^{-3}.

5. DESCRIPTION OF THE BENVENISTE FUNCTION

The Benveniste Function (Ben function) is given as

$$f(x) = \begin{cases} -2.186x - 12.864 & -10 \leq x < -2 \\ 4.246x & -2 \leq x < 0 \\ 10e^{-0.05x-0.5}sin[(0.03x+7)x] & 0 \leq x \leq 10 \end{cases}$$

Its plot is shown in Fig.3. It is a continuous function and analytic at all points except for those on the boundaries of the intervals of definition. It is very similar to the function employed in [1] to evaluate the performance of the WNN. In this work it is referred to as the 'Ben' function, after Benveniste, who presented the work. The frequency characteristic (shown in Fig 4.)of the Ben function exhibits a unique feature. It contains 2 distinct parts- A low-frequency peak and a high-frequency peak. The low-frequency peak and the high-frequency peak map almost exactly onto the initial$(-10 \leq x < 0)$ and later$(0 \leq x \leq 10)$ parts of the function respectively. Hence the corresponding sections in time of the Ben function will be referred to as the **'LF'** region and the **'HF'** region in future sections of this paper. This clear cut mapping of the low-frequency and high-frequency regions to distinct regions in time has been helpful in analysing the experimental results. A systematic study has been conducted employing this function to train the networks and is described below.

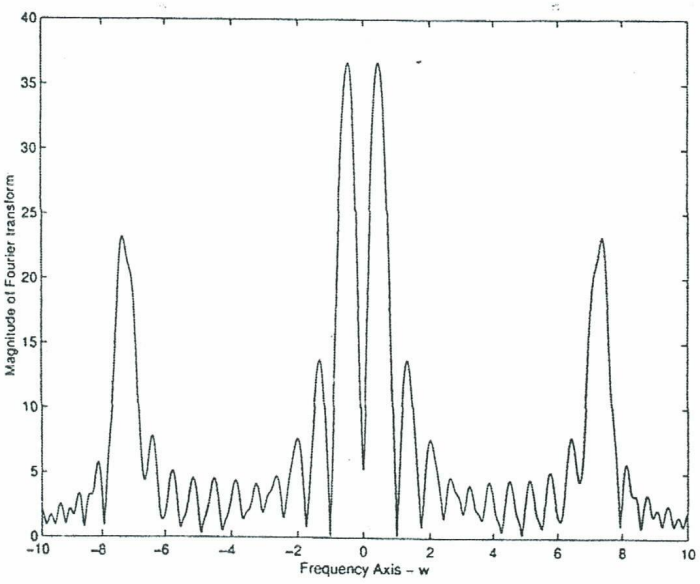

Fig. 4. Frequency Characteristic of the 'Benveniste' Function

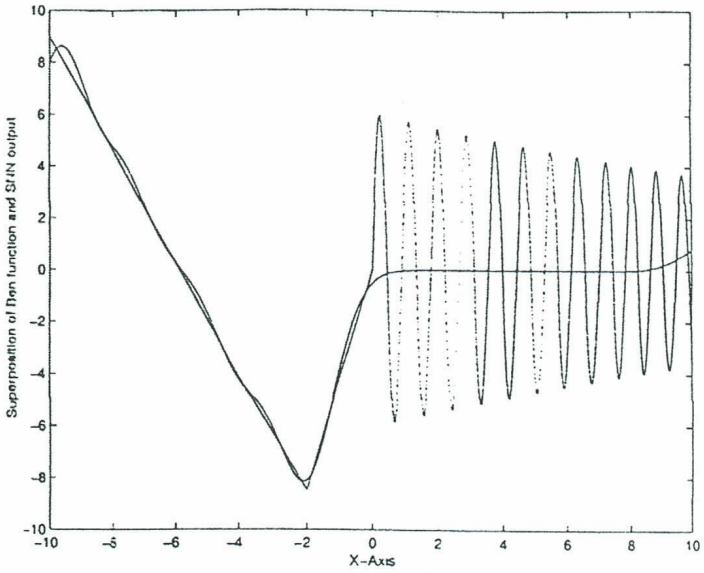

Fig. 5. 'Ben' function approximation of SNN with 8 Scalons and after 15,000 learning iterations

