NEAR EAST UNIVERSITY

GRADUATE SCHOOL OF APPLIED AND SOCIAL SCIENCES

Multiresolution Prediction using a Combination of Wavelet and Neural Network

Huthaifa Ahmad Al_Issa (991762)

Master Thesis

Department of Computer Engineering

Nicosia - 2005

DEPARTMENT OF COMPUTER ENGINEERING
DEPARTMENTAL DECISION

Date:22/03/2005

Subject: Completion of M.Sc. Thesis

Participants: Assoc. Prof. Dr. Rahib Abiyev, Assoc.Prof. Dr. Sameer Ikhdair, Assist.Prof. Dr. Adel Amirijanov, Huthaifa Ahmad Al_Issa.

DECISION

We certify that the student whose number and name are given below, has fulfilled all the requirements for a M .S. degree in Computer Engineering.

		CGPA
991762	Huthaifa Ahmad AL_Issa	3.83

Prof. Dr. Fakhraddin Mamedov, Supervisor, Dean of Faculty of Faculty of Engineering Department, Director, NEU

Assoc. Prof. Dr. Rahib Abiyev, Committee Chairman, Deputy Chairman of Computer Engineering Department, NEU

Assoc. Prof. Dr. Sameer Ikhdair, Committee Member, Electrical and Electronic Engineering Department, NEU

Assist. Prof. Dr. Adel Amirijanov, Committee Member, Computer Engineering Department, NEU

Chairman of Department
Assoc. Prof. Dr. Doğan İbrahim

Huthaifa Ahmad Al_Issa: Multiresolution Prediction using a Combination of Wavelet and Neural Network.

Approval of the Graduate School of Applied and Social Sciences

Prof. Dr. Fakhraddin Mamedov
Director

We certify this thesis is satisfactory for the award of the Degree of Master of Science in Computer Engineering

Examining Committee in charge:

Prof. Dr. Fakhraddin Mamedov, Supervisor, Dean of Faculty of Engineering Department, Director, NEU

Assoc. Prof. Dr. Rahib Abiyev, Chairman, Deputy Chairman of Computer Engineering Department, NEU

Assoc. Prof. Dr. Sameer Ikhdair, Member, Electrical and Electronic Engineering Department, NEU

Assist. Prof. Dr. **Adel Amirijanov**, Member, Computer Engineering Department, NEU

ACKNOWLEDGMENTS

I could not have prepared this thesis without the generous help of my supervisor, colleagues, friends and family.

I would like to express my gratitude to my supervisor Prof. Dr. Fakhraddin Mamedov for providing invigorating environment in which I could write this thesis.

I would like to thanks Assoc.Prof.Dr.Adnan Khasman, for his help when needed, his patience in correcting both my style and scientific knowledge.

Also I would like to thanks Assoc.Prof Dr.Rahib Abiyev, Assoc.Prof Dr.Sameer Ikhdair, Assist.Prof Dr.Adel Amirijanov for there help and making this thesis possible.

My deepest thanks are to Dr. Jamal Fathi for his help and answering any question I asked him.

Also I would like to express My gratitude to Dr. Tayseer Alshanableh and his family.

Finally, I could never have prepared this thesis without the encouragement and support of my mum, father, brothers and sisters.

To All of Them, My Love and Respect

ABSTRACT

Artificial Neural Networks (ANN) have provided an attractive tool for researcher, claiming improved performance over traditional linear and nonlinear models for prediction non-stationary time series. However ANN's provide best results for short-time prediction. To improve performance of the system, and provide best long time prediction in this thesis the integration of neural network (NN) with wavelet function is considered.

The description of wavelet functions, their operation principles have been explained. Also Neural Network structure their learning algorithms are given. The structure of Neuro-Wavelet system for prediction of time-series describing return-rates of company is described. The development of Neuro-Wavelet hybrid system for long-time prediction that incorporates multi-scale wavelet shell decompositions into a set of neural networks for a multistage prediction is carried out. In the result of modeling the prediction precision was increased, that demonstrate the efficiency of proposed methodology.

CONTENTS

INTRODUCTION

During the last two decades, various approaches have been developed for time series prediction. Among them linear regression methods such as autoregressive (AR) and autoregressive moving average (ARMA) models have been the most used methods in practice. The theory of linear models is well known, and many algorithms for model building are available.

Linear models are usually inadequate for financial time series as in practice almost all economic processes are nonlinear to some extent. Nonlinear methods are widely applicable nowadays with the growth of computer processing speed and data storage. Of the nonlinear methods, neural networks have become very popular. Many different types of neural networks such as MLP and RBF have been proven to be universal function approximators, which make neural networks attractive for time series modeling, and for financial time-series forecasting in particular.

An important prerequisite for the successful application of some modem advanced modeling techniques such as neural networks, however, is a certain uniformity of the data. In most cases, a stationary process is assumed for the temporally ordered data. In financial time series, such an assumption of stationarity has to be discarded. Generally speaking, there may exist different kinds of nonstationarities. For example, a process may be a superposition of many sources, where the underlying system drifts or switches between different sources, producing different dynamics. Standard approaches such as AR models or nonlinear AR models using MLPs usually give best results for stationary time series. Such a model can be termed as global as only one model is used to characterize the measured process. When a series is nonstationary, as is the cage for most financial time series, identifying a proper global model becomes very difficult, un1ess the nature of the nonstationarity is known. In recent years, local models have grown in interest for improving the prediction accuracy for nonstationary time series.

To overcome the problems of monolithic global models, another efficient way is to design a hybrid scheme incorporating multiresolution decomposition techniques such as the wavelet transform, which can produce a good local representation of the signal in both the time domain and the frequency domain. In contrast to the Fourier basis, wavelets can be supported on an arbitrarily small closed interval. Thus, the wavelet transform is a very powerful tool for dealing with transient phenomena.

There are many possible applications of combining wavelet transformations into financial time-series analysis and forecasting. Recently some financial forecasting strategies have been discussed that used wavelet transforms to preprocess the data. The preprocessing methods they used t_i are based on the translation invariant wavelet transform or a *trous* wavelet transform.

It is well known that any functions can be represented as a weighted sum of orthogonal basis functions. Such expansions can be easily represented as neural nets by having the selected basis functions as activation functions in each node, and the coefficients of the expansion as the weights on each output edge. Several classical orthogonal functions, such as sinusoids, Walsh functions, etc., but, unfortunately, most of them are global approximators and suffer, therefore, from the disadvantages of approximation using global functions. What is needed is a set of basis functions which are local and orthogonal. A special class of functions, known as wavelets, possess good localization properties while they are simple orthonormal bases. Thus, they may be employed as the activation functions of a neural network known as the Wavelet Neural Network (WNN). WNNs possess a unique attribute: In addition to forming an orthogonal basis are also capable of explicitly representing the behavior of a function at various resolutions of input variables.

The idea of wavelet preprocessing for enhancing prediction comes from multiresolution analysis provided by wavelet transform. The wavelet transform can decompose one time series into several time series with different resolutions which have different levels of smoothness. The smoother level is more predictable, whereas the rougher (detailed) level is less predictable, or more related to noise.

Aim of this thesis is development of a neuro-wavelet hybrid system that incorporates multi-scale wavelet analysis into a set of neural networks for a multistage time series prediction in stock marketing. The system will exploit a shift invariant wavelet transform called the autocorrelation shell representation (ASR) instead of the multi-scale orthogonal wavelet transform as was originally presented in [4]. It is cumbersome to apply the commonly defined DWT for real-time time series applications due to the lack of shift invariance, which plays an important role in time series forecasting. Using a shift invariant wavelet transform, we can easily relate the resolution scales exactly to the original time series and preserve the integrity of some short-lived events.

This thesis consists of four chapters, introduction, conclusion, and appendix.

Chapter 1 provides a brief background on wavelet transforms, Continuous-Time Wavelet, and their time and frequency domain and Discrete-Time Wavelet selectively.

Chapter 2 gives an overview of neural networks, its history, simple structure, biological analogy and the Perceptron Algorithm, and the Activation function and the Backpropagation algorithm, and their models.

Chapter 3 states the purpose of this thesis and provides some motivation behind the thesis.further, we extend the use of wavelet networks for function approximation to dynamic nonlinear input-output modeling of processes. We show how to train such networks by a classic minimization of a cost function through second order gradient descent implemented in a backpropagation scheme, with appropriate initialization of the translation and dilation parameters, and from orthogonal wavelet decomposition to wavelet networks.

Chapter 4 provides on Times-Series Comparison Method of Prediction, and Design of Wavelet Neural Network Multiresolution System, and their modeling, and presents some details of the experiments performed, and presents the results of those experiments.

Finally, we give the conclusions derived during the investigation of using neuro-wavelet networks for predictions are given.

1. INTRODUCTION TO WAVELET TRANSFORM

1.1 Overview

Wavelet transforms were provided by the fact that certain seismic signals can be modeled suitably by combining translations and dilations of a simple, oscillatory function of finite duration [1]. The early results were related to what is now known as the *continuous wavelet transform* (CWT). However, wavelet transform like expressions can be found in earlier work done in several fields such as function representation, quantum mechanics, and signal processing. In this chapter we provide an introduction to CWTs and DWTs their properties.

1.2 Continuous-Time Wavelet

Consider a real or complex-value continuous-time function $\psi(t)$ with the following two properties [1]:

1. The function integrates to zero:

$$\int_{-\infty}^{\infty} \psi(t)dt = 0 \tag{1.1}$$

2. It is square integrable or, equivalently, has finite energy:

$$\int_{-\infty}^{\infty} |\psi(t)|^2 dt < \infty \tag{1.2}$$

The function $\psi(t)$ is a *mother wavelet* or wavelet if it satisfies these two properties as well as the admissibility condition defined later in this chapter. While the admissibility condition is useful in formulating a simple inverse wavelet transform, properties 1 and 2 suffice to define the CWT, and they capture essentially the reasons for calling the function a wavelet. Property 2 implies that most of the energy in $\psi(t)$ is confined to a finite duration. Property 1 is suggestive of a function that is oscillatory or that has a wavy appearance. Thus, in contrast to a sinusoidal function, it is a "small wave" or a wavelet. The two properties are easily satisfied and there is infinity of functions that qualify as mother wavelets.

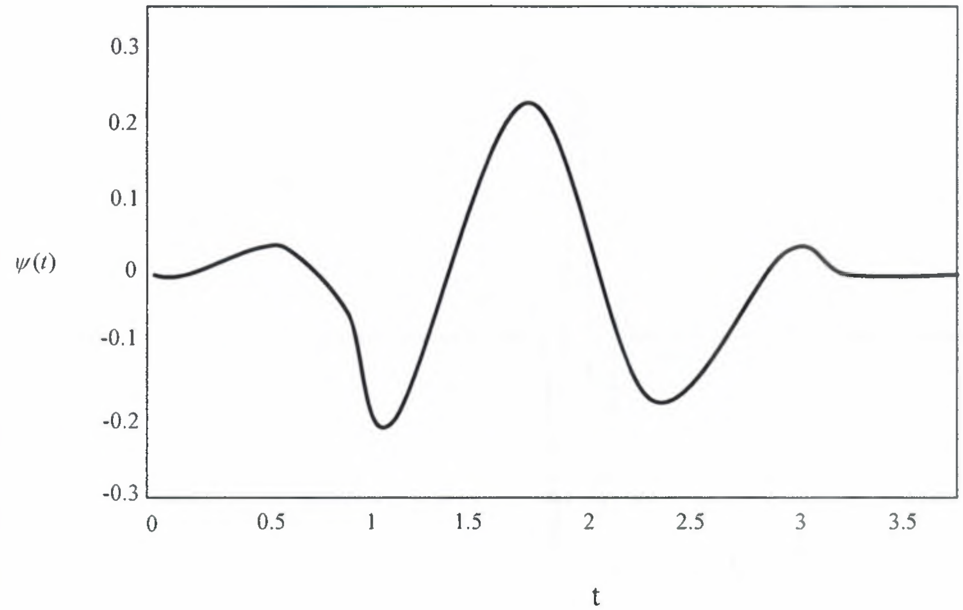

Figure 1.1 The Cubic B-Spline Wavelet.

Figure 1.1 shows the plot of a wavelet called the *cubic B-spline wavelet.* This particular wavelet is supported compactly. In other words, the entire wavelet has a finite duration: $0 \leq t \leq 4$ sec. However, it is possible to have wavelets that are not supported compactly. For example, a Morlet wavelet, which is constructed by modulating a sinusoidal function by a Gaussian function [2], is a wavelet of infinite duration. But most of the energy in this wavelet is confined to a finite interval. For instance, consider the real-value Morlet wavelet:

$$\psi(t) = e^{-t^2} \cos\left(\pi\sqrt{\frac{2}{\ln 2}}t\right) \tag{1.3}$$

Its plot is shown in figure 1.2. More than 99% of the total energy of the function is contained in the interval $|t| \le 2.5\,\text{sec}$.

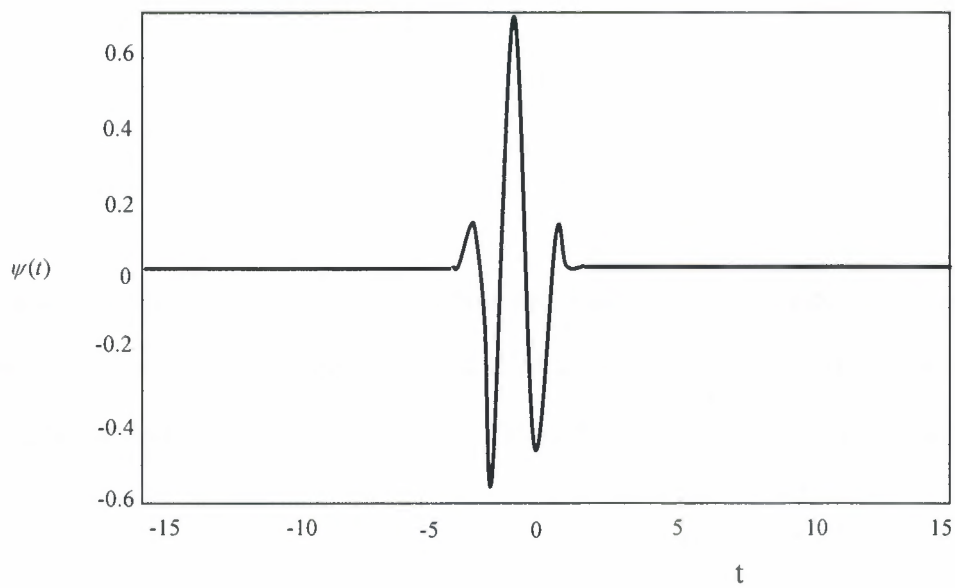

Figure 1.2 The Morlet Wavelet.

1.3 Definition of the CWT

Let f(t) be any square integrable function. The CWT or continuous-time wavelet transform of f(t) with respect to a wavelet $\psi(t)$ is defined as [1].

$$W(a,b) \equiv \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{|a|}}\psi^{*}\left(\frac{t-b}{a}\right)dt \tag{1.4}$$

where a and b are real variables and * denotes complex conjugation. Thus, the wavelet transform is a function of two variables. Observe that both f(t) and $\psi(t)$ belong to $L^2(R)$, the set of square integrable functions, also called the set of energy signals [3]. Equation 1.4 can be rewritten in a more compact form by defining $\psi_{a,b}(t)$ as

$$\psi_{a,b}(t) \equiv \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right) \tag{1.5}$$

Thus, combining equations 1.4 and 1.5 gives,

$$W(a,b) = \int_{-\infty}^{\infty} f(t)\psi^{*}_{a,b}(t)dt \tag{1.6}$$

Notice that

$$\psi_{1,0}(t) = \psi(t) \tag{1.7}$$

The normalizing factor of $1/\sqrt{|a|}$ ensures that the energy stays the same for all *a* and *b*; that is, for all a and b. For any given value of a, the function $\psi_{a,b}(t)$ is a shift of $\psi_{a,0}(t)$ by an amount b along the time axis. Thus, the variable b represents time shift or translation. From

$$\psi_{a,0}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t}{a}\right) \tag{1.8}$$

it follows that $\psi_{a,0}(t)$ is a time-scaled and amplitude-scaled version of $\psi(t)$. Since *a* determines the amount of time scaling or dilation, it is referred to as the *scale* or *dilation variable.* Figure 1.3 shows two dilations of the Morlet wavelet. If $a > 1$, there is a stretching of $\psi(t)$ along the time axis, whereas if $0 < a < 1$, there is a contraction of $\psi(t)$. Negative values of *a* result in a time reversal in combination with dilation. Since the CWT is generated using dilates and translates of the single function $\psi(t)$, the wavelet for the transform is referred to as the mother wavelet.

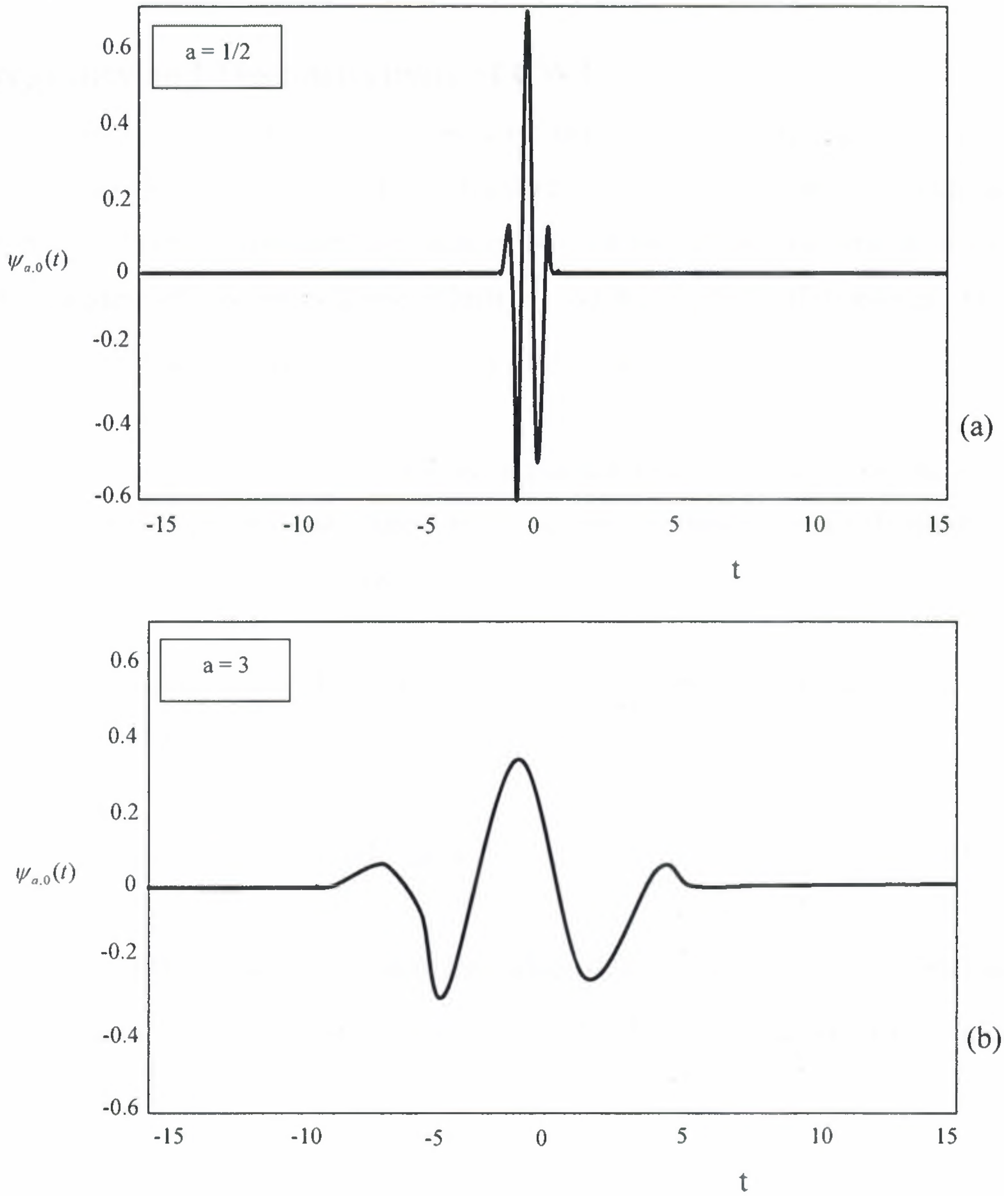

Figure 1.3 A Morlet Wavelet Dilated by Factors of $a = ½$ (a) and $a = 3$ (b).