5.1. Scalon Network

A scalon network as described previously was trained using the Ben function. The Gaussian e^{-x^2} was used as the scaling function.The scalon number was taken over a range of 5 to 29. η, the learning constant was set equal to 0.00001. Scalons were initialized uniformly over the domain. a_s was set equal to 3.5361 so that the frequency BW of the scaling function was equal to the low-frequency BW of the training function. b_{sk} was set using the procedure described above to achieve uniform allotment. The weight A_{sk} was set equal to $y(b_{sk}/a_s)$, where y is the Ben function. Around 15000 to 20000 learning iterations were carried out

What was observed was in good confirmation with the expected result. The approximation to the function (Fig. 5.)is fairly accurate in the LF region, but very poor in the HF region. In fact, for low scalon numbers, the approximation in HF is almost nil. From the plot of the frequency characteristic(Fig. 6.) it can be seen that the approximation to the function is almost exact in the LF region. The weight parameters of the scalons assigned to the HF region (5-8 in table 1.) were almost 0 in the final trained network. So these 'resources' are almost 'wasted' or used inefficiently. This experiment also strengthens the hypothesis that scalons are efficient in approximating the LF regions in a function.

The error of approximation was computed and plotted against the scalon number(Fig. 7.). It can be seen that it first increased and then decreased as a function of scalon number. The approximation quality decreases when the number of scalons is increased beyond a certain optimal number. Thus the loss is two fold: The scalon resources are wasted and the error of approximation in fact increases.

More informative trends can be found by plotting the error of approximation of the LF and HF regions independently against the scalon no.. The plots (plot no.s 8,9) reveal that the LF error for 8 scalons is the lowest and in fact is higher for higher scalons numbers while the HF error decreases uniformly as the scalon no. increases. This indicates of a possible *'trade-off'* between HF and LF errors. The HF error decreases as the scalon no. increases - but at the cost of the LF error itself. Moreover, since the scalon approximation to the HF is too poor to be of any significance, an optimal low scalon no. in the LF could lead to improved accuracy of approximation.

1. Scalon Parameters of SNN

	A_{sk}	a_s	b_{sk}
1.	8.310821	0.819249	-7.952273
2.	3.737510	0.819249	-6.355804
3.	-3.657662	0.819249	-3.216021
4.	-7.799890	0.819249	-1.650180
5.	0.019400	0.819249	1.925398
6.	0.999730	0.819249	8.666220
7.	0.453704	0.819249	14.116468
8.	0.567481	0.819249	20.860277

This experiment sheds light on certain very important aspects of the nature of approximation. Scalons are well suited for approximation of the LF regions in a function. So concentrating the scalon resources in the LF regions of the function improves resource utilization, and could be a first step towards optimal resource utilization.

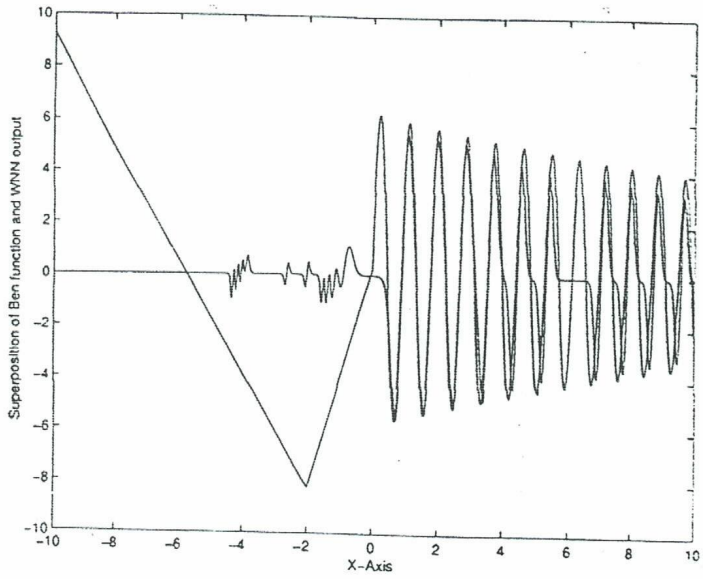

Fig. 10. WNN approximation to 'Ben' function with 20 wavelons and 10,000 training iterations