1.4 Frequency and Time Selectivity of CWT

The CWT offers time and frequency selectivity; that is, it is able to localize events, *both* in time and in frequency. Its ability to localize events in time is easier to establish. The segment of *f(t)* that influences the value of *W(a, b)* for any *(a, b)* is that stretch of *f(t)* that coincides with the interval over which $\psi_{a,b}(t)$ has the bulk of its energy. This windowing effect results in the time selectivity of the CWT.

The frequency selectivity of the CWT is explained using its interpretation as a collection of linear, time-invariant filters with impulse responses that are dilations of the mother wavelet reflected about the time axis.

For the purpose of illustrating the frequency selectivity of the CWT let us choose the *Mexican hat wavelet* [3].

$$\psi(t) = (1 - 2t^2)e^{-t^2} \tag{1.9}$$

which is obtained by taking the second derivative of the negative Gaussian function $-e^{-t^2}/2$. A plot of the wavelet is shown in figure 1.4. Let $\psi(w)$ denotes the Fourier transform of $\psi(t)$; thus

$$\Psi(w) = \int_{-\infty}^{\infty} \psi(t)e^{-jwt} \tag{1.10}$$

From the plot of $|\Psi(w)|^2$ shown in figure 1.5, it is seen that the wavelet is essentially a bandpass function with the Fourier transform magnitude centered at approximately 2 rad/sec.

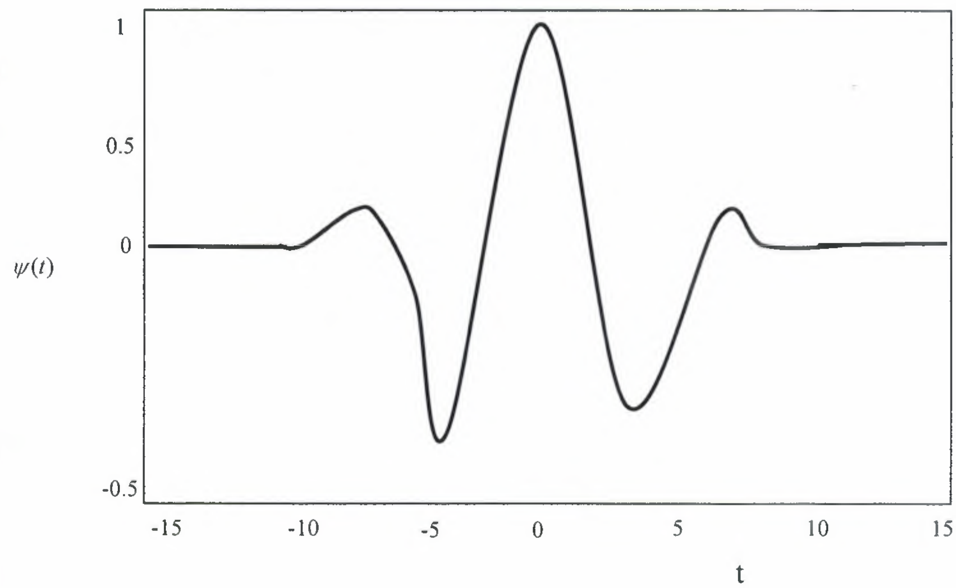

Figure 1.4 The Mexican Hat Wavelet.

Since time reversal does not change the Fourier transform magnitude or a real-valued wavelet, figure 1.5 applies to the frequency domain characterization of $\psi(-t)$ as well. The 3-dB bandwidth, defined as the difference between the two frequencies on either side of the peak at which the squared magnitude of the Fourier transform is exactly half its peak value, is approximately 2 rad/sec, covering the band from 1 rad/sec to 3 rad/sec. This gives a Q-factor (ratio of the center frequency to the 3-dB bandwidth) of roughly 1 for the bandpass function. The Q-factor is invariant with respect to wavelet dilation because

$$\Im[\psi(t/a)] = |a|\,\Psi(aw) \tag{1.11}$$

where $\Im[\,]$ denotes Fourier transformation. The center frequency of $\Im[\psi(t/a)]$ for any a is at $1/|a|$ times the center frequency of the mother wavelet, and its 3-dB bandwidth is also $1/|a|$ times the 3-dB bandwidth of the mother wavelet, thus yielding the same value for the Q-factor as before. Thus, the continuum of filters to which we alluded previously is actually a set of constant-Q bandpass filters. It is this bandpass nature that gives rise to the frequency selectivity of the CWT.

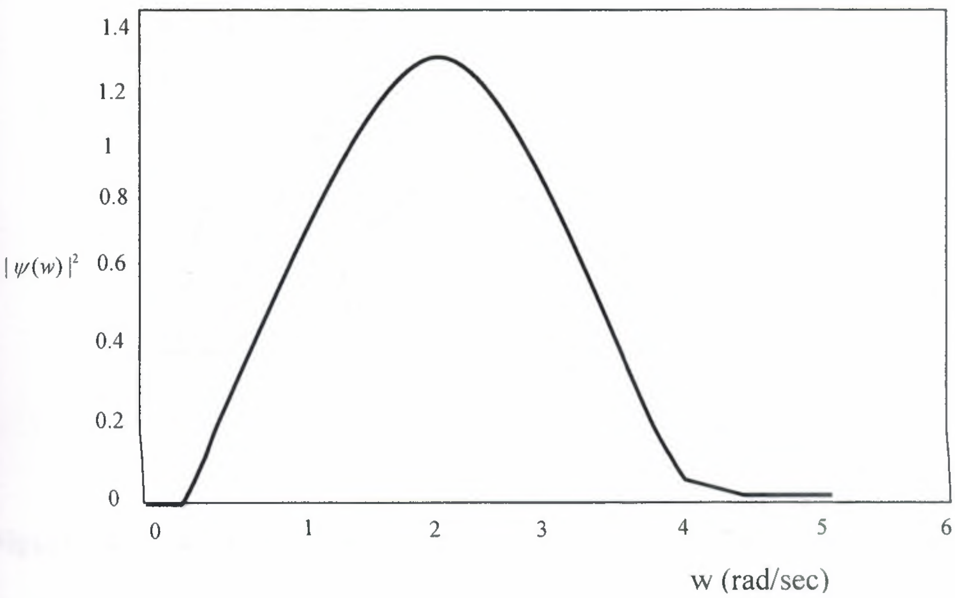

Figure 1.5 Squared Magnitude of the Fourier Transform of the Mexican Hat Wavelet.

The squared frequency response magnitudes for three different values of a for the Mexican hat wavelet are shown in figure 1.6. Observe that as a increases, the frequency response shifts to the lower end of the spectrum with a corresponding shrinking of the bandwidth to maintain the constant Q-factor.

It is not always possible for the CWT to resolve events in frequency. The same holds true for resolving events in time.

Quantitative metrics for time and frequency resolution are based on the duration and bandwidth respectively of the mother wavelet [4].

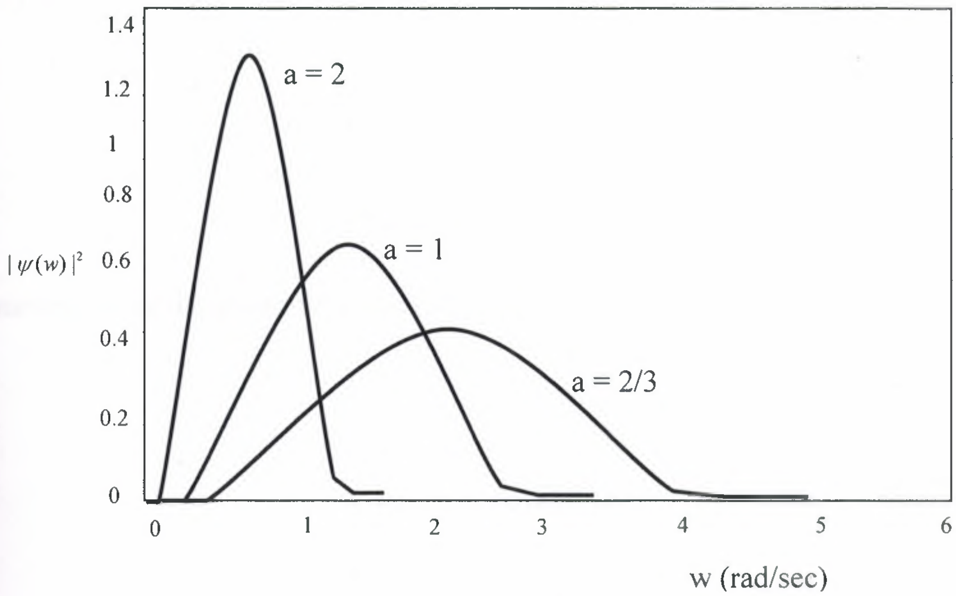

Figure 1.6 Squared Magnitude Frequency Responses of a Mexican Hat Wavelet as a Function of a. Here $\Psi_a(w) = \Im[\psi_{a,0}(-t)]$.

The first moment of a mother wavelet $\psi(t)$ is given by

$$t_0 \equiv \frac{\int_{-\infty}^{\infty} t\,|\psi(t)|^2\,dt}{\int_{-\infty}^{\infty} |\psi(t)|^2\,dt} \tag{1.12}$$

Here $|\psi(t)|^2 \,/\, \int_{-\infty}^{\infty} |\psi(t)|^2\,dt$ acts like a probability density function and, therefore, t_0 provides a measure of where $\psi(w)$, the Fourier transform of $\psi(t)$, along the frequency axis is done by taking its first moment

$$w_0 \equiv \frac{\int_{-\infty}^{\infty} w \,|\, \psi(w) \,|^2 \, dw}{\int_{-\infty}^{\infty} |\, \psi(w) \,|^2 \, dw} \tag{1.13}$$

A measure of the duration of the wavelet or the spread in time is given by [4]

$$\Delta t \equiv \sqrt{\frac{\int_{-\infty}^{\infty} (t - t_0)^2 \,|\, \psi(t) \,|^2 \, dt}{\int_{-\infty}^{\infty} |\, \psi(t) \,|^2 \, dt}} \tag{1.14}$$

The quantity under the radical sign is the second moment of the wavelet about t_0. This measure is known as the root mean square (RMS) duration. The RMS bandwidth of the wavelet is given similarly by

$$\Delta w \equiv \sqrt{\frac{\int_{-\infty}^{\infty} (w - w_0)^2 \,|\, \Psi(w) \,|^2 \, dw}{\int_{-\infty}^{\infty} |\, \Psi(w) \,|^2 \, dw}} \tag{1.15}$$

Fast decays in time and in frequency are required for the wavelet and its Fourier transform to have finite values for the numerator integrals in the previous equations. For functions that do not decay fast enough, alternative measures such as length of interval with substantial energy (say, 95% of total energy) can be used.

Let Δt_ψ and Δw_ψ, be the RMS duration and bandwidth respectively of a mother wavelet $\psi(t)$. The RMS duration of its dilation $\psi_{a,0}(t)$ is then $\Delta t_{\psi}(a) \equiv |a| \Delta t_{\psi}$. The corresponding RMS bandwidth is $\Delta w_{\psi}(a) \equiv \Delta w_{\psi} / |a|$. We have

$$\Delta t_{\psi}(a) \Delta w_{\psi}(a) = \Delta t_{\psi} \Delta w_{\psi} = c_{\psi} \tag{1.16}$$

where c_{ψ} is a constant. Thus, the product of the duration and the bandwidth is invariant to dilation. Equation 1.16 indicates that decreasing $\Delta t_{\psi}(a)$ results in an increase in $\Delta w_{\psi}(a)$ and vice versa. The smaller the value of $\Delta w_{\psi}(a)$, the better the ability of the CWT to resolve events closely spaced in time. Similarly, the smaller the value of $\Delta w_{\psi}(a)$, the better the ability of the CWT to resolve events closely spaced in frequency. At very small values of *a,* the CWT possesses good time resolution (ability to separate very close events in time) because the RMS duration of the corresponding dilated wavelet is small. However, the frequency resolution at such scales is poor because the RMS bandwidth of the dilated wavelet is large. The opposite is true for large values of *a.* Because of the reciprocal relationship between scale and frequency, it also follows that the CWT provides better frequency resolution at the lower end of the frequency spectrum and poorer resolution at the higher end of the frequency spectrum. This ability to provide variable time-frequency resolution is a hallmark of the wavelet transform. It makes the wavelet transform a natural tool in the analysis of signals in which rapidly varying high-frequency components are superimposed on slowly varying low-frequency components such as certain seismic signals and music compositions.

The variation of time and frequency resolution as a function of *a* is shown graphically using *time-frequency cells,* figure 1.7. The center of each rectangle, obtained from the first moments in time and frequency, indicates the position of the wavelet in time and frequency, whereas the rectangle itself bounds the spread of the wavelet in time and frequency as characterized by the corresponding second moments. Given an event localized at some *(T, Ω)* in the time-frequency plane, for any other event to be resolvable it should be localized in the time-frequency plane outside the bounding rectangle of the time-frequency cell at *(T, Ω)*. An alternative view is that associated with a time-frequency cell centered at *(T, Ω)*. There is an ambiguity in localizing events in time and frequency to within its bounding rectangle. It follows from Equation (1.16) that the area of each bounding rectangle is c_{ψ}. Because a decrease in the uncertainty of time localization is accompanied by an increase in the uncertainty of frequency localization (in other words, when the in time it is wider in frequency) and vice versa, it is not possible to reduce simultaneously the uncertainty in both dimensions. For this reason it is called the uncertainty principle governing time-frequency resolution.

The constant c_{ψ} in Equation (1.16) is a function of the wavelet used. A wavelet with a smaller value of c_{ψ} provides better simultaneous localization in the time-frequency plane than one with a larger value. This follows from Equation (1.16). How small can one make the time-bandwidth product of a function? It turns out that the smallest time-bandwidth product is associated with the Gaussian function

$$f(t) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{t^2}{2\sigma^2}} \tag{1.17}$$

And it is integration equal to ½ [5]. Therefore the uncertainty principle can be restarted as

$$\Delta t_{\psi}(a) \Delta w_{\psi}(a) \geq 1/2 \tag{1.18}$$

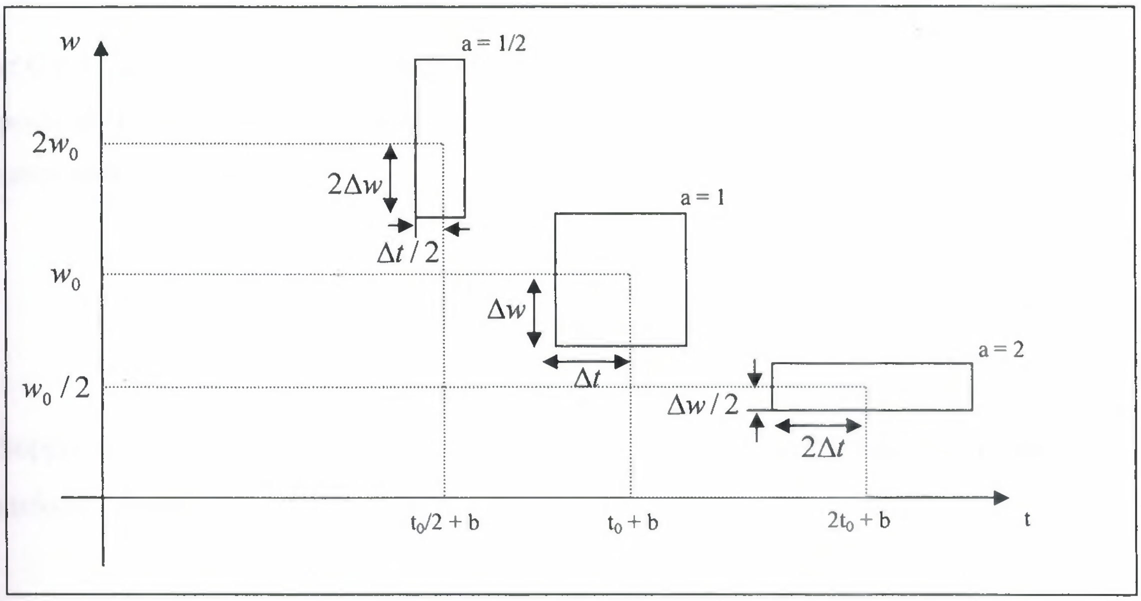

Figure 1.7 Time-Frequency Cells for $\psi_{a,b}(t)$ shown for a fixed b and three different values of a. the mother wavelet and its Fourier Transform are centered at t_0 and w_0 respectively.

1.5 Discrete Wavelet Transform

We saw in the previous sections that the CWT maps a one-dimensional function *f (t)* to a function *W(a, b)* of two continuous real variables *a* and *b,* which are the wavelet dilation and translation respectively. The region of support of *W(a, b)* is defined as the set of ordered pairs *(a, b)* for which $W(a, b) \neq 0$. In principle, the region of support of a CWT is unbounded; that is, it can be the entire plane defined by R^2, the set of all ordered real pairs. In this chapter we introduces a type of wavelet representation that has assumed considerable practical significance because of its link to digital filtering [6]. We take the approach of developing the underlying ideas entirely through an example involving the *Haar wavelet.*

The CWT provides a redundant representation of the signal in the sense that the entire support of *W (a, b)* need not be used to recover *f (t)*. Consider the function *f (t)* with the Fourier transform given by

$$F(w) = \begin{cases} 1 & 2 \leq |\omega| \leq 3 \\ 0 & otherwise \end{cases} \tag{1.19}$$

Suppose we obtain the CWT of *f(t)* using the wavelet $\psi(t)$ with the Fourier transform given by

$$\Psi(\omega) = \begin{cases} 1 & 1 \leq |\omega| \leq 4 \\ 0 & otherwise \end{cases} \tag{1.20}$$

Let *W (a, b)* be the resulting CWT and $W_a(w)$ its Fourier transform in the *b* variable so that

$$W_a(\omega) \equiv \int_{b=-\infty}^{\infty} W(a,b) e^{-jwb} db \tag{1.21}$$

Because $\Im[\psi_{a,0}(t)] = \sqrt{|a|}\Psi(a\omega)$, we have

$$W_a(\omega) = \sqrt{|a|} F(\omega)\Psi(a\omega) \tag{1.22}$$

We now use Equation (1.22) to derive the region of support of *W(a, b)*. Figure 1.8 (a) shows a plot of *F (w)* and figure 1.8 (b) shows a plot of $\Psi(w)$. From the fact that the support of, $\Psi(w)$ is $1 \leq |w| \leq 4$, it follows that the support of $\Psi(aw)$ for a given *a* is $1/|a| \leq |w| \leq 4/|a|$. Comparing this support with that of *F (w),* we find that the two overlap only over the range $1/3 \leq |a| \leq 2$, which is illustrated in figure 1.9. Thus the support of this CWT is restricted to that portion of R^2 for which $1/3 \leq |a| \leq 2$. However, we also see from figure 1.9 that for $1/2 \leq |a| \leq 4/3$, the support of *F (w)* is contained entirely within that of $\Psi(aw)$, and therefore

$$W_a(w) = \sqrt{|a|}F(w) \quad 1/2 \leq |a| \leq 4/3 \tag{1.23}$$

From Equation (1.23) it is seen that *F(w)* can be obtained from $W_a(w)$ or, equivalently, *f(t)* can be obtained from *W(a, b)* by a simple amplitude scaling of the latter for any fixed value of a in the range $1/2 \leq |a| \leq 4/3$ as $F(w) = W_a(w)/\sqrt{a}$. This indicates that it is not necessary to use the entire support of *W(a, b)* to reconstruct *f(t).* We have used a specific function and a specific wavelet to illustrate how redundancy arises in a CWT. However, it is possible to generalize the result to any CWT [2].