5.2. Wavelon Network

The previous experiment was repeated using a wavelon network. The initialization procedure was as described for the scalon network. However, the scale parameter a_{wk} was set equal to 7.07 for all wavelons, corresponding to the HF band-width. Weights were initialized to 4.00. The mean subtracted function was used to train the network. The final approximation displayed characteristics parallel to the scalon case. The network output approximates the function well in the HF region, but is poor in the LF region (Fig. 10.). The wavelon weights of those assigned to the LF region setted to very low magnitudes (1-10 in table 2.).

From this and the above results, a pattern can be found to emerge. *Wavelons are well suited for approximating the HF region and scalons, for the LF regions in a function.* This is in accordance with MRA based analysis where during the analysis stage, the HF bands are projected on a wavelet basis and the LF band is projected on a scaling function basis. An efficient allocation scheme preferentially allocates scalons to LF and wavelons to HF regions in a function.

To gather further evidence for such a trend, the following experiment was carried out. The approximation to Ben function of the scalon network was subtracted from the Ben function. The resultant function is in fact the Ben function minus the LF components, and hence comprises only the HF part. This resultant map was used to train the wavelon network. This map is almost 0 in $(-10 \leq x \leq 0)$. The wavlons were initialized uniformly over the domain. Initial values were same as before.

Now since all the parameters of the wavelons are adaptable one would expect the wavelons to shift to the HF region $(0 \leq x \leq 10)$ to minimize the error. However the results were quite the contrary. The wavelons allotted to the $(-10 \leq x \leq 0)$ region settled to almost 0 final weights. So these resources are wasted. It was also observed that the wavelons do not translate significantly in the domain and only the wavelon weights undergo significant change. This illustrates that wavelon resources improperly allotted over the domain are used inefficiently.

2. Wavelon Parameters of WNN

	A_{wk}	a_{wk}	b_{wk}
1.	2.197357	15.273399	-65.768867
2.	2.807608	14.080389	-58.906910
3.	2.587545	12.818356	-51.980099
4.	1.625462	11.511245	-45.089684
5.	1.042000	13.995013	-37.030735
6.	1.178440	14.462476	-29.349096
7.	2.268684	14.045589	-21.525517
8.	3.229549	10.856640	-15.076904
9.	3.059121	7.577526	-9.130875
10.	2.606002	4.128893	-3.725085
11.	12.973482	3.790877	3.427690
12.	12.457458	3.575250	6.369643
13.	11.931827	3.793694	10.076485
14.	11.207683	4.690423	16.660389
15.	10.088662	5.925420	6.297863
16.	9.305100	6.591591	34.994324
17.	9.007938	6.148407	43.222805
18.	8.435363	6.463120	50.942917
19.	7.920842	6.704997	58.519291
20.	7.453274	6.896806	65.983299

Following the previous initialization scheme, all the wavelons were alloted to the $(0 \leq x \leq 10)$ region. The accuracy of approximation improved and the error of approximation decreased from 6.50 to 4.02. This strengthens the proposition that an efficient scheme must preferentially allot wavelons to the HF regions in the function.

5.3. The Scalon-Wavelon Structure

Motivated by the results of the previous sections, the properties of a combined scalon-wavelon network were investigated. It was desired to check how far the results of the previous sections extend to the integrated network. The SWNN contains many scalon and wavelon units in parallel, analogous to a WNN.