In this chapter we introduce a type of nonredundant wavelet representation. In particular we look into a representation of the form

$$f(t) = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} d(k,l) 2^{-k/2} \psi(2^{-k} t - l) \tag{1.24}$$

which involves a continuum of dilations and translations. The dilation takes values of the form $a = 2^k$ where k is an integer. At any dilation 2^k, the translation parameter takes values of the form $2^k l$ where l is again an integer. The values $d(k,l)$ are related to values of the wavelet transform $W(a, b) = W[f(t)]$ at $a = 2^k$ and $b = 2^k l$, This corresponds to sampling the coordinates *(a, b)* on a grid such as the one shown in figure 1.10.

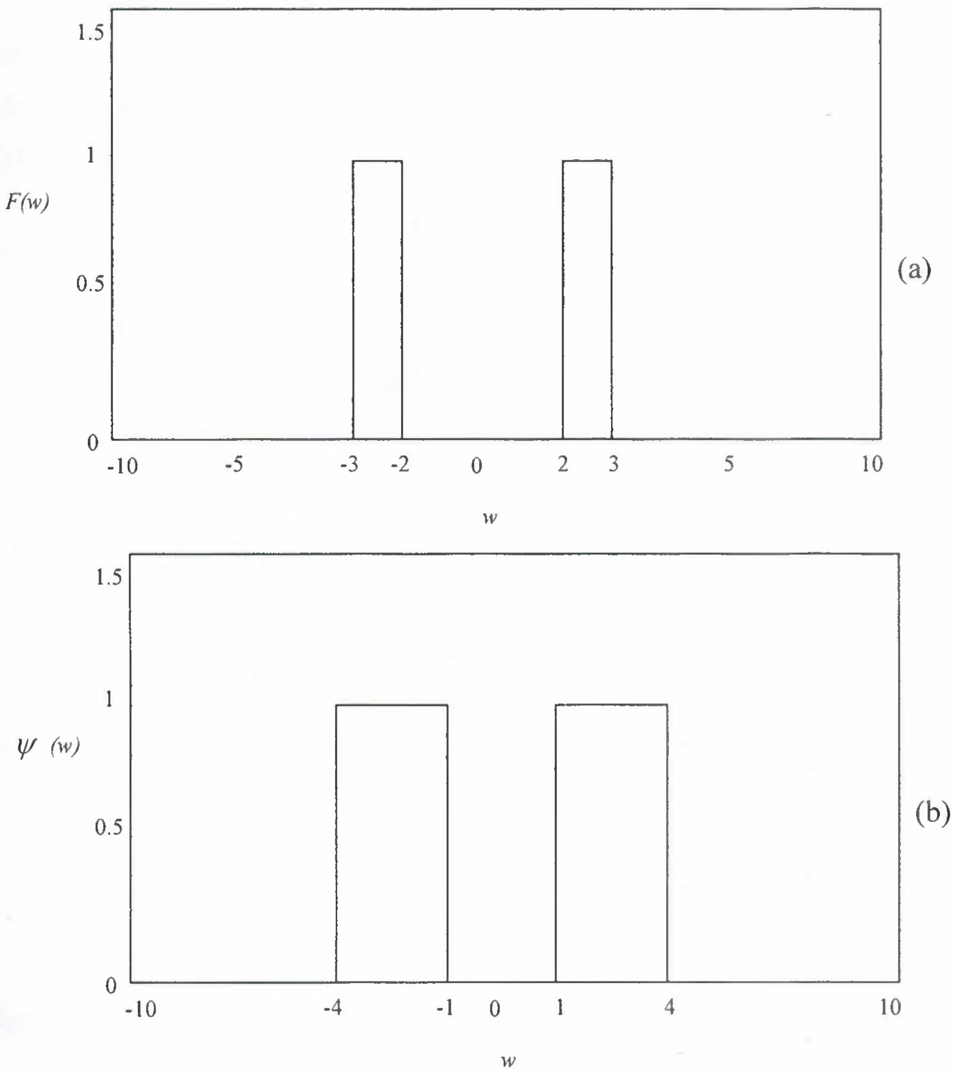

Figure 1.8 Fourier Transform of a Signal (a) and a Wavelet (b).

This process is called *dyadic sampling* because consecutive values of the discrete scales as well as the corresponding sampling intervals differ by a factor of two. The two-dimensional sequence $d\ (k,l)$ is commonly referred to as the *discrete wavelet transform* (DW'T) of $f(t)$ [6]. Observe that the DWT is still the transform of a continuous-time signal. The discretization is only in the a and b variables. In this sense it is analogous to the Fourier series, which involves a discrete frequency domain representation of a (periodic) continuous-time signal. For this reason the DWT has also been referred to as a continuous-time wavelet series [7]. The description and derivation of the DWT is developed here entirely through an example involving the Haar wavelet.

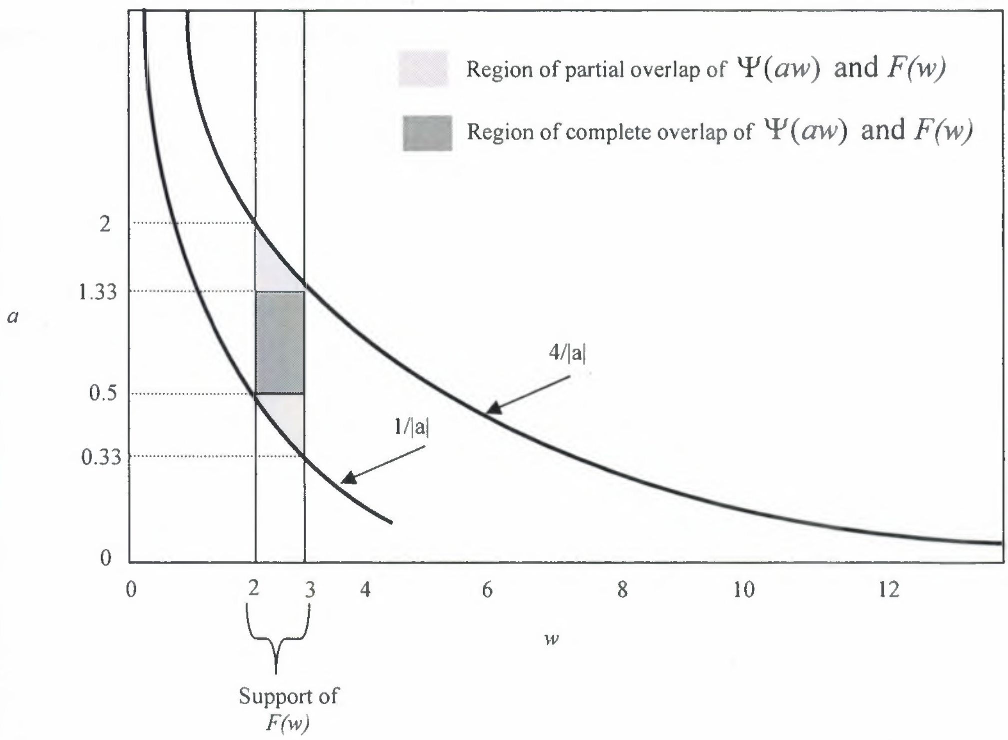

Figure 1.9 Overlap of Fourier Transforms of the Signal and Dilated Wavelet

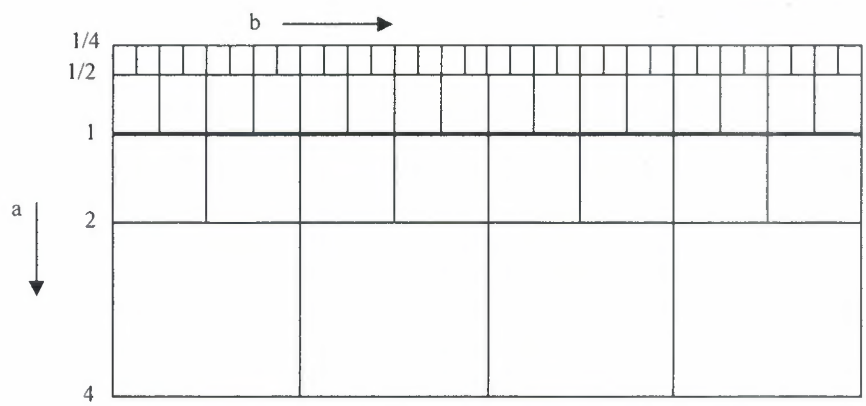

Figure 1.10 The Time-Frequency Cells that Correspond to Dyadic Sampling

1.6Summary

In this chapter were considered Continuous-Time Wavelet, and their time and frequency domain and Discrete-Time Wavelet selectively, and in the next chapter, we will discuss the Neural Networks and its details.

2. ARTIFICIAL NEURAL NETWORKS

2.1 Overview

This chapter presents an overview of neural networks, its history, simple structure, biological analogy and the Backpropagation algorithm.

2.2 Introduction to ANN

An Artificial Neural Network (ANN) is an information-processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true for ANNs as well.

Neural networks go by many aliases. Although by no means synonyms the names listed in figure 2.1 below.

- Parallel distributed processing models
- Connectivist /connectionism models
- Adaptive systems
- Self-organizing systems
- Neurocomputing
- Neuromorphic systems

Figure 2.1 Neural Network Aliases

All refer to this new form of information processing; we shall refer to some of these terms again when we talk about implementations and models. In general though we will continue to use the words "neural networks" to mean the broad class of artificial neural systems. This appears to be the one most commonly used.

2.3 Analogy to the Brain

The human nervous system may be viewed as a three stage system, as depicted in the block diagram of the block diagram representation of the nervous system.

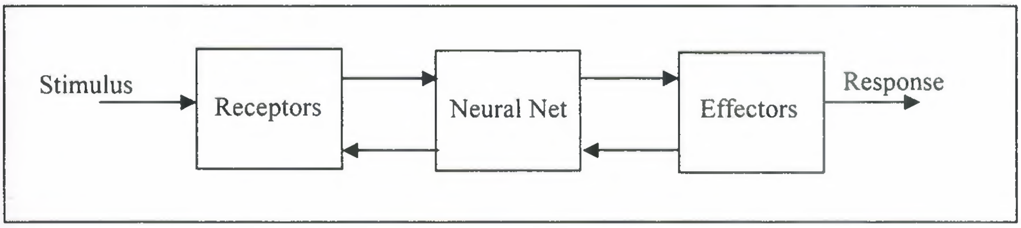

Figure 2.2 Block Diagram of the Nervous System.

Central to the system is the brain, represented by the neural (nerve) network which continually receives information, perceives if, and makes appropriate decisions. Two sets of arrows are shown in the block diagram. Those pointing from left to right indicate the forward transmission of information-bearing signals through the system. The receptors convert stimuli from the human body or the external environment into electrical impulses which convey information to the neural network (brain). The effectors convert electrical impulses by the neural network into discernible responses as system outputs.

2.3.1 Artificial Neuron

This chapter starts by copying the simplest element, the neuron call our artificial neuron a processing element or PE for short. The word node is also used for this simple building block, which is represented by circle in the figure 2.3 "a single mode or processing element PE or Artificial Neuron". The PE handles several basic functions: (1) Evaluates the input signals and determines the strength of each one, Calculates the total for the combined input signals and compare that total to some threshold level, and (3) Determines what the output should be.

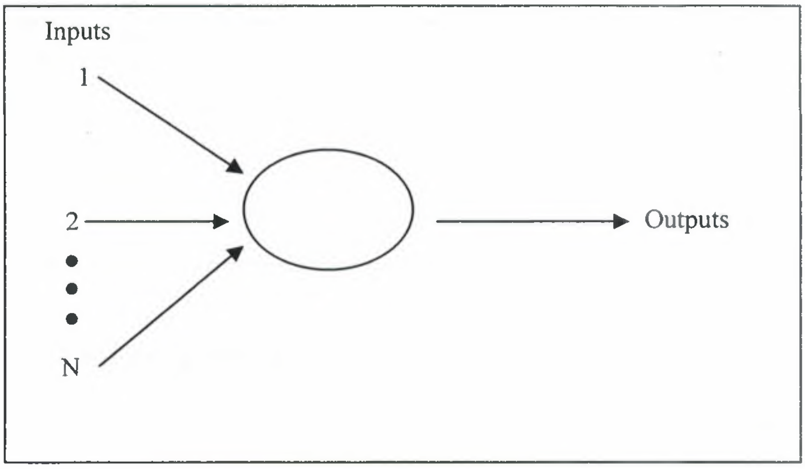

Figure 2.3 Artificial Neuron

Input and Output: Just as there are many inputs (stimulation levels) to a neuron there should be many input signals to our PE. All of them should come into our PE simultaneously. In response a neuron either "fires" or "doesn't fire" depending on some threshold level. The PE will be allowed a single output signal just as is present in a biological neuron. There are many inputs and only one output.

Weighting Factors: Each input will be given a relative weighting which will affect the impact of that input. In figure 2.4, "a single mode or processing element PE or Artificial Neuron" with weighted inputs.
This is something like the varying synaptic strengths of the biological neurons. Some inputs are more important than others in the way that they combine to produce an impulse.

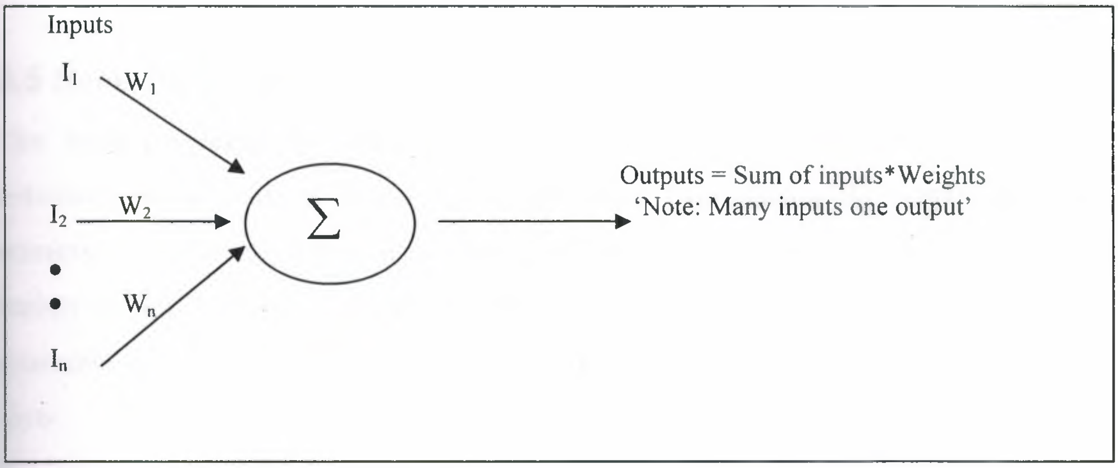

Figure 2.4 Single Mode Artificial Neuron

2.4 Model of a Neuron

The neuron is the basic processor in neural networks. Each neuron has one output, which generally related to the state of the neuron, its activation, which may fan out to several other neurons. Each neuron receives several inputs over these connections, called synapses. The inputs are the activations of the neuron. This is computed by applying a threshold function to this product. An abstract model of the neuron is shown in figure 2.5.

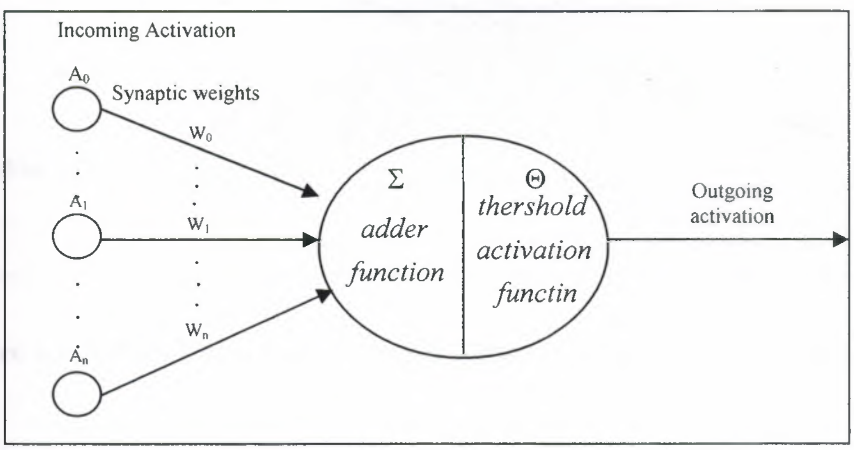

Figure 2.5 Diagram of Abstract Neuron Model.

2.5 Back-Propagation

The **back-propagation** learning algorithm works on multiplayer feed-forward networks, using gradient descent in weight space to minimize the output error. It converges to a locally optimal solution, and has been used with some success in a variety of applications. As with all hill-climbing techniques, however, there is no guarantee that it will find a global solution. Furthermore, its converge is often very slow.

2.5.1 Back-Propagation Learning

Suppose we want to construct a network for the restaurant problem. So we will try a two-layer network. We have ten attributes describing each example, so we will need ten input units. In figure 2.6, we show a network with four hidden nits. This turns out to be about right for this problem.

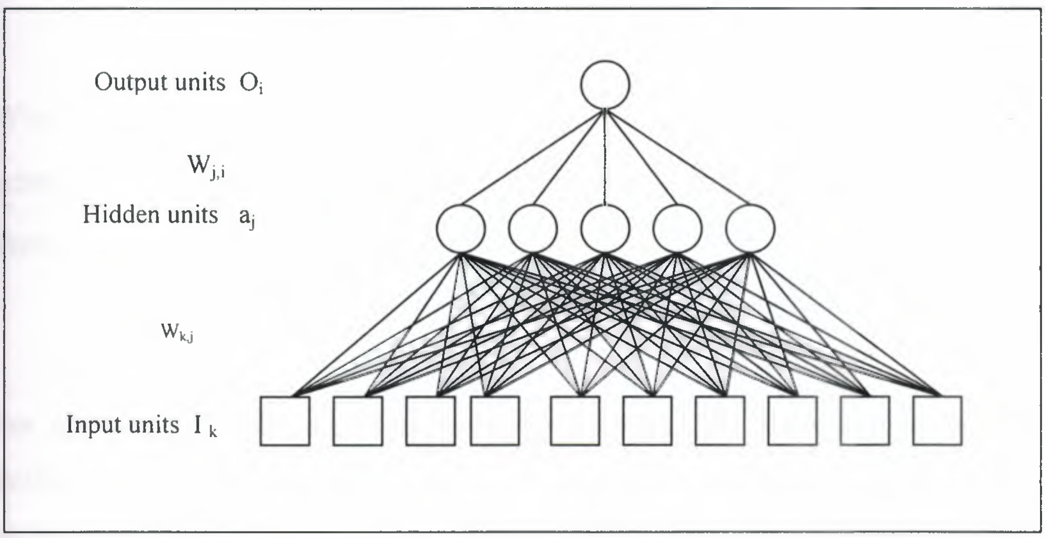

Figure 2.6 A Two Layer Feed Forward Network for the Restaurant Problem.

Example inputs are presented to the network, and if the network computes an output vector that matches the target, nothing is done. If there is an error (a difference between the output and target), then weights are adjusted to reduce this error. The trick is to assess the blame for an error and divide it among the contributing weights. In Perceptrons, this is easy, because there is only one weight connecting each input and output. But in multiplayer networks, there are many weights connecting each input to an output and each of these weights contributes to more than one output.

The back-propagation algorithm is a sensible approach to dividing the contribution of each weight. As in the Perceptron Learning Algorithm, we try to minimize the error between each target output and the output actually computed by the network. At the output layer the weight update rule is very similar to the rule for the perceptron. However, there are two differences. The activation of the hidden unit a_j is used instead of the input value; and the rule contains a term for the gradient of the activation function. If Err_i is the error (T_i-O_i) at the output node, then the weight update rule for the link from unit j to unit i is

$$W_{ji} \leftarrow W_{ji} + \alpha \times Err_i \times g'(in_i) \tag{2.1}$$

Where g' is the derivative of the activation g will find it convenient to define a new error term Δ_i which for output node is defined as $\Delta_i = Err_i g'(in_i)$. The update rule then becomes:

$$W_{ji} \leftarrow W_{ji} + \alpha \times a_j \times \Delta_i \tag{2.2}$$

For updating the connections between the input and the hidden units, we need to define a quantity analogous to the error term for output node. The propagation rule so the following:

$$\Delta_j = g'(in_j)\sum_i W_{ji}\Delta_i \tag{2.3}$$

Now the weight update rule for the weights between the inputs and the hidden layer is almost identical to the update rule for the output layer.

$$W_{kj} \leftarrow W_{kj} + \alpha \times I_k \times \Delta_j \tag{2.4}$$

Function Back-Prop-UPDATE (network, examples, α) returns a network with modified weights.
Inputs: network, a multiplayer network
Examples, asset of input/output pairs α, the learning rate.
Repeat
For each e **in** example **do**
$O \leftarrow TUN - NETWORK(network, I^e)$
$Err^e \leftarrow T^e - O$
$W_{j,i} \leftarrow W_{j,i} + \alpha \times a_j \times Err_i^e \times g'(in_i)$
for each subsequent layer **in** network **do**
$\Delta_j \leftarrow g'(in_j) \sum_i W_{j,i} \Delta_j$
$W_{k,j} \leftarrow W_{k,j} + \alpha \times I_k \times \Delta_j$
end
end
until network has converged
return network

Figure 2.7 Back Propagation Algorithm for Updating Weights in a Multiplayer Network

Back-propagation provides a way of dividing the calculation of the gradient among the unit so the change in each weight can be calculated by the unit to which the weight is attached using only local information.

We use the sum of squared errors over the output values:

$$E = \frac{1}{2} \sum_i (T_i - O_i)^2 \tag{2.5}$$

The key insight again is that the output values O_i are a function of the weights for general two-layer network, we can write:

$$E(W) = \frac{1}{2} \sum_i (T_i - g(\sum_j W_{j,i} a_j))^2 \tag{2.6}$$

$$E(W) = \frac{1}{2}\sum_i (T_i - g(\sum_j W_{j,i} g(\sum W_{k,j} I_k)))^2 \tag{2.7}$$

2.6 Activation Functions

This threshold function is generally some form of nonlinear function. One simple nonlinear function that is appropriate for discrete neural nets is the step function. One variant of the step function is:

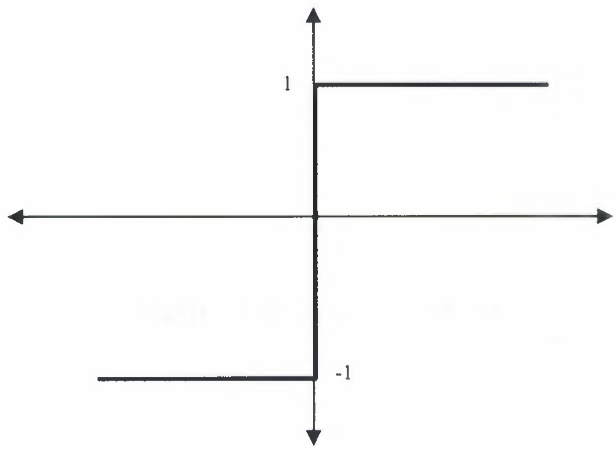

Figure 2.8 Hard Activation Functions

$$f(x) = \begin{cases} 1 & x > 0 \\ f'(x) & x = 0 \\ -1 & x < 0 \end{cases} \tag{2.8}$$

where f' (x) refers to the previous value of f(x) (that is the activation of the neuron will not change) and x is the summation (over all the incoming neurons) of the product of the incoming neuron's activation, and the connection:

$$X = \sum_{i=0}^{n} A_i w_i \tag{2.9}$$

The number of incoming neurons, is A the vector of incoming neurons and w is the vector of synaptic weights connecting the incoming neurons to the neurons we are examining. One more appropriate to analog is the sigmoid, or squashing, function; an example is the logistic functions illustrated in figure 2.9.

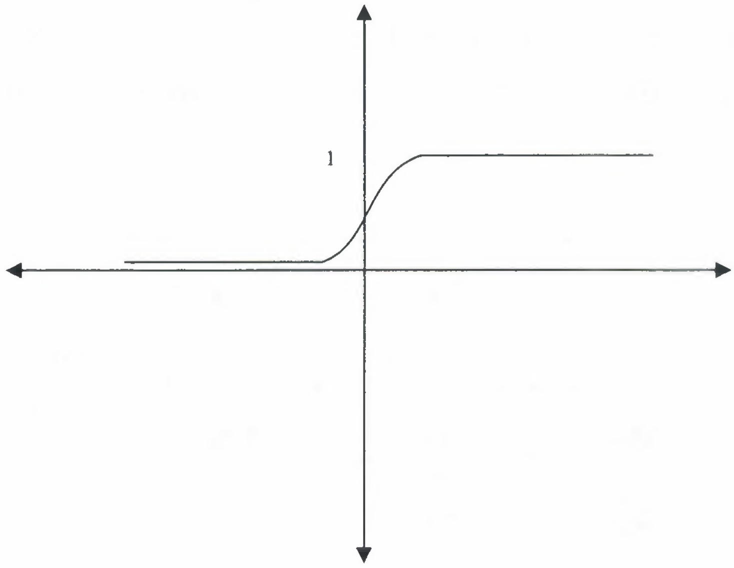

Figure 2.9 Sigmoid Functions

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.10}$$

Another popular alternative is:

$$f(x) = \tanh(x) \tag{2.11}$$

The most important characteristic of our activation function is that it is nonlinear. If we wish to use activation function as a multiplayer network, the activation function must be nonlinear, or the computational will be equivalent to a single-layer network.

2.7 Backpropagation Model

Backpropagation of errors is a relatively generic concept. The Backpropagation model is applicable to a wide class of problems. It is certainly the predominant supervised training algorithm. Supervised learning implies that we must have a set of good pattern associations to train with. The backpropagation model presented in figure 2.10.

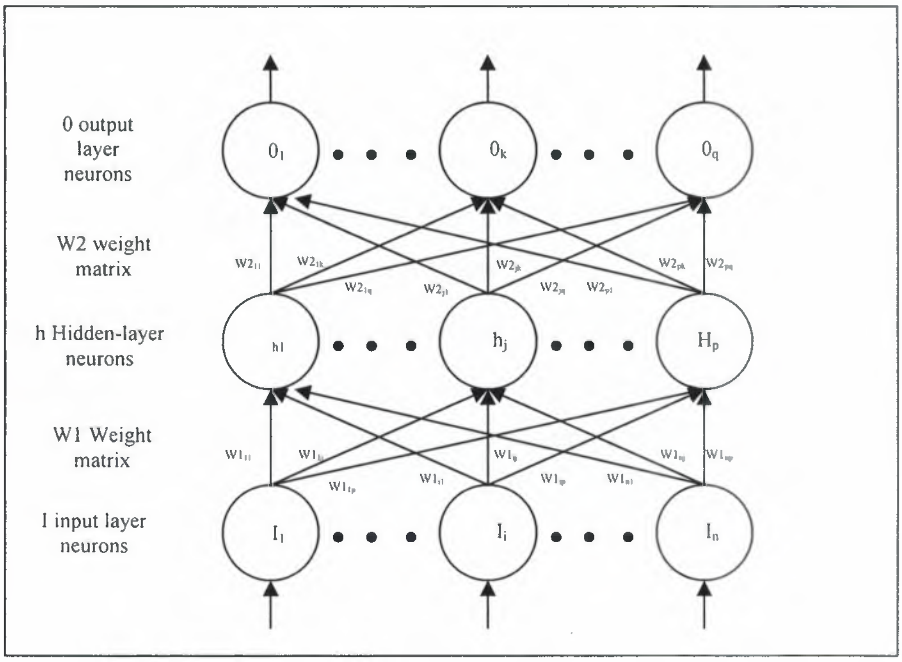

Figure 2.10 Diagram of Backpropagation Topology.

It has three layers of neurons: an input layer, a hidden layer, and an output layer. There are two layers of synaptic weights. There is a learning rate term, α in the subsequent formulas indicating how much of the weight changed to effect on each pass this is typically a number between 0 and 1. There is a momentum term Θ indicating how much a previous weight change should influence the current weight change. There is also a term indicating within what tolernce we can accept an output as good.

2.7.1 Back Propagation Algorithm

Assign random values between –1 and +1 to the weights between the input and hidden layers, the weights between the hidden and output layers, and the thershold for the hidden layer and output layer neurnos train the network by preforming the following procedure for all pattern pairs:

Forward Pass.

1. Computer the hidden layer neuron activations:

$$h=F(iW1) \tag{2.12}$$

   where h is the vector of hidden layer neurons i is the vector of input layer neurons and W1 the weight matrix between the input and hidden layers.

2. Compute the output layer neuron activation:

$$O=F(hW2) \tag{2.13}$$

   where O represents the output layer, h the hidden layer, W2 the matrix of synapses connecting the hidden and output layers, and FO is a sigmoid activation function we will use the logistic function:

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.14}$$

   Backward Pass.

3. Compute the output layer error (the difference between the target and the observed output):

$$d = O(1-O)(O-t) \tag{2.15}$$

   where d is the vector of errors for each output neuron, O is the output layer, and t is the target correct activation of the output layer.

4. Compute the hidden layer error:

$$e = h(1-h)W2d \tag{2.16}$$

   where e is the vector of errors for each hidden layer neuron.

5. Adjust the weights for the second layer of synapses:

$$W2 = W2 + \Delta W2 \tag{2.17}$$

where $\Delta W2$ is a matrix representing the change in matrix $W2$. It is computed as follows:

$$\Delta W2_t = \alpha hd + \Theta \Delta W2_{t-1} \tag{2.18}$$

where α is the learning rate, and Θ is the momentum factor used to allow the previous weight change to influence the weight change in this time period. This does not mean that time is somehow incorporated into the mode. It means only that a weight adjustment has been made. This could also be called a cycle.

6. Adjust the weights for the first layer of synapses:

$$W1 = W1 + W1_t \tag{2.19}$$

where

$$W1_t = \alpha ie + \Theta \Delta W1_{t-1} \tag{2.20}$$

Repeat step 1 to 6 on all pattern pairs until the output layer error (vector d) is within the specified tolerance for each pattern and for each neuron.

Recall:

Present this input to the input layer of neurons of our backpropagation net:

- Compute the hidden layer activation:

$$h = F(W1i) \tag{2.21}$$

- Computer the output layer:

$$O = F(W2h) \tag{2.22}$$

The vector O is our recalled pattern.

2.7.2 Strengths and Weaknesses

The Back Propagation Network has the ability to learn any arbitrarily complex nonlinear mapping this is due to the introduction of the hidden layer. It also has a capacity much greater than the dimensionality of its input and output layers as we will see later. This is not true of all neural net models.

However Backpropagation can involve extremely long and potentially infinite training time. If you have a strong relationship between input and outputs and you are willing to accept results within a relatively broad time, your training time may be reasonable.

2.8 Summary

In this chapter the followings were discussed Perceptron Algorithm, the Backpropagation algorithm and their models, and the Activation function, and in the next chapter we will discuss the combination of Neuro-Wavelet.

3. A WAVELET NEURAL NETWORK

3.1 Overview

This chapter introduces an introduction to Neuro-Wavelet Networks, Signal preprocessing, From Orthogonal Wavelet Decomposition to Wavelet Networks, Static Modeling Using Feedforward Wavelet Networks, Training Feedforward wavelet networks, Initialization of the network parameters, Stopping Conditions for Training, and Dynamic Modeling Using Wavelet Networks.

3.2 Wavelet Neural Networks

A neural network is composed of multiple layers of interconnected nodes with an activation function in each node and weights on the edges or arcs connecting the nodes of the network. The output of each node is a nonlinear function of all its inputs and the network represents an expansion of the unknown nonlinear relationship between inputs, *x*, and outputs, *F* (or *y*), into a space spanned by the functions represented by the activation functions of the network's nodes. Learning is viewed as synthesizing an approximation of a multidimensional function, over a space spanned by the activation functions $0_i(x), i = 1,2,...,m$, i.e.

$$F(x) = \sum_{i=1}^{m} c_i \phi_i(x) \tag{3.1}$$

The approximation error is minimized by adjusting the activation function and network parameters using empirical (experimental) data. Two types of activation functions are commonly used: global and local. Global activation functions are active over a large range of input values and provide a global approximation to the empirical data. Local activation functions are active only in the immediate vicinity of the given input value.

It is well known that functions can be represented as a weighted sum of orthogonal basis functions. Such expansions can be easily represented as neural nets by having the selected basis functions as activation functions in each node, and the coefficients of the expansion as the weights on each output edge. Several classical orthogonal functions, such as sinusoids, Walsh functions, etc., but, unfortunately, most of them are global approximators and suffer, therefore, from the disadvantages of approximation using global functions. What is needed is a set of basis functions which are local and orthogonal. A special class of functions, known as wavelets, possess good localization properties while they are simple orthonormal bases. Thus, they may be employed as the activation functions of a neural network known as the Wavelet Neural Network (WNN). WNNs possess a unique attribute: In addition to forming an orthogonal basis are also capable of explicitly representing the behavior of a function at various resolutions of input variables. The pivotal concept, in the formulation and design of neural networks with wavelets as basis functions, is the multiresolution representation of functions using wavelets. It provides the essential framework for the completely localized and hierarchical training afforded by Wavelet Neural Networks.

Early wavelets were cast as analyzers of one-dimensional signals, such as sound. A major challenge for wavelet theorists to extend the success they have had on one-dimensional signals to more dimensions [12]. As mentioned in chapter 1, the Continuous Wavelet Transform formulas extend to the space L^2 (R^n) by using a separable product wavelet $\psi(x) = \psi_1(x_1)\psi_2(x_2)...\psi_n(x_n)$, and squashing and translation vectors **a** and **b**, to construct

$$\psi_{a,b} = \sqrt{|diag(a)|}\psi(diag(\theta)(x-b)). \tag{3.2}$$

By linearly combining several such wavelets, a multiple-input/single-output neural network is obtained. The basic training algorithm is based on steepest descent.

Rotation matrices are also incorporated for versatility at the expense of training complexity. The authors in [14] demonstrate how a single-input/single-output multilayer perceptron (MLP) can be cast as a truncated wavelet series. A linear combination of three sigmoidal neurons are used to create each wavelet.

The authors in [15], and independently, in [16, 17], arrive at very similar formulations of the wavelet network that are closer to the wavelet expansion than to neural networks. The wavelet parameters are neither adapted as in [13], nor computed from prior Fourier data analysis as in [14], but are taken incrementally from a predefined space-frequency grid of orthogonal wavelets. This approach prescribes learning as a multiresolution, hierarchical procedure, and brings about the possibility of a type of network growth. The essence of multiresolution learning is that a function can be approximated to arbitrary precision by starting with a coarse approximation at some resolution, and then successively adding layers of detail at higher resolutions. Higher resolution wavelet nodes are added to areas of the input space where the existing nodes contribute the largest errors. Elliptic and radial wavelet neural networks (EWNNs and RWNNs) are extension of the intersection between Gaussian radial basis function (RBF) networks and wavelet networks. The local receptivity of RBF networks is adopted to obtain radially symmetric multidimensional wavelets of the form

$$\psi_{a,b}(x) = \sqrt{a^{n}}\,\psi\big(a\|x-b\|\big), \qquad a \geq 0.\ a \geq 0.\ a \tag{3.3}$$

This results in a new class of RBF neural networks referred to as radial WNNs. Whereas RBF networks represent functions in terms of time atoms, the WNN employs time-frequency atoms. To illustrate, let us introduce a basic scalar wavelet that will be recurrently employed, define

$$\mathrm{Costrap}(x) = \cos\left(\frac{3\pi}{2}x\right)\min\left\{\max\left\{\frac{3}{2}(1-|x|,0\right\},1\right\}. \tag{3.4}$$

This function, shown in figure 3.2, consists of two cycles of the cosine function, windowed by a trapezoid that linearly tapers two thirds of the endpoints to zero. Many other wavelets are possible, but highly oscillatory basis functions result in undesirable [18] lack of smoothness in the learned model. Taking $\psi(x) = \cos trap(x)$ and substituting $a = 1, b = 0$ in equation 3.3, induces the two-dimensional basic wavelet. A two-dimensional RWNN implements a linear combination of a family of these surfaces, deployed throughout the plane with various centers, widths, and heights.

The feature of adaptive anisotropy an be accommodated by computing a more general distance between row-vectors **x** and **b**, with wavelets in the form

$$\psi_{A,b}(x) = |A|^{1/4}\psi\left(\sqrt{(x-b)A(x-b)^T}\right), \qquad A \geq 0. \tag{3.5}$$

These types of wavelets constitute the basis for a new class of elliptic basis function networks termed elliptic WNNs. Here, we introduce a symmetric positive semidefinite squashing matrix **A**. With equal scales $1/a^2$ along all coordinates, **A** = $\mathrm{diag}\left(a^2,\dots,a^2\right), |A| = a^{2n}$.

3.2.1 Signal Pre-Processing

Raw data derived from sensors possess signal and measurement noise. This noise can be either random or systematic. The latter type can be filtered to increase the usability of the data. Noise can be further characterized as high-frequency gaussian noise, DC bias, etc. Proper signal processing is applied that this stage to decrease random noise levels. Preprocessing also involves data format conversion, sampling, quantization, digitization, etc. Further pre-processing at the signal level involves the following operations: The first operation is high-pass filtering used to (1) filter out low-frequency components-this is basically a detrending operation allowing for an absolute determination of the true signal peak, and (2) imitate a wavelet as a magnifier. The filter is designed as FIR (finite length) filter using a windowing technique known as the Hamming window and found as a MATLAB function in the commercial MATLAB toolbox.