The scaling function, ϕ considered was e^{-x^2} and the wavelet ψ employed was the Gaussian xe^{-x^2}. The output error is given by

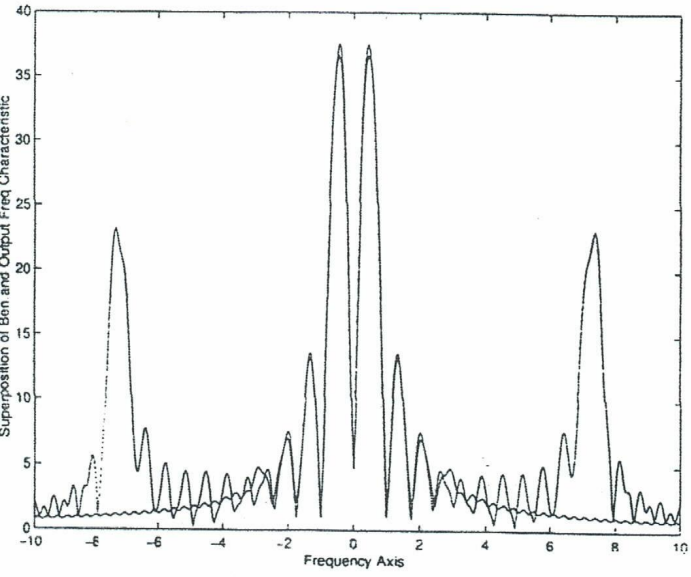

Fig. 6. Frequency domain plot of SNN approximation to 'Ben' function

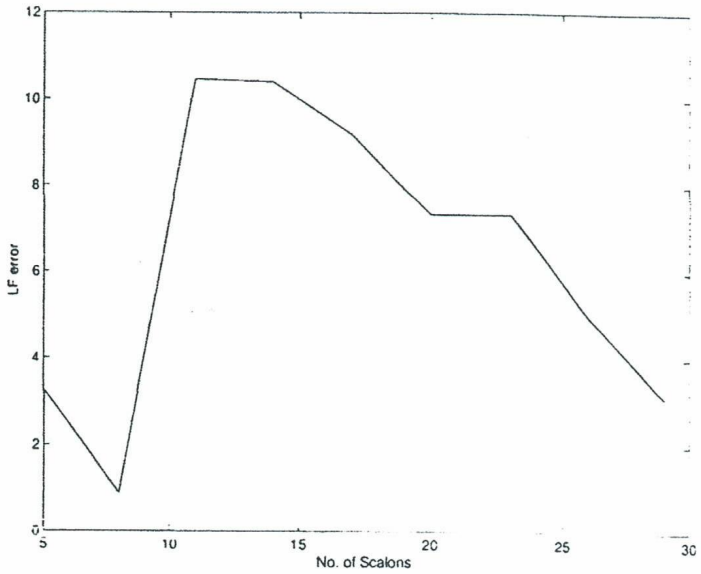

Fig. 8. 'LF' error vs scalon no. in SNN

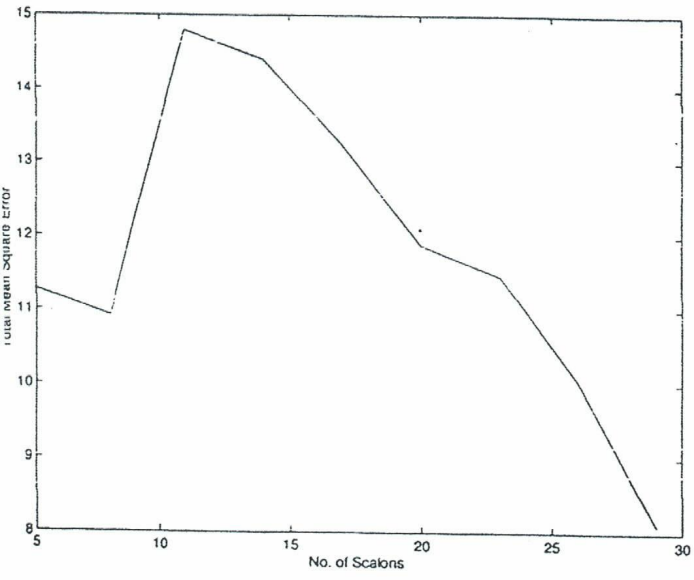

Fig. 7. MSE error vs scalon number in SNN

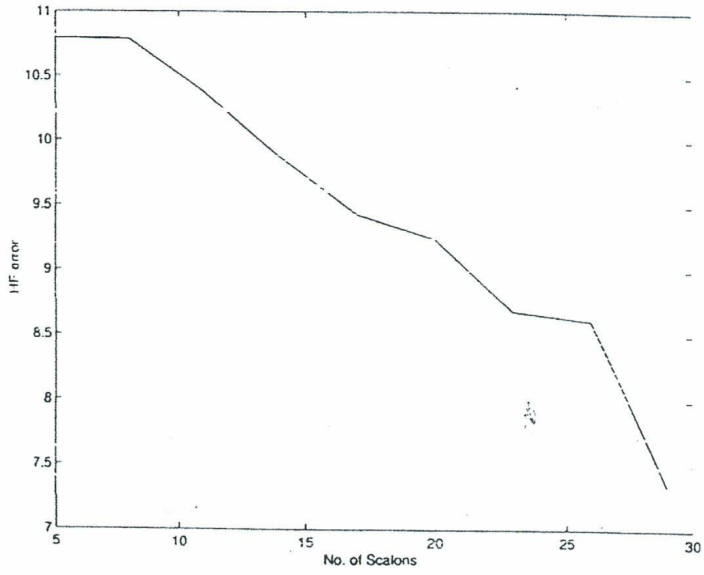

Fig. 9. 'HF' error vs scalon no. in SNN

$$\epsilon = \left(y - \sum_{k=1}^{k=N} A_{sk}\phi(a_s x - b_{sk}) - \sum_{k=1}^{k=M} A_{wk}\psi(a_{wk}x - b_{wk}) \right)^2 \quad (11)$$

where N is the scalon no. and M, the wavelon no. The corresponding parameter updating relations for scalons are the same as those given in section III, except that the error ϵ is now computed as given above. The relations for the wavelons are given as

$$A_{wk} = A_{wk} + 2\eta\epsilon(a_{wk}x - b_{wk})e^{-(a_{wk}x - b_{wk})^2} \quad (12)$$

$$d_{wk} = (1 - 2(a_{wk}x - b_{wk})^2) \quad (13)$$

$$a_{wk} = a_{wk} + 2\eta\epsilon A_{wk}xe^{-(a_{wk}x - b_{wk})^2}d_{wk} \quad (14)$$

$$b_{wk} = b_{wk} - 2\eta\epsilon A_{wk}e^{-(a_{wk}x - b_{wk})^2}d_{wk} \quad (15)$$

The network consisted of 20 wavelon and 10 scalon units. Initial scalon and wavelon scales were set equal to 2.00 and 7.07 respectively. Weight A_{wk} was set equal to $y(b_{wk}/a_{wk})$, where y is the training function. Initialization was uniform over the domain of approximation. Learning iterations ranged from around 10000 to 25000. This network was then trained to learn the Ben function. The final approximation indeed reflected a superposition of both the previous results. The scalons naturally dominated in magnitude in the LF region of the function and the wavelons dominated the HF region. Tables 3,4 give the parameter values and output is plotted in Fig. 11. The upper entries in the tables correspond to the LF region and the lower ones correspond to the HF region. It is thus very clearly seen that resources alloted improperly *i.e.* wavelons to LF and scalons to HF, are wasted.