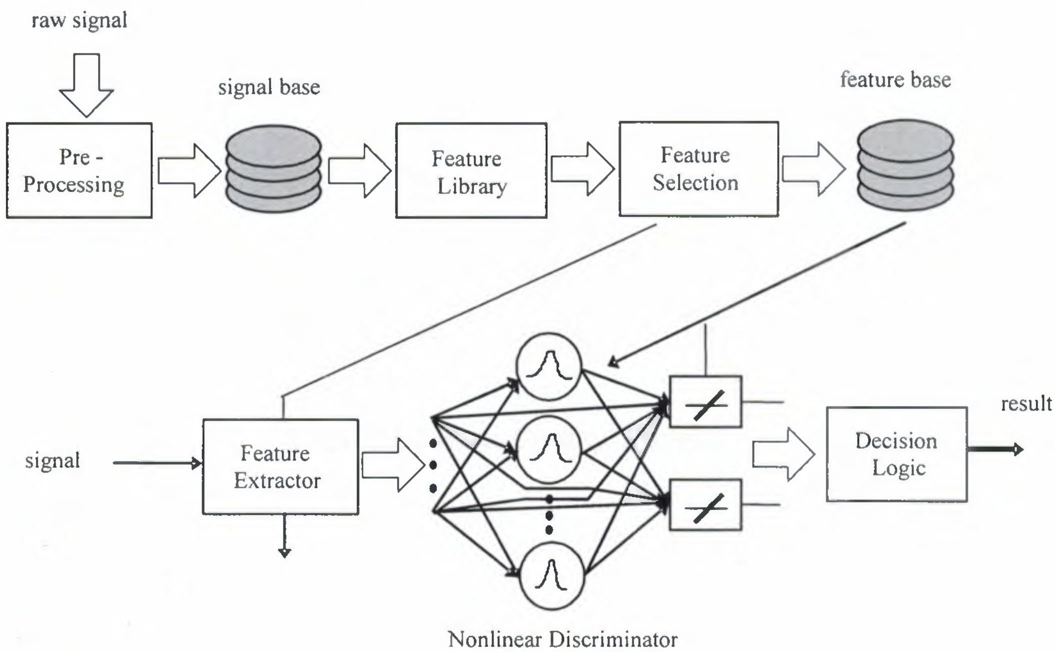

Figure 3.1 A Systematic Identification and Classification Methodology

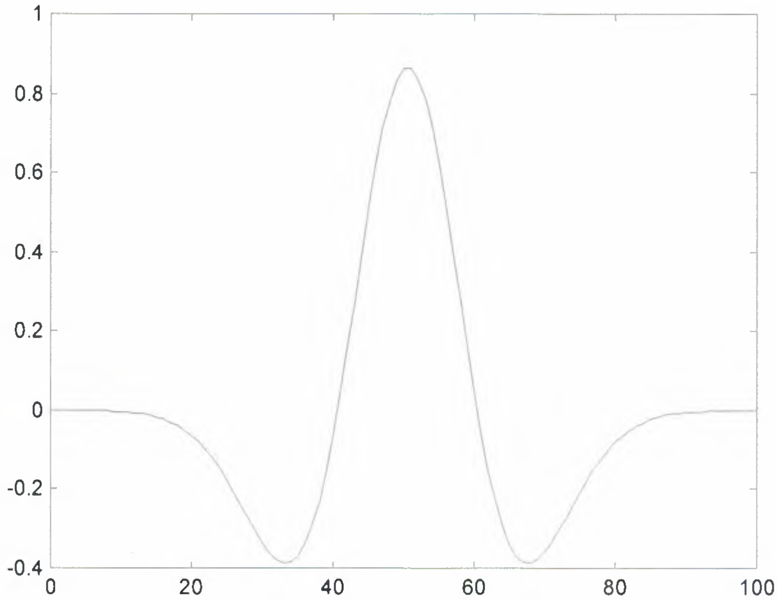

Figure 3.2 Basic Costrap Wavelet

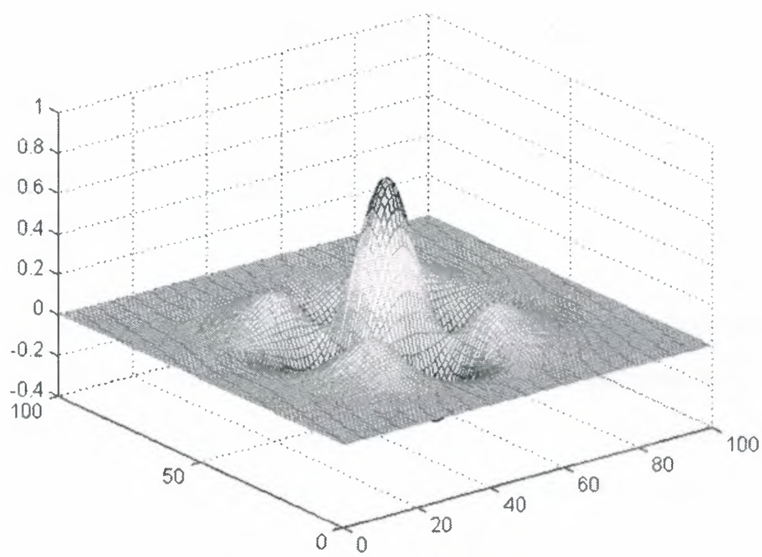

Figure 3.3 Basic Bivariate Costrap Wavelet

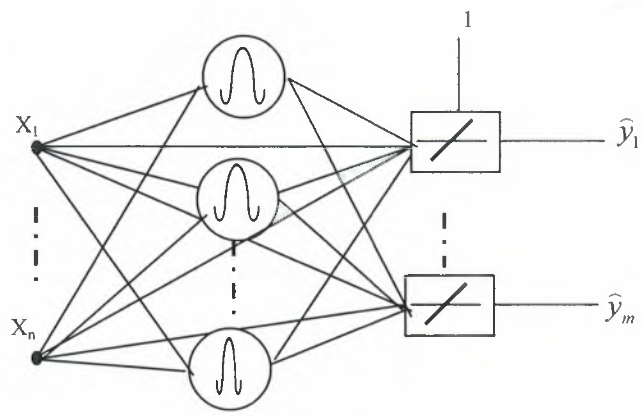

Figure 3.4 Complete Wavelet Neural Network

3.3 From Orthogonal Wavelet Decomposition to Wavelet Networks

The theory of wavelets was first proposed in the field of multiresolution analysis; among others, it has been applied to image and signal processing [16]. A family of wavelets is constructed by translations and dilations performed on a single fixed function called the *mother wavelet.* A wavelet ϕ_j is derived from its mother wavelet ϕ by

$$\phi_j(z) = \phi\left(\frac{x - m_j}{d_j}\right) \tag{3.6}$$

where its translation factor m_j and its dilation factor d_j are real numbers ($d_j > 0$). We are concerned with modeling problems, i.e. with the fitting of a data set by a finite sum of wavelets. There are several ways to determine the wavelets for this purpose:

- From orthogonal wavelet decomposition theory, it is known that, with a suitable choice of ϕ and if m_j and d_j are integers satisfying some conditions, the family $\{\phi_j\}$ forms an orthogonal wavelet basis. A weighted sum of such functions with appropriately chosen m_j and d_j can thus be used; in this way, only the weights have to be computed [18].

- Another way to design a wavelet network is to determine the m_j and d_j according to a space frequency analysis of the data; this leads to a set of wavelets which are not necessarily orthogonal [10, 11].

- Alternatively, one can consider a weighted sum of wavelets functions whose parameters m_j and d_j are adjustable real numbers, which are to be trained together with the weights.

In the latter approach, wavelets are considered as a family of parameterized nonlinear functions which can be used for nonlinear regression; their parameters are estimated through "training".

This chapter introduces training algorithms for *feedback* wavelet networks used for dynamic modeling, which are similar in spirit to training algorithms used for feedback neural networks.

Choice of a mother wavelet

We choose the first derivative of a Gaussian function, $\phi(x) = \pm x \exp\left(\pm \frac{1}{2} x^2\right)$ as a mother wavelet. It may be regarded as a differentiable version of the Haar mother wavelet, just as the sigmoid is a differentiable version of a step function, and it has the universal approximation property [17]. This mother wavelet has also been used in reference [17]. More complex wavelet functions, such as the second derivative of the Gaussian may be used, but they will not be considered here.

The wavelet network.

In the case of a problem with N_i inputs, multidimensional wavelets must be considered. The simplest, most frequent choice is that of separable wavelets, i.e. the product of N_i monodimensional wavelets of each input:

$$\phi_j(x) = \prod_{k=1}^{N_i} \phi(z_{jk}) \quad with \quad z_{jk} = \frac{x_k - m_{jk}}{d_{jk}} \tag{3.7}$$

(3) Can be viewed as a network with N_i inputs, a layer of N_w wavelets of dimension N_i, a bias term, and a linear output neuron. When linear terms are expected to play an important role in the model, it is customary to have additional direct connections from inputs to outputs, since there is no point in using wavelets for reconstructing linear terms. Such a network is shown in figure 3.5.

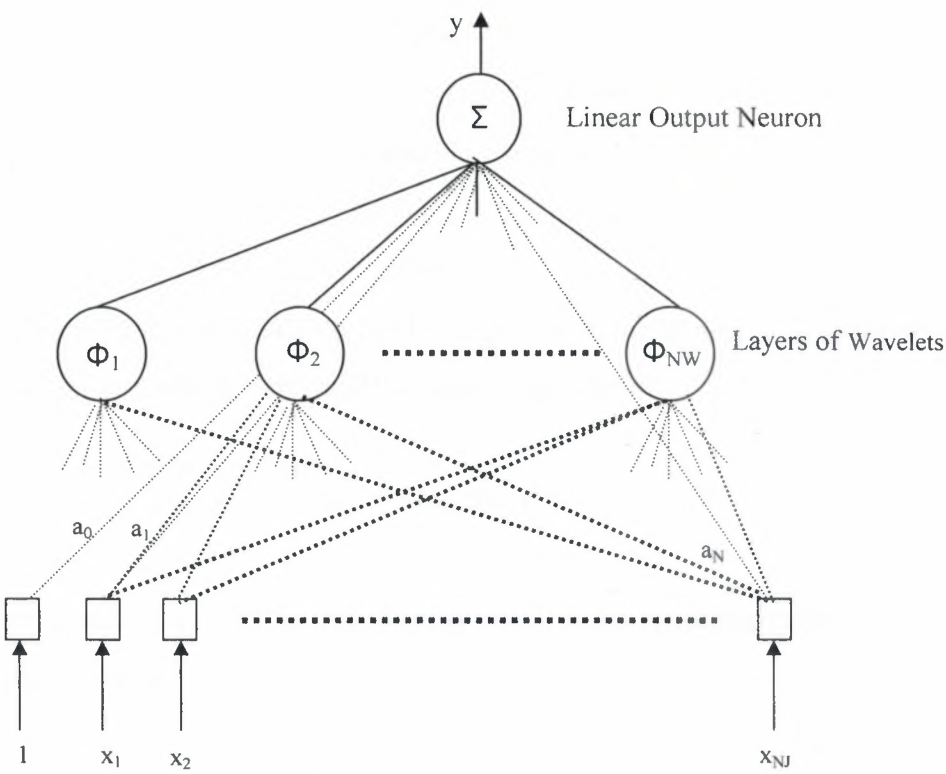

Figure 3.5 A Feedforward Wavelet Network

3.4 Static Modeling Using Feedforward Wavelet Networks

Static modeling with wavelet networks has been investigated by other authors in [17]. We consider a process with N_i inputs and a scalar output y_p. Steady-state measurements of the inputs and outputs of the process build up a training set of N examples $(x^n y_p^n)$ $x^n = \left[x_1^n, \ldots\ldots, x_{n_i}^n\right]^T$ being the input vector for example n and y_p^n the corresponding measured process output. In the domain defined by the training set, the static behavior of the process is assumed to be described by:

$$y_p^n = f(x^n) + w^n \qquad n = 1 \; to \; N \tag{3.8}$$

Where f is an unknown nonlinear function, and wn denotes a set of independent identically distributed random variables with zero mean and variance σ_w^2. We associate the following wavelet network to the assumed in equation (3.8).

$$y^n = \Psi(x^n, \theta) \qquad n = 1 \; to \; N \tag{3.9}$$

Where y^n is the model output value related to example n, the nonlinear function Ψ is given by relation (3.7), and θ is the set of adjustable parameters:

$$\theta = \{m_{jk}, d_{jk}, c_j, a_k, a_0\} \quad with \quad j = 1, \ldots\ldots N_w \; and \; k = 1, \ldots\ldots, N_i \tag{3.10}$$

θ is to be estimated by training so that Ψ approximates the unknown function f on the domain defined by the training set.

3.4.1 Training Feedforward Wavelet Networks

As usual, the training is based on the minimization of the following quadratic cost function:

$$j(\theta)=\frac{1}{2}\sum_{n=1}^{N}(y_p^n-y^n)^2=\frac{1}{2}\sum_{n=1}^{N}(e^n)^2 \tag{3.11}$$

The minimization is performed by iterative gradient-based methods. The partial derivative of the cost functions with respect to θ is:

$$\frac{\partial J}{\partial \theta}=-\sum_{n=1}^{N}e^n\frac{\partial y^n}{\partial \theta} \tag{3.12}$$

where $\frac{\partial y^n}{\partial \theta}$ is a short notation for $\frac{\partial y}{\partial \theta}|_{x=x^n}$ The components of the latter vector are:

- parameter a_0:

$$\frac{\partial y^n}{\partial a_0}=1 \tag{3.13}$$

- direct connection parameters:

$$\frac{\partial y^n}{\partial a_k}=x_k^n \quad k=1,\dots\dots,N_i \tag{3.14}$$

- weights:

$$\frac{\partial y^n}{\partial c_j}=\Phi_j(x^n) \quad j=1,\dots\dots,N_w \tag{3.15}$$

- translations:

$$\frac{\partial y^n}{\partial m_{jk}} = \pm \frac{c_j}{d_{jk}} \frac{\partial \Phi_j}{\partial z_{jk}} |_{x=x^n} \quad k = 1,....,N_i \ and \ j = 1,...,N_w \tag{3.16}$$

with

$$\frac{\partial \Phi_j}{\partial z_{jk}} |_{x=x^n} = \phi(z^n_{j1})\phi(z^n_{j2}).....\phi^{'}(z^n_{jk})....\phi(z^n_{jNi}) \tag{3.17}$$

where $\phi^{'}(z^n_{jk})$ is the value of the derivative of the scalar mother wavelet at point z^n_{jk}

$$\phi^{'}(z^n_{jk}) = \frac{d\phi(z)}{dz} |_{z=z^n_{jk}} \tag{3.18}$$

- dilations:

$$\frac{\partial y^n}{\partial djk} = \pm \frac{c_j}{d_{jk}} z^n_{jk} \frac{\partial \Phi_j}{\partial z_{jk}} |_{x=x^n} \quad k = 1,....,N_i \ and \ j = 1,...,N_w \tag{3.19}$$

At each iteration, the parameters are modified using the gradient (3.8), according to:

$$\Delta\theta = -M \frac{\partial J}{\partial \theta} \tag{3.20}$$

where M is some definite positive matrix ($M = \mu\ Id$, $\mu > 0$ in the case of a simple gradient descent, or $M = \mu H^1$, $\mu > 0$ where H^1 is an approximation, updated iteratively, of the inverse Hessian, for quasi-Newton methods).

3.4.2 Initialization of the Network Parameters

Initializing the wavelet network parameters is an important issue. Similarly to Radial Basis Function networks (and in contrast to neural networks using sigmoidal functions), a random initialization of all the parameters to small values (as usually done with neural networks) is not desirable since this may make some wavelets too local (small dilations) and make the components of the gradient of the cost function very small in areas of interest. In general, one wants to take advantage of the input space domains where the wavelets are not zero. Therefore, we propose an initialization for the mother wavelet

$\phi(x) = \pm x \exp\left(\pm \frac{1}{2} x^2\right)$ based on the input domains defined by the examples of the training sequence. We denote by $[\alpha_k, \beta_k]$ the domain containing the values of the k-th component of the input vectors of the examples. We initialize the vector m of wavelet j at the center of the parallelepiped defined by the N_i intervals $\{[\alpha_k, \beta_k]\}$: $m_{jk} = \frac{1}{2} (\alpha_k + \beta_k)$. The dilation parameters are initialized to the value $0.2(\beta_k - \alpha_k)$ in order to guarantee that the wavelets extend initially over the whole input domain. The choice of the a_k ($k = 1, \ldots, N_i$) and c_j ($j = 1, \ldots, N_w$) is less critical: these parameters are initialized to small random values.

3.4.3 Stopping Conditions for Training

The algorithm is stopped when one of several conditions is satisfied: the Euclidean norm of the gradient, or of the variation of the gradient, or of the variation of the parameters, reaches a lower bound, or the number of iterations reaches a fixed maximum, whichever is satisfied first.

The final performance of the wavelet network model depends on whether: (i) the assumptions made about the model are appropriate, (ii) the training set is large enough, (iii) the family contains a function which is an approximation of f with the desired accuracy in the domain defined by the training set, (iv) an efficient (i.e. second-order) training algorithm is used.

3.5 Dynamic Modeling Using Wavelet Networks

We propose to extend the use of wavelet networks to the dynamic modeling of single-input single-output (SISO) processes. The training set consists of two sequences of length N: the input sequence $\{u(n)\}$ and the measured process output $\{y_p(n)\}$. As in the static case, the aim is to approximate f by a wavelet network.

Depending on the assumptions about the noise, either feedforward or feedback predictors may be required [9]. For example, if it is assumed that the noise acting on the process is state noise i.e. if a Nonlinear AutoRegressive with eXogeneous inputs (NARX, or Equation Error) model

$$y_p(n) = f\big(y_p(n\pm 1), y_p(n\pm 2),\ldots, y_p(n\pm N_s), u(n\pm 1),\ldots., y_p(n\pm N_e)\big) + w_n \qquad (3.21)$$

is assumed to be valid, then *the optimal associated predictor is a feedforward one,* whose inputs are past outputs *of the process* y_p and the external inputs u:

$$y(n) = f\big(y_p(n-1), y_p(n-2),\ldots., y_p(n-N_s), u(n-1),\ldots., u(n-N_e)\big) \qquad (3.22)$$

Where f is a unknown nonlinear function, which is to be approximated by a wavelet network Ψ.

Conversely, if it is assumed that the noise is output noise, i.e. if an Output Error model

$$s(n) = f\big(s(n\pm 1), s(n\pm 2),\ldots., s(n\pm N_s), u(n\pm 1),\ldots u(n\pm N_e)\big) \qquad (3.23)$$

$$y_p(n) = s(n) + w(n) \qquad (3.24)$$

is assumed to be valid, then *the optimal associated predictor is a feedback one,* whose inputs are past outputs *of the model* y and the external inputs u:

$$y(n) = f\big(y(n-1), y(n-2),\ldots., y(n-N_s), u(n-1),\ldots.., u(n-N_e)\big) \qquad (3.25)$$

In the absence of noise, either feedforward or feedback predictors can be used. If the goal is the design of a simulation model, i.e. of a model that can compute the output more than one time step ahead, a feedback predictor should be trained [9].

If all cases, θ is to be estimated so that Ψ approximates the unknown function f on the domain defined by the training set.

We define the copy *n* ($n = 1,\ldots, N$) as the wavelet network configuration giving $y(n)$ at its output in the case of a feedforward predictor, and as the feedforward part of the network canonical form in the case of a feedback predictor [8]. In order to keep the notations equivalent with the previous section we note: $y^n = y(n)$.

3.5.1 Training Feedforward Wavelet Predictors

In this case, the *N* copies are independent, and the training is similar to that of a static model. Therefore, the input vector of copy *n* can be viewed as the vector x^n and $\{y_p(n)\}$ as the process output defined as y_p^n. More precisely, the inputs of copy *n* can be renamed as:

- external inputs: $x_k^n = u(n-k)$ with $k = 1,\ldots, N_e$
- state inputs: $x_k^n = y_p(n-k+N_e)$ with $k = N_e + 1,\ldots, N_e + N_s$

Since the state inputs of the copies are forced to the corresponding desired values, the predictor is said to be trained in a *directed* [8].

3.5.2 Training Feedback Wavelet Predictors

In this case, the *N* copies are not independent: the *N* output values $y^n = y(n)$ of the network may be considered as being computed by a large feedforward network made of *N* cascaded copies of the feedforward part of the canonical form of the feedback network [8]: the state inputs of copy *n* are equal to the state outputs of copy *n*-1. The inputs and outputs of copy *n* are renamed as:

- External inputs: $x_k^n = u(n-k)$ with $k = 1,\ldots, N_e$.
- State inputs: $x_k^n = u(n-k+N_e)$ with $k = N_e + 1, \ldots, N_e + N_s$.
- State outputs: $x_k^n = y(n-k+N_e+N_s+1)$ with $k = N_e + N_s + 1, \ldots, N_e + 2N_s$.

$x_{N_e+N_s+1}^n = y(n) = y^n$ is the n-th value of the output of the network.

$\theta^n = \{m^n_{jk}, d^n_{jk}, c^n_j, a^n_k, a^n_0\}$ With $j = 1, \ldots, Nw$ and $k = 1, \ldots, Ne + Ns$

is the set of parameters of copy *n*. The feedback predictor network and copy *n* are shown in figure 3.6.

Since the state inputs of the first copy only are forced to desired values, the predictor is said to be trained in a *semi-directed* fashion [8], (also known as *backpropagation through time* [15]: the gradient of the cost function is computed by a single backpropagation through the N copies). The gradient of $J(\theta) = \frac{1}{2}\sum_{n=1}^{N}\left(y_p^n - y^n\right)^2 = \frac{1}{2}\sum_{n=1}^{N}\left(e^n\right)^2$ with respect to θ can be expressed as the sum of the gradient with respect to each of the N copies θ^n of θ:

$$\frac{\partial J}{\partial \theta} = \sum_{n=1}^{N}\frac{\partial J}{\partial \theta^n} = \sum_{n=1}^{N}\frac{\partial J}{\partial y^n}\frac{\partial y^n}{\partial \theta^n} \tag{3.26}$$

The analytical expressions of $\frac{\partial y^n}{\partial m^n_{jk}}, \frac{\partial y^n}{\partial d^n_{jk}}, \frac{\partial y^n}{\partial c^n_j}, \frac{\partial y^n}{\partial a^n_k}, \frac{\partial y^n}{\partial a^n_0}$ which are the components of

$\frac{\partial y^n}{\partial \theta^n}$ are identical to those given (without superscript *n* for θ), for the training of feedforward nets.

The set of partial derivatives $\left\{\frac{\partial J}{\partial y^n}\right\}$ can be computed by backpropagation through the feedforward network consisting of the N cascaded copies. We introduce the intermediate variables $\{q^n_k\}$, q^n_k being the partial derivative of $-J$ with respect to x^n_k, the state variable x_k of the n-th copy:

$$q^n_k = -\frac{\partial J}{\partial x^n_k} \tag{3.27}$$

Copy *N*:

- output:

$$q_{out}^{N} = q_{N_e+N_s+1}^{N} = e^{N} \tag{3.28}$$

- other output state variables:

$$q_k^n = 0 \text{ with } k = N_e + N_s + 2, \ldots, N_e + 2N_s. \tag{3.29}$$

- for the *Ns* state inputs :

$$q_k^N = \left(a_k^N + \sum_{j=1}^{N_w} \frac{c_j^N}{d_{jk}^N} \frac{\partial \Phi_j}{\partial z_{jk}^N} \right) q_{out}^N \quad with\ k = N_e + 1, \ldots, N_e + N_s \tag{3.30}$$

Copies n = N-1 to 2:

- output:

$$q_{out}^{N} = e^{n} + q_{N_e+1}^{n+1} \tag{3.31}$$

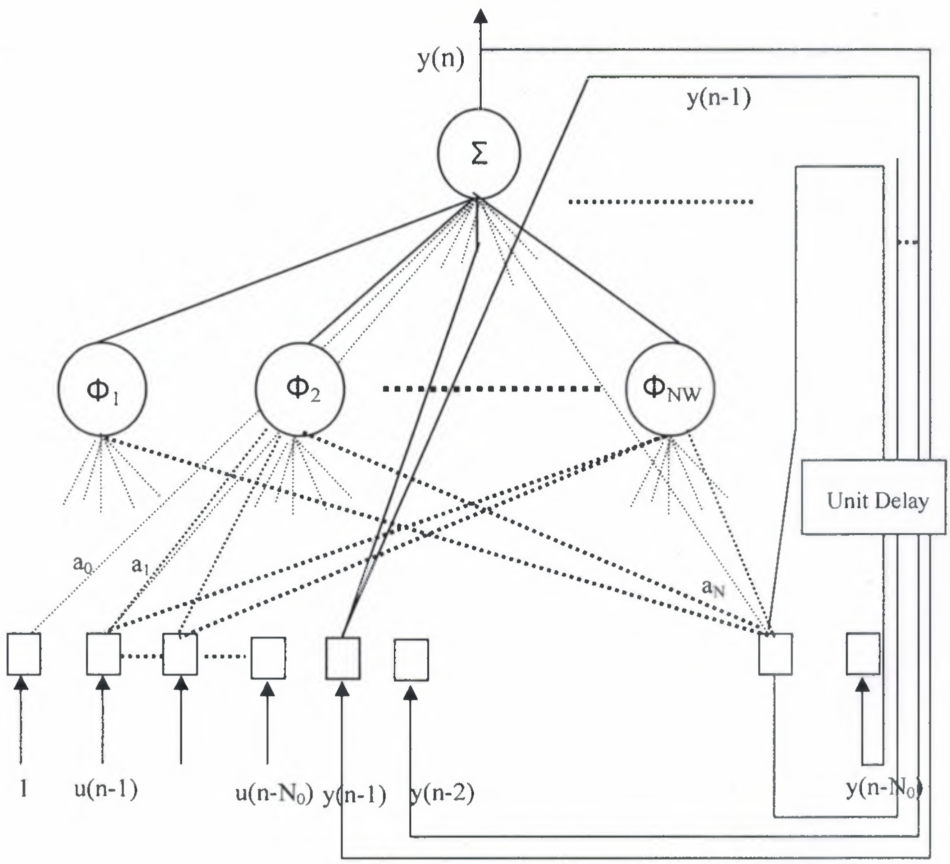

Figure 3.6 Feedback Predictor Network

3.6 Summary

In this chapter, we extend the use of wavelet networks for function approximation to dynamic nonlinear input-output modeling of processes.

The WNN approach offers additional advantages in terms of learning and optimization functions that may be carried out off-line or on-line. Furthermore, the neural net topology suggests means for parallel processing - useful in high frequency processes.

The scheme shows promise as an effective model for the analysis of vibration and process data for many industrial and other engineered systems.

The training procedure is illustrated on the modeling of simulated and real processes in the next chapter.

4. PRACTICAL CONSIDERATION USING MATLAB

4.1 Overview

This chapter introduces, Times-Series Comparison Method of Prediction, Design of Wavelet Neural Network Multiresolution System, and Modeling and Experimental Results.

4.2 Times-Series Comparison Method of Prediction

During the last two decades, various approaches have been developed for time series prediction. Among them linear regression methods such as autoregressive (AR) and autoregressive moving average (ARMA) models have been the most used methods in practice. The theory of linear models is well known, and many algorithms for model building are available.

Linear models are usually inadequate for financial time series as in practice almost all economic processes are nonlinear to some extent. Nonlinear methods are widely applicable nowadays with the growth of computer processing speed and data storage. Of the nonlinear methods, neural networks have become very popular. Many different types of neural networks such as MLP and RBF have been proven to be universal function approximators, which make neural networks attractive for time series modeling, and for financial time-series forecasting in particular.

An important prerequisite for the successful application of some modem advanced modeling techniques such as neural networks, however, is a certain uniformity of the data [14]. In most cases, a stationary process is assumed for the temporally ordered data. In financial time series, such an assumption of stationarity has to be discarded. Generally speaking, there may exist different kinds of nonstationarities. For example, a process may be a superposition of many sources, where the underlying system drifts or switches between different sources, producing different dynamics. Standard approaches such as AR models or nonlinear AR models using MLPs usually give best results for stationary time series. Such a model can be termed as global as only one model is used to characterize the measured process. When a series is nonstationary, as is the cage for most financial time series, identifying a proper global model becomes very difficult, unless the nature of the nonstationarity is known. In recent years, local models have grown in interest for improving the prediction accuracy for nonstationary time series.

4.3 Design of Wavelet Neural Network Multiresolution System

The efficient way is to design a hybrid scheme incorporating multiresolution decomposition techniques such as the wavelet transform, which can produce a good local representation of the signal in both the time domain and the frequency domain [13]. In contrast to the Fourier basis, wavelets can be supported on an arbitrarily small closed interval. Thus, the wavelet transform is a very powerful tool for dealing with transient phenomena.

There are many possible applications of combining wavelet transformations into financial time-series analysis and forecasting. Recently some financial forecasting strategies have been discussed that used wavelet transforms to preprocess the data. The preprocessing methods they used are based on the translation invariant wavelet transform [7] or a *trous* wavelet transform [4].

In this work, we have developed a neuro-wavelet hybrid system that incorporates multiscale wavelet analysis into a set of neural networks for a multistage time series prediction. Compared to the work in [11], our system exploits a shift invariant wavelet transform called the autocorrelation shell representation (ASR) [4] instead of the multiscale orthogonal wavelet transform as was originally presented in [13]. It is cumbersome to apply the commonly defined DWT for real-time time series applications due to the lack of shift invariance, which plays an important role in time series forecasting. Using a shift invariant wavelet transform, we can easily relate the resolution scales exactly to the original time series and preserve the integrity of some short-lived events [2].

Basically, we suggest the direct application of the *a trous* wavelet transform based on the ASR to financial time series and the prediction of each scale of the wavelet's coefficients by a separate feedforward neural network. The separate predictions of each scale are proceeded independently. The prediction results for the wavelet coefficients can be combined directly by the linear additive reconstruction property of ASR, or preferably, as we propose in this chapter, by another NN in order to predict the original time series. The aim of the last network is to adaptively choose the weight of each scale in the final prediction [11], as illustrated in figure 4.1.

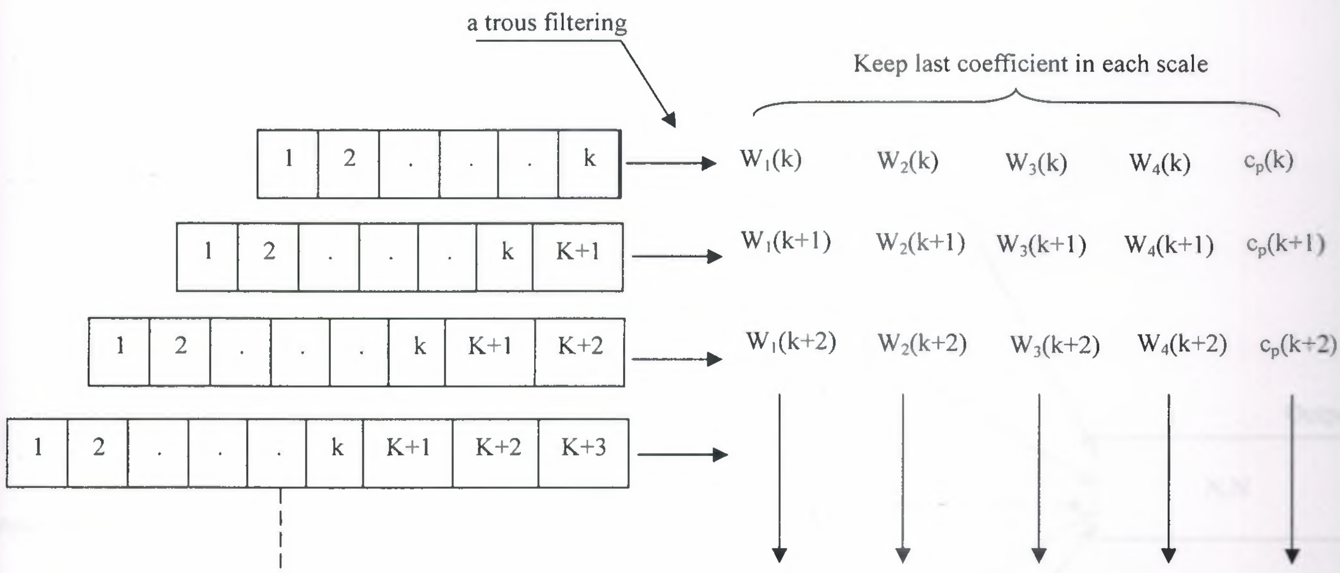

Figure 4.1 Illustration of the Procedure for Preparing Data in the Hybrid Neuro-Wavelet Prediction Scheme.

Figure 4.2 shows our hybrid neuro-wavelet scheme for time series prediction. Given the time series f(n), n = 1,....., N, so our aim is to predict the lth sample a head, f(N+l), of the series. That is, $I = 1$ for single step prediction; for each value of I we train a separate prediction architecture. The hybrid scheme basically involves three stages, which bear a similarity with the scheme in [11]. In the first stage, the time series is decomposed into different scales by autocorrelation shell decomposition. In the second stage, each scale is predicted by a separate NN and in the third stage, the next sample of the original time series is predicted, using the different scale's prediction, by another NN. More details are expounded as follows.

For time series prediction, correctly handling the temporal aspect of data is our primary concern. The *time-based a trous* transform as described above provides a simple method.

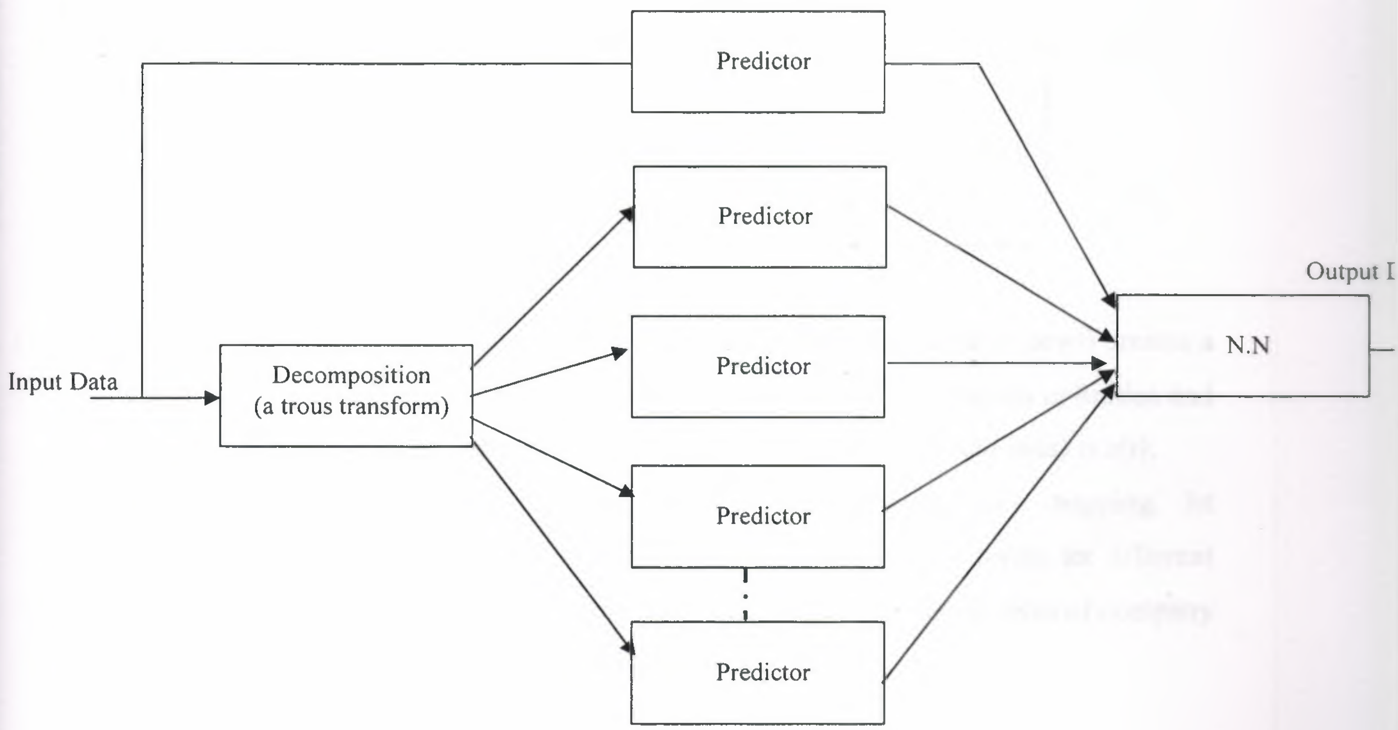

Figure 4.2 Wavelet/Neural Net Multiresolution System.

In time series prediction is estimated a function r(t+p) at time (t+p), using the k-time steps back time from t time.

4.4 Modeling and Experimental Results

For modeling we used 1200 data, saved in MATLAB software under Huthaifa.m, we developed 2-layer (hidden and output) feedforward ANN based on adaptive learning algorithm using the following MATLAB command.

```
net=newff(minmax(p),[50,1],{'tansig','tansig'},'traingda');
net.trainParam.show = 50;
net.trainParam.lr = 0.05;
net.trainParam.lr_inc = 1.04;
net.trainParam.epochs = 3000;
net.trainParam.goal = 1e-3;
[net,tr]=train(net,p,t);
 a = sim(net,p);
```

The first step in our design is to create the network object. The function newff creates a trainable ANN involving p = [r(t); r(t-1); r(t-2)] three inputs, 50 neurons in hidden and one neuron in output layer, target is d = r(t+1); ANN output (predicted data) is a(t). MATLAB provides a function called bilinear to implement this mapping. Its invocation is similar to impinvr function, but it also takes several forms for different input-output quantities. The plot of input data that characterize return rates of company taking from stock marketing is given in figure 4.3.

```
x=load ('huthaifa.m');
plot(x)
title('original signal')
ylabel('0:1200');
```

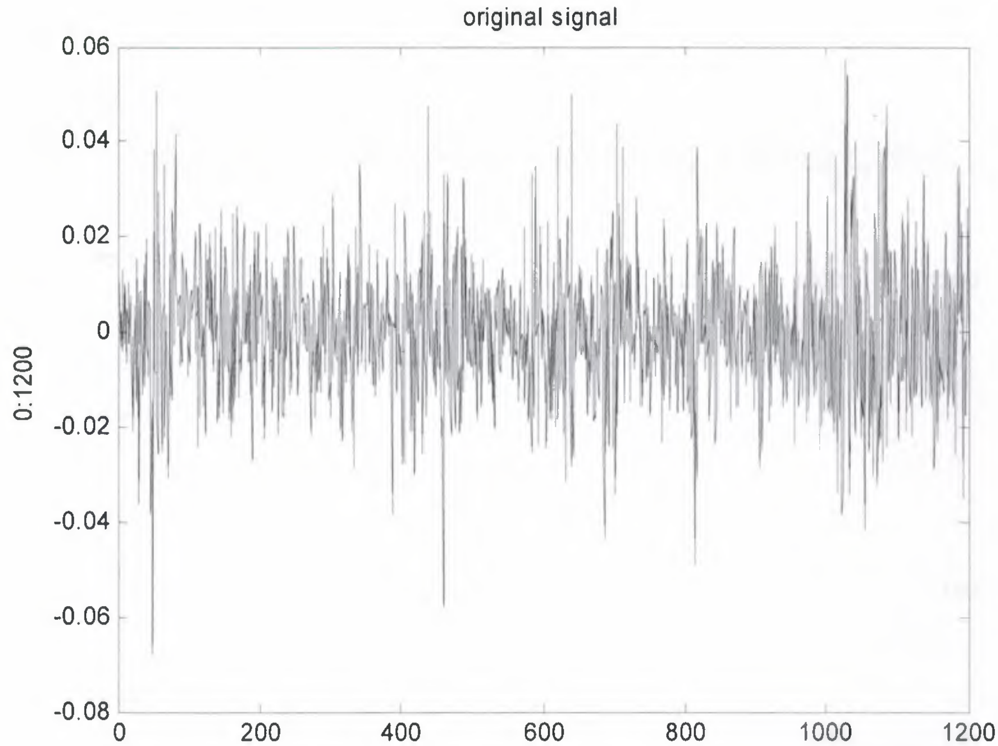

Figure 4.3 Input Data.

```
% Illustrate Decomposition using Wavelet "Haar" to Obtain approximation and Detail
x=load ('huthaifa.m');
 subplot(3,1,1);
plot(x)
title('original signal')
ylabel('0:1200');
ts=x(1:150);
is=length(ts);
[ca1,cd1]=dwt(ts,'haar');
%Perform one step reconstruction of ca1 and cd1
a1=upcoef('a',ca1,'haar',1,is);
subplot(3,1,2);
plot(a1)
title('a1'); ylabel('1:150');
d1=upcoef('d',cd1,'haar',1,is);
subplot(3,1,3);
plot(d1)
title('d1'); ylabel('1:150');
```

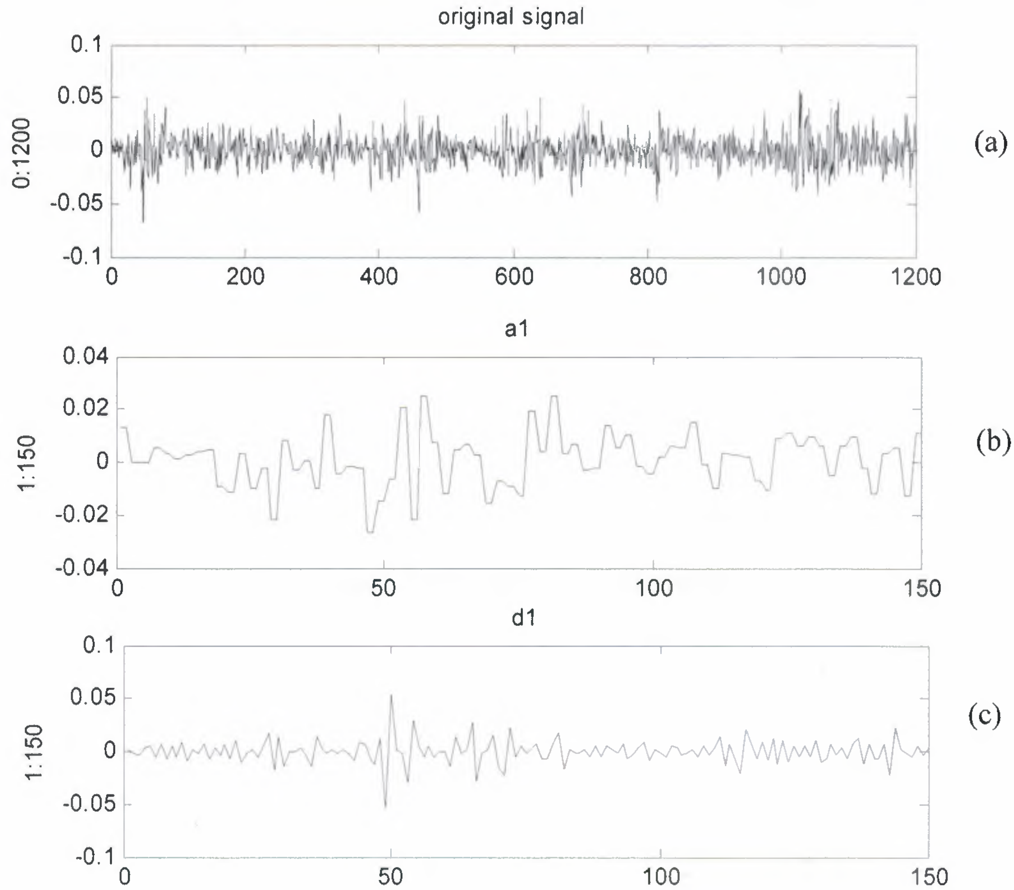

Figure 4.4 (a) Original Signal, (b) Approximation of Decomposed Signal, and (c) Detail of Decomposed Signal.

```
% Illustrate Prediction algorithm for the period of 1:150 from the original signal and the
% output of the Neural Network
x1=lagmatrix(a1,1)';
x2=lagmatrix(x1,1)';
x3=lagmatrix(x2,1)';
y1=x1(:,4:146);
y2=x2(:,4:146);
y3=x3(:,4:146);
d0=d1';
d1=d0(:,4:146);
[d1' y1' y2' y3'];
p = [y1;y2;y3];
t = [d1];
figure
```