3. Scalon Parameters of SWNN

	A_{wk}	a_s	b_{wk}
1.	8.361009	0.846162	-8.151902
2.	3.762330	0.846162	-6.527406
3.	-3.311086	0.846162	-3.418227
4.	-5.284047	0.846162	-2.015855
5.	-3.542618	0.846162	-1.268351
6.	2.624141	0.846162	3.247147
7.	-1.970824	0.846162	3.961793
8.	0.369626	0.846162	7.254347
9.	0.752692	0.846162	13.330645
10.	0.657769	0.846162	17.021389

To complete the experiment, the previously uniformly initialized scalons and wavelons were now shifted. Scalons were uniformly initialized over the LF domain and wavelons over the HF domain. The initialization procedure was the same as before. The accuracy of approximation was better as compared to the previous one (Fig. 12). This can easily be seen from a significant decrease in the error from 5.40 to 3.76. This further decreased to 3.16 after 40000 learning iterations.

4. Wavelon Parameters of SWNN

	A_{wk}	a_{wk}	b_{wk}
1.	4.232576	14.932308	-66.083488
2.	5.078144	13.638769	-59.239033
3.	1.979597	12.236225	-52.129311
4.	0.747620	10.980485	-45.189137
5.	-3.917887	8.822230	-38.497181
6.	-0.641682	8.425750	-31.635147
7.	-0.558577	7.033739	-24.887068
8.	-0.820315	7.954968	-17.158894
9.	-0.683539	9.440727	-8.809009
10.	-0.384227	6.650891	-4.755777
11.	-14.483710	3.265695	1.504464
12.	-14.204625	3.009618	4.041595
13.	-13.747286	2.925975	6.496335
14.	-13.209433	3.498991	10.767590
15.	12.761871	4.034819	17.821144
16.	11.148128	4.955145	26.273531
17.	9.785856	6.166983	38.062695
18.	-9.512387	6.287778	46.899429
19.	-8.686493	6.679718	55.521416
20.	8.358355	6.758178	64.605858

The above result extends the previous results to the integrated SWNN. This can be used to formulate an approach for efficient resource utilization in SWNNs.

6. CONCLUSIONS AND A HEURISTIC OPTIMIZATION SCHEME

Based on the experimental results described in the sections above, certain conclusions can be derived.

1. Scalons well approximate the LF regions and wavelons, the HF regions in a function.

2. Resources can be utilized effectively by allocating scalons to the LF and wavelons to the HF regions in a function, if the frequency characteristic of the function is known apriori. Else, the final parameter values of the trained network for a uniformly initialized network can help identify the HF and LF regions of the function. This information can then be used for initialization.

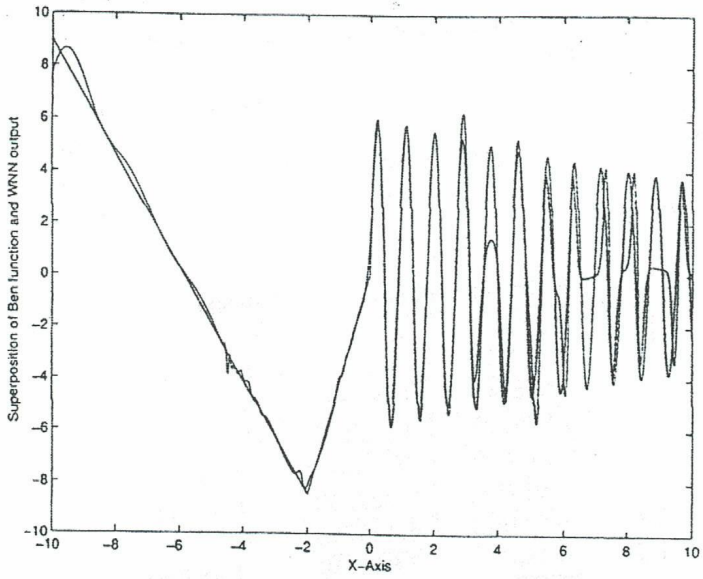

Fig. 11. Approximation of SWNN with 10 scalons and 20 wavelons after 25,000 iterations