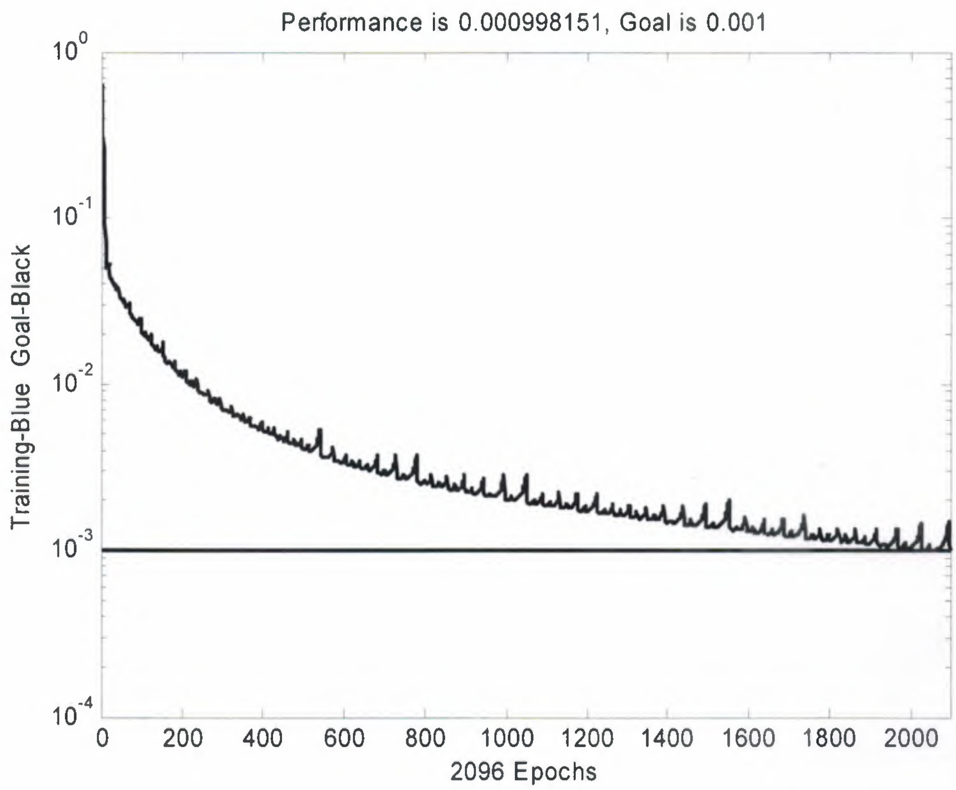

Figure 4.5 Output of Neural Network Training

```
% Illustrates the calculation of error and beta signal
error=d1-a;
b=max(a);
beta=error*100./b;
subplot(3,1,1)
plot(d1)
title('detail signal');
subplot(3,1,2)
plot(a)
title('beta signal');
subplot(3,1,3)
plot(error)
title('error signal');
[d1' a' beta'];
```

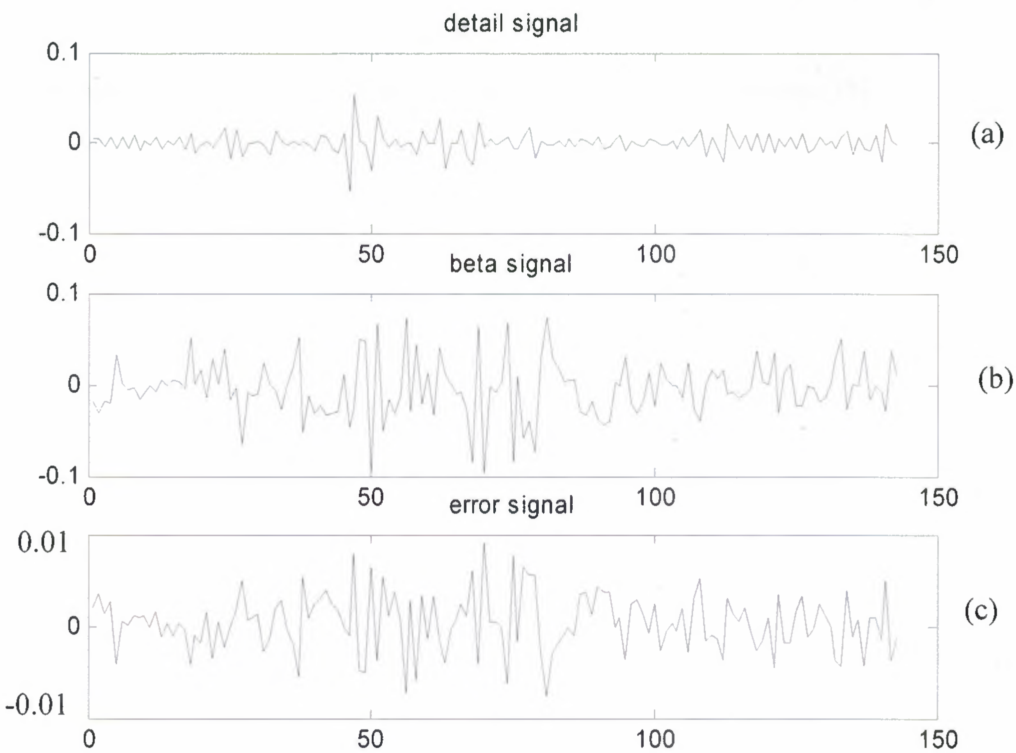

Figure 4.6 (a) Detail Signal as Input Signal to Neural Network, (b) Beta Signal as Target of N.N, and (c) Error Signal.

The parameters used are as follows: training rate $\eta = 0.05$, maximal number of epochs is set to 3000, training goal (graidient) is 10^{-3}. The training function train involves network object net, input vector –p, and target vector d. The function sim simulates the network- takes the network input p, network object net and return a network output a(t). The training stopped after n = 2096 epochs when training error remains within of performance index (goal) that is set to 10^{-3}. The training is shown in figure 4.6.

4.5 Summary

This chapter were considered 2-layers feedforward back propagation ANN based on adaptive learning algorithm for time series signal prediction developed ANN has 3-inputs, 50-neurons in hidden layer and single output. The number of updated weights is 200. (The rest of the data are in the Appendix).

Experimental results show that the relative error of prediction is less than 3%.

CONCLUSION

Literature review on application of the wavelet for time series prediction was given. Continuous and discrete wavelet transforms, their properties, time and frequency selectivity were analyzed.

ANN based on backpropagation algorithm with different activation functions were examined.

Neuro-wavelet hybrid system training with appropriate translation and dilation parameter were designed. Neuro-wavelet system with wavelet multi-scale analyzer connected to parallel ANN predictors, were designed Output predicted signal is obtained by combinations of the output of ANNs.

The developed neuro-wavelet network is used for prediction return rates in stock market for prediction of return rates of company.

Wavelet neural network based on multi-resolution prediction allow to increase precision of prediction of long term forecasting non-stationary time series.

REFERENCES

[1] A. Aussem and F. Murtagh, "Combining neural networks forecasts on wavelet-transformed time series," *Connection Sci.*, vol. 9, pp. 113–121, 1997.

[2] A. Aussem, J. Campbell, and F. Murtagh, "Wavelet-based feature extraction and decomposition strategies for financial forecasting," *J. Comput. Intell. Finance*, pp. 5–12, Mar. 1998.

[3] A. M. Azoff, *Neural Network Time Series Forecasting of Financial Markets*. New York: Wiley, 1994.

[4] G. Beylkin and N. Satio, "Wavelets, their autocorrelation functions and multiresolution representation of signals," *IEEE Trans. Signal Processing*, vol. 7, pp. 147–164, 1997.

[5] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford,U.K.: Oxford Univ. Press, 1995.

[6] Bayesian methods for neural networks," Tech. Rep.NCRG/95/009, 1995.

[7] R. R. Coifman and D. L. Donoho, "Translation-invariant de-noising,"in *Wavelets and Statistics, Springer Lecture Notes*, A. Antoniades,Ed. New York: Springer-Verlag, 1995.

[8] D. R. Dersch, B. G. Flower, and S. J. Pickard, "Exchange rate trading using a fast retraining procedure for generalized radial basis function networks," in *Proc. Neural Networks Capital Markets*, 1997.

[9] R. J. Van Eyden, *The Application of Neural Networks in the Forecasting of Share Prices*. Haymarket, VA: Finance & Technology Publishing, 1995.

[10] B. G. Flower, T. Cripps, M. Jabri, and A. White, "An artificial neural network based trade forecasting system for capital markets," in *Proce. Neural Networks Capital Markets*, 1995.

[11] A. B. Geva, "ScaleNet—Multiscale neural-network architecture for time series prediction," *IEEE Trans. Neural Networks*, vol. 9, pp. 1471–1482, 1998.

[12] A. Kehagias and V. Petridis, "Predictive modular neural networks for time series classification," *Neural Networks*, vol. 10, pp. 31–49, 1997.

[13] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 674–693, 1989.

[14] K. R. Müller, J. Kohlmorgen, and K. Plawelzik, "Analysis of switching dynamics with computing neural networks," Univ. Tokyo, Tech. Rep., 1997.

[15] D. J. C. MacKay, "A practical Bayesian framework for backpropagation networks," *Neural Comput.*, vol. 4, pp. 448–472, 1992.

[16] Bayesian nonlinear modeling for the 1993 energy prediction competition," in *Maximum Entropy and Bayesian Methods, SantaBarbara 1993*, G. Heidbreder, Ed. Dordrecht, The Netherlands: Kluwer, 1995.

[17] T. Masters, *Neural, Novel & Hybrid Algorithms for Time Series Prediction*. New York: Wiley, 1995.

[18] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting, Methods and Applications*, 3rd ed. New York: Wiley, 1998.

[19] F. F. Murtagh, "Wedding the wavelet transform and multivariate data analysis," *J. Classification*, vol. 15, pp. 161–183, 1998.

[20] V. Petridis and A. Kehagias, "Modular neural networks for MAP classification of time series and the partition algorithm," *IEEE Trans. Neural Networks*, vol. 17, pp. 73–86, 1996.

[21] N. Saito and G. Beylkin, "Multiresolution representations using the auto-correlation functions of compactly supported wavelets," *IEEE Trans. Signal Processing*, 1992.

[22] W. F. Sharpe, "The Sharpe ratio," *J. Portfolio Management*, pp. 49–58, Fall 1994.

[23] M. J. Shensa, "The discrete wavelet transform: Wedding the á trous and Mallat algorithms," *IEEE Trans. Signal Processing*, vol. 10, pp. 2463–2482, 1992.

[24] M. R. Thomason, "Financial forecasting with wavelet filters and neural networks," *J. Comput. Intell. Finance*, pp. 27–32, Mar. 1997.

[25] A. S. Weigend and M. Mangeas, "Nonlinear gated experts for time series: Discovering regimes and avoiding overfitting," *Int. J. Neural Syst.*, vol. 6, pp. 373–399, 1995.

[26] P. Zuohong andW. Xiaodi, "Wavelet-based density estimator model for forecasting," *J. Comput. Intell. Finance*, pp. 6–13, Jan. 1998.

[27] Z. Gonghui, J.-L. Starck, J. Campbell, and F. Murtagh, "The wavelet transform for filtering financial data streams," *J. Comput. Intell. Finance*, pp. 18–35, June 1999.

[28] J. S. Zirilli, *Financial Prediction Using Neural Networks*: International Thomson Computer Press, 1997.

[29] V Bjorn. Multiresolution methods for Financial time series prediction. In Proceedings of the IEEE/IAFE 1995 on Computational Intelligence for Financial Engineering, page 97, 1995.

[30] L. Hong, G. Chen, and C.K. Chui. A Filter-bank based Kalman Filter technique for wavelet estimation and decomposition of random signals. IEEE Trans. on Circuit Systems – II Analog and Digital Signal Processing., 45(2):237{241, 1998.

APPENDIX A

MATLAB Program for Wavelet Decomposition and Prediction of Signal using Backpropagation with Adaptive Training

First Part of the Signal

```
x=load ('huthaifa.m');
plot(x)
title('original signal')
ylabel('0:1200');
% Illustrate Decomposition using Wavelet "Haar" to Obtain approximation and Detail
x=load ('huthaifa.m');
 subplot(3,1,1);
plot(x)
title('original signal')
ylabel('0:1200');
ts=x(1:150);
is=length(ts);
[ca1,cd1]=dwt(ts,'haar');
%Perform one step reconstruction of ca1 and cd1
a1=upcoef('a',ca1,'haar',1,is);
subplot(3,1,2);
plot(a1)
title('a1'); ylabel('1:150');
d1=upcoef('d',cd1,'haar',1,is);
subplot(3,1,3);
plot(d1)
title('d1'); ylabel('1:150');
% Illustrate Prediction algorithm for the period of 1:150 from the original signal and the
% output of the Neural Network
x1=lagmatrix(a1,1)';
x2=lagmatrix(x1,1)';
x3=lagmatrix(x2,1)';
y1=x1(:,4:146);
y2=x2(:,4:146);
y3=x3(:,4:146);
d0=d1';
d1=d0(:,4:146);
[d1' y1' y2' y3'];
p = [y1;y2;y3];
```

```
t = [d1];
figure
% Illustrates the calculation of error and beta signal
error=d1-a;
b=max(a);
beta=error*100./b;
subplot(3,1,1)
plot(d1)
title('detail signal');
subplot(3,1,2)
plot(a)
title('beta signal');
subplot(3,1,3)
plot(error)
title('error signal');
[d1' a' beta'];
```

Second Part of the Signal

```
x=load ('huthaifa.m');
s= x(1:300);subplot(3,1,1);
plot(s)
title('original signal')
ylabel('200:300');
is=length(s);
%Perform one step decomposition of S using haar
 [ca1,cd1]=dwt(s,'haar');
%Perform one step reconstruction of ca1 and cd1
a2=upcoef('a',ca1,'haar',1,is);subplot(3,1,2);
plot(a2)
title('a2'); ylabel('200:300');
d2=upcoef('d',cd1,'haar',1,is);subplot(3,1,3);
plot(d2)
title('d2'); ylabel('200:300');
x1=lagmatrix(s,1)';
x2=lagmatrix(x1,1)';
x3=lagmatrix(x2,1)';
```

```
y1=x1(:,204:296);
y2=x2(:,204:296);
y3=x3(:,204:296);
d1=d2';
d2=d1(:,204:296);
[d2' y1' y2' y3'];
p = [y1;y2;y3];
t = [d2];
net=newff(minmax(p),[50,1],{'tansig','tansig'},'traingda');
net.trainParam.show = 50;
net.trainParam.lr = 0.03;
net.trainParam.lr_inc = 1.04;
net.trainParam.epochs = 7000;
net.trainParam.goal = 1e-3;
[net,tr]=train(net,p,t);
a = sim(net,p);
error=d2-a;
b=max(a);
beta=error*100./b;
figure
subplot(3,1,1)
plot(d1)
title('detail signal');
subplot(3,1,2)
plot(a)
title('beta signal');
subplot(3,1,3)
plot(error)
title('error signal');
[d1' a' beta'];
```

Third Part of the Signal

```
x=load ('huthaifa.m');
s= x(1:500);subplot(3,1,1);
plot(s)
title('original signal')
ylabel('300:500');
is=length(s);
%Perform one step decomposition of S using haar
 [ca1,cd1]=dwt(s,'haar');
%Perform one step reconstruction of ca1 and cd1
a3=upcoef('a',ca1,'haar',1,is);subplot(3,1,2);
plot(a3)
title('a3'); ylabel('300:500');
d3=upcoef('d',cd1,'haar',1,is);subplot(3,1,3);
plot(d3)
title('d3'); ylabel('300:500');
x1=lagmatrix(s,1)';
x2=lagmatrix(x1,1)';
x3=lagmatrix(x2,1)';
y1=x1(:,304:496);
y2=x2(:,304:496);
y3=x3(:,304:496);
d2=d3';
d3=d2(:,304:496);
[d3' y1' y2' y3'];
p = [y1;y2;y3];
t = [d3];
net=newff(minmax(p),[50,1],{'tansig','tansig'},'traingda');
```

```
net.trainParam.show = 50;
net.trainParam.lr = 0.03;
net.trainParam.lr_inc = 1.04;
net.trainParam.epochs = 7000;
net.trainParam.goal = 1e-3;
[net,tr]=train(net,p,t);
a = sim(net,p);
error=d3-a;
b=max(a);
beta=error*100./b;
figure
subplot(3,1,1)
plot(d3)
title('detail signal');
subplot(3,1,2)
plot(a)
title('beta signal');
subplot(3,1,3)
plot(error)
title('error signal');
[d3' a' beta'];
```

APPENDIX B

Original Signal

1	0.014757
2	0.011961
3	-0.00318
4	0.003471
5	0.004668
6	-0.00408
7	0.012982
8	-0.00186
9	0.009517
10	-0.00231
11	0.01015
12	-0.0067
13	0.00498
14	0.000739
15	0.010633
16	-0.00235
17	0.007814
18	0.002331
19	-0.00223
20	-0.01607
21	-0.00085
22	-0.0209
23	0.000921
24	0.005671
25	-0.01484
26	-0.00445
27	0.015766
28	-0.01949
29	-0.00734
30	-0.03624
31	0.008684
32	0.007583
33	-0.00017
34	-0.00579
35	-0.01307
36	0.014257
37	-0.00859
38	-0.01131
39	0.019685
40	0.016177
41	-0.00285
42	-0.00588
43	-0.00949

44	0.006382
45	0.004328
46	-0.00792
47	-0.03837
48	-0.01482
49	-0.06801
50	0.03863
51	-0.0038
52	-0.0083
53	-0.00852
54	0.050899
55	-0.01686
56	-0.02585
57	0.029453
58	0.020475
59	0.00773
60	0.007517
61	-0.02545
62	0.001197
63	0.003725
64	0.005606
65	0.035411
66	-0.02192
67	0.001947
68	0.003771
69	0.000315
70	-0.03051
71	-0.03011
72	0.016434
73	-0.014
74	-0.00402
75	-0.01413
76	-0.01158
77	0.026005
78	0.013531
79	-0.00292
80	0.010786
81	0.041729
82	0.008525
83	0.005651
84	0.00145
85	0.00563
86	0.008001

87	-0.00724
88	0.001541
89	-0.00651
90	0.002581
91	0.016703
92	0.011732
93	0.011769
94	-0.00068
95	0.007049
96	0.01357
97	0.006315
98	-0.00947
99	-0.00172
100	-0.00646
101	-0.00293
102	0.007184
103	0.009016
104	0.003037
105	0.004529
106	0.007104
107	0.009492
108	0.021194
109	-0.00439
110	0.00328
111	0.0046
112	-0.02407
113	0.010012
114	-0.00343
115	-0.01802
116	0.023128
117	0.009314
118	-0.00532
119	0.001786
120	-0.01561
121	0.001236
122	-0.02166
123	0.018954
124	-0.00077
125	0.015526
126	0.006822
127	0.012466
128	0.000607
129	0.020747
130	-0.00185
131	-0.00064

132	0.013317
133	-0.00796
134	-0.00219
135	-0.00092
136	0.013582
137	0.02214
138	-0.00205
139	0.004221
140	-0.00879
141	-0.01928
142	-0.00412
143	-0.01799
144	0.025631
145	0.00703
146	0.00369
147	-0.01708
148	-0.00807
149	0.007174
150	0.014854
151	-0.0073
152	0.017858
153	0.011277
154	-0.00519
155	-0.00865
156	0.007987
157	-0.01854
158	-0.00728
159	0.003526
160	-0.02221
161	0.006093
162	0.024919
163	-0.01907
164	0.009544
165	-0.01437
166	0.010825
167	0.001568
168	0.026565
169	-0.00075
170	-0.01398
171	-0.00669
172	-0.00537
173	-0.00175
174	-0.00862
175	0.001795
176	0.015427

177	0.023126
178	0.005692
179	-0.00225
180	0.005469
181	0.008424
182	-0.00238
183	0.009787
184	-0.00067
185	-0.00655
186	0.014432
187	-0.01311
188	-0.00175
189	-0.02688
190	0.00511
191	0.016869
192	-0.00557
193	0.021336
194	-0.00719
195	-0.01106
196	0.005714
197	0.021179
198	-0.00244
199	0.006829
200	0.01288
201	0.003252
202	0.007624
203	-0.00648
204	-0.01584
205	-0.00421
206	-0.00291
207	-0.02238
208	0.012943
209	0.02293
210	0.016989
211	-0.00145
212	0.002351
213	0.002029
214	-0.00872
215	-0.00598
216	-0.0057
217	0.014567
218	-0.01671
219	0.011494
220	-0.01133
221	0.009722
222	-0.00349
223	0.011423
224	0.006189
225	0.00261
226	-0.02176
227	0.001263
228	-0.00461
229	0.008183
230	-0.00402
231	-0.00638
232	-0.01777
233	-0.01703
234	0.015852
235	-0.0179
236	0.015943
237	-0.00582
238	0.000425
239	0.003653
240	0.021708
241	0.005099
242	-0.01288
243	0.000994
244	-0.012
245	-0.00705
246	0.000278
247	0.005533
248	0.02248
249	0.007133
250	0.002194
251	0.004587
252	-0.00973
253	-0.00211
254	-0.01296
255	-0.00036
256	0.012195
257	0.015097
258	0.015731
259	0.00601
260	0.00743
261	-0.00223
262	0.005576
263	-0.00103
264	0.006354
265	-0.00298
266	-0.00396

267	0.003308
268	0.008189
269	0.006498
270	-0.00784
271	-0.0217
272	0.00159
273	-0.01328
274	-0.00296
275	-0.00677
276	0.011189
277	0.001878
278	-0.01785
279	-0.00918
280	-0.0005
281	-0.00442
282	-0.01274
283	0.00642
284	-0.01022
285	-0.00191
286	-0.01261
287	0.015998
288	-0.0029
289	0.02274
290	0.002327
291	0.010062
292	-0.00842
293	-0.00694
294	0.009837
295	0.017664
296	0.002411
297	0.013414
298	-0.01431
299	-0.01009
300	-0.01799
301	-0.00273
302	0.008073
303	-0.00899
304	0.028906
305	-0.005
306	-0.00467
307	0.002611
308	0.002968
309	-0.00557
310	-0.00583
311	-0.01371

312	0.000387
313	0.012848
314	8.24E-05
315	-0.02093
316	0.002241
317	-0.02297
318	-0.00238
319	0.004658
320	-0.00086
321	-0.01079
322	0.011306
323	7.80E-05
324	0.016986
325	-0.00249
326	0.018481
327	-0.00585
328	0.013949
329	-0.00061
330	-0.0166
331	-0.02094
332	-0.00166
333	-0.02806
334	0.005387
335	0.005733
336	0.022286
337	-0.00451
338	0.014054
339	-0.00616
340	-0.00906
341	0.011545
342	0.035266
343	0.015263
344	-0.00646
345	-0.00471
346	0.005335
347	0.00569
348	0.00557
349	0.004948
350	-0.00852
351	0.005991
352	0.005825
353	0.010569
354	-0.0012
355	0.018417
356	-0.00659

357	0.010087
358	-0.00206
359	-0.00075
360	-0.01147
361	0.008856
362	-0.00032
363	-0.0062
364	-0.01344
365	0.006343
366	0.008099
367	0.017217
368	-0.00696
369	-0.00995
370	-0.00375
371	0.003013
372	0.006342
373	-0.00128
374	-0.00851
375	0.007241
376	0.003856
377	0.001586
378	-0.00207
379	0.010817
380	0.001884
381	0.015465
382	-0.00085
383	0.000384
384	0.003979
385	0.00069
386	0.003264
387	-0.00955
388	-0.03834
389	0.001922
390	0.000956
391	0.02709
392	0.01119
393	-0.01306
394	-0.00439
395	0.01217
396	0.010671
397	-0.00683
398	0.000522
399	-0.0071
400	-0.00291
401	-0.02763

402	0.006065
403	-0.00421
404	-0.00394
405	-0.02746
406	0.025218
407	0.010628
408	-0.00011
409	0.011248
410	-0.00042
411	-9.13E-05
412	0.012273
413	-0.02082
414	0.003627
415	-0.02097
416	0.002033
417	0.008713
418	-0.01026
419	0.000425
420	-0.03038
421	0.004517
422	0.006301
423	-0.00534
424	-0.01483
425	0.011017
426	0.013627
427	0.009346
428	0.001863
429	0.019837
430	-0.0127
431	-0.02563
432	0.008173
433	0.025602
434	-0.00472
435	-0.00821
436	-0.01769
437	0.024273
438	0.047646
439	0.004114
440	-0.00535
441	0.025566
442	0.004532
443	0.017799
444	7.20E-05
445	-0.00236
446	-0.01058

447	0.000524
448	-0.01366
449	0.007164
450	0.004931
451	-0.00746
452	-0.00492
453	0.009392
454	0.009998
455	-0.00784
456	-0.00257
457	-0.02227
458	-0.01817
459	-0.05828
460	0.033084
461	0.028663
462	-0.00981
463	0.004953
464	-0.00326
465	0.033276
466	-0.01113
467	0.00269
468	-0.00853
469	0.010892
470	-0.01496
471	-0.02157
472	-0.00391
473	0.01636
474	-0.00591
475	-0.00845
476	-0.0206
477	0.017902
478	0.009341
479	0.022098
480	0.009419
481	-0.01244
482	-0.00731
483	-0.02105
484	-0.00443
485	-0.01918
486	0.018335
487	-0.01253
488	-0.00253
489	0.032242
490	-0.0013
491	0.019858

492	0.019637
493	-0.00652
494	-0.00667
495	0.009274
496	-0.00659
497	-0.00323
498	-0.00752
499	0.01621
500	0.000749
501	0.005569
502	-0.00965
503	0.014708
504	-0.00676
505	0.002155
506	-0.01822
507	-0.00737
508	0.009594
509	-0.00327
510	0.002944
511	-0.00854
512	0.008465
513	0.010271
514	-0.01586
515	0.007219
516	0.015261
517	-0.00222
518	0.003565
519	0.00813
520	0.001956
521	0.009453
522	0.000338
523	-0.01109
524	-0.00789
525	0.009184
526	-0.01028
527	-0.01074
528	0.006952
529	-0.01495
530	-0.00193
531	-0.02051
532	0.007705
533	0.005081
534	0.000417
535	0.009634
536	0.007139

537	0.011204
538	0.002352
539	-0.0067
540	-0.00857
541	0.007937
542	0.013398
543	-0.00478
544	-0.00309
545	0.010961
546	-0.00291
547	0.005202
548	-0.0009
549	0.005233
550	0.001554
551	-0.00123
552	0.005072
553	-0.00281
554	-0.0048
555	0.010043
556	0.002036
557	-0.009
558	-0.00984
559	0.006876
560	-0.00533
561	-0.00351
562	-0.00488
563	0.00197
564	-0.00272
565	-0.01017
566	-0.01453
567	0.010654
568	-0.00586
569	-0.00158
570	-0.00023
571	-0.00669
572	-0.00821
573	-0.00045
574	0.022235
575	-0.01494
576	-0.00019
577	-0.0068
578	0.00551
579	0.001367
580	-0.019
581	-0.00494

582	-0.01071
583	-0.01617
584	-0.02551
585	0.033381
586	0.000327
587	-0.01793
588	-0.00581
589	0.034743
590	0.005883
591	-0.00082
592	0.001684
593	-0.02377
594	-0.00034
595	0.011096
596	0.01383
597	0.021978
598	-0.00572
599	0.004996
600	-0.00114
601	0.003855
602	-0.00022
603	-0.01578
604	-0.00649
605	-0.0244
606	-0.01078
607	0.023452
608	0.00496
609	-0.01258
610	-0.00335
611	-0.01835
612	0.003523
613	-0.01855
614	0.014678
615	0.005366
616	-0.00955
617	0.004371
618	-0.02011
619	0.000213
620	0.007406
621	0.038922
622	-0.01822
623	-0.00585
624	0.019605
625	0.007526
626	-0.00654

627	-0.00816
628	-0.01401
629	-0.02146
630	0.008071
631	-0.01296
632	-0.0313
633	0.008002
634	0.024387
635	0.007075
636	0.01044
637	0.003988
638	-0.01045
639	-0.02803
640	0.050099
641	-0.01055
642	-0.02624
643	-0.00192
644	0.003812
645	0.009586
646	0.010318
647	-0.00623
648	0.006143
649	0.002126
650	0.013915
651	-0.00403
652	0.000268
653	0.013031
654	0.002867
655	-0.00498
656	-0.00189
657	0.006805
658	0.007008
659	-0.00562
660	0.005461
661	-0.01747
662	0.003587
663	-0.00151
664	-0.00841
665	-0.00623
666	-0.01334
667	0.011827
668	-0.00865
669	-0.00218
670	0.008124
671	-0.01891

672	-0.01736
673	-0.01851
674	-0.00195
675	-0.00556
676	0.01749
677	-0.00766
678	-0.01431
679	0.00104
680	-0.00568
681	0.005858
682	0.009981
683	0.006452
684	0.002259
685	-0.02476
686	-0.04318
687	0.014828
688	-0.02584
689	0.005871
690	-0.01962
691	0.017627
692	-0.02408
693	-0.01792
694	-0.00406
695	0.019909
696	0.011282
697	0.025575
698	-0.02443
699	-0.00463
700	0.010784
701	-0.01246
702	-0.03439
703	-0.0029
704	0.04368
705	-0.01998
706	0.008117
707	0.027066
708	-0.00213
709	0.015104
710	-0.00323
711	0.010282
712	0.03889
713	0.012543
714	-0.00854
715	-0.01498
716	-0.01216

717	0.015941
718	0.004696
719	0.01501
720	-0.00287
721	0.01359
722	0.000782
723	-0.01487
724	0.01444
725	-0.00245
726	-0.00183
727	-0.00449
728	-0.00029
729	-0.00758
730	0.002609
731	0.000416
732	0.028453
733	0.002724
734	0.002693
735	0.016154
736	-0.00263
737	-0.01553
738	0.003196
739	-0.01182
740	-0.00779
741	-0.01566
742	0.006202
743	0.003862
744	0.005108
745	0.01299
746	-0.01055
747	0.005457
748	-0.0094
749	-0.00836
750	0.001164
751	-0.01135
752	-0.0175
753	-0.00452
754	-0.00488
755	0.003434
756	0.008709
757	0.011364
758	-0.00945
759	-0.00551
760	-0.00151
761	-0.00468

762	0.012493
763	-0.00148
764	0.010079
765	-0.00184
766	-0.01232
767	-0.0235
768	0.006879
769	-0.0144
770	-0.00113
771	0.023691
772	0.006241
773	-0.01088
774	0.009971
775	-0.00554
776	0.006053
777	-0.00343
778	-0.01637
779	-0.01627
780	0.01608
781	0.010449
782	0.002402
783	-0.00108
784	0.005571
785	0.00388
786	0.003964
787	-0.00524
788	-0.01142
789	0.003265
790	-0.01733
791	-8.45E-05
792	0.005687
793	0.000949
794	-0.00383
795	-0.00734
796	0.00309
797	-0.01666
798	0.008124
799	-0.01208
800	0.006956
801	-0.00276
802	0.019654
803	-0.00483
804	-0.01501
805	-0.01115
806	-0.017

807	0.00403	852	0.001582
808	-0.00056	853	-0.00177
809	-0.00106	854	0.018563
810	-0.02239	855	0.001861
811	-0.01864	856	0.000903
812	0.006226	857	-0.00314
813	-0.04922	858	0.010899
814	-0.0058	859	-0.0073
815	-0.01611	860	-0.00493
816	-0.03106	861	0.011706
817	-0.01903	862	0.006155
818	0.038983	863	-0.00684
819	0.00879	864	-0.01825
820	-0.00517	865	0.01035
821	0.011489	866	-0.00066
822	0.021922	867	-0.00838
823	-0.0023	868	0.013187
824	0.012306	869	0.022318
825	0.019927	870	-0.00278
826	-0.00247	871	-0.00753
827	0.001636	872	-0.01587
828	-0.00834	873	-0.00278
829	-0.00536	874	0.000273
830	0.022938	875	-0.01556
831	0.015208	876	0.003314
832	-0.00527	877	0.010035
833	-0.00153	878	0.007546
834	0.006936	879	0.00581
835	-0.01863	880	-0.00838
836	-0.00787	881	0.004351
837	0.004557	882	-0.00021
838	0.015296	883	0.004124
839	-0.0047	884	0.006752
840	0.000387	885	0.003362
841	0.013721	886	-0.01115
842	0.004109	887	0.00574
843	-0.02382	888	0.00918
844	-0.01717	889	0.006213
845	-9.44E-06	890	-0.0065
846	0.022948	891	-0.00359
847	0.00286	892	-0.0048
848	0.014386	893	0.001221
849	0.014526	894	-0.00947
850	-0.00273	895	-0.00628
851	0.002456	896	0.006834

897	-0.01625
898	0.01003
899	-0.00992
900	-0.00733
901	0.007925
902	0.003519
903	0.000998
904	-0.00019
905	-0.02861
906	0.011748
907	0.014934
908	-0.00708
909	-0.02474
910	-0.00404
911	-0.00597
912	-0.00308
913	0.014859
914	0.01434
915	-0.00399
916	0.009941
917	-0.00181
918	-0.01102
919	-0.01887
920	0.013514
921	-0.01551
922	0.008224
923	0.017975
924	-4.51E-05
925	0.00046
926	-0.00285
927	0.022634
928	0.019491
929	-0.00667
930	0.01451
931	-0.0045
932	0.005849
933	0.003393
934	-0.00229
935	-0.00986
936	-0.00091
937	0.011379
938	-0.00052
939	0.004067
940	-0.01576
941	0.001511

942	-0.00424
943	-0.01465
944	0.005849
945	0.005349
946	0.002455
947	-0.00074
948	-0.00853
949	-0.00999
950	0.000835
951	-0.00321
952	0.00228
953	-0.00666
954	0.011335
955	-0.02369
956	0.006632
957	-0.00761
958	0.023418
959	-0.00204
960	-0.00142
961	0.000623
962	-0.01541
963	-0.0062
964	-0.0071
965	-0.00152
966	-0.01389
967	-0.0101
968	0.010765
969	0.008859
970	-0.00175
971	-0.01026
972	-0.01934
973	-0.00302
974	0.037504
975	-0.01455
976	-0.01679
977	0.01855
978	0.021144
979	-0.00566
980	0.006562
981	0.007612
982	-0.01329
983	-0.01099
984	0.005686
985	0.010184
986	-0.01209

987	-0.00855
988	-0.00641
989	-0.00707
990	0.006631
991	-0.0248
992	9.61E-06
993	0.00885
994	-0.01976
995	-0.00157
996	0.003124
997	-0.01663
998	0.006571
999	-0.01049
1000	-0.00227
1001	0.028691
1002	0.000936
1003	-0.01654
1004	-0.01343
1005	-0.01704
1006	0.003619
1007	-0.0167
1008	-0.00267
1009	0.016733
1010	-0.02139
1011	-0.02123
1012	0.006223
1013	0.03673
1014	-0.01218
1015	-0.02472
1016	-0.03396
1017	0.007496
1018	-0.00645
1019	-0.00376
1020	-0.01851
1021	0.005661
1022	-0.02702
1023	-0.03835
1024	-0.03291
1025	-0.02702
1026	0.057327
1027	-0.00563
1028	0.016884
1029	0.054078
1030	0.004249
1031	0.009792
1032	-0.02957
1033	-0.02308
1034	-0.0343
1035	0.029919
1036	0.02001
1037	0.032722
1038	0.003512
1039	-0.00533
1040	-0.02168
1041	0.040047
1042	0.011559
1043	-0.00159
1044	0.023612
1045	-0.01396
1046	0.012726
1047	0.014052
1048	-0.02269
1049	0.007536
1050	-0.01385
1051	-0.01813
1052	-7.63E-05
1053	-0.00188
1054	-0.04154
1055	0.017517
1056	-0.01595
1057	0.0168
1058	0.010113
1059	0.007331
1060	-0.00014
1061	-0.02478
1062	0.00327
1063	0.00145
1064	-0.01973
1065	-0.00465
1066	-0.03006
1067	0.002455
1068	-0.01383
1069	-0.01728
1070	0.024863
1071	0.01821
1072	-0.03226
1073	-0.01461
1074	0.040023
1075	-0.02359
1076	-0.01082

1077	-0.02243
1078	-0.01911
1079	0.016898
1080	-0.02729
1081	0.034966
1082	0.039059
1083	0.007327
1084	0.047336
1085	-0.02411
1086	0.022302
1087	0.005903
1088	0.017334
1089	-0.01063
1090	0.006718
1091	-0.01522
1092	0.017167
1093	-0.00827
1094	-0.00908
1095	0.009704
1096	-0.00556
1097	0.01716
1098	0.008202
1099	0.00775
1100	0.009144
1101	-0.02285
1102	-0.00876
1103	-0.02073
1104	0.007715
1105	-0.00048
1106	0.024634
1107	0.006149
1108	-0.01041
1109	-0.00402
1110	0.019415
1111	0.021452
1112	-0.00344
1113	0.002493
1114	-0.02097
1115	0.027986
1116	-0.00273
1117	-0.0019
1118	-0.01475
1119	-0.01542
1120	0.006266
1121	-0.02218

1122	0.013957
1123	0.000564
1124	-0.00373
1125	-0.01342
1126	0.023519
1127	-0.00814
1128	-0.01315
1129	-0.00771
1130	0.013017
1131	0.001809
1132	-0.00547
1133	-0.00315
1134	-0.01603
1135	0.004558
1136	0.000489
1137	0.0332
1138	-0.00048
1139	0.022474
1140	-0.00654
1141	-0.01409
1142	0.019386
1143	0
1144	-0.00141
1145	0.00583
1146	-0.01443
1147	-0.00394
1148	-0.01402
1149	-0.0157
1150	-0.01043
1151	0.010224
1152	-0.02923
1153	-0.01616
1154	0.01305
1155	0.006779
1156	-0.02285
1157	0.01313
1158	0.005399
1159	-0.01409
1160	-0.00544
1161	-0.00645
1162	-0.01009
1163	0.007569
1164	-0.0081
1165	-0.01269
1166	-0.0016

1167	0.021435
1168	0.0195
1169	-0.0071
1170	-0.0095
1171	0.013224
1172	-0.01838
1173	0.007195
1174	-0.01314
1175	0.011758
1176	0.004622
1177	-0.00754
1178	-0.01536
1179	0.009562
1180	-0.00934
1181	0.008259
1182	-0.02583
1183	-0.00836
1184	0.004321
1185	0.034457
1186	0.001647
1187	0.035427
1188	0.004242
1189	0.008737
1190	0.001888
1191	0.022977
1192	-0.03523
1193	0.012161
1194	-0.00548
1195	-0.00164
1196	-0.00578
1197	-0.01774
1198	0.012144
1199	0.026116
1200	-0.00505