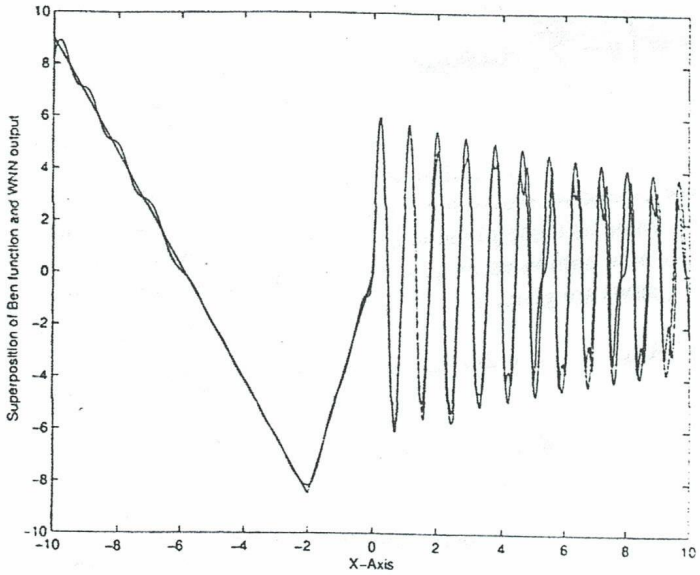

Fig. 12. SWNN approximation after shifting scalons to LF and wavelons in the HF region

3. Scalon and wavelon resources improperly initialized on the domain of approximation will be 'wasted', thereby meaning that they will be used inefficiently.

A heuristic initialization procedure can be suggested based on the above conclusions. The domain of the function or the input-output map to be approximated can be initially partitioned. Over each such partition, the frequency characteristic of the function can be evaluated. Then depending on the nature of the characteristic, scalons or wavelons can be allotted to that particular partition during initialization.

In cases where it might not be possible to neatly resolve the characteristic as HF or LF, depending on the magnitude of the characteristic in the LF and HF regions a proportional allotment can be made. This procedure can be crudely speaking called a "NEED-BASED" allotment scheme, where resources are alloted depending on the nature of the frequency domain characteristic. This sort of an initialization scheme also brings to the fore-front, the "Time-Frequency" nexus in wavelet based analysis.

7. FUTURE WORK

In the future, it is intended to develop an analytical formulation based on the same initialization procedure. Also, a heuristic algorithm to arrive at optimal scalon and wavelon numbers is to be developed.

8. REFERENCES

[1] Q Zhang and A Benveniste, "Wavelet Networks", *IEEE Tr. on Neural Networks*, vol. 3, no. 6, pp. 889-898, Nov. 1992.

[2] Harold H. Szu and Shubha Kadambe, "Neural Network Adaptive Wavelets for Signal Representation and Classification", *Optical Engineering*, vol. 31, no. 9, pp. 1907-1916, September 1992.

[3] C.Sidney Burrus, Ramesh A. Gopinath, Haitao Guo, "Introduction to Wavelets and Wavelet Transforms A Primer", *Prentice-Hall International, Inc.*, 1998.

[4] Shashidara H.L., Sumit Lohani, Vikram M. Gadre, "Function Learning Using Wavelet Neural Networks", *Proc. of IEEE, International Conference on Industrial Technology 2000*, vol. 1, pp. 335-339, Jan. 2000.

[5] Shashidara H.L., Suneel T.S., Vikram M. Gadre, S.S. Pande, "Accuracy Improvement for CNC System using Wavelet-Neural Networks", *Proc. of IEEE, International Conference on Industrial Technology 2000*, vol. 1, pp. 341-346, Jan. 2000.

[6] Raghuveer M. Rao and Ajit S. Bopardikar, "Wavelet Transforms", Addison-Wesley, 1998.

[7] Bernard Widrow and Michael A. Lehr, "30 Years of Adaptive Neural Networks: Perceptron, Madaline and Backpropogation", *Proc. of the IEEE*, vol. 78, no. 9, pp. 1415-1441, Sep. 1990.